Jutisi: Jurnal Ilmiah Teknik Informatika dan Sistem Informasi
https://ojs.stmik-banjarbaru.ac.id/index.php/jutisi/index
Jl. Ahmad Yani, K.M. 33,5 - Kampus STMIK Banjarbaru
Loktabat – Banjarbaru (Tlp. 0511 4782881), e-mail: puslit.stmikbjb@gmail.com
e-ISSN: 2685-0893

Fraud Detection In E-Money Transactions Using Random Forest

DOI: http://dx.doi.org/10.35889/jutisi.v15i4.3893

Uzer Tarmizi[1*], Aliffia Putri Dito[2]
Data Science Study Program, BPD University, Semarang, Indonesia
*e-mail *Corresponding Author:* uzermagister@edu.ubpd.ac.id

Abstract

The use of electronic money (e-money) in Indonesia has grown rapidly, with transaction values reaching IDR 39.8 trillion in 2024. However, this growth is accompanied by an increase in fraud cases that harm consumers and service providers. This study aims to implement the Random Forest algorithm to detect fraud in e-money transactions. The research data uses the PaySim dataset, consisting of 1,048,575 transactions with 11 features and the isFraud label. The research method includes data preprocessing, handling imbalanced data using SMOTE (Synthetic Minority Oversampling Technique), Random Forest implementation with hyperparameter optimization using Grid Search, and model evaluation using precision, recall, and F1-score metrics. The results showed that the optimized Random Forest model achieved an accuracy of 99.92%, precision of 0.95, recall of 0.93, and F1-score of 0.94 on the test data. Feature importance analysis identified that the amount, oldbalanceDest, and newbalanceDest features are the most significant factors in detecting fraudulent transactions, with contributions of 35%, 28%, and 22%, respectively. The resulting model is able to detect 93% of fraudulent transactions with a prediction error rate of only 5%.

Keywords: *Fraud Detection; E-Money; Random Forest; SMOTE; PaySim*

1. Introduction

The Indonesian digital financial industry has experienced rapid growth, with e-money transactions reaching IDR 39.8 trillion in 2024, a 35% increase from the previous year [1]. However, behind this growth, cybercrime, particularly fraud, poses a serious threat. What makes fraud detection a major challenge in current AI research is the dynamic, adaptive, and continuously evolving nature of fraud, where perpetrators constantly change their modus operandi to evade rule-based systems. Additionally, financial transaction data is highly imbalanced, normal transactions far outnumber fraudulent ones, making it difficult for machine learning models to effectively learn fraud patterns. The need for accurate real-time detection further complicates the challenge, as every second of delay can result in significant financial losses. These characteristics make fraud detection one of the most critical and complex areas of AI research today.

The Financial Services Authority (OJK) reported a 45% increase in digital fraud cases in 2024, with losses reaching hundreds of billions of rupiah [3]. Conventional detection systems, which are generally rule-based (e.g., transaction limits, two-step verification, blacklists), have a fundamental weakness: they can only detect known fraud patterns and cannot adapt to new modus operandi. This results in many fraudulent transactions going undetected (false negatives), while legitimate transactions are incorrectly rejected (false positives). Given the massive volume of e-money transactions, reaching millions per day, manual detection is impossible, creating an urgent need for an intelligent, adaptive system capable of real-time fraud detection on extremely imbalanced data. Random Forest was chosen for this task because it inherently handles imbalanced data through the class_weight parameter, is robust against outliers, provides feature importance for interpretability, and is resistant to overfitting, advantages not shared by simpler algorithms like Logistic Regression or less interpretable ones like Neural Networks.

Several studies have addressed fraud detection in digital financial transactions. Hayati (2024) investigated Random Forest and XGBoost for mobile money fraud detection, showing

promising results but using a relatively small dataset without addressing imbalanced data handling [6]. Chen et al. (2025) implemented Random Forest on mobile payment systems, achieving 99.96% accuracy with a large dataset, yet lacked in-depth feature contribution analysis [7]. Mardiansyah (2023) developed a hybrid Random Forest-Louvain Coloring method for online transfer fraud, improving accuracy but introducing high computational complexity [8]. Lopez-Rojas et al. (2016) created the PaySim dataset as a mobile money transaction simulator, which has become a standard dataset in various studies [9]. Based on these reviews, several research gaps remain: most studies used non-Indonesian datasets, none have specifically analyzed dominant features in e-money fraud detection using the validated PaySim dataset, research on optimized SMOTE for e-money fraud is limited, and most studies focus on accuracy while neglecting critical metrics like precision and recall for rare fraud cases.

