International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 06, ACSCON - 2026

Financial Fraud Detection Using Artificial Intelligence and Machine Learning

1st Dr. Divya S
Assistant Professor
Dept. of CSE
ACS College of Engineering
Bangalore, India

2nd Ajay M
Dept. of CSE
ACS College of Engineering
Bangalore, India
ajaylishanth@gmail.com

3rd Aisiri N
Dept. of CSE
ACS College of Engineering
Bangalore, India
aisirinagaraja@gmail.com

4th Bharath Gowda N
Dept. of CSE
ACS College of Engineering
Bangalore, India
bharathgowda05@gmail.com

5th Disha Gopal
Dept. of CSE
ACS College of Engineering
Bangalore, India
dishagopal08@gmail.com

***Abstract*—Spotting money scams now uses smart tech instead of old methods. Digital payments flood in fast, making mistakes harder to catch by hand. Rules written years ago fail when scammers change tactics overnight. False alarms pile up, slowing down real purchases without reason. Learning machines adapt as new tricks emerge across transaction trails. Patterns form through repeated behavior, not preset conditions coded once. The model watches how people spend, then flags odd shifts quietly. No human writes each step it follows; it relies purely on guidance at the start. It grows sharper over time, fed fresh examples from live networks, continuously improving its ability to catch complex attempts. Silent changes in timing, amount, or location raise quiet alerts behind the scenes. A close look at how data moves through this system shows careful cleaning steps come first. After that, useful patterns get pulled out so models can learn better. Instead of using one method alone, several work together-like Logistic Regression, Decision Trees, Naive Bayes, and Support Vector Machines. Rather than relying on just one viewpoint, the setup gathers opinions from all four models. It uses a strict majority voting rule to make the final call, ensuring mistakes from any single model get cancelled out. What stands out is the way rare cases are handled when spotting fraud. To fix uneven groups in the numbers, extra examples are added using SMOTE. That shift helps stop skewed results before they start. Once balanced, the tools sort transactions fast enough to keep up with live activity. Running on a web platform means responses happen right away. Tests show combining methods lifts performance high, especially through the ensemble voting approach. Mistakes drop sharply, meaning fewer false alarms pop up later. Less time gets wasted checking things by hand because the model handles more correctly. Accuracy climbs without slowing down operations overall.**

***Keywords*—Financial Fraud Detection, Machine Learning, Artificial Intelligence, Imbalanced Datasets, SMOTE, SVM, Naive Bayes, Majority Voting, Ensemble Learning**

I. INTRODUCTION

As more people move away from cash, how money changes hands has shifted fast. Because online shopping grows so quickly alongside apps that handle payments without banks digital trades happen faster and in greater numbers than ever before. Here's the problem, though: this tech boom quietly opened doors for criminals looking to profit. Crimes like fake credit cards, stolen identities, or moving money without permission now occur more often, and their methods grow sharper each time. Banks and customers face serious losses because of it not just money, but trust too.

In the past, finance teams fought back using strict systems built on fixed rules. Older setups rely on fixed rules, like spotting big buys, odd login spots, or too many failed logins one after another. Because past methods set a starting point for safety, they still show up everywhere today. Only responding when harm is already visible defines their core flaw. A new trick slips through until someone notices it, writes it down, then builds a guard against it. That delay means unknown attacks slip under the radar easily. Clever scams pretending to be normal activity go unnoticed by design.

When payments grow fast and furious, old logic struggles to keep pace. Too many warnings pop up where none should. Useless alerts pile high because the system lacks smart filtering. Blind spots stay open long after risks evolve. A real customer facing a wrongly stopped payment often feels frustrated, quickly losing faith in the service. Because of this, companies hire large groups of people to check alerts by hand a costly move that grows harder to justify over time.

Now, smart systems built on machine learning offer a different path forward. Instead of fixed rules like" if this, then that," these models learn naturally from years of past activity, spotting complex links across behaviors. They shift as new risks appear, staying alert without constant oversight. Watching many tiny signals at once, they spot odd shifts from normal habits almost instantly.

