Detecting Fraud Transaction using Ripper Algorithm Combines with Ensemble Learning Model

Vo Hoang Khang[1], Cao Tung Anh[2], Nguyen Dinh Thuan[3]
Faculty of Information Technology, HUTECH University, Ho Chi Minh City, Vietnam[1, 2]
Faculty of Information Systems-University of Information Technology,
Vietnam National University Ho Chi Minh City (VNUHCM - UIT), Ho Chi Minh City, Vietnam[3]

***Abstract*—In the context of the 4.0 technology revolution, which develops and applies strongly in many fields, in which the banking sector is considered to be the leading one, the application of algorithms to detect fraud is extremely important. necessary. In recent years, credit card transactions including physical credit card payments and online payments have become increasingly popular in many countries around the world. This convenient payment method attracts more and more criminals, especially credit card fraud. As a result, many banks around the world have developed fraud detection and prevention systems for each credit card transaction. Data mining is one of the techniques applied in these systems. This study uses the Ripper algorithm to detect fraudulent transactions on large data sets, and the results obtained with accuracy, recall, and F1 measure of more than 97%. This research then used the Ripper algorithm combined with Ensemble Learning models to detect fraudulent transactions, the results are more than 99% reliable. Specifically, this model using the Ripper algorithm combined with the Gradient Boosting method has improved the predictive ability and obtained very reliable results. The use of algorithms combined with machine learning models is expected to be a new topic and will be widely applied to banks' or organizations' activities related to e-commerce.**

***Keywords*—Financial fraud; data mining; credit card fraud; transaction; ensemble methods**

I. INTRODUCTION

The e-commerce sector is expected to account for 21.8% of total global retail sales by 2024 [1]. As the worldwide e-commerce market share increases risks also increase, trusting becoming a concern as the number of frauds and scams has increased significantly over the past few years [2]. The e-commerce market in 2021 faces a risk of US$20 billion due to fraudulent transactions [3]. The most common types of fraud include account takeovers, identity theft, covert fraud, and chargeback fraud [3]. These frauds can affect the finances of both consumers and merchants and reduce overall confidence in e-commerce [4].

Data mining techniques have shown promising performance in detecting fraud [5], [6], [7] and rule from large data sets [8]. To solve the problem of transaction fraud in e-commerce, there have been many studies using data mining rules to build laws, thereby detecting transaction fraud. These studies and their results will be introduced in Part II (Related Works).

In many forecasting models, the Repeated Incremental Pruning to Produce Error Reduction (Ripper) algorithm is chosen to apply [9] because this algorithm has high reliability and coverage. Next, this study uses a synchronous learning method in machine learning to combine with Ripper algorithm to improve the reliability and accuracy of fraud detection. The results of study have yielded very positive results. This model has been applied in many other fields but is still quite new in the field of e-commerce in the world as well as in Vietnam.

The article structure consists of five sections. Section I is an overview of the research, Section II introduces the research done related to the article, and Section III is the proposed solutions and techniques to solve the problem dealing with the problem posed. Section IV lists the order of steps to conduct the experiment as well as the software and tools used. Finally, Section V summarizes and concludes with the issues achieved in the study as well as suggestions for future research.

II. RELATED WORKS

Research related to fraudulent transaction detection is numerous and has been conducted in a variety of fields, including finance, e-commerce, credit cards, multi-level commerce, and many others. Data mining techniques used include Support vector machine (SVM), Fuzzy Logic based system (FL), Hidden Markov Model (HMM), Artificial Neural network (ANN), Genetic Algorithm (GA), Bayesian Network (BN), K-Nearest Neighbor Algorithm (KNN), Decision Tree (DT), Logistic Regression (LR), Outlier's Detection and other Classification techniques [6], Below is a summary of the articles, mining techniques used, and research results.

- Support Vector Machine:

N. K. Gyamfi and J. D. Abdulai used SVM to detect bank fraud [6]. The result shows an improvement of the SVM method by 80% which is considered better performance.

M. Jeragh, M. AlSulaimi combined autoencoders and one class support vectors machine for fraudulent credit card transactions detection [6]. The result of two combined models achieved promising results, especially when evaluated using metrics performance.

K. Poongodi and D. Kumar used SVM with Information Gain Based Classification algorithm to identify fraudulent and legal transactions [10]. The result shows that the accuracy of detection is enhanced in the credit card fraud detection system by using SVM with information gain-based classification.

Y. Kumar, S. Saini, and R. Payal proposed a comparison between three data mining algorithms: Logistic Regression, Random Forest, and Support Vector Machine [11]. The study shows that when comparing the result of three algorithms the random forest algorithm gives better results for the classification of fraud with 81.79% accuracy.

- Artificial Neural Network:

A. A. Rizki, I. Surjandari, and R.A. Wayasti used data mining applications to detect financial fraud in Indonesia's public companies [6]. The finding proved that feature selection helped in increasing the accuracy of the SVM method with 88.37% while ANN gave the most effective accuracy with 90.97%.

I. Sadgali, N. Sael, and F. Benabbou proposed a comparative study using Neural Networks Techniques, Neural Networks (NN), Multilayer Perceptron Layer (MPL), and Convolutional Neural Networks (CNN), for credit card fraud detection [12]. The result shows that CNN has the highest accuracy.

