Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 00 (2024) 000–000

Procedia Computer Science

www.elsevier.com/locate/procedia

9th International Conference on Computer Science and Computational Intelligence 2024 (ICCSCI 2024)

Comparative Analysis Using Random Forest and K-Nearest Neighbors for Money Laundering Detection

First Author[a], Second Author[b], Third Author[a,b,*]

[a]*First affiliation, Address, City and Postcode, Country*
[b]*Second affiliation, Address, City and Postcode, Country*

Abstract

Detecting money laundering activities within financial transactions is crucial for maintaining the integrity and security of financial systems. This study looks at the performance of Random Forest and K-Nearest Neighbors (KNN) algorithms for detecting fraudulent transactions. The study evaluates these models in four configurations: not standardized and not hypertuned, hypertuned but not standardized, standardized but not hypertuned, and both standardized and hypertuned. Performance metrics such as accuracy, precision, recall, and F1-score are used to evaluate the models. The results show that Random Forest models consistently outperform KNN models, with great accuracy and robustness due to their ensemble method of learning. The Random Forest model scored the highest accuracy of 99.17%, followed by the KNN model at 99.07%. This study emphasizes the importance of data preprocessing and hyperparameter tuning in improving model performance. Our findings indicate that, while Random Forest is extremely effective for fraud detection, KNN with appropriate tuning can also be a viable alternative. Future research concerns include integrating additional features and researching real-time implementation of these models for practical applications in financial systems.

Keywords: Anti-Money Laundering; Classification Algorithm; Data Standardization; Financial Transactions; Fraud Detections; Hyperparameter Tuning; K-Nearest Neighbors; Machine Learning; Model Evaluation; Money Laundering; Money Laundering Detection; Prediction; Random Forest

* Corresponding author. Tel.: +0-000-000-0000 ; fax: +0-000-000-0000 .
E-mail address: author@institute.xxx

1. Introduction

The issue of money laundering remains a significant challenge in the financial sector, posing a major threat to both national and international economies [1]. It is estimated that approximately 5% of the world's Gross Domestic Product (GDP) is laundered through illegal activities each year, emphasizing the significance of systems to detect and prevent this [2]. Banks and other financial institutions play an important role in combatting money laundering, especially as they are obligated to follow tight rules when updating their ways of combating money laundering operations [3]. Advances in transaction methods, along with increasingly sophisticated laundering techniques, make this task more difficult [4]. Traditional Anti-Money Laundering (AML) systems often struggle to find suspicious activities without also generating a lot of false alarms [5].

Although banks have implemented advanced technology solutions, they continue to have difficulties in identifying true money laundering transactions from lawful ones without causing disruptions [6–8]. In this circumstance, Machine Learning (ML) provides promising solutions by allowing the analysis of large datasets to discover suspicious patterns indicative of money laundering [9]. Among the several machine learning algorithms, Random Forest and K-Nearest Neighbor (KNN) stand out for their applicability in classification tasks, including fraud detection. Random Forest, known for its high accuracy and robustness against overfitting, operates by constructing multiple decision trees and voting on the most common output class [10]. On the other hand, KNN, a simpler yet effective algorithm, classifies new cases through the majority vote of the 'k' nearest points in the feature space [11].

This research performs a comparative analysis of these two algorithms, specifically focusing on their accuracy and performance metrics in the domain of AML. While extensive research has explored the individual merits of these algorithms, direct comparisons under identical experimental conditions remain scarce [12]. By systematically comparing KNN and Random Forest, this paper aims to elucidate which algorithm not only provides higher accuracy but also better performance metrics, thereby aiding financial institutions in their efforts to optimize AML processes.

The use of machine learning in detecting money laundering has been extensively explored in various research studies, highlighting the effectiveness of different algorithms in addressing this pervasive issue [13,14]. Several studies have demonstrated the application of Random Forest and KNN among other algorithms, indicating a broad consensus on the utility of machine learning models for improving AML systems.

