A Feature Extraction Method for Credit Card Fraud Detection

Yu Xie, Guanjun Liu, Ruihao Cao, Zhenchuan Li, Chungang Yan, and Changjun Jiang
Department of Computer Science
Tongji University
Shanghai, China
e-mail: 740222684@qq.com, liuguanjun@tongji.edu.cn, leonruihao@163.com, 1510482@tongji.edu.cn,
yanchungang@tongji.edu.cn, cgyan2@163.com

***Abstract*—As credit card fraud has caused huge economic losses and harm cardholders seriously, credit card fraud detection is important and has been paid much attention. An effective feature engineering is the key to build an effective model of detecting fraud. However, we find that the current feature engineering that is based on the frequency of transactions is not perfect. Although frequency-based feature engineering depicts the temporal features of user transactions, it does not enough consider the fraud characteristics and the distinction of transaction behaviors. Starting from the two points, we propose a rule-based feature engineering that considers both individual behavior and group behavior, and portrays the individual behavior as group features, and thus can more effectively distinguish legitimate and fraudulent transactions. Our experiments illustrate the advantages of our method.**

***Index Terms*—credit card fraud detection, rule-based feature engineering, transaction behavior**

I. Introduction

Nowadays, more and more people use credit cards for offline or online transactions. Internet technology and intelligent mobile devices make these transactions simple, convenient and comfortable. However, credit card fraud events often take place since internet environment is open and criminals can utilize some techniques to carry out frauds, such as eavesdropping, phishing, intrusion, denial-of-service, database stealing and man-in-the-middle attack [1]. Credit card fraud has become increasingly rampant and caused huge economic losses worldwide [2]. Therefore, how to accurately and efficiently detect credit card fraud has become a research hotspot. An important way of detecting fraud is to use the advanced data mining technology to analyze the historical transaction behavior deeply, find out the hidden knowledge and rules, and then develop a model that can evaluate the risk degree of each transaction.

The use of machine learning for fraud detection has been an inevitable trend in recent years. When constructing a fraud detection model, there are several important factors during the training phase: Concept drift, class imbalance, transaction behavior, verification latency, cost-sensitivity of the application, transaction aggregation, and preprocessing of the features [3]–[11]. In this paper, we mainly focus on the transaction behavior and the features preprocessing.

Hidden Markov chains [12] and self-organizational networks [13] are often used as the models of individual behaviors. When a new transaction of a user does not match the individual behavior model of the user, the transaction is thought of being conducted by a defrauder. However, individual behavior methods usually have two flaws: 1) it is difficult for many users to create an accurate individual behavior model since their transaction data are relatively less; 2) an individual behavior model cannot characterize some new transaction behaviors of a user.

Classification methods, e.g. random forest [3], [14]–[16] and neural network [17]–[19], are often used to train a model of group behaviors. Generally, all given legal transactions of all users form a group and all given fraud transactions form another group. Then, a classification method can be used over them to train a classifier. If a new transaction (from any user) is classified into the fraud group by the classifier, it is thought of as a fraud; and otherwise it is legal. These methods can partly make up for the flaws of individual behavior models due to the data of similar users, and thus they become popular [20]. However, they also have some flaws. One is the class imbalance problem [4], [21] since fraud transactions are much less compared to legal ones. It can seriously influence the performance of classification. Another one is that it will spend too much time to train a classifier if all legal transactions take part in the training (since the amount of data is very huge). An alternative way is to select a part of data, but this easily leads to the case that some behaviors cannot be characterized in the classifier since the related data are not selected.

Our method belongs to the latter, but we consider some individual behaviors into it.