A Sahu, GM Harshvardhan, MK. Gourisaria detected fraudulent transactions by several machine learning models and the artificial neural network [13]. The result shows that all the models have a very high percentage of true negatives due to the heavy count of non-fraud cases.

- Decision Tree:

V. Jain, M. Agrawal, and A. Kumar [14] used three machine learning algorithms, Decision Tree, Random Forest, and XGBoost applied to a data set have the data of 284808 credit cards. The performance of the XGBoost algorithm is found best with the highest accuracy of 99.962 percent. The performance of the Decision Tree is minimum with an accuracy of 99.923 and the performance of the Random Forest algorithm is 99.957 percent in credit card transactions.

J. Soyemi and H. Mudasiru implemented a decision tree algorithm augmented with regression analysis for fraud detection in credit cards [15]. This study was able to achieve its aim by building a machine-learning model that is capable of detecting fraud.

In this article, this study used Ripper algorithm [7] to perform a reliability test when predicting transaction fraud when the transaction volume is increasing and unbalanced, then combine Ripper and Ensemble Learning Methods to create a more reliable predictive model (ROC increase from 88% to 98%).

III. Proposed Solutions and Techniques

A. Implementing Data Set

Focusing on the objective of this study, "Aggregated Financial Dataset for Fraud Detection" was selected [16], [17]. The dataset contains approximately 6.3 million data about mobile money transactions and consists of 10 attribute columns, and one target column. This dataset includes a 30-day simulation of real-time transactions and is executed in steps, each mapping to an hour. The original data set includes the following columns:

- step: transaction step
- amount: the number of coins traded
- name_org: sender account name
- old_balance_org: initial balance of the sending account
- new_balance_org: the following balance of the sending account
- name_dest: recipient account name
- old_balance_dest: initial balance of the receiving account
- new_balance_dest: the following balance of the receiving account
- is_fraud: whether the transaction is fraudulent or not
- is_flagged_fraud: whether the transaction is suspected of fraud
- transaction_type: transaction type (5 types)
- isFraud: This is a transaction made by rogue agents inside the simulation. In this particular dataset, the fraudulent behavior of the agents aims to gain profits by controlling, taking over the customer's account, attempting to withdraw money by switching to another account, and then withdrawing money from the system.
- isFlaggedFraud: The business model aims to control large transfers from one account to another and flag illegal attempts. An illegal attempt in this dataset is an attempt to transfer more than 200000 in a single transaction. The isFlaggedFraud column is used when there is an attempt to transfer more than 200000 in a transaction, so basically for isFlaggedFraud, it will be 1 if the transaction amount exceeds 200000. When isFlaggedFraud equals 1, the value of isFraud equals 1, so isFlaggedFraud can be discarded on processing if necessary.

B. Fraudulent Transaction

Transaction fraud is an act of fraud in the process of conducting financial transactions, intending to appropriate property, money, or personal benefits of the offender.

Types of transaction fraud can range from using fake information to open an account or borrow money, to performing invalid transactions, transferring money to the wrong address, or spoofing identity to cheat others.

Transaction fraud not only causes damage to the assets of the parties involved but also affects the trust and reputation of the financial system, reducing the ability to conduct safe and efficient transactions. Therefore, detecting and preventing transaction fraud is very important to protect the safety and reliability of the financial system.

In addition to the transaction that is considered fraudulent as mentioned above, in the data set under consideration, a transaction is suspected to be fraudulent if it violates integrity constraints. There are two types of integrity constraints on this data set, domain-related integrity constraints and inter-attribute integrity constraints. The first is a domain-related constraint: it ensures that a field's value must meet a specific condition, ensuring that a field's value falls within a specific range. The second constraint is an inter-attribute constraint: a type of constraint in a relational database that ensures that the values in one or more pairs of attributes are logically related to each other. Specifically, if the value of one attribute changes, the values in other attributes that depend on it will also change accordingly to ensure data integrity and accuracy.

Here are some examples of the two integrity constraints mentioned above.

+ R1: For all money transfers and withdrawals, the old balance cannot be equal to 0.

∄q: (q.type = TRANSFER ∧ q.oldBalanceOrig = 0) ∨ (q.type = CASH_OUT ∧ q.oldBalanceOrig = 0)

+ R2: After transferring money, the new balance cannot be equal to 0.

∄q: (q.type = TRANSFER ∧ q.newBalanceDest ≤ 0)

+ R3: For all money transfers and withdrawals, the old balance is always greater than the amount to be transferred.

∄q: (q.type = TRANSFER ∧ q.oldBalanceOrig > q.amount) ∨ (q.type = CASH_OUT ∧ q.oldBalanceOrig > q.amount)

+ R4: In all money transfers, the old balance is always greater than the new balance.

∄q: (q.type = TRANSFER ∧ q.oldBalanceOrig > q.newBalanceOrig)

+ R5: In all money transfers and withdrawals, the old balance minus the new balance is always greater than or equal to the amount to be transferred-withdrawn.

∄q: (q.type = TRANSFER ∧ q.oldBalanceOrig - q.newBalanceOrig < q.amount) ∨ (q.type = CASH_OUT ∧ q.oldBalanceOrig - q.newBalanceOrig < a.amount)

+ R6: Recipient did not receive enough transaction funds.

