RESEARCH ON SOFTWARE USER BEHAVIOR CREDIBILITY ANALYSIS MODEL BASED ON MULTI-STRATEGY LEARNING ALGORITHM

Xuejun YU[1], Yang LIU[2,*]

It is necessary to evaluate the credibility of software user behavior for the timely identification and control of abnormal user behavior risks in order to ensure the security of the software system and the credibility of user behavior. The user behavior log dataset is generated by using the simulated county and city government office system, and the credibility analysis algorithm model of numerous individual learning and ensemble learning are constructed to evaluate the effect of the credibility analysis model in the study, and by comparing and analyzing the heterogeneous ensemble learning algorithm models of different combinations, the accuracy of software user behavior credibility analysis has been improved to more than 97%. This study shows that the model is able to identify untrusted users in a flexible yet accurate manner and guarantees the security of the fixed domain work system.

Keywords: software credibility; feature extraction; user behavior; ensemble learning

1. Introduction

Although software technology brings convenience to our lives, it also creates numerous security issues for the software systems, such as virus attacks, network attacks, and information leakage. With the continuous development of software systems, a number of varied-level software systems are faced with issues relating to the misuse of user identity, which may result in security and reliability concerns if left unattended [1]. Hence, merely relying on the current user system login authentication and software role-based permissions system to control access of malicious users to the software system is insufficient to address the security concerns of software system and ensure the credibility of its users. Users are both users and maintainers of software systems, as well as threats and destroyers of software systems. As such, the credibility of user behavior is in direct relation to the security of the software system. It is therefore crucial to assess the credibility of a user's behavior on a software system.

[1] Associate Prof., Faculty of Information Technology, Beijing University of Technology, China, e-mail: yuxuejun@bjut.edu.cn

[2]* Master of Software Eng., Faculty of Information Technology, Beijing University of Technology, China, corresponding author, e-mail: liuyang6617@163.com

At present, in similar research field, Li Haibin et al. [2] proposed an unsupervised method for detecting user behavior anomalies in database which focuses mainly on the analysis of user behavior at the database level whereby the unsupervised kernel density estimation algorithm is used; Cheng Luxiao et al. [3] proposed a Bayesian network model for authentication level prediction based on user behavior attributes which is a type of authentication method for remote sensing cloud user behavior based on the Bayesian network to identify the invasion of untrusted users in remote sensing cloud service platform; Ussath et al. [4] proposed using neural networks to identify suspicious user behavior based on suspicious user data in the network; Chiu et al. [5] proposed a frequent pattern-based user behavior anomaly detection in cloud systems which is a framework to utilize anomaly detection and random re-sampling techniques for profiling user's behaviors via the frequent patterns of activated system processes.

Through the above research and analysis of related papers, it is found that most of the relevant research focus on database, cloud security, network security, cloud systems and other security aspects of user behavior research. Moreover, the research field of software user behavior is relatively broad, and there is no specific research that focuses on the characteristics of the software system. Software systems in different fields have different factors that affect the credibility of user behavior [6]. Therefore, this paper limits the research content to the field of high-sensitivity and high-security office work software systems, and it is more valuable to study its user behavior data. In addition, the standard signature-based or abnormal-based user behavior detection methods used in these papers are relatively simple, fixed, and inflexible, and cannot fully evaluate the credibility of user behavior. In this paper, the experimental method uses an ensemble learning algorithm model of a variety of different combination strategies, that is, a multi-strategy learning algorithm [7], which analyzes user behavior data more flexibly and adaptively.

2. Research Scheme

In the field of office software system, through the two dimensions of data engineering and algorithm models, the influencing factors of software user behavior credibility are studied, and a suitable software user behavior credibility analysis model is evaluated.

2.1 Definition of credibility

In view of the fact that most real-life events follow the principle of normal distribution, it is only natural that the software user behavior logs generated by the simulated system in this study adopts this mode of distribution. Assuming that most of the data of normal user behavior is subject to a certain unknown distribution X , and the function $y = f(x)$ is used to represent its unknown

probability density function, the collected historical dataset D can be considered to be composed of a certain number of samples independently extracted from distribution X [8].

