Journal of Information Systems Engineering and Management
2025, 10(45s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Enhancing Fraud Detection in UPI Transactions using Ensemble Learning and Neural Networks

Ms. Viha Dave[1], Mr. Dhaval Chudasama[2]
[1]PG Student, Department of Computer Engineering, Gandhinagar University.
[2]Assistant Professor, Department of Cyber Security, Gandhinagar University.

ARTICLE INFO

Received: 29 Dec 2024
Revised: 15 Feb 2025
Accepted: 24 Feb 2025

ABSTRACT

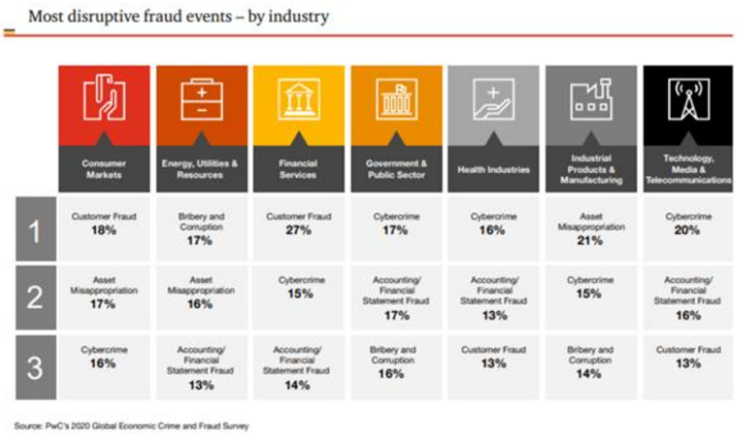
The prevalence of Unified Payments Interface (UPI) transactions has become ubiquitous among individuals, who have come to rely on this method for numerous essential daily transactions. The utilization of UPI transactions is now widespread, encompassing small vendors to large firms. Individuals find UPI more convenient than alternative payment methods due to its ease of use. However, concurrent with the evolution of digitalization in online payments through UPI transactions, the risk of experiencing fraudulent activities has also increased proportionally. Perpetrators of fraud have enhanced their methodologies to manipulate and defraud unsuspecting individuals, resulting in financial losses. Traditional fraud prevention approaches, such as awareness programs, have proven insufficiently effective, as individuals continue to fall victim to the schemes devised by fraudsters, ultimately incurring monetary losses. Consequently, there is a significant need for solutions that can contribute to loss prevention before fraudsters succeed in causing disruptions and adversely affecting individuals' financial status.

Keywords: Fraud Detection; UPI; Fraud in UPI transactions; ensemble learning; machine learning; Gradient boosting; LightGBM; XGBoost; Neural networks; MLP classification

INTRODUCTION

The rapid growth of mobile payments, particularly Unified Payments Interface (UPI) transactions, is being facilitated by diverse individuals utilizing the system for both sending and receiving funds. These individuals encompass Mobile Network Operators, service providers, vendors, merchants, and users who significantly rely on UPI for their financial transactions. While UPI offers ease and convenience of use, there are concurrent challenges in digital payment security, as fraudulent actors have evolved their strategies to exploit vulnerabilities, resulting in various forms of financial fraud. [5]

Statista Company's data reveals that the global e-commerce industry incurred losses of 20 billion US dollars in 2021 due to online Payment Fraud. The Reserve Bank of India (RBI) reported approximately 9,103 instances of bank fraud across the country in fiscal year 2022, indicating a slight increase from the previous year and reversing a decade-long trend. The total amount lost to these fraudulent activities amounted to 604 billion Indian rupees, a decrease from 1.38 trillion rupees. This figure represents an increase of over 14% when compared to the 17.5 billion dollars documented in the preceding year. Financial institutions are actively implementing measures to combat this form of deception. [2]



Most disruptive fraud events – by industry

	Consumer Markets	Energy, Utilities & Resources	Financial Services	Government & Public Sector	Health Industries	Industrial Products & Manufacturing	Technology, Media & Telecommunications
1	Customer Fraud 18%	Bribery and Corruption 17%	Customer Fraud 27%	Cybercrime 17%	Cybercrime 16%	Asset Misappropriation 21%	Cybercrime 20%
2	Asset Misappropriation 17%	Asset Misappropriation 16%	Cybercrime 15%	Accounting/ Financial Statement Fraud 17%	Accounting/ Financial Statement Fraud 13%	Cybercrime 15%	Accounting/ Financial Statement Fraud 16%
3	Cybercrime 16%	Accounting/ Financial Statement Fraud 13%	Accounting/ Financial Statement Fraud 14%	Bribery and Corruption 16%	Customer Fraud 13%	Bribery and Corruption 14%	Customer Fraud 13%