This work builds a live system to catch money scams

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 06, ACSCON - 2026

as they happen. It focuses on creating smart tools that fit right into current banking websites, spotting shady deals the moment they appear. Instead of relying on a single algorithm, it tests multiple cutting-edge models—specifically Logistic Regression, Naive Bayes, Decision Trees, and Support Vector Machines (SVM). Crucially, the system aggregates the outputs of these four algorithms using a Majority Voting mechanism to ensure maximum reliability under pressure. Challenges like rare fraud cases in data piles and fast processing demands are tackled head-on. Through clever cleanup steps paired with strong ensemble learning tricks, the approach stays sharp and ready. Designed for today's messy online cash world, it offers a scalable, intelligent, and highly automated approach to combating financial fraud.

II. PROBLEM DEFINITION

Money scams keep shifting shapes, pushing old security walls to their limits. Checking payments by hand just doesn't work anymore; it takes too long and people naturally make mistakes. Those errors mean either a real scam slips through, or a normal customer gets blocked by accident. Old setups cannot adapt naturally when fraudsters invent new tricks, causing dangerous delays in security updates.

Furthermore, building and keeping up advanced fraud systems costs a massive amount of money. Putting the tech together requires heavy investments in both software and experts who know how to run it. This high price tag leaves smaller banks stuck behind, unable to protect themselves properly.

Privacy also takes a hit. Spotting fraud requires digging through highly sensitive personal data. If that data gets exposed during the checking process, the fallout is massive. Systems running live checks face constant threats from cyber-attacks, like hackers sneaking malicious code into web forms to steal database info. Because fraud cases make up less than one percent of total payments, the data is also extremely lopsided. This imbalance tricks standard computer programs into thinking everything is perfectly safe, totally missing the rare but dangerous scams. The core problem this project solves is building a fast, cost-effective, and highly accurate AI/ML system that automatically handles this lopsided data and verifies predictions across multiple algorithms without exposing user privacy.

III. LITERATURE SURVEY

Looking back at past studies shows how fraud detection grew slowly over time. From basic number crunching, methods shifted toward complex systems built on combined models. Recent papers point out machine learning can do impressive work spotting scams. Yet gaps remain, pushing researchers to keep refining their tools. Progress happens, though problems still pop up now and then.

Early explorations into ML-based fraud detection primarily focused on benchmarking standalone, foundational algorithms. Research conducted by Elhusseny et al. provided a systematic evaluation of models such as Support Vector Machines (SVM), Naive Bayes, and Logistic Regression, establishing that dynamic learning systems significantly outperform traditional statistical rules by adapting to new data without requiring manual reprogramming [1]. Building upon this foundation, Sulaiman, Schetinin, and Sant expanded the scope to evaluate the operational challenges of deploying these systems, specifically emphasizing the difficulty of maintaining user privacy during model training and the severe impact of class imbalance within financial datasets [2].

The issue of imbalanced data where legitimate transactions outnumber fraudulent ones by extreme margins (often greater than 99 to 1) has become a focal point in recent studies. If left unaddressed, models inherently develop a majoritarian bias, achieving superficial accuracy by blindly approving all transactions. Mahajan, Baghel, and Jayaraman demonstrated that applying rigorous over-sampling and under-sampling techniques prior to training allows even linear models, such as Logistic Regression, to achieve highly reliable binary classification outcomes [4].

To maximize predictive reliability, methodologies have continuously refined ensemble models. Studies have consistently proven that combining distinct algorithms—such as geometric models (SVM), probabilistic models (Naive Bayes), and linear models (Logistic Regression)—creates a highly resilient defense against complex fraud patterns, neutralizing the blind spots of any single algorithm.

This literature review underscores the necessity of a holistic, multi-layered approach. An effective modern system must synergize rigorous data sanitization, synthetic balancing techniques, and a collaborative voting architecture to securely navigate the digital financial landscape.

