Cryptocurrency Fraud Detection

Team 20
Yukti Bishambu (ybishambu3@gatech.edu),
Randall A Escalante (rescalante6@gatech.edu)

Submission date: April 21st, 2025

Course number: **ISYE 7406**
Course name: **Data Mining and Statistical Learning**

Abstract

Fraud detection in cryptocurrency transactions is a critical challenge due to the pseudonymous nature of blockchain technology. This study investigates the effectiveness of machine learning algorithms in classifying fraudulent and legitimate Ethereum accounts based on transaction patterns, ERC20 token activities, and account behaviors. The project focuses on three main objectives: evaluating classification performance of machine learning algorithms, identifying predictive features, and comparing different fraud detection methods namely logistic regression, random forest, and gradient boosting model. The dataset is preprocessed extensively, involving the removal of null and zero-variance columns, duplicate record handling, missing value treatment, and correlation-based dimensionality reduction. Our analysis reveals that ensemble models outperform the baseline logistic regression model, achieving significantly higher F1-score. Feature importance analysis highlights key indicators of fraudulent behavior, such as time span between an account's first and last transaction and total amount of Ether received in an account. The findings demonstrate the potential of supervised machine learning in enhancing security within blockchain ecosystems while addressing challenges posed by imbalanced data and high-dimensional feature spaces.

Key Words: Cryptocurrency, Blockchain, Fraud Detection, Machine Learning, Logistic Regression, Random Forest, Gradient Boosting, Precision, Recall, F1 Score

1. Introduction

Blockchain technology has gained significant attention in recent years, with Ethereum being one of the most widely used platforms for decentralized applications and smart contracts. However, its widespread adoption in financial applications has introduced critical security challenges, primarily due to the anonymity, transparency, and irreversibility of blockchain transactions. A significant concern is the presence of fraudulent accounts, which exploit these features to perform illicit activities without detection. This project focuses on detecting fraudulent Ethereum accounts by analyzing transaction patterns and behavioral data, aiming to improve trust and security within the Ethereum ecosystem.

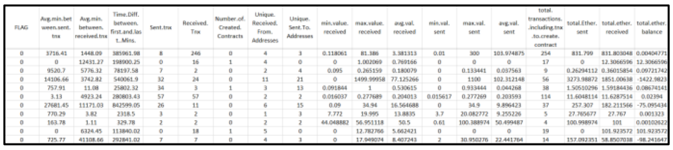
The core data mining challenge lies in accurately identifying fraudulent behavior in a highly imbalanced and complex dataset where malicious activities are often subtle and evolving. Traditional rule-based systems are ineffective, prompting the need for more robust approaches. To address this, we employ supervised machine learning techniques to uncover hidden patterns indicative of fraud. Feature engineering and model evaluation are critical steps in developing effective detection mechanisms.

Through this project, we gained practical insights into applying machine learning to real-world blockchain data, understanding the nuances of fraud detection, and evaluating model performance in an imbalanced setting.

The report is organized as follows: we begin with a detailed overview of the dataset and preprocessing steps, followed by a discussion of the machine learning methods used. We then present our results and conclude with key takeaways and future work.

2. Data Source and Problem Statement

The dataset used in this study has been obtained from Kaggle, where Ethereum fraud detection data is publicly available. Each row in the dataset represents a unique Ethereum account. The dataset consists of 9.8K observations and 51 attributes including average time between sent and received ether transactions for a given account address, total number of sent and received ether transactions, number of unique addresses they are sent to or received from, the cumulative amount that has been sent or received, the cumulative ether amount sent or received via smart contract in exchange for ERC-20 tokens, etc. To set some context, Ether (ETH) is the native cryptocurrency of the Ethereum network, serving as the 'fuel' for deploying and operating smart contracts. ERC-20 tokens are distinct tokens created on Ethereum network through a smart contract that defines the token's properties. The dataset also has labels indicating whether an account is fraudulent (label=1) or legitimate (label=0). It requires extensive preprocessing before any machine learning algorithm can be applied.



