Anomaly Detection For Fraud Prevention in Cryptocurrency

Sai Mohith Gandrapu, Sawan Shivanand Beli, Sonali Arcot

MOTIVATION

Cryptocurrency transactions are vulnerable to fraud due to their pseudonymous and decentralized nature. As cryptocurrency adoption grows, developing effective fraud detection systems is crucial to build trust and integrity in cryptocurrency networks. This study was motivated by the need to leverage machine learning to detect fraudulent transactions, identify suspicious patterns, and prevent financial losses. Additionally, most prior fraud detection research has focused on centralized financial systems, with limited work on decentralized networks. By creating tailored techniques for cryptocurrencies, this study aims to advance the security of blockchain ecosystems. Overall, building robust fraud protection will support mainstream cryptocurrency adoption by providing users safeguards against theft and scams.

BACKGROUND

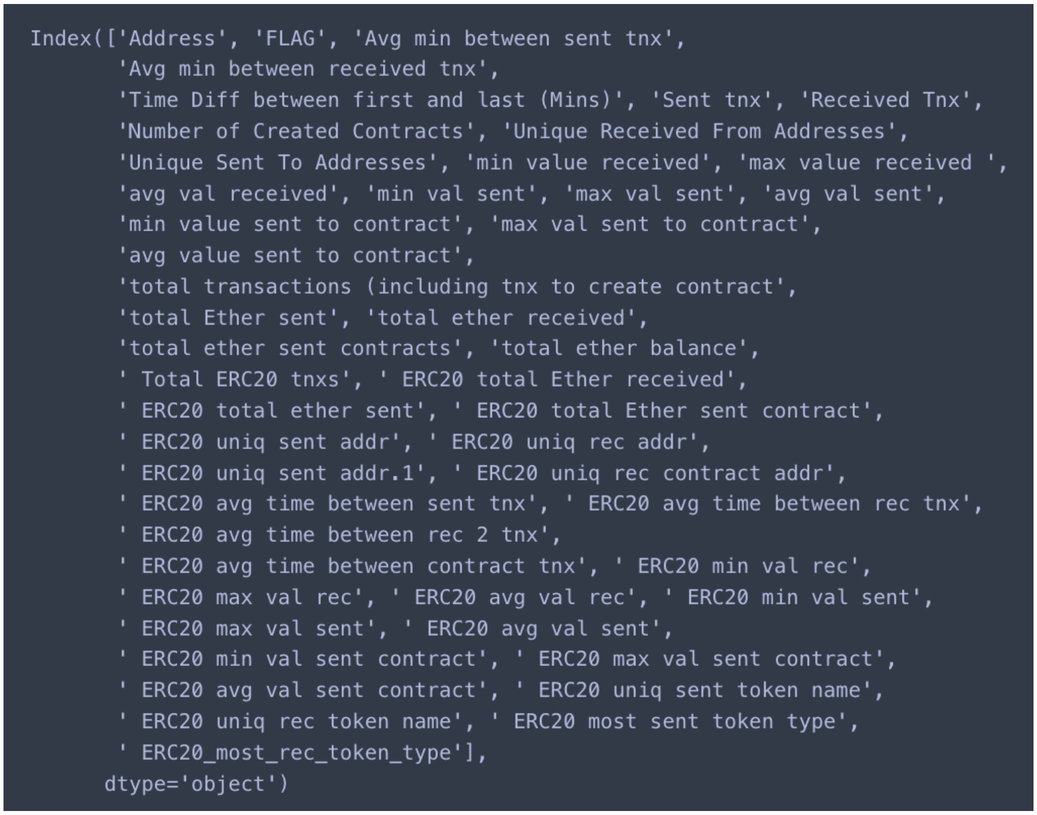
Cryptocurrency transactions rely on public-private key cryptography to authorize transfers between pseudonymous accounts. While this allows for transparency, it also enables fraudulent actors to cover their tracks by creating multiple accounts. Common techniques include using mixer services to obfuscate money trails or distributing stolen funds across many wallets. Prior research has developed rule-based systems or applied simpler machine learning like logistic regression. However, these methods have difficulty detecting evolving attack vectors and complex distribution techniques. More advanced algorithms like neural networks can uncover subtle patterns among many features like time intervals, frequencies, transaction sizes, and wallet connections. This study employs deep learning on the Ethereum blockchain data to build user profiles and identify deviations indicative of fraud. Overall, this advanced approach aims to adapt to new fraud tactics and provide robust protection for cryptocurrency users.

