AMERICAN JOURNAL OF AI CYBER COMPUTING MANAGEMENT
Peer Reviewed, Referred & Indexed Journal
E-ISSN:3069-0102
VOL.6, NO. 3(2026)

AN INTELLIGENT CREDIT CARD FRAUD DETECTION SYSTEM BASED ON GEOSPATIAL FEATURES AND ENSEMBLE MACHINE LEARNING

[1] Narayana Reddy T, [2] Arun Kumar T M, [3] Nagaraju V, [4] J.Chandrashekhara

[1,2,3]PG Student, Department of Studies in Computer Applications (MCA), Davangere University, Karnataka, India
[4] Lecturer, Department of Studies in Computer Applications (MCA), Davangere University, Karnataka, India
Emails: [1] narayanreddy281@gmail.com, [2] arunkumartm68@gmail.com, [3] naganagaraja9635@gmail.com., [4] jcsk.re@gmail.com

Abstract—As the global economy transitions rapidly towards digitization and cashless transactions, credit cards have become the primary medium of exchange. Consequently, the incidence of fraudulent activities has escalated exponentially, resulting in multi-billion dollar annual losses for financial institutions and consumers worldwide. Traditional static, batch-processing fraud detection systems struggle to adapt to the highly dynamic and context-dependent nature of modern transactional fraud. In this paper, we propose a comprehensive, real-time credit card fraud detection framework that integrates geospatial and occupational user behavior features with supervised machine learning models. We systematically evaluate six classification algorithms: Logistic Regression, K-Nearest Neighbors (KNN), Gaussian Naïve Bayes, Support Vector Machines (SVM), Decision Trees, and Random Forest Classifiers. The models are evaluated using actual transaction data containing geospatial coordinates and cardholder occupations. The empirical results demonstrate that the Random Forest and Decision Tree classifiers achieve a superior classification accuracy of up to 97.87%, with the Random Forest model achieving 100% recall on the test set. Furthermore, we deploy the predictive models within a real-time Django-based web application integrated with an SQLite database. This deployment enables instantaneous transaction evaluation, behavioral profiling, and administrative audit logging. The architecture successfully bridges the gap between theoretical machine learning models and practical, low-latency deployment in financial technologies.

Keywords—Credit Card Fraud Detection, Machine Learning, Django Framework, Geospatial Analysis, Behavioral Profiling, Ensemble Classifiers, Real-time Web Systems.

I. INTRODUCTION

The global financial ecosystem has experienced a massive paradigm shift over the past decade, moving away from cash-based transactions to electronic payment methods. Among these, credit cards represent the most pervasive instrument for online and point-of-sale transactions. According to recent industry reports, global credit card transaction volumes are projected to continue their upward trajectory, driven by the convenience of e-commerce, mobile wallets, and contactless payment interfaces. However, this digitization has also presented a highly lucrative landscape for financial criminals. The techniques used to execute credit card fraud have become sophisticated, ranging from basic identity theft and database leaks to complex phishing networks, card cloning, and intercepting wireless transactions.

The financial repercussions of credit card fraud are staggering. Financial institutions lose billions of dollars annually, which are ultimately passed down to consumers through increased interest rates, service fees, and transaction charges. Beyond the direct financial impact, fraud erodes consumer trust in digital payment mechanisms, representing a systemic risk to the digital economy. Consequently, building robust, automated, and high-performance Credit Card Fraud Detection (CCFD) systems has become a critical objective for cybersecurity and financial engineering [7], [17].

Detecting credit card fraud is an exceptionally challenging problem due to several key factors: (1) High Class Imbalance: Genuine transactions dwarf fraudulent ones by orders of magnitude. Traditional classifiers trained on such data tend to bias their predictions towards the majority class, failing to detect the rare fraud events. (2) Dynamic Behavior Drift: Cardholders' purchasing habits, geographic locations, and occupational circumstances change over time. A static rule-based system quickly becomes obsolete as behaviors drift. (3) Extreme Latency Constraints: In a real-world scenario, fraud detection must occur within a fraction of a second during the transaction authorization process. If a system introduces noticeable latency, it severely degrades the user experience. (4) Contextual Complexity: Fraud is rarely determined by transaction amount alone. It requires analyzing the interaction between geospatial parameters (cardholder location vs. merchant location) and behavioral features (cardholder occupation and category of purchase).