This study defines the "credibility" of software user behavior as follows: most of its historical behavior trajectories are distributed within a certain confidence interval $[a,b]$ in the historical behavior log of software users, and its confidence level $Y = y\%$ is the credibility that the software user behavior belongs to a certain category.

Considering the fact that the behavior and habits of software users differ from person to person, it is therefore not advisable to define user behavior generally as a fixed distribution X in the process of generating user behavior log. In order to reflect the differences between users, in the case of a certain Y, according to formula (1), the mean value M and the standard deviation sd are randomly adjusted to obtain the confidence interval of the software user behavior sample. This is to ensure that the majority of user behavior in its sample does not deviate from the confidence interval $[a,b]$. In this study, $Y \approx 95\%$, and M adjusts its change interval according to the difference in the characteristics of software user behavior. In this way, the generalization space for software user behavior samples, as well as the space for differences between samples are being reserved.

$$Y = \begin{cases} y(a = M - 1\times sd, b = M + 1\times sd)\times 100\% \approx 68.3\% \\ y(a = M - 2\times sd, b = M + 2\times sd)\times 100\% \approx 95.5\% \\ y(a = M - 3\times sd, b = M + 3\times sd)\times 100\% \approx 99.7\% \end{cases} \quad (1)$$

2.2 Research process of credibility analysis model

This study adopts an approach based on the data engineering and algorithm model dimensions. In view of the numerous factors that influence the software user behavior log dataset, it is necessary to quantify it after it is being generated by the simulated system. Firstly, through the use of software user behavior log feature statistics module embedded in the simulated system, a quantized dataset is obtained. Secondly, it is necessary to standardize the quantized dataset, and also to apply the Principal Components Analysis (PCA) [9] and the dimension-reduction [10] techniques to it. Thirdly, experiments are conducted on individual learning and homogeneous ensemble learning algorithm models respectively, and subsequently a better algorithm model is obtained through a comparative analysis of the evaluation indexes. Finally, building on the previous step, the different individual learning and homogeneous ensemble learning algorithm models are integrated, and the effects of the different heterogeneous ensemble learning models are evaluated through various tests.

The flow chart of the research process in this paper is as shown in Fig. 1:

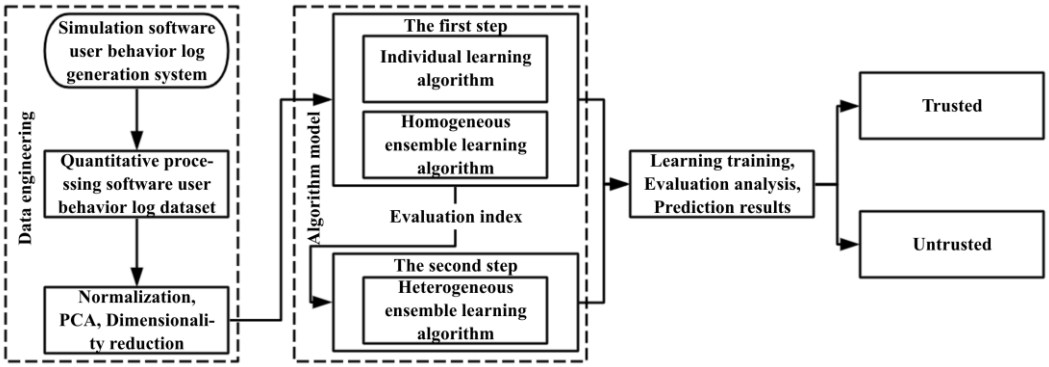

Fig. 1. Flow chart of the research process of software user behavior credibility analysis

3. Data Engineering

By analyzing the characteristics of the county and city government office systems, simulations are performed to generate user behavior log datasets that can extract key features, quantify them, and provide data support for the study of credibility analysis models.