LITERATURE SURVEY

Using CNN, LSTM, bagged and boosted variants, Umer et al. [4] used a deep learning ensemble for the Ethereum fraud detection dataset. Even though there were limited fraudulent transactions in their dataset, they raised concerns about generalizing the approach. According to Martins and Brito [3], CNN achieved 95% accuracy in previous research [4], along with RF, DT, and KNN. Deep learning as well as graph representation can assist in fraud detection, according to Martins and Brito [3]. Traditional methods, such as RF, are highlighted as having challenges, such as manual labeling and balancing recall and FPR, as stated in the paper. According to their findings, graph-based methods such as Graph Convolutional Networks (GCNs) and node2vec, which encode graph structures into dense representations, have demonstrated significant potential. Nevertheless, these models tend to ignore transactional information pertaining to temporality. This limitation was addressed by Alarab and Prakoonwit [1] achieving 98% accuracy by integrating LSTM and GCN models, emphasizing the temporal dimension of transactions. Using a dataset of 1 million Ethereum transactions, a comprehensive comparison study has shown that CNN can accurately detect fraud with over 99% accuracy [2]. In addition to emphasizing the importance of data privacy, the authors recommended expanding research horizons to include emerging fraud tactics. Based on an ensemble of machine learning models, including RF, XGBoost, and AdaBoost, Zik et al. [5] utilized blockchain-based crowdfunding platforms for securing them. XGBoost showed the highest precision and recall on datasets such as ESC and VSC, demonstrating its robustness against DDoS, malware, and apt attacks. Despite the potential of deep learning for the detection of anomalies in cryptocurrency fraud, there are still a number of challenges including the lack of generalization, the scarcity of data, and the evolution of fraud tactics.

