Journal of Global Research in Electronics and Communication
Volume 1, No. 7, July 2025
Available Online at: www.jgrec.info
RESEARCH PAPER

# Transaction Analysis of Categorizing Ethereum Addresses Based on Advanced Supervised Machine Learning Approach for Predictive Modeling

Raja Lodhi
Mtech Scholar
Department of Computer Science and Engineering,
LNCT Group of Colleges
Bhopal, M.P., India
ndrajalodi@gmail.com

Raj Kumar Sharma
Assistant Professor
Department of Computer Science and Engineering,
LNCT Group of Colleges
Bhopal, M.P., India
Rajkumar.s@lnct.ac.in

***Abstract*—Ethereum has become one of the most significant cryptocurrencies in terms of transaction volume. Given Ethereum's recent rise, experts and the cryptocurrency community are eager to learn more about how Ethereum transactions behave. A machine learning system-based methodology exists for addressing Ethereum addresses to enable transaction classification. The preprocessing of CEAT dataset containing 4,371 entries with 15 features utilizes a systematic process that selects relevant features then handles missing data along with using SMOTE for class distribution balancing and converting categorical elements to numbers. The data is standardized with Min-Max Scaler to improve model performance. Exploratory data analysis is done by visualizations such as Heat map, Histogram and Pair plot to compute the features which are correlated. The four models of machine learning algorithms include Decision Tree (DT), LightGBM, Neighbors Classifier (KNN), and CatBoost Classifier, are trained with the best optimized hyperparameters. The classification reports, confusion matrices, and ROC curves are used for model evaluation for each model. Comparing these models, LightGBM has the highest accuracy of 91.99%, second is CatBoost Classifier with 91.23%, the Decision Tree is 85.16%, and the KNN model 78.91%. An important benefit of this study is that the results show that it is possible to use a machine-learning approach for classifying Ethereum addresses to enhance transaction security and avoid fraud in decentralized financial systems.**

***Keywords*—Ethereum addresses, Machine Learning, Decision tree, K-neighbors, CatBoost and LightGBM,**

## I. INTRODUCTION

Ethereum stands as a well-known cryptocurrency. According to the quantity of recorded financial transactions, it has grown to be among the biggest cryptocurrencies at the moment. Since its launch in February 2020, the network has processed over 470M transactions, or 9 per second, and the resulting market capitalization is over $27 billion USD. Furthermore, Ethereum's primary benefit over Bitcoin, the original cryptocurrency, has been thought to be its ability to handle programmed contracts, or smart contracts, which it adds to its financial transactions [1][2][3]. In a cryptocurrency system, a distributed consensus process can determine if a transaction was successful or unsuccessful after it was executed [4][5]. Here, transaction confirmation is crucial as failing to do so might result in users losing their money (i.e., the charge paid to network operators or miners to execute a transaction is not reimbursed) [6][7].

Additionally, there are no failure risks disclosed when a transaction enters the network to be processed. As a result, failures might negatively impact users' experience and discourage them from making future transactions [8][9][10]. In contrast, the distributed consensus process required to execute and verify each Ethereum transaction is complicated, and the tiny proportion of failed transactions, when taking into account millions of recorded transactions, makes it difficult to build models to forecast confirmation [11][12].

Ethereum addresses receive primary classification through transaction analysis methods that provide understanding about user behaviors and their transaction patterns [13][14]. Researchers utilize machine learning methods to parse meaningful data patterns from the Ethereum address transaction records for classification between individual users, exchanges, miners and malicious actors [15][16][17]. Finding any transactions that were thought to have unusual characteristics would require a laborious and time-consuming manual search of all of these transactions. An use of ML methods to aid in an identification of patterns associated with suspicious behavior is recommended by the fast development of such network blocks, especially smart contracts and transactions [18]. Building prediction models under supervised learning branches of ML requires training on extremely large datasets containing labeled samples with their actual outputs noted [19]. Therefore, its usage of labeled datasets for model training is the primary differentiator from other ML types [20][21].

### *A. Motivation and Contribution*

A motivation for this research stems from the growing complexity and security challenges in Ethereum transactions, where traditional fraud detection methods fall short against evolving threats. With the increasing adoption of blockchain technology, there is a pressing need for automated, scalable, and high-accuracy classification models to distinguish between legitimate users and malicious actors. By leveraging advanced supervised machine learning, this study aims to enhance blockchain security, improve fraud detection, and contribute to more robust financial forensics in decentralized ecosystems. Here are key research contributions from this study on categorizing Ethereum addresses using advanced supervised machine learning for predictive modeling:

- This research improves our capacity to distinguish between real and fraudulent transactions by introducing a strong supervised learning framework

for Ethereum address classification based on transactional patterns.
- A ML evaluation of Decision Tree, LightGBM, K Neighbors Classifier, and CatBoost Classifier is to identify the most effective model.
- SMOTE successfully resolves class imbalance, leading to more equitable training and improved generalizability across various Ethereum address categories.
- The study demonstrates the impact of Min-Max Scaler for feature normalization, which enhances model performance and stability by reducing variance in Ethereum transaction data.
- The study systematically evaluates model performance using ROC curves, confusion matrices, and classification reports, offering a structured approach to selecting the best-performing classifier.

### *B. Novelty and Justification*

The novelty of this research lies in its systematic application of advanced supervised machine learning techniques to accurately categorize Ethereum addresses, addressing the growing complexity of blockchain transactions. Unlike prior studies that rely on single models or basic heuristics, this work provides a comparative evaluation of multiple high-performing algorithms—Decision Tree, LightGBM, Neighbors, and CatBoost highlighting their strengths and limitations in distinguishing diverse address types. This approach is justified by the pressing need for scalable, automated, and precise classification methods to improve fraud detection, enhance security, and support regulatory compliance in decentralized financial ecosystems.

### *C. Structure of the Paper*

Here is the breakdown of the study: A review of the current literature on Ethereum categorization is presented in Section II. Methods used to gather information for this research are detailed in Section III. Classification of text findings and analysis are presented in Section IV. Finally, the conclusion is provided in Section V.

## II. LITERATURE REVIEW

This section discusses the literature review on transaction analysis of categorizing Ethereum addresses based on advanced machine-learning approach for predictive modeling. Table I also includes the abstracts of the following research reviews:

Bani-Hani, Shatnawi and Al-Yahya (2024) using deep learning methods, Ethereum transaction vulnerabilities may be categorized and detected. These transactions are transformed into RGB and greyscale pictures, which are subsequently analyzed by the binary and multi-label classification algorithms ResNet50, DenseNet201, VGG19, KNN, and RF. The best method for binary classification was RF, which had an accuracy rate of 86.66% and a score of 86.66% [22].

Yan and Kompalli (2023) determine whether a certain set of transactions follows the same course of execution on the current blockchain state. Their analysis of more than 1.3 billion Ethereum transactions successfully uncovers suspicious behaviors linked to an accuracy of 83.8 percent [23].

Saleem et al. (2023) have used the publicly accessible Ethereum blockchain's tagged dataset consisting of 300 million transactions. Using eleven feature vectors and 200 window widths, the XGBoost classifier achieved the greatest attainable accuracy of 73% in predicting the function of an unknown address, according to the test data. An impressive 86% accuracy rate was reached by the CNN model when it came to predicting labels using the dataset [24].

Aziz et al. (2023) proposed methods that were evaluated in relation to the performance and efficiency measures of other well-known methods for identifying fraudulent activity on Ethereum. These methods included KNN, LR, MLP, XGBoost, LGBM, RF, and SVC. With maximum accuracy, the recommended approach and SVC models surpass all other models. When used in conjunction with the suggested optimization approach, DL achieves a performance of 91%, which is somewhat better than the RF model [25].

