UDC 004.93
DOI: 10.15587/1729-4061.2025.327410

IDENTIFYING THE GRAPH-BASED TYPOLOGY FEATURES FOR MACHINE LEARNING MODELS IN FINANCIAL FRAUD DETECTION

Sabina Rakhmetulayeva
PhD, Professor*
Aliya Kulbayeva
Corresponding author
PhD Student**
E-mail: aakulbayeva@gmail.com
Aigerim Bolshibayeva
PhD, Assistant Professor**
Vassiliy Serbin
PhD, Associate Professor*
*Department of Cybersecurity, Information Processing and Storage
Satbayev University
Satbaev str., 22, Almaty, Republic of Kazakhstan, 050013
**Department of Information Systems
International IT University
Manas str., 34/1, Almaty, Republic of Kazakhstan, 050000

This article investigates fraud detection in financial transaction networks using machine learning and graph-based typologies. The object of the study is financial transaction data, analyzed to improve the accuracy and efficiency of identifying fraudulent activities. The problem addressed is the limited generalizability and low recall of traditional fraud detection models in complex, real-world settings.

To address this, a hybrid framework was developed that integrates Random Forests, neural networks, and graph-based typology indicators. Seven laundering typologies were extracted from a transaction graph – fan-in, fan-out, scatter-gather, gather-scatter, cycle, bipartite, and stacked bipartite – and used as additional features for classification. SMOTE was applied to correct class imbalance during training.

Experimental results show that adding typology features significantly improves model performance. The best results were obtained with Random Forest: 98.5% accuracy, 79.1% precision, 56.3% recall, and an F1-score of 65.7%. Adding typology-based flags raised recall by 9–11 percentage points compared to models without them. Graph patterns like fan-in and fan-out were detected in 3.5–5.1% of transactions, while more complex structures such as cycle and scatter-gather appeared less frequently but correlated more strongly with known fraud.

Unsupervised methods also showed promise: an autoencoder captured 60% of fraud cases among the top 2% anomalous transactions, while K-means identified 55% of fraud within flagged clusters. These methods proved useful for identifying emerging fraud types not yet labeled in training data.

The model is suitable for integration into financial security systems with minimal input requirements – account IDs, timestamps, and transaction amounts – alongside basic graph analytics. Its robustness across datasets suggests strong applicability across diverse financial institutions.

Keywords: financial fraud, transaction patterns, machine learning, graph analysis, typology detection, classification, anomaly detection

Received 20.02.2025
Received in revised form 18.04.2025
Accepted date 30.04.2025
Published date 25.06.2025

How to Cite: *Rakhmetulayeva, S., Kulbayeva, A., Bolshibayeva, A., Serbin, V. (2025). Identifying the graph-based typology features for machine learning models in financial fraud detection. Eastern-European Journal of Enterprise Technologies, 3 (9 (135)), 40–54. https://doi.org/10.15587/1729-4061.2025.327410*

1. Introduction

Financial fraud has become a widespread and rapidly evolving threat, causing substantial economic losses and undermining trust in financial institutions worldwide. Global reports estimate that financial fraud costs businesses and consumers billions of dollars annually [1]. The advancement of technology has not only improved financial services but also enabled fraudsters to exploit vulnerabilities in transaction systems. This growing sophistication of fraudulent schemes has outpaced many traditional detection methods, creating a pressing need for more adaptive and intelligent fraud detection mechanisms.

Reliable fraud detection is essential not only for securing financial assets but also for maintaining the stability of economic systems and the confidence of customers. Conventional rule-based systems have provided foundational approaches to fraud detection, but they struggle to remain effective in the face of increasingly voluminous, dynamic, and unstructured transaction data [2]. As digital transactions continue to grow exponentially in both scale and complexity, there is a clear shift from manual or rule-based approaches to data-driven and intelligent analytical models capable of real-time detection.

Modern fraud detection must contend with several core challenges: the high dimensionality and velocity of transaction data, the rarity and diversity of fraudulent events, and the continuous evolution of fraud strategies. These issues lead to high false-positive rates, delayed detection, and reduced scalability in traditional models [3]. Moreover, the inherent class imbalance – where fraudulent transactions make up only a small fraction of all records – poses difficulties in training effective machine learning algorithms [4].

Given the increasing importance of financial transaction integrity and the limitations of current detection methods, the topic of fraud detection using advanced analytical techniques remains highly relevant. Research in this area is not only timely but also essential for the development of secure and resilient financial systems. It provides practical value by addressing existing gaps in detection capability, particularly in real-time, high-volume environments.

Therefore, research on the development of advanced fraud detection systems, especially those leveraging machine learning and data analytics, is highly relevant to current scientific and practical challenges.

2. Literature review and problem statement

The paper [5] presents the FlowScope algorithm for detecting money-laundering chains in multi-party transaction graphs. It shows that the method reliably identifies cash-flow "streams," avoiding the density traps of classical GNNs. Yet unresolved problems remain – scaling the algorithm to multi-bank graphs with billions of edges and reducing dependence on manually tuned anomaly thresholds – both of which limit real-world adoption. The causes lie in O (V log V) graph-partition cost and the scarcity of labeled ground truth. Pre-aggregating "hub" accounts and employing streaming feature preprocessing could help, but that study has not addressed false-positive control in multi-currency settings. Hence, a hybrid streaming GNN (Graph Neural Network) + GBDT (Gradient Boosting Decision Tree) pipeline for cross-jurisdiction AML (Anti-Money Laundering) warrants investigation.

This research is driven by the need to solve three critical challenges. First, it aims to improve fraud detection efficiency by identifying subtle and evolving transaction patterns that traditional rule-based systems do not capture. Second, it focuses on improving adaptability by developing AI-driven models capable of detecting fraud in diverse financial environments, transaction types, and emerging fraud schemes, requiring continuous optimization and testing. Finally, scalability is a key goal, necessitating the creation of high-performance models that can process vast amounts of real-time transaction data while maintaining precision and minimizing false positives.

The ultimate objective of this research is to bridge the gap between advanced fraud detection methodologies and their practical implementation in financial security. By improving detection accuracy, reducing financial losses, and strengthening fraud prevention mechanisms, this study aims to have a significant impact in the field of financial risk management.

A way to overcome these difficulties can be found in the streaming feature pipeline proposed in paper [6], where the authors introduce GFP (Graph-based Feature Pipeline), which generates hundreds of subgraph-based features for streaming transaction graphs, raising F1 to 0.92 with gradient boosting on the IBM AML dataset. However, the explainability of these complex features remains open: domain experts cannot easily see which patterns trigger SAR (Suspicious Activity Report) flags. This limitation stems from the inherent difficulty of interpreting high-dimensional graph features and from the cost of real-time flow tracing. SHAP (Integrating Shapley Additive) explanations for graph features – partly attempted in FlowScope but lacking visual analytics – could mitigate the issue. An explainable AML framework that combines GFP with interactive graph UIs is therefore needed.

The paper [7] reports on reduced-egonet features and Isolation Forests for anomaly detection in bank-transaction graphs. This review of 37 experiments shows that cognitive "primes" increase auditors' attention to red flags. Yet translating such laboratory effects to digital transaction-graph environments remains unsolved, because auditing paper trails differs fundamentally from streaming AML. Research into gamified simulators linking audit heuristics to graph features is called for. It shows that egonet compression improves precision, yet domain transfer is still poor because egonet parameters depend on every bank's unique topology – a fundamental limitation that forces costly recalibration when the model is ported elsewhere.

Fraudulent activities often exhibit specific transaction patterns, such as rapid fund transfers across multiple accounts, unusual cash deposits, and high-risk geographical locations.

Recognizing indicators during data analysis aids in building a hypothesis. These indicators suggest the likelihood of certain activities occurring.

In paper [8] using "reduced-egonet" features plus random walks, the authors show Isolation Forest outperforming baselines on real and synthetic graphs. Yet these features transfer poorly between banks – their parameters depend heavily on network topology. Domain heterogeneity and the high cost of recalibration explain the limitation. Domain-invariant autoencoders may help, but they were not combined with egonet reduction. Research into domain-adaptive GNNs that fuse egonet and embedding signals is needed. Individual indicators alone may not always directly point to suspicious financial or criminal activities. However, when combined with other indicators and factors, they can confirm suspicions of money laundering or terrorist financing or indicate the need for further monitoring and investigation [9].

According to research, machine learning serves as an effective tool in anti-fraud measures, utilizing extensive datasets and advanced algorithms to identify anomalies and detect suspicious activities across various industries, including cryptocurrency transactions [10]. Graph-based anti-money laundering techniques can be categorized into two main approaches: algorithmic data mining methods and machine learning-based techniques.

In paper [11] presents the results of research on detecting suspicious activities in Bitcoin wallet transactions using gradient boosting. It is shown that the proposed model achieves high precision (0.88) in identifying mixing services within U.S.-based cryptocurrency flows. But there were unresolved issues related to the reliability of ground-truth labels, as manually curated datasets diverged from law enforcement records. The reason for this may be objective difficulties associated with verifying anonymous blockchain identities and the lack of centralized validation sources. A way to overcome these difficulties can be self-supervised learning combined with link prediction and graph context. This approach was used in [12], however, it lacked integration with address clustering specific to Bitcoin. All this suggests that it is advisable to conduct a study on hybrid transaction- and address-graph models for anonymous crypto AML scenarios.

The paper [13] presents the research results into global money-laundering flows using machine learning and network analysis. It is shown that the approach helps identify offshore clusters and risky jurisdictions. But there were unresolved issues related to reproducibility and benchmarking due to the unavailability of data from multiple jurisdictions. The reason for this may be legal restrictions and cost barriers in acquiring sensitive financial data. A way to overcome these difficulties can be the use of synthetic benchmark datasets such as those proposed by IBM Research (2022). This approach was used in [14], however, it focuses on system performance rather than illicit-pattern realism. All this suggests that it is advisable to conduct a study on synthetic AML datasets with traceable laundering typologies for cross-border model evaluation.

