DEVELOPMENT OF MACHINE LEARNING-BASED ALGORITHM FOR ENHANCING TRANSACTION FRAUD DETECTION

by

Nguyen Duy Anh Luong, BSc, Western University, 2024

A Major Research Project
presented to Toronto Metropolitan University

in partial fulfillment of the requirements for the degree of

Master of Science
in the Program of
Data Science and Analytics

Toronto, Ontario, Canada, 2025

AUTHOR'S DECLARATION FOR ELECTRONIC SUBMISSION OF A MAJOR RESEARCH PROJECT (MRP)

I hereby declare that I am the sole author of this Major Research Paper. This is a true copy of the MRP, including any required final revisions, as accepted by my examiners.

I authorize Toronto Metropolitan University to lend this MRP to other institutions or individuals for the purpose of scholarly research.

I further authorize Toronto Metropolitan University to reproduce this MRP by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

I understand that my MRP may be made electronically available to the public.

Nguyen Duy Anh Luong

ACKNOWLEDGEMENTS

First and foremost, I would like to sincerely thank **Dr. Shengkun Xie**, my Major Research Project supervisor, for his consistent support and guidance throughout this work. His insightful feedback, depth of knowledge, and willingness to make time for questions were invaluable to the development of this project. I am truly grateful for his mentorship and encouragement along the way.

I would also like to thank the professors in the *Master of Science in Data Science and Analytics* program at Toronto Metropolitan University. Their teaching, guidance, and support throughout the program helped me grow both technically and intellectually. This experience has been truly formative and inspiring.

Finally, I would like to thank **Kaggle** for providing open access to the dataset used in this study. The availability of such real-world data was essential to the development and evaluation of this research.

DEVELOPMENT OF MACHINE LEARNING-BASED ALGORITHM FOR ENHANCING TRANSACTION FRAUD DETECTION

Nguyen Duy Anh Luong
Master of Science 2025
Data Science and Analytics
Toronto Metropolitan University

ABSTRACT

This Major Research Project explores the development of a supervised machine learning-based algorithm to enhance transaction fraud detection using the IEEE-CIS Fraud Detection dataset. The study addresses key challenges such as severe class imbalance, anonymized features, and the need for early, accurate fraud detection. A comprehensive literature review and exploratory data analysis guided the selection of relevant features and modeling techniques. Several machine learning models were evaluated, including Logistic Regression, Random Forest, Artificial Neural Networks (ANN), Deep Neural Networks (DNN), and XGBoost. Performance was assessed using precision, recall, F1-score, and PR-AUC. Results show that XGBoost performed strongly, and an XGBoost+DNN ensemble achieved the highest precision-recall performance, offering a good balance between detecting fraud and avoiding false positives. Feature engineering and identity-based variables such as device type and email domain further improved model effectiveness. The final model offers a scalable, interpretable, and high-performing solution for fraud detection in financial systems.

Key words:

Fraud Detection; Machine Learning; XGBoost; Deep Learning; Imbalanced Data

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. Introduction

1.1 Background

The rapid growth of online transactions has transformed the way people shop, pay bills, and transfer money. While these services offer convenience and speed, they also create opportunities for fraudulent activity. Financial fraud not only results in direct monetary losses but can also erode public trust and harm the reputation of financial institutions.

Fraudulent activities are constantly evolving, making them difficult to detect with traditional rule-based systems, which often rely on fixed patterns and manual updates. In recent years, machine learning (ML) has emerged as a more adaptive solution, capable of analyzing large volumes of data and identifying patterns that are difficult to detect manually. Supervised ML models, in particular, have shown promise in detecting fraudulent transactions by learning from historical data and adapting to new, unseen examples.

1.2 Problem Statement

Despite advances in Machine Learning, transaction fraud detection remains challenging. One of the most significant issues is class imbalance, as fraudulent cases make up only a small fraction of total transactions. In the IEEE-CIS dataset used in this study, they account for about 3.5% of all records. This imbalance can lead models to favor predicting the majority class, resulting in missed detections.

The dataset also presents other difficulties, including anonymized feature names, high dimensionality, and complex feature relationships. These factors require careful data preprocessing and feature engineering to extract meaningful patterns. In addition, a practical fraud detection system must work quickly and accurately to stop fraudulent transactions before losses occur, making both performance and efficiency critical.

1.3 Research Objectives

The main goals of this project are to:

1. Identify the supervised ML model that provides the best trade-off between detection accuracy and false positives.
2. Examine whether the addition of identity-based features, such as device type and email domain, improves detection rates.
3. Test different techniques for handling class imbalance, including resampling, class weighting, and threshold tuning.
4. Evaluate whether combining models in an ensemble improves performance compared to single-model approaches.
5. Explore the role of feature engineering in enhancing model accuracy and robustness.

2. Literature Review

This literature review focuses on supervised machine learning approaches for transaction fraud detection in real-world scenarios with imbalanced and anonymized data. Insights are drawn from fifteen peer-reviewed journal and conference papers, examining techniques for handling class imbalance, applying effective feature engineering, selecting appropriate models, and leveraging identity-based variables to improve fraud detection performance.

2.1 Supervised Learning Models in Fraud Detection

The exploration began with how supervised learning models are applied in fraud detection, as they form the foundation of most modern systems. These models are trained on historical data, where each transaction is labeled as either fraudulent or legitimate. By learning from these patterns, predictions can be made on new, unseen transactions.

One of the most notable approaches reviewed is the Distributed Deep Neural Network (DDNN) model proposed by Lei et al. (2023), which focuses on both accuracy and user privacy [1]. Instead of centralizing all transaction data, this method allows financial institutions to train local models on companies' own data while only sharing model parameters with a central server. This approach not only protects user privacy but also reduces data handling costs and improves training efficiency through parallel computing. Experimental results showed that the DDNN model outperformed centralized models in terms of accuracy, precision, recall, and F1-score.

In addition, more interpretable models such as decision trees and random forests were explored. The 2023 study Credit Card Fraud Detection using Decision Tree and Random Forest by Shah and Sharma examined these two algorithms using a simulated credit card transaction dataset [2]. The results showed that while decision trees are easy to understand, they tend to overfit the training data. Random forests, which combine multiple decision trees, performed better overall, particularly after hyperparameter tuning. However, both models continued to struggle with class imbalance, which limited their effectiveness in accurately detecting fraudulent transactions.

To explore this challenge further, a large-scale study by Alfaiz and Fati (2022) was examined, in which 66 combinations of nine machine learning algorithms and nineteen resampling techniques were evaluated [3]. This approach was particularly valuable, as a real-world dataset was used and each combination was systematically compared. The best results were achieved by combining CatBoost, an advanced gradient boosting model, with the AllKNN undersampling method. This combination produced an F1-score of 87.40%, a recall of 95.91%, and an AUC of 97.94%, outperforming many traditional methods. The findings emphasized the importance of selecting both an effective model and an appropriate data balancing strategy when working with imbalanced fraud datasets.

Overall, these studies provided a clearer understanding of the strengths and limitations of different supervised learning models. Deep learning approaches like DDNN have demonstrated strong performance, particularly in scenarios where privacy and scalability are critical. In contrast, interpretable models such as random forests and CatBoost have also proven to be highly effective when supported by thoughtful preprocessing and well-selected data balancing techniques.

2.2 Imbalanced Data: Core Challenge and Solutions

As supervised learning models were explored more deeply, class imbalance quickly stood out as one of the biggest challenges in fraud detection. In most real-world datasets, fraudulent transactions make up only a small fraction of the total, which often causes models to favor legitimate transactions and miss fraud cases. To build a system that performs well, this imbalance must be addressed so that rare fraud cases can be detected without generating too many false positives.

In the paper titled Hybrid Undersampling and Oversampling for Handling Imbalanced Credit Card Data, Alamri and Ykhlef (2022) introduced a hybrid resampling technique called BCB-SMOTE, which combines Tomek links for noise reduction, BIRCH clustering, and Borderline SMOTE to generate synthetic samples near the decision boundary [4]. When tested with a random forest classifier, the proposed method achieved an F1-score of 85.2% and successfully addressed common oversampling issues such as data overlap and overfitting. Similarly, in Performance Evaluation of Machine Learning Methods for Credit Card Fraud Detection Using SMOTE and AdaBoost, Ileberi et al. (2021) demonstrated that combining SMOTE with AdaBoost has a positive impact on the performance across several machine learning models [5].

