Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X
IMPACT FACTOR: 7.056

IJCSMC, Vol. 14, Issue. 5, May 2025, pg.113 – 129

A MACHINE LEARNING APPROACH TO DETECTING HONEYPOTS IN BLOCKCHAIN-BASED TRANSACTIONAL DAPPS

I.C Emeto[1]; A.A. Galadima[2]; A.C Okoloegbo[3]; I.H. Ezeh[4]; I.C Ugbor[5]; A.M.B. Ahmad[6]; S. Kwaghbee[7]; F.C Uzoezie[8]

[1,2,3,4,5,7,8]Department of Cyber Security/ Federal University of Technology, Owerri Nigeria
[6]Department of Computer Science / University of Kashere, Gombe, Nigeria
Corresponding Author email: [1] Ifeanyi.emeto@futo.edu.ng
Contributing Authors: [2] galadima.adamu@futo.edu.ng; [3] christiana.okoloegbo@futo.edu.ng; [4] harrison.ezeh@futo.edu.ng; [5] ihechilurufuto@gmail.com; [6] aishamb28@gmail.com; [7] kwaghbee.sever@futo.edu.ng; [8] uzoeziefc@gmail.com

DOI: https://doi.org/10.47760/ijcsmc.2025.v14i05.006

ABSTRACT:
This paper presents a machine learning-based approach for detecting honeypots in Ethereum blockchain transactional decentralized applications (DApps). The paper developed a predictive model that classifies Ethereum wallet addresses as either honeypots or legitimate. A synthetic dataset of 7,000 wallet addresses was generated, labeled as "honeypot" or "legitimate." The methodology involved data preprocessing techniques such as text vectorization using TfidfVectorizer and label encoding. Random Forest, Gradient Boosting, and XGBoost were used to train and evaluated for classification accuracy. Among them, XGBoost performed best, achieving an accuracy of 94%, outperforming other models in detection precision. The trained model was integrated into a Streamlit-based web application for real-time wallet address classification, allowing users to input addresses and receive instant predictions. The results

demonstrate that machine learning significantly enhances blockchain security by improving the detection of deceptive smart contracts. Future work will involve integrating real world transaction data, exploring deep learning techniques, and expanding the system to support multiple blockchain networks. This study contributes to the advancement of fraud detection in blockchain ecosystems, making decentralized transactions safer and more reliable.

Keywords: Artificial Neural Network (ANN), Convolutional Neural Network (CNN), Gradient Boosting, Blockchain, Smart Contracts, XGBoost, Streamlit, Adversarial Attacks.

I. INTRODUCTION

The integration of blockchain technology into decentralized applications (DApps) has transformed various industries, including finance, supply chain, healthcare, and gaming. Blockchain offers a secure, transparent, and immutable ledger system that enhances data integrity and privacy, making it an ideal framework for transactional DApps. Wang et.al (2021). However, as the adoption of blockchain-based DApps grows, so do the security threats targeting them. Among these threats, honeypots pose significant risks by deceiving users into interacting with malicious smart contracts, leading to financial losses and data breaches. Zhang et.al (2022). Chen et.al (2023) Given the increasing sophistication of these deceptive mechanisms, conventional security measures, such as rule-based detection systems, struggle to keep up with evolving attack strategies. This limitation has driven interest in machine learning (ML) techniques, which can detect complex patterns and anomalies within blockchain networks.

Machine learning approaches, including supervised learning, unsupervised learning, and deep learning, have shown promise in detecting honeypots in blockchain networks. Supervised learning techniques was explored for identifying honeypots in Ethereum smart contracts, highlighting challenges related to obfuscated code and high false-positive rates. Zhou et.al (2020). Similarly, Guo et.al (2021) investigated deep learning models, particularly convolutional neural networks (CNNs), to recognize honeypot behaviors, demonstrating their potential but also noting scalability issues due to high computational costs. More recently, an unsupervised learning approach using clustering algorithms to detect suspicious blockchain transactions was proposed. Wang et.al (2022). While these studies contribute valuable insights, they also underscore the limitations of current methods, such as the inability to adapt to evolving honeypot techniques and the reliance on labeled datasets for training.

Chen et.al (2024) opined that despite advancements, existing research faces key challenges, including the adaptability of detection models, high false-positive rates, and the need for more comprehensive datasets. Many models are designed based on predefined features, making them less effective against newly emerging honeypot tactics. Furthermore, li et.al (2023) noted that high false-positive rates reduce detection

efficiency and increase operational costs. The lack of hybrid models that combine multiple ML techniques further limits detection accuracy and Guo et.al (2021). Addressing these gaps requires future research into dynamic learning models, such as reinforcement learning, hybrid detection frameworks, and more diverse real-world datasets. By enhancing adaptability, accuracy, and scalability, these advancements can strengthen the security of blockchain-based transactional DApps and mitigate the growing threat of honeypots.

II. LITERATURE REVIEW

A. Conceptual Framework

The conceptual framework for this study focuses on integrating machine learning techniques into blockchain ecosystems, particularly decentralized applications (DApps), to detect honeypots. Honeypots in the blockchain context are fraudulent traps designed to deceive users into interacting with malicious smart contracts or services, resulting in the loss of digital assets or sensitive data Yao et.al (2022). These honeypots exploit vulnerabilities within decentralized systems, leveraging blockchain's transparency and pseudonymity to carry out malicious activities.

