

Ethereum Transaction Success Clasification with Machine Learning and XAI

Nurul Adi Prawira^{*1}, Yan Rianto²

^{1,2}Department of Computer Science, Nusa Mandiri University, West Java, Indonesia

e-mail: ^{*}14240013@nusamandiri.ac.id, ²yan.yrt@nusamandiri.ac.id

Abstrak

Penelitian ini mengembangkan model untuk mengklasifikasi keberhasilan transaksi Ethereum dengan memanfaatkan Machine Learning dan Explainable Artificial Intelligence. Dataset diperoleh dari Google BigQuery Public Datasets dan setelah proses pembersihan terdiri atas 111.617 transaksi. Tahapan penelitian meliputi preprocessing, feature engineering, evaluasi penanganan ketidakseimbangan kelas, pembangunan Logistic Regression, Random Forest, XGBoost, dan LightGBM, hyperparameter tuning, Stratified Cross Validation, AutoML menggunakan FLAML, interpretasi SHAP, dan McNemar Test. Hasil menunjukkan bahwa dataset tanpa balancing memberikan performa terbaik. Random Forest menjadi model konvensional terbaik dengan Accuracy 98,37%, Precision 98,47%, Recall 99,88%, F1-Score 99,17%, dan ROC-AUC 93,93%. FLAML memilih XGBoost sebagai model terbaik dengan Accuracy 98,28%, Precision 98,54%, Recall 99,71%, F1-Score 99,12%, dan ROC-AUC 94,80%. Analisis SHAP menunjukkan gasUsed, gas_efficiency, dan gasLimit sebagai fitur paling berpengaruh. McNemar Test menghasilkan p-value 0,377494 sehingga tidak ditemukan perbedaan prediksi yang signifikan antara Random Forest dan model FLAML. Hasil ini menunjukkan bahwa pendekatan Machine Learning dapat memberikan performa klasifikasi yang tinggi dan tetap dapat diinterpretasikan pada data transaksi Ethereum.

Kata kunci— Ethereum, Machine Learning, Random Forest, FLAML, SHAP

Abstract

This study develops a model for clasification Ethereum transaction success using Machine Learning and Explainable Artificial Intelligence. The dataset was obtained from Google BigQuery Public Datasets and contained 111,617 transactions after data cleaning. The research pipeline included preprocessing, feature engineering, class-imbalance evaluation, Logistic Regression, Random Forest, XGBoost, and LightGBM modeling, hyperparameter tuning, Stratified Cross Validation, FLAML-based AutoML, SHAP interpretation, and the McNemar Test. The results show that the original unbalanced dataset produced the best performance. Random Forest was the best conventional model, achieving 98.37% Accuracy, 98.47% Precision, 99.88% Recall, 99.17% F1-Score, and 93.93% ROC-AUC. FLAML selected XGBoost as its best estimator, achieving 98.28% Accuracy, 98.54% Precision, 99.71% Recall, 99.12% F1-Score, and 94.80% ROC-AUC. SHAP identified gasUsed, gas_efficiency, and gasLimit as the most influential features. The McNemar Test produced a p-value of 0.377494, indicating no statistically significant difference between Random Forest and the FLAML model. The findings demonstrate that Machine Learning can provide high classification performance while maintaining model interpretability for Ethereum transaction data.

Keywords— Ethereum, Machine Learning, Random Forest, FLAML, SHAP

1. INTRODUCTION

Ethereum is one of the most widely used blockchain platforms for cryptocurrency transactions and smart contracts. The increasing number of transactions recorded on the Ethereum network generates large volumes of structured transaction data that can be analyzed to understand transaction behavior. One important issue is transaction success, since failed transactions may result in wasted transaction fees and reduced transaction efficiency. Therefore, identifying patterns associated with successful and failed transactions is important for developing data-driven transaction analysis approaches.

Machine Learning has been widely applied to blockchain analytics because transaction data contain various attributes that can be used to identify complex patterns. Previous studies have demonstrated the effectiveness of Machine Learning for Ethereum-related tasks, including transaction analysis, confirmation-time prediction, and transaction classification. Oliveira et al. compared Logistic Regression, Decision Tree, Support Vector Machine, and Random Forest for Ethereum transaction analysis and reported that Random Forest provided the best performance, with `gasUsed` identified as an influential transaction attribute [1]. Singh and Hafid applied Random Forest and Multilayer Perceptron to Ethereum transaction data and showed that Machine Learning could improve transaction confirmation-time prediction compared with conventional statistical approaches [2].