Building on these ideas, the study Credit Card Fraud Detection in Imbalanced Datasets: A Comparative Analysis of Machine Learning Techniques by Lahbiss and Chtouki (2024) evaluated traditional, ensemble, and deep learning models using SMOTE and SMOTE-ENN [6]. It was found that Random Forest combined with SMOTE-ENN achieved an AUC-ROC of 0.85, while LSTM with SMOTE-ENN reached an even higher AUC-ROC of 0.90. These results indicate that the combination of resampling strategies with strong classifiers can significantly improve fraud detection performance.

Together, the reviewed studies demonstrate that no one-size-fits-all solution exists. Instead, effective fraud detection in imbalanced datasets is often achieved by combining resampling techniques with models such as Random Forest or LSTM to enhance predictive accuracy.

2.3 Feature Engineering and Identity-Based Enhancements

After the impact of class imbalance was explored, attention was turned to feature engineering. Even the most advanced algorithms cannot perform well without meaningful input features. In fraud detection, basic variables such as transaction amount or card type provide limited predictive value on their own. Significant improvements are typically achieved by transforming features and incorporating identity-related data that more accurately reflects user behavior.

One study that demonstrated the value of feature engineering was conducted by Lei et al. (2020), where XGBoost was applied to the IEEE-CIS dataset [7]. Emphasis was placed on careful data cleaning, handling of missing values, consistent label encoding, and feature elimination. By combining transaction data with identity fields such as device type and email domain, model performance was significantly improved, achieving a ROC-AUC of 0.942 and an accuracy of 97.6%. Another paper by Lucas et al. (2019) introduced a feature engineering method based on Hidden Markov Models (HMMs) [8]. Instead of using standard transaction aggregates, sequences were modeled using eight combinations derived from three binary perspectives: whether the sequence was linked to a cardholder

or a terminal, whether it was based on transaction amounts or time elapsed, and whether the history was genuine or fraudulent. When the multi-perspective HMM-based features were incorporated into a Random Forest classifier, a 15.1% boost in Precision-Recall AUC was observed.

In addition, Bahnsen et al. (2016) proposed a strategy combining transaction aggregation with periodic behavior modeling [9]. Features were created based on customer spending patterns, and the von Mises distribution was used to model typical transaction times. These features led to a 13% increase in savings, reflecting improved prevention or recovery of losses due to fraud.

Overall, these studies show that advanced feature engineering, such as the use of identity-based data and behavioral patterns, plays a crucial role in improving model accuracy, adaptability, and robustness in real-world fraud detection.

2.4 Ensemble Learning and Hybrid Models

After the performance of individual models and the impact of feature engineering were examined, attention was directed toward ensemble learning to evaluate whether combining models could lead to improved results. Ensemble methods are designed to combine multiple classifiers to enhance accuracy, reduce bias, and better capture the complexity of fraud patterns, which often vary across users and evolve over time.

A study conducted by Carcillo et al. (2021) introduced a hybrid approach that integrated unsupervised and supervised learning techniques [10]. In this approach, outlier scores were first generated using unsupervised models and then passed into a supervised classifier to make the final decision. Fraud was evaluated at multiple levels, such as globally across the dataset, locally for individual cardholders, and within clusters of similar customers, enabling the model to detect a broader range of fraud behaviors more effectively. In a related study, a dynamic ensemble selection method was proposed by Achakzai and Peng (2023), where the most effective classifiers were selected for each transaction based on local competence [11]. This dynamic approach consistently outperformed static ensemble classifiers in both accuracy and overall performance.

Further supporting the value of ensemble approaches, Jahnavi et al. (2024) proposed a hybrid model that combines decision trees with logistic regression for fraud detection [12]. An accuracy of 98.1% was achieved, with strong performance observed in both sensitivity and flexibility. By combining logistic regression with the decision-making structure of decision trees, a model was developed that can adapt to evolving fraud tactics. In a related study, several ensemble ML models, including Decision Tree, Random Forest, XGBoost, CatBoost, and Gradient Boosting, were compared by Chaurasia et al. (2024) [13]. Both balanced and imbalanced versions of the European cardholder dataset were used, and XGBoost was found to deliver the best performance in terms of F1-score and recall, making it particularly effective at identifying rare fraud cases.

Together, these studies highlight that selecting an appropriate ensemble approach and combining it with effective data balancing strategies can significantly improve the accuracy and reliability of fraud detection systems.

2.5 Comparative Reviews

In addition to individual models and techniques, broader research comparing multiple machine learning approaches for fraud detection has also been examined. These comparative reviews provide a clearer understanding of which methods tend to perform well across different scenarios and offer practical insights into model selection.

A comparative review conducted by Patel et al. (2024) evaluated several supervised machine learning models for credit card fraud detection, including artificial neural networks (ANN), logistic regression (LR), Naive Bayes (NB), and K-nearest neighbors (KNN) [14]. The results showed that ANN achieved the highest accuracy at 98.9%, followed by logistic regression at 98.6%, Naive Bayes at 98.4%, and KNN at 96.6%. Although ANN delivered the top performance, the study also emphasized that models such as logistic regression and Naive Bayes demonstrated strong performance across multiple evaluation metrics.

In a related study, deep learning methods were compared with traditional machine learning techniques, such as support vector machines, random forests, and logistic regression, by Jyoti et al. (2024), using three datasets [15]. A deep neural network (DNN), trained with the Adam optimizer, achieved an accuracy of 99.4% on the European credit card dataset. The advantages of using Adam were emphasized, including computational efficiency, low memory usage, and suitability for large datasets. Fraudulent transactions were accurately classified, and error rates were kept low.

Together, these studies indicate that although deep learning models often achieve the highest accuracy, traditional models such as logistic regression continue to offer strong and dependable performance. This highlights the importance of choosing models based on both effectiveness and practicality in real-world fraud detection.

2.6 Gaps and Contributions of This Project

In summary, this project builds on existing research in transaction fraud detection by developing an optimized supervised machine learning algorithm designed to perform effectively on a large, complex, and imbalanced real-world dataset. Although many previous studies have explored different models and techniques, few have addressed all the key challenges in combination, such as class imbalance, meaningful feature selection, and the use of identity-related data like device type and email domain. For this Major Research Project, several machine learning models will be compared, including logistic regression, random forest, artificial neural networks, deep neural networks, and XGBoost, to determine the most effective approach.

3. Data Description

In this section, patterns in fraudulent behavior were explored using a combination of visualizations and statistical analysis. The goal was to uncover meaningful insights that could inform model development and improve fraud detection accuracy. Each analysis was aligned with the previously defined research questions, with a focus on identifying trends, anomalies, or potential feature transformations that could enhance the model's ability to detect fraud effectively.

3.1 Data Source

For this project, the IEEE-CIS Fraud Detection dataset was used, which was made available through a Kaggle competition hosted in collaboration with Vesta, a global fraud prevention company. The dataset represents real-world e-commerce transaction environments and is intended to support the development of machine learning models for fraud detection.
Source: `https://www.kaggle.com/competitions/ieee-fraud-detection/data`

3.2 Data Acquisition

The dataset was downloaded directly from Kaggle. It includes separate files for transaction and identity information, which are joined using the `TransactionID` key. The training set contains labeled examples (with the binary `isFraud` target), while the test set includes similar features but does not contain fraud labels.

3.3 Data Files

- `train_transaction.csv` and `train_identity.csv`: Training data with features and the isFraud label (1 = fraud, 0 = non-fraud)
- `test_transaction.csv` and `test_identity.csv`: Test data without labels

Transaction Features:

- `TransactionDT`: Relative timestamp
- `TransactionAmt`: Transaction amount in USD
- `ProductCD`: Product category code
- `card1`–`card6`: Card-related attributes (e.g., type, issuer)
- `addr1`, `addr2`: Geographic location codes
- `C1`–`C14`, `D1`–`D15`, `V1`–`V339`: Engineered and anonymized features
- `P_emaildomain`, `R_emaildomain`: Purchaser and recipient email domains
- `M1`–`M9`: Binary match indicators

Identity Features:

- `DeviceType`, `DeviceInfo`: Device metadata
- `id_12`–`id_38`: Browser, OS, network, and identity-related signals

3.4 Data Constraints and Preprocessing

Before the analysis was conducted, the raw transaction and identity datasets were cleaned to make them easier to work with. Column names containing hyphens and extra spaces were modified by replacing hyphens with underscores and removing any leading or trailing whitespace to avoid issues in Python. Missing values also required significant attention during preprocessing. For columns containing text or categorical data, missing values were replaced with the word "missing" to clearly indicate that information was originally not provided. For numerical columns, missing values were retained as None, allowing them to be handled more carefully during the analysis or modeling phase. This preprocessing step was performed using a simple Python script that cleaned both `train_transaction.csv` and `train_identity.csv`. The cleaned versions, `cleaned_train_transaction.csv` and `cleaned_train_identity.csv`, were then saved and used for all subsequent steps in the exploratory data analysis (EDA).