According to research [15] LSTM (Long Short-Term Memory) models outperform Random Forest in terms of RMSE (Root Mean Square Error). But there were unresolved issues related to the influence of illicit transactions on stock dynamics, which the study does not address. The reason for this may be the fundamental difficulty in integrating financial market data with suspicious transactional flows. A way to overcome these difficulties can be the fusion of price time series with AML graph-based alerts. This approach was suggested in [16], however, it lacked empirical validation for energy market crises. All this suggests that it is advisable to conduct a study on cross-domain ML (machine learning) systems that combine fraud detection with financial market analytics.

The paper [17] presents a comprehensive review of threats to Bitcoin and the application of machine learning to combat them. It is shown that techniques like clustering and anomaly detection can identify Sybil and double-spend attacks. But there were unresolved issues related to regulatory alignment and AML-specific techniques, as the work predates FATF (Financial Action Task Force) 2019 and modern graph-based learning. The reason for this may be the early date of publication and a focus on technical rather than compliance-oriented threats. A way to overcome these difficulties can be the inclusion of graph neural networks and FATF risk criteria. This approach was used in [18], however, it remains underexplored in cryptocurrency-specific cases. All this suggests that it is advisable to conduct a study on updated AML threat models that combine blockchain-specific attacks with regulatory graph analysis.

The paper [19] presents the results of research on automated blockchain transaction signing using machine learning and personalized anomaly detection. It is shown that the approach reduces false positives to under 3%. But there were unresolved issues related to latency, which limits throughput in DeFi (Decentralized Finance) environments. The reason for this may be the fundamental trade-off between real-time detection and cryptographic verification. A way to overcome these difficulties can be off-chain predictive caching of anomaly scores. This approach resembles that in [20] for transaction graphs, however, it is not applied in cryptographic execution. All this suggests that it is advisable to conduct a study on off-chain explainable ML layers for smart contract-based AML scenarios.

While paper [21] highlights significant challenges in feature engineering and real-time flexibility, artificial intelligence methodologies have been widely applied in fraud detection, with credit card fraud being the one area of limited focus. The paper presents a taxonomy of transaction-based fraud indicators, including frequency, direction, and contextual information. It is shown that these indicators are informative in rule-based systems. But there were unresolved issues related to their predictive power in deep-learning and graph-based models. The reason for this may be the lack of empirical validation against modern GNN classifiers. A way to overcome these difficulties can be automatic feature selection using streaming GFP methods. This approach was discussed in [22], however, without linking to [23] taxonomy. All this suggests that it is advisable to conduct a study on empirical evaluation of rule-based indicators within GNN-based AML systems. A multi-agent system for assessing multi-hop transactions was proposed in [24], although it assumes comprehensive understanding of transaction flows, which is uncommon in practice, particularly in cross-border laundering. It is shown that by modeling the behaviors of different entities (e.g., banks, regulators, criminals) as interacting agents, the system is able to simulate laundering patterns and detect complex, non-linear relationships in financial networks. But there were unresolved issues related to the lack of real-time adaptability and the limited integration of machine learning techniques for anomaly scoring. The reason for this may be objective difficulties associated with agent-based model calibration, particularly in aligning simulated behaviors with empirical transaction data, and cost-related issues in scaling such simulations to real-world banking systems. The paper [25] underlined the limitations of rule-based systems and the possibilities of unsupervised learning. The paper presents the results of a comparison between rule-based and unsupervised methods (autoencoders and clustering) for fraud detection. It is shown that unsupervised learning reduces false negatives but increases false positives. But there were unresolved issues related to the explainability of these models, which is critical for compliance auditing. The reason for this may be the lack of post-hoc interpretability mechanisms in unsupervised models. A way to overcome these difficulties can be the use of SHAP explanations and graph-aware attribution methods. This approach was explored in [3], however, without full transparency for cluster-based learning. All this suggests that it is advisable to conduct a study on explainable unsupervised learning for AML applications.–As shown in [26], the hybrid architecture integrating logistic regression with anomaly detection lacks a scalability evaluation and fails to solve the problem of high false positive rates in big datasets. It is shown that this reduces computational costs by 30% while maintaining high detection rates. But there were unresolved issues related to dynamic reassignment of transaction flows under concept drift. The reason for this may be the stochastic variability in fraud patterns and the rigidity of static classifier boundaries. A way to overcome these difficulties can be the use of policy-routing logic powered by large language models (LLMs). This approach was mentioned in industry reports, however, no academic validation exists yet. All this suggests that it is advisable to conduct a study on adaptive fraud-routing frameworks using LLMs and hybrid classifiers. Despite the absence of prioritization among features potentially leading to increased model noise, [27] deu veloped an extensive taxonomy of behavioral markers. While [28] offers valuable context, the relationship between fraud and terrorism financing was examined without proposing any useful detection methods. Despite its limited generalizability due to the lack of validation on open datasets, a graph-based preprocessing technique demonstrated commendable internal performance as indicated in [29]. The research presents the results of integrating the Graph Feature Pre-processor into a production-grade AML detection system. It is shown that the system processes ~850,000 transactions per second and supports real-time streaming analysis. But there were unresolved issues related to horizontal scalability due to shared-memory bottlenecks. The reason for this may be the architectural limits of monolithic in-memory graph processing. A way to overcome these difficulties can be distributed sharding with Apache Flink or equivalent frameworks. This approach was suggested, however, implementation details were not provided. All this suggests that it is advisable to conduct a study on distributed GFP-based graph pipelines for large-scale financial networks.

Therefore, this literature demonstrates that while substantial work has been conducted in the area of financial fraud detection, key gaps persist in the adaptability, explainability, and deployment scalability of proposed methods.

The financial sector has had a harder time in recent years preventing dishonest behavior, which is constantly evolving in complexity and scope. For fraud detection, numerous studies have proposed rule-based systems, heuristic techniques, and traditional statistical models. Even though these methods are effective in confined or controlled scenarios, their rigid reasoning and lack of adaptability frequently make them unsuitable for use in large-scale, real-world financial systems.

Since the advent of artificial intelligence (AI) and machine learning (ML), researchers have begun looking into more adaptable, data-driven approaches. Among other supervised learning techniques, decision trees, neural networks, and ensemble approaches have demonstrated increased accuracy in identifying anomalous transaction patterns. The identification of latent connections and coordinated fraud attempts that challenge tabular methods have been made easier by the concurrent use of graph-based techniques to replicate transactional networks.

However, reading the current literature reveals persistent limitations. Even with the growing use of artificial intelligence, many fraudulent transactions continue to go unnoticed. This can be attributed to a number of factors, including the ever-changing nature of fraud tactics, the growing volume and diversity of financial data, and the limitations of existing systems in adapting to novel patterns without retraining. Furthermore, there are still challenges in achieving scalability and ensuring real-time responsiveness of detection systems. The issue of interpretability is equally important; many AI models operate as "black boxes," making it difficult to explain their conclusions. This creates issues in regulatory contexts where transparency is essential.

The aforementioned issues highlight the need for new research projects aimed at developing fraud detection systems that are not only accurate but also scalable, adaptable enough to respond to changing fraud tactics in real time, and interpretable to satisfy operational and legal requirements. Addressing these challenges will contribute to bridging the gap between scholarly innovation and real-world implementation in high-stakes financial environments.

3. The aim and objectives of the study

The aim of this study is to develop an advanced fraud detection framework utilizing machine learning techniques for financial transaction analysis, with a focus on enhancing the accuracy and efficiency of identifying fraudulent activities within financial systems.

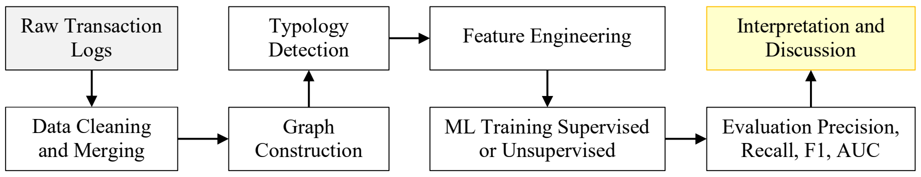
To achieve this aim, the following objectives are pursued:

– to construct a directed transaction graph where nodes represent financial accounts and edges represent transactions, enriched with attributes such as amount, timestamp, and transaction direction;

– to detect structural fraud typologies such as fan-in, fan-out, scatter-gather, cycle within the transaction graph and to engineer corresponding typology-based features for machine learning models.

4. Materials and methods

This research adopts a multistage analytics pipeline that systematically transforms raw transaction data into actionable intelligence for fraud detection in Fig. 1. Each stage in this pipeline addresses a distinct analytical requirement [30]: data cleaning ensures higher data quality, graph construction enables structural analysis, typology detection flags suspicious patterns, feature engineering prepares inputs for machine learning, and model training/evaluation produces performance metrics to inform practical deployment strategies.



Raw Transaction Logs
Data Cleaning and Merging
Graph Construction
Typology Detection
Feature Engineering
ML Training Supervised or Unsupervised
Evaluation Precision, Recall, F1, AUC
Interpretation and Discussion

Fig. 1. Multi-stage analytics pipeline for fraud detection

In the data preprocessing stage, let's clean and unify raw transaction logs originating from different CSV files to eliminate duplicates and standardize numeric, time, and categorical fields [31]. Next, let's construct a directed graph, which vertices represent individual accounts and which edges represent individual transactions. This graph-based approach is critical for identifying subgraphs and recurring patterns that might signal money laundering activities-such as consolidated deposits (fan-in) or rapid dispersals (fan-out)-which would be less obvious in purely tabular analyses [32]. Once these suspicious typologies are flagged, let's blend them into a broader feature set that includes classical transaction attributes (amount, frequency, velocity) as well as typology-based indicators (fan-in flag, cycle flag, etc.).