Source: PwC's 2020 Global Economic Crime and Fraud Survey

Figure 1 [3] Percentage of different fraud events categorized by the industry

The UPI payment service provider applications support Peer to Peer (P2P), Peer to Business (P2B) and Business to Peer (B2P) transactions, which was not the case in the existing digital payment systems. Additionally, it facilitates bank transfer payments by providing the option to add multiple bank accounts on a single application. The key factors contributing to a UPI transaction are a sender, a receiver, service provider and network operator. [5]

Digital payment systems have evolved to such an extent that the convenience of using them is substantial. The widespread adoption of digital payments, particularly UPI transactions, has created opportunities for fraudsters to employ various deceptive tactics to manipulate users and perpetrate fraud. Consequently, there is a need for a system that can comprehensively analyze the nature of UPI transactions and assess the likelihood of fraudulent activity. However, this objective cannot be achieved without techniques that can provide precise results with minimal human intervention and assist in analyzing specific traits of such transactions to verify the authenticity of UPI transactions. [1]

Enhancing the security of UPI transactions can be accomplished by analyzing the patterns and characteristics of UPI transaction data categorized as fraudulent and non-fraudulent. The application of machine learning can be highly effective as it employs an approach based on comprehensive observation of the data, followed by the training of a model capable of accurate and precise classification. This methodology can not only generate results based on data patterns but also contribute to creating a trustworthy and secure environment by detecting potential frauds that may affect users. [4] This paper focuses on detecting frauds from UPI transaction data using ensemble techniques such as XGBoost, LightGBM and MLP. The results of fraud detection are presented as a binary classification, categorizing each transaction as either fraudulent or non-fraudulent. The implementation involves various steps, including data collection, data cleaning and preprocessing, and execution of the defined algorithms to extract results.

LITERATURE SURVEY

In order to understand how the solution can be implemented related to the problem addressed, we need to have the proper review about the existing work on the same. So here, various studies have been identified and analysed which mentions the approaches and methodologies involved. Following is the table which gives summary of the literature survey conducted on this problem which can help in identifying the base for implementing the solution.

Table 1 Comparative Literature Survey

Name of the study	Algorithms used	Findings in the study
L.SaiSampanPatrudu, D.Noorisha, E.Sidharatha, V.SaiRagaSudha, DrK.V.Satyanarayana. DynamicFraud Detection in UPI Transactions. Industrial Engineering Journal, 53(04). (2024).	Deep Q-Network algorithm from Deep Reinforcement Learning	Focuses on getting rewards rather than prioritizing the higher accuracy in implementation [1].
Manav Mangukiya, Meet Savani, Anuj Vaghani,	Decision tree, Random	Implemented both convolutional

Aryan Khunt. Financial Fraud Detection Approaches Using Machine Learning. Gujarat Technological University, PAPER ID: PCP386. (2023).	Forest, Logistic Regression, XGBoost, LSTM and CNN	and deep learning methods and also proposed some other approaches using flow charts and explanation of the same [2].
Krishnan Chari. Fraud Risk in a Digitized Fintech ecosystem Troubling trends, issues and approaches to mitigate Fraud Risk. (2020).	Methods of mitigating the risks in using the digital payments.	Shows the causes of increasing frauds in this digitalized era like lack of literacy and awareness in people and also proposes different ways to mitigate the risks [3].
Harshith Kumar S, Dr. H.R. Divakar. UPI Fraud Detection Using Machine Learning. International Research Journal of Modernization in Engineering Technology and Science, 06(08). (2024). https://www.doi.org/10.56726/IRJMETS60849	Decision tree, Random forest and Support Vector Machine	They have shown the architectural flow diagram for system implementation and implemented the problem with the algorithms on a dataset [4].
Ashish Joy, Dr Rejikumar, Dr Dhanya M. Payment App's Revamping, Perspectives from Text Mining Analysis. SIMSARC 2018. (2019). https://doi.org/10.4108%2Feai.18-12-2018.2285156	Text Mining methods	Perceptions of the users were collected through different platforms and text mining analysis was performed [5].
Dr. S. Jamuna, Dr. J.R. Gaur, Anshu Singh, Dharam Barot. A REVIEW RESEARCH ON ONLINE FINANCIAL FRAUDS IN INDIA. The American Journal of Management and Economics Innovations, 05(01), 1-7. (2023).	Different methods to improve users awareness and literacy	Gives the statistical analysis of increased frauds in digital payment methods and also describes different types of frauds affecting the security of the overall ecosystem [6].
J. Kavitha, G. Indira, A. Anil kumar, A. Shrinita, D. Bappan. Fraud Detection In UPI Transactions Using ML. *EPRA International Journal of Research and Development (IJRD), 09(04). (2024). https://doi.org/10.36713/epra16459	CNN	Gave the characteristics of existing and the proposed system in a simple and clear way [7].
Mr. R. Ramakrishnan, S. Vanisri, D. Yuvalakshmi. Unified Payment Interface Seamless Transaction Using RNN Model. INTERNATIONAL JOURNAL OF PROGRESSIVE RESEARCH IN ENGINEERING MANAGEMENT AND SCIENCE (IJPREMS) , 04(05), 1279-1283. (2024). https://www.doi.org/10.58257/IJPREMS34325	RNN	Implements the fraud detection using Continuous Authentication with Sequential Sampling and Recurrent Neural Networks [8].
Sayalee S. Bodade, P.P. Pawade. Review Paper on UPI Fraud Detection Using Machine Learning. International Journal for Research in Applied Science & Engineering Technology (IJRASET), 11(12). (2023). https://doi.org/10.22214/ijraset.2023.57551	CNN	Implemented CNN to perform fraud detection in UPI transactions [9].
Yarramreddy Chandrasena Reddy, Polavarapu Nagendra Babu, Venkata Sai Pavan Ravipati, Velpula Chaitanya. UPI Fraud Detection Using Convolutional Neural Networks(CNN). Research Square. (2024). https://doi.org/10.21203/rs.3.rs-4088962/v1	FNN, CNN, Decision Trees, Naïve Bayes, Logistic Regression (L1 and L2 regularization)	Performs the predictive analysis using the given algorithms and also gives comparative result studies for each of them [10].