For instance, the use of multiple machine learning models, including Random Forest, is effective for detecting suspicious money laundering transactions within banks [15]. Their work emphasized the capability of Random Forest to handle large datasets and feature sets efficiently, which is critical in the banking sector where vast amounts of data are processed.

Similarly, neural networks were integrated with traditional machine learning models to analyze user behavior in financial systems, also highlighting the potential of including simpler yet effective models like KNN for detecting anomalies [16]. This approach underscores the importance of adaptability in AML systems, leveraging both complex and straightforward models to enhance detection rates.

In another relevant study, the application of unsupervised learning models, including KNN, was explored to identify hidden patterns in financial data that could indicate money laundering activities [17]. This research exemplifies the versatility of KNN in clustering and anomaly detection, which are crucial for identifying subtle patterns indicative of illicit activities [18].

Furthermore, a comparative analysis demonstrated the strengths and weaknesses of various machine learning models, including Random Forest and KNN, in detecting fraudulent financial statements [19]. This study not only explored the individual capabilities of each model but also their comparative effectiveness, providing insights into their practical applications in real-world scenarios [20].

These studies demonstrate the evolving environment of machine learning applications in financial security, particularly in money laundering detection. They provide a solid foundation for this research paper, which aims to delve deeper into the comparative effectiveness of Random Forest and KNN in this domain, offering new insights and contributing to the broader field of financial crime prevention.

2. Method

2.1. Dataset

The dataset used in this study is sourced from Kaggle and consists of simulated financial transactions aimed at predicting fraudulent activities. It includes a total of 6,362,620 transactions and 11 features, as described in the table below:

Table 1. Description of Dataset Features.

Features	Description
step	Represents a unit of time in the simulation, where each step corresponds to an hour (total 744 steps).
type	Indicates the type of transaction (CASH-IN, CASH-OUT, DEBIT, PAYMENT, TRANSFER).
amount	The amount of the transaction in local currency.
nameOrig	The identifier of the customer initiating the transaction.
oldbalanceOrg	The initial balance of the originating account before the transaction.
newbalanceOrig	The new balance of the originating account after the transaction.
nameDest	The identifier of the customer receiving the transaction.
oldbalanceDest	The initial balance of the destination account before the transaction (missing for merchants).
newbalanceDest	The new balance of the destination account after the transaction (missing for merchants).
isFraud	Indicates whether the transaction is fraudulent (1) or not (0).
isFlaggedFraud	Indicates transactions flagged as potential fraud based on business rules (e.g., large transactions).

The preprocessing of the dataset involved several critical steps to ensure the data was clean, balanced, and ready for model training. Initially, the dataset was loaded into a pandas DataFrame, which allowed for easy manipulation and analysis. An Exploratory Data Analysis (EDA) was conducted to understand the structure and distribution of the data. This included displaying the first few rows, summarizing data types, and checking for missing values and duplicates. It was found that the dataset contained no missing values or duplicates, and the features were categorized into categorical and numerical columns.

To prepare the categorical data for machine learning algorithms, the type variable was converted into numerical values using label encoding. Additionally, feature engineering was performed to create two new features: origBalanceChange, which represents the change in the balance of the originating account, and destBalanceChange, which represents the change in the balance of the destination account. These new features provided additional insights and improved the model's ability to detect fraudulent activities.

Given the significant class imbalance in the dataset, with fraudulent transactions being much fewer than non-fraudulent ones, Random UnderSampling was employed to balance the classes. This step was crucial to prevent the model from becoming biased towards the majority class. The dataset was then split into training and testing sets to evaluate the model's performance.