∄q: (q.type=TRANFER ∧ q.oldBalanceDest + q.amount < q.newBalanceDest)

+ R7: After transferring money, the recipient's old balance must be less than the recipient's new balance.

∄q: (q.type = TRANSFER ∧ q.oldBalanceDest > q.newBalanceDest)

+ R8: Do not transfer and withdraw all funds or do not transfer and withdraw more than the amount available when making the withdrawal.

∄q: (q.type = TRANSFER ∧ q.amount ≥ q.oldBalanceOrig) ∨ (q.type = CASH_OUT ∧ q.amount ≥ q.oldBalanceOrig)

Thus, a transaction in the "Aggregated Financial Dataset for Fraud Detection" dataset that violates one of the abovementioned constraints will be considered a fraudulent transaction.

C. Proposed Algorithm

This study implemented the RIPPER (Repeated Incremental Pruning to Produce Error Reduction) algorithm. In machine learning, iterative incremental pruning to reduce errors is a propositional rule learner proposed by William W. Cohen (1995) [9] as an optimal version encoding of the IREP algorithm. The Ripper algorithm is a machine learning algorithm used to detect fraud in financial transactions. This algorithm works by classifying transactions into two categories: trusted transactions and fraudulent transactions. It then uses machine learning techniques to identify attributes that are important to distinguish between these two types of transactions. RIPPER is a greedy algorithm implemented in the following steps:

+ Step 1: Divide the training data into two parts: the training set and the test set.

+ Step 2: Determine the starting rule set by finding all the rules that can be determined from the initial training set.

+ Step 3: Use the starting rule set to classify the records in the training set.

+ Step 4: Repeat the following steps to find and remove unnecessary rules sequentially:

a) evaluate all rules in the current rule set,

b) remove unnecessary rules,

c) construct a new set of rules from the remaining set of rules.

+ Step 5: Use the final rule set to classify the records in the test set.

The RIPPER algorithm is used to solve classification problems where the goal is to classify objects into known classes. This algorithm has proven to be effective in many practical applications, including handwriting recognition, disease detection, and document classification.

When implemented, this research handled the execution steps of Ripper as follows:

+ Step 1: Read transaction data from the CSV file.

+ Step 2: Convert the column’s type to match the requirement.

+ Step 3: Apply Ripper with several RipperK optimization iterations equals 10.

+ Step 4: Train the model.

Ripper has improved IREP with the OPIMIZERULESET process. This process optimized IREP's code and reduced it to a leaner one. Ripper works well on data sets with unbalanced class distribution. In a dataset, if there is a record set in which most of the records belong to a particular class and the remaining records belong to different classes, then the data set is said to have a non-distribution balance between classes. It works well with noisy datasets because it uses a validation set to prevent the model from overfitting.

D. Proposed Models

In machine learning, no one algorithm is always good for all applications and all data sets because machine learning algorithms are usually based on a set of parameters (hyperparameters) or a certain assumption about the distribution data. Therefore, to find the right algorithms for our dataset, we may need a lot of time to test different algorithms and then adjust the algorithm's parameters (tuning hyperparameters) to obtain the highest accuracy.

To increase the accuracy of the dataset is to combine several models. This method is called Ensemble Learning. Different models with different capabilities can best perform different types of work, and when combined properly, form a powerful hybrid model capable of improving performance, and overall performance compared to using the models alone. Ensemble Methods are divided into the following three categories: Boosting, Bagging, and Stacking.

1) Gradient boosting [18]: Gradient Boosting is a machine learning algorithm of type Ensemble Learning, it uses a reinforcement process based on sub-models to create a better overall model.

Gradient Boosting handles data classification or regression using a series of sub-models that are built on top of each other, each of which targets the errors of the previous model. This process results in a better overall model, with more accuracy than each sub-module.

Gradient Boosting is commonly used in classification and regression problems, such as handwriting classification, spam email classification, stock price forecasting, and other problems in the field of artificial intelligence.

- Gradient Boosting Model: illustrated in Fig. 1 below.

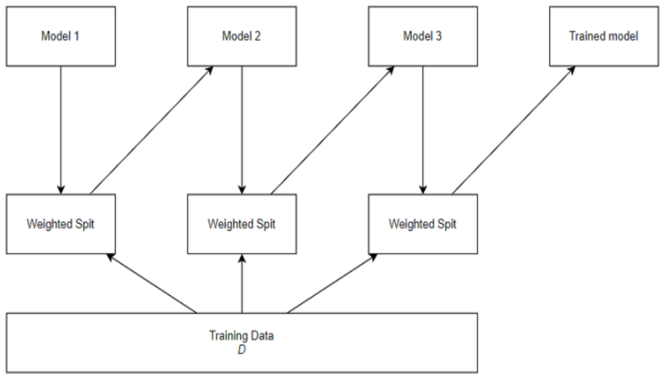

Fig. 1. Gradient boosting operation model.

- Processing steps of the Gradient Boosting Model:

+ Step 1: Define a lost function.

+ Step 2: Build a base model to make a prediction.

+ Step 3: Compute each prediction's residuals (the difference between the predicted and actual values).

+ Step 4: Fit a new model to predict these residuals.