3.1 Generation process of software user behavior log dataset

The study found that there are few open source datasets in related research areas, and the sample distribution in these datasets is uneven. The features of these datasets are difficult to extract, so it is not suitable for the study of user behavior in the field of office software systems. This paper constructs a set of dataset generation system of Java code simulation county and city government office software, which can generate logs in line with the user behavior habits in the real system by configuring the distribution of samples. Fig. 2 shows a user behavior log information generated according to the template of the standard HTTP request header, in which the “Data-Time” and “Ip-Address” fields are used for auxiliary log information recording.

```
GET /document_management/meeting_archive.html?action=statistics&record_number=655815 HTTP/1.1
Host: www.office.gov.cn
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_6_8) AppleWebKit/537.13+ (KHTML, like Gecko) Version/5.1.7 Safari/534.57.2
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: pt-PT,pt;q=0.5
Accept-Encoding: deflate
Referer: http://www.office.gov.cn/document_management/meeting_archive.html?action=statistics&record_number=655815
Connection: keep-alive
If-None-Match: "v9"
Cache-Control: max-age=0

Date-Time: 2019-04-16 14:25:00
Ip-Address: 107.91.205.242
```

Fig. 2. A software user behavior log generated by simulation.

In order to configure the information in the standard HTTP request header template, that is, the parameters in the log generation template, we need to determine the configuration item information required in the template in advance in the data generation system, mainly including the URL function module path configuration, user information configuration, browser configuration, IP address configuration, date configuration, special character configuration, etc. Among them, about the URL function module path, it is defined by studying the design scheme of the county and city government office system [11]. The process of data generation is by adjusting the range of generating intervals of various configuration item information, that is, confidence interval $[a,b]$, determine the mean value of interval M , and then calculate the standard deviation sd in reverse according to the second equation (Y≈95.5%) in formula (1). Thus, the gaussian function of various configuration item information for generating user behavior log is determined according to the M and sd parameters, and the positive or negative sample log for each user is generated by combining the random function. Therefore, the simulation system can generate log data that accord with normal distribution and software user behavior habits under different conditions, which makes the simulation process closer to the real scene, and each log data generated has at least 95% confidence that belongs to the user's behavior habits. The software user behavior log simulation generation process is shown in Fig. 3.

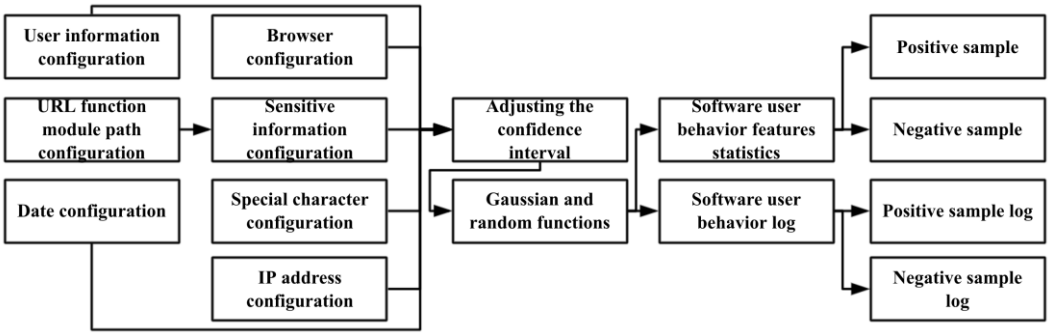

Fig. 3. Flow chart of software user behavior log simulation generation.

3.2 Quantitative process of software user behavior log dataset

3.2.1 Statistics of features

Feature statistics mainly include common features and system features. Among them, system features are features that are related to the credibility of software user behavior by analyzing the characteristics of functional modules of county and city government office systems. And through the correlation coefficient function [12] to calculate the similarity between each feature, the 29 features in Table 1 with significant influence on the study of the credibility analysis model of software user behavior are defined. Feature statistics is

completed using the software user behavior log feature statistics module embedded in the simulated system, and through this process to filter the generated software user behavior logs, identify key feature characters, and perform statistical calculations on the number of features in each dimension of each user. The statistical results of all features are of numerical type. Finally, the results of feature statistics are saved as an Excel file.