IV. SYSTEM REQUIREMENTS AND SCOPE

Handling big loads of data flows smoothly through the system, while heavy math runs nonstop when models train, yet responses stay quick once live via tight API paths. Python 3.x powers the setup, chosen so number crunching happens fast without delays creeping in. Pandas shapes raw information into usable form inside the preprocessing stage, working alongside NumPy to manage calculations precisely. Scikit- learn steps in to train the individual classifiers and set up the majority voting logic, also double-checking how well each model actually performs before anything moves forward. Handling uneven datasets happens through the Imbalanced- learn tool. Built on Flask, the backend processes API calls without issue. Meanwhile, user details, activity records, and evaluation outputs get stored safely in MySQL.

V. METHODOLOGY

A careful step-by-step process defines how data moves through the system. Financial inputs often arrive messy-full of gaps, inconsistencies, and clutter. Shaping that chaos into something useful demands precision from start to finish. What sets this approach apart isn't flash-it's structure.

A. Data Collection and Sanitization

Before math models come into play, data needs cleaning first. Through careful scanning, the preprocessing step pulls out records with missing parts. To keep things balanced, gaps

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 06, ACSCON - 2026

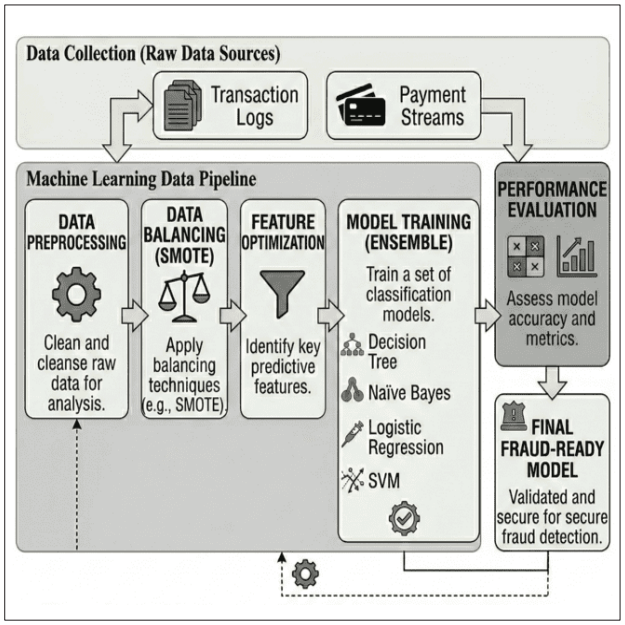

Fig. 1. The end-to-end Machine Learning data pipeline, highlighting preprocessing, SMOTE balancing, and parallel model training.

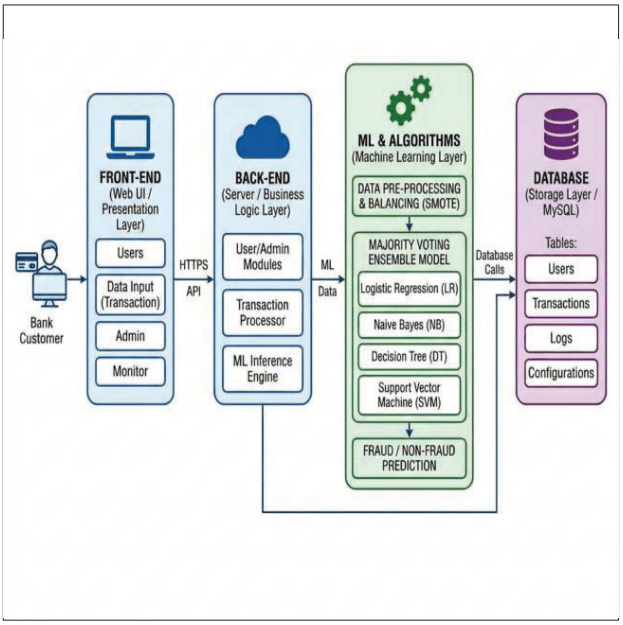

Fig. 2. Web application architecture diagram detailing the interaction between the frontend, Flask backend, and the ML voting engine.

in info get filled using smart guesses based on averages. Identical entries that repeat? They're quietly removed so no single event skews results too much.

B. Normalization and Feature Scaling

When numbers in money data stretch across wild ranges- say, payments from one dollar up to ten thousand, alongside time gaps hitting eighty-six thousand seconds- scaling smooths things out. A tool called Standard Scaler adjusts these figures so they average zero with a spread near one. Without this tweak, math steps using straight-line distances (which is particularly vital for SVM algorithms) could tilt hard toward bigger numbers, warping outcomes silently.