M. ValavanAnita B. Desai, Dr. Ravindra Deshmukh and S. Rita. Predictive-Analysis-based Machine Learning Model for Fraud Detection with Boosting Classifiers. (2022). 10.32604/csse.2023.026508	Decision Tree, Gradient Boosting, Linear Regression and Random Forest	On a dataset requested from a lending website, the given algorithms are implemented and their comparative analysis is given [11].

METHODS

Ensemble Learning

Due to significant advancements in technology, machine learning predictions require newly developed strategies that can accommodate the nature of data produced in order to achieve the expected accurate results. Consequently, relying solely on a single machine learning model may not always be sufficient to obtain optimal results. To address all conditions within the dataset and to achieve accurate results across various situations, it is necessary to employ a collection of models that can make predictions on the same dataset using different combinations and subsequently combine their results. This approach can encompass all conditions and provide accurate predictions in diverse scenarios. This technique of combining results from multiple models is known as ensemble learning.

There are primarily three types of ensemble learning: Bagging, Boosting, and Stacking

Bagging

Bagging is a technique that leverages the results of different base models to produce output. It is also referred to as bootstrap sampling, as the original training dataset is randomly split into various subsets and then distributed to different base models, which can be any machine learning models. These weak learners are trained on the provided dataset and subsequently make predictions. In regression problems, the predictions made by the base models are averaged to produce the final prediction, while in classification tasks, a maximum voting method is employed to derive the final prediction from the predictions made by base models. An example of the bagging technique is the Random Forest algorithm

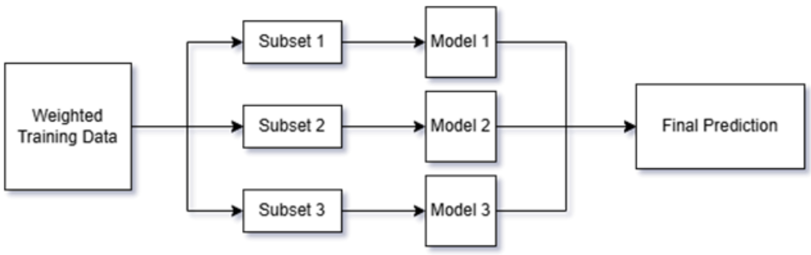


Figure 2 Bagging

Boosting

The boosting technique also considers multiple models, but its implementation differs from the bagging technique. A weighted training dataset is input into a model. The boosting technique utilizes the incorrect predictions or errors from one model and provides them as input to another model, which subsequently passes its errors to the next model. Through this process, the boosting technique generates a strong learner from a series of diverse weak learners. During the training process, instances that were incorrectly classified are assigned higher weights to prioritize them when fed to the successor model. Additionally, stronger models are assigned higher weights compared to weaker models to appropriately consider them when combining their outputs. Examples of the boosting technique include Gradient Boost, XGBoost, and LightGBM.

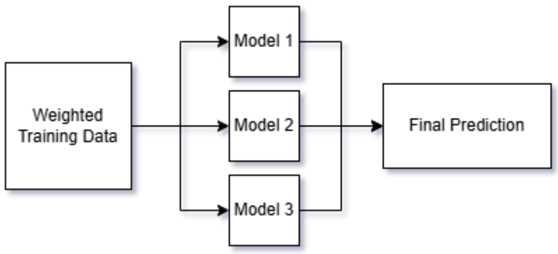


Figure 3 Boosting

Stacking

Stacking utilizes the predictions of multiple base models as input for a meta-model. The predictions made by the meta-model are considered the final prediction. The base models and the meta-model must not be of the same type. An example of the stacking technique is a Decision Tree paired with a Support Vector Machine.