Table 1

Feature statistics table of software user behavior log dataset

No.	Category	Feature description
1	Different type	Number of different IP address types
2		Number of different browser types
3		Number of different operating system types
4	Non-working hour	Number of visits times during non-working hours
5		Number of different operating system types during non-working hours
6		Number of different IP address types during non-working hours
7		Number of different browser types during non-working hours
8		Number of login times during non-working hours
9		Number of logout times during non-working hours
10	Login and logout	Number of login times
11		Number of logout times
12	Password	Number of password errors
13		Number of password changes
14	Illegal	Number of illegal requests
15		Number of illegal redirects
16	File	Total number of file downloads
17		Total number of file deletions
18		Total number of file modifications
19		Total number of file queries
20	Sensitive file	Total number of downloads of sensitive files
21		Total number of deletions of sensitive files
22		Total number of modifications of sensitive files
23		Total number of queries of sensitive files
24	Special character	Total number of times containing special characters
25		Number of times containing special characters in the addition
26		Number of times containing special characters in the modification
27		Number of times containing special characters in the query
28	Unrecognized	Number of unrecognized browsers
29	Action	Number of actions

3.2.2 Standardization of features

It is evident from the statistical features of the software user behavior log that the range of values of features in the different dimensions varies greatly, which will to a certain extent affect the credibility analysis results of most algorithm models.

As for the features of the 29 dimensions in Table 1, the standard deviation is standardized in all the algorithm models except the tree-based model, since it does not require normalization, as its concern is not the value of the variables, but

rather the distribution of the variables and the conditional probability between the variables. The idea behind standardization of the standard deviation is to achieve mean removal and variance normalization of the features of the 29 dimensions, so that the processed data conforms to the standard normal distribution, that is, the mean is 0 and the standard deviation is 1. By definition, the statistical result of the j-th feature of the i-th user is $R_{i,j}$ where $1 \le i \le 5250$ and $1 \le j \le 29$. The mean value of the k-th dimension in the dataset is $M_k = mean(R_{i,k})$, and the standard deviation of the k-th dimension is $S_k = std(R_{i,k})$. According to formula (2), the standard deviation of the j-th feature of the i-th user is standardized as $R^*_{i,j}$.

$$R^*_{i,j} = \frac{R_{i,j} - M_j}{S_j} \tag{2}$$

3.2.3 Dimension reduction of features

In order to reduce the dimensionality of the dataset, remove the correlation between the features and highlight their differences, this study uses PCA to map the features of the 29 dimensions to 13 dimensions using the feature standardization method. While PCA dimensionality reduction works well on some algorithm models. PCA is an effective method for reducing the number of dimensions and denoising data based on the variance-covariance matrix, which is essentially K-L transform [13].

Through experiments, the ratio of the difference in information between samples represented by the new feature to the total difference in information as shown in Table 2. The results show that proportion of principal components in the first 13 dimensions is 98.02%. Generally speaking, most of the information in the data is concentrated on the first few principal components.

Table 2

Analysis table of variance proportion of features

Variance proportion of the first 13 new features									Sum
0.4396	0.1764	0.0854	0.0816	0.0446	0.0348	0.0309	0.021	0.0195	0.9802
0.018	0.0173	0.008	0.0031	-	-	-	-	-	

4. Experiments of Algorithm Models

Based on the software user behavior log dataset processed by data engineering in the previous chapter, the performance indexes of individual learning, homogeneous ensemble learning and heterogeneous ensemble learning algorithm models are compared and analyzed, and a better software user behavior credibility analysis model in this research field is obtained.

4.1 Description of experimental environment

(1) Data: the experimental data was processed through quantitative statistics, standardization, PCA, and dimensionality reduction to obtain a dataset containing 5,250 software user behavior samples, of which the ratio of positive and negative samples is 8:13.

(2) Algorithm model: a variety of algorithm models have been implemented through the python 3.7 and scikit-learn 0.21.3 development kits. Among them, the individual learning algorithms include: support vector machine (SVM), decision tree (DT), logistic regression (Logistic), neural network (BP) and K nearest neighbor (KNN) algorithms [14], and the homogeneous ensemble learning algorithms include: Random Forest (RF) and XGBoost (XGBT) [15]. The heterogeneous ensemble learning algorithms are implemented by combining different individual learning and homogeneous ensemble learning algorithms through soft voting strategies [16]. All the algorithm models applied default model parameters to the dataset in order to eliminate the differences between the models.

(3) Training process: the same training set and test set was used in a 3:1 random split for the different algorithm models, and after calculating the mean, the P-R curve evaluation index of the comparison model was obtained, after which the complete dataset was again used for 10 rounds of 10-fold cross-validation, to calculate the average, and evaluate the performance indicators as well as the accuracy of the model.