+ Step 5: Add this new model's predictions to the previous model's predictions.

2) Bagging [19]: Bagging (Bootstrapped Aggregating) is a machine learning algorithm of type Ensemble Learning, it uses several sub-models to create a better overall model.

Bagging deals with the classification or regression of data using several sub-models built on the same data set, but each of the sub-models is trained on a bootstrapped dataset (the sampled data set), by starting from an original data set and then choosing a random number of data points to replace the new data set). This process results in a better overall model, with more accuracy than each submodule.

Bagging is commonly used in classification and regression problems, such as handwriting classification, spam email classification, stock price forecasting, and other problems in the field of artificial intelligence.

- Bagging model: illustrated in Fig. 2 below.

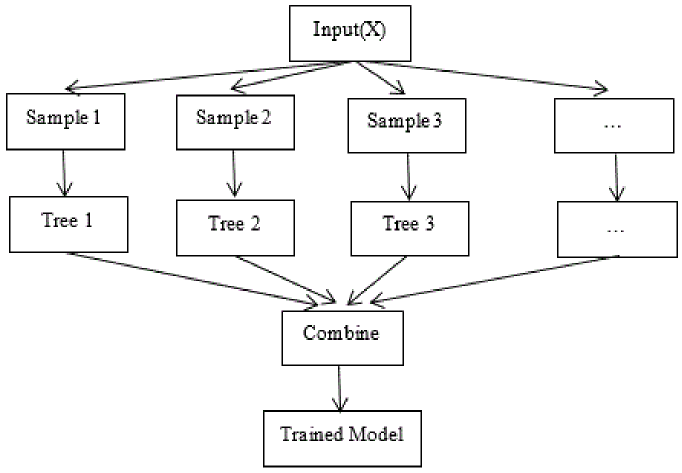

Fig. 2. Bagging operation model.

- Processing steps of the Bagging Model:

+ Step 1: Build k bootstrap samples from the original dataset.

+ Step 2: For each bootstrap sample, build a decision tree.

+ Step 3: Average the predictions of each tree to produce a final model.

3) Stacking [20]: Stacking (stacked generalization) is a machine learning algorithm in the Ensemble Learning category. It uses a set of submodules to create a better overall model.

In Stacking, sub-models are trained on an initial dataset and then used to predict values for the new data set. The prediction results of each sub-model are used as input to the overall model. The population model is then trained on the sub-model prediction results to predict the final value.

- Stacking model: illustrated in Fig. 3 below.

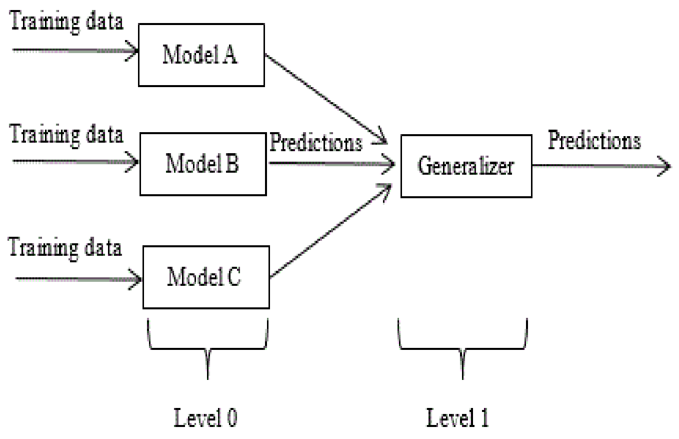

Fig. 3. Stacking operation model.

- Processing steps of the Stacking model:

+ Step 1: Split the dataset into training and testing datasets.

+ Step 2: Train multiple models on the training dataset.

+ Step 3: Make a prediction using these models on the testing dataset.

+ Step 4: Use these predictions as input features to train a new model (the meta-model) on the testing dataset.

E. *Combination of Ripper with Ensemble Learning Model*

1) *Ripper combines with gradient boosting:*

+ Step 1: Read transaction data from the CSV file.

+ Step 2: Declare Gradient Boosting algorithm using exponential loss function, 1000 boosting stages, and square error criterion to qualify the quality of every split action.

+ Step 3: Train the model.

+ Step 4: Consecutively apply Ripper to improve model metrics.

2) *Ripper combines with bagging:*

+ Step 1: Read banking data from the CSV file.

+ Step 2: Initialize the Bagging algorithm with the number of rules generated in the training process equal to 1000.

+ Step 3: Apply Ripper to improve the accuracy.

+ Step 4: Train the model.

3) *Ripper combines with stacking:*

+ Step 1: Read banking data from the CSV file.

+ Step 2: Initialize Random Forest algorithm with a random state is 10 and 100 trees in the forest.

+ Step 3: Initialize Linear SVC algorithm with a random state equal to 42.

+ Step 4: Initialize Standard Scaler algorithm.

+ Step 5: Initialize the Stacking algorithm with the three algorithms mentioned above and Logistic Regression as the final algorithm.

+ Step 6: Apply Ripper to improve the model's metrics.

+ Step 7: Train the model.

IV. EXPERIMENT

A. *Experimental Tools and Software*

Programming environment and tools: Python and PyCharm. Python is a programming language widely used in data science and machine learning. PyCharm is a fully integrated development platform (IDE) designed specifically for Python programmers.