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	total.transactions.including.tnx.to.create.contract	total.Ether.sent	total.ether.received	total.ether.balance
0	3716.41	1448.09	385961.98	8	246	0	4	3	0.118061	81.386	3.381313	0.01	300	103.974875	254	831.799	831.803048	0.00404771
0	0	12431.27	198900.25	0	16	1	4	0	0	1.002069	0.769166	0	0	0	17	0	12.3066596	12.3066596
0	9520.7	5776.32	78197.58	7	2	0	2	4	0.095	0.265159	0.180079	0	0.133441	0.037563	9	0.26294112	0.36015854	0.09721742
0	14106.66	3742.82	540061.9	32	24	0	11	21	0	1499.99958	77.125266	0	1100	102.312148	56	3273.98872	1851.00638	-1422.9823
0	757.91	11.08	25802.32	34	3	1	3	13	0.091844	1	0.530615	0	0.933444	0.044268	38	1.50510296	1.59184436	0.08674141
0	3.13	4923.24	280803.43	57	57	0	2	2	0.016037	0.277689	0.204013	0.015617	0.277269	0.203593	114	11.6048114	11.6287514	0.02394
0	27681.45	11171.03	842599.05	26	11	0	6	15	0.09	34.94	16.564688	0	34.9	9.896423	37	257.307	182.211566	-75.095434
0	770.29	3.82	2318.5	3	2	0	1	3	7.772	19.995	13.8835	3.7	20.082772	9.255226	5	27.765677	27.767	0.001323
0	163.78	1.11	329.78	2	2	0	2	2	44.048882	56.951118	50.5	0.61	100.388974	50.499487	4	100.998974	101	0.00102622
0	0	6324.45	113840.02	0	18	1	5	0	0	12.782766	5.662421	0	0	0	19	0	101.923572	101.923572
0	725.77	41108.66	292841.02	7	7	0	4	3	0	17.949074	8.407243	2	30.950276	22.441764	14	157.092351	58.8507038	-98.241647

Fig. 1: Data snippet containing few features with their values.

Fraudulent accounts significantly harm the Ethereum ecosystem by enabling financial scams such as Ponzi schemes, phishing attacks, and rug pulls, leading to substantial monetary losses for users and undermining trust in decentralized finance (DeFi). These malicious actors also congest the network with spam transactions and fake token sales, driving up gas fees and degrading performance for legitimate users. Beyond immediate financial and operational impacts, persistent fraud tarnishes Ethereum's reputation, discouraging institutional adoption and inviting stricter regulatory scrutiny. Collectively, these effects erode the security, efficiency, and credibility essential for Ethereum's long-term growth and mainstream acceptance.

This study explores three key questions. First, we aim to determine whether machine learning algorithms can accurately classify fraudulent and legitimate Ethereum accounts. Second, we seek to identify the most influential features contributing to fraudulent account detection. Lastly, we compare different fraud detection techniques to evaluate their effectiveness in identifying suspicious accounts.

3. Methodology

3.1. Exploratory Data Analysis and Pre-processing

The dataset undergoes comprehensive preprocessing before model application. Based on the initial summary of the data, preliminary feature selection was performed by removing seven null or zero columns along with two index columns. Two non-numeric columns related to token type sent and received were also removed as more than 70% of the records in those columns were NULL and the remaining values can potentially increase the complexity of the machine learning models without adding significant value to the results. Duplicate addresses were eliminated while preserving unique transaction patterns. Missing values in ERC20 token related features were replaced with zeros, interpreting these as absence of those specific type of transactions. Variance calculation revealed that features like cumulative amount of Ether received or sent in exchange of ERC-20 tokens by a specific address and the time difference between first and last transactions for a given address have very high variance across the dataset which indicates that these features have high potential of explaining the variation in the target variable as well. Log-transformed feature distributions (non-zero values only) were compared between legitimate and fraudulent accounts to detect anomalous patterns.