For algorithms like K-Nearest Neighbors (KNN), which are sensitive to the scale of data, numerical features were standardized to ensure they were on a similar scale. This standardization process involved transforming the features to have a mean of zero and a standard deviation of one using StandardScaler. Lastly, hyperparameter tuning was performed using Grid Search to find the optimal parameters for both Random Forest and KNN models. This process involved testing different combinations of hyperparameters to identify the ones that provided the best performance. By following these comprehensive preprocessing steps, the dataset was well-prepared for model training, ensuring that the features were clean, balanced, and standardized, thereby enhancing the performance and reliability of the machine learning models used for fraud detection.

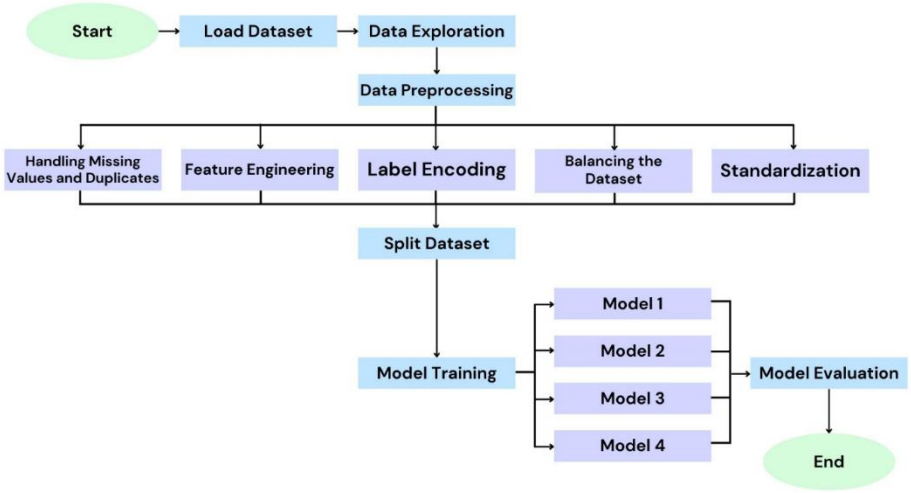

Fig. 1. Illustrates the data preprocessing and modelling workflow.

For algorithms like KNN, which are sensitive to the scale of data, numerical features were standardized to ensure they were on a similar scale. This standardization process involved transforming the features to have a mean of zero and a standard deviation of one using StandardScaler. Lastly, hyperparameter tuning was performed using Grid Search to find the optimal parameters for both Random Forest and KNN models. This process involved testing different combinations of hyperparameters to identify the ones that provided the best performance. By following these comprehensive preprocessing steps, the dataset was well-prepared for model training, ensuring that the features were clean, balanced, and standardized, thereby enhancing the performance and reliability of the machine learning models used for fraud detection.

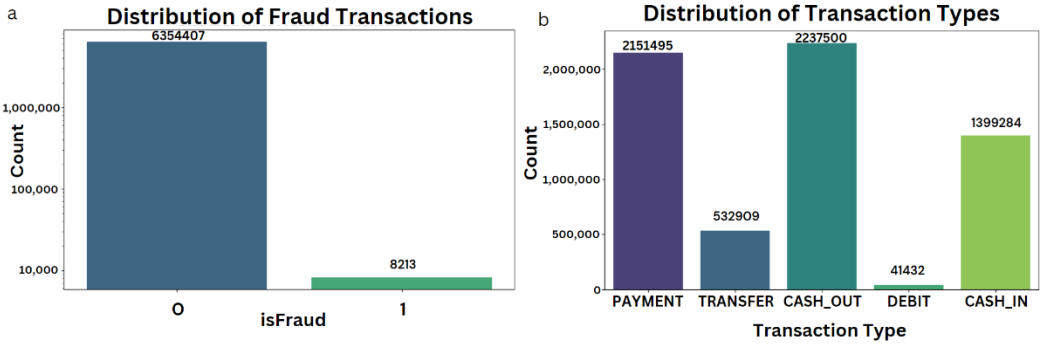

Fig. 2. (a) Distribution of Fraud Transactions; (b) Distribution of Transaction Types.