For any methods, feature engineering is necessary. Some studies use raw data directly. However, the features in raw data are often similar. This is because the data collected during a credit card transaction must comply with international financial reporting standards [5]. But the information that these raw features can describe is very limited. To deal with this, some studies use feature engineering techniques [3], [5]–[8], [22]. Generally, these feature engineering techniques are based on the time series of transactions. For example, the computation of the created features consists in grouping the transactions made during the last given number of hours, first by card number, then by transaction type, merchant group or other, followed by calculating the number of transactions or the total amount spent on those transactions [5], [6]. This method can describe the transactions' timing characteristics. However, this is not enough. On the one hand, although the features based on frequency can enrich information to some extent, the created feature cannot fully depict a cardholder's historical transaction (i.e., individual behavior). On the other hand, the created features cannot reflect the relationship between legitimate transactions and fraud. For example, in the fraud case, it is not uncommon for fraudsters to copy a user's credit card at a distance after obtaining the user's bank card information and then transfer the money into their account. This behavior cannot be characterized by time-frequency. In addition, fraudsters first make some small transactions and then make a big one. This behavior cannot represented by the frequency-based feature engineering. Generally, the lack of association between legitimate transactions and fraud leads to the deficiency of the excavation of fraudulent behavior.

In this paper, we propose a set of rules over which a set of new features of transaction behaviours are created and we encapsulate these rules as Behavior Space. Based on each rule, we can output a value of True or False for each transaction record, and such a value is extended to the transaction record as a new element. Based on these extended transaction records (including legal and fraud ones), we use random forest to train a set of classifiers by which every new transaction can be checked if it is fraud or not. Our experiments show that our rule-based feature creation method outperforms the frequency-based one.

The rest of this paper is organized as follows: Section II introduces the diverse features and the rules we built. Section III presents the related experiment. Section IV concludes this paper.

II. Behavior Space

This section, we first introduce the basic attributes of transaction data and the frequency-based feature creation. Next, we describe our rules and the created features based on them.

A. Frequency-based feature creation

Table I lists the raw features of credit card transactions. Some fraud detection methods use only the raw features [23]–[25]. However, they lose a lot of important information such as the user's behavior habits. To deal with this, methods in [5]–[7], [22] create some new features based on some raw features. These new features utilize the time-series of transactions. In Table II, the frequency-based features are summarized. According to the time and the merchant of the transaction, the frequency analysis is carried out from the sequential logic, and then the feature of the raw data is expanded.

These approaches have some flaws. Generally, two or three different attributes are combined to construct a new feature. If there are time-related attributes, researchers often draw different time windows, get the eigenvalues of the same attribute at different times, or decompose/slice a feature. For example, the $Trade\_time$ in the data are constructed according to the quarter and cycle. However, this method cannot reflect the relationship between legitimate transactions and fraud. To deal with this, we propose a rule-based method to generate features. In Table III, the rule-based features are summarized. Next, we introduce these rule-based features.

TABLE I Primary Attributes

Attributes name	**Description**
Common_phone	Customer's usual mobile phone number
Pay_bind_phone	Customer's number bound on the electronic payment platform
Pre_trade_result	Customer's verification results of the last transaction
Is_common_ip	Whether this transaction is a common IP
Trade_amount	Amount of a transaction
Pay_single_limit	Limit on the amount of a single transaction
Pay_accumulate_limit	Total daily transaction amount limit
Account_number	Credit card number
Client_mac	MAC address of a transaction
Trade_date	Date of transaction
Trade_time	Exact time of transaction
White_list_mark	Whether the account is in the trusted list
Card_balance	Account balance before payment
Transaction_object	Is the receiver a person or a business
Receiver_number	Receiver number of account's last transaction
Last_trade_time	Account's last transaction time

TABLE II
Derived Attributes Based On Frequency

Attributes name	**Description**
Amount_over_month	Average amount spent per transaction over a month
Average daily over month	Average amount spent per day over the past 30 days
Average over 2 months	Average amount spent over the course of 1 week during past 2 months.
Amount Transaction_object over month	Average amount per day over a 30 day period on all transactions up to this one on the same transaction_object as this transaction
Number Transaction_object over month	Total number of transactions with same transaction_object during 30 days
Amount Transaction_object over 2 months	Average amount spent over the course of 1 week during past 2 months on the same transaction_object as this transaction
Amount same day	Total amount spent on the same day up to this transaction
Number same day	Total number of transactions on the same day up to this transaction
Number client_mac over month	Total number of transactions with same client_mac_ during 30 days