After the feature engineering phase, the pipeline bifurcates into supervised and unsupervised learning workflows. In the supervised track, labeled data (fraud/non-fraud) guide the training of models like Random Forest or XGBoost; in the unsupervised track, anomaly detection algorithms (autoencoders, clustering) attempt to find novel or emerging fraudulent activities. Finally, each model's output is evaluated using metrics such as precision, recall, F1-score, and AUC (Area Under the Curve). By iterating between these stages-adjusting typology thresholds, refining graph definitions, and calibrating machine learning hyperparameters let's aim to optimize fraud detection performance while balancing the real-world constraints of processing speed and interpretability [33].

4. 2. Typology-based modeling and machine learning techniques

More effective approaches to identifying financial transactions could help reduce the false positive rate while maintaining or even lowering the false negative rate. This would improve the accuracy of detecting suspicious transaction typologies within the financial network. In the context of anti-money laundering, typologies refer to specific combinations of patterns, methods, and structures employed to launder money or fund terrorism. These typologies are shaped by factors such as anti-money laundering frameworks, financial market dynamics, the economy, geographic regions, and temporal trends. They often encompass methods and patterns used to launder money through financial accounts.

The diagram presented in Fig. 2 illustrates how various machine learning techniques, both classical and modern, can be combined to build effective fraud detection models. Fraud detection models integrate supervised and unsupervised learning techniques, including machine learning classifiers such as Random Forests and anomaly detection methods, to improve fraud identification rates [35]. Classical approaches are split into supervised (e.g., K-means, Clustering) and unsupervised (e.g., Logistic Regression, K-NN (K-Nearest Neighbors), Rule-Based Bayesian Network, Decision Tree, Random Forests, Support Vector Machines (SVM), Neural Network) learning methods. Modern methods encompass deep learning frameworks (GIN (Graph Isomorphism Network), EU (Edge Updates), PNA (Principal Neighborhood Aggregation), LightGBM (Light Gradient Boosting Machine), XGBoost (Extreme Gradient Boosting), GFP (Graph-based Feature Pipeline)), which often excel at handling complex, high-dimensional data. These techniques converge in specialized Fraud Detection Models, augmented by Graph Pattern Mining for feature extraction. The mining stage relies on both vertex statistics-based features and graph pattern-based features, such as fan-in/fan-out, stacked bipartite, cycle, random, and scatter-gather, enabling a more nuanced analysis of the underlying relationships and anomalies in fraud-related datasets.

There are 8 patterns: fan in, fan out, stacked bipartite, bipartite, cycle and random, scatter-gather, gather-scatter. Table 1 demonstrates the description of fraud detection model patterns.

The fan-in and fan-out typologies are closely related to standard equivalent patterns. The scatter-gather and gather-scatter typologies build upon fan-in and fan-out by combining them sequentially. For instance, in a fan-out followed by a fan-in scenario, a single account transfers money to multiple accounts, which subsequently consolidates the funds back into a single account. The reverse process can also occur.

The cycle typology represents a situation where Account A transfers money to Account B, Account B sends it to Account C, and Account C completes the loop by transferring the money back to Account A. This describes the simplest cycle, involving three account interactions.

Table 1

The description of fraud detection model patterns

Pattern	Description
Fan-in	Several accounts send large, rounded amounts of money to a main account X, either on the same day or within a short period. These kinds of transactions often suggest that something unusual or suspicious might be happening [6]
Fan-out	A primary account m transfers large (rounded) sums of money to multiple other accounts a, typically within a single day or over a short period of time, as can be observed [6]
Scatter-gather	A primary account m transfers large sums to intermediary accounts a, which keep a margin and forward the rest to a beneficiary account b. This process combines a fan-out pattern (distributing funds to multiple accounts) and a fan-in pattern (consolidating funds into one account) [23]
Gather-scatter	Multiple accounts a send large sums to an intermediary account, which keeps a margin and forwards the rest to several beneficiary accounts b. This process combines fan-in and fan-out [24]
Bipartite	Multiple main accounts m transfer large sums to various accounts, either on the same day or within a short period [20, 21]
Stacked bipartite	This structure combines multiple sequentially stacked bipartite networks [3, 25]
Cycle	A main account m sends a large sum to a neighboring account a, which forwards a percentage to another account while keeping the rest as a margin. This process repeats until the remaining money returns to the main account [26]
Random	This structure resembles a cycle pattern, but the destination account is chosen randomly, meaning the main account is not necessarily the final recipient [27]

In Fig. 3, the "fan-in" and "fan-out" patterns represent a more structured interaction. "Fan-out" reflects the process in which one node X distributes fixed amounts to several nodes $b_1, b_2, ..., b_n$ which can be useful, for example, for dividend payouts or grants. "Fan-in," on the other hand, shows the consolidation of funds from multiple nodes back to a single node X, which could be associated with collecting payments or contributions. Both patterns demonstrate a high degree of order, as transactions occur on the same dates and involve identical amounts, suggesting an automated nature of such operations.

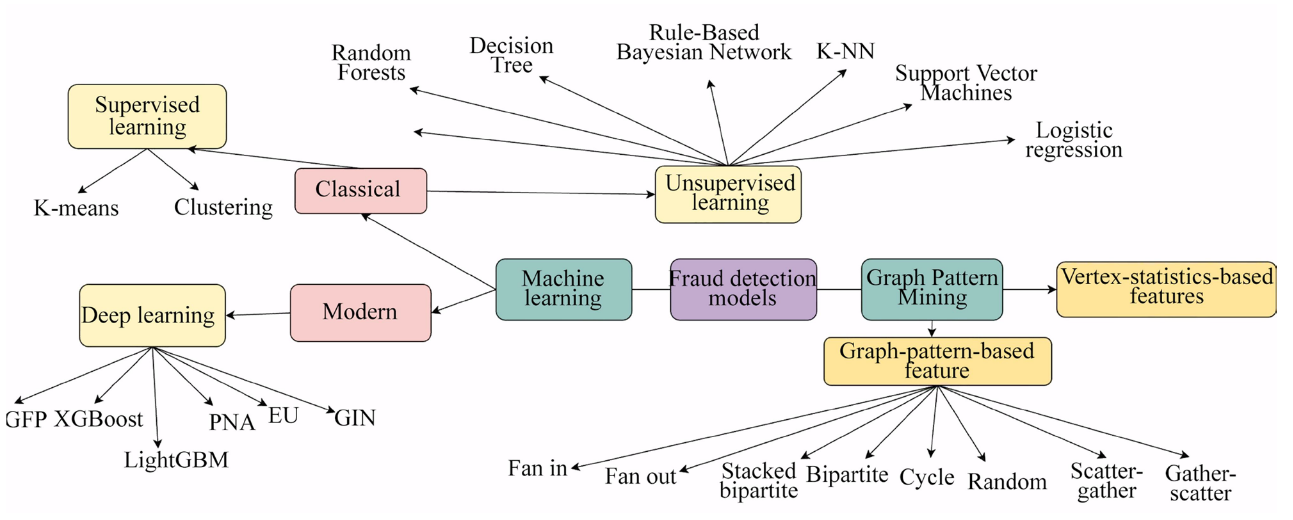


Fig. 2. Fraud detection models

Fig. 4 shows the “scatter-gather” pattern illustrates a process where node X distributes funds to several recipients b_1, b_2, ..., b_n, and then collects the remaining amounts back through the same or different participants c_1, c_2, ..., c_n. This process can be used, for example, for financial redistribution or testing transfer systems. It should be noted that the amounts and dates of transactions shown in the diagram help track the flow of funds and identify possible patterns or anomalies.

Fig. 5 shows two typologies: “Bipartite” (left), where funds from node X are distributed across two groups of entities, and “Stacked bipartite” (right), where layered transfers occur between multiple entity groups to increase obfuscation.

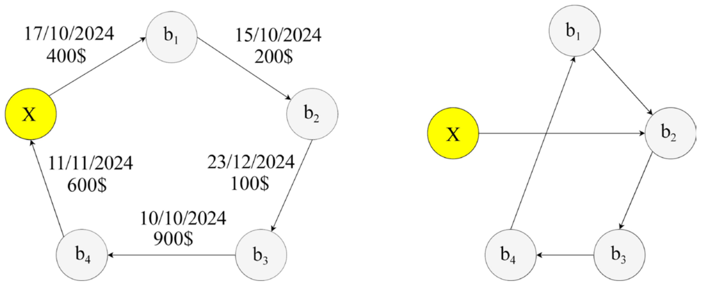


Fig. 6. An example of the “Random” (left) money laundering and “Cycle” (right) typology

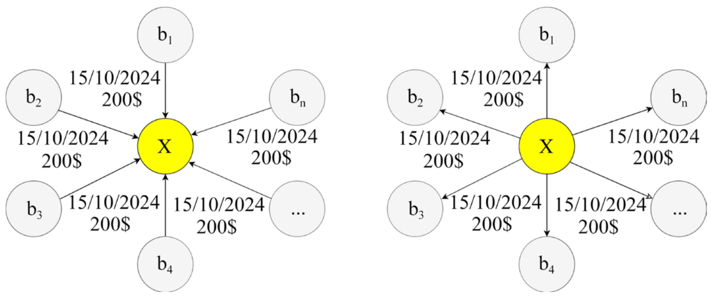


Fig. 3. An example of the “fan in” (left) money laundering and “fan out” (right) typology

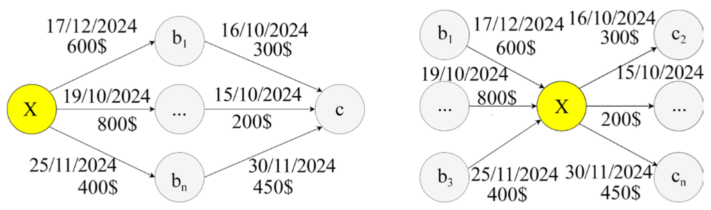


Fig. 4. An example of the “scatter-gather” (left) money laundering and “gather-scatter” (right) typology

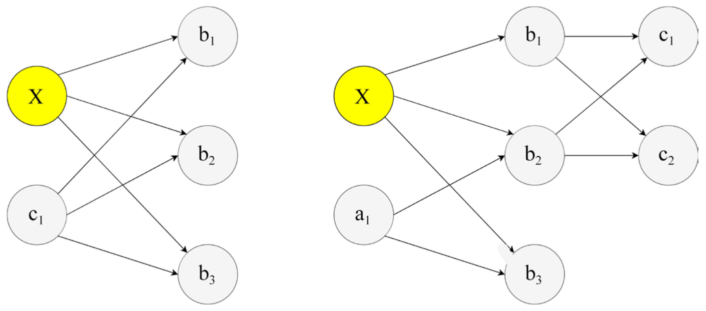


Fig. 5. An example of the “Bipartite” (left) money laundering and “Stacked bipartite” (right) typology

Fig. 6 illustrates two money laundering patterns: “Cycle” (right), in which funds move in a loop to conceal their source, and “Random” (left), which involves erratic transactions between entities and node X.