Pragasam et al. (2023) illustrate that the RF, GB, and XGBoost classifiers for address category prediction were trained and evaluated using a dataset including 4371 samples. The XGBoost classifier outperformed all other models in this problem set, with a macro-averaged F1Score of 0.689 and an accuracy of 75.3%. The Random Forest classifier came in second, with a macro-averaged F1Score of 0.641 and an accuracy of 73.7%. With gradient boosting, the accuracy rate was 73% [26].

Dritsas and Trigka (2023) conducted experiments with various supervised ML models to identify early-stage symptoms of SARS-CoV-2 infection. The results demonstrated that the Stacking ensemble model achieved the best results, with accuracy, precision, recall, and F-measure of 90.9% [27].

Recent studies on Ethereum address classification use ML and DL models like Random Forest, XGBoost, and CNN, achieving moderate to high accuracy but struggling with imbalanced data, closely related address types, and complex transactional patterns. Most focus on single models without systematic comparisons. The proposed work addresses these gaps by integrating multiple supervised models, balancing data, and standardizing features. This enables more accurate, scalable, and reliable categorization of Ethereum addresses, particularly for challenging or closely related classes, enhancing predictive modeling for blockchain transaction monitoring.

TABLE I. SUMMARY OF LITERATURE REVIEW TRANSACTION ANALYSIS OF ETHEREUM BASED ON MACHINE LEARNING APPROACHES

| Author | Dataset | Methods | Key Findings | Accuracy | Limitation / Gap |
|---|---|---|---|---|---|
| Bani-Hani, Shatnawi & Al-Yahya (2024) | Ethereum transactions transformed into RGB and greyscale images | ResNet50, DenseNet201, VGG19, KNN, RF | RF was best for binary classification, achieving highest accuracy and score | 86.66% | Limited to image-based representation; scope for feature-based analysis |
| Yan & Kompalli (2023) | >1.3 billion Ethereum transactions | Unsupervised analysis of execution flow patterns | Detects suspicious behavior with high accuracy by analyzing transaction execution flow | 83.8% | Focused on execution sequence patterns, lacks diverse feature usage |

| | | | | | |
|---|---|---|---|---|---|
| Saleem et al. (2023) | Public Ethereum dataset (300 million transactions) | XGBoost, CNN | XGBoost achieved 73% accuracy; CNN achieved 86% accuracy for address function prediction | 86% (CNN) | Limited feature engineering detail; black-box nature of CNN |
| Aziz et al. (2023) | Not specified | KNN, LR, MLP, XGBoost, LGBM, RF, SVC, Deep Learning (DL) | DL achieved highest performance (91%) in combination with optimization techniques, surpassing RF and others | 91% | Dataset details not provided; lacks explainability of DL models |
| Pragasam et al. (2023) | 4371 samples | RF, GB, XGBoost | XGBoost outperformed others: Macro F1-score 0.689, accuracy 75.3%; RF followed with 73.7% accuracy | 75.3% | Small dataset size; limited scalability to real-world datasets |
| Dritsas & Trigka (2023) | Medical dataset (SARS-CoV-2) | Various supervised ML models, Stacking ensemble | Stacking ensemble achieved best results: Accuracy 90.9%, high precision, recall, and F-measure | 90.9% | Related to medical domain, not blockchain-specific |

## III. Methodology

Figure 1 demonstrates the process flow of the proposed technique for classifying Ethereum addresses using supervised ML. It starts with data preparation and continues with visualization, model training, and performance assessment. The CEAT dataset, containing 4,371 entries and 15 features, undergoes cleaning by removing unnecessary columns, handling missing values, encoding categorical variables, and balancing imbalanced data using SMOTE. Min-Max Scaler standardizes the features to improve model performance. Exploratory Data Analysis (EDA) includes heatmaps, histograms, and pair plots to understand feature correlations. The dataset is split into an 80-20 train-test ratio, and four ML models—DT, LightGBM, Neighbors Classifier, and CatBoost Classifier are implemented with optimized hyperparameters. Each model is evaluated based on classification reports, confusion matrices, and ROC curves to assess predictive accuracy. Finally, model comparison is performed to determine the best-performing algorithm for Ethereum address classification.

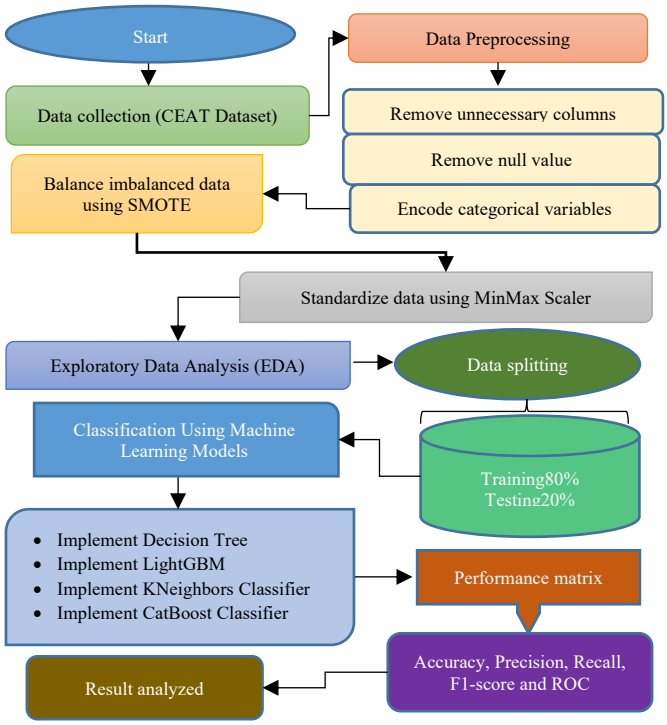


Fig. 1. Proposed Flowchart Ethereum Transaction classification using machine learning techniques

A whole process of proposed methodology shows in Figure 1. also, each and every step is discussed below:

### A. Data Collection

The "CEAT" project repository, created on June 30, 2023, appears to focus on data processing phase, contain 4371 rows of data and analysis. The structure suggests a workflow for preparing data, extracting features, and optimizing models, likely for machine learning or statistical analysis purposes,

### B. Data Preprocessing

The primary first stage in data analysis is data preparation. It enables to convert unstructured data into a form that can be analyzed more effectively [28]. Data Preprocessing for Ethereum Classification on CEAT data:

- **Data Cleaning:** The dataset, consisting of 4,371 entries and 15 features, undergoes a cleaning process to ensure the removal of unnecessary or irrelevant columns [29]. The necessary features are identified for retention during this step to optimize the dataset for model training purposes.
- **Handling Missing Values:** Methods are put in place to deal with datasets that include missing values. It can involve operations like imputing the missing observations or altogether dropping the rows with missing data if the dataset needed to be cleansed before the machine learning techniques were applied on them.
- **Encoding Categorical Variables:** This transformative process refers to the conversion of categorical variables available within the data set into other types of data using the process of encoding. This step helps to avoid the ambiguity of categorical data, for example, regarding the type of transaction or the Ethereum address for the model input.

### C. Data Normalization Using Min-Max Scaler

The dataset receives standardization treatment through normalization procedures. An essential part of data preparation, data scaling [30] seeks to standardize and make comparable all numerical properties [31]. A popular approach for this is Min-Max Scaler. For the purpose of normalization, the Min-Max Scaler method was used. Equation (1) shows the formula that was used to normalize the data.

$$x' = \frac{x - x_{min}}{x_{max} - x_{min}} \quad (1)$$

Where x stands for the pre-normalization value, and x' refers to the value that remains after normalization; one may

find the maximum and lowest values of the sample data by looking at the variables x max and x min, respectively.

*D. Balancing an Imbalanced Dataset*

The easiest way to increase the size of the minority class, while it may lead to overfitting, is to randomly increase the sample size. By inserting duplicate instances into the training set using KNN, SMOTE decreases the likelihood of overfitting [32]. Specifically, SMOTE makes use of Equation (2).

$$x_{syn} = x_i + (k_{nn} - x_i)xt \quad (2)$$

Where t is a random integer between 0 and 1, and xi is a feature vector, which is known for the KNN.