4.2 Comparison and analysis of credibility analysis models

4.3.1 Evaluation index

The evaluation index is used to analyze the evaluation results of the different models in the same dataset. As the focus of this study is on algorithm models, and in order to reflect the differences between samples of untrusted user behavior as well as to highlight certain key features, an uneven ratio for the positive and negative samples is adopted. Therefore, in the selection of evaluation indexes, an analysis on the accuracy (formula (3)), precision (formula (4)), recall (formula (5)), comprehensive evaluation rate (F1), and Precision-Recall curve (P_R curve) is conducted.

$$Accuracy = \frac{TT + TU}{TT + FT + FU + TU} \tag{3}$$

$$\Pr ecision = \frac{TT}{TT + FT} \tag{4}$$

$$\mathrm{Re}\, call = \frac{TT}{TT + FU} \tag{5}$$

$$F_1 = \frac{2 \times \Pr ecision \times \mathrm{Re}\, call}{\Pr ecision + \mathrm{Re}\, call} \tag{6}$$

The confusion matrix for the performance index that defines the credibility analysis model is shown in Table 3 below:

Table 3

Performance index confusion matrix of credibility analysis model

	Predictive value 1 (Trusted)	Predictive value 0 (Untrusted)
Actual value 1	TT (True Trusted)	FU (False Untrusted)
Actual value 0	FT (False Trusted)	TU (True Untrusted)

Among them, the comprehensive evaluation rate is defined based on the harmonic mean of precision and recall. In view of the inverse relationship between precision and recall, any increase in the precision may result in a decrease in the recall, and vice versa. There is therefore a need to find a balance between the two. The special method for its calculation is shown in formula (6).

The P-R curve can intuitively show the precision and recall of the algorithm model on the overall sample. In many cases, we can sort the samples according to the prediction outcome of the algorithm model. The samples that are considered to be "most likely" positive examples by the algorithm model are ranked first, while those that are considered "least likely" to be positive examples are ranked last. By fixing different thresholds in this order, a prediction was performed on each of the positive samples, so that it is possible to calculate the precision and recall of the current samples every time. The P-R curve is a plot of the precision as the vertical axis and the recall as the horizontal axis. According to the size of the curve, the area enclosed by the curve and the two axes, as well as the comprehensive evaluation rate, the advantages and disadvantages of the algorithm model can be evaluated more intuitively. Obviously, the closer the P-R curve is to the outside, the larger the area is, and the better the effect of the algorithm model.

4.3.2 Experimental comparison *and result analysis*

The first step: the analysis results shown in Table 4 are obtained by comparing SVM, DT, Logistic, BP, KNN in the individual learning algorithms with RF and XGBT in homogeneous ensemble learning algorithms.

Table 4

Performance index comparison evaluation table between algorithm models

Algorithm Model	Accuracy	Precision	Recall	Comprehensive Evaluation Rate
SVM	0.956	0.961	0.927	0.943
DT	0.895	0.877	0.884	0.880
Logistic	0.872	0.827	0.839	0.832
BP	0.958	0.957	0.953	0.954
KNN	0.937	0.907	0.931	0.918
RF	0.953	0.981	0.893	0.934
XGBT	0.951	0.950	0.921	0.935

It is evident from the table that in the credibility analysis of software user behavior, the BP algorithm can produce a better comprehensive evaluation rate, as compared to the homogeneous ensemble learning algorithm. Its accuracy, precision, recall is relatively balanced, and the index value is relatively high.

Conversely, Logistic comprehensive evaluation rate is the lowest, and its index values are the lowest of all algorithm models. By comparing and analyzing the P-R curve in Fig. 4, it is evident that there is relatively little difference in the area covered by the BP, XGBT, SVM and RF algorithm models, all reached 97.5%, with an overall prediction accuracy of more than 95% for all the models. Among them, the size of the area reflects the quality of the prediction results of the algorithm model. It can be seen that the area of the area of the Logistic and DT algorithm models is relatively small, and the "most likely" prediction is that there are fewer positive examples.