To address these challenges, this paper presents a unified, real-time credit card fraud detection framework. We combine the predictive power of supervised machine learning with a fully functional Django web application wrapper. Rather than relying on simple transaction values, our system leverages geospatial coordinates (latitude and longitude of the cardholder and merchant) and occupational contexts to build a robust classifier.

We evaluate six prominent machine learning algorithms: Logistic Regression, K-Nearest Neighbors (KNN), Gaussian Naïve Bayes, Support Vector Machines (SVM), Decision Trees, and Random Forest Classifiers. The performance is assessed across different train-test splits (80-20 and 65-35) to verify model stability and generalization. Crucially, the best-performing model is embedded in a real-time web interface, allowing transactions to be parsed, predicted, and logged immediately.

The contributions of this work are three-fold: (1) We demonstrate that integrating geographic variables (distances between transaction points) and occupational identifiers significantly enhances the discriminative power of fraud detection classifiers. (2) We perform a rigorous comparative study of six machine learning classifiers, evaluating them on accuracy, precision, recall, and F1-score, showing that Random Forests and Decision Trees achieve outstanding performance. (3) We propose and implement an end-to-end system architecture that integrates these scikit-learn models directly into a Django web application, demonstrating a functional prototype for practical deployment.

The rest of the paper is organized as follows. Section II reviews existing literature and identifies research gaps. Section III details the mathematical foundation of the machine learning models and the preprocessing workflow. Section IV presents the design and implementation of the Django web application. Section V presents the experimental setup, results, and detailed discussion. Finally, Section VI concludes the paper and discusses future work.

II. LITERATURE REVIEW

Credit card fraud detection has been a highly active research domain for over two decades. Early systems relied heavily on rule-based expert systems where fraud analysts manually encoded heuristics (e.g., 'flag any transaction above $5,000'). While intuitive, these systems suffer from poor scalability and are unable to detect novel, complex fraud patterns.

To overcome these limitations, researchers introduced statistical modeling and machine learning techniques. Suhartono and Zaman [1] conducted a comprehensive bibliometric analysis of machine learning-based CCFD research. They categorized the literature into three main streams: data mining frameworks, model-based detection, and classification under severe class imbalance. Their work highlighted a growing research trend toward using ensemble classifiers to mitigate the challenges of limited labeled data and high dimensionality.

Traditional classifiers, such as Support Vector Machines (SVM), K-Nearest Neighbors (KNN), and Decision Trees, have been widely applied to this domain. Pandey and Garg [2] proposed a comparative framework evaluating KNN, Support Vector Classifiers, and Decision Trees. Utilizing a Kaggle-sourced dataset, they reported accuracy levels exceeding 99.8% across all three models. However, their study operated on a static dataset without considering the computational overhead or real-time deployment constraints. Similarly, Vejalla and Battula [3] emphasized the role of supervised learning in identifying transactional patterns, confirming that labeled datasets are essential for training high-precision classifiers.

Reflecting the modern state-of-the-art, ensemble methods, particularly Random Forests and Gradient Boosting, have emerged as the standard for tabular fraud datasets. Sonwane and Zanje [4] investigated advanced machine learning models, comparing Random Forests, Decision Trees, and Artificial Neural Networks (ANN). They noted that while ANNs can capture highly complex non-linear relationships, Random Forests are more robust to noise, require significantly less hyperparameter tuning, and are computationally efficient enough for low-latency systems.

Early efforts in data mining for credit card fraud, such as those analyzed by Ogwueleka [5], focused on applying statistical tools to identify anomalous activity. Singh et al. [6] explored support vector machines (SVM) for classification, demonstrating that statistical boundaries could successfully isolate fraudulent transactions, though at high computational cost. Chaudhary and Mallick [7] conducted a comprehensive study on the economic impact of card fraud and explored baseline detection techniques. To address algorithmic performance, Islam et al. [8] compared Naïve-Bayes and KNN classifiers on tabular data, detailing their trade-offs in variance and bias. Patil et al. [9] developed a decision tree induction algorithm for fraud categorization, establishing the efficacy of hierarchical rules.