B. Rule-based feature creation

Rule 1: *Consistency feature matching rule*. For a lot of fraudulent transactions, fraudsters use a new electronic payment phone number to ensure the safety of his/her common phone number and avoid being detected. Therefore, we construct a rule called consistency feature matching rule. The matching function $Match(a, b)$ is constructed to detect whether *common_phone* and *pay_bind_phone* match in a transaction record. The new feature *phone_matching* is assigned by 0 if they match, and otherwise it is 1. An example

TABLE III
DERIVED ATTRIBUTES BASED ON RULES

Attributes name	Description
Phone_matching	Whether common_phone and pay_bind_phone is match
Uncertain_validation	Bank staff are unable to give timely and accurate judgments about suspected transactions
Sensitive_single_amount	Whether the client's single transaction amount is abnormal
Sensitive_daily_total_amount	Whether the client's total daily transaction amount is abnormal
Sensitive_test_amount	Whether the account has conducted exploratory trading
Large_amount	Whether the account has made transactions of large amount in a short time
Untrusted_frequent_trade	Untrusted accounts are traded frequently
Big_one-time_deal	A large one-time transaction transfers all the balance in the account
Untrusted_time	Accounts are traded at irregular times to untrusted accounts
Untrusted_place	The locations of the two transactions in the account are too wide

TABLE IV
EXAMPLE CALCULATION OF RULE 1

PRIMARY			DERIVED
Account Number	*Common Phone*	*Pay Bind Phone*	*Phone Matching*
1	10000	10000	0
2	10000	10101	1

is shown in Table IV.

$$Match(NUM_{com}, NUM_{e-pay}) = \begin{cases} True & NUM_{com} = NUM_{e-pay} \\ False & otherwise \end{cases} \tag{1}$$

$$phone\_matching = \begin{cases} 1 & Match(NUM_{com}, NUM_{e-pay}) = False \\ 0 & otherwise \end{cases} \tag{2}$$

Rule 2: *Uncertain validation rule.* Some transactions judged to be suspicious by the detection model will be confirmed by the phone with the cardholder by the staff, some of which are confirmed to be legitimate, some of which are confirmed to be fraud, and some of which cannot be determined because no one is answering the phone. The feedback of a bank staff's return visit (*pre_trade_result*) to a suspicious transaction under interception by the system is used as a verification: $Vf = \{V_0, V_1, NULL\}$, where V_0 represents the verification result as genuine transaction, V_1 represents the verification result as fraudulent transaction and $NULL$ indicates that a suspicious transaction cannot be determined by the bank staff. These $NULL$ transactions will be considered as legitimate transactions after a period of time, which is called the validation delay [4]. This will lead to a lot of misreporting. Therefore, we combine *pre_trade_result* with is_common_ip for further verification. $is\_common\_ip \in \{True, False\}$ means whether a transaction uses a common IP. The derived feature is $uncertain\_validation$. When $Vf = NULL$ and $is\_common\_ip = False$, the following function is invoked and the return value is used as a new eigenvalue. An example is shown in Table V.

TABLE V
EXAMPLE CALCULATION OF RULE 2

PRIMARY			DERIVED
Account Number	*Vf*	*Is_Common_Ip*	*Uncertain_Validation*
1	NULL	FALSE	1
2	NULL	TRUE	0
3	V0	FALSE	0

$$uncertain\_validation = \begin{cases} 1 & Vf = NULL \wedge is\_common\_ip = False) \\ 0 & otherwise \end{cases} \tag{3}$$

Rule 3: *Sensitive amount rule.* Amount is one of the most important features. The features which are related to amount are *trade_amount*, *pay_single_limit* and *pay_accumulate_limit*. For the above attributes, fraudulent transactions usually have the following characteristics:

1) *trade_amount* of each transaction is close to the *pay_single_limit*;

2) The total *trade_amount* per day is close to the *pay_accumulate_limit*;

3) A fraudster generally makes a small trial deal before making a large transaction in order to avoid being found as a fraudster by a fraud detection system.

