Fraud Detection Optimization in Open Metaverse Blockchain Transactions: Case study based on ML-GRU Neural Networks

Dr.Habib ZOUAOUI
habib.zouaoui@univ-relizane.dz
Faculty of economic sciences ,University of Relizane, Algeria
Orcid id: https: //orcid.org/0000-0001-7694-2473

Dr.Meryem-Nadjat NAAS
meryemnadjat.naas@univ-relizane.dz
Faculty of economic sciences ,University of Relizane, Algeria
Orcid id:https: //orcid.org/0009-0004-4018-1261

Submitted 6th November 2025
Accepted 9th February 9, 2026
Published 26th February, 2026

Abstract :

The study explores anomaly detection in blockchain transactions within the Open Metaverse Our proposed system aims to achieve three critical objectives: detecting fraudulent transactions by identifying suspicious patterns and anomalies, assessing transaction risks through the categorization of risk profiles, and analyzing user behavior to enhance security, personalize financial services, and improve user experiences. Through the meticulous classification of transactions for fraud detection, risk assessment, and user behavior analysis, this project aspires to bolster the security and transparency of Open Metaverse finance, setting the stage for a safer and more user-centric virtual financial landscape.

DATASET: -The dataset comprises 78,600 entries, each depicting a transaction within the metaverse. These entries are characterized by various attributes including the Timestamp, Hour of the Day, Sending and Receiving Addresses, Transaction Amount, Type, Location Region, and IP Prefix, along with user-centric data like Login Frequency, Session Duration, Purchase Patterns, Age Group, Risk Score, and potential Anomalies.

Source: https://www.kaggle.com/datasets/faizaniftikharjanjua/metaverse-financial-transactions-dataset/data

INTRODUCTION: -

The Open Metaverse is a growing digital space where users can trade, invest, and exchange virtual assets using blockchain technology. This technology ensures that transactions are secure and transparent. As the Open Metaverse expands, it offers new opportunities for economic activities in a decentralized and innovative environment.

DATA OVERVIEW: -The dataset includes 78,600 entries, each depicting a transaction within the metaverse. These entries are characterized by various attributes including the

- **Timestamp:** Date and time of the transaction.
- **Hour of Day:** Hour from the transaction timestamp.
- **Sending Address:**Sender's blockchain address.
- **Receiving Address:** Receiver's blockchain address.
- **Amount:** Simulated transaction amount.
- **Transaction Type:** Type of transaction (transfer, sale, etc.).
- **Location Region:** Simulated geographical region.
- **IP Prefix:** Simulated IP address prefix.
- **Login Frequency:** Login frequency by age group.
- **Session Duration:** Duration of user sessions in minutes.
- **Purchase Pattern**: Purchase behavior pattern**.**
- **Age Group:** User category based on activity history**.**
- **Risk Score:** Calculated risk score based on user and transaction data**.**
- **Anomaly:**Risk level (high, moderate, low).

Source: https://www.kaggle.com/datasets/faizaniftikharjanjua/metaverse-financial-transactions-dataset/data

SOFTWARE & TOOLS: -

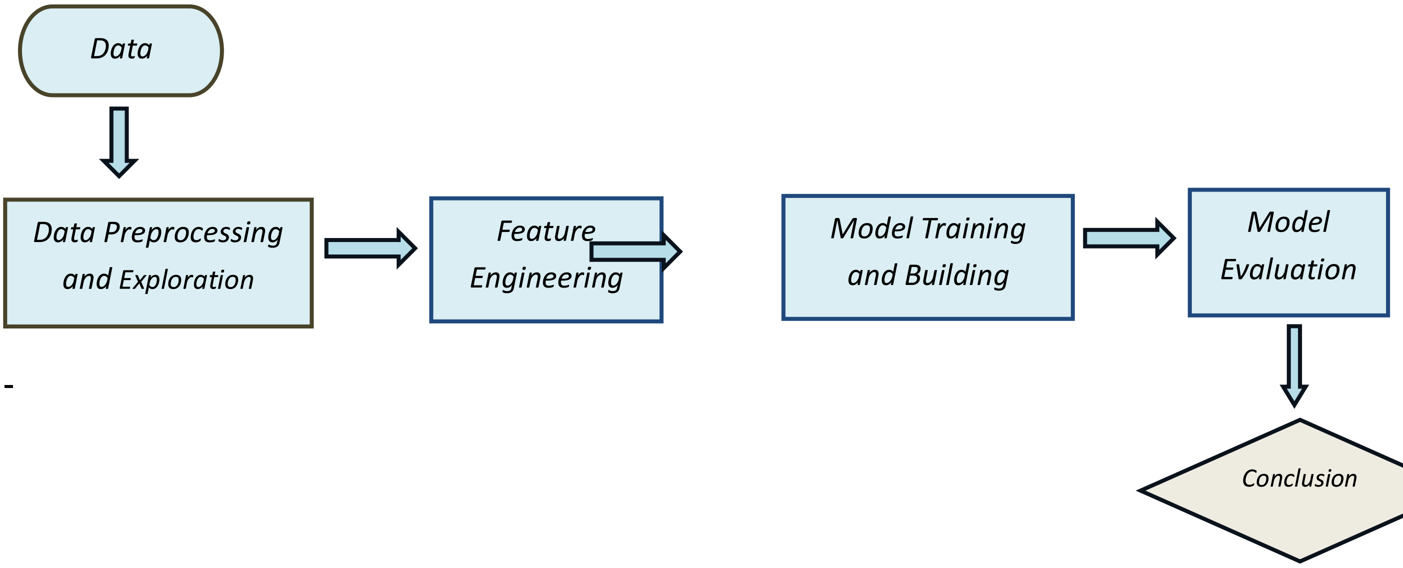
Programming Language: Python
Data Processing Libraries: Pandas & NumPy
Machine Learning Library: Scikit-learn
Visualization Tools: Matplotlib & Seaborn
Development Environments: Jupyter Notebook