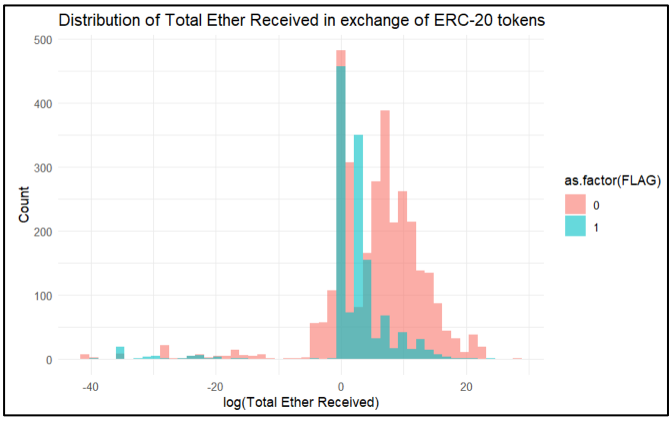


Fig. 2: Distribution of log scaled time cumulative amount of ETH received in exchange for ERC-20 tokens.

Fig.2 reveals two key patterns. Firstly, a small fraction of accounts (with non-zero values) holds cumulative amounts below 1 ETH, reflected by negative log-scale values on the x-axis. These maybe new or developing legitimate accounts or failed fraud attempts or abandoned scam accounts. Secondly, sharp peak near zero indicates that the vast majority of accounts - both legitimate and fraudulent - cluster around a cumulative balance of 1 ETH. This suggests possible "minimum balance" behavior, where fraudulent accounts maintain this amount to appear legitimate while avoiding scrutiny. Such accounts typically show limited ETH accumulation through ERC-20 exchanges while legitimate accounts tend to receive significantly larger cumulative amounts of Ether. Please note that extreme points are treated as normal datapoints rather than outliers because financial data is inherently volatile, and removing them could result in loss of critical information.

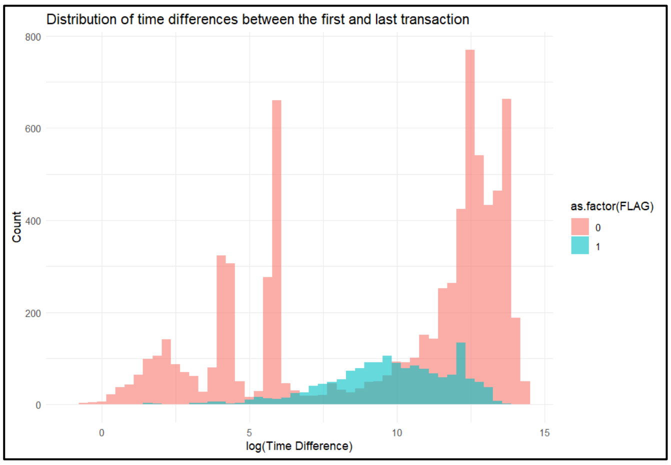


Fig. 3: Distribution of log scaled time difference between an account's first and last transaction.

Fig. 3 also reveals distinct patterns. Legitimate accounts show a polarized distribution, clustering at either very short (rapid activity) or very long (sustained activity) time intervals. Fraudulent accounts predominantly fall in the middle range, suggesting neither immediate churn nor long-term engagement.

Similar behavioral patterns emerged for other features with high variance. Further refinement of data included removing five low-variance features because they provide little to no useful information for distinguishing between classes. They remain nearly constant across all observations and don't help the model learn meaningful patterns, potentially adding noise and increasing computation without improving performance. This was followed by eliminating six randomly selected features from the set of nine highly correlated features (with absolute correlation coefficients exceeding 0.85), resulting in a final set of 27 predictive variables. All numerical features were normalized using center-scaling transformation before correlation calculation.