Other studies have emphasized the broader application of Machine Learning and feature engineering in blockchain transaction analysis. Kayikci and Khoshgoftaar reviewed the integration of blockchain and Machine Learning and highlighted the potential of Machine Learning for analyzing blockchain data [3]. Jatoth et al. demonstrated the effectiveness of feature engineering and ensemble learning for blockchain transaction classification [4]. Crisostomo et al. further reviewed Machine Learning approaches for Ethereum-related analysis, highlighting the growing application of data-driven methods in blockchain environments [5]. These findings indicate that transaction attributes contain useful information that can be exploited by classification models.

The development of Machine Learning methods has subsequently encouraged the use of ensemble algorithms such as Random Forest, XGBoost, and LightGBM [6]–[8]. These algorithms are capable of modeling nonlinear relationships among transaction attributes and have demonstrated competitive performance on structured and blockchain-related datasets. These algorithms are capable of modeling nonlinear relationships among transaction attributes and have demonstrated competitive performance on structured and blockchain-related datasets. However, existing studies on Ethereum transaction analysis have generally focused on improving predictive performance, while the integration of automated model selection, model interpretability, and statistical comparison remains relatively limited. In particular, previous research on Ethereum transaction success has not comprehensively combined conventional Machine Learning models with AutoML and Explainable Artificial Intelligence (XAI) within a single experimental framework.

Another challenge in Ethereum transaction classification is the imbalance between successful and failed transactions. Successful transactions generally constitute the majority class, while failed transactions represent a substantially smaller proportion of the dataset. Under such conditions, Accuracy alone may not adequately describe classifier performance because a model can obtain high Accuracy by favoring the majority class. Therefore, evaluation should consider multiple metrics, particularly Precision, Recall, F1-Score, and ROC-AUC. In addition, different strategies for handling imbalanced data should be evaluated empirically rather than assuming that oversampling or undersampling will always improve model performance.

Model interpretability is also important when Machine Learning is applied to blockchain transaction analysis. A model with high predictive performance does not necessarily explain

which transaction attributes contribute to its decisions. Explainable Artificial Intelligence provides methods for examining the contribution of individual features to model predictions. SHapley Additive exPlanations (SHAP), for example, can quantify the contribution of individual features to model predictions and provide a global interpretation of tree-based Machine Learning models [10], [11]. The use of XAI can therefore provide additional insight into the transaction characteristics that are most influential in the classification process.

Automated Machine Learning (AutoML) provides another complementary approach by automating the selection of algorithms and hyperparameters. FLAML is a lightweight AutoML framework designed to efficiently search for suitable learners and hyperparameter configurations [9]. Although AutoML has been increasingly used in Machine Learning research, its application to Ethereum transaction-success classification remains limited. Furthermore, differences in predictive metrics between two models do not necessarily indicate that their prediction behavior is statistically different. A statistical test such as the McNemar Test can be used to compare paired classification outcomes and determine whether the observed difference is statistically significant [15]. The combination of AutoML and statistical testing therefore provides a more comprehensive evaluation than comparing performance metrics alone.

Based on these limitations, this study proposes a comprehensive framework for predicting Ethereum transaction success using Machine Learning and Explainable Artificial Intelligence. The study compares Logistic Regression, Random Forest, XGBoost, and LightGBM, followed by hyperparameter tuning and Stratified Cross Validation. Several class-imbalance strategies, including RandomOverSampler, RandomUnderSampler, and SMOTE, are also evaluated. FLAML AutoML is subsequently used to obtain an automatically optimized model and compare its performance with the conventional Machine Learning approach. SHAP is employed to identify the most influential transaction features, while the McNemar Test is used to determine whether differences between the best conventional model and the AutoML model are statistically significant.

The contribution of this study is therefore not limited to obtaining a high classification score. The proposed framework integrates feature engineering, class-imbalance evaluation, conventional Machine Learning, hyperparameter optimization, Stratified Cross Validation, AutoML, explainable Machine Learning, and statistical significance testing into a single evaluation pipeline. This approach provides both predictive and interpretative evidence regarding Ethereum transaction-success classification and addresses limitations identified in previous studies.