Therefore, three new features in raw data are created: *sensitive_single_amount*, *sensitive_daily_amount* and *sensitive_test_amount*. The daily transaction amount for a bank card is set to $\boldsymbol{A = \{a_1, a_2, a_3 \cdots a_c\}}$, where a_c represents the amount currently being detected. The calculation is described as follows and an example is shown in Table VI.

$$sensitive\_single\_amount = \begin{cases} 1 & a_c \in [A_1 - \varepsilon_1, A_1] \\ 0 & otherwise \end{cases} \tag{4}$$

$$sensitive\_daily\_amount = \begin{cases} 1 & \sum_{i=1}^{c} a_i \in [A_2 - \varepsilon_2, A_2] \\ 0 & otherwise \end{cases} \tag{5}$$

$$sensitive\_test\_amount = \begin{cases} 1 & a_{c-1} \in A_{small} \wedge a_c \in A_{large} \\ 0 & otherwise \end{cases} \tag{6}$$

Note that A_1 and A_2 represent a single transaction limit and a daily transaction limit, respectively. A_{small} is a set of small amounts, A_{large} represents large amount, and they both are defined by financial institutions. $n \in (1, c)$ is an integer. ε_1 and ε_2 are the parameters for the floating range of the single transaction amount and the daily transaction amount, respectively.

TABLE VI
EXAMPLE CALCULATION OF RULE 3

PRIMARY					DERIVED		
Account Number	*Trade Time*	*Trade Amount*	*Pay Single Limit*	*Pay Accumulate Limit*	*Sensitive Single Amount*	*Sensitive Daily Amount*	*Sensitive Test Amount*
1	01/01 18:00	1000	5000	12000	0	0	0
1	01/01 18:30	1	5000	12000	0	1	1
1	01/01 18:33	4998	5000	12000	1	1	1
1	01/01 18:37	4999	5000	12000	1	1	1
1	01/01 18:40	2000	5000	12000	0	1	1

TABLE VII
EXAMPLE CALCULATION OF RULE 4

PRIMARY				DERIVED
Account Number	*Trade Time*	*Trade Amount*	*Pay Single Limit*	*Large Amount*
1	01/01 12:00	1000	5000	0
1	01/01 12:30	4000	5000	0
1	01/01 12:32	5000	5000	1
1	01/01 12:35	5000	5000	1

TABLE VIII
EXAMPLE CALCULATION OF RULE 5

PRIMARY			DERIVED
Account Number	*Trade Amount*	*Balance*	*Big Onetime Deal*
1	1000	10000	0
2	20000	20000	1
3	14998	15000	1

TABLE IX
EXAMPLE CALCULATION OF RULE 6

PRIMARY				DERIVED
Account Number	*Trade Time*	*Trade Amount*	*Mark*	*Untrusted Frequent Trade*
2	01/01 11:00	2000	V1	1
2	01/01 12:20	3000	V1	1
2	01/01 12:30	5000	V1	1
2	01/01 12:50	5000	V1	1

TABLE X
EXAMPLE CALCULATION OF RULE 7

PRIMARY				DERIVED
Account Number	*Trade Time*	Trade Amount	*Location*	*Untrusted Place*
1	01/01 10:00	5000	Beijing	1
1	01/01 10:10	5000	Shanghai	1

Rule 4: *Frequent large amount transaction rule.* When a credit card is stolen, the fraudsters often do a lot of the large transactions in a short period of time. The rule adds a new feature $large\_amount$ to the raw data. We define a function $\boldsymbol{Gap(a,b)}$ to represent whether a big transaction is done in a short period of time. An example is shown in Table VII.

$$large\_amount = \begin{cases} 1 & Gap(a,b) \le 3min \wedge a_c \in A_{large} \\ 0 & otherwise \end{cases} \quad (7)$$

Rule 5: *Elderly rule.* Some fraud transactions specifically focus on the old person. Fraudsters will entice an old person to transfer all the balance of their accounts at once. According to the original feature $trade\_amount$, $card\_balance$, the rule adds a new feature $big\_onetime\_deal$ to raw data. An example is shown in Table VIII.

$$big\_onetime\_deal = \begin{cases} 1 & trade\_amount \to card\_balance \\ 0 & otherwise \end{cases} \quad (8)$$