The Distribution of Fraud Transactions bar chart shows a significant class imbalance, with 6,354,407 non-fraudulent transactions compared to only 8,213 fraudulent transactions, emphasizing the need for techniques like undersampling to balance the classes. The Distribution of Transaction Types bar chart reveals that CASH-OUT transactions are the most frequent (2,237,500), followed by PAYMENT (2,151,495), CASH-IN (1,399,284), TRANSFER (532,909), and DEBIT (41,432), indicating diverse transaction behaviors in the dataset.

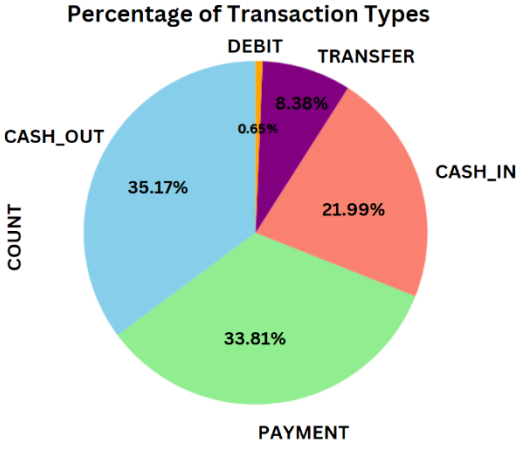

Fig. 3. Percentage of Transaction Types.

The Percentage of Transaction Types pie chart further breaks down the composition of transaction types, with CASH-OUT accounting for 35.17%, PAYMENT for 33.81%, CASH-IN for 21.99%, TRANSFER for 8.38%, and DEBIT for 0.65%. This visualization helps in understanding the proportional distribution of each transaction type.

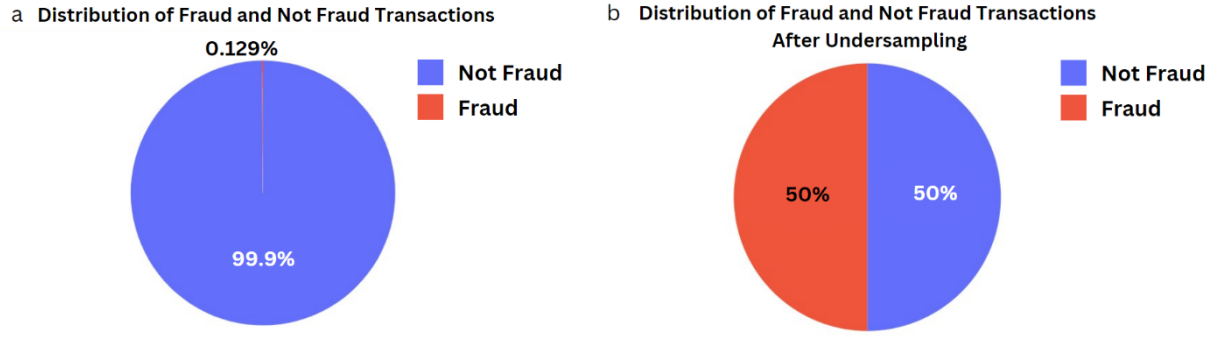

Fig. 4. (a) Distribution of Fraud and Not Fraud Transactions; (b) Distribution of Fraud and Not Fraud Transactions After Undersampling.

The Distribution of Fraud and Not Fraud Transactions pie charts illustrate the significant class imbalance in the dataset. Before undersampling, fraudulent transactions represent only 0.129% of the total transactions, while non-fraudulent transactions make up 99.9%. After applying Random UnderSampling to balance the classes, both fraudulent and non-fraudulent transactions each account for 50% of the dataset. This balancing is crucial for training models to prevent bias toward the majority class.