*E. Data Splitting*

In order to assess the effectiveness of a model's dataset, data splitting is a crucial stage in machine learning models. Specifically, 80% of the data is used for training, while 20% is used for testing.

*F. Classification of ML Models with Hyperparameter Tuning*

In this section provide the classification machine learning models (Decision Tree (DT), Neighbors Classifier, LightGBM (LGBM and CatBoost) for the Ethereum Transaction classification on CEAT data

*1) K-Nearest Neighbor (KNN) Classifier*

Regression and classification are two applications of the ML technique KNN. Using the labels or values of the KNN linked to the new data point in the training set, one may predict the new data point's label or value [33]. Simply said, KNN saves the complete training dataset, therefore training time is unnecessary. The "k" in k-nearest neighbors indicates that the method may consider any specified number of neighbors in the training set, rather than only the neighbor to the close-by data point. kNN is one of the algorithms that make up the instance-based learning family. A data is pushed to a neighboring class with a most immediate proximity. As an expansion, a quantity of closest neighbors, the value of k (with a common choice being n_neighbors=2), precision may increment [34]. KNN Euclidean, Manhattan and Minkowski calculate as formula (3,4 and 5):

KNN Euclidean distance Equation (3):

$$\sqrt{\sum_{i=0}^{k}(x_i - y_i)^2} \quad (3)$$

KNN Manhattan distance Equation (4):

$$\sum_{i=1}^{k}|x_i - y_i| \quad (4)$$

KNN Minkowski distance Equation (5):

$$\left(\sum_{i=1}^{k}(|x_i - y_i|)^q\right)^{\frac{1}{q}} \quad (5)$$

Where q is a real number between zero and one.

*2) Catboost Classifier*

The CatBoost classifier is a gradient-boosting technique that uses binary decision trees to generate predictions and is particularly good at handling category information. Both numerical and binary answer variables may be accommodated. Parameters that were set for the CatBoost Classifier model to achieve a compromise between training duration and model complexity were a depth of 6, a learning rate of 0.1, and 1000 iterations. With the 'Multiclass' loss function and the 'Accuracy' evaluation metric, the model is trained to perform well on tasks involving classifications between several classes. The regularization parameter, l2_leaf_reg, is set to 3 to reduce overfitting by penalizing large leaf values. Lastly, to make sure the findings are reproducible between runs, the random state is set to 42. These settings were chosen to optimize model training, balancing performance and generalization.

*3) Light Gradient Boosting Method (LGBM)*

A fast and effective tree-based gradient enhancement method, Light Gradient Boosting (or "Light GBM") [35]. The classifier uses a tree-based approach with vertical tree development, which is where the term "light" appears. It outperforms techniques based on horizontal trees in terms of efficiency [36]. Large dataset processing benefits from the time and resource efficiency of the Light gradient boosting technique. Light GBM is different from other techniques in that it grows trees leaf-wise, or vertically, as opposed to horizontally, such as most other methods do. The leaf with the greatest delta loss will be chosen for agricultural use. Compared to a level-based technique, a leaf-wise approach to growing the same leaf may reduce waste more effectively [37]. The LightGBM model was configured with the following parameters: num_leaves set to 50, which aids in managing model complexity and overfitting by controlling the amount of leaves in each tree. A learning rate of 0.3 is used to balance the speed of learning and the potential for overfitting, ensuring faster convergence. The max_depth parameter is set to -1, allowing the model to grow trees without a predefined depth limit, promoting better fitting to the data. With 1000 estimators, the model utilizes a sufficient number of trees to improve predictive accuracy. These adjustments are made with the main objective of maintaining high training efficiency as well as enhancing the model's ability for complex input patterns.

*4) Decision Tree*

Data can be divided into sub-sets based on feature values with a DT which is a type of supervised ML model implemented by a tree that utilizes nodes to account for features as well as leaves for the outcomes. It is used in classification and regression techniques for the purpose of creating the best split which maximizes information gain or minimizes variance. DT are easy to interpret and can accommodate both numerical and categorical data and its main disadvantage is overfitting when the trees are deep; this can be resolved by pruning or even aggregation techniques [38]. The Decision Tree model was generated with the following settings of the Decision Tree Algorithm: criterion=log_loss, when making the decision on the best split. CNN splitting with splitter = 'best' guarantees that the chosen split at each node is the best possible with an outlook towards perfect decision boundaries. The min_samples_split=2 enables the splitting of a node as early as there are two samples which helps to capture more detailed patterns in the data. Finally, the max_depth=10 constrains the decision trees up to the depth of 10 in order to avoid overfitting but at the same model to have sufficient level of complexity. These settings were chosen because they should provide high accuracy on the problem while being as general as possible.

*G. Performance Measures*

To evaluate each model's effectiveness, four different performance criteria have been used: F1-score, precision,

 

recall, and accuracy [39]. A confusion matrix is one of the most well-known academic performance measures used to analyse the outcomes. The matrix displays the outcome data using four main qualities; this data is the total of the outcomes from classifications. A result is considered true positive (TP) if the actual value of the classification equals the expected value. Similar in nature, true negative (TN) principles are centered on zero. A false negative (FN) happens when the opposite is true, while a false positive (FP) happens when the expected value is 1 but the actual value is 0.

*1) Accuracy (Acc)*

Accuracy is the ratio of correctly classified cases to the total error in class prediction. The accuracy Equation (6) is:

$$Acuracy = \frac{TP+TN}{TP+TN+FN+FP} \quad (6)$$

*2) Precision (Prec)*

Precision refers to the degree of accuracy in assigning instances to the correct class. Precision is formulated in Equation (7):

$$Precision = \frac{TP}{TP+FP} \quad (7)$$

*3) Recall (Rec)*

This context calculates the percentage of all bearers of the ailment that the classifier correctly represents. Recall is Equation (8).

$$Recall = \frac{TP}{TP+FN} \quad (8)$$

*4) F1-score*

When rec and prec are weighted harmonically, the result is the F1-score, also called the F-measure Equation (9):

$$F1 - score = 2 * \frac{precision*recall}{precision+recall} \quad (9)$$

*5) ROC Curve*

The ROC curve and AUC score are shown to evaluate the model's class-differentiation capabilities.

The accuracy and efficacy of the model in forecasting the CEAT variable are shown by these indicators taken together.

## IV. Result Analysis and Discussion

The experimental results for Ethereum Transaction classification using ML techniques on the CEAT dataset model are shown in this part. Performance metrics, including Rec, Acc, Prec, and F1-score, are examined, along with the classification report and ROC confusion matrix. To meet the computational requirements of the suggested models, a hardware platform with an NVIDIA GTX 1660i GPU with 8 GB of VRAM and 16 GB of RAM was installed. This platform included the Python programming language, Jupyter Notebook, Google Colab, and Python Sk-learn, NumPy, seaborn, Pandas, and matplotlib, among other libraries and toolboxes. The results for classification of Ethereum transactions using approaches mentioned above are demonstrated further in this sections with the help of CEAT data visualization and analysis.

### A. Data Analysis and Visualization

The CEAT dataset also contains number balance, transaction frequency and volume of transaction which is transformed into a balanced dataset for testing the address categorization systems. This dataset is categorized into three distinct classes of Ethereum addresses, enabling the development and evaluation of supervised ML models for predictive analytics. The goal is to enhance the detection and classification of address behaviors, ensuring robust and accurate categorization in the Ethereum blockchain ecosystem dataset.