3.5 Descriptive Statistics

Before visual exploration was performed, basic statistics were reviewed to better understand the structure of the dataset. The transaction data contains over 590,000 records, with only 3.5% labeled as fraudulent. The variable `TransactionAmt` ranges from a few cents to several thousands of dollars, with a median of around $68. Some features contain a high amount of missing values, particularly within the identity data. Categorical variables such as `ProductCD` and `card4` have a small number of unique values, while anonymized numerical features (`V1`–`V339`) have wide distributions. Overall, these initial findings helped guide the direction of the subsequent exploratory data analysis (EDA).

4. Exploratory Data Analysis (EDA)

4.1 Class Imbalance and Baseline Performance

When examining the dataset for the first time, one of the most noticeable patterns is the large imbalance between legitimate and fraudulent transactions. Out of a total of 590,540 records, only 20,663 transactions (3.50%) are labeled as fraudulent, while the remaining 96.5% are legitimate. This shows that fraud is not only rare but also potentially very costly when it occurs.

A simple bar chart (Figure 1) was created to show the distribution of the two classes. As expected, legitimate transactions dominate the chart. Percentages were added above each bar to highlight the imbalance: 96.5% legitimate vs. 3.5% fraudulent.

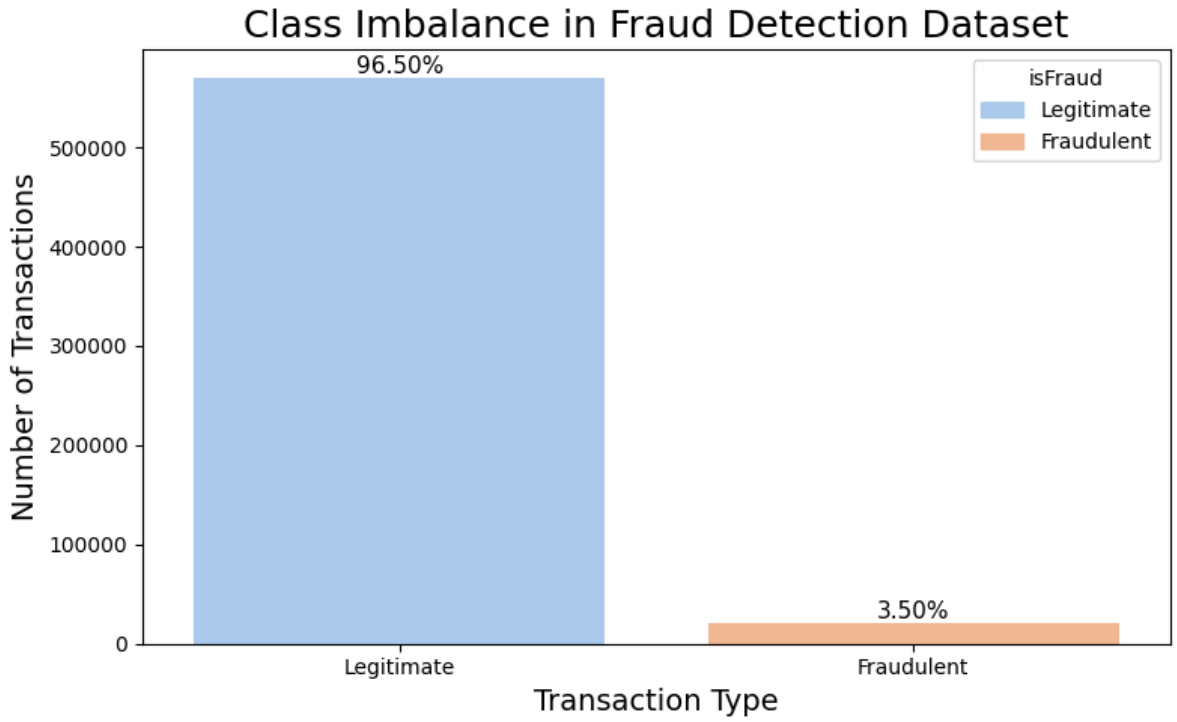

Figure 1: Fraud Distribution by Transaction Type

To demonstrate why this imbalance is a serious issue for machine learning, a basic test was done using a dummy classifier that always predicts the majority class (not fraud). The model achieved 96.5% accuracy, which might seem impressive at first, but it failed to catch any fraud cases. Precision, recall, and F1-score were all 0.000, making the model completely ineffective for detecting fraud.

To put this into a real-world perspective, consider an estimate where each missed fraud costs about $500. Missing 20,663 fraud cases would translate to over $10 million in potential losses. This highlights why accuracy alone is not a reliable performance measure for fraud detection.

Overall, this analysis confirms that fraud detection cannot be treated as a standard classification problem. Because of how rare and high-impact fraud cases are, special techniques such as resampling (e.g., SMOTE), class weighting, or ensemble models like XGBoost are needed to address the imbalance and improve detection performance.

4.2 Transaction Amount and Product Category Patterns

The distribution of TransactionAmt was first visualized for both fraudulent and legitimate transactions. While the majority of transactions in both classes occurred at lower amounts, fraudulent transactions were more widely distributed, especially in the $100 to $500 range. This pattern was clearly visible in the KDE plot (Figure 2) below.

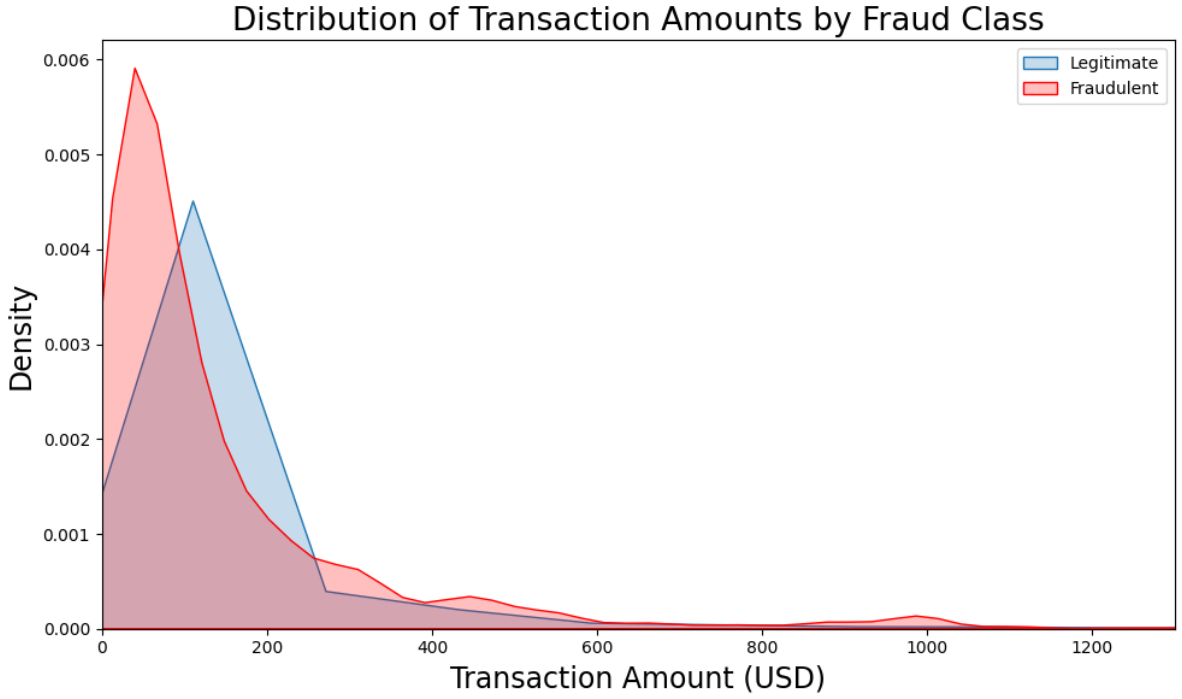

Figure 2: Distribution of Transaction Amounts by Fraud Class

The boxplot (Figure 3) also provided additional insight. Fraudulent transactions showed a greater number of high-value outliers compared to legitimate ones. This supports the idea that certain fraud attempts involve larger purchases, possibly with the intent of maximizing value before detection.