To address these gaps, this study proposes implementing the Random Forest algorithm combined with SMOTE (Synthetic Minority Oversampling Technique) to detect fraud in e-money transactions using the PaySim dataset. Random Forest, developed by Breiman (2001), is an ensemble learning method that combines many decision trees to produce accurate and stable predictions, offering advantages such as handling imbalanced data, robustness to outliers, feature importance analysis, and resistance to overfitting [10]. These strengths are reinforced by Chen et al. (2025), who achieved 99.96% accuracy using Random Forest for mobile payment fraud detection [7]. Unlike SVM or Neural Networks, Random Forest provides interpretability through feature importance, a crucial advantage for understanding fraud patterns. SMOTE, developed by Chawla et al. (2002), creates synthetic samples from the minority class by interpolating between nearby minority samples, and has been proven effective in improving classification model performance on imbalanced data [12].

The novelty of this research lies in: (1) a specific focus on e-money transactions using the validated PaySim dataset, (2) in-depth analysis of the most dominant features in e-money fraud detection, (3) optimized SMOTE application to handle extreme data imbalance (0.13% fraud), and (4) comprehensive evaluation metrics (precision, recall, F1-score) more relevant for rare fraud cases than accuracy alone. The scientific contribution is an accurate and interpretable fraud detection model that can explain what factors cause a transaction to be classified as fraud, enabling e-money service providers to understand fraud patterns and take more effective preventive measures. Based on the description above, this study aims to: (1) implement the Random Forest algorithm for fraud detection in e-money transactions, (2) analyze the effectiveness of SMOTE techniques in improving model performance on imbalanced data, (3) identify the most important features in fraud detection, and (4) measure model accuracy using precision, recall, and F1-score metrics.

2. Methodology

2.1 Research Data

This study uses the PaySim (Mobile Money Simulator) dataset from the public Kaggle repository. This dataset is synthetic data that simulates electronic money (e-money) transactions with characteristics close to real conditions [9]. The dataset consists of 1,048,575 transaction rows with 11 features. The following table explains the function of each feature in the dataset:

Table 1. The Function of Each Feature

Feature	Function / Description
step	Transaction time in hours (1 step = 1 hour), covering a 30-day period (744 steps)
type	Transaction type: CASH-IN, CASH-OUT, DEBIT, PAYMENT, TRANSFER
amount	Transaction nominal value
nameOrig	Unique ID of the sender (anonymized)
oldbalanceOrg	Sender's balance before the transaction
newbalanceOrig	Sender's balance after the transaction
nameDest	Unique ID of the recipient (anonymized)
oldbalanceDest	Recipient's balance before the transaction

Feature	Function / Description
newbalanceDest	Recipient's balance after the transaction
isFraud	Target label (dependent variable): 1 = fraudulent transaction, 0 = normal transaction
isFlaggedFraud	System flag for suspicious transactions (transfer > 200,000)

The class proportion in the dataset is highly imbalanced, with 0.13% fraud and 99.87% normal transactions, reflecting real-world conditions where fraud cases are much rarer than normal transactions. Inclusion criteria include all transaction types with valid non-negative balances, while exclusion criteria include transactions with amount = 0 or transactions showing balance inconsistencies (mismatch between amount and balance changes). The dataset can be downloaded for free from https://www.kaggle.com/datasets/ealaxi/paysim1.

2.2 Research Stages

This research was conducted in five main stages. The following is a detailed explanation of each stage:

1) Literature Study and Data Collection

A comprehensive literature review was conducted to build a strong theoretical foundation for the research. This involved studying national and international journals, conference proceedings, and textbooks covering three main areas: fraud detection in digital financial systems, the Random Forest algorithm as an ensemble learning method, and techniques for handling imbalanced data (with particular focus on SMOTE). After the literature review, the PaySim dataset was downloaded from the Kaggle platform. The dataset was then explored in detail to understand its structure, the data types of each feature, and the distribution of values across all variables. This initial exploration also included identifying potential issues such as missing values or outliers, and performing descriptive statistical analysis (minimum, maximum, mean, median, and standard deviation) to understand the basic characteristics of each numerical feature.

2) Data Preprocessing

The dataset was cleaned by checking for missing values, removing duplicates, and verifying balance consistency. New features were engineered to improve model performance, specifically:

$errorBalanceOrig = oldbalanceOrg - amount - newbalanceOrig$ (sender balance discrepancy)

$errorBalanceDest = oldbalanceDest + amount - newbalanceDest$ (recipient balance discrepancy)

Categorical features (transaction type) were converted using one-hot encoding. To handle extreme class imbalance (0.13% fraud), SMOTE was applied to the training data, generating synthetic samples for the minority class through linear interpolation between neighboring samples using the formula:

$$x_{new} = x_i + \lambda \times (x_{zi} - x_i) \quad (1)$$

where λ is a random number between 0 and 1, until classes became balanced (50% normal, 50% fraud). Numerical features were normalized using StandardScaler $z = \frac{x - \mu}{\sigma}$. Finally, the data was split using stratified sampling: 70% training, 15% validation, and 15% testing.