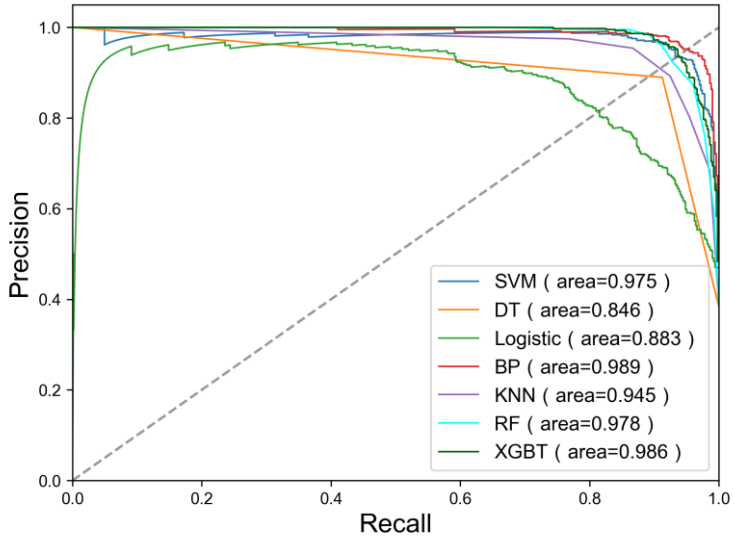

Fig. 4. P-R curve comparative analysis chart.

The second step: as evident in the first step, the individual learning algorithm models BP and SVM, and the homogeneous ensemble learning algorithm model XGBT and RF yield good evaluation results, with a relatively high comprehensive evaluation rate as compared to other algorithm models. However, in view of the fact that the BP and XGBT models are relatively stable and have the best prediction credibility in terms of balancing the P-R curve, precision, and recall, a comparison of the heterogeneous ensemble learning algorithms consisting of different bases and an analysis on the prediction accuracy of software user behavior credibility of the combined model are conducted based on the results in step one. By comparing and analyzing the accuracy and comprehensive evaluation rate of the heterogeneous ensemble learning algorithm

model BP + XGBT, BP + XGBT + SVM, BP + XGBT + RF, BP + XGBT + SVM + RF, the effect shown in Fig. 5 is obtained.

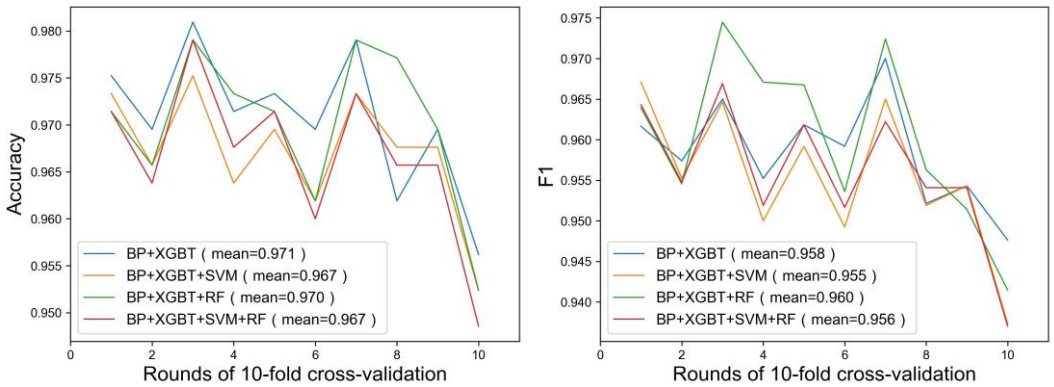

Fig. 5. Accuracy and comprehensive evaluation rate (F1) comparative analysis chart.

It is evident from the indexes of the two dimensions, that, overall, the heterogeneous ensemble learning algorithm is better than the individual learning algorithm and the homogeneous ensemble learning algorithm with an average accuracy of more than 97%, and an increased average comprehensive evaluation rate of more than 96%. Overall, the BP + XGBT + RF combination in the heterogeneous ensemble learning algorithm model has the best effect, resulting in improved accuracy of the software user behavior credibility analysis.