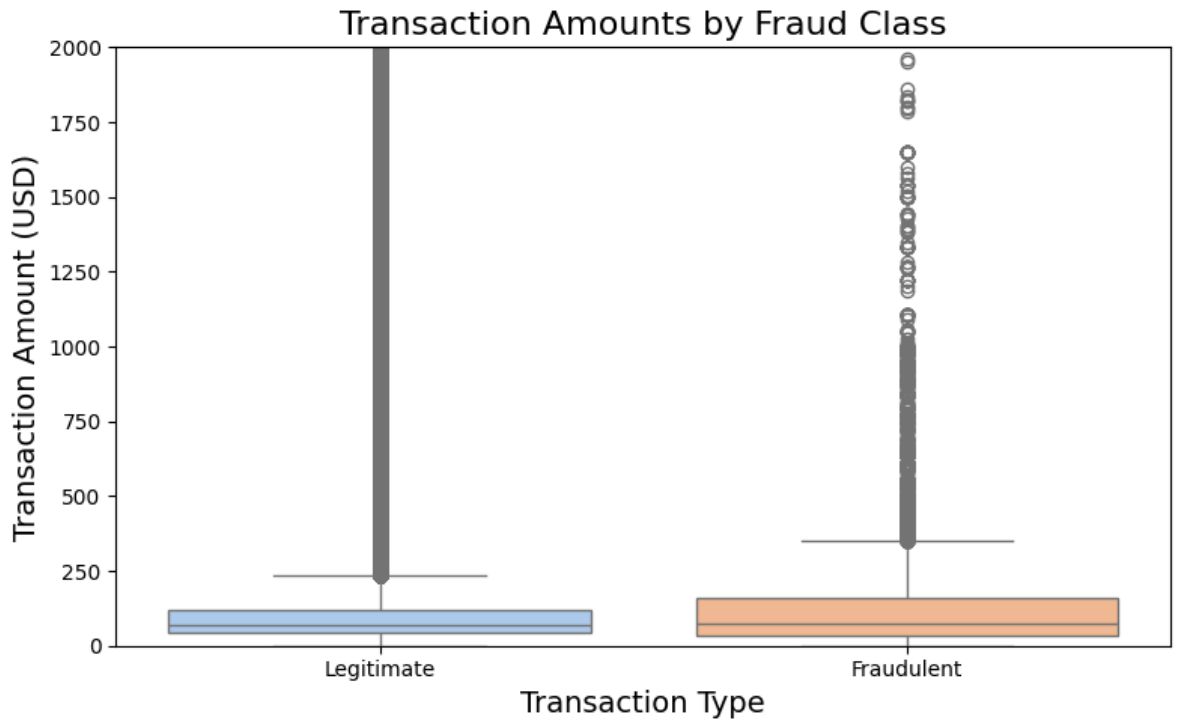

Figure 3: Transaction Amounts by Fraud Class (Boxplot)

Additionally, the variable ProductCD, which represents product or service types, was also examined. Fraud rates varied noticeably across categories:

- Product C had the highest fraud rate at approximately 11.7%.
- Product W had the lowest, around 2.1%.

This suggests that some product categories are more frequently targeted in fraud attempts, offering valuable information for feature selection.

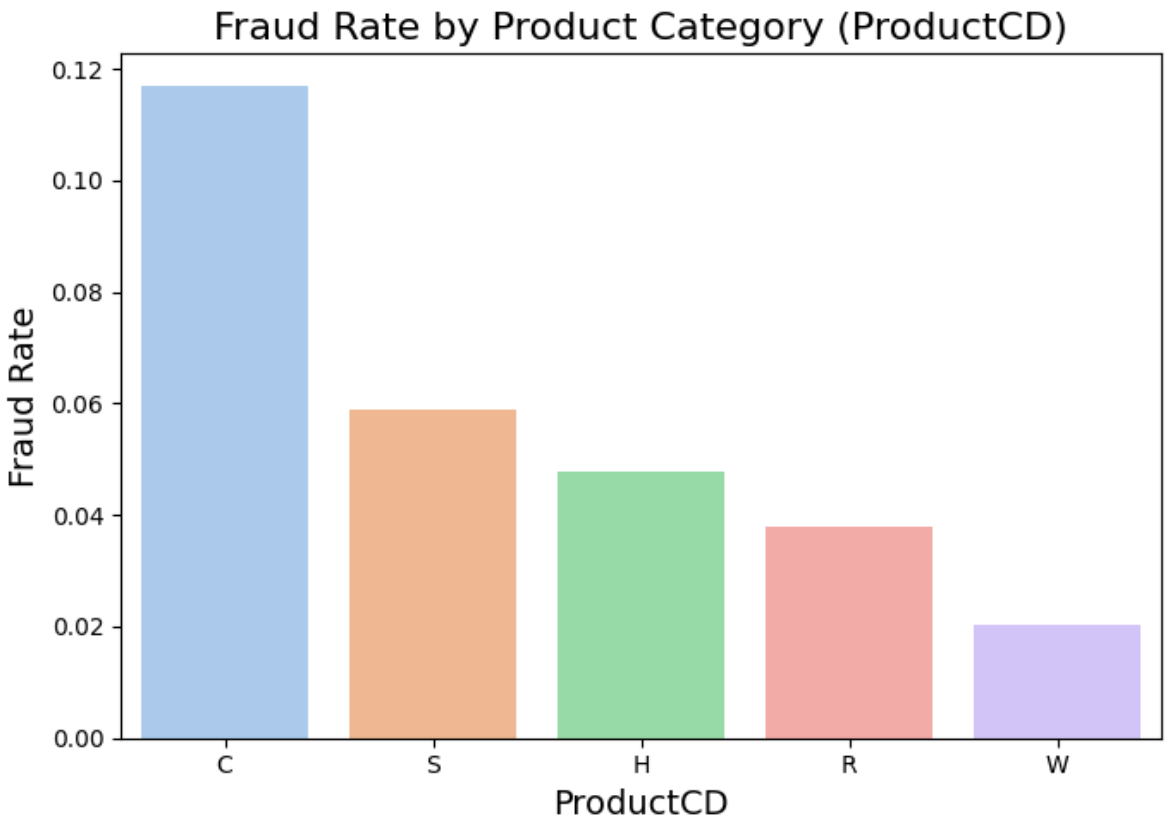

Figure 4: Fraud Rate by Product Category (ProductCD)

To further explore the relationship between transaction amount and product type, a new feature called amt_over_500_risky_product was created, which flags transactions over $500 in the three product categories with the highest fraud rates (C, S, and H). This group showed a slightly higher fraud rate (3.64%) than the rest of the data (3.50%), which suggests the combination does capture a modestly higher-risk segment.

Two key insights emerge from this analysis. First, both transaction amount and product type provide useful fraud signals, particularly when considered in combination. Second, feature engineering can uncover important patterns, but assumptions should always be validated with data, as not every engineered feature yields significant improvement.

Overall, these findings will help inform the selection and refinement of input features for machine learning models. Models such as XGBoost and Deep Neural Networks are especially likely to benefit from well-designed features that expose meaningful fraud-related patterns.

4.3 Temporal Patterns in Fraudulent Activity

This part of the analysis focused on uncovering time-based patterns in fraudulent transactions using the TransactionDT field. Since this field represents a time delta in seconds, it was transformed into more interpretable units: transaction day, hour of day, and a weekend indicator. These transformations supported the investigation of whether fraud is more likely to occur at specific times, aligning with RQ5: How does feature engineering affect fraud detection?

The fraud rate by hour of day revealed a clear and meaningful pattern. Fraudulent activity was most frequent in the early morning hours, peaking sharply around 7 AM with a fraud rate exceeding 10%, and gradually declining throughout the rest of the day. This suggests that early hours may be targeted by fraudsters, possibly due to lower detection activity during off-peak periods.