2.2. Algorithm Selection and Configurations

Random Forest and KNN algorithms were chosen due to their distinct characteristics and suitability for classification tasks, particularly in the context of financial fraud detection. Random Forest is an ensemble learning

method that constructs multiple decision trees during training and outputs the mode of the classes (classification) or mean prediction (regression) of the individual trees. It is known for its high accuracy, robustness against overfitting, and ability to handle large datasets with high dimensionality. The ensemble nature of Random Forest enhances its predictive performance by reducing variance and improving generalization.

KNN is an instance-based learning algorithm that classifies a data point based on the majority class among its k-nearest neighbors in the feature space. It is appreciated for its interpretability and ease of implementation. KNN is particularly useful for detecting patterns and anomalies in data, making it suitable for fraud detection tasks. However, its performance can be sensitive to the choice of k and the distance metric used.

For this study, four different configurations of both the Random Forest and KNN models are used to comprehensively evaluate their performance in detecting money laundering activities:

Table 2. Experiment details

Model	Standardized	Hypertuned
1	No	No
2	No	Yes
3	Yes	No
4	Yes	Yes

In the first configuration, the models were trained using the default algorithm parameters without any standardization of the feature values. This setup serves as a baseline to compare the effects of standardization and hyperparameter tuning. By not applying any transformations or parameter adjustments, this configuration provides a reference point for understanding the inherent capabilities of the algorithms.

In the second configuration, hyperparameter tuning was performed using Grid Search to determine the optimal parameters for each algorithm, without standardizing the feature values. Hyperparameter tuning involves systematically searching through a predefined set of hyperparameters to find the combination that yields the best performance. This configuration aims to enhance the model's effectiveness by optimizing its internal parameters while maintaining the original scale of the data.

Table 3. Hyperparameters for Random Forest and K-Nearest Neighbors

Algorithm	Hyperparameters	Values
Random Forest	n_estimators	[100, 200, 500]
	max_depth	[2, 4, 8]
	min_samples_split	[2, 5, 10]
K-Nearest Neighbors	n_neighbors	[3, 5, 7, 9, 11]
	weights	['uniform', 'distance']
	algorithm	['auto', 'ball_tree', 'kd_tree', 'brute']

In the third configuration, the feature values were standardized to have a mean of zero and a standard deviation of one. Standardization helps ensure that each feature contributes equally to the model, preventing features with larger scales from disproportionately influencing the results. The models were then trained using the default algorithm parameters. This configuration highlights the impact of feature scaling on the model's performance without the influence of hyperparameter tuning.

In the fourth configuration, both standardization and hyperparameter tuning were applied to optimize the model's performance. By applying both techniques, this setup aims to maximize the model's effectiveness by ensuring that all features are on a comparable scale and that the model parameters are fine-tuned for optimal performance. This comprehensive approach provides insights into the combined benefits of data preprocessing and parameter optimization.

2.3. Evaluation Metrics

To assess the effectiveness of the Random Forest and KNN models in detecting money laundering activities, a comprehensive suite of performance metrics is employed. These include accuracy, precision, recall, and F1-score, which are derived from the confusion matrix. The confusion matrix consists of True Positives (TP), False Positives (FP), True Negatives (TN), and False Negatives (FN).

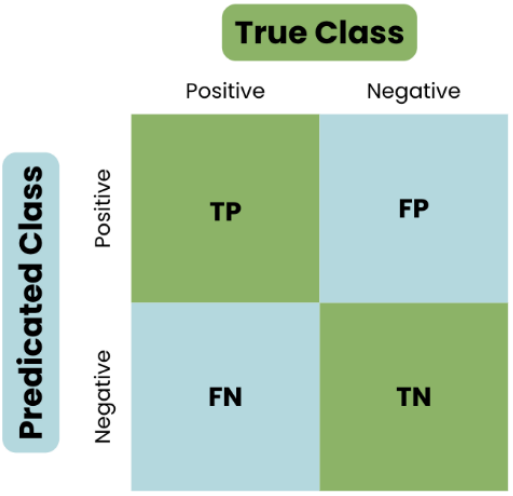

Fig. 4. Confusion Matrix.