2. METHODS

The research methodology consists of a sequential experimental workflow, starting from dataset preparation and preprocessing, followed by feature engineering and stratified dataset splitting. The prepared data are then used for baseline Machine Learning modeling and class-imbalance comparison. The selected configuration is subsequently optimized using RandomizedSearchCV and evaluated using Stratified Cross Validation. FLAML AutoML is then applied as an automated modeling approach. Finally, model performance is evaluated using classification metrics and confusion matrices, while SHAP is used for model interpretation and the McNemar Test is applied to statistically compare the prediction outcomes of the selected models. The overall research workflow is presented in Figure 1.

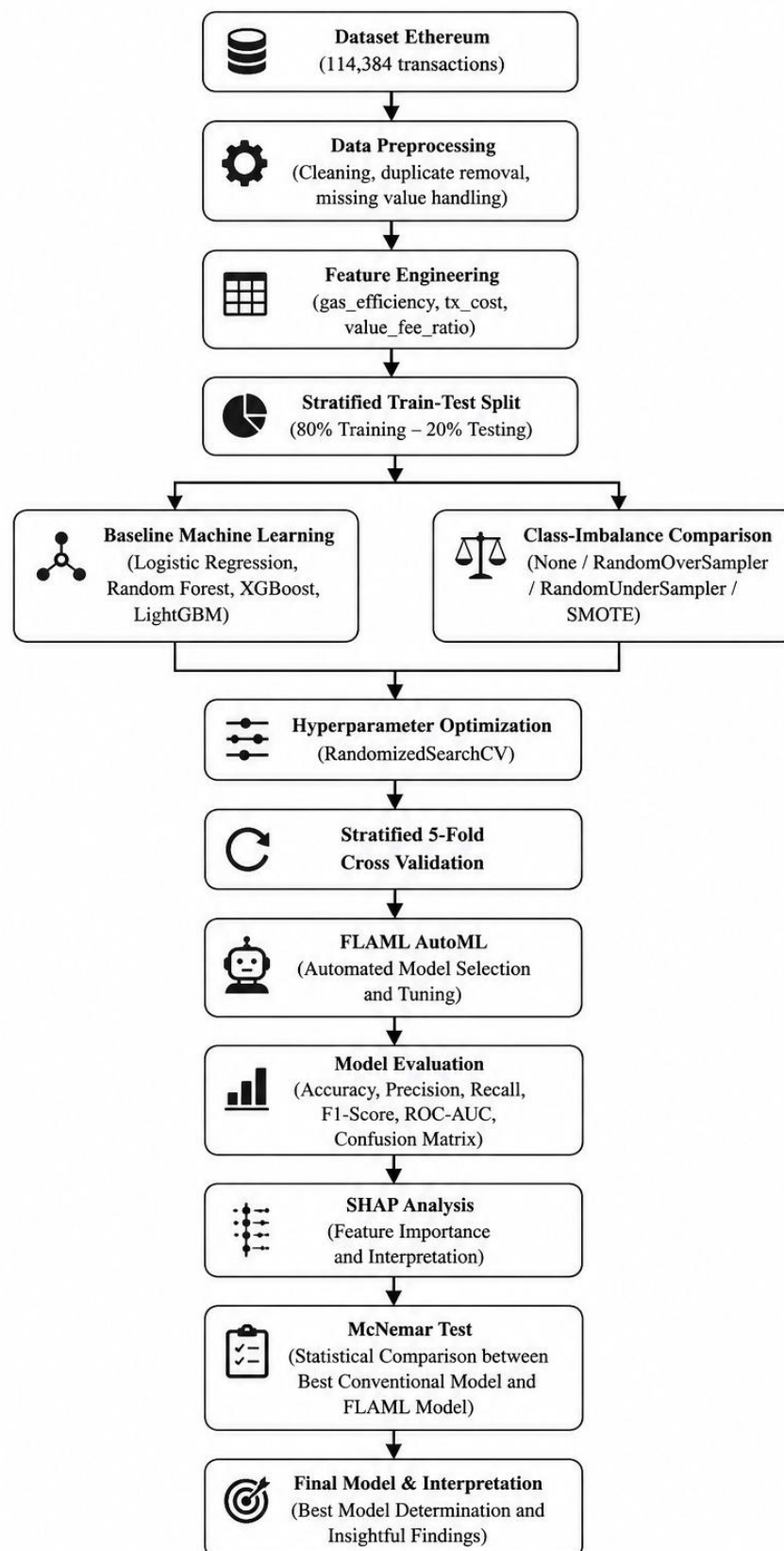


Figure 1. Research Methodology Workflow

2.1 Dataset

The dataset used in this study consists of historical Ethereum transaction records obtained from Google BigQuery Public Datasets. The initial dataset contained 114,384 transaction records. The target variable is status, which represents the transaction outcome, where 1 indicates a successful transaction and 0 indicates a failed transaction. The predictor variables include gasLimit, gasUsed, gasPrice, and value. After the preprocessing stage, 111,617 transaction records were retained for the subsequent modeling process.

The dataset contains a highly imbalanced class distribution, with successful transactions substantially outnumbering failed transactions. Therefore, the experimental design explicitly evaluates several class-imbalance strategies rather than assuming that resampling will necessarily improve classification performance.

2.2 Data Preprocessing and Feature Engineering

Data preprocessing was performed before model development. The preprocessing stage included attribute selection, data-type checking, missing-value inspection, duplicate removal, and dataset validation. After preprocessing, feature engineering was performed to generate additional variables representing transaction characteristics.

Three derived features were constructed. The first feature, gas_efficiency, represents the ratio between the amount of gas used and the allocated gas limit:

$$gas_{efficiency} = \frac{gasUsed}{gasLimit} \quad (1)$$

The second feature, tx_cost, represents the estimated transaction cost based on gas consumption and gas price:

$$tx_{cost} = gasUsed \times gasPrice \quad (2)$$

The third feature, value_fee_ratio, represents the ratio between transaction value and transaction cost:

$$value_{fee_{ratio}} = \frac{value}{tx_{cost}} \quad (3)$$

The resulting dataset therefore consists of seven predictor variables, namely gasLimit, gasUsed, gasPrice, value, gas_efficiency, tx_cost, and value_fee_ratio, with status used as the target variable. The feature-engineering definitions are consistent with the variables specified in the thesis methodology.

2.3 Dataset Splitting

After preprocessing and feature engineering, the dataset was divided into training and testing sets using a Stratified Train-Test Split. Eighty percent of the observations were allocated to the training set and 20% to the testing set. Stratification was applied to preserve the proportion of successful and failed transactions in both subsets. A random state of 42 was used to ensure reproducibility. The training set was used for model development, class-imbalance experiments, and hyperparameter optimization, whereas the testing set was reserved for final performance evaluation. The same data split was maintained across the evaluated algorithms to enable an objective comparison.

2.4 Machine Learning Models

Four conventional classification algorithms were evaluated: Logistic Regression, Random Forest, XGBoost, and LightGBM. Logistic Regression was used as a linear baseline classifier, while Random Forest, XGBoost, and LightGBM were selected to represent tree-based ensemble learning approaches. The baseline experiment was initially conducted using the models' baseline configurations. The results were subsequently used to identify the models and data

configuration for the optimization stage. The same processed dataset and target definition were used for all conventional classifiers to ensure a consistent comparison.

2.5 Class-Imbalance Handling

Because the dataset contains an imbalanced distribution between successful and failed transactions, four experimental conditions were compared: None, using the original training distribution;

RandomOverSampler, which increases the representation of the minority class;

RandomUnderSampler, which reduces observations from the majority class; and

SMOTE (Synthetic Minority Over-sampling Technique), which generates synthetic observations for the minority class [12].

The performance of each strategy was evaluated using the same testing procedure. The strategy producing the strongest overall performance was subsequently used for the hyperparameter optimization stage. In the experiments, the None condition produced the strongest overall performance and was therefore retained for subsequent optimization.