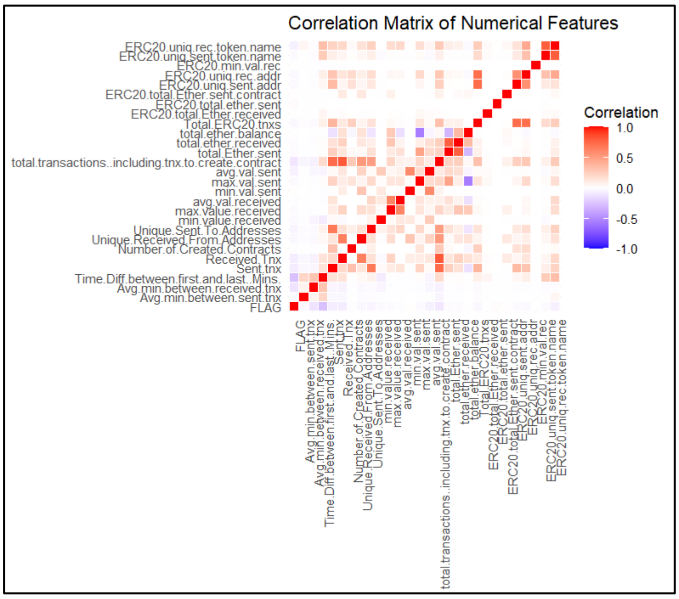


Fig. 4: Correlation heatmap of selected features.

Exploratory analysis also revealed a class imbalance with only 22.2% of the data representing fraudulent accounts. Handling class imbalance is crucial because most models assume equal class distribution. If one class dominates, the model may become biased toward it, leading to poor performance in detecting the minority class - especially critical in fraud detection where catching rare cases matters most. To address the 3.5:1 class imbalance and prevent bias towards majority class, SMOTE (Synthetic Minority Over-sampling Technique) was applied to generate synthetic samples of the underrepresented fraud cases. Applying SMOTE on normalized data ensures realistic synthetic samples and avoids data leakage. This was followed by strategic downsampling of the majority class (legitimate accounts) to reduce its volume while preserving key patterns. This hybrid method achieved balanced class distributions without excessively inflating the dataset size (as pure SMOTE would) or losing critical information (as pure downsampling might). By maintaining dataset representativeness and mitigating overfitting risks, it created an optimal foundation for model training and evaluation.

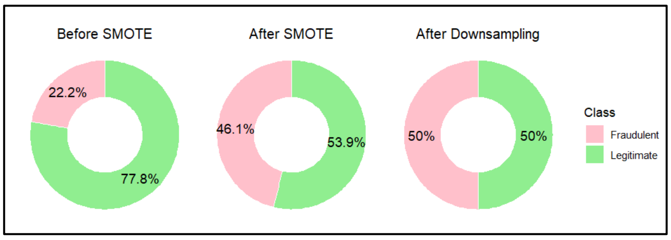


Fig. 5: Results from hybrid resampling strategy addressing class imbalance.

With the data now prepared for modeling through feature selection and balancing, we proceed to the next critical step which is splitting the dataset into separate training (80%) and testing (20%) subsets. This division enables us to train the model on one portion of the data while reserving an independent set for evaluation, ensuring an objective assessment of the model's performance on unseen data.

3.2 Machine Learning Algorithms

We begin with a baseline classification model using logistic regression, a statistical approach that predicts class probabilities by applying a logit function to transform linear combinations of predictors into probabilities. This model performs optimally when the relationship between features and the target variable is linear, though its effectiveness may diminish with non-linear patterns.

Our initial model was trained on the imbalanced dataset to establish a performance benchmark before addressing class imbalance. The predicted probabilities were converted to class labels using a 0.3 decision threshold to account for the sensitive nature of fraud detection. Based on the p-values, the model identified two particularly influential factors in detecting fraudulent accounts - the time span between an account's first and last transaction, and the average amount of Ether received from that account. The coefficient values showed that accounts with longer transaction histories or higher average transfer values were statistically less likely to be fraudulent.

When the same model was trained on balanced data, an additional important factor emerged - the total number of outgoing transactions. Accounts with more outgoing transactions also tended to be less likely to be fraudulent, reinforcing the earlier trends observed in the model trained on imbalanced data.

Having established our baseline model, we now progress to more sophisticated approaches, beginning with a Random Forest classifier. This ensemble method constructs numerous decision trees, each trained on random subsets of both the data and features, to predict the target variable. Unlike simpler models, Random Forest excels at capturing complex, non-linear relationships between variables while naturally resisting overfitting through its built-in randomness.

To optimize performance, we systematically test various combinations of key hyperparameters, such as the number of trees, minimum samples per leaf, and number of randomly selected features considered for splitting at each node. Each configuration is evaluated using 5-fold cross-validation to ensure reliable performance estimates. The final model, trained on the balanced dataset, uses the best combination, which is 200 trees, 3 observations per leaf, and 9 random features per split, yielding the lowest cross-validation error. This hyperparameter set ensures stable predictions without unnecessary complexity and maintains diversity among trees while allowing meaningful splits. This balance ensures the model generalizes well to new data while keeping computations efficient.