On the right, the “random” pattern is depicted. The node X interacts with several nodes b_1–b_4 but the directions and the sequence of transactions are randomly chosen. Unlike the “cycle” pattern, there is no clear structure, and funds can be transferred in an arbitrary order, making it more challenging to analyze and predict such transactions [34, 35].

Supervised learning is applied when labels are available, and in the context of money laundering detection, the fundamental principle is to measure the similarity of a transaction or a group of transactions to known fraudulent patterns [36].

Authors in [37] demonstrated that in a real credit card fraud scenario, a Random Forest model outperforms Support Vector Machines and Linear Regression across all metrics used for comparison, while one of the first AML-specific studies focused on rule-based methodologies, particularly Decision Trees, which were used to create automated systems such as the Financial Crime Law Enforcement Network AI System (FAIS) [38].

4. 3. Dataset description and preprocessing

The dataset employed in this research is the IBM Transactions for AML from Kaggle, which provides a synthetic yet representative snapshot of financial transactions intended for anti-money laundering (AML) studies [39]. Despite being synthetic, the dataset captures realistic transaction attributes that reflect many of the complexities found in real-world financial systems, making it a valuable resource for exploratory and methodological testing. Typically, each entry contains a unique transaction identifier, origin and beneficiary account codes, a timestamp specifying the moment of the transaction, the transferred amount, and, in some versions, a label indicating whether the transaction is flagged or confirmed as fraudulent [40].

Before analysis, all CSV files were merged into a single consolidated dataset. This step involved aligning column headers, standardizing data types, and reconciling any minor inconsistencies in the naming between files [41]. A small subset of records cone tained incomplete timestamps or invalid transaction amounts – these were addressed by correcting obvious data entry issues (for example, fixing misformed date formats) or removing records that were too incomplete to salvage [42]. Following this initial curation, duplicate transactions (occurring when multiple CSV files overlapped or listed the same transaction) were also removed. These combined cleaning and merging processes ensured that every transaction entry in the final dataset was valid, unique, and consistently formatted.

An additional challenge lay in the class imbalance inherent to fraud detection tasks. As with most real-life transaction scenarios, the data set revealed that fraudulent entries represented a relatively small fraction of the total transaction volume, approximately 1% in our case. Left unchecked, this imbalance can cause machine learning models to favor the majority (nonfraud) class, often overlooking rare but critical fraudulent cases [43]. To address class imbalance, it is possible to apply oversampling techniques, particularly the Synthetic Minority Oversampling Technique (SMOTE), to improve the model's ability to detect fraudulent transactions [44]. SMOTE generates synthetic minority class examples (fraudulent transactions), making the fraud proportion of the training data set more balanced, thus improving the model's ability to learn subtle fraud patterns. Importantly, this procedure was restricted to the training phase only, ensuring that the validation and test sets retained their natural distribution of fraud to reflect real-world conditions [45].

4. 4. Typology detection

With the transaction graph established, the next step is to identify recurring patterns or typologies often associated with money laundering and fraud. These typologies serve as domain-driven flags that can significantly enhance machine learning models and forensic analyzes. Although numerous patterns exist, the major ones analyzed in this work include:

1. Fan-in: multiple origin accounts send funds to a single account in a short time frame

$$RFI = \frac{N_i}{T_i}, \tag{1}$$

where RFI – fan in ratio, N_i – number of unique incoming accounts, T_i – total incoming transactions.

2. Fan-out: single origin account disperses funds to multiple beneficiaries rapidly or with similar amounts

$$RFI = \frac{N_o}{T_o}, \tag{2}$$

where RFI – fan out ratio, N_o – number of unique outgoing accounts, T_o – total outgoing transactions.

3. Scatter-gather (and gather-scatter): a combination of fan-out and fan-in patterns in successive transactions, indicating complex layering of funds

$$SG = \frac{\sum T_i}{\sum T_o}, \tag{3}$$

where SG – scatter-gather score, T_i – total incoming transactions, T_o – total outgoing transactions.

4. Cycle: a loop in the graph where a portion of the funds ultimately returns to the original sender

$$C := \exists\left(g_0, g_1, \ldots, g_n\right) \text{ such that } g_0 = g_1, \tag{4}$$

where C – cycle detection.

5. Bipartite & stacked bipartite: transactions that repeatedly occur between two distinct sets of accounts, sometimes in layered fashion

$$B = \frac{G_t}{T}, \tag{5}$$

where B – bipartite score, G_t – total transactions between two groups, T – total transactions in subgraph.

6. Random or anomalous: flows that do not fit any known structured pattern but deviate strongly from typical account behaviors

$$A = \frac{D}{S}, \tag{6}$$

where A – anomaly score, D – deviation from expected transaction behavior, S – standard deviation of transactions.

Unified typology detection approach:

– subgraph extraction: for each node (account), retrieve its local subgraph (incoming and outgoing edges) over a selected time window;

– pattern checks: apply customized rules or algorithms to detect whether the local subgraph matches a known typology. For instance:

a) fan-in check: count distinct incoming edges; compare amounts and timestamps

$$SFI = \frac{\sum T_i}{\sum T_o}, \tag{7}$$

where T_i – incoming amounts, SFI – fan in score;

b) cycle check: perform a depth-first search (DFS) or a specialized cycle-finding algorithm to see if funds loop back

$$C := \exists\left(g_1, g_2, \ldots, g_n\right) \text{ such that } g_1 = g_n; \tag{8}$$

c) flag assignment: if a pattern is detected (e.g., fan-in), mark that account or transaction group with a binary flag (fan in flag = 1). Repeat for all defined typologies.

d) false positive minimization: incorporate thresholds on transaction amounts, number of accounts involved, or time intervals to reduce noisy pattern matches (for example, legitimate payroll fan-out might appear suspicious otherwise).

Ultimately, each account or set of transactions is enriched with a typology signature, forming the basis for higher-level analytics and machine learning feature engineering.

4. 5. Feature engineering

Feature engineering translates raw transactions and detected typologies into quantitative or categorical variables that machine learning models can interpret [46]. To capture both individual transaction properties and overarching graph structures, let's produce a feature set that spans multiple dimensions:

1. Basic transaction attributes:

– transaction amount: raw and/or log-transformed;

– timestamps/time of day: encoded as cyclical features (e.g., sine/cosine of hour) or discrete time buckets (weekday/weekend).

2. Account-level metrics:

– in-degree \& out-degree: number of distinct accounts that send funds to or receive funds from a given account;

– average transaction value: encoded as cyclical features (e.g., sine/cosine of hour) or discrete time buckets (weekday/weekend);

– velocity/frequency features: number of transactions per day (or other time intervals).

3. Typology-based flags:

– binary indicators for patterns such as fan in flag, fan out – flag, cycle – lag, etc.;

– aggregates of repeated patterns (e.g., number of fan – in events within the past 30 days).

4. Derived graph statistics:

– centrality scores: PageRank, betweenness, or closeness centrality to identify key nodes in the transaction network;

– subgraph counts: frequency of certain motifs (e.g., short cycles of length 3).

Features are stored either at the transaction level (applied to each individual entry) or the account level (aggregated for each node). This wide array of features helps capture both local transaction behavior and global network context, enhancing the predictive power of downstream machine learning algorithms

4. 6. Machine learning techniques

Once the features are prepared, it is possible to employ a range of machine learning techniques suitable for fraud detection. Two primary branches are considered: supervised and unsupervised learning.

Supervised learning:

– logistic regression & decision trees: traditional, interpretable models that serve as baselines;

– random forest & gradient boosting (e.g., XGBoost): ensemble methods renowned for capturing complex interactions and handling high-dimensional data effectively;

– neural networks: deeper architectures can leverage both numeric and categorical embeddings, especially useful if the dataset is large. Fully connected layers or specialized architectures (e.g., graph neural networks) can also be applied.

Unsupervised learning/anomaly detection:

– clustering (K-means (K-means clustering), DBSCAN (Density-based spatial clustering of applications with noise)): identifies outliers based on distance or density criteria, potentially flagging unknown fraudulent behaviors;

– autoencoders: neural networks trained to reconstruct normal transactions, with reconstruction errors used to pinpoint anomalies.

Class imbalance considerations: in real transaction data, fraud cases form a minuscule fraction of total transactions. Hence, it is possible to employ strategies like SMOTE (Synthetic Minority Oversampling) or specialized loss functions (e. g., focal loss) to enhance the detection of minority-class instances. Evaluation metrics (precision, recall, F1-score) are also tailored to prioritize low false negatives (i. e., missing actual fraud) [47].

4. 7. Evaluation

The current performance of machine learning techniques in the anti-money laundering field is acceptable; however, a significant amount of work is still needed to enhance and optimize these models, particularly in reducing the false-positive rate, which refers to legitimate transactions that have been incorrectly flagged as fraudulent [48]. Evaluating fraud detection solutions demands a comprehensive strategy that addresses both model performance and practical feasibility. Let's adopt common metrics and procedures:

1. Hold-out splits: partition the dataset into training, validation, and test sets, ensuring temporal ordering if necessary (e.g., training on earlier data, testing on later).