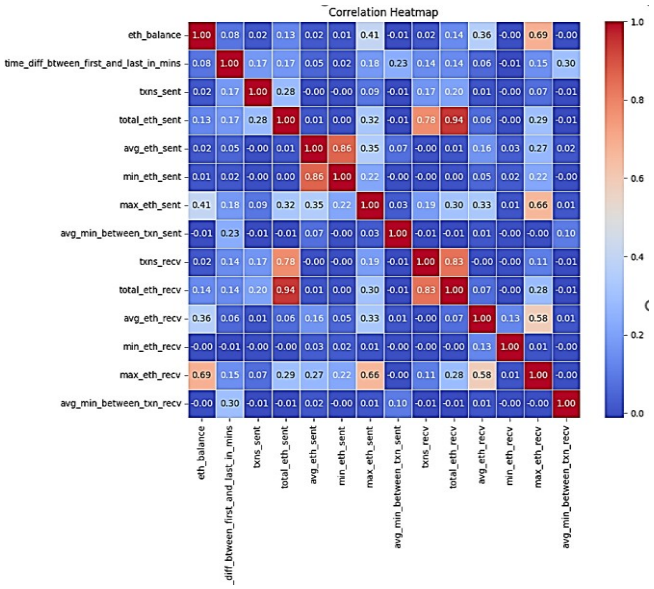


Fig. 2. Heatmap of CEAT dataset

The following Figure 2, the heatmap of the CEAT dataset image, depicts a correlation matrix, visualizing the pairwise correlations between multiple variables. The color intensity ranges from dark red (strong positive correlation) to dark shades (neutral or negative correlations). Closer numbers near 1 show a very positive connection, whereas those closer to -1 show a highly negative association.

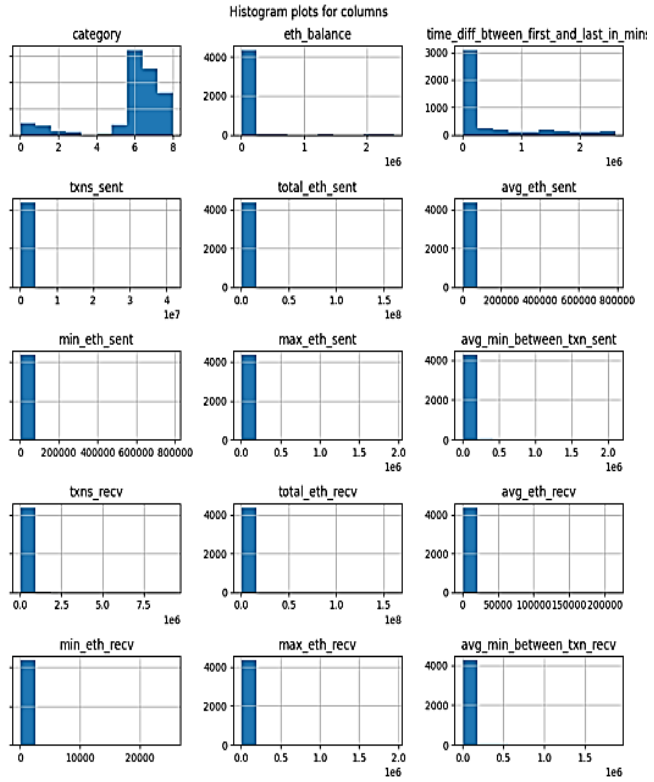


Fig. 3. Distribution in histogram CLEAT dataset

The following Figure 3 shows the distribution series of histogram plots representing the distribution of various

features in a dataset. The features include transactional and balance-related metrics for Ethereum accounts. The first plot shows the "category" feature, which seems to be categorical with values distributed across a range. Features associated with Ethereum transactions, such as eth_balance, time_diff_between_first_and_last_in_min, txns_sent, total_eth_sent, average_eth_sent, and others, have very skewed distributions, with the majority of values centred around zero and a small number of severe outliers. This pattern is consistent across features for both sent and received transactions, including the number, total, average, minimum, and maximum values, as well as the average time between transactions.

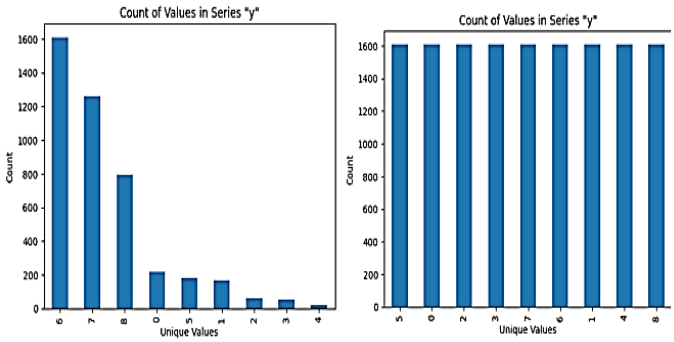


Fig. 4. Bar graphs count of values in series y after and before SMOTE

Figure 4 shows a bar graph comparing the count of values in series "y" before and after applying SMOTE. Before SMOTE, a significant class imbalance exists, with one dominant value (likely "5"). After SMOTE, the minority classes are augmented, and the distribution becomes more balanced, confirming SMOTE's effectiveness in addressing class imbalance for improved model performance.

### *B. Experiment results*

This section presents the experiment results of the used ML models in terms of performance metrics for Ethereum Transaction categorization in table and graph style.

TABLE II. PROPOSED MODELS PERFORMANCE ON CEAT DATASET FOR ETHEREUM TRANSACTION CLASSIFICATION

| Measure | DT | LightGBM | KNN | CatBoost |
|---|---|---|---|---|
| Accuracy | 85.16 | 91.99 | 78.91 | 91.23 |
| Precision | 84.85 | 91.89 | 78.49 | 91.11 |
| Recall | 85.16 | 91.99 | 78.91 | 91.23 |
| F1-score | 84.92 | 91.93 | 78.06 | 91.13 |

Table II presents the performance of four ML models—DT, LightGBM, KNN, and CatBoost on the CEAT dataset for Ethereum Transaction classification, as measured by Recall, Accuracy, Precision, and F1-score. LightGBM and CatBoost demonstrate superior performance, achieving accuracy scores above 91%, while DT achieves a moderate accuracy of 85.16%. KNN exhibits the lowest performance among the four, with an Acc of 78.91%. F1score, which balances Prec and Rec, follows a similar trend, with LightGBM and CatBoost leading at around 91.93% and 91.13%, respectively, and KNN lagging at 78.06%. This indicates that ensemble methods like LightGBM and CatBoost are more effective for this classification task compared to traditional methods like DT and KNN.

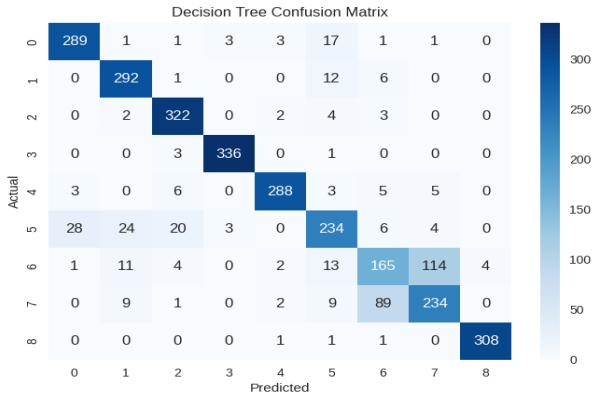


Fig. 5. Confusion matrix of Decision Tree

The Decision Tree performs well for several Ethereum address classes shown in Figure 5, achieving perfect classification for class 8 (308/308) and near-perfect for class 3 (336/336). Classes 1, 2, and 4 also show strong accuracy with 292, 322, and 288 correct predictions, respectively. However, the model struggles with classes 5, 6, and 7, showing notable misclassifications—class 5 is confused with classes 0–2, class 6 with classes 5 and 7, and class 7 with class 6. These results suggest effective classification for distinct patterns but difficulty distinguishing closely related address types, indicating potential benefits from enhanced feature engineering or ensemble approaches.