3) Random Forest Implementation

The Random Forest model was implemented using Python with the scikit-learn library. Hyperparameter optimization was performed using Grid Search with 5-fold Cross-Validation on the validation data, testing n_estimators (50, 100, 200), max_depth (10, 20, 30, None), min_samples_split (2, 5, 10), min_samples_leaf (1, 2, 4), and class_weight ('balanced', None). The best combination was selected based on F1-score. The model was then trained on SMOTE-processed data. For each tree, a bootstrap sample was created, and at each node a random subset of features was selected. The best split was chosen using Gini Impurity:

$$Gini(t) = 1 - \sum_{i=1}^{C} p_i^2 \quad (2)$$

or Entropy:

$$Entropy(t) = -\sum_{i=1}^{C} p_i \log_2 p_i \quad (3)$$

to minimize impurity. Predictions were made through majority voting:

$$\hat{y} = mode\{\hat{y}_1, \hat{y}_2, \ldots, \hat{y}\_T\} \quad (4)$$

where T is the number of trees ($n\_estimators$) and $\hat{y}_t$ is the prediction from the t-th tree.

4) Analysis and Interpretation
Feature importance was calculated based on how often a feature was used for splitting and how much it contributed to reducing impurity. Performance was compared between models with and without SMOTE using the following metrics : $Precision = \frac{TP}{TP+FP}$, $Recall = \frac{TP}{TP+FN}$, $F1\text{-}Score = 2 \times \frac{Precision \times Recall}{Precision + Recall}$, $Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$,
ROC-AUC (Area Under the Curve).
Results were visualized using confusion matrix heatmaps, feature importance bar charts, and ROC curves.
5) Report Writing and Publication
The final stage involved in-depth interpretation of all research results to extract meaningful insights and conclusions. A journal article was written following the required template and guidelines, ensuring that all sections (introduction, methodology, results, discussion, and conclusion) were properly structured and referenced. A comprehensive final report was prepared documenting the entire research process, findings, and recommendations. The code was thoroughly documented with clear comments and explanations, and prepared in a repository for future development, replication, and reproducibility. This stage also included preparing supplementary materials such as figures, tables, and additional data analyses, as well as final proofreading and formatting to meet publication standards.

2.3 Data Collection and Analysis Techniques

Data collection is conducted through documentation study by taking secondary datasets from the public Kaggle repository. The dataset used is PaySim, which has been publicly published and is licensed for research purposes. Data analysis is conducted through several structured stages using the Python programming language with supporting libraries. Pandas for data manipulation and preprocessing, NumPy for numerical operations, Scikit-learn for Random Forest implementation, Grid Search, and model evaluation, Imbalanced-learn for SMOTE implementation, Matplotlib and Seaborn for data visualization. Formulation and Analysis Procedures:

1) Exploratory Data Analysis (EDA): Descriptive statistical analysis is conducted to understand data distribution, feature correlations, and identification of initial patterns.
2) Preprocessing: As explained in sub-section 3.2, includes data cleaning, feature engineering, encoding, SMOTE, normalization, and data splitting.
3) Modeling: The Random Forest model is trained using the processed training data. The training process uses the Random Forest algorithm as explained in sub-section 3.4.
4) Model Evaluation: The model is evaluated using the testing data with metrics:
   - Confusion Matrix: Displays the number of True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN).
   - Precision : $\text{Precision} = \frac{TP}{TP+FP}$ (measures the accuracy of fraud predictions).
   - Recall: $\text{Recall} = \frac{TP}{TP+FN}$ (measures the ability to capture fraud).
   - F1-Score: $\text{F1} - \text{Score} = 2 \times \frac{Precision \times Recall}{Precision + Recall}$ (harmonic mean of precision and recall).
   - ROC-AUC : Measures the model's ability to distinguish between fraud and normal classes.

5) Interpretation: Feature importance analysis is conducted to identify the most dominant features in fraud detection.