OBJECTIVE: -

This project tackles these challenges head-on by leveraging the power of machine learning, specifically classification techniques. We aim to develop a robust system that analyzes blockchain financial transactions within the Open Metaverse. This system will be equipped to:

1. **Identify fraudulent transactions:**By classifying transaction patterns using machine learning models, the system will flag suspicious activities and anomalies, categorizing transactions based on their inherent risk profiles.
2. **Analyze user behavior:** Classification algorithms will help us understand user behavior patterns within the Open Metaverse financial landscape. This knowledge can be used to improve user experience, personalize financial services, and enhance overall security.

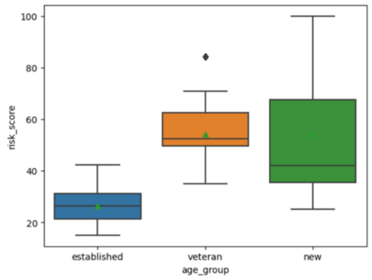
METHODOLOGY:

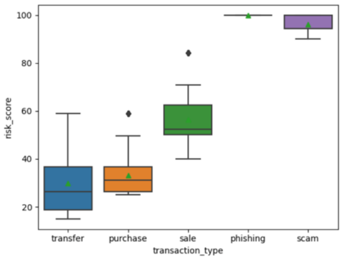

Data
Data Preprocessing and Exploration
Feature Engineering
Model Training and Building
Model Evaluation
Conclusion

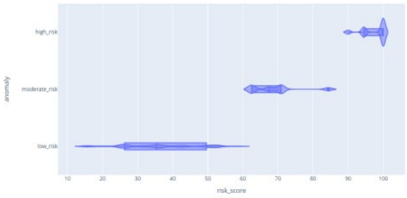
-

DATA PREPROCESSING AND EDA:

1.The dataset is cleansed by removing duplicates, converting timestamps to datetime format, and dropping irrelevant columns like sending and receiving addresses, and IP prefix.
2. A comparison of 'Risk-score' and 'Age-group' reveals that newer members have higher and more variable risk scores than more established groups.
3. Scam transactions have the highest risk scores, followed by transfers, with purchases and sales having the lowest.
4. The 'high risk' category shows significant spread and variability, suggesting multiple factors influence higher riskassessments.





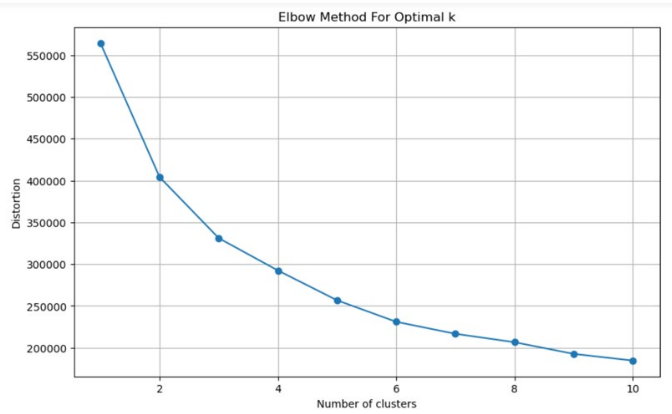


FINDINGS: -

I. ANALYZING USER BEHAVIOR

The first phase of model development involves preparing the data with essential Python libraries, selecting critical numerical and categorical features, and constructing a preprocessing pipeline using StandardScaler and OneHotEncoder. This setup ensures that our data is well-suited for the clustering algorithm by normalizing features and converting categorical data for better predictions.