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.90 | 0.91 | 0.91 | 316 |
| 1 | 0.86 | 0.94 | 0.90 | 311 |
| 2 | 0.90 | 0.97 | 0.93 | 333 |
| 3 | 0.98 | 0.99 | 0.99 | 340 |
| 4 | 0.97 | 0.93 | 0.95 | 310 |
| 5 | 0.80 | 0.73 | 0.76 | 319 |
| 6 | 0.60 | 0.53 | 0.56 | 314 |
| 7 | 0.65 | 0.68 | 0.67 | 344 |
| 8 | 0.99 | 0.99 | 0.99 | 311 |
| accuracy | | | 0.85 | 2898 |
| macro avg | 0.85 | 0.85 | 0.85 | 2898 |
| weighted avg | 0.85 | 0.85 | 0.85 | 2898 |

Fig. 6. Classification report of Decision tree

The DT achieves an overall Acc of 85% in classifying nine Ethereum address categories, shows in Figure 6. It performs exceptionally for classes 3 and 8 (Prec, Rec, and f1score 0.98–0.99) and well for classes 0, 1, 2, and 4 (f1-score 0.90–0.95). However, classes 5, 6, and 7 show weaker performance, with f1-scores of 0.76, 0.56, and 0.67, respectively, indicating difficulty distinguishing these closely related address types. Macro and weighted averages match the overall accuracy, and support values suggest that performance gaps arise from feature similarity rather than class imbalance.

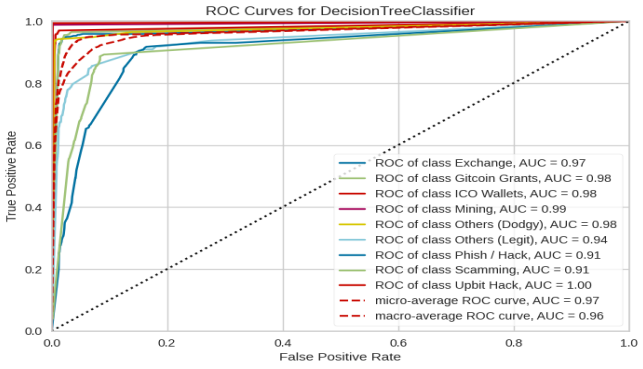


Fig. 7. Roc curve of Decision Tree

Figure 7 displays the Decision Tree ROC curves, which provide solid discriminative performance with AUC values ranging from 0.91 to 1.00. The model achieves perfect separation for the "Upbit Hack" class (AUC = 1.00) and near-perfect for "Mining" (AUC = 0.99), while classes like "Exchange," "Gitcoin Grants," "ICO Wallets," and "Others (boggy)" score 0.96–0.98. Even weaker-performing classes maintain good AUCs (0.91–0.94). Micro- and macro-average AUCs of 0.97 and 0.96 confirm high overall predictive ability, validating the Decision Tree's effectiveness in distinguishing various Ethereum address categories despite some misclassifications.

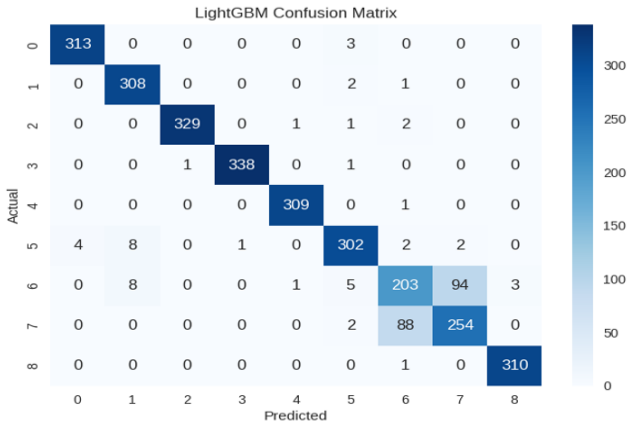


Fig. 8. Confusion matrix of LightGBM model

The LightGBM model demonstrates excellent classification across 9 Ethereum transaction classes, shown in Figure 8, with strong diagonal performance indicating accurate predictions. Most classes show minimal confusion, except for classes 6 and 7, which have mutual misclassifications (94 and 88 instances respectively). Overall, the model outperforms the previously analyzed Decision Tree model with clearer classification boundaries.

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.99 | 0.99 | 0.99 | 316 |
| 1 | 0.95 | 0.99 | 0.97 | 311 |
| 2 | 1.00 | 0.99 | 0.99 | 333 |
| 3 | 1.00 | 0.99 | 1.00 | 340 |
| 4 | 0.99 | 1.00 | 1.00 | 310 |
| 5 | 0.96 | 0.95 | 0.95 | 319 |
| 6 | 0.68 | 0.65 | 0.66 | 314 |
| 7 | 0.73 | 0.74 | 0.73 | 344 |
| 8 | 0.99 | 1.00 | 0.99 | 311 |
| accuracy | | | 0.92 | 2898 |
| macro avg | 0.92 | 0.92 | 0.92 | 2898 |
| weighted avg | 0.92 | 0.92 | 0.92 | 2898 |

Fig. 9. Classification report of LightGBM model

The LightGBM classification report shows exceptional performance across most classes, shown in Figure 9. Classes 0-5 and 8 achieve outstanding metrics with Prec, Rec, and F1scores ranging from 0.95 to 1.00. Class 3 achieves perfect precision and near-perfect recall, while Class 4 shows perfect recall. However, Classes 6 and 7 show relatively lower performance, with F1scores of 0.66 and 0.73 respectively, indicating some classification challenges. The model performs robustly over the 2,898 total data, achieving a good overall accuracy of 0.92 with constant weighted and macro averages.

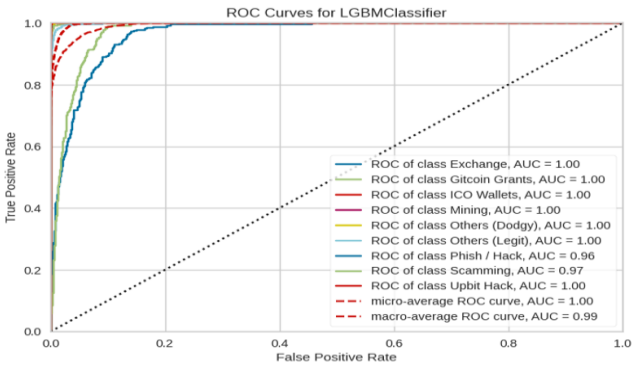


Fig. 10. Roc curve for LightGBM model

An ROC curves for a LightGBM model demonstrate exceptional classification performance across all transaction types, as shown in Figure 10. Most classes, including Exchange, Gitcoin Grants, ICO Wallets, Mining, Others (Dodgy), Others (Legal), and Upbit Hack, achieve perfect AUC scores of 1.00. The Phishing/Hack class shows strong performance with an AUC of 0.96, while Scamming transactions achieve an AUC of 0.97. Results from the Decision Tree model are greatly outperformed by the model, as shown by the micro-average ROC curve (AUC=1.00) and macro-average ROC curve (AUC=0.99).

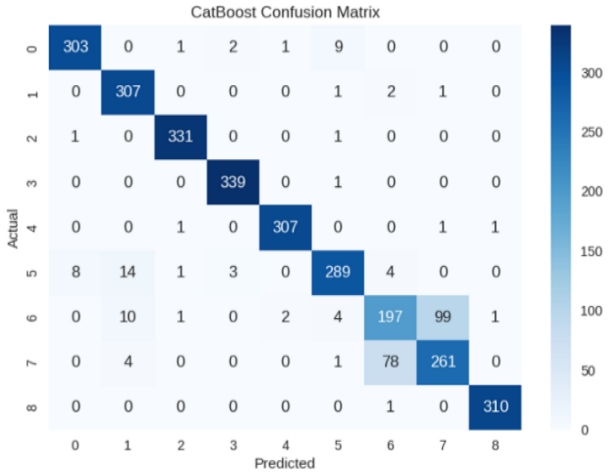


Fig. 11. Confusion matrix of Catboost model

Figure 11 illustrates that the CatBoost model performs well throughout 9 classes, with particularly good accuracy along the diagonal. Classes 0-4 and 8 show excellent prediction accuracy. Notable confusion exists between classes 6 and 7, with 99 and 78 misclassifications, respectively. Class 5 shows some confusion with classes 0 and 1, but maintains good overall performance.