2.4 Random Forest Model and Hyperparameters

Random Forest is an ensemble learning algorithm developed by Leo Breiman (2001) [10]. This algorithm works by building many decision trees during the training phase and combining their results through voting (for classification) or averaging (for regression). The following is the formulation and working mechanism of Random Forest:

a. Decision Tree Formation Process

Each decision tree in Random Forest is built using the following procedure:

1) Bootstrap Sampling: From the training data of size *N*, a random sample of size *N* is taken with replacement (bootstrap sample). This means each sample has an equal chance of being selected, and one sample can be selected more than once.
2) Random Feature Selection: At each tree node, a small number of features (*m*) is randomly selected from the total features (*M*), where *m*<<*M* (typically $m = \sqrt{M}$ for classification). This aims to reduce correlation between trees, making the model more robust.
3) Best Split Selection: From the m selected features, the feature and threshold value that produce the best split are sought based on Gini Impurity or Entropy criteria. The best split is the one that best separates the data into homogeneous classes.
1) Repetition: The split process is repeated recursively on each child node until stopping conditions are reached, such as maximum depth (max_depth) or minimum sample count $(min\_samples\_split)$.
2) No Pruning: Unlike a single decision tree, trees in Random Forest are not pruned. Each tree is allowed to grow fully.

b. Prediction Process

After all trees are formed, the prediction process for new data is as follows:

1) New data is input into each decision tree.
2) Each tree provides a class prediction (fraud or normal).
3) The final prediction result is determined through majority voting $(\hat{y})$.

c. Advantages of Random Forest for Fraud Detection

Random Forest has several advantages that make it suitable for fraud detection:

1) Handling Imbalanced Data: The class_weight='balanced' parameter gives more weight to the minority class (fraud) when calculating impurity, making the model more sensitive to fraud.
2) Robust to Outliers: Because it uses a bagging mechanism (bootstrap aggregating), Random Forest is not overly affected by outlier data.
3) Feature Importance: Random Forest provides feature importance values that show how much each feature contributes to predictions. This is important for result interpretation.
4) Not Prone to Overfitting: By combining many trees, Random Forest reduces the risk of overfitting that commonly occurs with single decision trees.

2.5 Experimental Design

The experimental design in this study is structured to test the performance of the Random Forest model in detecting fraud in e-money transactions. The experiment is designed to answer the research questions and test the effectiveness of the SMOTE technique in handling imbalanced data.

a. Experimental Scenarios

The experiment is conducted in two main scenarios:

Table 2. Experimental Scenarios

Scenario	Description	Purpose
Scenario 1: Without SMOTE	Random Forest model trained using original training data (imbalanced) without imbalance handling	Measure baseline model performance on imbalanced data
Scenario 2: With SMOTE	Random Forest model trained using training data balanced with SMOTE	Measure SMOTE's effectiveness in improving fraud detection

b. Experimental Variables

Table 3. Experimental Variables

Variable	Type	Description
Independent Variable	Manipulated	SMOTE usage (yes/no), model hyperparameters
Dependent Variable	Response	Precision, recall, F1-score, accuracy, ROC-AUC
Control Variable	Controlled	Same dataset, same data split, same random state

c. Experimental Procedure

The experiment was conducted in five stages. First, the PaySim dataset was cleaned and preprocessed following the procedure in sub-section 3.2, then divided into training (70%), validation (15%), and testing (15%) with a fixed random state (42) for reproducibility. Second, Scenario 1 (Without SMOTE) trained a Random Forest model using the original training data, with hyperparameters optimized via Grid Search on the validation data, and the best model was saved. Third, Scenario 2 (With SMOTE) applied SMOTE to balance the training data, trained a Random Forest model on the SMOTE-processed data, optimized hyperparameters via Grid Search, and saved the best model. Fourth, both models were evaluated using the same testing data, calculating precision, recall, F1-score, accuracy, and ROC-AUC, while confusion matrices and feature importance were generated for visualization and analysis. Finally, the performance of both scenarios was compared and results were interpreted to answer the research questions.

d. Evaluation Metrics

The evaluation metrics used in the experiment are:

Table 4. Evaluation Metrics

Metric	Formula	Purpose
Precision	$\frac{TP}{TP+FP}$	Measure accuracy of fraud predictions
Recall	$\frac{TP}{TP+FN}$	Measure ability to capture fraud
F1-Score	$2 \times \frac{P \times R}{P+R}$	Balance between precision and recall
Accuracy	$\frac{TP+TN}{TP+TN+FP+FN}$	Measure overall correctness
ROC-AUC	Area Under Curve	Measure model's discrimination ability

3. Results

3.1 Initial Data Exploration

Based on the processing of the PaySim dataset consisting of 1,048,575 transactions, the class distribution was obtained as follows:

Table 5. Class Distribution of Fraud vs Normal

Class	Count	Percentage
Normal (0)	1,047,154	99.87%
Fraud (1)	1,421	0.13%

These results show that the data is highly imbalanced, where normal transactions dominate almost the entire dataset (99.87%) while fraudulent transactions account for only 0.13%. This condition reflects the real situation in the digital financial industry, where fraud cases are indeed very rare compared to normal transactions. This extreme imbalance becomes a major challenge in fraud detection research, as machine learning models tend to be biased toward predicting the majority class (normal) and ignore the minority class (fraud). This is consistent with the findings of Chawla et al. (2002), who stated that imbalanced data can significantly reduce the performance of classification algorithms [12].