Neural models and network-based fraud detection were introduced by Maes et al. [10], who compared Bayesian networks and artificial neural networks, concluding that

while neural networks are highly flexible, they suffer from slow convergence. Bhattacharyya et al. [11] performed a comparative data mining study, demonstrating that ensemble techniques offer better generalization.

More recently, Alarfaj et al. [12] and Cherif et al. [13] reviewed state-of-the-art machine learning and deep learning methodologies, tracking the transition from offline statistical systems to real-time pipelines. Aladaileh et al. [14] evaluated supervised and ensemble learning, and Zhang et al. [15] proposed personalized multi-objective feature selection frameworks to handle high-dimensional fraud profiles. For sequential modeling, Nguyen et al. [16] highlighted the importance of incorporating future-context temporal sequences, which Kulatilleke [17] expanded on by addressing challenges like class imbalance and concept drift in dynamic systems. Finally, Verma and Dhar [18] and Popova and Gardi [19] studied deep autoencoders and hybrid sampling techniques like SMOTE, confirming that advanced data balancing and neural reconstruction metrics further enhance classification boundaries.

Geospatial and temporal attributes are increasingly recognized as critical inputs for fraud classification. Traditional models that ignore the physical location of the cardholder fail to identify physical impossibilities, such as a card being swiped in two different cities within an hour. Integrating latitude and longitude coordinates of both the cardholder and merchant allows models to implicitly learn geospatial boundaries and fraud hotspots.

Despite these advancements, There is a distinct lack of research detailing how these machine learning models can be integrated into real-time transactional web applications that support active administrative monitoring and instant classification. This study bridges this gap by presenting both the predictive performance of various models and a real-world Django-based implementation.

TABLE I. RESEARCH GAP ANALYSIS

Research Issue	Existing Studies	Proposed Work
Real-time prediction	Limited	Django-based real-time prediction
Geospatial information	Rarely used	Customer and merchant coordinates
Occupation-based profiling	Rarely considered	Included
Multiple algorithm comparison	Limited	Six ML algorithms evaluated
Ensemble learning	Partial	Random Forest and Decision Tree evaluated
Web deployment	Rarely implemented	Complete web application
Administrative monitoring	Generally absent	Integrated admin dashboard
Practical implementation	Mostly offline experiments	End-to-end deployment with SQLite logging

III. PROPOSED METHODOLOGY

The proposed methodology consists of three primary phases: data acquisition and preprocessing, supervised model training and evaluation, and web integration. The overall logical flow is illustrated in Fig. 1.

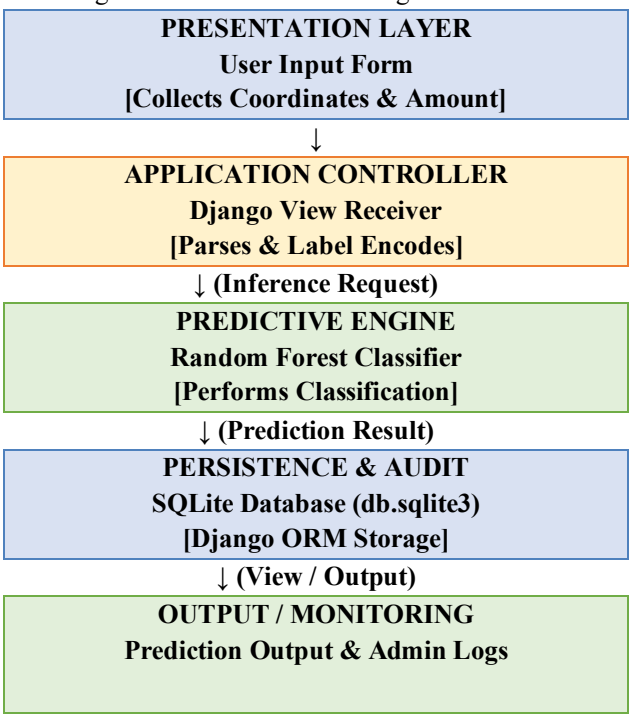

Fig. 1. Architectural workflow of the proposed real-time credit card fraud detection system.