C. Addressing Class Imbalance via SMOTE

Think fraud detection: shady deals make up almost nothing in live systems. To even the field, SMOTE jumps in, artificially growing rare cases by smart interpolation instead of leaving them drowned. SMOTE does not just copy fraud cases. Through k-nearest neighbor logic, it studies how those cases appear across different traits. Following that, it builds fresh fake examples by drawing lines between known fraud samples. These made-up entries increase the number of fraud markers during training. As a result, all four models learn patterns more evenly.

VI. SYSTEM ARCHITECTURE

Now shifting into structure turning abstract ideas into working code demands solid planning. What comes next is a design built in layers, split into clear parts. The system outlined here works as a three-level online tool with separate modules. Each part handles one role.

A. Presentation Layer (Frontend)

Up front, things run on familiar web tools so they work just about anywhere. A clean entry point lets users start payments safely while checking past records too. Admins get a responsive control panel showing real-time money movements instead. That space breaks down unusual events into clear summaries, displaying the individual predictions of all four models alongside the final ensemble verdict.

B. Logic Layer (Backend Server)

Running behind it all, the server acts like the core rhythm of the whole setup. Python's Flask shapes this middle section with lightweight precision. Once someone sends a deal through the surface layer, info moves quietly underneath. REST-style links carry those details without exposing anything risky. Right away, the Flask server grabs the incoming data, setting off a chain of cleanup steps before shoving it toward the four locked-down machine learning models. The server tabulates the votes and shoots the final answer back to the user screen without delay.

C. Data Persistence Layer (Database)

Tucked beneath everything sits a steady MySQL database holding every piece together. Every move gets recorded forever in the Transaction Schema time stamps, dollar amounts, shop identifiers, all etched into place. Suspicious hits land in the Fraud Schema instead, isolated like evidence in a vault, ready for review or model updates later.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 06, ACSCON - 2026

VII. MACHINE LEARNING ALGORITHMS AND ENSEMBLE LOGIC

At the core, smart math does the deciding. Different classifiers chew through layers of info in their own ways. One after another, they sort normal from risky using sharp number logic. The study rigorously evaluates Logistic Regression, Naive Bayes, Decision Trees, and Support Vector Machines, capping them off with a robust voting mechanism.

A. Logistic Regression

One way to tackle yes-or-no choices in data work starts with an old but useful method. This approach guesses how likely it is that a specific activity counts as fraud. Instead of allowing numbers to run wild, it traps them between zero and one through a special S-shaped curve. That shift happens by pushing weighted inputs through a formula shaped like this: chance equals one divided by one plus e raised to the opposite of beta-zero plus sums of features times their weights.

The mathematical formulation is expressed as:

$$P(Y = 1) = \frac{1}{1 + e^{-(\beta_0 + \sum_{i=1}^{n} \beta_i X_i)}} \quad (1)$$

The chance of a fake transaction shows up as $P(Y = 1)$, with β_0 setting the baseline and each β_i adjusting for traits in the data called X_i. Once the model learns from the past, it draws a line right at the 0.5 mark.

B. Naive Bayes

The Naive Bayes classifier relies on Bayes' Theorem, operating on the "naive" assumption of conditional independence between every pair of features given the value of the class variable. Despite this oversimplified assumption, it performs remarkably well in complex real-world situations like fraud detection due to its rapid computation.

Given a transaction feature vector $X = (x_1, x_2, ..., x_n)$ and a class label y (fraud or not fraud), Bayes' theorem states:

$$P(y|X) = \frac{P(X|y)P(y)}{P(X)} \quad (2)$$

Assuming feature independence, this simplifies to:

$$P(y|X) \propto P(y) \prod_{i=1}^{} P(x_i|y) \quad (3)$$

The algorithm calculates the probability of both classes and assigns the transaction a vote based on the highest probability.