We implemented the algorithms on a laptop with the following configuration: CPU: Intel core i7-12700H, GPU: Intel Iris XE Graphics, RAM: 16Gb.

- Implement Ripper algorithm to experiment stability with unbalanced data:

+ Initialize Ripper: ripper = RIPPER()

The parameters of RIPPER() are shown in Table I.

TABLE I. PARAMETERS OF RIPPER()

Parameters	**Description**	**Default value**	**Data type**
k	The number of loops to optimize	2	int
prune_size	Pruning ratio (prune into two sets, grown set, and prune set)	0.33	float
Random_state	Rules generating seed (rules generated are the same if seeds are the same)	none	int
max_rules	Maximum number of rules	none	int
max_rule_conds	A maximum number of conditions in each rule.	none	int
max_total_conds	A maximum number of conditions in a whole rule set.	none	int
verbosity	Between 0 and 5, print to the terminal rule-generating process. 0 means do not print. The larger the number the more detail: 1: Print the main process. 2: Print the optimization phases. 3: Print the calculations needed to optimize. 4: Print the optimization/pruning process step by step. 5: Print the operations needed to add/prune rules.	0	int

+ Training: ripper.fit (<file data>, <classify col>, <positive value>…)

The parameters of the .fit() method are shown in Table II.

TABLE II. THE PARAMETERS OF THE .FIT()

Parameter	Description	Data type
trainset	The data set used to train.	DataFrame/Numpy array
y	Class labels corresponding to trainset rows. Use if class labels aren't included in the trainset.	str/int/bool
class_feat	Column name or index of the class feature. The use of the class feature is still in the trainset.	str, int
pos_class	Name of the positive class.	str/bool
feature_names	Specify feature names. If None, feature names default to column names for a DataFrame, or indices in the case of indexed iterables such as an array or list.	List<str>
initial_model	Pre-existing model from which to begin training.	str
cn_optimize	Use algorithmic speed optimization.	bool

+ Dataset processing:

This study used the "Aggregated Financial Dataset for Fraud Detection" dataset introduced above for processing. First, remove unnecessary columns to reduce processing time, those columns include name_org and name_dest. Next, divide the data set into the train and test sets. The train set consists of 6000 lines of cheat data and 4 million lines of non-cheat data. The test set consists of 2000 lines of fraudulent data and 2 million lines of non-cheat data.

Continue to divide the training set into four small datasets and increase the imbalance rate between the sets to evaluate the effectiveness of Ripper against large unbalanced sets:

- The first set contains 6000 fraudulent data and 60,000 non-fraud data.
- The second set contains 6000 fraudulent data and 100,000 non-fraud data.
- The third set includes 6000 fraudulent data and 150,000 non-fraud data.
- The fourth set contains 6000 fraudulent data and 300,000 non-fraud data.

The test set consists of 2 thousand fraudulent lines and 2 million non-fraud lines. This set contains data that RIPPER has never encountered before, used to evaluate accuracy.

+ Results:

Results were obtained after performing RIPPER with 4 data sets (Table III).

TABLE III. STATISTICAL RESULTS AFTER TRAINING

Dataset	TPR	FPR	Preci-son	Recall	F1 score	ROC
6_60	0.996	0.03	0.97	0.96	0.96	0.977
6_100	0.992	0.026	0.97	0.86	0.91	0.949
6_150	0.995	0.031	0.97	0.88	0.92	0.906
6_300	0.996	0.023	0.98	0.79	0.87	0.880

Meaning of columns

In the Ripper algorithm, "Positive" and "Negative" are two types of labels used to evaluate and classify transactions.

+ Positive: represents valid transactions, no fraud.

+ Negative (negative): represents fraudulent transactions.

During training, the algorithm will use transactions labeled Positive and Negative to build a classification model for predicting whether new transactions will be fraudulent or not. In a data set of credit card transactions, a transaction labeled Positive would be considered a valid and fraud-free transaction, while a transaction labeled Negative would be considered a likely transaction cheat. Positive and Negative labels play an important role in evaluating and improving the accuracy of the Ripper algorithm in detecting transaction fraud.

- Dataset: dataset used for training
- True positive rate (TPR): The ratio of correctly predicting fraud cases, TPR is calculated using the formula:
- True Positive / (True Positive + False Negative)
- False positive rate (FPR): The ratio of false predictions of fraud cases, FPR is calculated using the formula:
- False Positive / (False Positive + True Negative)
- Precision: Precision is the ratio of True Positive predictions that are correct to the total number of True Positive predictions; precision is calculated using the formula:
- True Positive / (True Positive + False Positive) = Correct True Positive Predictions / True Positive Predictions
- Recall: Recall is the ratio of True Positive predictions that are correct to the total number of True Positives on the whole dataset; recall is calculated using the formula:
- True Positive / (True Positive + False Negative) = Number of correct TP predictions / Total number of TP
- F1 score: F1 score is the harmonic mean of precision and recall, the more accurate the F1 score, the better the model predicts; F1 score is calculated using the formula: 2 * ((precision * recall) / (precision + recall))

ROC: It has a value of [0,1], representing the model's prediction accuracy. The higher the ROC score, the more accurate the model is. Table IV below is the rule list with the number of iterations of Ripper and the refined rule set.