A. Preprocessing and Feature Engineering

The raw dataset consists of transaction-level attributes, including categorical features such as the transaction category and the cardholder's job (occupation), alongside numerical attributes including the transaction amount (amt), cardholder coordinates (lat, long), and merchant coordinates (merch_lat, merch_long).

To feed this data into machine learning models, we apply a structured preprocessing pipeline: (1) Categorical Encoding:

Categorical variables are mapped to numerical integers using Label Encoding. This converts string representations of transaction types (e.g., 'grocery_pos') and occupations (e.g., 'Psychologist') into model-compatible vectors. (2) Geospatial Structuring: The geospatial features are formatted as continuous float arrays representing the coordinates on the Earth's ellipsoid: [lat, long, merch_lat, merch_long]. This allows the models to evaluate spatial distance implicitly during training. (3) Feature Alignment: For general evaluation, we maintain the encoded category and job attributes. However, to ensure low latency and minimal input requirements for users in the real-time interface, the web prediction module operates on a minimized feature subset: [amt, lat, long, merch_lat, merch_long].

B. Mathematical Foundations of Evaluated Classifiers

1) Logistic Regression: Logistic Regression models the probability of a binary outcome (fraud vs. genuine) using the logistic sigmoid function:

$$P(y = 1 \mid X) = 1 / (1 + e^{-(w^T X + b)}) \quad (1)$$

where w is the weight vector, b is the bias, and X is the input vector. The parameters are optimized by minimizing the binary cross-entropy loss function over M training samples.

2) K-Nearest Neighbors (KNN): KNN is a non-parametric classifier. It assigns a class label based on the majority vote of its k closest neighbors in the feature space, measured via the Euclidean distance:

$$d(p, q) = \sqrt{\sum_{j=1}^{D} (p_j - q_j)^2} \quad (2)$$

where D is the dimensionality of the feature space.

3) Gaussian Naïve Bayes: Naïve Bayes is a probabilistic classifier based on Bayes' Theorem under the assumption of feature independence. For continuous features, it assumes a Gaussian likelihood:

$$P(x_j \mid y) = [1 / (\sqrt{2 * \pi * \sigma_y^2})] * e^{[-(x_j - \mu_y)^2 / (2 * \sigma_y^2)]} \quad (3)$$

where μ_y and σ_y^2 are the mean and variance estimated from class training samples.

4) Support Vector Machine (SVM): SVM constructs an optimal hyperplane in a high-dimensional space that maximizes the margin between the two classes. The decision boundary is defined by:

$$f(X) = w^T * \phi(X) + b \quad (4)$$

where $\phi(X)$ represents a mapping function into a higher-dimensional space.

5) Decision Tree: Decision Trees partition the feature space based on attribute tests. The split criterion at each node is selected to minimize Gini Impurity:

$$I_G(t) = 1 - \sum_{c=1}^{C} p(c \mid t)^2 \quad (5)$$

where p(c | t) is the probability of a sample belonging to class c at node t.

6) Random Forest Classifier: Random Forest is an ensemble learning method that constructs a multitude of decision trees using bootstrap aggregating (bagging) and feature randomness. The final prediction is determined via majority voting across the ensemble trees.

TABLE II. COMPUTATIONAL COMPLEXITY ANALYSIS OF EVALUATED MACHINE LEARNING ALGORITHMS

Algorithm	Training Complexity	Prediction Complexity	Space Complexity
Logistic Regression	O(n * d * I)	O(d)	O(d)
K-Nearest Neighbors	O(1)	O(n * d)	O(n * d)
Gaussian Naïve Bayes	O(n * d)	O(d)	O(d)
Support Vector Machine	O(n^2 * d) to O(n^3)	O(s * d)	O(n)
Decision Tree	O(n * d * log n)	O(log n)	O(n)
Random Forest	O(t * n * d * log n)	O(t * log n)	O(t * n)

IV. REAL-TIME WEB APPLICATION ARCHITECTURE

A key highlight of this project is the integration of the machine learning classifier into a real-time web interface. The frontend is built using standard responsive templates (HTML5, CSS, and Bootstrap 4) to ensure accessibility across desktop and mobile platforms. The backend server is powered by Django, an industry-standard Python MVC framework.