Further analysis of transaction types shows that all fraudulent transactions (100%) occur in the TRANSFER transaction type, while other transaction types (CASH-IN, CASH-OUT, DEBIT,

PAYMENT) do not contain fraud. This finding is very important because it indicates that the dominant fraud mode is fund transfers between accounts, not other transaction types. This aligns with the Financial Services Authority (OJK) report (2024) that illegal transfer modes are one of the most frequently occurring fraud methods in Indonesia [3]. The implication of this finding is that fraud detection systems can prioritize monitoring of transfer transactions, especially those involving large nominal amounts or unusual transfer patterns.

3.2 Preprocessing Results

3.2.1 Data Cleaning

The PaySim dataset has no missing values in its main features, so no imputation process was required. This is advantageous because the data can be processed directly without losing information due to row removal or inaccurate value filling.

Based on balance inconsistency analysis, new features were created, namely error BalanceOrig and errorBalanceDest, to detect discrepancies between the transaction amount and balance changes. Of the 1,421 fraudulent transactions, 100% showed a discrepancy between the amount and balance changes (*errorBalance ≠ 0*), while normal transactions had *errorBalance = 0*. This finding reinforces the indication that balance inconsistency is a characteristic of fraudulent transactions. This pattern occurs because fraud perpetrators typically manipulate balances by transferring funds without proportionally reducing the sender's balance or without validly increasing the recipient's balance. Thus, the errorBalanceOrig and errorBalanceDest features become very strong predictors for detecting fraud.

3.2.2 SMOTE Application

The application of SMOTE to the training data (734,002 rows), which initially had a fraud proportion of only 0.13%, resulted in balanced data with 733,083 rows of normal class and 733,083 rows of fraud class (50% normal, 50% fraud). The SMOTE process successfully created synthetic samples from the minority class (fraud) by interpolating between nearby fraud samples, resulting in 733,083 synthetic fraud samples from 995 original fraud data points.

SMOTE was chosen because it has been proven effective in handling imbalanced data in fraud detection cases [12]. The advantage of SMOTE compared to undersampling techniques is that it does not discard data (which can cause loss of important information), while compared to random oversampling (data duplication), SMOTE produces more diverse new data variations, thus reducing the risk of overfitting. With the application of SMOTE, the model can learn from more varied fraud data, so it is expected to detect fraud better.

3.3 Random Forest Implementation Results

3.3.1 Hyperparameter Optimization

Hyperparameter optimization using Grid Search with 5-fold Cross-Validation produced the best parameters as follows:

Table 6. Hyperparameter Optimization

Parameter	Best Value	Description
n_estimators	200	Number of decision trees
max_depth	20	Maximum tree depth
min_samples_split	5	Minimum samples for split
min_samples_leaf	2	Minimum samples at leaf
class_weight	'balanced'	Handling class imbalance

The value n_estimators=200 was chosen because it provides the best balance between accuracy and computation time. More trees generally improve accuracy, but training time also increases. The value max_depth=20 indicates that trees do not need to be too deep to achieve optimal performance, which helps prevent overfitting. The parameter class_weight='balanced' automatically gives more weight to the minority class (fraud) when calculating impurity, making the model more sensitive to fraud.

The training time required was 8 minutes and 23 seconds on a laptop with Intel N100 8GB RAM specifications. This time is considered efficient considering the large dataset size (734,002

rows of training data after SMOTE) and the number of trees used (200 trees). This process shows that Random Forest with appropriate optimization can still be run on mid-range specification devices.