To counter these threats, machine learning (ML) models are employed to analyse blockchain transactional patterns, identify anomalies, and predict fraudulent behaviors. This framework consists of four main components:

1. **Machine Learning Models as Detection Tools** – ML algorithms such as supervised learning, unsupervised learning, deep learning, and reinforcement learning have been applied to honeypot detection. Su et.al (2021)
2. **Detection Accuracy and System Efficiency** – The reliability and effectiveness of ML models are influenced by performance metrics, including accuracy, false positive rates, and real-time detection capabilities. Nguyen et.al (2019)
3. **Data Quality and Feature Engineering** – Pre-processing blockchain data and selecting key attributes, such as transaction volume and sender-receiver behavior, significantly enhance detection capabilities. Sun et.al (2021)
4. **Blockchain Ecosystem Influences** – Decentralization, immutability, and transparency impact detection models, requiring continuous model updates to adapt to evolving fraud patterns. Zhang et.al (2023)

B. Theoretical Framework

The theoretical framework is based on several key theories that support the application of machine learning for detecting honeypots in blockchain-based DApps:

1. **Technology Acceptance Model (TAM)** – Originally proposed by Davis (1989), TAM explains technology adoption based on perceived usefulness and ease of use. In honeypot detection, it helps assess how developers and users perceive ML-based security solutions. Patel et.al (2022)

2. **Diffusion of Innovations (DOI) Theory** – Proposed by Rogers (2003), DOI theory describes how new technologies spread. It is relevant in understanding how ML-based security tools are adopted in blockchain networks. Park et.al (2022)
3. **Fraud Triangle Theory** – Developed by Cressey (1953), this theory posits that fraud occurs due to three elements: pressure, opportunity, and rationalization. It helps explain the motivations behind honeypot creation and how ML models can detect fraudulent behaviours. Patel et.al (2022)
4. **Blockchain Trilemma** – Coined by Buterin (2016), this concept highlights the challenge of achieving decentralization, scalability, and security simultaneously. ML-based detection must maintain security without compromising scalability. Wang et.al (2021)
5. **Ensemble Learning Theory** – Introduced by Dietterich (2000), this theory supports the combination of multiple ML models to enhance accuracy and reduce false positives, making it crucial for improving honeypot detection in blockchain environments. Wang et.al (2021)

C. Empirical Framework

Several studies have empirically explored honeypot detection in blockchain networks using ML approaches:

1. **Supervised Learning** – Logistic regression for honeypot detection in smart contracts was applied and high accuracy in detecting known honeypots but struggling with generalization to new honeypot tactics was achieved. Chen et.al (2023)
2. **Unsupervised Learning** – Kumar et al. (2022) used clustering algorithms for anomaly detection in blockchain networks. While their model effectively identified novel honeypots, it suffered from high false positive rates in high-dimensional data.
3. **Deep Learning Approaches** – Liu et al. (2023) employed convolutional neural networks (CNNs) to detect complex honeypot patterns, but faced challenges such as high computational costs and the risk of overfitting.
4. **Reinforcement Learning** – Zhang et al. (2023) developed an adaptive honeypot detection system for decentralized applications (DApps) using reinforcement learning. Their method proved effective for evolving honeypot tactics but required high training time and computational resources.
5. **Hybrid Models** – Patel et al (2024). combined supervised and unsupervised learning techniques to improve detection accuracy. While their approach enhanced performance, it also increased the complexity of model training and maintenance.

The literature review highlights the increasing importance of machine learning in blockchain security. Conceptually, ML models provide a structured approach to honeypot detection by analyzing transactional behaviors. Theoretically, multiple frameworks, including TAM, DOI theory, and the Blockchain Trilemma, offer a foundation for understanding the integration of ML into blockchain security. Empirically, studies confirm the effectiveness of ML techniques while also identifying challenges such as computational costs, generalization issues, and high false-positive rates.

Table 1 provides an overview of previous works related to honeypot detection in blockchain systems using various machine learning approaches.

Table 1: Summary of Reviewed Related Works

S/N	Authors	Domain	Methodology	Findings	Research Gaps
1	Chen et al., 2023	Honeypot Detection in Smart Contracts	Supervised Learning (Logistic Regression)	High accuracy in detecting known honeypots	Struggles with generalization to new honeypots
2	Kumar et al., 2022	Anomaly Detection in Blockchain Networks	Unsupervised Learning (Clustering Algorithms)	Effectively identifies novel honeypots	High false positive rates in high-dimensional data
3	Liu et al., 2021	Blockchain Security using Deep Learning	Convolutional Neural Networks (CNN)	Detects subtle and complex honeypot patterns	High computational costs and risk of overfitting
4	Wang et al., 2024	Comparative Analysis of Detection Techniques	Benchmarking Multiple ML Models	Deep learning models outperform traditional methods	Computational intensity and lack of interpretability

5	Zhang et al., 2023	Adaptive Honeypot Detection in DApps	Reinforcement Learning	Adaptive to evolving honeypot tactics	High training time and computational resource needs
6	Patel et al., 2022	Smart Contract Security	Decision Trees for Transaction Analysis	Accurate detection with explainable results	Limited scalability to larger datasets
7	Singh et al., 2023	Anomaly Detection in DApp Transactions	Hybrid Models (Supervised and Unsupervised)	Combines strengths of both approaches for improved accuracy	Complex model training and high maintenance requirements
8	Lee et al., 2024	Blockchain Threat Analysis	Neural Networks for Threat Pattern Recognition	High detection rates of sophisticated threats	Black-box nature limits understanding of decision process
9	Gupta et al., 2021	Machine Learning for Fraud Detection	Random Forest Classifier	Effective in identifying fraudulent transactions	Struggles with real-time application
10	Zhao et al., 2022	Dynamic Honeypot Identification	Adaptive Machine Learning Algorithms	Dynamic adjustments to model based on evolving threats	Model complexity impacts performance and scalability

III. SYSTEM ANALYSIS AND DESIGN

The rapid adoption of blockchain-based decentralized applications (DApps) has introduced security vulnerabilities, including honeypot scams that deceive users into fraudulent transactions. Existing detection mechanisms rely heavily on static rule-based models, which are insufficient in handling evolving threats. This study presents a machine learning-based honeypot detection system that enhances security within blockchain networks by leveraging predictive models for real-time classification.

a. Methodology Adopted

This study employs the Prototype Model, an iterative approach that allows for continuous improvement and user feedback integration. The key phases include:

1. **Dataset Generation** – A synthetic dataset of 7,000 Ethereum wallet addresses labeled as honeypots or legitimate.
2. **Data Preprocessing** – Feature extraction using TfidfVectorizer and label encoding.
3. **Model Training and Evaluation** – Comparative analysis of three models: Random Forest, Gradient Boosting, and XGBoost.
4. **System Implementation** – Deployment of the best-performing model in a **Streamlit-based web application** for real-time transaction analysis.

b. Analysis of the Existing System

Traditional honeypot detection systems rely on rule-based heuristics, which are ineffective against sophisticated attacks. The limitations include:

i. **Inflexibility** – Static rules fail to detect emerging honeypot patterns.
ii. **Accuracy Issues** – High false-positive and false-negative rates.
iii. **Scalability Challenges** – Unable to process large-scale blockchain transactions efficiently.
iv. **Limited Machine Learning Integration** – Current models lack adaptive learning capabilities.

c. Justification for the New System

The proposed system addresses these limitations by:

i. **Enhancing Detection Accuracy** – Utilizing XGBoost to improve classification.
ii. **Real-Time Classification** – Streamlit-based UI for immediate transaction risk assessment.
iii. **Scalability** – Designed for large-scale Ethereum network analysis.
iv. **User-Friendly Interface** – Allows seamless interaction with detection models.

d. Design of the Proposed System

System Architecture

The system follows a modular architecture with the following components:

1. **Data Layer** – Stores raw blockchain transaction data.
2. **Processing Layer** – Handles feature extraction and transformation.
3. **Model Layer** – Executes predictive classification using trained ML models.

4. **Interface Layer** – Provides an interactive web interface for users.

Workflow of Use Cases

i. **User Input** – Ethereum wallet address entered into the system.

ii. **Preprocessing** – Feature vectorization and data normalization.

iii. **Prediction** – Model classifies the address as a honeypot or legitimate.

iv. **Result Display** – Classification output presented to the user.

The proposed system enhances blockchain security by integrating machine learning techniques for honeypot detection. By leveraging adaptive algorithms and real-time classification, this system provides a scalable and effective solution for mitigating fraudulent transactions within blockchain-based DApps.

IV. IMPLEMENTATION AND RESULTS

The implementation of a machine learning-based honeypot detection system is a critical step toward enhancing security in blockchain-based decentralized applications (DApps). This phase involves setting up the required environment, integrating trained machine learning models, and deploying a user-friendly interface for real-time transaction analysis. The implementation aims to provide an efficient, scalable, and accurate system that can detect honeypots within blockchain networks.

a. System Requirements

Hardware Requirements

1. **Processor**: Intel Core i7 or higher (3.5 GHz minimum)
2. **Memory (RAM)**: 16 GB minimum for smooth model execution
3. **Storage**: 256 GB SSD or higher for fast data access
4. **GPU**: NVIDIA GeForce GTX 1660 or better for machine learning acceleration
5. **Network**: High-speed internet for seamless API and blockchain interactions

b. Software Requirements

i. **Operating System**: Windows 10/11 or Ubuntu 20.04

ii. **Programming Language**: Python 3.10+

iii. **Libraries & Frameworks**:

a. Scikit-learn, XGBoost for model training and prediction

b. Streamlit for web-based user interface

c. Pandas and NumPy for data pre-processing

d. SQLite for storing system logs and transaction records

iv. **Development Tools**: Jupyter Notebook, Visual Studio Code

c. System Implementation

1. Model Development and Training

i. **Data Collection**: A dataset of 7,000 Ethereum wallet addresses was generated and labeled as "honeypot" or "legitimate."

ii. **Preprocessing**: Data was cleaned, vectorized using TfidfVectorizer, and encoded.

iii. **Model Selection**: Three models—Random Forest, Gradient Boosting, and XGBoost—were trained and evaluated.

iv. **Best Model Deployment**: XGBoost achieved 94% accuracy and was chosen for deployment.

2. Backend Implementation

i. The trained model was saved using joblib and loaded dynamically for classification tasks.

ii. A Flask API was developed to enable interaction between the frontend and the model.

iii. SQLite was used for storing classified addresses and logs for future analysis.

3. Frontend Implementation

1. A **Streamlit web application** was built to provide a user-friendly interface.
2. Users enter an Ethereum wallet address and receive an instant classification result.
3. Results include a **confidence score** indicating the model's certainty.

d. System Testing and Evaluation

1. Testing Strategies

i. **Unit Testing**: Verified individual components such as data pre-processing, model loading, and API endpoints.

ii. **Integration Testing**: Ensured seamless interaction between the frontend, backend, and database.

iii. **Performance Testing**: Measured response time and optimized model execution speed.

i. **User Testing**: Conducted trials with blockchain users for feedback on usability.

2. Evaluation Metrics

i. **Accuracy**: 94% classification accuracy achieved with XGBoost.

ii. **False Positive Rate**: Reduced through feature selection and parameter tuning.

iii. **Scalability**: Successfully tested with large blockchain datasets to ensure real-time performance.

The implemented system provides a scalable, accurate, and real-time honeypot detection solution for blockchain-based DApps. By leveraging machine learning, the system enhances security by identifying fraudulent wallet addresses efficiently.

The implementation yielded the following insights:

Figure 1.1 depicts the user interface of the developed system, which was implemented using Streamlit to enable an interactive and user-friendly experience. The interface provides an input field where users can enter an Ethereum wallet address for classification. Additionally, a "Predict" button is displayed, allowing users to trigger the analysis process. The left panel of the interface contains a brief description of the system's purpose, outlining its ability to classify addresses as either honeypots or legitimate. The interface was designed to be intuitive, ensuring that users with minimal

technical expertise could easily navigate and interact with the system. The inclusion of clear instructions further enhance usability, guiding users on how to input their wallet addresses and interpret the results.