C. Decision Trees

Because they map choices step by step, Decision Trees sort information using splits that best separate outcomes. One split led to another when a trait cuts the group most sharply- this measure is known as Information Gain. At every branch point, a question forms around one attribute; answers guide flow until reaching an endpoint. That last node hands out the verdict: fraud or not.

D. Support Vector Machines (SVM)

Support Vector Machines operate on the principle of finding the optimal geometric hyperplane that cleanly separates the fraud class from the legitimate class in a high-dimensional space. The objective is to maximize the mathematical margin between the closest data points of the two classes.

Given a feature vector x_i and a weight vector w, the hyperplane is defined as:

$$w^T x_i + b = 0 \quad (4)$$

Because financial fraud patterns are rarely linearly separable, SVM utilizes the Radial Basis Function (RBF) to project the data into higher dimensions where a linear separator can be found, minimizing misclassifications while casting its vote.

E. Ensemble Prediction (Majority Voting)

To drastically reduce the false positive rates inherent to standalone classifiers, the system implements a Hard Majority Voting ensemble. When a live transaction is processed, it is concurrently evaluated by the four aforementioned algorithms. Each model casts a discrete binary vote: Safe (0) or Fraud (1).

The final ensemble prediction P_{final} is determined mathematically by calculating the statistical mode of the individual
predictions:

$$P_{final} = \text{mode}(P_{LR}, P_{NB}, P_{DT}, P_{SVM}) \quad (5)$$

If the majority of the models agree that a transaction is anomalous, the system triggers a definitive Fraud alert. This collab- orative cross-verification guarantees that isolated algorithmic errors are overridden by the consensus, ensuring unparalleled real-time accuracy.

VIII. IMPLEMENTATION AND TESTING

Deploying an unverified system poses an unacceptable risk. The FFDS was subjected to a rigorous testing matrix to ensure absolute operational reliability.

Unit tests independently validated the mathematical accuracy of isolated scripts verifying that the SMOTE generation executed without corrupting class boundaries. Integration test

focused on the connectivity tissues of the architecture.

Developers simulated JSON data payloads originating from the frontend to verify that the Flask API accurately parsed the requests, fed them to the four serialized models, and correctly computed the Majority Vote. Furthermore, Load Testing evaluated the system's viability for enterprise-level deployment, confirming that the Python inference engine could sustain low-latency responses.

IX. RESULTS

The empirical results generated during the evaluation phase unequivocally validate the efficacy of the proposed machine learning architecture. Because financial datasets are imbalanced, relying solely on standard Accuracy is deeply mis-leading. Therefore, the system was evaluated using a comprehensive scorecard comprising Accuracy, Precision, Recall, F1- Score, and the Receiver Operating Characteristic Area Under the Curve (ROC-AUC).

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 06, ACSCON - 2026

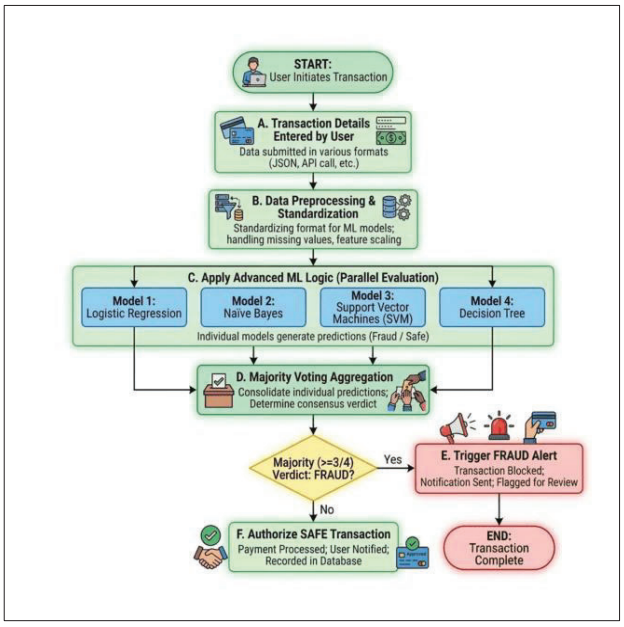

Fig. 3. Process flow diagram illustrating the real-time transaction evaluation and the Majority Voting alert mechanism.