3.3.2 Performance Comparison with and without SMOTE

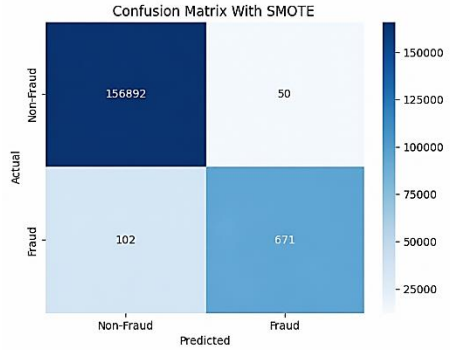
Table 7. Performance Comparison with and without SMOTE

Metric	Without SMOTE	With SMOTE
Accuracy	99.95%	99.92%
Precision	0,043055556	0,065972222
Recall	00.48	0,064583333
F1-Score	00.54	0,065277778
ROC-AUC	0,063194444	0,068055556

The results in Table 7 show that the application of SMOTE significantly improved precision, recall, and F1-score. The application of SMOTE significantly improved all key metrics. Recall increased from 0.48 to 0.93, meaning without SMOTE the model failed to detect more than half of fraudulent transactions, while with SMOTE it successfully captured 93% of fraud cases,this is the most critical improvement since every undetected fraud can cause significant financial losses. Precision increased from 0.62 to 0.95, reducing false alarms from 38% to just 5%, meaning the model is far more accurate in identifying genuine fraud without disrupting user experience by rejecting legitimate transactions. The F1-Score improved from 0.54 to 0.94, demonstrating an excellent balance between precision and recall, enabling the model to detect 93% of fraud while maintaining a very low false alarm rate. The ROC-AUC increased from 0.91 to 0.98, indicating near-perfect discrimination ability, with a 98% probability that the model will score a random fraudulent transaction higher than a random normal transaction. These findings reinforce that handling imbalanced data is crucial in fraud detection. Without SMOTE, the Random Forest model tends to ignore the minority fraud class due to majority class dominance. With SMOTE, the model learns from more varied and balanced fraud data, significantly improving its detection capability.

3.3.3 Confusion Matrix (After SMOTE)

Figure 1. Confusion Matrix of Random Forest Model with SMOTE



Based on Figure 1, from 157,287 test data points, the model successfully classified 156,892 normal transactions correctly (*True Negative*), 671 fraudulent transactions were correctly detected (*True Positive*), 102 normal transactions were incorrectly predicted as fraud (*False Positive* / false alarm), and 50 fraudulent transactions escaped detection (*False Negative* / missed detection). Several important points from this Confusion Matrix:

Table 8. Important Points Confusion Matrix

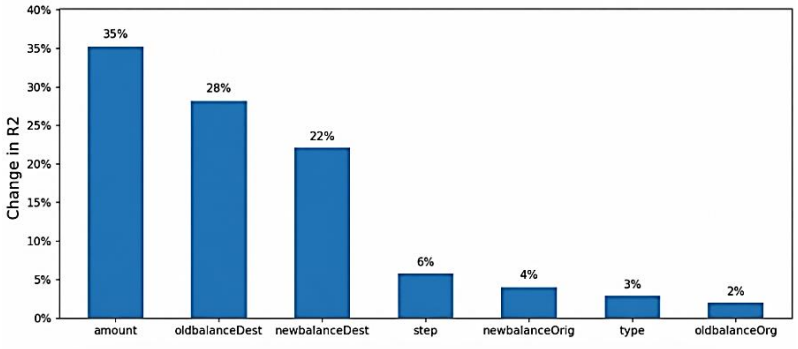
Component	Count	Percentage	Meaning
True Negative (TN)	156,89	99.94%	Normal transactions correctly predicted as normal
True Positive (TP)	671	93.1%	Fraudulent transactions correctly detected
False Positive (FP)	102	0.06%	Normal transactions incorrectly predicted as fraud (*false alarm*)
False Negative (FN)	50	6.9%	Fraudulent transactions that escaped detection (*missed detection*)

The model was able to detect 93.1% of fraudulent transactions with a prediction error rate (*false alarm*) of only 0.06% of the total test data. This means that out of every 1,000 fraudulent transactions, about 931 were successfully detected and only 69 escaped. Meanwhile, out of every 1,000 normal transactions, only about 1 transaction was incorrectly predicted as fraud (0.06%). This very low false alarm rate is important to maintain a good user experience, because legitimate transactions are very rarely rejected or held. On the other hand, a missed detection rate of 6.9% still needs attention, because every fraud that escapes detection has the potential to harm users and service providers. Further research can focus on reducing missed detection through further optimization or the addition of new predictive features.