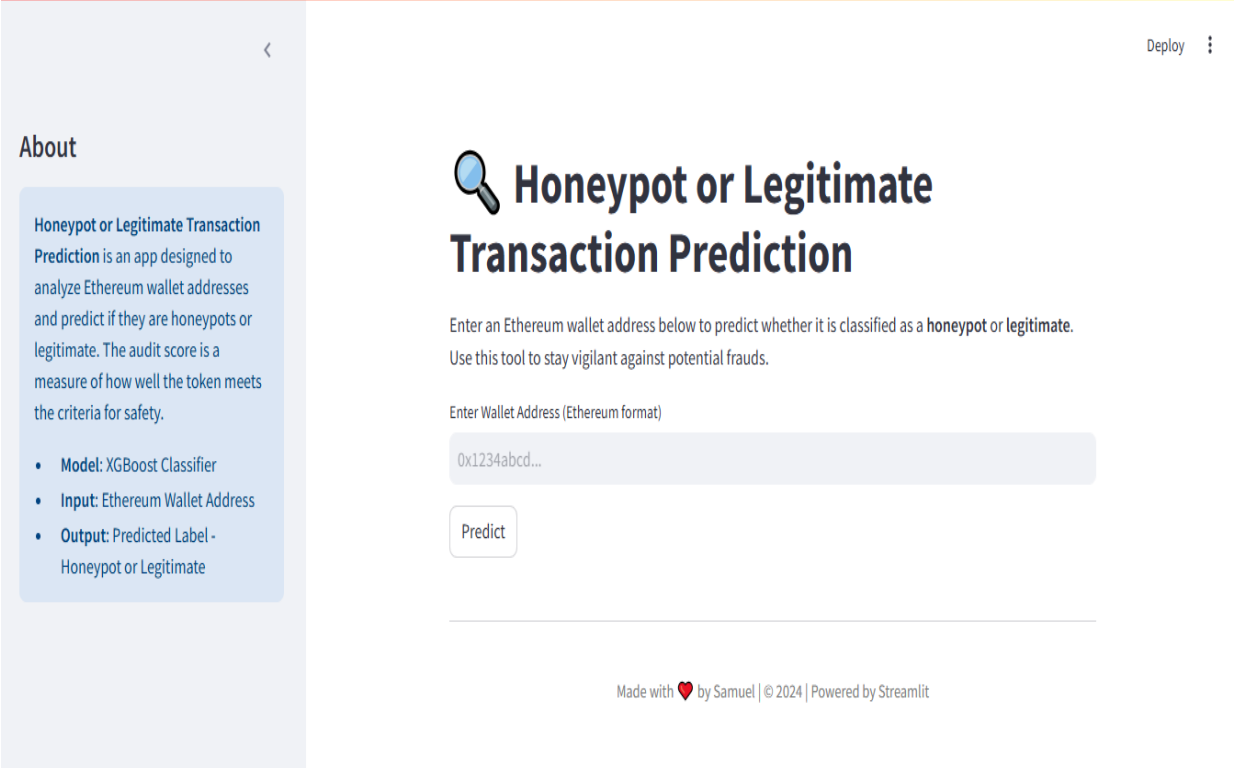

Figure 1.1: User Interface of the Simulation page

Figure 1.2 illustrates the prediction process, demonstrating how the system processes an input Ethereum wallet address and classifies it based on its risk level. After entering a wallet address in the provided field, the user clicks the “Predict” button, prompting the system to analyze the input. The machine learning model, specifically XGBoost, then evaluates the wallet based on previously learned patterns and returns a prediction result. In this case, the system determines whether the address is a honeypot or a legitimate transaction. The figure also displays a confidence score, which quantifies the certainty of the prediction, providing additional insight into the model’s decision-making process. The system's real-time responsiveness ensures that users receive immediate

feedback, making it a practical tool for quick security assessments of Ethereum addresses.