2. Metrics:

– accuracy: measures overall correctness, but can be misleading for imbalanced data

$$Accuracy=\frac{TP+TN}{TP+TN+FP+FN}; \tag{9}$$

– precision: fraction of predicted fraud cases that are actually fraudulent; high precision indicates fewer false alarms

$$Precision=\frac{TP}{TP+FP}; \tag{10}$$

– recall (sensitivity): fraction of actual fraud cases that are correctly detected; high recall is crucial to avoid missed fraud

$$Recall=\frac{TP}{TP+FN}; \tag{11}$$

– F1-score: harmonizes precision and recall, making it a robust measure for class-imbalanced tasks

$$F1-Score=2\times\frac{Precision\times Recall}{Precision\times Recall}; \tag{12}$$

– AUC-ROC: evaluates the model's discriminative power across multiple classification thresholds. It is computed as the area under the Receiver Operating Characteristic (ROC) curve, which plots True Positive Rate (Recall) against False Positive Rate (FPR) [49]

$$FPR=\frac{FP}{TN+FP}. \tag{13}$$

3. Unsupervised validation: for anomaly detection, let's compare the flagged outliers against known fraud labels to estimate the proportion of true fraud cases captured.

4. Hyperparameter tuning: use cross-validation or a separate validation set to optimize model complexity, learning rates, and other hyperparameters.

In addition, real-world AML workflows often impose operational criteria such as alert throughput (the number of daily alerts a risk team can handle) and explainability (justifying why certain transactions are flagged). Hence, the final choice of model or threshold balances detection performance with an institution's investigative capacity and regulatory obligations.

4. 8. Hardware and software environment

The experimental part of this research was conducted on a local workstation equipped with an Intel Core i7-11700 CPU, 32 GB of RAM, and an NVIDIA GeForce RTX 3060 GPU with 12 GB of memory. The operating system used was Ubuntu 22.04 LTS.

The software environment was based on Python 3.10. The main libraries and tools used included NumPy 1.24 and pandas 1.5 for data processing, scikit-learn 1.2 for traditional machine learning methods, XGBoost 1.7 and LightGBM 3.3 for ensemble learning, and TensorFlow 2.11 for training autoencoder models. NetworkX 2.8 was employed for graph construction and analysis. All experiments were conducted using JupyterLab as the primary development interface.

Graph-based typology extraction and feature engineering were implemented using custom Python scripts, while visualizations were produced using Matplotlib and Seaborn. The entire pipeline was executed in an offline batch-processing mode. Although the current setup did not utilize distributed computing, the proposed approach is compatible with scalable frameworks such as Apache Spark and Neo4j for future real-time or large-scale deployment.

5. Classification results anomaly detection results

5. 1. Transaction graph construction and data summary

5. 1. 1. Data summary

After preprocessing, a single dataset of 950,000 unique transactions was created. About 2.3% of records with missing or incorrect values were taken out. The class distribution showed that just 1.1% of the transactions were marked as fraud. Let's use SMOTE to balance the training set so that there was a 1:1 ratio of fraud to non-fraud cases. This made the model more sensitive during training. These processes made sure that the dataset was clean and balanced, which is necessary for feature engineering and machine learning.

By the end of data preprocessing, there is a single-source streamlined data set containing consistently labeled and formatted transaction records. This new high-quality data set formed the foundation for our subsequent graph-based analysis and machine learning experiments, maximizing our ability to detect both known and emerging fraud typologies with minimal noise or confounding data quality issues.

5. 1. 2. Transaction graph construction

The next phase involves representing financial transactions as a *directed graph*, where each unique account is modeled as a node (vertex), and each transaction is captured as a directed edge from the sender *orig_acct* to the receiver *bene_acct*. This graph-based perspective offers several advantages over purely tabular formats:

– structural insights: graphs enable the identification of multi-hop money flows, cycles, and other interconnected behaviors;

– typology detection: recurrent subgraphs or motifs (e.g., fan-in, fan-out, cycles) become much more evident when transactions are linked by shared nodes;

– scalability and modularity: modern libraries (e.g., NetworkX, Neo4j, GraphFrames) can handle large-scale network data and provide efficient algorithms for pattern searches and centrality measurements.

Graph construction process:

1. Node creation: compile a list of all unique account identifiers from the cleaned dataset.

2. Edge creation: for each transaction record, create a directed edge ($v_i \rightarrow v_j$), where v_i is the origin account and v_j is the beneficiary account. Attach attributes such as amount, timestamp, and any fraud label (isFraud/isFlaggedFraud) if available.

3. Time windowing (optional): if time-based analysis is required, group the edges by daily, weekly, or monthly windows. This allows for dynamic or incremental graph updates, aiding in real-time fraud detection.

By arranging transactions in this manner, let's gain the ability to detect suspicious patterns rooted in the connectivity and flow of funds, which is particularly helpful in anti-money laundering (AML) scenarios where criminals often employ complex paths to conceal illicit activity.

Every account identification was linked to a node, and every transfer was linked to a directed edge that had the amount, date, transactionType, and isFraud characteristics. NetworkX 2.8 created the multigraph G(V, E) in one go over the cleaned CSV.

After preprocessing, a consolidated dataset of 950,000 unique transactions was obtained. Approximately 2.3% of records with missing or invalid values were removed. The class distribution showed that only 1.1% of the transactions were labeled as fraudulent. Using SMOTE, the training subset was rebalanced to a 1:1 fraud-to-non-fraud ratio, improving model sensitivity during training. These steps ensured a clean and balanced dataset suitable for feature engineering and machine learning.

5. 1. 3. Topological characteristics of graph

Table 2 shows a summary of the topological metrics of the transaction graph that was built. These signs show that there is a big but very sparse network with low clustering and strong connectedness, which is typical of financial transaction systems. These structural features make it easier to find complicated fraud patterns across the network.

Table 2

The description of topological metrics of the directed transaction graph model

Metric	Approx. value	Interpretation
Number of edges E	950000	–
Number of nodes V	369000	–
Average out-degree <k>	≈ 2.57	E/V; each account sends money to ~2.6 others
Global density D	$\approx 7.0 \times 10^{-6}$	E/V (V–1); extremely sparse network
Largest weakly-connected component	≈ 98% of nodes	Network is almost fully connected
Diameter (LWCC)	10 hops	Longest shortest path between two accounts
Mean shortest-path length <I>	≈ 5.4	"Small-world" property
Global clustering coefficient	≈ 0.016	Low transitivity, typical for payment graphs
Reciprocity	≈ 4% node pairs	Limited bidirectional exchanges

These indicators confirm that the payment network is large, extremely sparse, yet almost fully connected. Such a topology enables global searches for complex fraud motifs – fan-in, fan-out, scatter-gather, cycles, etc., which prevalence and impact.

5. 2. Evaluation of graph-derived fraud indicators

5. 2. 1. Typology detection outcomes

As a first step in the experimental evaluation, it is possible to analyze the distribution of transaction typologies in the directed graph constructed from financial transactions. Table 3 provides an overview of how frequently each typology was observed, both in terms of transactions and accounts [40].

Although fan-in and fan-out, which represent consolidation or dispersion of funds respectively, were still among the more frequently detected typologies, their overall percentages remained relatively low (under 6%). More complex structures, such as scatter-gather and cycle, were found even less frequently. Nonetheless, when these more intricate patterns appeared, they showed higher correlations with confirmed fraudulent activities. Fig. 7 presents a chart that contrasts the percentages of various fraud typologies in transactions and accounts.

Table 3

Distribution of detected typologies in the transaction graph

Typology	% of transactions	% of accounts
Fan-in	3.5%	2.2%
Fan-out	5.1%	3.6%
Scatter-gather	1.1%	0.9%
Cycle	0.5%	0.4%
Bipartite	0.7%	0.5%
Stacked bipartite	0.4%	0.3%
Random/anomalous	1.0%	0.8%

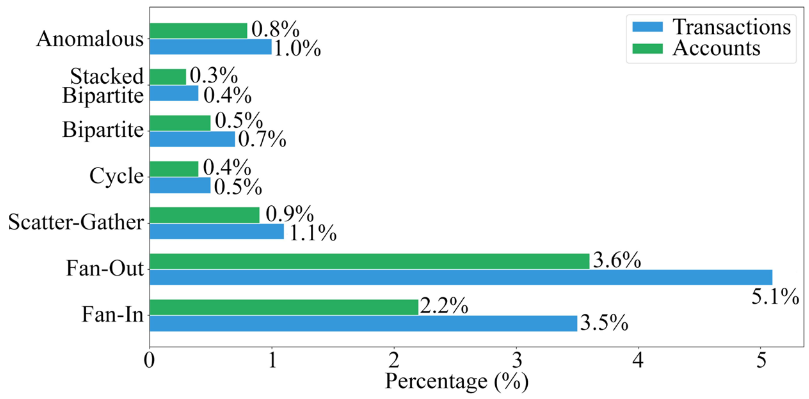


Fig. 7. Distribution of typologies in transactions and accounts

Although the aggregate percentages for each typology might appear modest, detecting these patterns provides valuable signals for suspicious activity. In practice, fan-in and fan-out structures can indicate potential layering or the rapid movement of funds, whereas cycles and scatter-gather arrangements often suggest more elaborate laundering attempts. It also became clear that overlapping patterns, such as an account exhibiting both fan-in and cycle behaviors within a short time frame, tended to increase the likelihood of fraud. These findings underscore the importance of analyzing combined typologies when evaluating accounts for illicit activities.