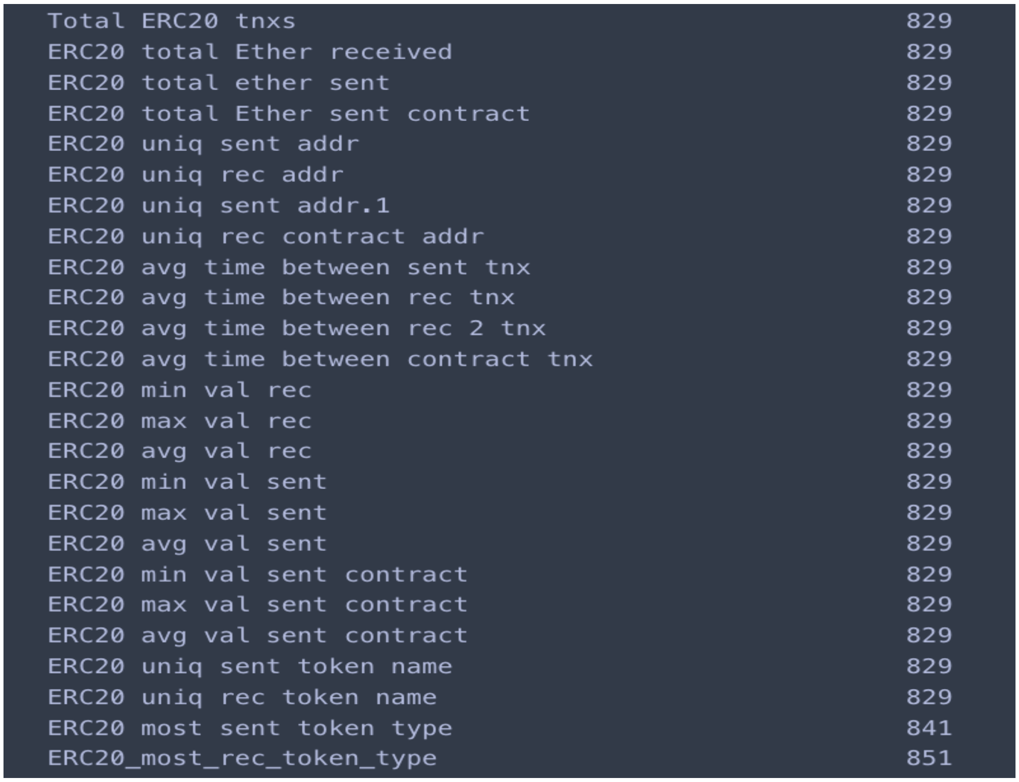
DATA COLLECTION

This project utilized real-world Ethereum transaction data mined from Etherscan using their API capabilities. Etherscan maintains an ongoing list of Ethereum accounts that have been associated with confirmed fraudulent activity based on internal analysis as well as accounts flagged by cryptocurrency exchanges and other blockchain security firms. These include accounts linked to phishing scams, hacks, or general suspicious behavior patterns noticed across multiple transactions. By querying Etherscan's API to retrieve the full transaction histories for over 1500 labeled "phish-hack” accounts, a robust labeled dataset was constructed with extensive examples of confirmed fraud. In total, retrieving all transactions for these suspicious accounts yielded over 4300 observations now confirmed to exhibit fraud by reputable external labeling. Prior to augmenting with the API-mined data, the project dataset only contained a limited number of positively-labeled fraud cases resulting in severe class imbalance. The additional real-world examples dramatically improved the proportion of fraudulent transactions to over 45% of the observations. Analyzing the sequence of transactions for each account also enabled engineering meaningful time-series attributes related to changes in transaction frequency, size, counterparty, and other temporal patterns. These dynamic new features are critically useful in modeling to accurately identify shifts indicative of evolving fraudulent tactics. In summary, programmatically tapping into Etherscan’s industry-curated list of high-risk accounts provided the necessary fraudulent examples to build a balanced, multi-faceted training dataset well-equipped to develop finely tuned fraud detection models. Figure 1 illustrates the entirety of features within the extracted dataset. Subsequently, superfluous columns were eliminated due to their representation of outdated values and the presence of missing data.



```
Index(['Address', 'FLAG', 'Avg min between sent tnx',
       'Avg min between received tnx',
       'Time Diff between first and last (Mins)', 'Sent tnx', 'Received Tnx',
       'Number of Created Contracts', 'Unique Received From Addresses',
       'Unique Sent To Addresses', 'min value received', 'max value received ',
       'avg val received', 'min val sent', 'max val sent', 'avg val sent',
       'min value sent to contract', 'max val sent to contract',
       'avg value sent to contract',
       'total transactions (including tnx to create contract',
       'total Ether sent', 'total ether received',
       'total ether sent contracts', 'total ether balance',
       ' Total ERC20 tnxs', ' ERC20 total Ether received',
       ' ERC20 total ether sent', ' ERC20 total Ether sent contract',
       ' ERC20 uniq sent addr', ' ERC20 uniq rec addr',
       ' ERC20 uniq sent addr.1', ' ERC20 uniq rec contract addr',
       ' ERC20 avg time between sent tnx', ' ERC20 avg time between rec tnx',
       ' ERC20 avg time between rec 2 tnx',
       ' ERC20 avg time between contract tnx', ' ERC20 min val rec',
       ' ERC20 max val rec', ' ERC20 avg val rec', ' ERC20 min val sent',
       ' ERC20 max val sent', ' ERC20 avg val sent',
       ' ERC20 min val sent contract', ' ERC20 max val sent contract',
       ' ERC20 avg val sent contract', ' ERC20 uniq sent token name',
       ' ERC20 uniq rec token name', ' ERC20 most sent token type',
       ' ERC20_most_rec_token_type'],
      dtype='object')
```