After preprocessing the data, we apply the Elbow Method and determine that four clusters optimally represent our dataset, as evidenced by the significant decrease in distortions up to four clusters and minimal gains thereafter.



We then segment users into four distinct groups:

Cluster 0:

- Behavioral Traits: Engage infrequently with very low login frequencies and short session durations.
- Transaction Patterns: Mainly involved in purchases with a 'random' purchase pattern.
- Risk Profile: Exhibits the lowest risk levels, indicating cautious transaction behavior.

Cluster 1:

- Behavioral Traits: Highly active users with long session durations and frequent logins.
- Transaction Patterns: Primarily engaged in high-value sales.
- Risk Profile: Moderate risk scores with significant 'moderate_risk' anomalies.

Cluster 2:

- Behavioral Traits: Regular users with moderate engagement.
- Transaction Patterns: Focused solely on purchases with consistent behavior.
- Risk Profile: Very low risk, making it the safest among all clusters.

Cluster 3:

- Behavioral Traits: Similar in engagement to Cluster 0 but involved in risky behaviors.
- Transaction Patterns: Engaged in 'phishing' and 'scam' activities.

- Risk Profile: Contains all high-risk anomalies, indicating potentially fraudulent behavior.

This refined analysis helps in deploying targeted strategies for user engagement, risk management, and security enhancements tailored to the distinct needs of each cluster, thus fostering a secure and engaging environment in the OpenMetaverse.

II. IDENTIFY FRAUDULENT TRANSACTIONS

By classifying transaction patterns, the system will flag suspicious activities and anomalies, safeguarding the financial integrity of the Open Metaverse.

1. **Data Preparation:**
   - Loaded from "metaverse_transactions_dataset.csv".
   - Unnecessary columns like 'ip_prefix', 'timestamp', and 'risk_score' are dropped.
2. **Feature Encoding:**
   - The 'anomaly' column is label-encoded to transform risk levels into numerical labels suitable for logistic regression.
   - Categorical variables such as 'transaction_type', 'location_region', 'purchase_pattern', and 'age_group' are one-hot encoded to ensure distinct and non-ordinal treatment in the model.
3. **Data Splitting:**
   - The data is split into a training set (80%) and a testing set (20%) to facilitate model training and subsequent evaluation.

MODEL 1: LOGISTIC REGRESSION

A logistic regression model is initialized and trained on the processed training data. The model is configured to iterate up to 1000 timesto converge on the optimal coefficients.The model was tasked with classifying metaverse transactions into three categories of risk: 'high risk', 'low risk', and 'moderate risk'. The performance of the model across these classes was evaluated using precision, recall, and F1-score metrics. Below is a summary of how the model performed for each risk category, along with overall accuracy and average metrics.

The model's performance in classifying risk in transactions shows:

- **High Risk (Class 0):**
  - Precision, Recall, F1-Score: 100% each, indicating perfect detection and identification of high-risk transactions.
  - Support: 1,251 transactions.
- **Low Risk (Class 1):**
  - Precision, Recall, F1-Score: 97% each, showing high accuracy in identifying low-risk transactions.
  - Support:12,848 transactions, the largest class.
- **Moderate Risk (Class 2):**
  - Precision, Recall, F1-Score: 79% each, suggesting room for improvement.
  - Support:1,621 transactions.

Overall Model Performance:

- **Accuracy**: 96%, reflecting high effectiveness across all classes.
- Macro Average: 92% for precision, recall, and F1-score, indicating good uniform performance.

- Weighted Average:96%, accounting for class prevalence, confirming strong performance in predominant classes.

MODEL 2: DECISION TREES

Decision trees are a non-parametric supervised learning method that recursively splits the dataset based on features to create a tree-like structure for classification.

1. Initial Decision Tree Model:

Model Description: The initial Decision Tree Classifier achieved perfect accuracy on the test dataset. This may be sign of overfitting so we tried hyperparameter tuning with grid search may be helpful to address this issue.