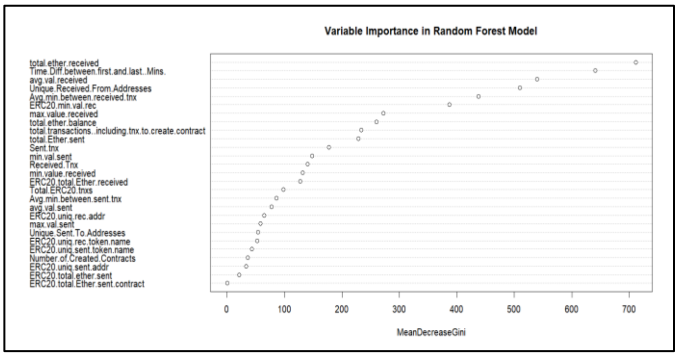


Fig. 6: Feature Importance plot for Random Forest model

While powerful, this approach comes with reduced interpretability compared to linear models. The model estimates feature importance by tracking how frequently and significantly each variable is used to split nodes across all decision trees, weighted by how much these splits improve classification purity. This identifies the most influential predictors, but it doesn't indicate whether higher or lower values correlate with fraud risk. Fig. 6 highlights that the total amount of Ether received by an account and the duration between its first and last transaction are the two most significant predictors of fraudulent activity. While other features contribute to the model's decision-making process, their relatively lower importance scores imply they play more supplementary roles. This suggests these two factors are particularly valuable for detection, though additional analysis would be needed to determine the exact nature of their relationships with fraud.

Like Random Forest, Gradient Boosting builds multiple decision trees, but with a key difference. Instead of training trees independently, it constructs them sequentially, with each new tree specifically correcting errors made by the previous ones. This allows the model to focus on difficult-to-classify examples, refining its predictions step by step. While boosting is powerful and can lead to high accuracy, it requires careful tuning to avoid overfitting, as its iterative nature can make it sensitive to noise in the data.

To ensure robustness, we perform 10-fold cross-validation across a range of hyperparameter combinations, including shrinkage factor, tree depth, and number of trees. The grid is designed based on commonly effective ranges found in literature and prior empirical results - shrinkage factors between 0.01 and 0.3, tree depths from 3 to 10 to balance complexity and overfitting risk, and tree counts from 100 to 500 for performance stability. The optimal configuration of 400 trees, depth of 10, and shrinkage factor of 0.1 is selected based on the lowest cross-validation error, offering the best trade-off between accuracy and generalization.

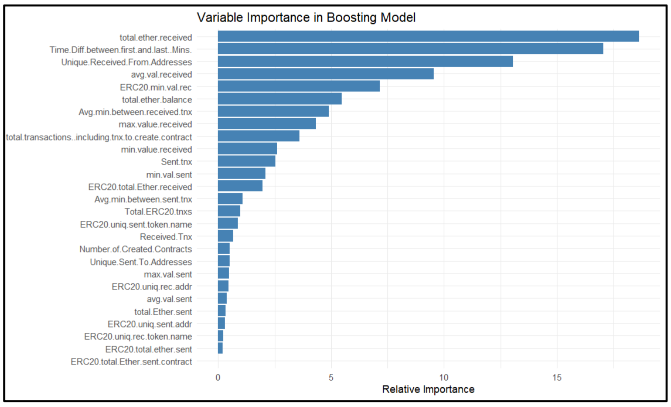


Fig. 7: Feature Importance plot for Random Forest model

Similar to random forest, gradient boosting model also identifies the total amount of Ether received and the transaction span as key predictors of fraud, based on feature importance and doesn't reveal the direction of their relationship with fraud risk. Other features contribute less, indicating they play more minor roles.

Note that due to the longer training time required for Random Forest models, 10-fold cross-validation was not feasible, as it significantly increased computational time. In contrast, Gradient Boosting proved to be faster and more efficient, allowing for quicker model evaluation while still delivering strong performance.