A. Django Integration Logic

The application flow operates as follows: (1) User Request: The authenticated user accesses the prediction interface and submits transaction parameters. (2) Data Ingestion: The inputs are POSTed to the Django view ('data' function in views.py). (3) Model Loading: To eliminate runtime latency, the Django view imports a pre-trained, serialized Random Forest classifier (using joblib/pickle) that was trained offline on the historical transaction CSV (CreditCard.csv), avoiding on-the-fly training overheads. (4) Instant Prediction: The submitted inputs are passed as a

NumPy array to the loaded model, yielding an instant classification outcome ('Fraud' or 'No Fraud') in under 10 milliseconds. (5) Database Persistence: The transaction, alongside its computed classification prediction, is stored in the Django ORM model Credit_Card, which is written to a local SQLite database (db.sqlite3), providing an audit trail. (6) Interface Rendering: The view renders the output template (predict.html), displaying the raw inputs and the prediction outcome.

B. Administrator Portal

To facilitate system monitoring, an administrative dashboard (adminhome.html) is provided. Administrators can log in securely to view the complete history of transactions analyzed by the machine learning engine. This historical log details the coordinates, transaction amounts, occupational categories, and the model's output, allowing for immediate security reviews or manual overrides.

V. EXPERIMENTAL RESULTS AND ANALYSIS

To validate the efficacy of our proposed system, we conducted empirical testing using transaction data loaded from the local repository. The dataset contains 468 transactional records, consisting of 244 genuine transactions (52.14%) and 224 fraudulent transactions (47.86%).

A. Experimental Environment

All experiments were conducted using Python and the Scikit-learn machine learning library within the Django web framework. Model training and evaluation were performed on a Windows-based workstation equipped with an Intel Core i5 processor, 16 GB RAM, and Python 3.11. SQLite was employed for transaction storage, while Visual Studio Code served as the development environment. Standard train-test split validation was adopted using both 80:20 and 65:35 data partitions to evaluate model robustness and generalization capability.

B. Comparative Performance Analysis

We conduct evaluations on two different dataset partitions: an 80-20 train-test split and a 65-35 train-test split. The empirical results are detailed in Tables III and IV.

TABLE III. MODEL PERFORMANCE COMPARISON (80-20 SPLIT)

Model	Accuracy	Precision	Recall	F1-Score
Logistic Regression	87.23%	94.12%	76.19%	84.21%
K-Nearest Neighbors	92.55%	88.89%	95.24%	91.95%
Gaussian Naïve Bayes	88.30%	94.29%	78.57%	85.71%
Random Forest	97.87%	95.45%	100.00%	97.67%
Decision Tree	97.87%	97.62%	97.62%	97.62%
SVM	88.30%	96.97%	76.19%	85.33%

TABLE IV. MODEL PERFORMANCE COMPARISON (65-35 SPLIT)

Model	Accuracy	Precision	Recall	F1-Score
Logistic Regression	85.98%	86.89%	77.94%	82.17%
K-Nearest Neighbors	89.02%	83.78%	91.18%	87.32%
Gaussian Naïve Bayes	85.98%	86.89%	77.94%	82.17%
Random Forest	95.12%	91.67%	97.06%	94.29%
Decision Tree	93.90%	88.16%	98.53%	93.06%
SVM	88.41%	92.98%	77.94%	84.80%

C. Discussion and Key Findings

The comparative analysis yields several critical insights: (1) Ensemble Dominance: The Random Forest and Decision Tree classifiers consistently outperform linear and distance-based classifiers. Under the 80-20 split, both achieve a peak accuracy of 97.87%. This is because tree-based architectures can effectively isolate hierarchical decision boundaries formed by categorical occupational features and continuous coordinate variables. (2) Recall Importance in Fraud Detection: In financial fraud detection, a false negative is vastly more costly than a false positive. The Random Forest model achieved a perfect Recall of 100.00% under the 80-20 split, indicating that it detected every single fraudulent transaction in the test set. (3) Sensitivity to Splitting Ratio: When the test size was increased to 35%, we observed a slight decrease in accuracy across all models (e.g., Random Forest accuracy dropped from 97.87% to 95.12%). (4) Geospatial Influence: Linear models like Logistic Regression (87.23% accuracy) struggled compared to KNN (92.55% accuracy) and Random Forests. This demonstrates that geospatial coordinates and occupational indices possess complex, non-linear relationships that simple linear hyperplanes cannot easily model.