2.6 Hyperparameter Optimization

Hyperparameter optimization was performed using RandomizedSearchCV. RandomizedSearchCV was selected because it explores randomly selected parameter combinations and can be more computationally efficient than exhaustive Grid Search when the search space is large [13]. This method was selected because it explores randomly selected parameter combinations and can be more computationally efficient than exhaustive Grid Search when the search space is large. The optimization process was combined with Stratified Cross Validation to maintain class proportions across validation folds.

The optimization was applied to Logistic Regression, Random Forest, XGBoost, and LightGBM. The best configurations obtained from the experiments were as follows: Random Forest: `n_estimators = 300`, `min_samples_split = 5`, `min_samples_leaf = 1`, `max_features = log2`, and `max_depth = None`; XGBoost: `subsample = 1.0`, `n_estimators = 200`, `max_depth = 10`, `learning_rate = 0.1`, and `colsample_bytree = 1.0`; LightGBM: `num_leaves = 50`, `n_estimators = 300`, `min_child_samples = 20`, `max_depth = 30`, and `learning_rate = 0.05`; Logistic Regression: `solver = lbfgs`, `penalty = l2`, and `C = 0.01`.

2.7 Stratified Cross Validation

Following hyperparameter optimization, model stability was evaluated using five-fold Stratified Cross Validation. Stratification was used to preserve the relative class distribution in each fold, which is particularly relevant for imbalanced classification problems [14]. The mean and standard deviation of the evaluation metrics were calculated across the five folds. This procedure was used to determine whether the observed model performance remained consistent across different training-validation partitions rather than depending on a single data split.

2.8 FLAML AutoML

FLAML AutoML was employed as an automated alternative to the conventional model-selection and hyperparameter-tuning process. FLAML automatically searches for suitable learners and parameter configurations under the specified experimental constraints.

The resulting FLAML model was evaluated on the same held-out testing set used for the conventional Machine Learning models. This allowed the automated approach to be compared directly with the manually optimized classifiers. The thesis identifies FLAML as the fifth predictive approach evaluated in the study.

2.9 Model Evaluation

The classification performance was evaluated using Accuracy, Precision, Recall, F1-Score, and ROC-AUC. In addition, confusion matrices were generated to examine the

distribution of correct and incorrect predictions for the Success and Failed classes. F1-Score was emphasized because the dataset has an imbalanced class distribution and the metric provides a balance between Precision and Recall. ROC-AUC was additionally used to evaluate the ability of the models to discriminate between the two transaction classes across different classification thresholds.

2.10 Explainable Artificial Intelligence

The selected Random Forest model was interpreted using SHAP (SHapley Additive exPlanations) [10], [11]. SHAP was used to quantify the contribution of each predictor to the model's output. Global feature importance was obtained by calculating the mean absolute SHAP value for each feature. The resulting ranking was used to identify the transaction characteristics that contributed most strongly to the model's predictions. The thesis methodology specifically uses SHAP to interpret the selected model and identify influential transaction features.

2.11 McNemar Statistical Test

The McNemar Test was used to statistically compare the prediction outcomes of the best conventional Machine Learning model and the FLAML model [15]. The test was performed using predictions generated on the same testing observations, allowing the comparison to be conducted on paired classification outcomes. The null hypothesis states that there is no statistically significant difference between the prediction outcomes of the two compared models. A significance level of $\alpha = 0.05$ was applied. If the resulting p-value was greater than 0.05, the null hypothesis was not rejected. The thesis methodology specifies that McNemar testing is performed using the predictions generated on the testing set.

3. RESULTS AND DISCUSSION

The baseline experiment was conducted using Logistic Regression, Random Forest, XGBoost, and LightGBM. The models were evaluated using Accuracy, Precision, Recall, F1-

Table 1 Comparison of Algorithm A and Algorithm B

Model	Accuracy	Precision	Recall	F1-Score	ROC-AUC
Logistic Regression	0.972765	0.972765	1.000000	0.986194	0.428126
Random Forest	0.983650	0.984655	0.998757	0.991656	0.939330
XGBoost	0.981141	0.983604	0.997237	0.990373	0.940175
LightGBM	0.981321	0.983519	0.997513	0.990467	0.934006

Random Forest achieved the highest Accuracy, Precision, Recall, and F1-Score, while XGBoost obtained the highest ROC-AUC. The F1-Score of 0.991656 indicates that Random Forest provides the best balance between Precision and Recall among the conventional models. Although Logistic Regression achieved a Recall of 1.000000, its ROC-AUC was only 0.428126. This indicates that the high Recall does not correspond to strong discrimination between successful and failed transactions. In contrast, the tree-based ensemble models obtained ROC-AUC values above 0.93.