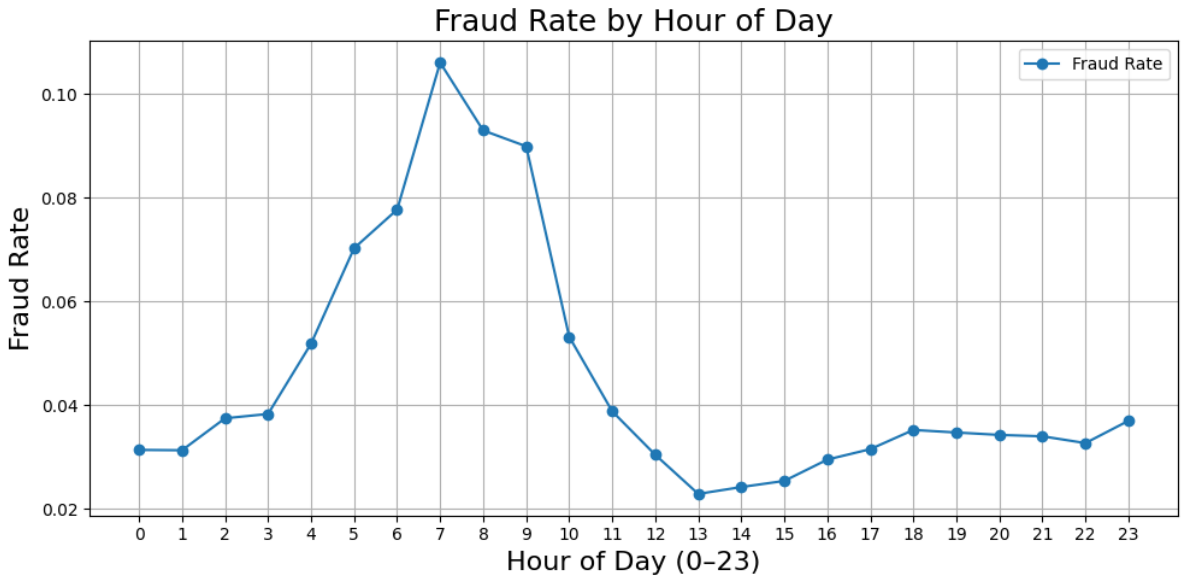

Figure 5: Fraud Rate by Hour of Day

Fraud trends by transaction day were also examined. While the pattern was less consistent, several noticeable spikes were observed in the fraud rate over time. These spikes may indicate periodic fraud campaigns or temporary system vulnerabilities that were exploited.

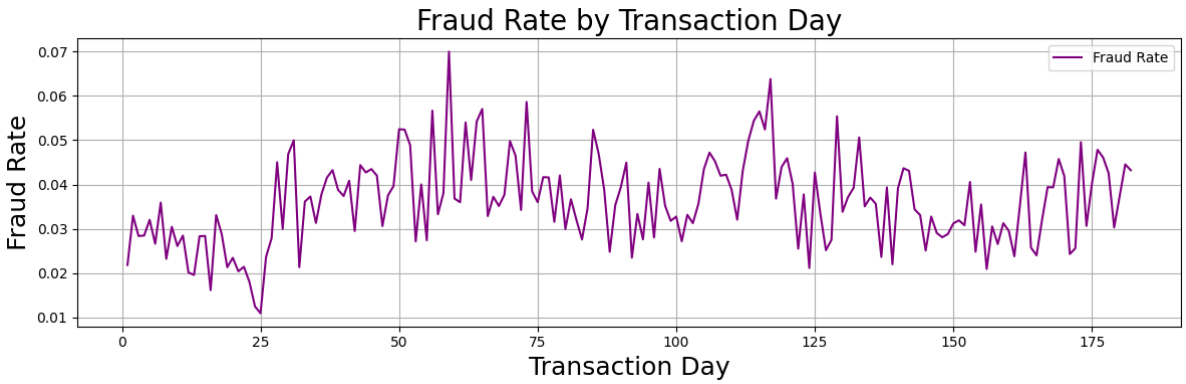

Figure 6: Fraud Rate by Transaction Day

Additionally, fraud rates were compared between weekdays and weekends. A slightly higher fraud rate was observed on weekdays (3.55%) compared to weekends (3.38%), though the difference was modest. This may reflect higher transaction volumes and more varied behavior during business days.

Overall, the results suggest that time-based features contain valuable behavioral signals for fraud detection. Patterns observed in this analysis can guide future feature engineering, and variables such as transaction_hour or is_weekend may contribute meaningfully to the performance of supervised models like XGBoost or deep neural networks.

4.4 Regional and Geolocation Signals

This analysis examined whether the location-based features addr1 (region) and addr2 (country) show patterns related to fraud. This supports RQ5, which focuses on identifying features, particularly engineered ones, that enhance model performance.

Fraud rates were found to vary significantly across different addr1 regions. For example, in Figure 7, billing region 465 showed fraud rates above 8%, while many other regions remained considerably lower. This suggests that regional fraud risk may be influenced by geographic, demographic, or institutional factors.

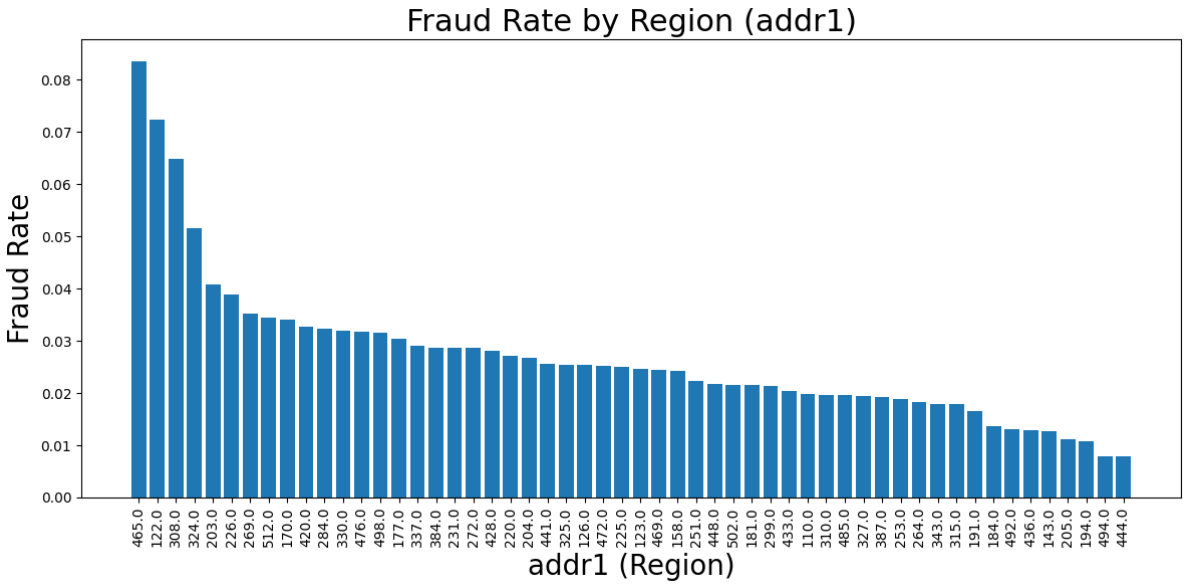

Figure 7: Fraud Rate by Region (addr1)

In the case of addr2, a sharp contrast in fraud rates across country codes was observed. Transactions from country code 87 (likely representing domestic U.S. cards) had a relatively low fraud rate of 2.4%, while transactions associated with other codes, such as 60 and 96, showed significantly higher rates ranging from 8.9% to 13.9%.

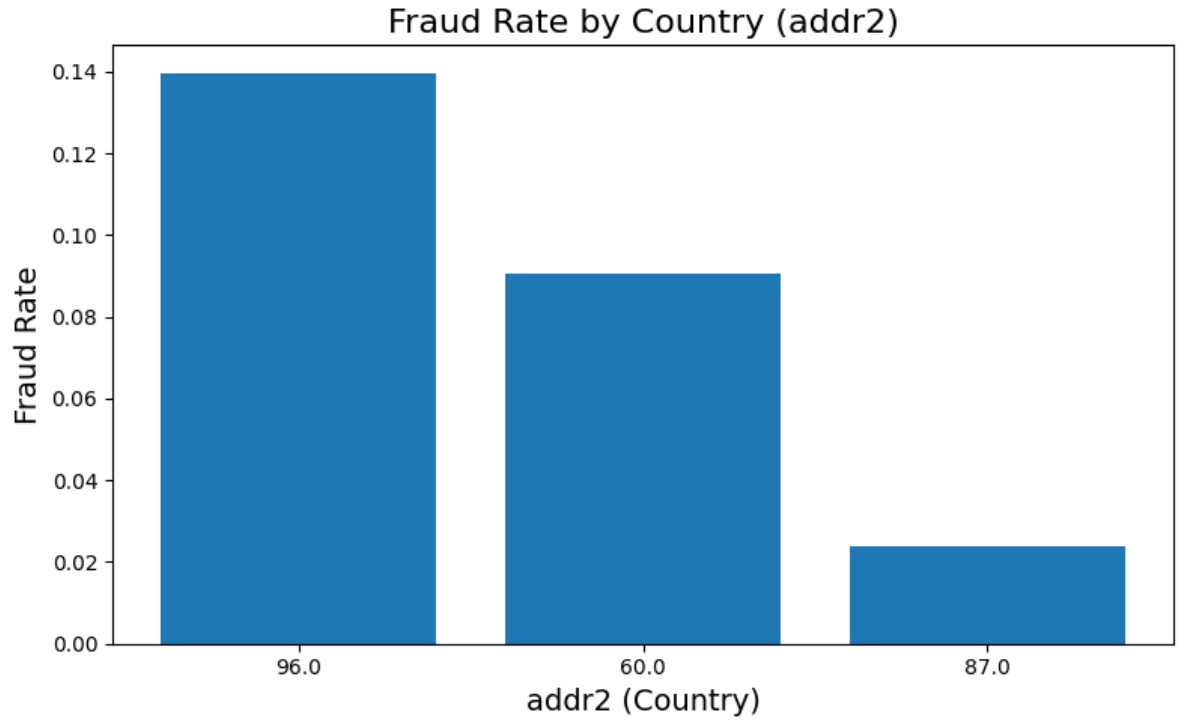

Figure 8: Fraud Rate by Country (addr2)

To further investigate this pattern, a new binary feature named foreign_transaction_flag was created, labeling transactions as foreign if addr2 is not equal to 87. This feature revealed a strong signal: foreign transactions had a fraud rate of 11.7%, nearly five times higher than that of domestic transactions (2.40%).