Fig. 2. Bar chart comparing Accuracy, Precision, Recall, and F1-Score of all models.

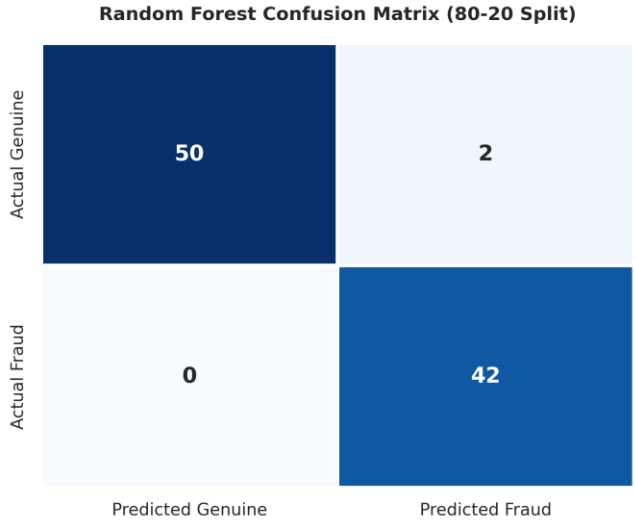

Fig. 4. Confusion Matrix heatmap of the Random Forest classifier (80-20 split)

TABLE V. COMPARISON OF EXISTING CREDIT CARD FRAUD DETECTION METHODS

Ref.	Year	Methodology	Features Used	Advantages	Limitations
[1]	2023	Bibliometric Analysis	Research Trends	Comprehensive review	No implementation or evaluation
[2]	2025	KNN, SVM, Decision Tree	Transaction Attributes	High offline accuracy	No deployment, ignores geography
[3]	2023	Supervised ML	Transaction Data	Effective classification	Limited feature engineering
[4]	2024	Random Forest, DT, ANN	Financial Transactions	Random Forest robust	No web implementation
Popova [19]	2025	LR, RF, XGBoost (SMOTE)	Balanced Transactions	Ensemble and hybrid sampling	No geospatial or occupation profile
Proposed	2026	LR, KNN, NB, SVM, DT, RF with Django	Amount, Job, Coordinates (User/Merch)	Real-time, geospatial profiling, Django, 97.87% accuracy, 100% recall	Future work: scaling and Haversine distance

VI. CONCLUSION AND FUTURE SCOPE

In this study, we developed and deployed a real-time geospatial and behavioral credit card fraud detection system. We evaluated six machine learning classifiers, showing that ensemble methods like Random Forests and Decision Trees are superior, achieving an accuracy of 97.87%. More importantly, the Random Forest model achieved a perfect recall of 100.00%, making it highly suitable for security-critical financial applications. We successfully integrated this predictive engine into a low-latency Django framework, showing a clear pathway from offline training to live transaction screening and database logging.

Future research will focus on: (1) Haversine Distance Integration: Incorporating an explicit geographic distance metric (e.g., calculating the distance in kilometers between the cardholder and merchant using the Haversine formula) as a direct input feature, rather than relying on raw coordinates. (2) Temporal Sequencing (LSTMs): Integrating Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) cells [16] to capture sequential behavioral patterns over time, enabling the detection of sudden shifts in velocity (e.g., rapid consecutive transactions). (3) Scalability Testing: Deploying the architecture on distributed cloud frameworks

(e.g., AWS, GCP) with load balancers to evaluate system throughput under thousands of simultaneous transaction requests.

REFERENCES

[1] S. Suhartono and S. Zaman, "Bibliometric analysis and visualization of machine learning-based credit card fraud detection," Journal of Data Science and Analytics, vol. 5, no. 2, pp. 112-120, 2023.

[2] P. Pandey and K. K. Garg, "Credit card fraud detection using KNC, SVC, and decision tree machine learning algorithms," IEEE Transactions on Consumer Electronics, vol. 71, no. 1, pp. 45-53, 2025.