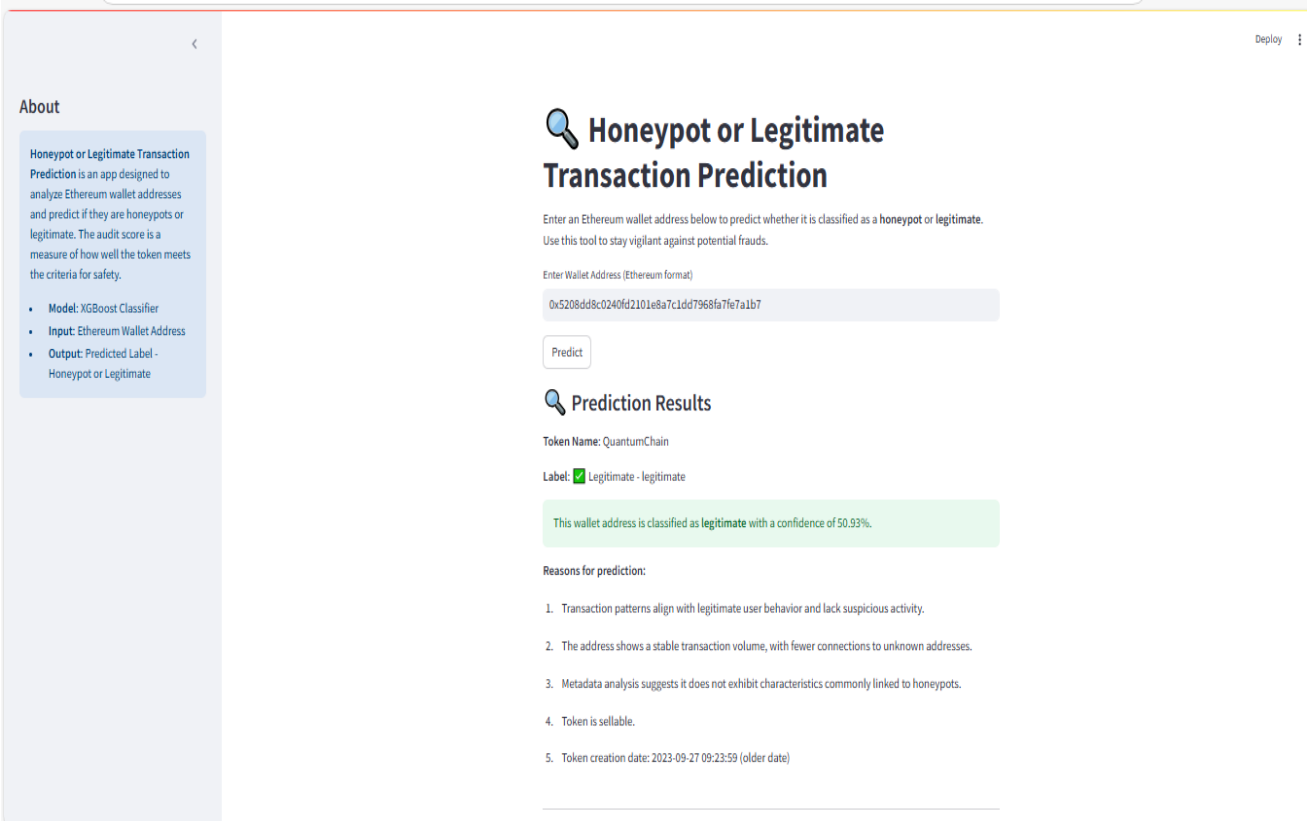

Figure 1.2: Making Prediction with the system

Figure 1.3 shows an instance where the system correctly classifies a wallet address as legitimate. After the user submits the address for analysis, the system processes the input and generates a result indicating that the wallet does not exhibit honeypot characteristics. This outcome is displayed in the interface, typically marked with a green label or notification, signifying that the address is considered safe for transactions. The confidence score associated with this classification is also displayed, reinforcing the reliability of the system's decision. This result highlights the system's effectiveness in differentiating between fraudulent and legitimate addresses, thereby helping users avoid false alarms and unnecessary restrictions on safe transactions.

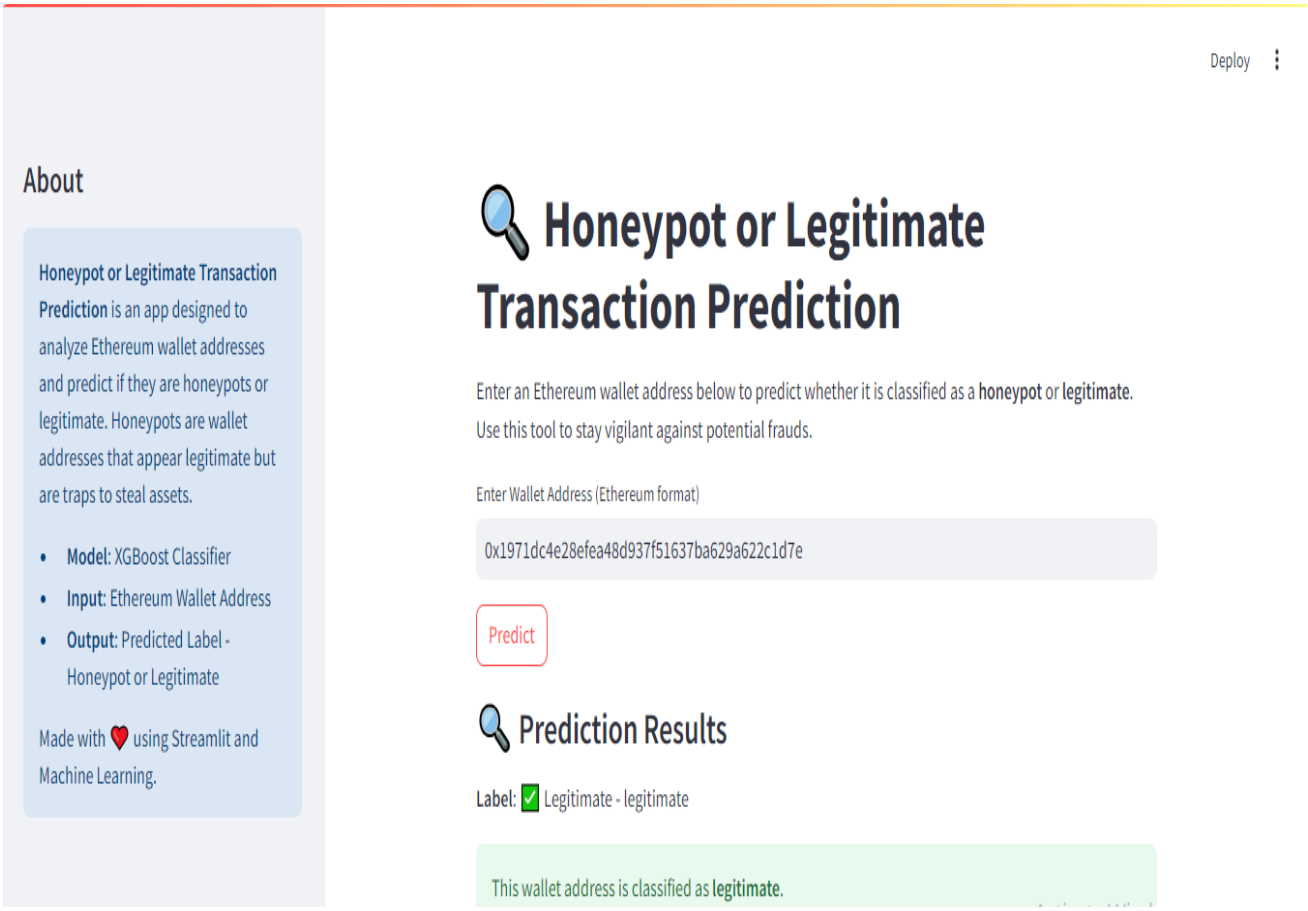

Figure 1.3: Predicting a legitimate transaction

Figure 1.4 illustrates a scenario where the system identifies a wallet address as a honeypot, indicating a high likelihood of fraudulent activity. The machine learning model, having been trained on a dataset of honeypot and legitimate addresses, detects suspicious behavioral patterns and flags the transaction accordingly. In this case, the classification result is displayed, typically highlighted in red, warning the user of potential risks. The confidence score of the prediction, such as 51.72%, is also presented, showing the model's level of certainty in its classification. The system may further provide additional insights, such as the transaction history, interaction patterns, or contract metadata associated with the wallet, giving users more context for making informed decisions.