Rule 6: *Non-trusted account rule*. New accounts usually have a low level of trust. Some untrusted accounts are traded frequently on the same day, and the possibility of fraud is greatly increased. In the original feature $white\_list\_mark = \{V_0, V_1\}$, V_0 indicates that the account is in the trusted list and V_1 indicates that the account is suspected. Therefore,we add a new feature $untrusted\_frequent\_trade$. Num denotes the number of transactions in one day. An example is shown in Table IX.

$$untrusted\_frequent\_trade = \begin{cases} 1 & white\_list\_mark = V_1 \wedge Num \ge 4 \\ 0 & otherwise \end{cases} \quad (9)$$

Rule 7: *Non-trusted location rule.* When a user makes a transaction with a credit card, some fraudsters use some illegal means to obtain the user's credit card information, and then make a transaction elsewhere immediately. For example, a user has just completed a transaction in Beijing, but another transaction is immediately done in Shanghai. The time interval between the two transactions is very close, but the locations are very far. According to the original feature $trade\_time$ and $client\_mac$, the rule adds new feature $untrusted\_place$ to raw data. D represents the distance between two $client\_mac$. An example is shown in Table X.

$$untrusted\_place = \begin{cases} 1 & D \ge 60km \wedge Gap(a,b) \le 1h \\ 0 & otherwise \end{cases} \quad (10)$$

Rule 8: *Non-trusted time rule.* Fraudsters usually do transactions during nonworking hours(20:00 to 6:00). According to the original feature $white\_list\_mark$ and $trade\_time$, the rule adds new feature $untrusted\_time$ to raw data. An example is shown in Table XI.

$$untrusted\_time = \begin{cases} 1 & white\_list\_mark = V_1 \wedge nonworking \\ 0 & otherwise \end{cases} \quad (11)$$

III. EXPERIMENTS

We first introduce our dataset and then compare our method with others.

TABLE XI
EXAMPLE CALCULATION OF RULE 8

PRIMARY			DERIVED
Account Number	*Trade Time*	*White List Mark*	*Untrusted Time*
1	01/01 23:30	V1	1
1	01/01 23:40	V1	1
2	01/01 22:40	V1	1
2	01/01 23:00	V1	1

TABLE XII
CONFUSION MATRIX

	Actual positive y = 1	Actual negative y = 0
Predicted positive c=1	TP	FP
Predicted negative c=0	FN	TN

A. Dataset

Our dataset comes from a financial company in China. It includes 5 million of B2C transactions from November 2016 to June 2017. All transactions contain labels. Label ”0” means that a transaction is legal (negative sample) and ”1” means fraud (positive sample). The dataset contains 64 original features, including transaction date, transaction amount, transaction location and account number, etc.

B. Behavior Space

Our experiments are conducted to compare three different feature engineering methods: The original features of the data, frequency-based feature engineering and rule-based feature engineering. As described in part II, we define the original feature as raw, and the frequency-based feature is F_1, and the rule-based feature is defined as F_2. The features are grouped by raw, raw/F_1, raw/F_2, $raw/F_1/F_2$. We divide the dataset into training sets and test sets, the former occupies 75% of all data, and the latter occupies 25%. We use Random Forest for experiments. As a binary classification, its performance measures are extracted by using a confusion matrix as shown in Table XII. From this table, several measures are extracted.

- $Precision = \frac{TP}{TP+FP}$
- $Recall = \frac{TP}{TP+FN}$
- $F_1Score = 2\frac{Precision*Recall}{Precision+Recall}$

The performance measures also include AUC [26] which represent the area under the ROC curve. The result is as shown in the Figures 1-5.

From Figure 1, raw/F_1 is slightly higher than others on precision. In Figure 2, the combination with F_2 is significantly better than the combination with F_1 on recall, and $raw/F_1/F_2$ is the best. In Figure 3, $raw/F_1/F_2$ and raw/F_2 again shows the best performance, followed by raw/F_1 and raw. On the AUC, F_2 shows the best performance. In Figure 5, we can see that F_1 brings a very small lift, but F_2 is very remarkable for performance improvement.