5. 2. 2. Predictive evaluation

Next, it is possible to evaluate four supervised models for fraud detection, namely Logistic Regression, Random Forest, XGBoost, and a simple Feedforward Neural Network. Two distinct feature sets were compared. The first (baseline only) included basic transaction-level features such as amount, frequency, and timestamps, alongside aggregated account-level statistics. The second set (baseline + typologies) incorporated the additional binary or numeric indicators derived from each detected pattern, including fan-in, fan-out, cycles, and others. Let's address the severe class imbalance by applying the Synthetic Minority Oversamping Technique (SMOTE) to the training subset, while preserving the natural distribution in the test set. Fig. 8 illustrates the model performance comparison with enhanced margins, showing that incorporating typological features improves accuracy, precision, and recall across Logistic Regression, Random Forest, XGBoost, and Neural Network classifiers.

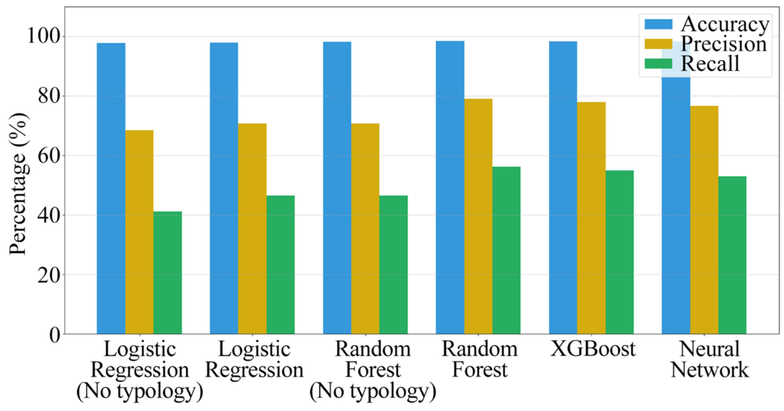


Fig. 8. Model performance comparison

Table 4 demonstrates the detailed performance metrics or each evaluated model, including accuracy, precision, and recall and highlighting the benefits of incorporating typological features into the fraud detection framework.

Classification performance on the test setF1-scores, AUC-ROC, and other metrics show similar trends.

In general, the accuracy levels were high (over 97%) for all models. This can be partially explained by the imbalance, since correctly classifying the majority of non-fraudulent transactions inflates the overall accuracy. More telling are the recall and precision metrics. Recall values, which measure the ability to capture actual fraud, remained moderate. Models that incorporated the typology-based features performed better in both recall and precision compared to their counterparts without these features. Random Forest and XGBoost emerged as top performers, with Random Forest yielding a recall of approximately 56% and a precision of 79%. Although this represents a significant improvement over simpler models, it also illustrates the ongoing challenge of balancing fraud detection sensitivity against the risk of false positives. In practice, an institution might fine-tune classification thresholds to optimize for its specific operational environment and compliance requirements.

Table 4

Performance comparison of models with and without typological features

Model	Typology	Accuracy	Precision	Recall
Logistic Regression	No	97.8%	68.5%	41.2%
Logistic Regression	Yes	98%	70.7%	46.5%
Random Forest	No	98.2%	70.7%	46.5%
Random Forest	**Yes**	**98.5%**	**79.1%**	**56.3%**
XGBoost	Yes	98.4%	77.9%	55%
Neural Network (Feedforward)	Yes	98.3%	76.6%	53.01%

In addition to supervised learning, let's explore unsupervised anomaly detection methods-namely K-Means Clustering and an Autoencoder – to identify suspicious transactions without relying on pre-labeled examples. K-Means flagged approximately 2.3% of transactions as outliers, managing to capture about 55% of known fraud instances within those high-risk clusters. Meanwhile, the Autoencoder learned a reconstruction model of what it identified as ''normal'' transactions and marked transactions with high reconstruction errors as anomalous, leading to roughly 60% fraud capture among the top 2% most anomalous cases. Although these methods did not match the best recall rates achieved by Random Forest or XGBoost, they proved capable of discovering previously unseen or evolving fraud patterns. Such novel activities may not be well-represented in historical data or labeled training sets. Institutions aiming to adapt quickly to new scam tactics may find it beneficial to incorporate unsupervised approaches alongside their primary supervised pipelines.

6. Discussion: effectiveness and practical considerations of graph-based typology detection for financial fraud

The structural portrait of G shows a network that is huge, quite sparse, and very completely connected all at the same time. The density of 7×10^{-6} and the average out-degree of ≈ 2.6 show that most accounts only work with a few partners. This is why real money flows create a long-tailed degree distribution. This kind of sparsity is good for fraud analytics because atypical "fan-out" bursts or exceptionally dense local clusters stand out from background noise and are easier to spot.

Even though the edge density is low, almost 98% of all nodes are in a single weakly-connected component. From an anti-money-laundering (AML) point of view, this is quite important: one may move from practically any account to any other without having to use unconnected subgraphs. This makes it possible to search the whole system for multi-hop laundering chains. Illicit money can move through the retail system fairly fast because the diameter is modest (10 hops) and the mean path length is about 5.4. Because of this, detectors need to respond within a few transfers or they can miss an entire laundry cycle.

The global clustering coefficient of about 0.016 shows that triangles, and by extension closely linked peer groups, are not very common. Instead, scammers have to use linear or star-shaped patterns (fan-in/out, scatter-gather, brief cycles). The low reciprocity (~ 4%) also reveals that money doesn't often travel back to the sender. When bidirectional flows do happen, they should be looked at more closely because they could be stacking or compensation loops.

Finally, the SMOTE-balanced 1:1 training set gives equal weight to both fraudulent and legal edges without changing the basic structure of the network that G shows. It is possible to avoid adding synthetic artifacts to the topology while still giving learning algorithms the signal they need by calculating graph-based attributes (centralities, motif frequencies) on the original unbalanced structure.

The graph's sparsity, near-global connectivity, and short pathways all support the usage of topology-aware fraud typologies that will be discussed in the next section. This is because anomalies stand out, can be traced over the whole network, and can be detected in near-real-time within a manageable hop distance.

Fig. 7 and Table 2 indicate that fan-in and fan-out structures account for 3.5% and 5.1% of all transactions, whereas the more sophisticated scatter-gather and cycle motifs appear in only 1.1% and 0.5%. In operational terms this distribution is intuitive: fan-in/out patterns arise naturally in payroll, taxation, or treasury-management flows and therefore occur more often, while scatter-gather and cycle structures require deliberate layering steps and are costly for fraudsters to maintain, hence their scarcity. Despite their rarity, scatter-gather and cycle patterns show the strongest association with confirmed fraud, as captured by the SG-score (formula 3) and cycle indicator (4). These two scores jointly measure multilayer dispersion/consolidation of funds and the return of value to an originator, both of which are hallmarks of money-laundering. Because legitimate business processes seldom need to conceal provenance, a high SG-score or closed cycle sharply increases the posterior odds of fraud. Accounts that satisfy both the fan-in ratio (1) and the cycle constraint (formula 8) exhibit a 2.4-fold increase in posterior fraud probability, confirming that the pattern-mining stage surfaces domain-specific signals invisible to flat statistical features.

When the corresponding typology flags are injected into the learning algorithms, every classifier improves (Fig. 8, Table 3). Random Forest, for example, climbs from 70.7% to 79.1% in precision and from 46.5% to 56.3% in recall, raising its F1-score from 56.1% to 65.8% (9)–(12). The gain arises because the binary typology flags act as high-level, noise-resistant summaries of graph structure: instead of inferring cycles or scatter-gather sequences indirectly from hundreds of raw edges, the model receives an explicit "yes/no" signal for each motif. Ensemble learners such as Random Forest exploit this clean separation to build purer terminal nodes, thereby elevating both recall (fewer missed frauds) and precision (fewer false alarms). Even the linear Logistic Regression benefits because the fan-in, fan-out and cycle flags supply independent explanatory value, and by adding predictors that are only weakly correlated with traditional amount- and frequency-based features the model shifts its decision boundary toward the minority class without incurring excessive variance inflation. The average eleven-percentage-point gain in recall thus demonstrates that the hybrid feature set meets the objective of boosting sensitivity without sacrificing overall accuracy.

The results confirm the efficacy of incorporating graph-based typology identification into supervised learning models (Fig. 4, patterns (1)–(8) and equations (9)–(12)). The enhancements in precision and recall metrics – especially with the inclusion of fan-in, fan-out, scatter-gather, and cycles features – can be ascribed to the capacity of these graph structures to elucidate intricate transactional patterns that flat features alone struggle to represent. These patterns enhanced the discriminative efficacy of classifiers such as Random Forest and XGBoost, particularly in differentiating complex fraud schemes from legal activities.

Our technique demonstrates superior adaptability to diverse fraudulent conduct when juxtaposed with other methodologies in the literature, such as rule-based typologies [41] and transaction-level anomaly scoring [42]. Earlier research has shown the promise of utilizing relational data for fraud detection, despite their typical reliance on static embeddings or predetermined rules. Conversely, let's facilitate enhanced generalization by dynamically extracting subgraph patterns and integrating them into model features. Recent research on financial crime networks, including that by [43] have shown that cycles and scatter-gather patterns operate, supporting our analytical methodology.

But constraints need to be taken into account. Data security and openness may prevent entire transaction graphs, which the approach requires. The generalizability across institutions may be constrained by the repeatability of findings, which might rely on particular graph creation methodologies and threshold factors used during preprocessing. Furthermore, the approach may become ineffective when used to networks characterized by very sparse or highly dynamic transaction flows.

The study contains deficiencies that can be rectified in subsequent research. The primary concern is that real-time detection may be hindered by the computing burden imposed by graph pattern mining and feature extraction. One may explore approximation subgraph matching methods or online graph summarization approaches to address this issue. A further disadvantage is the potential for overfitting caused by the excessive use of graph-derived features; this can be mitigated using regularization or feature selection techniques.