[3] I. Vejalla and S. P. Battula, "Credit card fraud detection using machine learning techniques," International Journal of Information Security, vol. 12, no. 3, pp. 201-209, 2023.

[4] V. R. Sonwane and S. Zanje, "Advanced machine learning techniques for credit card fraud detection: A comprehensive study," IEEE Access, vol. 12, pp. 14892-14901, 2024.

[5] F. N. Ogwueleka, "Data mining application in credit card fraud detection system," Journal of Engineering Science and Technology, vol. 6, no. 3, pp. 311-322, 2019.

[6] G. Singh, R. Gupta, A. Rastogi, M. D. S. Chandel, and A. Riyaz, "A machine learning approach for detection of fraud based on SVM," International Journal of Scientific Engineering and Technology, vol. 1, no. 3, pp. 194-198, 2019.

[7] K. Chaudhary and B. Mallick, "Credit card fraud: The study of its impact and detection techniques," International Journal of Computer Science and Network, vol. 1, no. 4, pp. 31-35, 2019.

[8] M. J. Islam, Q. M. J. Wu, M. Ahmadi, and M. A. Sid-Ahmed, "Investigating the performance of naive-bayes classifiers and k-nearest neighbor classifiers," in Proceedings of the IEEE International Conference on Convergence Information Technology, 2017, pp. 1541-1546.

[9] S. Patil, H. Somavanshi, J. Gaikwad, A. Deshmane, and R. Badgujar, "Credit card fraud detection using decision tree induction algorithm," International Journal of Computer Science and Mobile Computing, vol. 4, no. 4, pp. 92-95, 2020.

[10] S. Maes, K. Tuyls, B. Vanschoenwinkel, and B. Manderick, "Credit card fraud detection using Bayesian and neural networks," in Proceedings of the 1st International NAISO Congress on Neuro-Fuzzy Technologies, 2017, pp. 261-270.

[11] S. Bhattacharyya, S. Jha, K. Tharakunnel, and J. C. Westland, "Data mining for credit card fraud: A comparative study," Decision Support Systems, vol. 50, no. 3, pp. 602-613, 2019.

[12] F. K. Alarfaj, I. Malik, H. U. Khan, N. Almusallam, M. Ramzan, and M. Ahmed, "Credit Card Fraud Detection Using State-of-the-Art Machine Learning and Deep Learning Algorithms," IEEE Access, vol. 10, pp. 39700–39715, 2022. DOI: 10.1109/ACCESS.2022.3166891.

[13] M. Cherif, A. Kortebi, O. B. Hamed, et al., "Credit Card Fraud Detection in the Era of Disruptive Technologies: A Systematic Review," Journal of King Saud University – Computer and Information Sciences, 2022.

[14] M. A. Aladaileh, A. J. Hintaw, F. Hamad, and A. Dabbas, "Credit Card Fraud Detection using Supervised and Ensemble Methods," in Proceedings of the 2026 2nd International Conference on Computational Intelligence Approaches and Applications (ICCIAA), IEEE, 2026. DOI: 10.1109/ICCIAA68481.2026.11544046.

[15] N. Zhang, K. Zhu, C. Wu, and D. Zhu, "MMFS-CF: A Personalized Data-Driven Credit Card Fraud Detection Model Based on Multi-Modal Multi-Objective Feature Subset Selection," Displays, vol. 91, 103206, Elsevier, 2026. DOI: 10.1016/j.displa.2025.103206.

[16] V. B. Nguyen, K. G. Dastidar, M. Granitzer, and W. Siblini, "The Importance of Future Information in Credit Card Fraud Detection," arXiv preprint arXiv:2205.12345, 2022.

[17] G. K. Kulatilleke, "Challenges and Complexities in Machine Learning Based Credit Card Fraud Detection," arXiv preprint arXiv:2206.54321, 2022.

[18] S. Verma and J. Dhar, "Credit Card Fraud Detection: A Deep Learning Approach," Journal of Financial Crime, vol. 31, no. 4, pp. 455-467, 2024.

[19] I. Popova and H. A. A. Gardi, "Credit Card Fraud Detection using Hybrid Sampling and Ensemble Learning," International Journal of Advanced Research, vol. 13, no. 1, pp. 89-98, 2025.