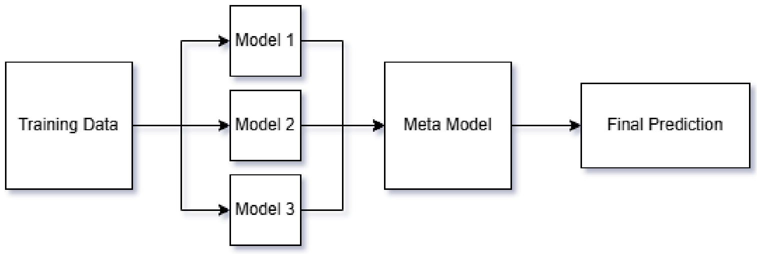


Figure 4 Stacking

Here in this usecase, two techniques from boosting is being used:

XGBoost

XGBoost, an abbreviation for Extreme Gradient Boosting, is an optimized version of Gradient Boosting. It develops decision trees in series by correcting errors from previous trees. The XGBoost algorithm functions by minimizing the objective function as given below:

Objective function = $\sum_{i=1}^{n} \quad L(y_i, \hat{y}_i) + \sum_{t=1}^{T} \quad \Omega(f_t)$

where,

$L(y_i, \hat{y}_i)$ is the loss function

$\Omega(f_t)$ is regularization term

T is the number of trees

Here, loss function is

$$L(y,p) = -\sum_{i-1}^{n} \quad y_i \log(p_i) + (1-y_i)\log(1-p_i)$$

where,

y_i is true label

p_i is predicted probability

XGBoost works by following steps:

Step-I: Take the average of target variable and consider it as first prediction.

Step-II: Calculate Gradient g_i and Hessian h_i by following formula:

$g_i = p_i - y_i$

$h_i = p_i(1 - p_i)$

Here, Gradient measures how much the prediction is wrong and Hessian helps control how big the updates should be.

Step-III: Construct a new tree by splitting data into nodes.

The leaf node value can be calculated as:

$$w* = -\frac{\sum g_i}{\sum h_i + \lambda}$$

where, λ is the regularization parameter

Step-IV: Update the predictions using the following formula:

$F_t(x)=F_{t-1}(x)+\eta \cdot$Tree Output

where, η is the learning rate

Then, the process gets repeated for other trees.

In XGBoost, the regularization parameter helps in preventing overfitting as the following:

$$\Omega(f_t) = \gamma T + \frac{\lambda}{2}\sum \left(w_j^2\right)$$

where,

T is the number of leaf nodes in the tree.

γ is the penalty for additional leaf nodes.

λ prevents large values in leaf weights.

LightGBM

LightGBM is an open-source framework developed by Microsoft. It is notably fast, accurate, and efficient compared to XGBoost as it employs leaf-wise tree growth rather than level-wise growth. The algorithm selects the leaf node with the largest loss reduction and expands it. Its histogram-based feature binning facilitates easier handling and expedites predictions.

LightGBM works by following steps:

Step-I: Create the histograms by grouping the continuous feature values into discrete bins for reducing complexity and time.

Step-II: Take the average of target variable and consider it as first prediction.

Step-III: Optimize the following objective function:

Objective function = $\sum_{i=1}^{n} L(y_i, \hat{y}_i) + \sum_{t=1}^{T} \Omega(f_t)$

where,

$L(y_i, \hat{y}_i)$ is the loss function

$\Omega(f_t)$ is regularization term

T is the number of trees

Here, loss function is

$$L(y,p) = -\sum_{i-1}^{n} \quad y_i \log(p_i) + (1 - y_i)\log(1 - p_i)$$

where,

y_i is true label

p_i is predicted probability

Step-IV: Compute the Gradient and Hessian by following formula:

$g_i = p_i - y_i$

$h_i = p_i(1 - p_i)$

Step-V: Split the leaf nodes with highest loss reduction for making the tree greedy in the growth. This makes the tree grow in Greedy manner.

Step-VI: Update the predictions using the following formula:

$F_t(x)=F_{t-1}(x)+\eta \cdot$Tree Output

where, η is the learning rate

To control overfitting, it includes L1 and L2 regularization:

$$(f_t) = \gamma T + \frac{\lambda}{2}\sum \quad \left(w_j^2\right)$$

where,

T is the number of leaf nodes in the tree.

γ is the penalty for additional leaf nodes.

λ prevents large values in leaf weights.

The implementation includes the multi-layer perceptron from neural networks also which is as follows:

Multi-layer Perceptron

A multi-layer perceptron is a feedforward neural network. It comprises three layers: input layer, hidden layer, and output layer. Multiple hidden layers may be present. The neurons in one layer are fully connected with the neurons in the subsequent layer. The input layer is visible as it merely transmits the input to its next layer without modification. The hidden layers neither receive input directly nor send outputs directly to the environment. The output layer is responsible for producing a single output value or a vector of values. Linear or non-linear activation functions are employed in the defined layers. In binary classification problems, the sigmoid function is utilized.