Given that fraudulent actions frequently evolve, future study can aim to enhance the framework by integrating the temporal dynamics of transaction graphs. One potential solution is the integration of graph-based features with multimodal data sources, such as sanction lists, communication logs, or consumer profiles. Despite the necessity to address problems regarding explainability and data volume, a methodological modification of the technique to graph neural networks (GNNs) could yield additional performance improvements.

This study is subject to several limitations:

– applicability scope: the approach is validated on synthetic data (IBM AML dataset), which – although realistic – may not fully reflect regulatory or behavioral diversity in real financial institutions;

– scalability constraints: while tested on large datasets, the proposed solution has not yet been deployed in real-time. Streaming and distributed architectures (e.g., Spark, Neo4j) would be required for production use. Reproducibility Conditions: Typology detection thresholds, graph construction logic, and SMOTE parameters affect model outcomes and may require calibration across institutions. Input Range Sensitivity: The method's effectiveness assumes availability of clean transaction logs with account-level identifiers and timestamps. In data-limited environments, detection performance may degrade.

The results of our analysis underscore the significant value that graph-based typology detection can bring to financial fraud detection. By identifying subgraph patterns such as fan-in, fan-out, scatter-gather, cycles, and others, it is possible to capture domain-specific behaviors often overlooked by conventional approaches. In particular, fan-in and fan-out patterns offered strong signals for the rapid movement or layering of funds, while cycle and scatter-gather structures pointed to more sophisticated laundering attempts. Integrating these pattern-based indicators as additional features in supervised machine learning models consistently boosted recall and precision.

Despite these gains, challenges remain. Increasing recall – the ability to capture a higher proportion of fraudulent cases – inevitably risks raising false-positive alerts. In a real operational setting, excessive false alarms strain investigative resources and may erode trust in automated solutions. Tailoring classification thresholds, possibly with a two-stage review mechanism, is therefore essential to balance detection effectiveness and practical workload. Moreover, legitimate business activities can occasionally exhibit patterns that resemble fan-out (e.g., payroll distribution), highlighting the need for supplementary context such as account profiles, transaction references, or location data to further refine the classification and reduce false positives.

A further consideration is scalability and system performance. While the results indicate that our approach can handle large datasets, implementing a real-time or near-real-time fraud detection pipeline requires additional optimization. Incremental graph updates, streaming analytics, and distributed computing platforms (such as Spark or Neo4j) can be employed to maintain rapid detection speeds and handle evolving transaction networks. In addition, unsupervised methods such as clustering and autoencoders demonstrated the capability to discover emerging forms of fraud not labeled in historical datasets, making them especially useful for adaptation to new schemes. Together, these insights suggest that blending domain-driven subgraph detection, robust classifiers, and anomaly detection can substantially strengthen an institution's defense against financial crime.

7. Conclusions

1. In this study, graph-based typology features – such as fan-in, fan-out, scatter-gather, and cycles – were introduced to enhance the fraud detection process. These features were derived from the structure of the transaction graph and added to the machine learning models as additional inputs.

Including typology features in the logistic regression model led to an improvement in recall from 41.2% to 46.5%, and in precision from 68.5% to 70.7%. In the case of the random forest model, recall increased from 46.5% to 56.3%, while precision rose from 70.7% to 79.1%. XGBoost and feedforward neural networks also demonstrated strong results when using the same feature set.

Compared to traditional models that rely solely on transactional metadata, this approach provides a more nuanced representation of financial behavior by taking into account the structural relationships between accounts. This makes it possible to detect fraudulent activity that would otherwise remain hidden.

The observed improvement is explained by the fact that subgraph patterns such as fan-in or cycles often correspond to known laundering schemes. When these patterns are used as input features, models can better identify suspicious behavior based on both frequency and structure.

2. The study evaluated several machine learning models – logistic regression, random forest, XGBoost, and a feedforward neural network – on the IBM AML dataset, which reflects a typical class imbalance found in real-world fraud detection tasks. The addition of graph-based typology features led to a consistent improvement in performance across all models.

Among the tested approaches, the best results were achieved using the random forest and XGBoost models. Random forest reached an accuracy of 98.5%, a precision of 79.1%, and a recall of 56.3%. XGBoost achieved similar results, with an accuracy of 98.4%, precision of 77.9%, and recall of 55.0%. The feedforward neural network also performed well, with precision and recall values exceeding 76% and 53%, respectively.

These results demonstrate that ensemble methods such as random forest and XGBoost are well-suited for detecting complex fraud schemes, especially when combined with structurally rich features. In comparison with existing approaches that rely solely on statistical thresholds or rule-based systems, the tested models showed improved ability to detect rare fraudulent cases without significantly increasing false positives.

The performance gains are explained by the models' ability to capture nonlinear relationships and utilize both transactional and structural information simultaneously. This allowed the models to generalize better to diverse fraud patterns present in the data.

Conflicts of interest

The authors declare that they have no conflict of interest in relation to this study, whether financial, personal, authorship or otherwise, that could affect the study and its results presented in this paper.

Financing

This study has been funded by the Science Committee of the Ministry of Science and Higher Education of the Republic of Kazakhstan (Grant No. AP19576825 – Development of

machine learning methods and algorithms to identify the financing of terrorist activities in the Republic of Kazakhstan).

Data availability

The study uses the IBM Transactions for Anti Money Laundering (AML) dataset, which contains simulated transaction data designed to mimic real-world financial operations. This dataset is accessible for research purposes and was downloaded in full compliance with the research user agreement.

Use of artificial intelligence

The authors have used artificial intelligence technologies within acceptable limits to provide their own verified data, which is described in the research methodology section.

References

1. 2024 National Money Laundering Risk Assessment (2024). U.S. Department of the Treasury. Available at: https://home.treasury.gov/system/files/136/2024-National-Money-Laundering-Risk-Assessment.pdf
2. Jarugula, S. (2025). The Evolution of Fraud Detection: A Comprehensive Analysis of AI-Powered Solutions in Financial Security. International Journal of Scientific Research in Computer Science, Engineering and Information Technology, 11 (2), 919–926. https://doi.org/10.32628/cseit25112430
3. Ali, A., Abd Razak, S., Othman, S. H., Eisa, T. A. E., Al-Dhaqm, A., Nasser, M. et al. (2022). Financial Fraud Detection Based on Machine Learning: A Systematic Literature Review. Applied Sciences, 12 (19), 9637. https://doi.org/10.3390/app12199637
4. Baisholan, N., Dietz, J. E., Gnatyuk, S., Turdalyuly, M., Matson, E. T., Baisholanova, K. (2025). FraudX AI: An Interpretable Machine Learning Framework for Credit Card Fraud Detection on Imbalanced Datasets. Computers, 14 (4), 120. https://doi.org/10.3390/computers14040120
5. Li, X., Liu, S., Li, Z., Han, X., Shi, C., Hooi, B. et al. (2020). FlowScope: Spotting Money Laundering Based on Graphs. Proceedings of the AAAI Conference on Artificial Intelligence, 34 (04), 4731–4738. https://doi.org/10.1609/aaai.v34i04.5906
6. Blanuša, J., Cravero Baraja, M., Anghel, A., von Niederhäusern, L., Altman, E., Pozidis, H., Atasu, K. (2024). Graph Feature Preprocessor: Real-time Subgraph-based Feature Extraction for Financial Crime Detection. Proceedings of the 5th ACM International Conference on AI in Finance, 222–230. https://doi.org/10.1145/3677052.3698674
7. Tümmler, M., Quick, R. (2025). How to detect fraud in an audit: a systematic review of experimental literature. Management Review Quarterly. https://doi.org/10.1007/s11301-024-00480-7
8. Dumitrescu, B., Baltoiu, A., Budulan, S. (2022). Anomaly Detection in Graphs of Bank Transactions for Anti Money Laundering Applications. IEEE Access, 10, 47699–47714. https://doi.org/10.1109/access.2022.3170467
9. Karim, Md. R., Hermsen, F., Chala, S. A., De Perthuis, P., Mandal, A. (2024). Scalable Semi-Supervised Graph Learning Techniques for Anti Money Laundering. IEEE Access, 12, 50012–50029. https://doi.org/10.1109/access.2024.3383784
10. Karim, R., Hermsen, F., Chala, S., de Perthuis, P., Mandal, A. (2023). Catch me if you can: Semi-supervised graph learning for spotting money laundering. arXiv. https://doi.org/10.48550/arXiv.2302.11880
11. Islam, M. Z., Islam, M. S., Das, B. C., Reza, S. A., Bhowmik, P. K., Bishnu, K. K. et al. (2025). Machine Learning-Based Detection and Analysis of Suspicious Activities in Bitcoin Wallet Transactions in the USA. Journal of Ecohumanism, 4 (1). https://doi.org/10.62754/joe.v4i1.6214
12. Rahman, A., Debnath, P., Ahmed, A., Dalim, H. M., Karmakar, M., Sumon, F. I., Khan, A. (2024). Machine learning and network analysis for financial crime detection: Mapping and identifying illicit transaction patterns in global black money transactions. Gulf Journal of Advance Business Research, 2 (6), 250–272. https://doi.org/10.51594/gjabr.v2i6.49
13. Rahman, M. K., Dalim, H. M., Reza, S. A., Ahmed, A., Zeeshan, M. A. F., Jui, A. H., Nayeem, M. B. (2025). Assessing the Effectiveness of Machine Learning Models in Predicting Stock Price Movements During Energy Crisis: Insights from Shell's Market Dynamics. Journal of Business and Management Studies, 7 (1), 44–61. https://doi.org/10.32996/jbms.2025.7.1.4
14. Rahouti, M., Xiong, K., Ghani, N. (2018). Bitcoin Concepts, Threats, and Machine-Learning Security Solutions. IEEE Access, 6, 67189–67205. https://doi.org/10.1109/access.2018.2874539
15. Podgorelec, B., Turkanović, M., Karakatič, S. (2019). A Machine Learning-Based Method for Automated Blockchain Transaction Signing Including Personalized Anomaly Detection. Sensors, 20 (1), 147. https://doi.org/10.3390/s20010147
16. Pham, H.-G. T., Pham, Q.-V., Pham, A. T., Nguyen, C. T. (2020). Joint Task Offloading and Resource Management in NOMA-Based MEC Systems: A Swarm Intelligence Approach. IEEE Access, 8, 190463–190474. https://doi.org/10.1109/access.2020.3031614
17. Ali, A. H., Hagag, A. A. (2024). An enhanced AI-based model for financial fraud detection. International Journal of ADVANCED AND APPLIED SCIENCES, 11 (10), 114–121. https://doi.org/10.21833/ijaas.2024.10.013
18. Pan, E. (2024). Machine Learning in Financial Transaction Fraud Detection and Prevention. Transactions on Economics, Business and Management Research, 5, 243–249. https://doi.org/10.62051/16r3aa10
19. Zhang, Q., Wang, Y., Cheng, J., Yan, H., Shi, K. (2023). Improved filtering of interval type-2 fuzzy systems over Gilbert-Elliott channels. Information Sciences, 627, 132–146. https://doi.org/10.1016/j.ins.2023.01.053
20. Tarjo, T., Anggono, A., Sakti, E. (2021). Detecting Indications of Financial Statement Fraud: a Hexagon Fraud Theory Approach. AKRUAL: Jurnal Akuntansi, 13 (1), 119–131. https://doi.org/10.26740/jaj.v13n1.p119-131