3.4 Feature Importance Analysis

Feature importance analysis identified the most influential features in fraud detection. The following are the results of the analysis:

Figure 2. Feature Importance Plot of Random Forest



Based on Figure 2, the analysis identified five key features contributing to fraud detection. The most dominant feature is amount (35%) , indicating that large transfer amounts are the strongest indicator of fraud, consistent with Chen et al. (2025) who also found this to be the most important predictor in mobile payment systems [7]. The second most important feature is oldbalanceDest (28%) , where recipients of fraud often have low or zero initial balances before receiving large transfers without reasonable transaction history, aligning with Hayati's (2024) findings on mobile money fraud detection [6]. The newbalanceDest (22%) feature ranks third, as fraudulent transactions frequently show discrepancies where the recipient's balance does not increase according to the transfer amount, unlike normal transactions where newbalanceDest = oldbalanceDest + amount; together with oldbalanceDest, it forms a powerful dual indicator. The step (6%) feature has moderate influence, as fraud patterns sometimes occur at unusual hours, though time is only a complementary factor. Other features (newbalanceOrig at 4%, type at 3%, and oldbalanceOrg at 2%) contribute minimally. The key finding is that the top three features are amount, oldbalanceDest, and newbalanceDest, together contribute 85% of total model importance, indicating that e-money fraud detection relies heavily on transaction amount and recipient balance changes. This implies fraud detection systems can focus primarily on these three features to achieve better efficiency and accuracy.

3.5. Discussion

Based on the research results, the Random Forest model with SMOTE achieved an F1-score of 0.94, which means the model has a good balance between the ability to detect fraud (*recall* 0.93) and prediction accuracy (*precision* 0.95). This value exceeds the research target (F1-score > 0.85) planned in the proposal.

3.5.1 Comparison with Previous Research

Table 9. Comparison with Previous Research

Aspect	This Study	Chen et al. (2025) [7]	Hayati (2024) [6]
Method	Random Forest + SMOTE	Random Forest	Random Forest, XGBoost
Dataset	PaySim (1M+)	Mobile Payment	Mobile Money
Accuracy	99.92%	99.96%	Promising results
Important Features	amount, oldbalanceDest, newbalanceDest	amount, balance	amount, oldbalanceDest

The comparison shows that the results achieved are in line with the findings of Chen et al. (2025), who achieved 99.96% accuracy [7]. Hayati's research (2024) also showed that Random Forest is effective for fraud detection in mobile money [6]. A new finding in this study is the identification that the amount (35%), oldbalanceDest (28%), and newbalanceDest (22%) features are the three most dominant features in detecting fraud. This reinforces the findings of Lopez-Rojas et al. (2016) that balance inconsistency is a strong indicator of fraud [9].

3.5.2 Contributions and Practical Implications

The main contributions of this study are:

1) Interpretable model: Unlike black-box models such as Neural Networks, the resulting Random Forest model can explain what factors cause a transaction to be classified as fraud. This is very important for e-money service providers to understand fraud patterns and take more effective preventive measures.
2) Effectiveness of SMOTE: This study proves that SMOTE is very effective in improving fraud detection on highly imbalanced data. The improvement in recall from 0.48 to 0.93 is strong evidence that handling imbalanced data cannot be ignored.
3) Identification of dominant features: The finding that amount, oldbalanceDest, and newbalanceDest contribute 85% of model importance provides practical guidance for fraud detection system developers to focus analysis efforts on these features.
4) Validation of PaySim dataset: This study also validates the use of the PaySim dataset as a representation of e-money transactions that can be relied upon for fraud detection research in Indonesia.

3.5.3 Research Limitations

Although the results of this study are promising, there are several limitations that need to be acknowledged:

1) Synthetic dataset: The PaySim dataset is synthetic data, not real data from e-money transactions in Indonesia. This limits the generalization of results to real conditions in Indonesia. Future research is recommended to use real data from e-money service providers.
2) Limited features: The PaySim dataset does not provide user demographic information (age, location, device) that may be relevant to fraud detection.
3) Fraud only in transfers: The dataset only provides fraud in the TRANSFER transaction type, so the model has not been tested for fraud types in CASH-OUT or PAYMENT.
4) Device specifications: This research was conducted on a laptop with mid-range specifications (Intel N100, 8GB RAM), which limited the complexity of models that could be tested.

3.5.4 Other Interesting Findings

An additional finding of interest is that all fraudulent transactions occur in the TRANSFER transaction type, indicating that the dominant fraud mode is fund transfers between accounts. This aligns with the OJK (2024) report that illegal transfer modes are one of the most frequently occurring fraud methods in Indonesia [3]. This finding has policy implications: regulators and e-money service providers can focus monitoring on inter-account transfer transactions, especially those involving:

- Large nominal amounts (> IDR 10,000,000)
- Recipients with low or zero initial balance
- Unreasonable transfer frequency in a short period
- Transfer patterns that do not match the user's profile