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.97 | 0.96 | 0.96 | 316 |
| 1 | 0.92 | 0.99 | 0.95 | 311 |
| 2 | 0.99 | 0.99 | 0.99 | 333 |
| 3 | 0.99 | 1.00 | 0.99 | 340 |
| 4 | 0.99 | 0.99 | 0.99 | 310 |
| 5 | 0.94 | 0.91 | 0.92 | 319 |
| 6 | 0.70 | 0.63 | 0.66 | 314 |
| 7 | 0.72 | 0.76 | 0.74 | 344 |
| 8 | 0.99 | 1.00 | 1.00 | 311 |
| accuracy | | | 0.91 | 2898 |
| macro avg | 0.91 | 0.91 | 0.91 | 2898 |
| weighted avg | 0.91 | 0.91 | 0.91 | 2898 |

Fig. 12. Classification report for catBoost model

The CatBoost model shows excellent metrics across most classes shown in Figure 12, with particularly high performance in classes 2, 3, 4, and 8 (precision and recall ≥0.99). Class 7's F1-score is 0.74 whereas Class 6's is 0.66, indicating lesser performance. Maintaining constant macro and weighted averages, the model achieves a robust overall accuracy of 0.91.

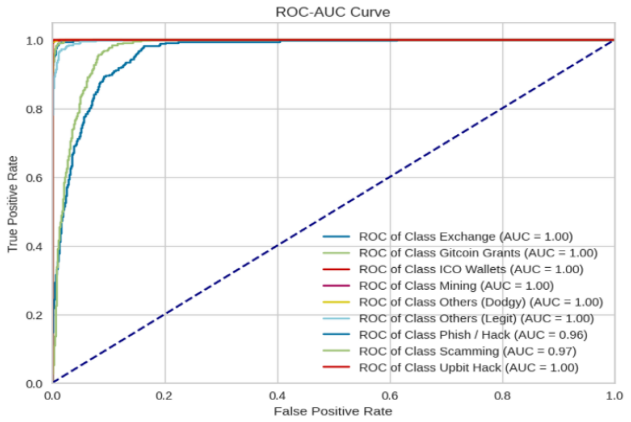


Fig. 13. Roc curve for cat Boost model

Figure 13 displayed the ROC-AUC curves for a CatBoost model classifying Ethereum transactions. Most classes achieve perfect or near-perfect AUC scores of 1.0, indicating excellent discriminatory power. Slightly lower AUCs of 0.96 and 0.97 for "Phish/Hack" and "Scamming," respectively, suggest slightly reduced performance in distinguishing these classes, though still remarkably high. Overall, the model demonstrates outstanding classification capabilities across all transaction types.

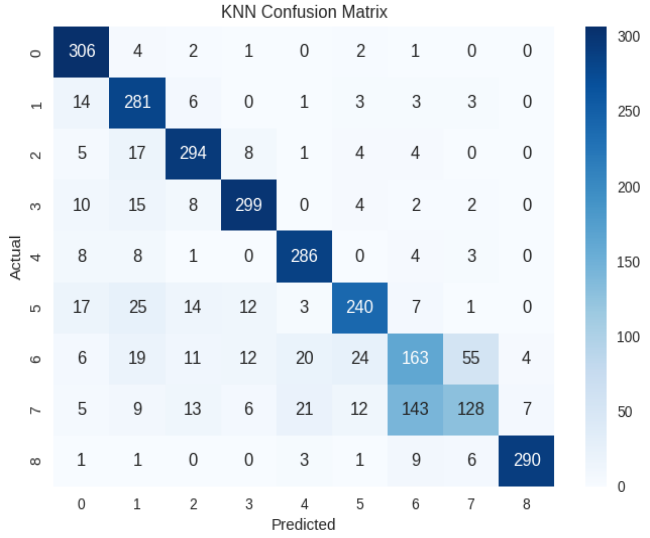


Fig. 14. Confusion matrix of KNN model

The KNN confusion matrix shows mixed performance across nine Ethereum address classes, as shown in Figure 14. The model performs well for classes 0, 4, and 8 (306, 286, 290 correct predictions) and solidly for classes 1, 2, and 3, though some misclassifications occur. Classes 5, 6, and 7 show notable confusion, with class 6 particularly problematic (163 correct, 55 misclassified as class 7) and class 7 often misclassified as class 6. These results suggest KNN struggles to distinguish address types with overlapping transactional patterns.

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.82 | 0.97 | 0.89 | 316 |
| 1 | 0.74 | 0.90 | 0.81 | 311 |
| 2 | 0.84 | 0.88 | 0.86 | 333 |
| 3 | 0.88 | 0.88 | 0.88 | 340 |
| 4 | 0.85 | 0.92 | 0.89 | 310 |
| 5 | 0.83 | 0.75 | 0.79 | 319 |
| 6 | 0.49 | 0.52 | 0.50 | 314 |
| 7 | 0.65 | 0.37 | 0.47 | 344 |
| 8 | 0.96 | 0.93 | 0.95 | 311 |
| accuracy | | | 0.79 | 2898 |
| macro avg | 0.79 | 0.79 | 0.78 | 2898 |
| weighted avg | 0.78 | 0.79 | 0.78 | 2898 |

Fig. 15. Classification report of K-neighbors

Figure 15 presents the classification report for K-neighbors transaction classification. It shows varying performance across classes, with classes 0, 2, 3, 4, 5, and 8 demonstrating relatively high precision, recall, and f1-scores. However, classes 6 and 7 exhibit significantly lower scores, indicating challenges in accurately classifying these transaction types, impacting the overall accuracy of 0.79, respectively.

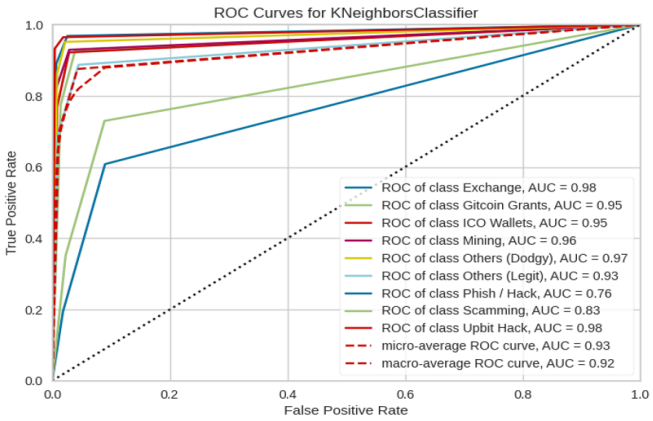


Fig. 16. ROC curve of K-Neighbors

Figure 16 shows the ROC curves for K-Neighbors transaction classification. Most classes achieve high AUCs, notably "Exchange" and "Upbit Hack" at 0.98. However, "Phish/Hack" (0.76) and "Scamming" (0.83) show lower performance, indicating reduced ability to distinguish these fraudulent classes. Overall strong but diverse performance is shown by the micro-average AUC of 0.93 and the macro-average AUC of 0.92.