If your inputs are supposed as input vector (x_1, x_2, x_3, ... , x_n), output is Y_n and learning rate is α. Then assign the weights and biases for all the connections in the network in the range [-0.5, +0.5] and follow the given steps:

Step-I: Forward Propagation: Calculate Input and Output for the first layer i.e input layer.

It is a direct function which simply passes the input to the output without making changes.

Input at node j 'I_j' can be $I_j = x_j$

where, x_j is the input given.

So the output will be $O_j = I_j$

Now the net input at the jth node in the output layer can be

$$I_j = \sum_{i=1}^{n} O_i w_{ij} + x_0 * \theta_j$$

where,

O_i is output from ith node

w_{ij} is weight in the link between ith and jth node

x_0 is input to bias node 0 which is always supposed as 1

θ_j is the weight in the link from bias node 0 to jth node

Then, output at jth node is given by

$$O_j = \frac{1}{1 + e^{-I_j}}$$

where,

$I_j$is the input received at jth node

Also, calculate error at the node in output layer.

Error = $O_{Desired}$ - $O_{Estimated}$

where,

$O_{Desired}$ is the desired output of the node

$O_{Estimated}$ is the estimated output of the node

Step-II: Backward Propagation: Calculate error at each node:

For each kth unit in output layer:

$Error_k = O_k(1 – O_k) (O_{Desired} - O_k)$

where,

O_k is the output at kth node

$O_{Desired}$ is the desired output

For each jth unit in hidden layer:

$$Error_j = O_j (1 - O_j) \sum_k Error_k w_{jk}$$

where,

O_j is the output at jth node

$Error_k$ is the error at kth node in output layer

w_{jk} is the weight in the link between jth and kth node

Update the weights:

$$\triangle w_{ij} = \propto \times Error_j \times O_i$$

$$w_{ij} = w_{ij} + \triangle w_{ij}$$

where,

O_i is the output at ith node

$Error_j$ is the error at jth node

$\propto$ is the learning rate

w_{ij} is the weight in the link from ith node to jth node

$\triangle w_{ij}$ is the difference in weight that needs to added to w_{ij}

Update the biases:

$\triangle \theta_j = \propto \times Error_j$

$\theta_j = \theta_j + \triangle \theta_j$

where,

$Error_j$ is error at jth node

$\propto$ is the learning rate

θ_j is the bias from 0^{th} node to jth node

$\triangle \theta_j$ is the difference in bias that needs to be added to θ_j

RESULTS

The results obtained from all three algorithms—multi-layer perceptron, XGBoost, and LightGBM—were observed to be satisfactory, with LightGBM outperforming the others due to its leaf-wise growth nature and ability to produce faster results compared to other algorithms. The results are compared using performance metrics such as accuracy, precision, recall, and F1-score.

Accuracy

Measures the proportion of correctly predicted instances out of all the instances.

$$\text{Accuracy} = \frac{\text{TP+TN}}{\text{FP+FN+TP+TN}}$$

where,

TP (True Positive): Correctly predicted positive instances

TN (True Negative): Correctly predicted negative instances

FP (False Positive): Incorrectly predicted positive instances (Type I error)

FN (False Negative): Incorrectly predicted negative instances (Type II error)

Precision

Measures proportion of the predicted positive instances are actually positive.

$$\text{Precision} = \frac{TP}{TP+FP}$$

Recall

Measures proportion of actual positive instances were correctly predicted.

$$\text{Recall} = \frac{TP}{TP+FN}$$

F1 score

F1-score is the **harmonic mean** of balanced precision and recall.

$$\text{F1-score} = 2 \times \frac{\text{Precision}\times\text{Recall}}{Precision+Recall}$$

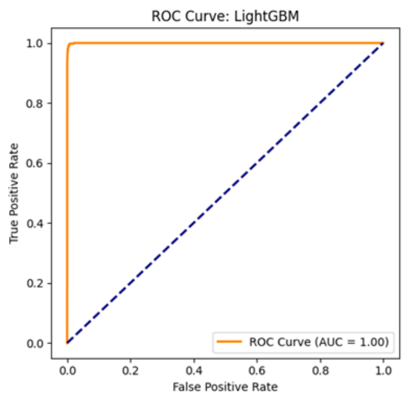
The results obtained on the first dataset using the algorithms that are selected. The comparative study of average precision can be more efficient while analyzing the performance of the model in case of imbalanced datasets and is given as bellow:

Table 2 Analysis of result metrics obtained from dataset-1

Algorithm	**Average precision on dataset-1**
MLP	0.9884904268272598
XGBoost	0.9719961418792108
LightGBM	0.9812803212687959

The ROC curve and precision-recall curves are very helpful in result analysis of an algorithm implemented on datasets whose values are not very balanced. The ROC curve plots the True Positive Rate vs False Positive Rates.AUC-ROC score is the area under the ROC curve which interprets better model performance when it is closer to 1. Precision-Recall curves plot precision vs recall at different threshold values. Higher precision and recall indicates the better model performance.