Accuracy is calculated as the ratio of correctly classified instances (both true positives and true negatives) to the total instances:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

Precision is the ratio of true positive predictions to the total predicted positives, providing insight into the model's ability to avoid false positives:

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

Recall is the ratio of true positive predictions to the total actual positives, indicating the model's ability to identify all relevant instances:

$$Recall = \frac{TP}{TP + FN} \tag{3}$$

F1-Score is the harmonic mean of precision and recall, providing a single measure that balances both metrics, especially useful when the cost of false positives and false negatives are both high:

$$F1 - Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{4}$$

3. Result

A comparative performance analysis was conducted for the Random Forest and KNN algorithms under different configurations. The performance metrics used for evaluation include accuracy, precision, recall, and F1-score. The models were evaluated under four configurations: not standardized and not hypertuned, hypertuned but not standardized, standardized but not hypertuned, and both standardized and hypertuned.

3.1. Random Forest

The performance of the Random Forest models is summarized in Table 4. The results show that all configurations of the Random Forest algorithm achieved high scores across all performance metrics.

Table 4. Random forest performance metrics

Name	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Model 1 (not standardized, not hypertuned)	99.17	99.00	99.00	99.00
Model 2 (not standardized, hypertuned)	98.99	99.00	99.00	99.00
Model 3 (standardized, not hypertuned)	99.15	99.00	99.00	99.00
Model 4 (standardized, hypertuned)	98.97	99.00	99.00	99.00

All configurations of the Random Forest algorithm demonstrated exceptionally high performance metrics, indicating the robustness and effectiveness of the algorithm for this task. The highest accuracy was achieved by Model 1, which was not standardized and not hypertuned, with an accuracy of 99.17%. Standardization slightly decreased the accuracy but maintained high precision, recall, and F1-score. Hyperparameter tuning did not significantly improve the performance, suggesting that the default parameters of Random Forest are already well-suited for this dataset. This can be attributed to the nature of decision trees, which are the building blocks of Random Forest. Decision trees are generally insensitive to the scale of the features because they are split based on the order of the feature values rather than their absolute values. Consequently, standardizing the features (scaling them to a common range) does not significantly impact the model's performance. The slight decrease in accuracy due to standardization might be caused by minor numerical instability introduced by transforming the feature values.

Moreover, the robustness of Random Forest to default parameters is due to its ensemble method, which constructs multiple decision trees using different subsets of the dataset and features. This ensemble approach mitigates overfitting and captures diverse patterns in the data, providing high accuracy and robustness even without extensive hyperparameter tuning. The default settings of Random Forest often strike a good balance between bias and variance, making them effective enough for a well-defined dataset like ours.

3.2. K-Nearest Neighbors

The performance of the KNN models is summarized in Table 5. The results indicate that the KNN algorithm's performance improved significantly with standardization and hyperparameter tuning.

Table 5. KNN performance metrics

Name	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Model 1 (not standardized, not hypertuned)	98.76	99.00	99.00	99.00
Model 2 (not standardized, hypertuned)	99.05	99.00	99.00	99.00
Model 3 (standardized, not hypertuned)	98.89	99.00	99.00	99.00
Model 4 (standardized, hypertuned)	99.07	99.00	99.00	99.00

The performance of KNN models showed significant improvements with standardization and hyperparameter tuning. Model 4, which combined standardization and hyperparameter tuning, achieved the highest performance

metrics, with an accuracy of 99.07%. This highlights the importance of preprocessing and parameter optimization for KNN. Unlike Random Forest, KNN's performance was more sensitive to data scaling and hyperparameter settings. This sensitivity is because KNN relies on calculating distances (typically Euclidean) between data points. If the features are not standardized, features with larger ranges can disproportionately influence the distance calculations, leading to biased predictions. Standardizing the features ensures that each feature contributes equally to the distance metric, thereby improving the model's performance.