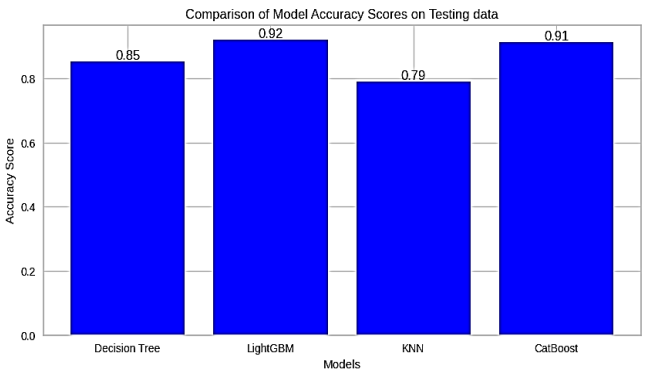


Fig. 17. Bar graph of accuracy comaprison between proposed mdoels

Figure 17 shows the comparison of accuracy among the four supervised machine learning models for Ethereum address classification. LightGBM achieves the highest

accuracy (0.92), followed closely by CatBoost (0.91), while Decision Tree reaches 0.85 and KNN lags at 0.79, highlighting the superior performance of gradient boosting methods for handling complex transactional patterns in the CEAT dataset.

### C. Discussion

The performance comparison of various models reveals distinct differences in classification results shows in Table III. The DT model achieved an Acc of 85.16%, with Prec of 84.85%, Rec of 85.16%, and F1score of 84.92%, indicating balanced performance but relatively lower compared to other models. LightGBM's superior accuracy (91.99%), high precision (91.89%), recall (91.99%), and F1-score (91.93%), all of which show that it effectively captures both genuine positives and negatives, surpasses all other models. KNN performed the least with an Acc of 78.91%, Prec of 78.49%, Rec of 78.91%, and F1score of 78.06%, showing lower performance. CatBoost was competitive with LightGBM, achieving 91.23% accuracy, 91.11% precision, 91.23% recall, and 91.13% F1 score. The Gaussian SVM model, with an accuracy of 78.47%, shows high precision (83.70%) but low recall (60.67%) and a poor F1-score (47.08%), indicating its failure in correctly identifying all positive instances. The AdaBoost model achieved the lowest accuracy of 67.7%, with a significantly high recall (98.8%), but low precision (39.3%) and F1-score (56.2%), highlighting its overfitting to positive class instances. Gradient Boosting showed moderate results with an Acc of 76.8%, Prec of 48%, and high Rec (97.9%) but a lower F1score (64.4%), indicating it is better at detecting positive cases but suffers from poor precision. Overall, LightGBM and CatBoost provide the most balanced and highest performance for Ethereum address classification.

TABLE III. COMPARISON BETWEEN PROPOSED MODELS AND ANOTHER MODEL PERFORMANCE FOR ETHEREUM TRANSACTION CLASSIFICATION

| Model | Accuracy | Precision | Recall | F1-score |
|---|---|---|---|---|
| DT | 85.16 | 84.85 | 85.16 | 84.92 |
| LightGBM | 91.99 | 91.89 | 91.99 | 91.93 |
| KNN | 78.91 | 78.49 | 78.91 | 78.06 |
| CatBoost | 91.23 | 91.11 | 91.23 | 91.13 |
| Gaussian SVM[15] | 78.47 | 83.70 | 60.67 | 47.08 |
| AdaBoost[40] | 67.7 | 39.3 | 98.8 | 56.2 |
| Gradient Boosting[40] | 76.8 | 48 | 97.9 | 64.4 |

The proposed models for Ethereum address classification, including LightGBM, CatBoost, Decision Tree, and KNN, offer distinct advantages and implications. LightGBM and CatBoost have best Acc, Prec, Rec, and F1score making it even suitable for large-scale and real-time blockchain data especially when handling categorical data. Decision Tree is simple, clear and offers fairly balanced performance; however, it has comparatively low accuracy and can be effectively used only as a basic model. When implementing KNN, the results show lower efficiency, which implies it is suitable for the case of small data sets. However, issues such as high imbalance, high computational cost, overfitting and high model complexity still persist, and solutions include tuning the hyperparameters, combining shallow and deep learning, SMOTE method for data preprocessing, as well as adopting real-time implementation measures and Explainable AI for trustful model interpretation.

## V. CONCLUSION AND FUTURE WORK

This research effectively shows how supervised ML can be used for classifying Ethereum addresses to improve the categorization of transactions. This means leaving a strong foundation for an effective data training since the methodology involved data preprocessing, feature selection, SMOTE-based data balancing and the Min-Max scaler normalization. DT, LightGBM, KNN, and CatBoost Classifier four classification models were developed and validated using several performance measures. In other words, the results indicated that LightGBM was the most accurate of the models with an accuracy of 91.99% and CatBoost Classifier had 91.23%. Accuracy was achieved as follows; Decision Tree was at 85.16% while the worst performer was KNN with 78.91 percent. LightGBM maintained its best performance metrics throughout all measurements of Prec, Rec and F1score. These outcomes indicate that LightGBM and CatBoost Classifier are the most effective models for Ethereum address classification in this context.

The study encounters limitations because its dataset is small and restricted in scope and does not account for all the aspects of Ethereum transactions. Supervised learning models restrict researchers from exploring alternative investigation methods during the analysis. Future studies may build a large database and consider incorporating from data that will enhance the accuracy of the model such as past addresses of the customer and their spending habits. It would also be beneficial to look into the possibility of improving the classification through classifying it as unsupervised learning methods, deep learning, and transfer learning. Real-time fraud detection systems using these models could also be developed to ensure better scalability and security for Ethereum transactions in dynamic environments.

## REFERENCES


[1] J. K. Chaudhary, S. Tyagi, H. P. Sharma, S. V. Akram, D. R. Sisodia, and D. Kapila, "Machine Learning Model-Based Financial Market Sentiment Prediction and Application," in *2023 3rd International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE)*, IEEE, May 2023, pp. 1456–1459. doi: 10.1109/ICACITE57410.2023.10183344.

[2] W. Chan and A. Olmsted, "Ethereum transaction graph analysis," in *2017 12th International Conference for Internet Technology and Secured Transactions, ICITST 2017*, 2018. doi: 10.23919/ICITST.2017.8356459.

[3] P. K. Vishwakarma and M. Suyambu, "The Assessments of Financial Risk Based on Renewable Energy Industry," *Int. Res. J. Mod. Eng. Technol. Sci.*, vol. 06, no. 09, pp. 758–770, 2024.

[4] S. Tyagi, "Analyzing Machine Learning Models for Credit Scoring with Explainable AI and Optimizing Investment Decisions," *Am. Int. J. Bus. Manag.*, vol. 5, no. 01, pp. 5–19, 2022.

[5] H. Kapadia and K. C. Chittoor, "Quantum Computing Threats to Web Encryption in Banking," *Int. J. Nov. Trends Innov.*, vol. 2, no. 12, pp. a197–a204, 2024.

[6] S. Hisham, M. Makhtar, and A. A. Aziz, "A comprehensive review of significant learning for anomalous transaction detection using a machine learning method in a decentralized blockchain network," *International Journal of Advanced Technology and Engineering Exploration*. 2022. doi: 10.19101/IJATEE.2021.876322.

[7] D. Patel, "Enhancing Banking Security: A Blockchain and Machine Learning- Based Fraud Prevention Model," *Int. J. Curr. Eng. Technol.*, vol. 13, no. 06, Dec. 2023, doi: 10.14741/ijcet/v.13.6.10.

[8] A. Q. Md, S. M. S. S. Narayanan, H. Sabireen, A. K. Sivaraman, and K. F. Tee, "A novel approach to detect fraud in Ethereum transactions using stacking," *Expert Syst.*, vol. 40, no. 7, Aug. 2023, doi: 10.1111/exsy.13255.


 

[9] G. Mantha, "Transforming the Insurance Industry with Salesforce: Enhancing Customer Engagement and Operational Efficiency," *North Am. J. Eng. Res.*, vol. 5, no. 3, 2024.