TABLE IV. RULE LIST WITH THE NUMBER OF ITERATIONS RIPPER AND REFINE RULE SET WITH K=10

Number	Rule
1	transaction_type==1 and new_balance_dest<=16123.02 and old_balance_dest<=1087.36
2	transaction_type==1 and new_balance_dest<=7306.88 and old_balance_dest<=225.92
3	transaction_type==1 and new_balance_dest<=16123.02 and old_balance_dest<=1087.36
4	new_balance_org<=1744.75 and transaction_type==1 and new_balance_dest<=17247.49 and old_balance_dest<=11186.22
5	new_balance_org<=1572.27 and amount>=2106972.01 and transaction_type==0
6	transaction_type==0 and amount>=3529762.55
7	new_balance_org<=1572.27 and old_balance_org>=1388977.19 and old_balance_org<=1908598.9
8	transaction_type==0 and amount>=5014130.87
9	amount>=5014130.87 and new_balance_dest<=17247.49
10	transaction_type==0 and amount>=1867568.7 and amount<=3529762.55
11	transaction_type==0 and amount>=1867568.7 and amount<=3529762.55
12	transaction_type==0 and amount>=2519153.53 and amount<=5014130.87
13	new_balance_org<=1572.27 and old_balance_org>=955088.21 and old_balance_org<=1388977.19 and transaction_type==0
14	new_balance_org<=1744.75 and old_balance_org>=946517.94 and old_balance_org <=1204862.32 and old_balance_dest<=11186.22
15	transaction_type==0 and amount>=1262654.09 and amount <=1867568.7 and
16	transaction_type==0 and amount>=1262654.09 and amount <=1867568.7
17	transaction_type==0 and amount>=1697480.93 and amount <=2519153.53
18	new_balance_org<=1572.27 and old_balance_org>=697672.86 and old_balance_org<=955088.21 and transaction_type==0

Match the rules shown in Table V.

TABLE V. RULES WITH HIGH COVERAGE

No	Rule	Ratio
1 ⊃ 2 ⊃ 3	transaction_type==1 and new_balance_dest<=16123.02 and old_balance_dest<=1087.36	50%
4	new_balance_org<=1744.75 and transaction_type==1 and new_balance_dest<=17247.49 and old_balance_dest<=11186.22	48%
5	new_balance_org<=1572.27 and amount>=2106972.01 and transaction_type==0	10%
6 ⊃ 8	transaction_type==0 and amount>=3529762.55	6.5%
7	new_balance_org<=1572.27 and old_balance_org>=1388977.19 and old_balance_org <=1908598.9	5.5%
9	amount>=5014130.87 and new_balance_dest<=17247.49	4,7%

At this stage, the retainer is used. The dataset is divided into 10 sets. The purpose of the division is that after each step, the percentage of fraudulent data is gradually increased, creating an imbalance and checking the stability of the rule set. Each set contains 200 fraudulent data and 2000 non-cheat data (the data of these 10 datasets do not overlap). Then, add each data set in turn, corresponding to 10 cycles to see the coverage of the rules after having new data. Table VI below is the simulation rule range for ten cycles.

TABLE VI. SIMULATION RULE RANGE FOR TEN CYCLES

	Rule 1 (%)	Rule 2 (%)	Rule 3 (%)	Rule 4 (%)	Rule 5 (%)	Rule 6 (%)
Base Coverage	50	48	10	6.5	6.5	4.7
1	49	46	8	5	4	6
2	48	44.5	7	5	6	5
3	51.5	50	7	4.5	8.5	3.5
4	43.5	42	8.5	6	6.5	3.5
5	50.5	46.5	7.5	5.5	7	4.5
6	52	49.5	11	7.5	7	5.5
7	48	46.5	8.5	4.5	4	3
8	50	46	8.5	5.5	8	2
9	49.5	48	13	7.5	7	3.5
10	48.5	47	7.5	3.5	3.5	4

After 10 cycles, although the amount of data gradually increases and causes a high imbalance, the test has shown that Ripper is stable and does not suffer from "overfitting".

B. Experimental Results

Finally, this study used the fourth dataset containing 6000 fraudulent data and 300000 non-fraud data above to check the criteria for the Ripper algorithm and Ripper combined with Ensemble Learning Methods.

1) Implement the Ripper

+ Results:

- The set of rules: 31.

- Metrics: described in Table VII below.

TABLE VII. METRICS OF RIPPER

	precision	recall	f1-score	support
0	1.00	1.00	1.00	60296
1	0.97	0.77	0.86	1241
accuracy			0.99	61537
macro avg	0.98	0.88	0.93	61537
weighted avg	0.99	0.99	0.99	61537

- Confusion matrix: False Positive = 60267, False Negative = 29, True Negative = 290, True Positive = 951.

- ROC = 0.88. Execution time: 6m 58s.