Hyperparameter tuning is also crucial for KNN. The number of neighbors (n_neighbors), the weighting of the neighbors' contributions (weights), and the algorithm used to compute the nearest neighbors (algorithm) are critical hyperparameters. Tuning these hyperparameters helps find the optimal settings that balance bias and variance, resulting in improved accuracy and robustness.

3.3. Comparative Analysis

The comparative analysis between Random Forest and KNN reveals that Random Forest consistently outperformed KNN across all configurations. Random Forest models, even without hyperparameter tuning and standardization, achieved higher accuracy compared to KNN models. Both algorithms benefited from standardization, but Random Forest was less sensitive to these changes compared to KNN. Hyperparameter tuning provided marginal improvements for both algorithms, with KNN showing a more noticeable enhancement.

Standardization improved the performance of KNN significantly, highlighting the importance of scaling features for distance-based algorithms. In contrast, standardization had a minimal impact on Random Forest, suggesting that this algorithm is robust to the scale of input features. Hyperparameter tuning provided slight improvements for both algorithms, indicating that while tuning can enhance performance, the default settings of these algorithms are reasonably effective for this dataset.

In summary, the Random Forest algorithm demonstrated superior performance in detecting money laundering activities within financial transactions. While KNN also showed competitive results, particularly when standardized and hypertuned, Random Forest's ensemble learning approach made it more robust and reliable across various configurations.

4. Discussion

In this research, we analyzed the effectiveness of Random Forest and KNN algorithms for detecting money laundering activities in financial transactions. Evaluated under four configurations, the models were assessed using accuracy, precision, recall, and F1-score. Our findings showed that Random Forest consistently outperformed KNN, exhibiting high accuracy and robustness due to its ensemble learning approach. The standardized and hypertuned Random Forest model achieved the highest performance, highlighting the importance of preprocessing and parameter optimization. While KNN's performance was generally lower, standardized and hypertuned KNN models showed competitive results, demonstrating that KNN can be a reliable alternative with proper tuning.

This study significantly contributes to financial fraud detection by providing a detailed comparative analysis of these algorithms. It emphasizes the robustness of Random Forest for practical implementation in anti-money laundering systems and the potential of KNN for scenarios where simplicity and interpretability are crucial. Our findings highlight the critical roles of data preprocessing, standardization, and hyperparameter tuning in enhancing model performance.

Future research should explore the integration of additional features, the application of advanced algorithms like deep learning, and the real-time implementation of these models in financial systems. Investigating techniques for handling class imbalances, such as synthetic data generation, could further improve detection rates of rare fraudulent activities. Overall, this study underscores the importance of model selection and optimization in developing effective machine learning models for fraud detection, providing valuable insights for financial institutions to enhance their anti-money laundering processes.

References

[1] Alotibi J, Almutanni B, Alsubait T, Alhakami H, Baz A. Money Laundering Detection using Machine Learning and Deep Learning. International Journal of Advanced Computer Science and Applications 2022;13. https://doi.org/10.14569/IJACSA.2022.0131087.

[2] Patil NM, Dondekar SM, Rawale C V., Memane K V., Kamble L. Combating money laundering using artificial intelligence. International Journal of Advanced Research, Ideas, and Innovations in Technology 2020;6:609–12.

[3] Chen Z, Soliman WM, Nazir A, Shorfuzzaman M. Variational Autoencoders and Wasserstein Generative Adversarial Networks for Improving the Anti-Money Laundering Process. IEEE Access 2021;9:83762–85. https://doi.org/10.1109/ACCESS.2021.3086359.

[4] Savage D, Wang Q, Chou P, Zhang X, Yu X. Detection of money laundering groups using supervised learning in networks 2016.

[5] Bakry AN, Alsharkawy AS, Farag MS, Raslan KR. Automatic suppression of false positive alerts in anti-money laundering systems using machine learning. J Supercomput 2024;80:6264–84. https://doi.org/10.1007/s11227-023-05708-z.