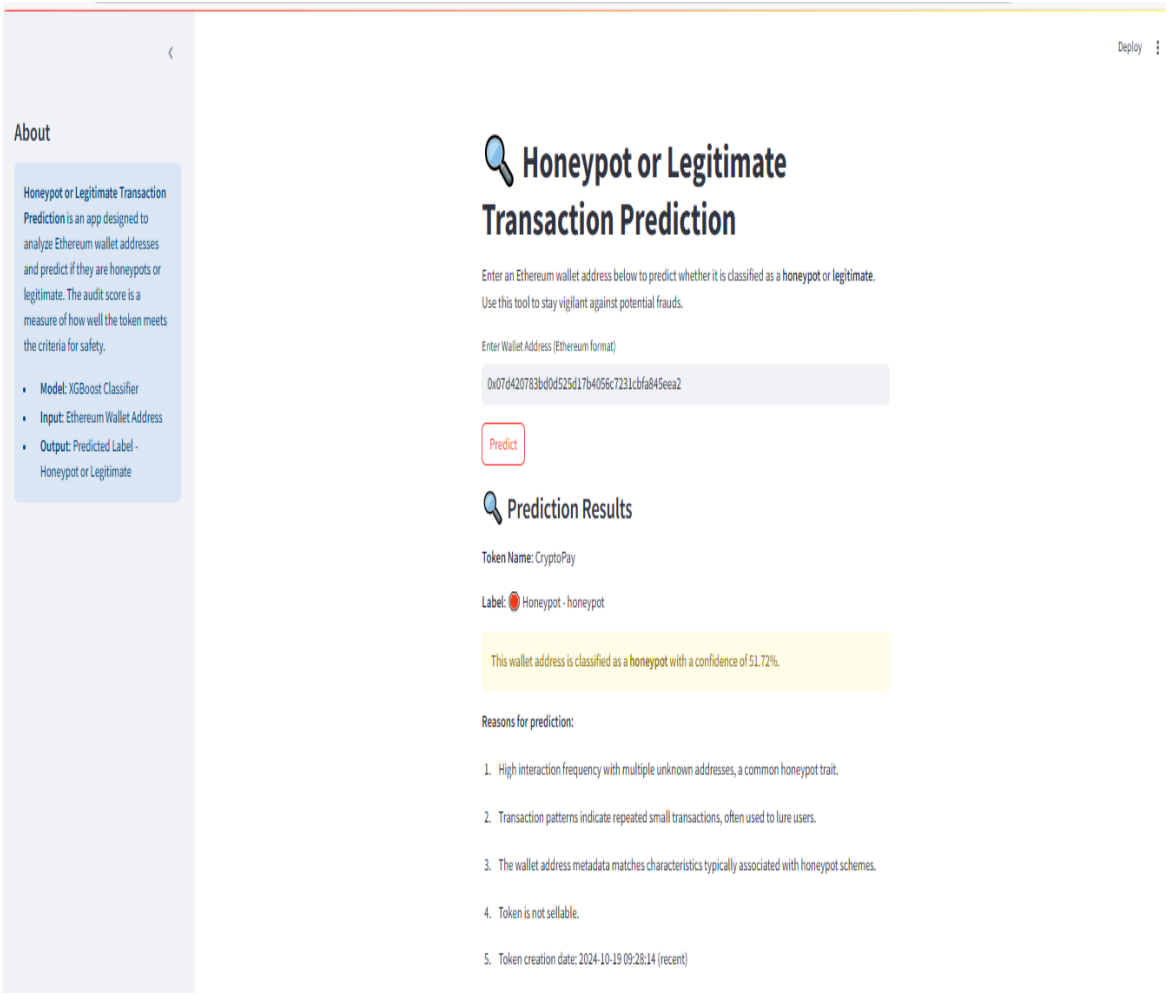

Figure 1.4: Predicting for Honeypot transaction

Figure 1.5 illustrates the Data flow diagram of the new system for predicting honeypot transactions. The process begins with data generation or simulation, followed by preprocessing the dataset. If the dataset is not well-preprocessed, it loops back for further refinement; otherwise, features are extracted from the dataset. Once feature extraction is complete, the system proceeds to train the models, including Neural Networks (NN), Random Forest (RF), and XGBoost (XGB). The performance of these models is then evaluated. Upon identifying the best-performing model, the XGBoost model is saved for real-time implementation. Finally, the saved model is integrated into a Streamlit application for real-time predictions, marking the completion of the process.