Fig 1: All the columns in the mined data



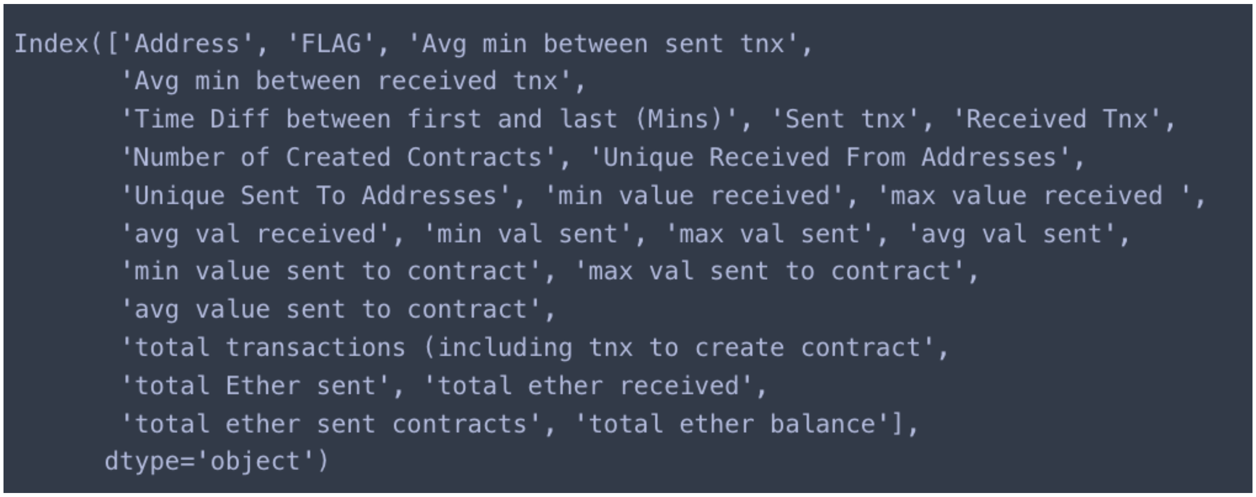
```
Total ERC20 tnxs                                         829
ERC20 total Ether received                               829
ERC20 total ether sent                                   829
ERC20 total Ether sent contract                          829
ERC20 uniq sent addr                                     829
ERC20 uniq rec addr                                      829
ERC20 uniq sent addr.1                                   829
ERC20 uniq rec contract addr                             829
ERC20 avg time between sent tnx                          829
ERC20 avg time between rec tnx                           829
ERC20 avg time between rec 2 tnx                         829
ERC20 avg time between contract tnx                      829
ERC20 min val rec                                        829
ERC20 max val rec                                        829
ERC20 avg val rec                                        829
ERC20 min val sent                                       829
ERC20 max val sent                                       829
ERC20 avg val sent                                       829
ERC20 min val sent contract                              829
ERC20 max val sent contract                              829
ERC20 avg val sent contract                              829
ERC20 uniq sent token name                               829
ERC20 uniq rec token name                                829
ERC20 most sent token type                               841
ERC20_most_rec_token_type                                851
```