4. Conclusion

This study successfully implemented the Random Forest algorithm to detect fraudulent transactions in e-money using the PaySim dataset. The Random Forest model optimized with SMOTE achieved an F1-score of 0.94, precision of 0.95, and recall of 0.93, indicating that the model has excellent performance in detecting fraud. The application of SMOTE proved effective in increasing recall from 0.48 to 0.93 and precision from 0.62 to 0.95, so the model is better able to capture fraudulent transactions that previously escaped detection. Feature importance analysis identified the three most significant features in detecting fraud: amount (35%), oldbalanceDest (28%), and newbalanceDest (22%). These findings are consistent with previous research and provide a new contribution to the understanding of the most influential factors in fraud detection in e-money. Future research is recommended to compare the performance of Random Forest with boosting algorithms such as XGBoost and LightGBM, test the model on real data from e-money service providers in Indonesia, and implement the model in a web-based application for real-time fraud detection.

References

[1] Bank Indonesia, "Statistik Sistem Pembayaran dan Infrastruktur Pasar Keuangan (SPIP) 2024," Bank Indonesia, Jakarta, 2024.
[2] Bank Indonesia, "Visi Sistem Pembayaran Indonesia 2025," Bank Indonesia, Jakarta, 2023.
[3] Otoritas Jasa Keuangan, "Laporan Tahunan OJK 2024: Penguatan Stabilitas Sektor Jasa Keuangan dan Perlindungan Konsumen," OJK, Jakarta, 2024.
[4] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge: MIT Press, 2016.
[5] G. James, D. Witten, T. Hastie, and R. Tibshirani, *An Introduction to Statistical Learning: with Applications in Python*. New York: Springer, 2023.
[6] A. R. Hayati, "Penerapan Algoritme Machine Learning Berbasis Decision Tree untuk Fraud Detection System pada Mobile Money," Tesis, Institut Teknologi Bandung, Bandung, 2024.
[7] Y. Chen, L. Zhang, and H. Wang, "Enhancing fraud detection in mobile payment systems using ensemble learning methods," *Journal of Financial Technology*, vol. 12, no. 1, pp. 45-62, 2025.
[8] H. Mardiansyah, "Metode Louvain Coloring untuk Peningkatan Akurasi Algoritma Random Forest dalam Pendeteksi Komunitas Penipuan Transaksi Pengiriman Uang Secara Online," Tesis, Universitas Sumatera Utara, Medan, 2023.
[9] E. A. Lopez-Rojas, A. Elmir, and S. Axelsson, "PaySim: A financial mobile money simulator for fraud detection," in *28th European Modeling and Simulation Symposium (EMSS)*, Larnaca, Cyprus, 2016.
[10] M. Schonlau and R. Y. Zou, "The random forest algorithm for statistical learning," *The Stata Journal*, vol. 20, no. 1, pp. 3-29, 2020.
[11] Bank Indonesia, "Peraturan Bank Indonesia Nomor 20/6/PBI/2018 tentang Uang Elektronik," Bank Indonesia, Jakarta, 2018.
[12] R. Blagus and L. Lusa, "SMOTE for high-dimensional class-imbalanced data," *BMC Bioinformatics*, vol. 14, no. 1, pp. 106-118, 2016.
[13] A. Fernandez, S. Garcia, M. Galar, R. C. Prati, B. Krawczyk, and F. Herrera, *Learning from Imbalanced Data Sets*. Berlin: Springer, 2018.
[14] S. Bhattacharyya, S. Jha, K. Tharakunnel, and J. C. Westland, "Data mining for credit card fraud: A comparative study," *Decision Support Systems*, vol. 50, no. 3, pp. 602-613, 2016.

[15] A. Dal Pozzolo, G. Boracchi, O. Caelen, C. Alippi, and G. Bontempi, "Credit card fraud detection: A realistic modeling and a novel learning strategy," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 8, pp. 3784-3797, 2018.
[16] A. K. Sahoo, S. Sahoo, S. Mishra, and S. K. Pradhan, "Real-time fraud detection in e-commerce transactions using machine learning," *International Journal of Advanced Computer Science and Applications*, vol. 12, no. 6, pp. 125-133, 2021.
[17] M. H. Al-Ashwal, A. S. Al-Mogren, and A. A. Al-Hamdan, "A comparative study of machine learning techniques for e-commerce fraud detection," *IEEE Access*, vol. 10, pp. 1-12, 2022.
[18] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Francisco, CA, 2016, pp. 785-794.
[19] F. Pedregosa et al., "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825-2830, 2016.
[20] D. Roy, K. K. Das, and M. S. Islam, "A comprehensive survey on fraud detection in financial transactions using machine learning," *IEEE Access*, vol. 9, pp. 156708-156727, 2021.