In a word, the rule-based feature engineering shows a better performance than the frequency-based one.

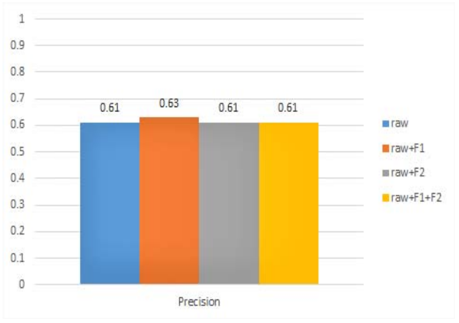

Fig. 1. *Precision* of Different Combinations.

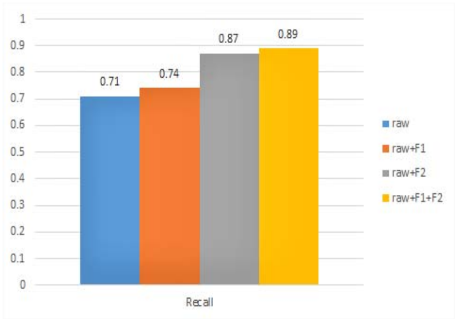

Fig. 2. *Recall* of Different Combinations.

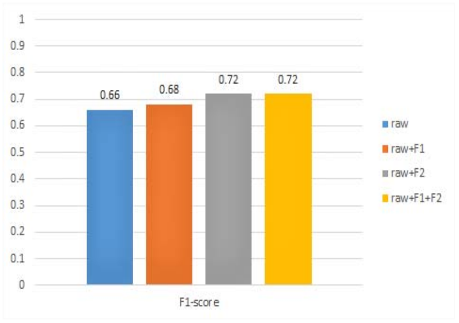

Fig. 3. F_1Score of Different Combinations.

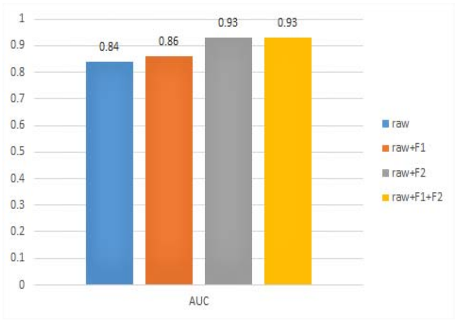

Fig. 4. AUC of Different Combinations.

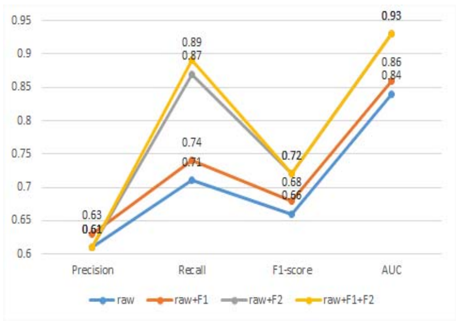

Fig. 5. Comparison of Different Combinations.

IV. Conclusion

In this paper we propose a novel approach to create some new features for credit card fraud detection. Our method portrays the individual behavior as group features, which effectively compensates for the problem of temporal features. Our experiments illustrate that this feature engineering can improve the performance of detecting fraud.

Acknowledgment

This paper is supported in part by the National Key Research and Development Program of China under Grant 2017YF-B1001804.

References

[1] Gupta S, Johari R. A New Framework for Credit Card Transactions Involving Mutual Authentication between Cardholder and Merchant, International Conference on Communication Systems and Network Technologies. IEEE, 2011:22-26.

[2] C. Mindware Research Group, Online Fraud Management Benchmark-Study 2014-2015, CyberSource, Tech. Rep., 2014. [Online]. Available:http://www.cybersource.com/

[3] Pozzolo A D, Caelen O, Borgne Y A L, et al. Learned lessons in credit card fraud detection from a practitioner perspective, Expert Systems with Applications, 2014, 41(10):4915-4928.

[4] Pozzolo A D, Boracchi G, Caelen O, et al. Credit Card Fraud Detection: A Realistic Modeling and a Novel Learning Strategy, IEEE Transactions on Neural Networks & Learning Systems, 2017, PP(99):1-14.