Fig 2: All the redundant columns that were dropped that also had missing values

DATA PREPROCESSING

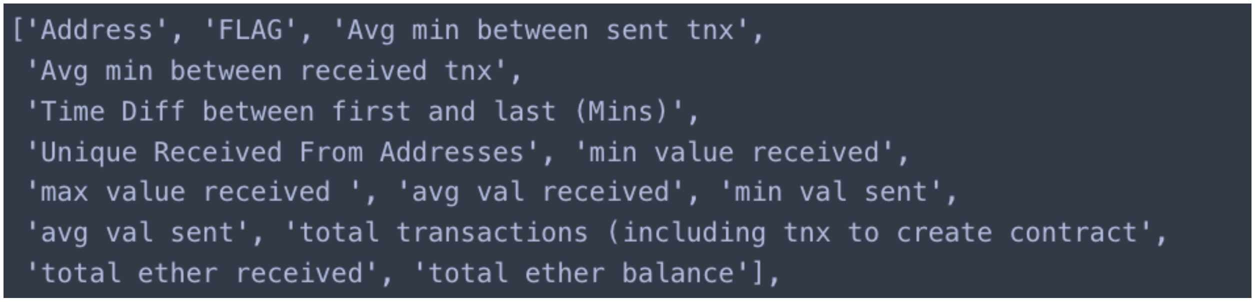
The raw transaction data mined from Etherscan underwent extensive preprocessing to prepare the most relevant features for modeling. While the initial dataset contained extensive transaction statistics for each account as shown in Figure 2, dimensional reduction techniques were applied to reduce noise and enable the models to focus on the most critical patterns. Statistical analysis and correlation scores revealed several features providing limited informational value towards identifying fraud, including: total ether sent contracts, max contract value sent, number of received transactions, and number of sent transactions. These as well as other redundant metrics were dropped, reducing the feature space down to the 14 most meaningful attributes per account as depicted in Figure 3.

The remaining features play a pivotal role in detecting unusual account behavior indicative of evolving fraud tactics. Temporal patterns such as average time between sent transactions and average time between received transactions capture shifts away from regular account activity rhythms. The difference between an account's first and last transaction also highlights short-lived scam accounts. Transaction value statistics like minimum, maximum, and average value provide thresholds to identify abnormal spikes, while total number of unique counterparty addresses can expose money-mixing techniques. As fraudsters iteratively adjust techniques to avoid detection, modeling the breadth of these multivariate dynamics will enable uncovering the subtle complex anomalies. Rather than rigid rules-based identification, machine learning provides the adaptive capacity to uncover these evolving transactional irregularities. By retaining only the core set of dynamic, continuous features within a refined dataset as shown in Figure 4, advanced algorithms can effectively profile accounts and transactions to reliably flag fraudulent patterns.



```
Index(['Address', 'FLAG', 'Avg min between sent tnx',
       'Avg min between received tnx',
       'Time Diff between first and last (Mins)', 'Sent tnx', 'Received Tnx',
       'Number of Created Contracts', 'Unique Received From Addresses',
       'Unique Sent To Addresses', 'min value received', 'max value received ',
       'avg val received', 'min val sent', 'max val sent', 'avg val sent',
       'min value sent to contract', 'max val sent to contract',
       'avg value sent to contract',
       'total transactions (including tnx to create contract',
       'total Ether sent', 'total ether received',
       'total ether sent contracts', 'total ether balance'],
      dtype='object')
```

Figure 3: All the columns after initial feature reduction



```
['Address', 'FLAG', 'Avg min between sent tnx',
 'Avg min between received tnx',
 'Time Diff between first and last (Mins)',
 'Unique Received From Addresses', 'min value received',
 'max value received ', 'avg val received', 'min val sent',
 'avg val sent', 'total transactions (including tnx to create contract',
 'total ether received', 'total ether balance'],
```

Figure 4: Final Columns after data preprocessing

MODELING APPROACH

With the refined dataset of 13 engineered features, a systematic evaluation of various classical machine learning algorithms was conducted to identify the optimal fraud detection model. Models were selected to represent diverse techniques including ensemble methods like Random Forest and AdaBoost, probabilistic models such as Naive Bayes, nonlinear models including Support Vector Machine and Multilayer Perceptron, as well as tree-based approaches with Decision Tree and Gradient Boosted models. This breadth provides adaptability to the intricacies of financial fraud versus relying on any individual technique. Additionally, the modeling process involved three phases: initial base models trained on all features, base models on the 8 most important features from feature selection, and hyperparameter-tuned versions using all features to optimize predictive performance. Hyperparameter optimization leveraged an exhaustive grid search approach to identify the best configurations for each algorithm.