These results indicate that location-based fields such as addr1 and addr2 carry meaningful fraud-related information. Insights from this analysis can guide the development of new features, such as region_risk_flags or foreign_transaction_indicators, that can improve model performance in later stages, especially for models like XGBoost and DNNs that benefit from meaningful categorical or binary inputs.

4.5 Identity-Based Features: Email Domain and Device Type

This analysis focused on identity-related features such as email domain and device type to assess whether they contribute to fraud detection. The results directly support RQ2, which investigates whether merging identity data with transaction data can improve model performance.

Purchaser email domains displayed notable differences in fraud risk. For example, transactions associated with outlook.com had a fraud rate of approximately 9.5%, while hotmail.com and gmail.com followed with rates of 5.3% and 4.4%, respectively. In contrast, domains such as yahoo.com and aol.com showed significantly lower fraud rates, typically around 2.1%. Even domains labeled as anonymous or missing carried a fraud risk, with rates ranging from 2.5% to 2.9%, indicating that they may still provide useful signals.

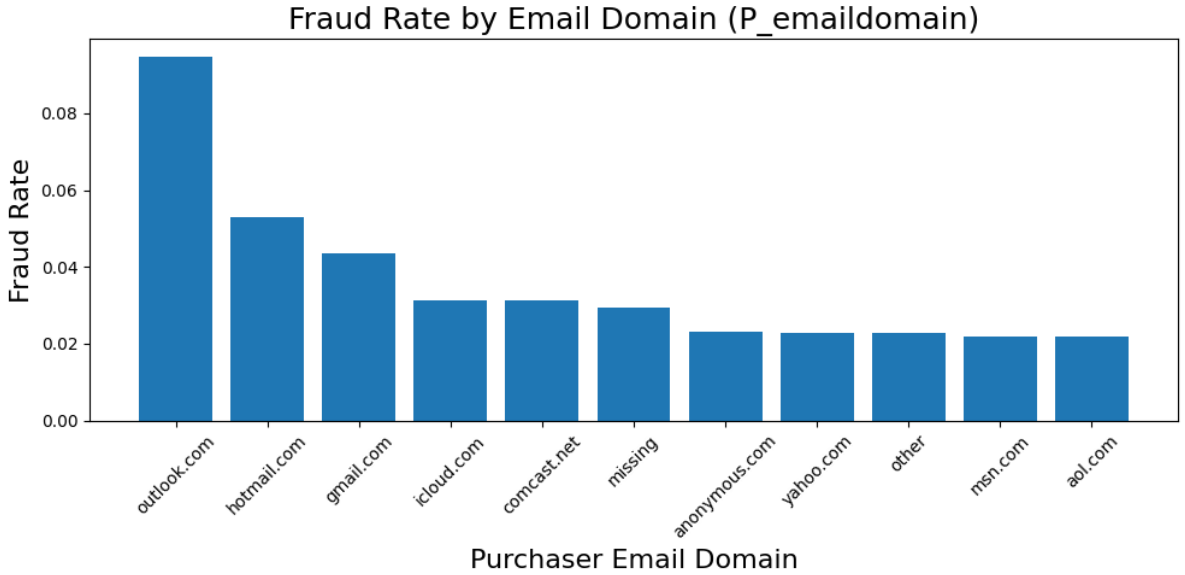

Figure 9: Fraud Rate by Email Domain (P_emaildomain)

Additionally, device type also revealed strong patterns related to fraud. Transactions conducted on mobile devices had the highest fraud rate at 10.1%, followed by desktop transactions at 6.5%. Records with missing device type information showed the lowest fraud rate at 3.1%.

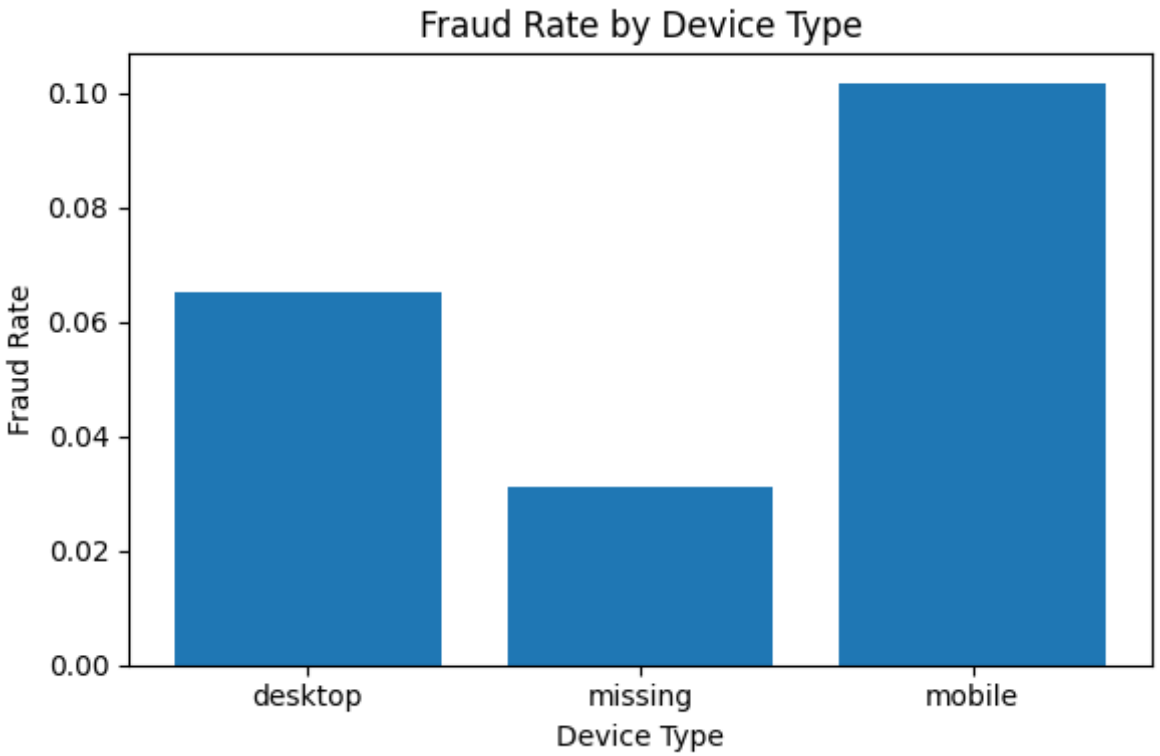

Figure 10: Fraud Rate by Device Type

Further insights were found in the DeviceInfo field. Certain Android models, such as "SM-J700M Build/MMB29K", showed fraud rates above 11%. In contrast, devices running macOS or browsers like Trident/7.0 showed much lower fraud rates, generally under 3%. Windows and iOS devices fell in between, with rates ranging from 6% to 8%.