[6] Lopez-Rojas EA, Axelsson S. Money Laundering Detection using Synthetic Data. Annual Workshop of the Swedish Artificial Intelligence Society, 2012.

[7] Canhoto AI. Leveraging machine learning in the global fight against money laundering and terrorism financing: An affordances perspective. J Bus Res 2021;131:441–52. https://doi.org/10.1016/j.jbusres.2020.10.012.

[8] Heidarinia N, Harounabadi A, Sadeghzadeh M. An Intelligent Anti-Money Laundering Method for Detecting Risky Users in the Banking Systems. Int J Comput Appl 2014;97:35–9. https://doi.org/10.5120/17141-7780.

[9] Alarab I, Prakoonwit S, Nacer MI. Comparative Analysis Using Supervised Learning Methods for Anti-Money Laundering in Bitcoin. Proceedings of the 2020 5th International Conference on Machine Learning Technologies, New York, NY, USA: ACM; 2020, p. 11–7. https://doi.org/10.1145/3409073.3409078.

[10] Sintayehu K, Seid H. Developing Anti Money Laundering Identification using Machine Learning Techniques. Irish Interdisciplinary Journal of Science & Research 2023;07:64–74. https://doi.org/10.46759/IIJSR.2023.7110.

[11] Hampo JAC, Nwokorie EC, Odii JN. A Web-Based kNN Money Laundering Detection System. European Journal of Theoretical and Applied Sciences 2023;1:277–88. https://doi.org/10.59324/ejtas.2023.1(4).27.

[12] Lokanan M. Predicting Money Laundering using Machine Learning and Artificial Neural Networks Algorithms in Banks 2022:1–27.

[13] Zhang Y, Trubey P. Machine Learning and Sampling Scheme: An Empirical Study of Money Laundering Detection. Comput Econ 2019;54:1043–63. https://doi.org/10.1007/s10614-018-9864-z.

[14] YANG Y, CHEN R, BAI X, CHEN D. Finance Fraud Detection With Neural Network. E3S Web of Conferences 2020;214:03005. https://doi.org/10.1051/e3sconf/202021403005.

[15] Jullum M, Løland A, Huseby RB, Ånonsen G, Lorentzen J. Detecting money laundering transactions with machine learning. Journal of Money Laundering Control 2020;23:173–86. https://doi.org/10.1108/JMLC-07-2019-0055.

[16] Rocha-Salazar J-J, Segovia-Vargas M-J, Camacho-Miñano M-M. Money laundering and terrorism financing detection using neural networks and an abnormality indicator. Expert Syst Appl 2021;169:114470. https://doi.org/10.1016/j.eswa.2020.114470.

[17] Ehab A, Shokry M, Rizka MA, Labib NM. COUNTER TERRORISM FINANCE BY DETECTING MONEY LAUNDERING HIDDEN NETWORKS USING UNSUPERVISED MACHINE LEARNING ALGORITHM. Proceedings of the International Conferences on ICT, Society and Human Beings (ICT 2020), Connected Smart Cities (CSC 2020) and Web Based Communities and Social Media (WBC 2020) 2020.

[18] Pettersson Ruiz E, Angelis J. Combating money laundering with machine learning – applicability of supervised-learning algorithms at cryptocurrency exchanges. Journal of Money Laundering Control 2022;25:766–78. https://doi.org/10.1108/JMLC-09-2021-0106.

[19] Zhongzhu L, Ye R, Ye R. Detecting Financial Statement Fraud with Interpretable Machine Learning 2021. https://doi.org/10.21203/rs.3.rs-640038/v1.

[20] Demetis DS. Fighting money laundering with technology: A case study of Bank X in the UK. Decis Support Syst 2018;105:96–107. https://doi.org/10.1016/j.dss.2017.11.005.