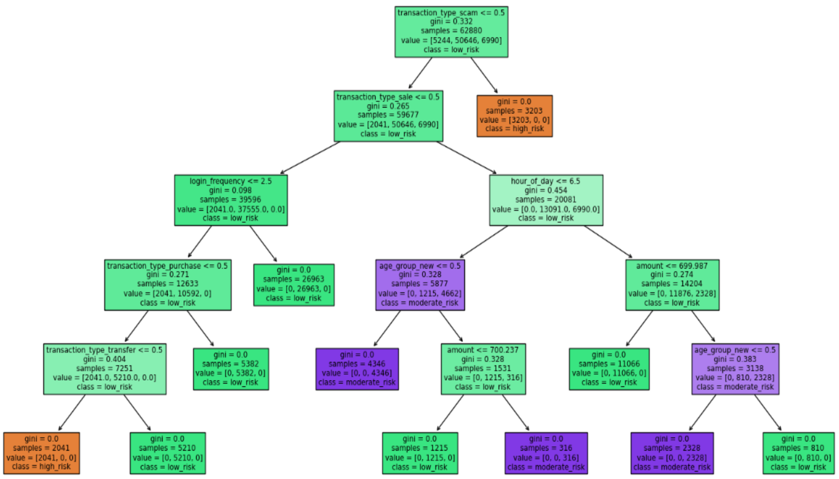
2. Hyperparameter Tuning with Grid Search:

Grid Search was employed to tune the hyperparameters of the Decision Tree Classifier. The best parameter configuration found was a maximum depth of 5. The final Decision Tree model trained with these parameters achieved perfect accuracy on the test dataset.

Model Evaluation:

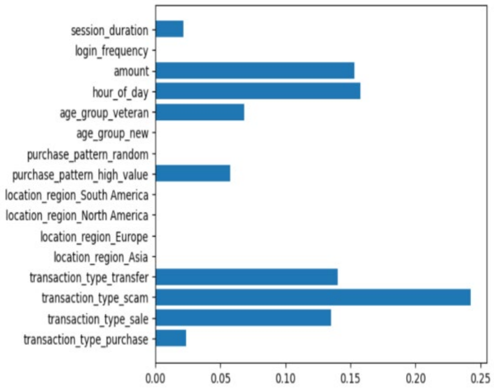
The Decision Tree model achieved an accuracy of 100% on the test dataset, indicating that it can effectively classify transactions as fraudulent or legitimate. The best hyperparameter configuration obtained through GridSearchCV was a maximum depth of 5, which suggests that the model's complexity was appropriately tuned to avoid overfitting.

Decision Tree and Feature Importance Analysis:



```
plt.barh(feature_names,reduced_decision_tree_model.feature_importances_)
```

```
<BarContainer object of 16 artists>
```



The feature importance analysis reveals that the most significant features for detecting fraudulent transactions are:
Transaction Type (e.g., purchase, sale, scam, transfer), Hour of the Day, Amount of Transaction, Age Group (e.g., new, veteran). Transaction types such as scams and sales contribute the most to the model's decision-making process, followed by features related to the hour of the day and transaction amount.

Based on the evaluation results, Utilize the trained Decision Tree model with a maximum depth of 5 for real-time fraud detection in transaction data.Focus on monitoring transactions categorized as scams and sales, especially during specific hours of the day.Investigate transactions with unusual amounts or involving new or veteran users, as they may exhibit anomalous behavior indicative of fraud.In conclusion, the Decision Tree model offers high accuracy and interpretable decision-making, making it a valuable tool for detecting fraudulent transactions in the dataset.

Model 3: SVM

Accuracy: The SVM model achieved an accuracy of approximately 83.21% on the test data. This indicates that the model correctly classified about 83.21% of the transactions as either fraudulent or non-fraudulent.

Interpretation: While the accuracy is decent, further analysis is needed to understand the model's performance across different classes and to identify areas for improvement.

Linear SVM with Hyperparameter Tuning:

Best Parameters: The best parameters obtained through grid search for the Linear SVM model were {'C': 0.001}, with a corresponding cross-validated accuracy score of approximately 95.45%.

Accuracy: After hyperparameter tuning, the Linear SVM model achieved an accuracy of approximately 95.87% on the test data.

- **Precision:** This metric indicates the proportion of correctly predicted instances among all instances predicted as belonging to a particular class.
  - ✓ Class 0 (high_risk): Precision of 100% indicates that all instances predicted as high risk were indeed high risk.
  - ✓ Class 1 (low_risk): Precision of 97% indicates that 97% of instances predicted as low risk were actually low risk.
  - ✓ Class 2 (moderate_risk): Precision of 84% indicates that only 84% of instances predicted as low risk were low risk.
- **Recall:** This metric indicates the proportion of correctly predicted instances among all instances that belong to a particular class.
  - ✓ Class 0 (high_risk): Recall of 100% indicates that all actual high-risk instances were correctly identified as high risk.
  - ✓ Class 1 (low_risk): Recall of 98% indicates that most of the actual low-risk instances were correctly identified as low risk.
  - ✓ Class 2 (moderate_risk): Recall of 74% indicates that only 74% of actual moderate-risk instances were correctly identified as moderate risk.