The ROC curve and PR curve for the algorithms are given as below:



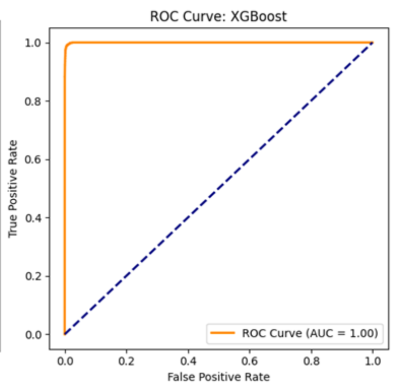
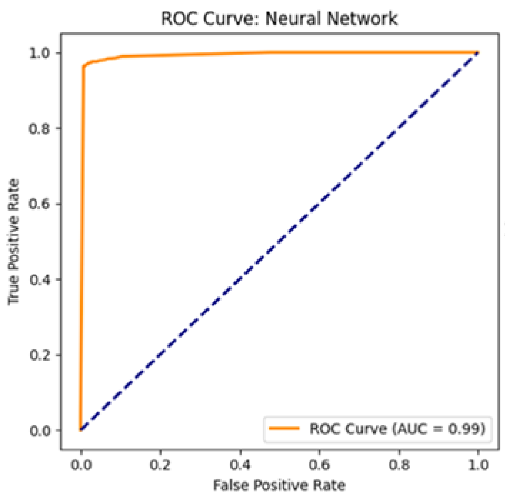


Figure 5 ROC Curve for LightGBM obtained on dataset-1 Figure 6 ROC Curve for XGBoost obtained on dataset-1



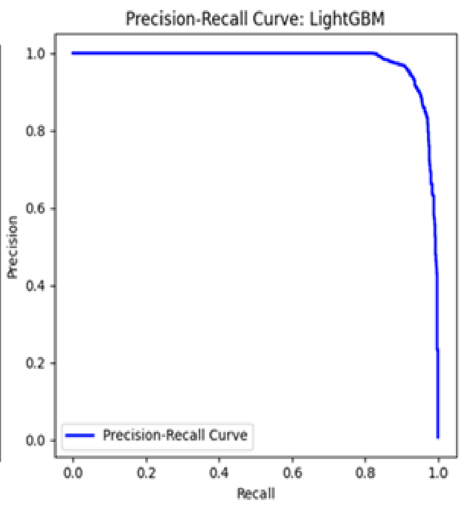
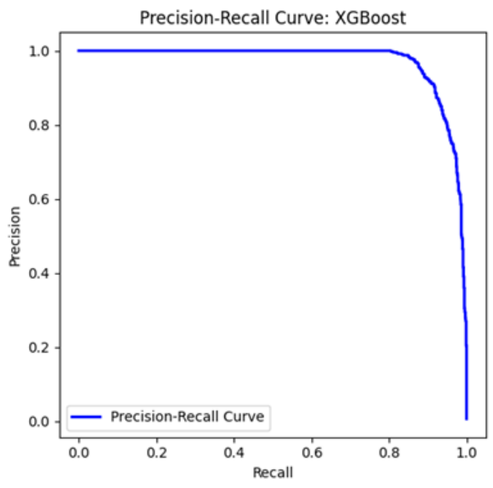


Figure 7 ROC Curve for MLP obtained on dataset-1 Figure 8 PR Curve for LightGBM obtained on dataset-1