5. Conclusion

In this study, a dataset that can satisfy the credibility analysis of software user behavior in related fields is generated through a set of software that simulates the county and city government office systems. Quantitative statistics, standardization, principal component analysis, and dimensionality reduction processing are performed on the dataset through feature engineering. Subsequently based on the evaluation indexes of the individual learning algorithm model, homogeneous ensemble learning algorithm model, and heterogeneous ensemble learning algorithm models that are analyzed on this dataset, it is determined that the heterogeneous ensemble learning algorithm model produces greater accuracy and a better comprehensive evaluation rate. This experiment proves that the BP + XGBT + RF combination in the heterogeneous ensemble learning algorithm model produces the best result, improving the accuracy of software user behavior credibility analysis to more than 97%.

This shows that it is feasible to analyze the user behavior credibility on the existing log dataset of user behavior in the software system, and thus able to conduct an effective evaluation on some illegal and abnormal behavior of

legitimate users in the software system under certain circumstances, thereby ensuring the security of the software system in a more flexible manner. It is inevitable that there are shortcomings in this study. For example, there is no study done to include the server log, whereby more feature dimensions of the software user behavior data are extracted, which would then be more productive for a comprehensive study of the credibility of software user behavior. These shortcomings will serve as areas for improvement and future research.

REFERENCES

[1]. *Liu Jian, Su Purui, Yang Min, et al*. "Software and Cyber Security—A Survey", Journal of Software, **vol. 29**, no. 1, 2018, pp. 42-68.

[2]. *Li Haibin, Li Qi, Tang Ruming, et al*. "User Behavior Anomaly Detection for Database Based on Unsupervised Learning", Journal of Chinese Computer Systems, **vol. 39**, no. 11, 2018, pp. 2464-2472.

[3]. *Cheng Luxiao, Yan Jinning, et al*. "Bayesian network method for remote sensing cloud user behavior authentication", Application Research of Computers, no. 02, 2019, pp. 1-8.

[4]. *Ussath M, Jaeger D, Cheng F, et al*. "Identifying suspicious user behavior with neural networks//2017 IEEE 4th International Conference on Cyber Security and Cloud Computing (CSCloud)", IEEE, 2017, pp. 255-263.

[5]. *Chiu C Y, Yeh C T, Lee Y J*. "Frequent Pattern Based User Behavior Anomaly Detection for Cloud System// Proceedings of the 2013 Conference on Technologies and Applications of Artificial Intelligence", IEEE Computer Society, 2013.

[6]. *Wang H, Ding B*. "Growing construction and adaptive evolution of complex software systems", Science China, **vol. 59**, no. 05, 2016, pp. 5-7.

[7]. *Gançarski P, Wemmert C*. "Collaborative multi-step mono-level multi-strategy classification", Multimedia Tools and Applications, **vol. 35**, no. 1, 2007, pp. 1-27.

[8]. *Ferguson T S*. "A course in large sample theory", Routledge, 2017.

[9]. *Groth D, Hartmann S, Klie S, et al*. "Principal components analysis//Computational Toxicology", Humana Press, Totowa, NJ, 2013, pp. 527-547.

[10]. *Oymak S, Tropp J A*. "Universality laws for randomized dimension reduction, with applications", Information and Inference: A Journal of the IMA, **vol. 7**, no. 3, 2018, pp. 337-446.

[11]. *Shen H, Su H Q*. "Service-oriented E-government General Office Automation System Design and Function to Achieve", Journal of Guangxi Academy of Sciences, **vol. 26**, no. 4, 2010, pp. 503-506.

[12]. *Lee Rodgers J, Nicewander W A*. "Thirteen ways to look at the correlation coefficient", The American Statistician, **vol. 42**, no. 1, 1988, pp. 59-66.

[13]. *Huang Xiaobin, Wan Jianwei, Wang Zhan*. "An Improved K-L Transform for Feature Extraction", Journal of national university of defense technology, no. 01, 2005, pp. 84-88.

[14]. *Osisanwo F Y, et al*. "Supervised machine learning algorithms: classification and comparison", International Journal of Computer Trends and Technology (IJCTT), **vol. 48**, no. 3, 2017, pp. 128-138.

[15]. *Petrakova A, Affenzeller M, et al*. "Heterogeneous versus homogeneous machine learning ensembles", Information Technology and Management Science, **vol. 18**, no. 1, 2015, pp. 135-140.

[16]. *Cao J, Kwong S, Wang R, et al*. "Class-specific soft voting based multiple extreme learning machines ensemble", Neurocomputing, **vol. 149**, 2015, pp. 275-284.