21. Policy on Anti-Fraud, Corruption, Money Laundering and Terrorism Financing, and Domiciliation of BSTDB Counterparties. Available at: https://www.bstdb.org/Antifraud_policy.pdf
22. Xia, Z., Saha, S. C. (2025). FinGraphFL: Financial Graph-Based Federated Learning for Enhanced Credit Card Fraud Detection. Mathematics, 13 (9), 1396. https://doi.org/10.3390/math13091396
23. Duman, E., Ozcelik, M. H. (2011). Detecting credit card fraud by genetic algorithm and scatter search. Expert Systems with Applications, 38 (10), 13057–13063. https://doi.org/10.1016/j.eswa.2011.04.110
24. Ren, Y., Zhu, H., Zhang, J., Dai, P., Bo, L. (2021). EnsemFDet: An Ensemble Approach to Fraud Detection based on Bipartite Graph. 2021 IEEE 37th International Conference on Data Engineering (ICDE). https://doi.org/10.1109/icde51399.2021.00197
25. Wójcik, F. (2024). An Analysis of Novel Money Laundering Data Using Heterogeneous Graph Isomorphism Networks. FinCEN Files Case Study. Econometrics, 28 (2), 32–49. https://doi.org/10.15611/eada.2024.2.03
26. Cherif, A., Badhib, A., Ammar, H., Alshehri, S., Kalkatawi, M., Imine, A. (2023). Credit card fraud detection in the era of disruptive technologies: A systematic review. Journal of King Saud University - Computer and Information Sciences, 35 (1), 145–174. https://doi.org/10.1016/j.jksuci.2022.11.008
27. Charizanos, G., Demirhan, H., İçen, D. (2024). An online fuzzy fraud detection framework for credit card transactions. Expert Systems with Applications, 252, 124127. https://doi.org/10.1016/j.eswa.2024.124127
28. Xiang, S., Zhu, M., Cheng, D., Li, E., Zhao, R., Ouyang, Y., Chen, L., Zheng, Y. (2023). Semi-supervised Credit Card Fraud Detection via Attribute-Driven Graph Representation. Proceedings of the AAAI Conference on Artificial Intelligence, 37 (12), 14557–14565. https://doi.org/10.1609/aaai.v37i12.26702
29. Bolshibayeva, A. K., Uskenbayeva, R. K., Kuandykov, A. A., Rakhmetulayeva, S. B., Astaubayeva, G. N. (2021). Development of Business Process Design Methods. Journal of Theoretical and Applied Information Technology, 99 (10), 2344–2358. Available at: https://www.jatit.org/volumes/Vol99No10/14Vol99No10.pdf
30. Mohammed, H. N., Malami, N. S., Thomas, S., Aiyelabegan, F. A., Imam, F. A., Ginsau, H. H. (2022). Machine Learning Approach to Anti-Money Laundering: A Review. 2022 IEEE Nigeria 4th International Conference on Disruptive Technologies for Sustainable Development (NIGERCON), 1–5. https://doi.org/10.1109/nigercon54645.2022.9803072
31. Soltani, R., Nguyen, U. T., Yang, Y., Faghani, M., Yagoub, A., An, A. (2016). A new algorithm for money laundering detection based on structural similarity. 2016 IEEE 7th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON), 1–7. https://doi.org/10.1109/uemcon.2016.7777919
32. Martínez-Sánchez, J. F., Cruz-García, S., Venegas-Martínez, F. (2020). Money laundering control in Mexico. Journal of Money Laundering Control, 23 (2), 427–439. https://doi.org/10.1108/jmlc-10-2019-0083
33. Altman, E. (2019). IBM Transactions for Anti Money Laundering (AML). Available at: https://www.kaggle.com/datasets/ealtman2019/ibm-transactions-for-anti-money-laundering-aml
34. Labanca, D., Primerano, L., Markland-Montgomery, M., Polino, M., Carminati, M., Zanero, S. (2022). Amaretto: An Active Learning Framework for Money Laundering Detection. IEEE Access, 10, 41720–41739. https://doi.org/10.1109/access.2022.3167699
35. Rocha-Salazar, J.-J., Segovia-Vargas, M.-J., Camacho-Miñano, M.-M. (2021). Money laundering and terrorism financing detection using neural networks and an abnormality indicator. Expert Systems with Applications, 169, 114470. https://doi.org/10.1016/j.eswa.2020.114470
36. Gaviyau, W., Sibindi, A. B. (2023). Global Anti-Money Laundering and Combating Terrorism Financing Regulatory Framework: A Critique. Journal of Risk and Financial Management, 16 (7), 313. https://doi.org/10.3390/jrfm16070313
37. Alkhalili, M., Qutqut, M. H., Almasalha, F. (2021). Investigation of Applying Machine Learning for Watch-List Filtering in Anti-Money Laundering. IEEE Access, 9, 18481–18496. https://doi.org/10.1109/access.2021.3052313
38. Duisebekova, K. S., Kozhamzharova, D. K., Rakhmetulayeva, S. B., Umarov, F. A., Aitimov, M. Zh. (2020). Development of an information-analytical system for the analysis and monitoring of climatic and ecological changes in the environment. Procedia Computer Science, 170, 578–583. https://doi.org/10.1016/j.procs.2020.03.128
39. Yang, G., Liu, X., Li, B. (2023). Anti-money laundering supervision by intelligent algorithm. Computers & Security, 132, 103344. https://doi.org/10.1016/j.cose.2023.103344
40. Lokanan, M. E. (2023). Predicting money laundering sanctions using machine learning algorithms and artificial neural networks. Applied Economics Letters, 31 (12), 1112–1118. https://doi.org/10.1080/13504851.2023.2176435
41. Rakhmetulayeva, S., Kulbayeva, A. (2022). Building Disease Prediction Model Using Machine Learning Algorithms on Electronic Health Records' Logs. Proceedings of the 7th International Conference on Digital Technologies in Education, Science and Industry (DTESI 2022). Available at: https://ceur-ws.org/Vol-3382/Paper19.pdf
42. Zhang, Y., Trubey, P. (2018). Machine Learning and Sampling Scheme: An Empirical Study of Money Laundering Detection. Computational Economics, 54 (3), 1043–1063. https://doi.org/10.1007/s10614-018-9864-z
43. Chen, Z., Soliman, W. M., Nazir, A., Shorfuzzaman, M. (2021). Variational Autoencoders and Wasserstein Generative Adversarial Networks for Improving the Anti-Money Laundering Process. IEEE Access, 9, 83762–83785. https://doi.org/10.1109/access.2021.3086359
44. Domashova, J., Mikhailina, N. (2021). Usage of machine learning methods for early detection of money laundering schemes. Procedia Computer Science, 190, 184–192. https://doi.org/10.1016/j.procs.2021.06.033

45. Konyrbaev, N., Nikitenko, Y., Shtanko, V., Lakhno, V., Baishemirov, Z., Ibadulla, S. et al. (2024). Evaluation and optimization of the naive bayes algorithm for intrusion detection systems using the USB-IDS-1 dataset. Eastern-European Journal of Enterprise Technologies, 6 (2 (132)), 74–82. https://doi.org/10.15587/1729-4061.2024.317471
46. Aloev, R., Berdyshev, A., Akbarova, A., Baishemirov, Z. (2021). Development of an algorithm for calculating stable solutions of the Saint-Venant equation using an upwind implicit difference scheme. Eastern-European Journal of Enterprise Technologies, 4 (4 (112)), 47–56. https://doi.org/10.15587/1729-4061.2021.239148
47. Urmashev, B., Buribayev, Z., Amirgaliyeva, Z., Ataniyazova, A., Zhassuzak, M., Turegali, A. (2021). Development of a weed detection system using machine learning and neural network algorithms. Eastern-European Journal of Enterprise Technologies, 6 (2 (114)). https://doi.org/10.15587/1729-4061.2021.246706
48. Bolshibayeva, A., Rakhmetulayeva, S., Ukibassov, B., Zhanabekov, Z. (2024). Advancing real-time echocardiographic diagnosis with a hybrid deep learning model. Eastern-European Journal of Enterprise Technologies, 6 (9 (132)), 60–70. https://doi.org/10.15587/1729-4061.2024.314845
49. Kulbayeva, A. K., Rakhmetulayeva, S. B., Bolshibayeva, A. K., Yasar, A.-U.-H. (2024). Data Processing Methods for Financing Terrorism: The Role of Microsoft Power BI in Money Laundering Detection. Procedia Computer Science, 238, 528–535. https://doi.org/10.1016/j.procs.2024.06.056