The class-imbalance experiment compared the original dataset without balancing with RandomOverSampler, RandomUnderSampler, and SMOTE. The original dataset produced the strongest overall performance and was therefore retained for the subsequent optimization stage. This result indicates that resampling was not necessarily beneficial for the investigated dataset. Consequently, class-imbalance handling should be determined empirically according to the characteristics of the data and the selected classifier

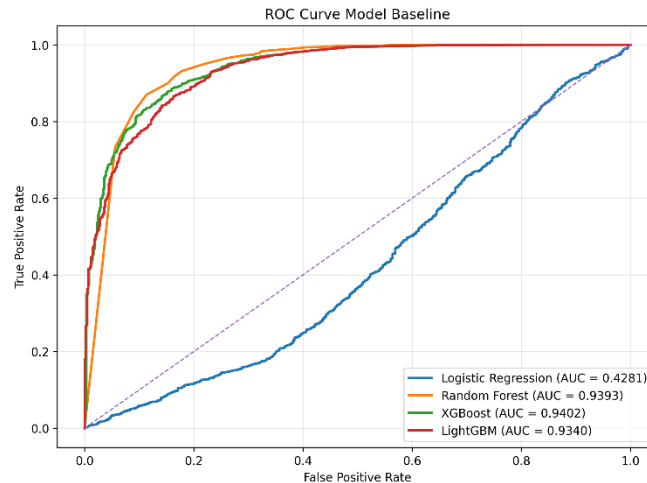


Figure 2. ROC curve of the baseline Machine Learning models

Hyperparameter optimization was subsequently performed using RandomizedSearchCV, followed by five-fold Stratified Cross Validation. Random Forest achieved the highest mean Accuracy, F1-Score, and ROC-AUC among the conventional models, with Accuracy of 0.983571 ± 0.000427 , F1-Score of 0.991616 ± 0.000217 , and ROC-AUC of 0.951198 ± 0.002412 .

Table 2 Stratified Cross Validation results

Model	Accuracy	F1-Score	ROC-AUC
Random Forest	0.983571 ± 0.000427	0.991616 ± 0.000217	0.951198 ± 0.002412
LightGBM	0.983291 ± 0.000534	0.991468 ± 0.000272	0.947658 ± 0.002678
XGBoost	0.983112 ± 0.000453	0.991375 ± 0.000231	0.944394 ± 0.002320
Logistic Regression	0.972753 ± 0.000027	0.986188 ± 0.000014	0.413724 ± 0.032861

The relatively small standard deviations indicate stable performance across the five folds. These results further support Random Forest as the strongest conventional model. FLAML AutoML was then applied as an automated approach for model selection and hyperparameter optimization. FLAML selected XGBoost as its best estimator. The resulting model achieved Accuracy of 0.982799, Precision of 0.985392, Recall of 0.997099, F1-Score of 0.991211, and ROC-AUC of 0.948002.

Table 3. Comparison between Random Forest and FLAML

Metric	Random Forest	FLAML/XGBoost
Accuracy	0.983650	0.982799
Precision	0.984655	0.985392
Recall	0.998757	0.997099
F1-Score	0.991656	0.991211
ROC-AUC	0.939330	0.948002

The results demonstrate that both approaches provide highly competitive performance. Random Forest obtains slightly higher Accuracy, Recall, and F1-Score, whereas FLAML/XGBoost obtains higher Precision and ROC-AUC. The relatively small differences indicate that AutoML can provide a competitive alternative while reducing the need for manual model selection and hyperparameter exploration.

SHAP analysis was subsequently performed on Random Forest to identify the features that contributed most strongly to its predictions. The three most influential features were gasUsed, gas_efficiency, and gasLimit.