MODEL 4: RANDOM FOREST CLASSIFIER:

A basic Random Forest Classification model is initialized and trained on the processed training data. It has returned an accuracy of 100%. As this model is without any hyperparameter tuning it is returning perfect accuracy and trying to overfit the data. To address this issue, we performed a model with grid-search with hyperparameter tuning that performs hyperparameter tuning using GridSearchCV.The GridSearchCV is used to search for the best combination of hyperparameters (n_estimators and max_depth) for the Random Forest model.The parameter grid specifies different values for n_estimators (50, 100, 200) and max_depth (3, 5, 7).The cv parameter is set to 5 for 5-fold cross-validation.

Best Parameters and Best Score:

After fitting the GridSearchCV object to the training data (X_train, y_train), the best parameters (best_params) and best score (best_score) are obtained.In this case, the best parameters are {'max_depth': 7, 'n_estimators': 200}

ModelEvaluation:

Predictions are made on the test data (X_test), and the test accuracy is calculated using accuracy_score.The test accuracy in this case is approximately 99.98%.

Featureimportance:

- ✓ Hour of Day: The hour of the day also has high importance, suggesting that certain times of the day may be more associated with fraudulent activities.
- ✓ Transaction Type: Different transaction types such as scams, sales, and transfers are also important features, indicating that the type of transaction influences the likelihood of fraud.
- ✓ Amount: The transaction amount is an essential predictor, with higher amounts potentially indicating higher risk.

MODEL 5: VOTING CLASSIFIER

1. Base Classifiers Configuration:

- Decision Tree Classifier: Configured with a maximum depth of 5.

- Logistic Regression: Uses a 'liblinear' solver with regularization parameter C set to 1.0.
- SVM (Support Vector Machine): Uses an RBF kernel with regularization parameter C set to 1.0, and the gamma parameter set to 'scale', which automatically adjusts gamma based on the number of features, making it suitable for probability estimation.

2. Voting Classifier:

- The ensemble method used is a soft voting classifier, which means it predicts the class label based on the argmax of the sums of the predicted probabilities, providing a way to weight classifiers based on their confidence.

3. Model Training and Evaluation:

- The combined classifier is trained on a dataset partitioned into training and testing sets.
- After training, the model is used to predict class labels on a test dataset.

4. Performance Metrics:

- The accuracy of the classifier on the test data is approximately 99.13%.
- The classification report shows high precision, recall, and F1-score across the classes, indicating excellent model performance.
    - Class 0: Precision, recall, and F1-score are all 1.00.
    - Class 1: Precision is 0.99, recall is 1.00, and F1-score is 0.99.
    - Class 2: Precision and F1-score are slightly lower at 1.00 and 0.96, respectively, with a recall of 0.92.

<u>CONCLUSION: -</u>

By segmenting users into distinct groups based on behavior and risk, targeted strategies can be developed to enhance user engagement and minimize risks. This segmentation allows for personalized user experiences and more effective fraud prevention measures tailored to the risk profile of each group. Throughout the analysis of various machine learning models designed to identify and manage user behavior and transaction fraud in the Open Metaverse, each model offered unique insights and demonstrated high capabilities in its respective applications. However, among all tested models, the Voting Classifier emerged as the most effective due to its robust performance metrics and the strategic advantage of combining multiple models to mitigate their individual weaknesses. The following reasons are given below:

1. High Accuracy: The Voting Classifier achieved an impressive accuracy of approximately 99.13%, demonstrating its ability to accurately classify and predict different risk levels associated with user transactions.
2. Strength in Ensemble: By integrating the predictions from a Decision Tree, Logistic Regression, and SVM, the Voting Classifier leverages the collective strengths of these models. This ensemble approach reduces the impact of any single model’s bias or variance, leading to more reliable predictions.
3. Precision and Recall: This model showed exceptional precision and recall across different risk categories, especially in identifying high-risk transactions, which are critical for fraud prevention. This is vital in maintaining the financial integrity and security of the Open Metaverse.
4. Flexibility and Scalability: The ensemble nature of the Voting Classifier allows for easy scalability and adaptability. It can be fine-tuned or expanded with additional models or data as the Open Metaverse grows and evolves.