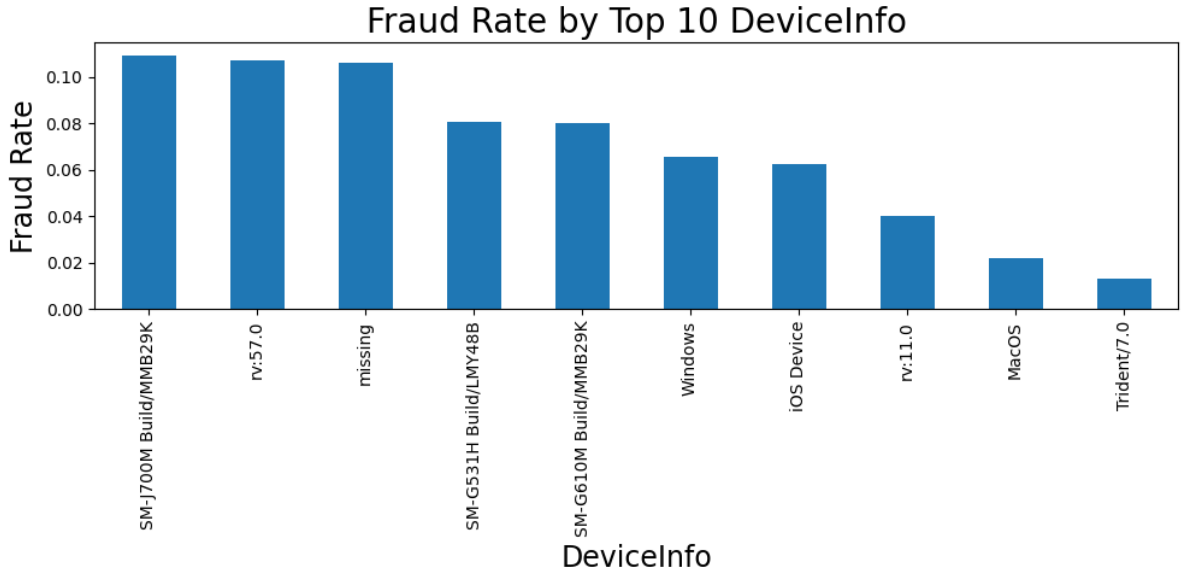

Figure 11: Fraud Rate by Top 10 DeviceInfo Values

These insights support several feature engineering opportunities that can directly applied into model development. For example, a binary variable such as is_disposable_email could be used to flag risky or missing email domains, while is_mobile_device could capture elevated risk in mobile-based transactions. Grouping DeviceInfo into a device_risk_level feature or simplifying email domains using a variable like email_domain_grouped could also help reduce cardinality and noise, making it easier for machine learning models to learn from these inputs. Overall, the analysis demonstrates that identity-related fields contain meaningful behavioral patterns and contribute valuable information for fraud detection.

5. Methodology

The methodology for this project was designed to systematically evaluate a range of supervised machine learning approaches for transaction fraud detection, with a focus on handling imbalanced, high-dimensional, and partially anonymized data. The process can be divided into four main stages: data preprocessing, class imbalance handling, model selection and training, and interpretability analysis.

5.1 Data Preprocessing

The training data was created by merging train_transaction.csv and train_identity.csv on TransactionID, combining payment and identity/device information. Missing categorical values were replaced with "missing", while numerical values were left as NaN for model handling. Categorical features (e.g., ProductCD, card4, DeviceType) were label-encoded.

Time-based variables such as hour_of_day were engineered from TransactionDT. An 80/20 stratified split maintained class proportions in training and validation sets.

Strengths: Preserves full data while ensuring fair evaluation across imbalanced classes.
Weaknesses: Retains potential noise from less-informative features.

5.2 Handling Class Imbalance

As mentioned in the earlier section, fraudulent transactions represented only about 3.5% of the dataset, making class imbalance a critical issue. To address this, class weighting and focal loss (for neural networks) were used instead of oversampling or synthetic data generation methods like SMOTE.

For tree-based models, the scale_pos_weight parameter was set to the ratio of majority to minority class samples, encouraging the model to give more attention to fraud cases without altering the dataset distribution.

Strengths: Preserves the integrity of real-world transaction patterns and avoids synthetic data artifacts.

Weaknesses: May not fully resolve imbalance if the decision boundary between classes is highly complex.

5.3 Model Selection and Training

Five models were selected based on their suitability for structured, imbalanced classification problems and their diversity in algorithmic approach:

- **Logistic Regression -** Serves as a simple, interpretable baseline.
- **Random Forest -** A robust, non-parametric ensemble method that handles non-linear feature interactions well.
- **XGBoost -** Gradient boosting algorithm known for strong performance on tabular datasets and effective handling of imbalance via scale_pos_weight.
- **Artificial Neural Network (ANN) -** Shallow architecture to benchmark basic deep learning performance.
- **Deep Neural Network (DNN) -** Deeper architecture with batch normalization, dropout, and focal loss to better capture complex relationships.

Hyperparameters for each model were tuned using prior literature guidance and exploratory testing. Early stopping and dropout regularization were used in neural networks to mitigate overfitting.

Strengths: Covers a range from interpretable linear models to high-capacity deep learning models, enabling comprehensive performance comparison.

Weaknesses: Neural networks require more computational resources and are harder to interpret without additional tools.

6. Experiments

This section outlines the experimental design used to evaluate the performance of multiple machine learning models for transaction fraud detection. The experiments were structured to ensure fair comparisons, address the class imbalance in the dataset, and assess both predictive accuracy and practical applicability.

6.1 Experimental Objectives

The experiments were designed to:

- Compare a range of supervised machine learning algorithms, from simpler baseline models to advanced ensemble and deep learning approaches.
- Assess the impact of class imbalance handling techniques on model performance.
- Evaluate the benefit of feature engineering, particularly time-based, geographic, and identity-based variables.
- Ensure that the evaluation framework reflects real-world fraud detection priorities, emphasizing recall and precision over raw accuracy.

6.2 Models Evaluated

6.2.1 Logistic Regression (Baseline): Selected as a transparent, interpretable benchmark model. It provides a point of reference for evaluating improvements from more complex approaches.

- **Strengths:** Simple, fast to train, easy to interpret, and resistant to overfitting in high-dimensional settings when regularized.
- **Weaknesses:** Limited ability to capture nonlinear relationships; performance can be constrained without extensive feature engineering.

6.2.2 Random Forest: Chosen for its robustness to noisy features and ability to model nonlinear interactions. Provides feature importance scores for interpretability.

- **Strengths:** Handles mixed data types well, less prone to overfitting compared to single decision trees, interpretable via feature importance.
- **Weaknesses:** Can be slower for very large datasets; may still struggle with extreme class imbalance without weighting or resampling.

6.2.3 XGBoost: A gradient boosting algorithm widely recognized for strong performance in tabular datasets. Natively handles missing values and supports class weighting.

- **Strengths:** Highly accurate, efficient, and effective with imbalanced datasets; strong handling of feature interactions.
- **Weaknesses:** More sensitive to hyperparameters; can overfit if not tuned carefully; less interpretable without additional tools such as SHAP.

6.2.4 Artificial Neural Network (ANN) – Shallow 2-layer: Serves as a simpler neural network baseline to compare against deeper architectures.

- **Strengths:** Achieves high precision on fraud predictions, making it suitable when minimizing false positives is critical.
- **Weaknesses:** Lower recall compared to tree-based models; may miss a substantial portion of fraud cases without further tuning.

6.2.5 Deep Neural Network (DNN) – Deeper 3+ layers: Designed to capture complex, high-dimensional feature interactions; uses focal loss to address class imbalance more effectively.

- **Strengths:** Balanced recall and precision; robust against overfitting with dropout and batch normalization; suitable for production environments requiring consistent performance.
- **Weaknesses:** Higher computational cost; less interpretable than tree-based models.

6.2.6 XGBoost + Deep Neural Network (Ensemble): Combines the structured feature learning of XGBoost with the high-dimensional representation capability of a fully connected DNN.

- **Strengths:** Leverages complementary strengths of gradient boosting and deep learning; can model complex nonlinearities and high-order feature interactions.
- **Weaknesses:** Higher computational cost; increased complexity may reduce interpretability; requires careful integration to prevent overfitting.

6.3 Evaluation Framework

Models were trained and validated using an 80/20 stratified split, preserving the fraud-to-non-fraud ratio in both sets. Performance was assessed using:

- ROC-AUC for overall discriminative ability.
- Precision-Recall AUC to better reflect imbalanced classification performance.
- Recall and Precision at fixed thresholds to simulate real-world operational cut-offs.
- F1-score to balance precision and recall.

6.4 Comparative Analysis Plan

Each model was first evaluated in its baseline form, followed by experiments incorporating class imbalance handling and engineered features. The final comparison will include ROC and PR curves for all models to visualize differences in discriminative performance. Feature importance analysis (via SHAP values for XGBoost and built-in measures for Random Forest) will be used to provide interpretability and insights into model decision-making.