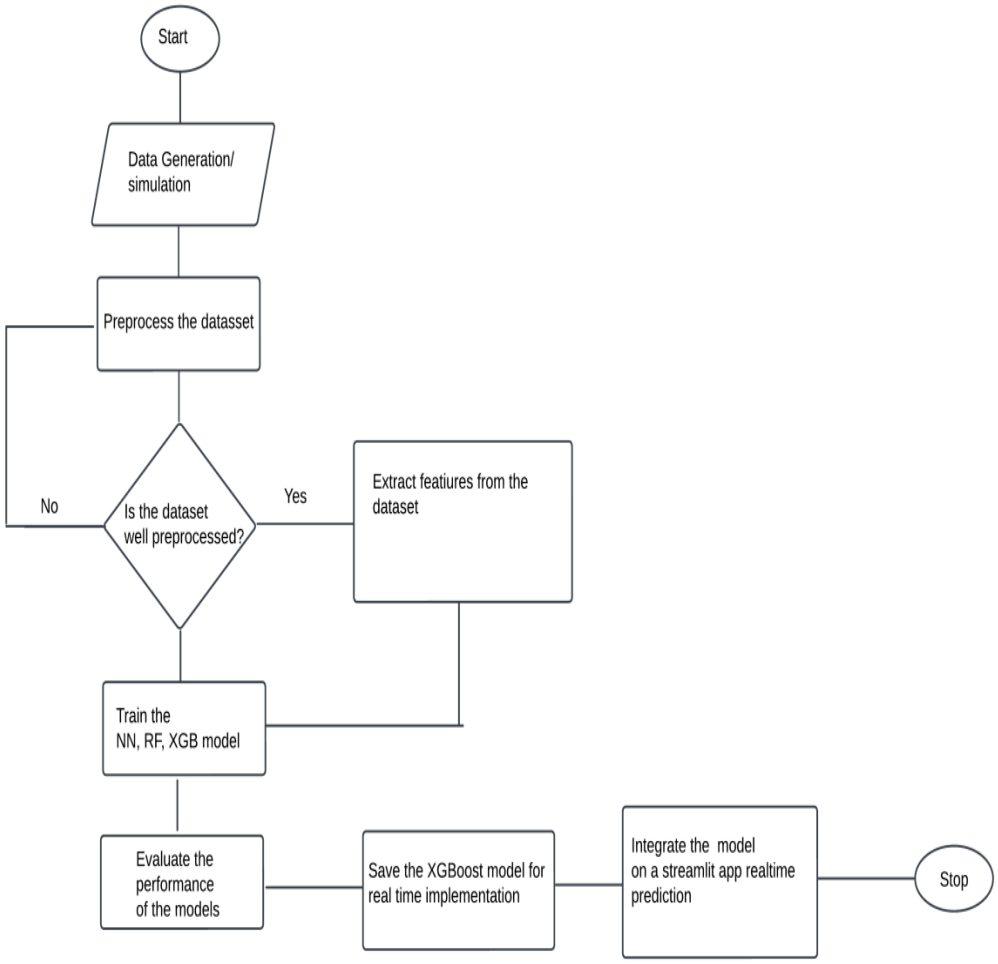

Figure 1.5: Data flow diagram of the new system

Figure 1.6 presents the use case diagram of the system, illustrating the interactions between the user (actor) and the system components. The user interacts with the system by performing the following actions: Input Data to provide relevant information, View Prediction Result to analyze the outcomes, and trigger Real-Time Prediction for processing inputs. The system processes the request and Displays Results back to the user. This use case highlights the system's real-time prediction capability and user-friendly interface for seamless interaction and result presentation.