Table 4. SHAP feature importance

Rank	Feature	Mean Absolute SHAP
1	gasUsed	0.031123
2	gas_efficiency	0.025406
3	gasLimit	0.024806

4	value_fee_ratio	0.010654
5	tx_cost	0.007999
6	value	0.006736
7	gasPrice	0.005416

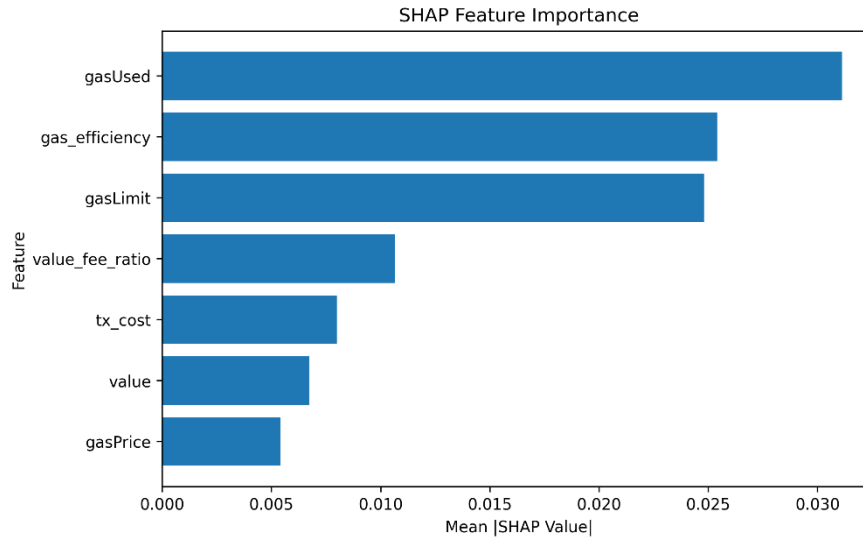


Figure 3. SHAP feature importance

The dominance of gas-related variables indicates that the model relies strongly on transaction execution and gas-related characteristics when distinguishing transaction outcomes. The presence of gas_efficiency among the three most influential features also indicates that the feature engineering process provides useful additional information. However, SHAP values represent model-based feature contributions and should not be interpreted as causal relationships.

Finally, the McNemar Test was conducted to compare Random Forest and the FLAML-selected XGBoost using the same testing observations. The test produced a statistic of 0.778846 and a p-value of 0.377494.

Table 5. McNemar Test result

Comparison	Statistic	p-value	Decision
Random Forest vs. FLAML/XGBoost	0.778846	0.377494	H ₀ not rejected

Since the p-value is greater than 0.05, there is insufficient statistical evidence to conclude that the prediction outcomes of Random Forest and FLAML/XGBoost are significantly different. Therefore, although Random Forest provides slightly higher Accuracy and F1-Score, the difference in paired prediction outcomes is not statistically significant.

Overall, the results demonstrate that Random Forest is the strongest conventional model for the investigated Ethereum transaction dataset, while FLAML provides a highly competitive automated alternative. The experiments also show that the original imbalanced dataset performs better than the evaluated resampling strategies. SHAP analysis identifies gasUsed, gas_efficiency, and gasLimit as the most influential features, while the McNemar Test indicates that the prediction outcomes of Random Forest and FLAML/XGBoost are not significantly different.

4. CONCLUSION

This study developed a Machine Learning framework for classifying Ethereum transaction success using conventional Machine Learning, AutoML, and Explainable Artificial Intelligence. The experimental results show that tree-based ensemble models provide better overall performance than Logistic Regression. Among the conventional models, Random Forest achieved the strongest performance, with an Accuracy of 0.983650 and an F1-Score of

0.991656 in the baseline evaluation. Stratified Cross Validation further confirmed its stable performance across different data partitions.

The class-imbalance experiments showed that the original dataset without resampling provided the strongest overall results compared with RandomOverSampler, RandomUnderSampler, and SMOTE. Hyperparameter optimization and Stratified Cross Validation further demonstrated that Random Forest remained the most competitive conventional model. Meanwhile, FLAML selected XGBoost as its best model and achieved comparable performance, with an Accuracy of 0.982799, F1-Score of 0.991211, and ROC-AUC of 0.948002.

The SHAP analysis identified gasUsed, gas_efficiency, and gasLimit as the three most influential features in the Random Forest predictions. This provides additional interpretability regarding the transaction characteristics used by the model. The McNemar Test produced a p-value of 0.377494, indicating that there was no statistically significant difference between the paired prediction outcomes of Random Forest and the FLAML-selected XGBoost.