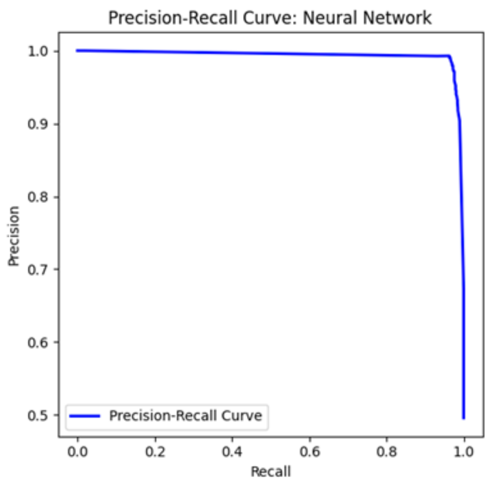


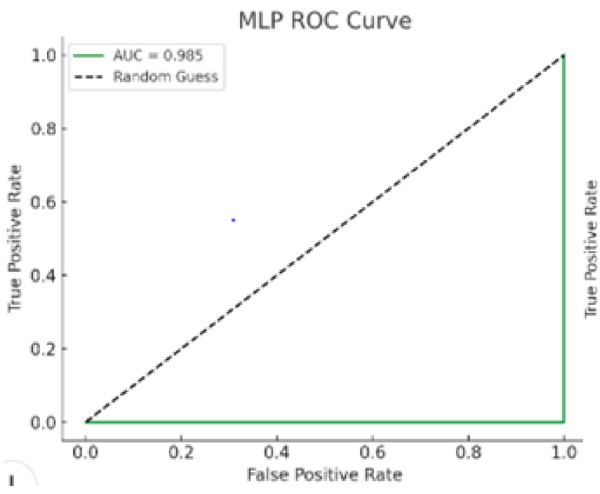
Figure 9 PR Curve for XGBoost obtained on dataset-1 Figure 10 PR Curve for MLP obtained on dataset-1

The results obtained from another dataset are as follows:

Table 3 Analysis of result metrics obtained from dataset-2

Algorithm	**Accuracy on dataset-2**	**Average precision on dataset-2**	**AUC on dataset-2**
MLP	0.95419537392812984	0.9604904268272598	0.985381923759281492
XGBoost	0.9717655673544217	0.9820222559907167	0.9941351519590189
LightGBM	0.9675136996460341	0.9764164348214088	0.9917974152941553

ROC curve and PR curve for the results obtained are as follows:



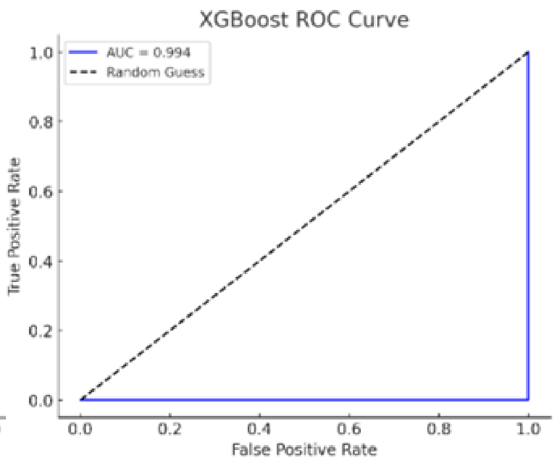
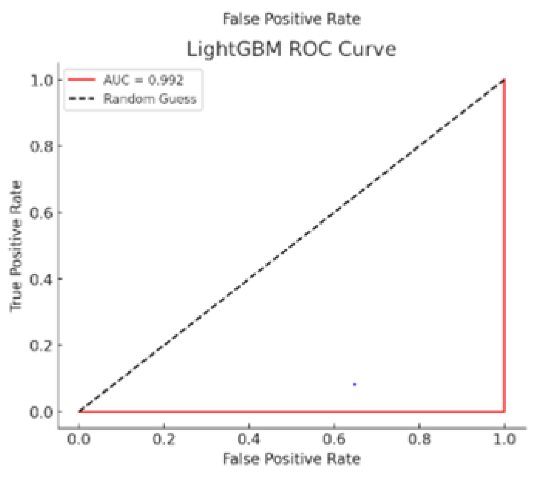


Figure 11 ROC Curve for MLP obtained on dataset-2 Figure 12 ROC Curve for XGBoost obtained on dataset-2



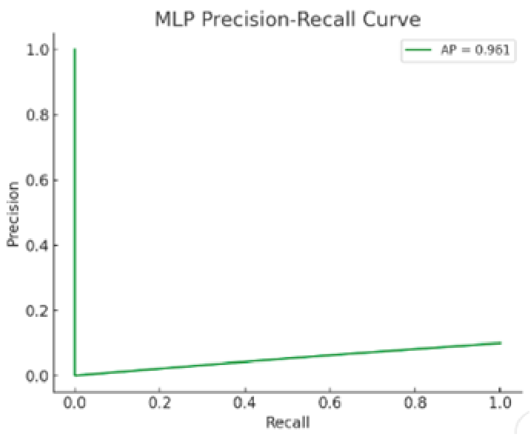
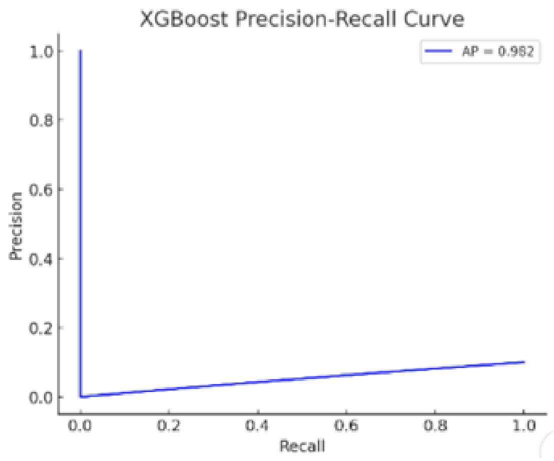