[10] H. Kali, "Optimizing Credit Card Fraud Transactions Identification and Classification in Banking Industry Using Machine Learning Algorithms," *Int. J. Recent Technol. Sci. Manag.*, vol. 9, no. 11, pp. 85–96, 2024.

[11] R. M. Aziz, M. F. Baluch, S. Patel, and P. Kumar, "A Machine Learning Based Approach to Detect the Ethereum Fraud Transactions with Limited Attributes," *Karbala Int. J. Mod. Sci.*, 2022, doi: 10.33640/2405-609X.3229.

[12] V. Verma, "Security Compliance and Risk Management in AI-Driven Financial Transactions," *Int. J. Eng. Sci. Math.*, vol. 12, no. 7, pp. 1–15, 2023.

[13] J. Eduardo A. Sousa *et al.*, "Fighting Under-price DoS Attack in Ethereum with Machine Learning Techniques," in *Performance Evaluation Review*, 2021. doi: 10.1145/3466826.3466835.

[14] S. B. Shah, "Improving Financial Fraud Detection System with Advanced Machine Learning for Predictive Analysis and Prevention," *Int. J. Sci. Res. Comput. Sci. Eng. Inf. Technol.*, vol. 10, no. 6, pp. 2451–2463, Nov. 2024, doi: 10.32628/CSEIT24861147.

[15] V. C. Oliveira *et al.*, "Analyzing Transaction Confirmation in Ethereum Using Machine Learning Techniques," in *Performance Evaluation Review*, 2021. doi: 10.1145/3466826.3466832.

[16] H. P. Kapadia, "Reducing Cognitive Load in Online Financial Transactions," *Int. J. Curr. Sci.*, vol. 12, no. 2, pp. 732–797, 2022.

[17] V. Verma, "Deep Learning-Based Fraud Detection in Financial Transactions: A Case Study Using Real-Time Data Streams," *ESP J. Eng. Technol. Adv.*, vol. 3, no. 4, pp. 149–157, 2023, doi: 10.56472/25832646/JETA-V3I8P117.

[18] N. Malali, "The Role of Devsecops in Financial AI Models: Integrating Security at Every Stage of AI/ML Model Development in Banking and Insurance," *Int. J. Eng. Technol. Res. Manag.*, vol. 6, no. 11, 2022, doi: 10.5281/zenodo.15239176.

[19] A. R. Bilipelli, "Forecasting the Evolution of Cyber Attacks in FinTech Using Transformer-Based Time Series Models," *Int. J. Res. Anal. Rev.*, vol. 10, no. 3, pp. 383–389, 2023.

[20] B. Mahesh, "Machine Learning Algorithms - A Review," *Int. J. Sci. Res.*, vol. 9, no. 1, pp. 381–386, 2020, doi: 10.21275/art20203995.

[21] A. Said *et al.*, "Detailed analysis of Ethereum network on transaction behavior, community structure and link prediction," *PeerJ Comput. Sci.*, 2021, doi: 10.7717/peerj-cs.815.

[22] R. M. Bani-Hani, A. S. Shatnawi, and L. Al-Yahya, "Vulnerability Detection and Classification of Ethereum Smart Contracts Using Deep Learning," *Futur. Internet*, vol. 16, no. 9, p. 321, Sep. 2024, doi: 10.3390/fi16090321.

[23] N. Ivanov, Q. Yan, and A. Kompalli, "TxT: Real-Time Transaction Encapsulation for Ethereum Smart Contracts," *IEEE Trans. Inf. Forensics Secur.*, 2023, doi: 10.1109/TIFS.2023.3234895.

[24] T. Saleem *et al.*, "Predicting functional roles of Ethereum blockchain addresses," *Peer-to-Peer Netw. Appl.*, vol. 16, 2023, doi: 10.1007/s12083-023-01553-2.

[25] R. M. Aziz, R. Mahto, K. Goel, A. Das, P. Kumar, and A. Saxena, "Modified Genetic Algorithm with Deep Learning for Fraud Transactions of Ethereum Smart Contract," *Appl. Sci.*, 2023, doi: 10.3390/app13020697.

[26] T. T. N. Pragasam, J. V. J. Thomas, M. A. Vensuslaus, and S. Radhakrishnan, "CEAT: Categorising Ethereum Addresses' Transaction Behaviour with Ensemble Machine Learning Algorithms," *Computation*, 2023, doi: 10.3390/computation11080156.

[27] E. Dritsas and M. Trigka, "Supervised Machine Learning Models to Identify Early-Stage Symptoms of SARS-CoV-2," *Sensors*, 2023, doi: 10.3390/s23010040.

[28] B. Boddu, "Serverless Databases Are the Future of Database Management," *https://jsaer.com/download/vol-6-iss-1-2019/JSAER2019-6-1-277-282.pdf*, vol. 6, no. 1, p. 5, 2020.

[29] B. Boddu, "Challenges and Best Practices for Database Administration in Data Science and Machine Learning," *IJIRMPS*, vol. 9, no. 2, 2021.

[30] B. Boddu, "Scaling Data Processing with Amazon Redshift DBA Best Practices for Heavy Loads," *Int. J. Core Eng. Manag.*, vol. 7, no. 7, 2023.

[31] M. Gopalsamy, "Identification and Classification of Phishing Emails Based on Machine Learning Techniques to Improvise Cyber Security," *IJSART*, vol. 10, no. 10, 2024.

[32] S. Mishra, "Handling Imbalanced Data : SMOTE vs . Random Undersampling"," *Int. Res. J. Eng. Technol.*, vol. 4, no. 8, 2017.

[33] H. Sinha, "An Examination of Machine Learning-Based Credit Card Fraud Detection Systems," *Int. J. Sci. Res. Arch.*, vol. 12, no. 2, pp. 2282–2284, Aug. 2024, doi: 10.30574/ijsra.2024.12.2.1456.

[34] V. N. G. Raju, K. P. Lakshmi, V. M. Jain, A. Kalidindi, and V. Padma, "Study the Influence of Normalization/Transformation process on the Accuracy of Supervised Classification," in *Proceedings of the 3rd International Conference on Smart Systems and Inventive Technology, ICSSIT 2020*, 2020. doi: 10.1109/ICSSIT48917.2020.9214160.

[35] S. Pandya, "Predictive Analytics in Smart Grids : Leveraging Machine Learning for Renewable Energy Sources," *Int. J. Curr. Eng. Technol.*, vol. 11, no. 6, pp. 677–683, 2021, doi: 10.14741/ijcet/v.11.6.12.

[36] H. Sinha, "Predicting Employee Performance in Business Environments Using Effective Machine Learning Models," *IJNRD - Int. J. Nov. Res. Dev.*, vol. 9, no. 9, pp. a875–a881, 2024.

[37] M. Fatima and M. Pasha, "Survey of Machine Learning Algorithms for Disease Diagnostic," *J. Intell. Learn. Syst. Appl.*, 2017, doi: 10.4236/jilsa.2017.91001.

[38] J. R. Quinlan, "Induction of decision trees," *Mach. Learn.*, 1986, doi: 10.1007/bf00116251.

[39] F. Torres-Cruz, S. Tyagi, M. Sathe, S. S. C. Mary, K. Joshi, and S. K. Shukla, "Evaluation of Performance of Artificial Intelligence System during Voice Recognition in Social Conversation," in *2022 5th International Conference on Contemporary Computing and Informatics (IC3I)*, IEEE, Dec. 2022, pp. 117–122. doi: 10.1109/IC3I56241.2022.10072741.

[40] C. Obi-Okoli, O. Jogunola, B. Adebisi, and M. Hammoudeh, "Machine Learning Algorithms to Detect Illicit Accounts on Ethereum Blockchain," in *Proceedings of the 7th International Conference on Future Networks and Distributed Systems*, New York, NY, USA: ACM, Dec. 2023, pp. 747–752. doi: 10.1145/3644713.3644838.

 