A. Understanding the Evaluation Metrics

Before reviewing the numbers, it helps to understand what they actually measure:

- **Accuracy:** Looks at the total amount of right guesses over everything. High accuracy looks good, but it hides problems when fraud is rare.
- **Sensitivity (Recall):** Measures how many actual fraud cases the system caught. High recall means very few scammers slipped through the net undetected.
- **Specificity:** Checks how well the system identifies normal behavior. This stops real customers from getting blocked by false alarms.
- **F1-Score:** Finds the middle ground between catching fraud and annoying users. It balances out the skewed numbers beautifully.

B. Algorithm Comparative Analysis

The experimental results highlighted significant performance variances across the tested models, which are detailed in Table I.

TABLE I
COMPARISON OF ACCURACY, F1-SCORE, AND ROC ACROSS MODELS

S.No	Method	Accuracy	F1-Score	ROC
1	**Majority Voting Ensemble**	**0.9996**	**0.8410**	**0.9850**
2	Support Vector Machine	0.9994	0.8228	0.9785
3	Logistic Regression	0.9989	0.6419	0.9776
4	Naive Bayes	0.9988	0.6350	0.9600
5	Decision Tree	0.9987	0.6153	0.9217

The Decision Tree established a solid baseline but struggled with its F1-Score (0.6153), indicating a struggle to perfectly balance precision and recall. Naive Bayes provided rapid computations and a strong ROC of 0.9600. Logistic Regression, bolstered by the SMOTE data balancing, performed admirably for a linear model. Support Vector Machines improved upon the standalone algorithms by successfully plotting high-dimensional boundaries.

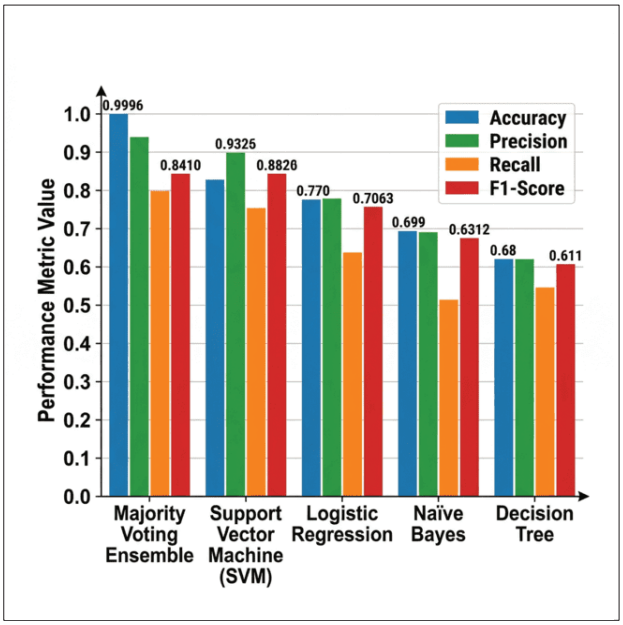

Fig. 4. Comparative analysis of Accuracy, Precision, Recall, and F1-Scores across the deployed machine learning models.

Ultimately, the **Majority Voting Ensemble** achieved absolute supremacy across all metrics. By aggregating the strengths of the RBF kernel (SVM), probabilistic mapping (NB), linear coefficients (LR), and hierarchical splits (DT), the Ensemble recorded near-perfect Accuracy (0.9996) and a dominant ROC (0.9850).

The Majority Voting mechanism demonstrated an unparalleled capability to filter out false positives while capturing the most subtle, deeply hidden correlations that signify modern cyber-fraud, establishing itself as the most reliable classification engine for this architecture.

X. CONCLUSIONS

The Financial fraud detection system gives a solid way to catch and stop scams in money matters. Spot-on scam spotting ensures that sneaky activity gets caught fast, so cash stays safe while folks feel more secure using it. Instead of guesswork, smart tech crunches clean numbers to spot weird behavior that stands out from the norm. Because every deal gets scanned live, warnings pop up if something looks off so security teams can jump in fast, meaning way less damage overall.