To formally compare model performance, 10-fold cross-validation was performed, followed by the application of both paired t-tests and Wilcoxon tests on the evaluation metrics across the folds. These statistical tests were chosen to account for the fact that the same data splits were used across models. By pairing metric values fold-by-fold, the analysis isolates the differences in model performance while controlling for variability introduced by specific train-test partitions. This approach ensures that any observed differences in model performance are not due to random chance, providing a robust basis for comparison.

4. Analysis and Results

The logistic regression model on imbalanced dataset achieved an overall accuracy of 66.4% (off-diagonal sum in the first confusion matrix in fig. 8) on the test set. However, accuracy alone does not reveal performance across individual categories. For example, the model flagged 46.6% of accounts as fraudulent, but only 17.8% (upper right box of the first confusion matrix in fig. 8) were true fraud cases. This ratio between true fraud and predicted fraud yields the metric called precision which is 38.2% in this case. Similarly, the ratio of correctly identified fraudulent cases with the actual 22.6% fraud accounts gives us a metric known as recall or sensitivity which is 78.7% in this case. While the model detects a decent percentage of truly fraud cases, its low precision means many false alarms, that is legitimate accounts flagged as fraud (lower right box of the first confusion matrix in fig. 8). This trade-off is typical for imbalanced dataset, where the model prioritizes the majority class.

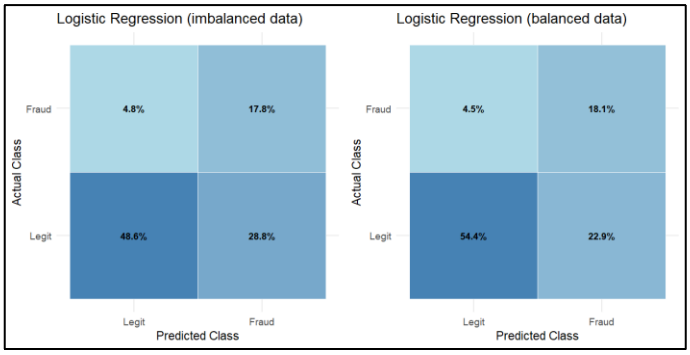


Fig.8: Confusion matrix for Logistic Regression models for imbalanced and balanced datasets.

On the other hand, the logistic regression model trained on balanced dataset classified a total of 72.6% observations accurately in the test set, with a recall of 80% for the fraud class. We see a significant increase in precision as 44.2% of the accounts predicted as fraudulent were truly fraudulent. This clearly shows that balancing the data reduced bias allowing the model to better distinguish fraud. The model learned more nuanced patterns from the additional synthetic data and could capture correct classes while giving fewer false alarms.

The random forest model outperformed both with 97.2% accuracy, 96.3% recall, and 97.5% precision, likely capturing non-linear patterns in the features and effectively learning patterns for both fraud and legitimate accounts. It caught almost all fraud while rarely mislabeling legitimate accounts.

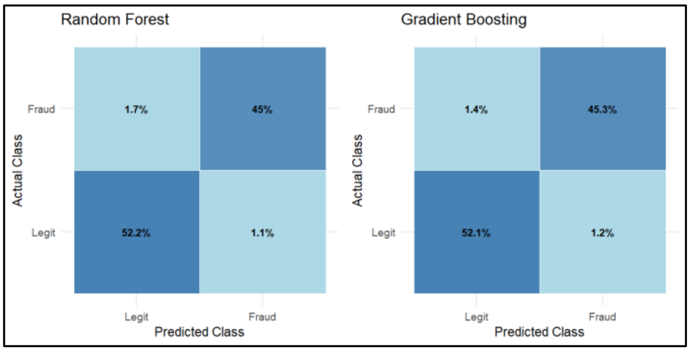


Fig.9: Confusion matrix for Ensemble models for balanced datasets.

Similar to random forest, the gradient boosting model achieved 97.4% accuracy, 96.9% recall, and 97.3% precision, demonstrating near-perfect balance in detecting fraud while minimizing false positives, making it the most robust approach for this task. A recall greater than 96% means <4% of fraud cases slip through, significantly reducing financial losses. Meanwhile, with precision >97%, the model flags only ~3% legitimate accounts as fraud which is critical for avoiding unnecessary account freezes or customer friction.