Figure 13 ROC Curve for LightGBM obtained on dataset-2   Figure 14 PR Curve for MLP obtained on dataset-2



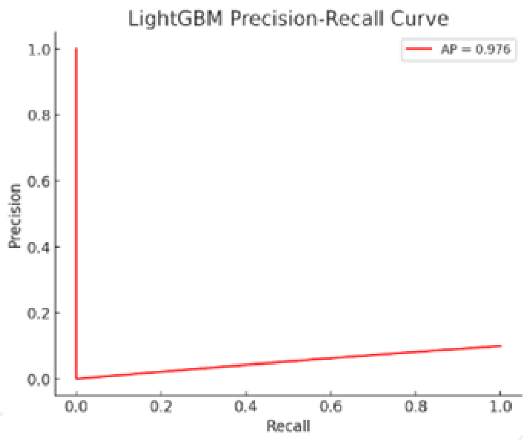


Figure 15 PR Curve for XGBoost obtained on dataset-2   Figure 16 PR Curve for LightGBM obtained on dataset-2

As evident in the generated results, the dataset is highly imbalanced, necessitating a comprehensive analysis by studying all transaction characteristics thoroughly. The nature of a transaction being fraudulent or non-fraudulent depends on multiple factors, making it crucial for the algorithm to delve deeper into the insights derived from the data to make predictions.

DISCUSSION

Dynamic model implementation: After training the model on diverse data, when real-time data is input, it will identify suspicious activity and contribute to taking necessary preventive actions if any suspicion is detected. Adaptation to unbalanced data: Given that the number of fraudulent transactions in real-time is significantly lower compared to non-fraudulent ones, the system is already trained on such data properties where the data is highly unbalanced, containing a very small number of fraudulent transactions relative to non-fraudulent ones.

REFRENCES

[1] L.SaiSampanPatrudu, D.Noorisha, E.Sidharatha, V.SaiRagaSudha, DrK.V.Satyanarayana. DynamicFraud Detection in UPI Transactions. Industrial Engineering Journal, 53(04). (2024).

[2] Manav Mangukiya, Meet Savani, Anuj Vaghani, Aryan Khunt. Financial Fraud Detection Approaches Using Machine Learning. Gujarat Technological University, PAPER ID: PCP386. (2023).

[3] Krishnan Chari. Fraud Risk in a Digitized Fintech ecosystem Troubling trends, issues and approaches to mitigate Fraud Risk. (2020).

[4] Harshith Kumar S, Dr. H.R. Divakar. UPI Fraud Detection Using Machine Learning. International Research Journal of Modernization in Engineering Technology and Science, 06(08). (2024). https://www.doi.org/10.56726/IRJMETS60849

[5] Ashish Joy, Dr Rejikumar, Dr Dhanya M. Payment App's Revamping, Perspectives from Text Mining Analysis. SIMSARC 2018. (2019). https://doi.org/10.4108%2Feai.18-12-2018.2285156

[6] Dr. S. Jamuna, Dr. J.R. Gaur, Anshu Singh, Dharam Barot. A REVIEW RESEARCH ON ONLINE FINANCIAL FRAUDS IN INDIA. The American Journal of Management and Economics Innovations, 05(01), (2023). 1-7

[7] J. Kavitha, G. Indira, A. Anil kumar, A. Shrinita, D. Bappan. Fraud Detection In UPI Transactions Using ML. *EPRA International Journal of Research and Development (IJRD), 09(04). (2024). https://doi.org/10.36713/epra16459

[8] Mr. R. Ramakrishnan, S. Vanisri, D. Yuvalakshmi. Unified Payment Interface Seamless Transaction Using RNN Model. INTERNATIONAL JOURNAL OF PROGRESSIVE RESEARCH IN ENGINEERING MANAGEMENT AND SCIENCE (IJPREMS) , 04(05), (2024). 1279-1283. https://www.doi.org/10.58257/IJPREMS34325

[9] Sayalee S. Bodade, P.P. Pawade. Review Paper on UPI Fraud Detection Using Machine Learning. International Journal for Research in Applied Science & Engineering Technology (IJRASET), 11(12). (2023). https://doi.org/10.22214/ijraset.2023.57551

[10] Yarramreddy Chandrasena Reddy, Polavarapu Nagendra Babu, Venkata Sai Pavan Ravipati, Velpula Chaitanya. UPI Fraud Detection Using Convolutional Neural Networks (CNN). Research Square. (2024). https://doi.org/10.21203/rs.3.rs-4088962/v1

[11] M. ValavanAnita B. Desai, Dr. Ravindra Deshmukh and S. Rita. Predictive-Analysis-based Machine Learning Model for Fraud Detection with Boosting Classifiers. (2022). 10.32604/csse.2023.026508