[5] Bahnsen A C, Aouada D, Stojanovic A. Feature engineering strategies for credit card fraud detection, Expert Systems with Applications An International Journal, 2016, 51(C):134-142.

[6] Bhattacharyya S, Jha S, Tharakunnel K, et al. Data mining for credit card fraud: A comparative study, Decision Support Systems, 2011, 50(3):602-613.

[7] Whitrow C, Hand D J, Juszczak P, et al. Transaction aggregation as a strategy for credit card fraud detection, Data Mining & Knowledge Discovery, 2009, 18(1):30-55.

[8] Kho J R D, Vea L A. Credit card fraud detection based on transaction behavior, Region 10 Conference, Tencon 2017 -. IEEE, 2017:1880-1884.

[9] Widmer G, Kubat M. Learning in the presence of concept drift and hidden contexts, Machine Learning, 1996, 23(1):69-101.

[10] Kulkarni P, Ade R. Logistic Regression Learning Model for Handling Concept Drift with Unbalanced Data in Credit Card Fraud Detection System, Proceedings of the Second International Conference on Computer and Communication Technologies. Springer India, 2016:681-689.

[11] Lu Q, Ju C. Research on Credit Card Fraud Detection Model Based on Class Weighted Support Vector Machine, Journal of Convergence Information Technology, 2011, 6(1):62-68.

[12] Srivastava A, Kundu A, Sural S, et al. Credit Card Fraud Detection Using Hidden Markov Model, IEEE Transactions on Dependable & Secure Computing, 2008, 5(1):37-48.

[13] Olszewski D. Fraud detection using self-organizing map visualizing the user profiles, Knowledge-Based Systems, 2014, 70(C):324-334.

[14] Dhanapal R, Gayathiri. P. Credit Card Fraud Detection using Decision Tree for Tracing Email and IP, International Journal of Computer Science Issues, 2012, 9(5).

[15] Malekipirbazari M, Aksakalli V. Risk assessment in social lending via random forests, Pergamon Press, Inc. 2015.

[16] Breiman L. Random Forests, Machine Learning, 2001, 45(1):5-32.

[17] Ghosh S, Reilly D L. Credit card fraud detection with a neural-network, Twenty-Seventh Hawaii International Conference on System Sciences. IEEE, 2011:621-630.

[18] Chen X, Gupta A. Webly Supervised Learning of Convolutional Networks, IEEE International Conference on Computer Vision. IEEE, 2016:1431-1439.

[19] Vlasselaer V V, Bravo C, Caelen O, et al. APATE : A novel approach for automated credit card transaction fraud detection using network-based extensions, Decision Support Systems, 2015, 75:38-48.

[20] Jiang C, Song J, Liu G, et al. Credit Card Fraud Detection: A Novel Approach Using Aggregation Strategy and Feedback Mechanism, IEEE Internet of Things Journal, 2018, PP(99):1-1.

[21] Panigrahi S, Kundu A, Sural S, et al. Credit card fraud detection: A fusion approach using DempsterCShafer theory and Bayesian learning, Information Fusion, 2009, 10(4):354-363.

[22] Bahnsen A C, Aouada D, Stojanovic A, et al. Detecting Credit Card Fraud Using Periodic Features, IEEE, International Conference on Machine Learning and Applications. IEEE, 2016:208-213.

[23] Brause R, Langsdorf T, Hepp M. Neural data mining for credit card fraud detection, IEEE International Conference on TOOLS with Artificial Intelligence, 1999. Proceedings. IEEE, 1999:103-106.

[24] Snchez D, Vila M A, Cerda L, et al. Association rules applied to credit card fraud detection, Expert Systems with Applications, 2009, 36(2):3630-3640.

[25] Zadrozny B, Elkan C. Obtaining Calibrated Probability Estimates from Decision Trees and Naive Bayesian Classifiers, International Conference on Machine Learning, 2001:609–616.

[26] Hand D J. Measuring classifier performance: a coherent alternative to the area under the ROC curve, Machine Learning, 2009, 77(1):103-123.