Model	Accuracy	F1-Score
Logistic Regression (Imbalanced)	66.4%	51.4%
Logistic Regression (Balanced)	72.6%	56.9%
Random Forest	97.2%	96.8%
Gradient Boosting	97.4%	97.1%

Table 1: Evaluation metrics on test data

The F1 scores of all implemented models are collated in Table 1, providing a unified measure of their ability to balance precision (minimizing false positives) and recall (capturing true fraud). As the harmonic mean of these metrics, the F1 score penalizes extreme disparities between the two making it ideal for evaluating fraud detection systems where both over flagging legitimate accounts and missing fraudulent activity carry high costs. For instance, boosting model's F1 score of 97.1% reflects its robust equilibrium, whereas logistic regression's lower F1 reveals a trade-off skewed toward one metric. This comparative analysis underscores why F1 is a critical benchmark for model selection in imbalanced classification tasks. For deployment, we can prioritize models with the highest F1 scores unless business constraints demand favoring precision (e.g., reducing false alarms) or recall (e.g., catching all fraud).

While the F1 scores offer valuable insights into the relative performance of the models, it is essential to determine whether the observed differences in these scores are statistically significant. To rigorously assess this, a 10-fold cross-validation was conducted for all three models using balanced dataset. In each fold, the models were trained using fixed hyperparameters - selected via prior cross-validation as detailed in the methodology section - and evaluated on held-out test sets. The F1 scores from each fold were recorded, providing a robust estimate of model performance under different data splits. On average, the logistic regression model achieved a mean F1 score of 37.8%, while the random forest and gradient boosting models achieved 97.3% and 97.1%, respectively, across the folds. These scores reflect each model's typical performance and were then paired across the models for each fold, creating a set of paired observations that represent the performance of each model on the same data splits.

Paired t-tests and Wilcoxon signed-rank tests were subsequently applied to the cross-validated F1 scores to assess whether the observed performance differences between models were statistically significant. As shown in Table 2, the p-values for both tests comparing logistic regression to random forest and logistic regression to gradient boosting are well below the standard significance threshold of 0.05. This provides strong evidence that the ensemble methods significantly outperform the baseline logistic regression model in terms of F1 score. These results are consistent across both parametric (t-test) and non-parametric (Wilcoxon test) testing approaches, further reinforcing the robustness of the observed performance gap.

Model Pair	t-test p-value	Wilcoxon test p-value
LR (bal.) - RF	0.0004658	0.001953
LR (bal.) - GBM	0.0004612	0.001953
RF - GBM	0.137	0.1602

Table 2: p-values from paired t-tests and Wilcoxon tests on F1-scores

In contrast, when comparing the two ensemble models—random forest and gradient boosting—the p-values from both the paired t-test and the Wilcoxon test are substantially higher than 0.05. This indicates that the difference in their mean F1 scores is not statistically significant, suggesting comparable performance between the two. Together, these findings validate the superiority of ensemble methods over logistic regression for this classification task, while showing that both random forest and boosting models deliver similarly strong results.

5. Conclusions

This study set out to examine the feasibility and effectiveness of using machine learning models for fraud detection on Ethereum accounts. Specifically, the investigation focused on assessing model performance in accurately classifying accounts as fraudulent or legitimate, identifying the most influential features contributing to this classification, and comparing different machine learning approaches to determine the most effective technique for the task.

The results clearly demonstrate that machine learning models - especially ensemble methods - can successfully detect fraudulent activity with high precision and recall. Both the random forest and gradient boosting models achieved over 97% mean accuracy and mean F1 scores above 97%, indicating excellent performance in distinguishing between fraudulent and legitimate accounts. In contrast, the logistic regression model, particularly when trained on the imbalanced dataset, struggled to perform reliably, exhibiting significantly lower precision and F1 scores. Even after balancing the dataset, its performance, while improved, remained far behind the ensemble methods.