By evaluating models both before and after feature selection and tuning, the incremental value of each optimization technique could be quantified. Across metrics like ROC AUC, recall, and F1-scores, feature selection followed by tuning yielded the greatest improvement. This indicates the significance of eliminating noisy variables, then customizing algorithm hyperparameters.

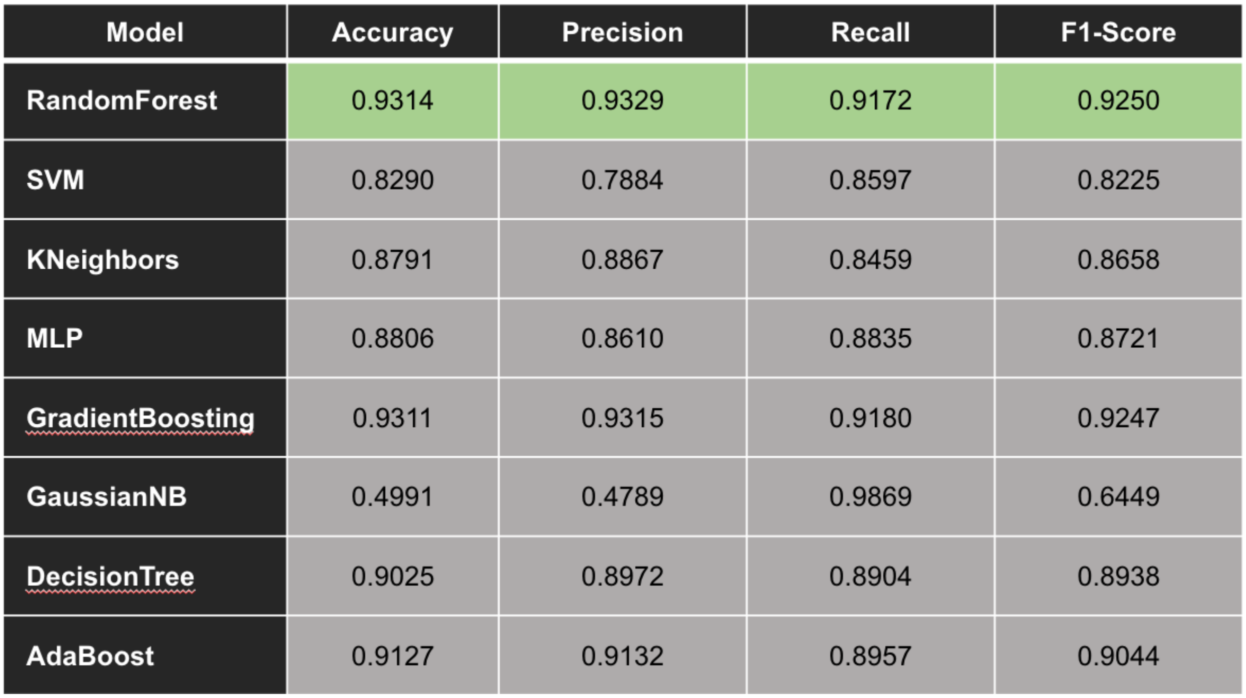
RESULTS

Model evaluation was conducted in three progressive phases to assess performance improvements from feature refinement and tuning:

1. Base models trained on the full dataset after initial feature selection
2. Base models trained on just the 8 most important features

3. Hyperparameter tuned models using all features

Hyperparameter tuning leveraged an exhaustive grid search to optimize each algorithm. Across the modeling phases, overall accuracy, precision, recall and F1 score were analyzed to provide multifaceted model quantification. For reliable cryptocurrency fraud detection, both high recall (accurate fraud flagging) and precision (few false alarms) are critical alongside general accuracy. Among all algorithms and stages, the fine-tuned Random Forest model using the full feature set performed strongest across the balanced evaluation metrics as depicted in Figure 5. The bagged decision trees leveraged non-linear decision boundaries well-suited for adapting to constantly evolving attack vectors. Precision and F1 score results showcase Random Forest's effectiveness at correctly identifying fraud cases without excessive false alerts. Furthermore, tuning hyperparameters like tree depth and number of estimators helped boost performance over base configurations. By systematically assessing incremental model improvements, Random Forest's superior capacity for capturing subtle patterns in multivariate time-series data could be definitively quantified.