Overall, the findings demonstrate that the combination of Machine Learning, AutoML, and explainable analysis can provide a reliable approach for analyzing Ethereum transaction outcomes. Future research can extend this work by using additional blockchain networks, temporal or graph-based transaction features, and features available before transaction execution to develop models suitable for real-time prediction.

ACKNOWLEDGEMENTS

The author would like to express sincere gratitude to Universitas Nusa Mandiri for providing academic support and facilities during the completion of this research. The author also gratefully acknowledges the support and guidance of the supervisors and all parties who contributed to the completion of this study.

The author further acknowledges the use of Google BigQuery Public Datasets as the source of the historical Ethereum transaction data used in this research.

REFERENCES

- [1] V. C. Oliveira et al., "Analyzing Transaction Confirmation in Ethereum Using Machine Learning Techniques," ACM SIGMETRICS Performance Evaluation Review, vol. 48, no. 4, pp. 12–15, May 2021, doi: 10.1145/3466826.3466832.
- [2] H. J. Singh and A. S. Hafid, "Transaction Confirmation Time Prediction in Ethereum Blockchain Using Machine Learning," arXiv, 2019, doi: 10.48550/ARXIV.1911.11592.
- [3] S. Kayikci and T. M. Khoshgoftar, "Blockchain meets machine learning: a survey," Journal of Big Data, vol. 11, no. 1, p. 9, Jan. 2024, doi: 10.1186/s40537-023-00852-y
- [4] C. Jatoth, R. Jain, U. Fiore, and S. Chatharasupalli, "Improved Classification of Blockchain Transactions Using Feature Engineering and Ensemble Learning," Future Internet, vol. 14, no. 1, p. 16, Dec. 2021, doi: 10.3390/fi14010016.
- [5] J. Crisostomo, F. Bacao, and V. Lobo, "Machine learning methods for detecting smart contracts vulnerabilities within Ethereum blockchain—A review," Expert Systems with Applications, vol. 268, p. 126353, Apr. 2025, doi: 10.1016/j.eswa.2024.126353.
- [6] L. Breiman, "Random Forests," Machine Learning, vol. 45, no. 1, pp. 5–32, Oct. 2001, doi:10.1023/A:1010933404324.
- [7] T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System," in Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, Aug. 2016, pp. 785–794, doi: 10.1145/2939672.2939785.
- [8] G. Ke et al., "LightGBM: A Highly Efficient Gradient Boosting Decision Tree," in Advances in Neural Information Processing Systems, vol. 30, 2017.

- [9] C. Wang, Q. Wu, M. Weimer, and E. Zhu, “FLAML: A Fast and Lightweight AutoML Library,” in Proc. Fourth Conference on Machine Learning and Systems (MLSys), 2021.
- [10] S. Lundberg and S.-I. Lee, “A Unified Approach to Interpreting Model Predictions,” arXiv, 2017, doi: 10.48550/ARXIV.1705.07874.
- [11] S. M. Lundberg et al., “From local explanations to global understanding with explainable AI for trees,” Nature Machine Intelligence, vol. 2, no. 1, pp. 56–67, Jan. 2020, doi: 10.1038/s42256-019-0138-9.
- [12] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, “SMOTE: Synthetic Minority Over-sampling Technique,” Journal of Artificial Intelligence Research, vol. 16, pp. 321–357, Jun. 2002, doi: 10.1613/jair.953.
- [13] J. Bergstra and Y. Bengio, “Random Search for Hyper-Parameter Optimization,” Journal of Machine Learning Research, vol. 13, pp. 281–305, 2012.
- [14] S. Szeghalmy and A. Fazekas, “A Comparative Study of the Use of Stratified Cross-Validation and Distribution-Balanced Stratified Cross-Validation in Imbalanced Learning,” Sensors, vol. 23, no. 4, p. 2333, Feb. 2023, doi: 10.3390/s23042333.
- [15] Q. McNemar, “Note on the Sampling Error of the Difference Between Correlated Proportions or Percentages,” Psychometrika, vol. 12, no. 2, pp. 153–157, Jun. 1947, doi: 10.1007/BF02295996