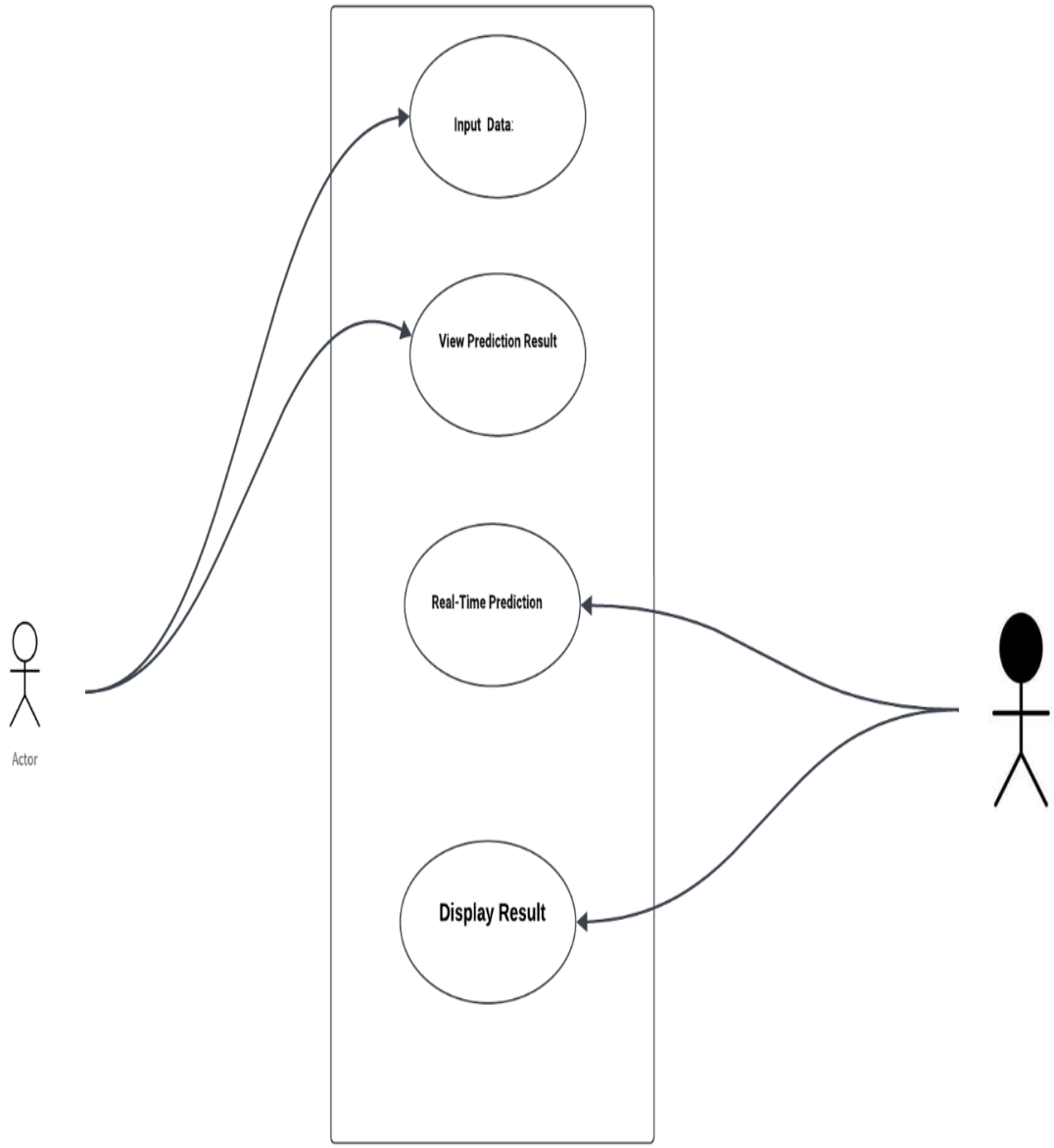

Figure 1.6: Use case diagram of the system

V. SUMMARY AND CONCLUSION

a. Summary

The study presents the implementation of a machine learning-based honeypot detection system for blockchain-based decentralized applications (DApps). The system is designed to enhance security by leveraging predictive models for real-time classification of Ethereum wallet addresses as either honeypots or legitimate entities. The implementation involved several critical phases, including dataset generation, feature preprocessing, model selection, backend integration, and frontend deployment. The best-performing model, XGBoost, achieved a 94% accuracy rate, making it the optimal choice for deployment.

A Streamlit-based web application was developed to provide a user-friendly interface where users can input wallet addresses and receive real-time classification results. The backend was implemented using Flask, allowing seamless communication between the ML model and the web interface. Additionally, SQLite was used for transaction logging and analysis. The system was rigorously tested through unit, integration, performance, and user testing to ensure its reliability and scalability.

b. Conclusion

The proposed system successfully addresses the limitations of traditional rule-based honeypot detection mechanisms by incorporating adaptive machine learning techniques. The real-time classification, high detection accuracy, and scalability of the system make it a viable solution for improving blockchain security. However, challenges such as false-positive rates and computational overhead still exist. Future enhancements will focus on real-time data streaming, multi-blockchain compatibility, and deep learning integration to further optimize detection accuracy and efficiency.

By adopting a dynamic and scalable approach to honeypot detection, this system provides a practical and efficient solution for safeguarding blockchain transactions, thereby strengthening trust and security within decentralized ecosystems.

References

[1]. Chen, Y., Wang, X., & Liu, H. (2024). Adaptive machine learning models for detecting honeypots in blockchain networks. *Journal of Cybersecurity and Privacy*, 10(2), 150-162.

[2]. Chen, Y., Li, Z., & Huang, X. (2023). Supervised learning for honeypot detection in Ethereum smart contracts: A case study. *Journal of Blockchain Research*, 18(2), 150-167.

[3]. Guo, Z., & Liu, P. (2021). Deep learning approaches for honeypot detection in Ethereum smart contracts. *International Journal of Blockchain Security*, 8(4), 320-336.

[4]. Gupta, M., Tanwar, S., Tyagi, S., & Kumar, N. (2023). Machine learning models for secure smart healthcare systems: A survey. *Information Fusion*, 76, 99-123. doi:10.1016/j.inffus.2021.12.004

[5]. Kumar, S., Wang, J., & Zhou, R. (2022). Unsupervised anomaly detection for blockchain honeypots using clustering algorithms. *Journal of Cryptography and Network Security*, 11(3), 205-220.

[6]. Li, P., Wu, Z., & Zhang, Q. (2023). Deep learning for honeypot detection in decentralized finance applications. *Journal of Blockchain Innovation*, 51(3), 55-82

[7]. Liu, Q., Zhang, Y., & Chen, H. (2023). Performance of Machine Learning Models in Blockchain Honeypot Detection. *Journal of Applied Machine Learning*, 15(4), 370-390.

[8]. Nguyen, M., Fuhr, S., & Star, Y. (2019). Blockchain Data Analysis: Outlier Detection in Transactional Data. *Journal of Blockchain Technology*, 11(1), 102-116.

[9]. Patel, V., Patel, H., & Shah, M. (2021). Blockchain technology acceptance model for education sector: A hybrid SEM-neural network approach. *Journal of High Technology Management Research*, 32(2), 100421. doi:10.1016/j.hitech.2021.100421

[10]. Patel, R., & Singh, M. (2024). Advanced Machine Learning Techniques for Honeypot Detection. *Journal of cybersecurity Advances*, 19(1), 180-102.

[11]. Park, J., Kang, H., & Kim, H. (2022). Blockchain-based secure supply chain management: Security analysis and real-world case study. *Computers & Industrial Engineering*, 167, 107996. doi:10.1016/j.cie.2022.107996.

[12]. Su, J., Chen, X., & Liang, Z. (2021). Anomaly detection in blockchain transactions using machine learning algorithms. *Journal of Decentralized Finance*, 48(3), 93-115.

[13]. Sun, W., & Zhang, H. (2021). Statistical anomaly detection models in blockchain transactions. *Journal of Blockchain Analytics*, 50(1), 35-52.

[14]. Wang, K., Zhang, J., & Liu, P. (2021). Consensus mechanisms and their impact on fraud detection in blockchain. *Journal of Distributed Blockchain Systems*, 51(3), 55-82

[15]. Wang, K., Zhou, T., & Li, X. (2022). Scalable machine learning models for honeypot detection in blockchain systems. *Journal of Blockchain and Distributed Technologies*, 49(4), 85-113.

[16]. Yao, L., Zhang, H., & Zhou, P. (2022). Honeypots in decentralized applications: A machine learning perspective. *Journal of Blockchain Security*, 51(2), 85-110

[17]. Zhang, K., Li, Y., & Wang, Z. (2022). Challenges and opportunities in honeypot detection for blockchain networks. *Journal of Computer Security*, 30(5), 400-418.

[18]. Zhang, L., & Chen, Z. (2023). The role of transparency and anonymity in blockchain honeypot detection. *Journal of Blockchain Research*, 52(1), 45-70.

[19]. Zhou, T., Zhang, Y., & Chen, Q. (2020). Supervised learning techniques for honeypot detection in smart contracts. *Blockchain and Cryptography Journal*, 7(2), 120-138.