The poor performance of logistic regression can be attributed to its linear nature, which limits its ability to capture complex, non-linear relationships within the data. Fraudulent behavior often involves intricate interactions between features, which a simple linear model cannot capture effectively. Additionally, logistic regression is more sensitive to imbalanced data, tending to favor the majority class without stronger signals from minority class patterns. In contrast, ensemble methods like Random Forest and Gradient Boosting aggregate multiple decision trees, enabling them to model non-linear interactions, higher-order feature combinations, and outliers more robustly. These methods can better learn subtle patterns of fraud spread across multiple features, especially in high-dimensional or noisy data, leading to superior classification performance.

Feature importance analysis revealed that the amount of Ether received and the transaction span were consistently among the most influential predictors of fraud across models. From a business standpoint, this implies that transactional behavior carries strong signals of legitimacy. Specifically, analysis from the logistic regression model - where coefficients reveal directionality - suggested that accounts receiving higher amounts of Ether and operating over a longer time span were less likely to be fraudulent. This aligns with real-world expectations that legitimate users like exchanges or long-term traders often exhibit stable, high-volume activity, while fraudulent accounts may display abrupt, short-lived patterns with smaller transaction sizes to avoid detection.

However, for ensemble models, feature importance scores only indicate how strongly a feature influences predictions and not the direction of the relationship. This reflects the black-box nature of these models, making it harder to interpret how exactly the features affect fraud prediction in ensemble models. To understand whether a feature increases or decreases the chance of fraud, we would need to use extra tools like SHAP (SHapley Additive exPlanations). Using these kinds of explainability methods in future work can help us better understand the model's decisions and make the results more useful for people who need to act on them.

Finally, paired statistical testing through both t-tests and Wilcoxon signed-rank tests confirmed that the performance differences between logistic regression and ensemble models were statistically significant. However, the difference between random forest and gradient boosting was not significant, indicating that either model could be effectively used in deployment depending on secondary factors. If faster training time and scalability are key concerns, gradient boosting might be the better option due to its more efficient training process. However, if ease of tuning and slightly better interpretability are prioritized, random forest may be preferred - despite its higher computational cost.

In summary, this study affirms that ensemble-based machine learning methods are highly capable of identifying suspicious Ethereum accounts. They provide consistent, significant improvements in predictive accuracy, making them a robust and scalable solution for fraud detection systems. Future work could focus on combining supervised learning with unsupervised anomaly detection to spot new fraud patterns that the model hasn't seen before. Deep learning approaches can also be explored. Collaborating with Ethereum users to test models on real transaction data would help ensure the models work well in the real world. Overall, this project shows that data-driven methods can help make blockchain more secure and trustworthy, as long as models continue to adapt to new fraud techniques.

6. Lessons learnt

Data quality is crucial, as a model's performance depends heavily on the quality of the data it is trained on. Handling class imbalance was essential to improve model performance, with SMOTE improving sensitivity but requiring downsampling to prevent overfitting. Trying to capture all fraud accounts led to more false positives, highlighting the need for a balanced approach between recall and precision to keep fraud detection actionable. Feature interpretation is vital for creating explainable AI systems, required for compliance. Despite Ethereum's pseudonymity, transaction patterns and account behaviors were effective in identifying fraudulent accounts, showing that anonymity doesn't equate to being untraceable. As fraudsters adapt their tactics, such as mimicking legitimate behavior, continuous model retraining will be necessary to maintain performance and stay ahead of evolving fraud patterns.

7. Bibliography

1. Ethereum Foundation Documentation. *https://ethereum.org/en/developers/docs/intro-to-ethereum/*
2. OnTheMarkData. *Making Sense of Ethereum Data for Analytics*. Medium, 2021.
3. Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction* (2nd ed.). Springer.
4. Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). *SMOTE: Synthetic Minority Over-sampling Technique*. Journal of Artificial Intelligence Research, 16, 321–357.
5. Lundberg, S. M., & Lee, S.-I. (2017). *A Unified Approach to Interpreting Model Predictions*. Advances in Neural Information Processing Systems (NeurIPS).

8. Credits

We acknowledge the use of R and several open-source packages - including randomForest, gbm, caret, and ggplot2 - for data analysis and modeling. This project also benefited from course lecture videos, peer discussions, feedback from teaching assistants, and clarification and debugging support from publicly available AI tools such as ChatGPT, Perplexity, etc.