2) Implement the Ripper combines with Gradient Boosting (Fig. 4)

+ Operation model:

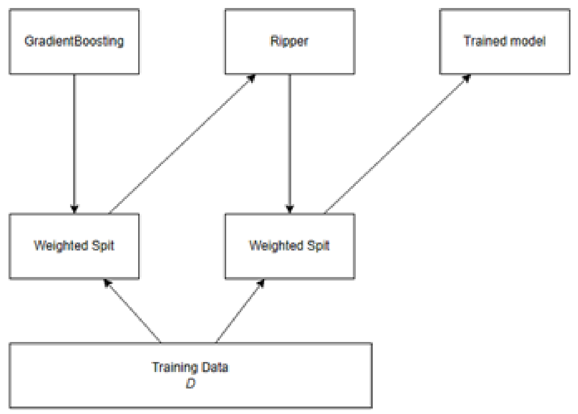

Fig. 4. Operation model by ripper combines with gradient boosting.

+ Results:

- The set of rules: 21

- Metrics: Described in Table VIII below.

TABLE VIII. METRICS OF RIPPER COMBINES GRADIENT BOOSTING

	precision	recall	f1-score	support
0	1.00	1.00	1.00	60372
1	0.98	0.92	0.95	1165
accuracy			1.00	61537
macro avg	0.99	0.96	0.97	61537
weighted avg	1.00	1.00	1.00	61537

- Confusion matrix: False Positive = 60349, False Negative = 23, True Negative = 90, True Positive = 1075.

- ROC = 0.96. Execution time: 10m 57.3s.

3) Implement the Ripper combines with Stacking (Fig. 5) + Operation model:

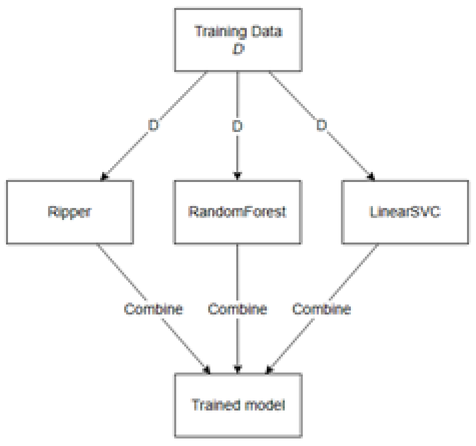

Fig. 5. Operation model by ripper combined with stacking.

+ Results: The set of rules: 45.

- Metrics: Described in Table IX below.

TABLE IX. METRICS OF RIPPER COMBINED WITH STACKING

	precision	recall	f1-score	support
0	1.00	1.00	1.00	60372
1	0.95	0.90	0.93	1165
accuracy			1.00	61537
macro avg	0.97	0.95	0.96	61537
weighted avg	1.00	1.00	1.00	61537

- Confusion matrix: False Positive = 60317, False Negative = 55, True Negative = 111, True Positive = 1054.

- ROC: 0.95. Execution time: 8m 46.8s.

4) Implement the Ripper combines with Bagging (Fig. 6) + Operation model:

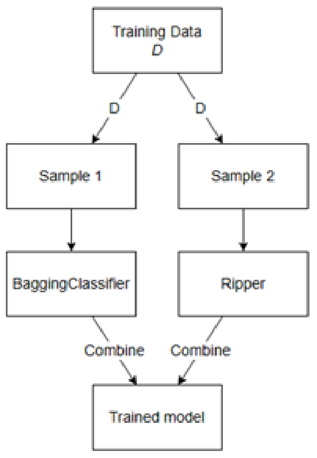

Fig. 6. Operation model by ripper combines with bagging.

+ Results: The set of rules: 44.

- Metrics: Described in Table X below.

TABLE X. METRICS OF RIPPER COMBINES BAGGING

	precision	recall	f1-score	support
0	1.00	1.00	1.00	60372
1	0.95	0.95	0.95	1165
accuracy			1.00	61537
macro avg	0.98	0.98	0.98	61537
weighted avg	1.00	1.00	1.00	61537

- Confusion matrix: False Positive = 60291, False Negative = 42, True Negative = 60, True Positive = 1144.

- ROC: 0.98. Execution time: 14m 41.7s.

C. Comparison

Table XI below summarizes the achieved indicators of each model.

TABLE XI. COMPARE CRITERIA BETWEEN METHODS

Criteria	Ripper (1)	Ripper combines Boosting (2)	Ripper combines Bagging (3)	Ripper combines Stacking (4)
Execution time	6m 58s	10m 57.3s	14m 41.7s	8m 46.8s
The set of rules	32	21	44	45
Precision	0.97	0.98	0.95	0.95
Recall	0.77	0.92	0.95	0.90
F1-score	0.86	0.95	0.95	0.93
ROC	0.883	0.961	0.976	0.952

Of the four models mentioned above, (1) gives the worst results because only a single algorithm is used. Although (2) produces the fewest rules, this is the most optimal set of rules because of its processing model. On the other hand, in models (3) and (4), the sets of rules are generated independently, and then combined, so the rules will tend to be more and overlap. This leads to models (3) and (4) generating more rules but not as efficient the model (2).

This result shows that Ripper combined with Gradient Boosting gives the best results. Compared with related studies in part I such as [4], [6], [7], [11], [12], this result achieved ACC and ROC higher than all. This study shows that Ripper combined with Gradient Boosting gives the best results.