The integration of advanced preprocessing pipelines, particularly the application of SMOTE to neutralize data imbalance, proved fundamental to the system's success. By transitioning from static, rule-based paradigms to a dynamic, multi-model architecture spearheaded by a Majority Voting Ensemble,

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 06, ACSCON - 2026

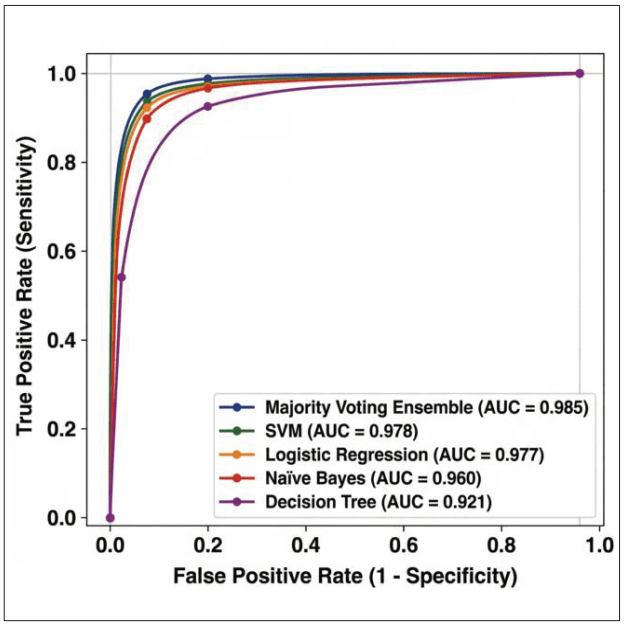

Fig. 5. Receiver Operating Characteristic (ROC) curve demonstrating the diagnostic ability of the algorithms at various threshold settings.

financial institutions can achieve a level of predictive accuracy previously considered unattainable. Thanks to its modular design, the system handles growing workloads without hiccups, evaluating four distinct models simultaneously without significant processing delays.

While the current framework exhibits stellar performance on structured transaction logs, the landscape of digital crime is perpetually evolving. Future iterations of this system will focus on expanding its analytical horizons. Upgrading the backend infrastructure to utilize distributed streaming platforms will facilitate the instantaneous ingestion and analysis of massive global data torrents. Finally, anchoring the transaction logs and immutable fraud alerts to a decentralized blockchain network presents a compelling avenue for guaranteeing absolute data integrity.

REFERENCES

[1] N. Samy Elhusseny, S. M. Ouf, and A. M. Idrees, "Credit Card Fraud Detection Using Machine Learning Techniques," Future Computing and Informatics Journal, 2022.
[2] R. B. Sulaiman, V. Schetinin, and P. Sant, "Review of Machine Learning Approach on Credit Card Fraud Detection," Human-Centric Intelligent Systems, 2022.
[3] J. Brownlee, "Imbalanced Classification with Python: Better Metrics, Balance Techniques, and Model Performance," Machine Learning Mastery, 2020.
[4] A. Mahajan, V. S. Baghel, and R. Jayaraman, "Credit Card Fraud Detection Using Logistic Regression with Imbalanced Dataset," IEEE Conference Proceedings, 2023.
[5] R. Raut, E. Govardhan, and A. Chandanshive, "Credit Card Fraud Detection Using Ensemble Modeling," 2023.
[6] F. Pedregosa et al., "Scikit-learn: Machine Learning in Python," Journal of Machine Learning Research, vol. 12, pp. 2825-2830, 2011.
[7] E. Ileberi, Y. Sun, and Z. Wang, "A Machine Learning-Based Credit Card Fraud Detection Using GA for Feature Selection," Journal of Big Data, 2022.
[8] G. Lemaitre, F. Nogueira, and C.K. Aridas, "Imbalanced-learn: A Python Toolbox to Tackle the Curse of Imbalanced Datasets in Machine Learning," Journal of Machine Learning Research, vol. 18, no. 17. pp. 1-5, 2017.
[9] J. Zhang et al., "Privacy-Preserving Machine Learning: A Survey of Techniques and Applications," ACM Computing Surveys, vol. 54, no. 6, pp. 1-39, 2022.