Model	Accuracy	Precision	Recall	F1-Score
RandomForest	0.9314	0.9329	0.9172	0.9250
SVM	0.8290	0.7884	0.8597	0.8225
KNeighbors	0.8791	0.8867	0.8459	0.8658
MLP	0.8806	0.8610	0.8835	0.8721
GradientBoosting	0.9311	0.9315	0.9180	0.9247
GaussianNB	0.4991	0.4789	0.9869	0.6449
DecisionTree	0.9025	0.8972	0.8904	0.8938
AdaBoost	0.9127	0.9132	0.8957	0.9044

Figure 5: Evaluation scores of models after hyperparameter tuning using all 13 features.

In summary, these thorough evaluation practices incorporating multi-faceted performance measures validate Random Forest as the optimal fraud detection solution fitting the intricacies of ever-changing blockchain transaction behavior. The ensemble method combines diverse decision boundaries to uncover emerging fraudulent patterns across evolving cryptocurrency transactions with both accuracy and precision.

FUTURE IMPROVEMENTS

While the tailored Random Forest model provided effective fraud detection performance, additional refinements may yield even stronger results and expanded applications. One area of improvement is incorporating network analysis methods to build graphs of account connections and transfers. Identifying high-risk clusters based on transaction flows between suspected accounts could further boost detection rates. Additionally, online learning techniques could allow the model to continuously update based on new examples without full retraining. This self-adapting capability would ensure the detector stays current amidst evolving attack strategies.

On the deployment side, integrating the algorithms directly with cryptocurrency exchanges or blockchain analytic tools would enable preemptive alerts to help mitigate financial losses and negative user impacts. Beyond Ethereum, customizing the approach across different blockchain implementations and currencies may uncover additional niche patterns particular to those ecosystems. Overall, the established methodology provides a robust foundation for numerous enhancements in model performance, applicability, and real-world protection against cryptocurrency fraud.

REFERENCES

[1] Alarab, I., & Prakoonwit, S. (2023). Graph-based LSTM for anti-money laundering: Experimenting temporal graph convolutional network with bitcoin data. *Neural Processing Letters*, *55*(1), 689-707.

[2] Bhowmik, M., Chandana, T. S. S., & Rudra, B. (2021, April). Comparative study of machine learning algorithms for fraud detection in blockchain. In *2021 5th international conference on computing methodologies and communication (ICCMC)* (pp. 539-541). IEEE.

[3] Martins, A. P., & Brito, M. A. FRAUD DETECTION AND ANTI-MONEY LAUNDERING APPLYING MACHINE LEARNING TECHNIQUES IN CRYPTOCURRENCY TRANSACTIONAL GRAPHS.

[4] Umer, Q., Li, J. W., Ashraf, M. R., Bashir, R. N., & Ghous, H. (2023). Ensemble Deep Learning Based Prediction of Fraudulent Cryptocurrency Transactions. *IEEE Access*.

[5] Zkik, K., Sebbar, A., Fadi, O., Kamble, S., & Belhadi, A. (2023). Securing blockchain-based crowdfunding platforms: an integrated graph neural networks and machine learning approach. *Electronic Commerce Research*, 1-37.

[6] Ethereum Fraud Detection Dataset. (2021, January 3). Kaggle. https://www.kaggle.com/datasets/vagifa/ethereum-frauddetection-dataset

[7] Ethereum Transactions Dataset.
https://etherscan.io/exportData