V. CONCLUSION

The Ripper algorithm has been studied and applied effectively in the field of data mining. Ripper is a classification algorithm in machine learning and is used in many different fields. Such as data classification: Ripper can be used to classify data based on their characteristics and values. Object recognition: Ripper can be used to identify objects in an image or video by classifying objects according to their features and values. Market Research: Ripper can be used to analyze market data to identify potential customer groups and other groups. Phrase Search: Ripper can be used to search for phrases in text or other textual data. The limitation of the study is that the algorithm executes slowly if the number of attributes of the data set is too large. In this study, the proposed method was improved by combining Ripper with Ensemble Learning Method, namely Gradient Boosting to create a new predictive model with higher accuracy and reliability (ROC increased from 88% to 98% compared to just using regular Ripper). Depending on the structure and distribution of the data set, this research can use the Ripper algorithm in combination with Gradient Boosting, Bagging, or Stacking to achieve the desired result. Thus, the proposed model has obtained better results than related studies (ACC and ROC). Besides, combining models together for training usually takes more processing time and takes up more memory resources than single models.

However, detecting transaction fraud is a constant challenge, as scammers are always looking for ways to change techniques and tricks to avoid detection. Therefore, the use of fraud detection solutions needs to be continuously updated and optimized to keep the financial system safe and meet the requirements of stakeholders.

REFERENCES

[1] [Online]. Available: https://tinyurl.com/4eh5ex6a. [Accessed 2022].

[2] R. K. Jamra, Kautsarina, D. I. Sensuse, "Systematic review of issues and solutions," In 2020 International Conference on Electrical Engineering and, pp. 1-5, 2020.

[3] N. Maynard, "ONLINE PAYMENT FRAUD: MARKET FORECASTS, EMERGING THREATS & SEGMENT," 2021. [Online]. Available: https://www.businesswire.com; https://www.juniperresearch.com/researchstore/fintech-payments/online-payment-fraud-.

[4] N. Chawla & B. Kumar, "E-commerce and consumer protection in India: The emerging trend," Journal of Business Ethics, pp. 1-24, 2021.

[5] S. Beigi. M. R. Amin Naseri, "Credit card fraud detection using data mining and statistical methods," Journal of AI and Data Mining, Vols. 8: 149-60, 2020.

[6] K. G. Al-Hashedi, P. Magalingam, "Financial fraud detection applying data mining techniques: A comprehensive review from 2009 to 2019," Computer Science Review, vol. 40:100402, 2021.

[7] A. A. N. M. R. A. K. M. B. H. A. K. M. N. I. a. R. M. R. Tahmid Hasan Pranto, "A Blockchain, Smart Contract, and Data Mining Based Approach toward the Betterment of E-Commerce," Cybernetics and Systems, vol. 53, pp. 443-467, 2021.

[8] Wang, Y., H. Liu, and Q. Liu, "Application Research of web log mining in the e-commerce. In 2020 Chinese Control and Decision Conference (CCDC)," IEEE, 2020.

[9] W. W. Cohen, "Grammatically biased learning: learning logic programs using an explicit antecedent description language Artificial Intelligence," vol. 68, pp. 303-366, 1994.

[10] K. Poongodi and D. Kumar, "Support Vector Machine with Information Gain Based Classification for Credit Card Fraud Detection System," The International Arab Journal of Information Technology, vol. 18, pp. 199-207, 7 9 2020.

[11] Y. Kumar, S. Saini, R. Payal, "Comparative Analysis for Fraud Detection Using Logistic Regression, Random Forest and Support Vector Machine," International Journal of Research and Analytical Reviews (IJRAR), vol. 7: 4, pp. 726-731, 2020.

[12] I. Sadgali, N. Sael, F. & Benabbou, "Comparative Study Using Neural Networks Techniques for Credit Card Fraud Detection," in Innovations in Smart City Application, Warszawa, Springer, 2020, pp. 287-296.

[13] A Sahu, GM Harshvardhan, MK Gourisaria, "A Dual Approach for Credit Card Fraud Detection using Neural Network and Data Mining

Techniques," in 2020 IEEE 17th India Council International Conference (INDICON), New Delhi, 2020.

[14] V. Jain, M. Agrawal, and A. Kumar, "Performance Analysis of Machine Learning Algorithms in Credit Cards Fraud Detection," in International Conference on Reliability, Infocom Technologies and Optimization (ICRITO) (Trends and Future Directions), 2020.

[15] J. Soyemi, and H. Mudasiru, "An implementation of decision tree algorithm augmented with regression analysis for fraud detection in credit card," 2020.

[16] "Aggregated financial datasets for fraud detection," 2022. [Online]. Available: Kaggle.com. [Accessed 12 1 2023].

[17] T. H. Pranto, A. A. Noman, M. Rahaman, A. K. M. Bahalul Haque, A. K. M. Najmul Islam & Rashedur M. Rahman, "A Blockchain, Smart Contract and Data Mining Based Approach toward the Betterment of E-Commerc," 2021.

[18] J. H. Friedman, "Stochastic gradient boosting," pp. 367-378, 2002.

[19] Bbeiman, Leo, "Bagging Predictors," Machine Learning, vol. 24, pp. 123-140, 1996.

[20] B. Pavlyshenko, "Using Stacking Approaches for Machine Learning Models," in IEEE Second International Conference on Data Stream Mining & Processing, 2018.