7. Results

7.1 Overall Performance Comparison

This section presents a comparative evaluation of all seven machine learning models developed for transaction fraud detection: Logistic Regression (LR), Random Forest (RF), XGBoost (XGB), Enhanced XGBoost (XGB-Adv), Artificial Neural Network (ANN), Deep Neural Network (DNN), and the XGBoost + DNN Ensemble. Models are compared using ROC-AUC, PR-AUC, accuracy, recall, precision, and F1-score, with a focus on imbalanced data performance.

The table below summarizes the validation performance of all models. Metrics include ROC-AUC and PR-AUC for ranking capability, as well as accuracy, precision, recall, and F1-score for fraud detection effectiveness.

Model	**Accuracy**	**Recall**	**Precision**	**F1-Score**	**ROC-AUC**	**PR-AUC**
Logistic Regression (LR)	77%	0.69	0.10	0.18	0.7959	0.1772
Random Forest (RF)	94%	0.68	0.33	0.45	0.9122	0.5856
XGBoost (XGB)	89%	0.81	0.22	0.35	0.9276	0.6187
Enhanced XGBoost (XGB-Adv)	95%	0.83	0.42	0.55	0.9628	0.7676
Artificial Neural Network (ANN)	98%	0.43	0.91	0.59	0.9160	0.6403
Deep Neural Network (DNN)	98%	0.51	0.82	0.63	0.9182	0.6582
XGB + DNN Ensemble	98%	0.69	0.78	0.74	0.9638	0.7897

Table 1: Performance comparison across all evaluated models

The results show a clear performance gap between baseline models and advanced approaches. Logistic Regression suffers from low precision (0.10) and F1-score (0.18), making it unsuitable for fraud detection. Random Forest improves on these metrics, reaching an F1-score of 0.45, but still falls behind gradient boosting methods in capturing complex relationships in the data. Standard XGBoost raises recall to 0.81, boosting fraud capture rates, but precision (0.22) remains low, leading to a higher false-positive rate.

Enhanced XGBoost demonstrates the most significant improvement among single models, delivering the highest recall (0.83) and substantially better precision (0.42) than standard XGBoost. This balance results in a higher F1-score (0.55) and strong PR-AUC (0.7676), reflecting better performance in the imbalanced fraud detection setting. The improvement can be attributed to careful hyperparameter tuning and regularization, which reduce overfitting while improving fraud detection sensitivity. For use cases prioritizing maximum fraud capture with acceptable false positives, Enhanced XGBoost is a strong candidate.

The XGB + DNN Ensemble outperforms all other models in overall balance, achieving the highest F1-score (0.74) and PR-AUC (0.7897), alongside a high recall (0.69) and precision (0.78). By combining Enhanced XGBoost's structured feature learning with the DNN's deep representation capability, the ensemble reduces weaknesses present in each standalone model. This makes it well-suited for real-world usage, where both detecting fraud and minimizing false positives are critical for efficiency and customer trust.

7.2 External Benchmark Comparison

To further assess the competitiveness of the proposed models, the MRP Enhanced XGBoost and the MRP XGBoost + DNN Ensemble was compared against the OLightGBM model developed by Taha and Malebary [16], which serves as a strong published benchmark for fraud detection in similar research areas. The comparison focuses on the ROC-AUC metric, which provides an overall measure of a model's discriminative ability across all classification thresholds.

Model	**ROC-AUC**
Enhanced XGBoost (XGB-Adv)	0.9628
XGB + DNN Ensemble	0.9638
OLightGBM (Taha & Malebary)	0.9288

Table 2: ROC-AUC comparison between the proposed models and the OLightGBM benchmark

Both proposed models outperform the OLightGBM benchmark in terms of ROC-AUC, demonstrating their competitive advantage in fraud detection tasks. The XGB + DNN Ensemble achieved the highest ROC-AUC score of 0.9638, surpassing the Enhanced XGBoost model at 0.9628, and both showing a notable improvement over the OLightGBM score of 0.9288. These results highlight the robustness of the proposed approaches and their potential to deliver more reliable fraud detection in real-world industry.

7.3 ROC and Precision–Recall Curve Analysis

This section compares four models to evaluate how the proposed approaches perform against strong baselines. Random Forest (RF) is included as a representative of traditional machine learning methods, while Deep Neural Network (DNN) represents deep learning approaches. These are compared with the Enhanced XGBoost (XGB-Adv) and the XGB + DNN Ensemble, which combine boosting and neural networks. This comparison highlights the improvements of the proposed methods over both classical and deep learning models.

7.3.1 ROC Curve

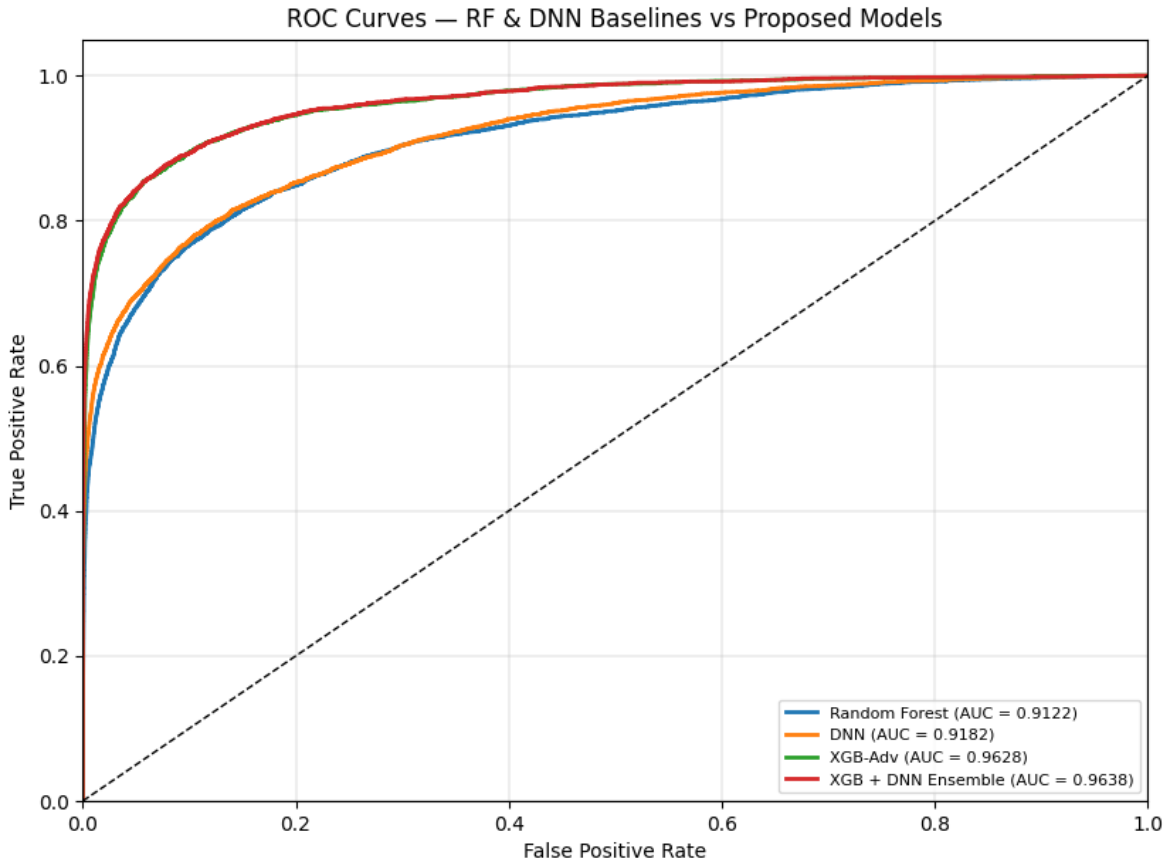

Figure 12: ROC Curves of Baseline and Proposed Models

Figure 12 above compares the Receiver Operating Characteristic (ROC) curves of the two baseline models, Random Forest (RF) and Deep Neural Network (DNN), against the proposed Enhanced XGBoost (XGB-Adv) and XGB + DNN Ensemble models. The ROC curves show that both proposed models achieve superior discriminative ability compared to the baselines.

The XGB + DNN Ensemble achieved the highest AUC (0.9638), followed closely by XGB-Adv (0.9628), indicating that both proposed models are able to distinguish between positive and negative cases with high reliability. Among the baselines, the DNN slightly outperformed the Random Forest, achieving an AUC of 0.9182 versus 0.9122. The steep initial slope and proximity of the proposed models' curves to the top-left corner of the plot reflect their strong classification performance across a wide range of thresholds.

7.3.2 Precision–Recall Curve

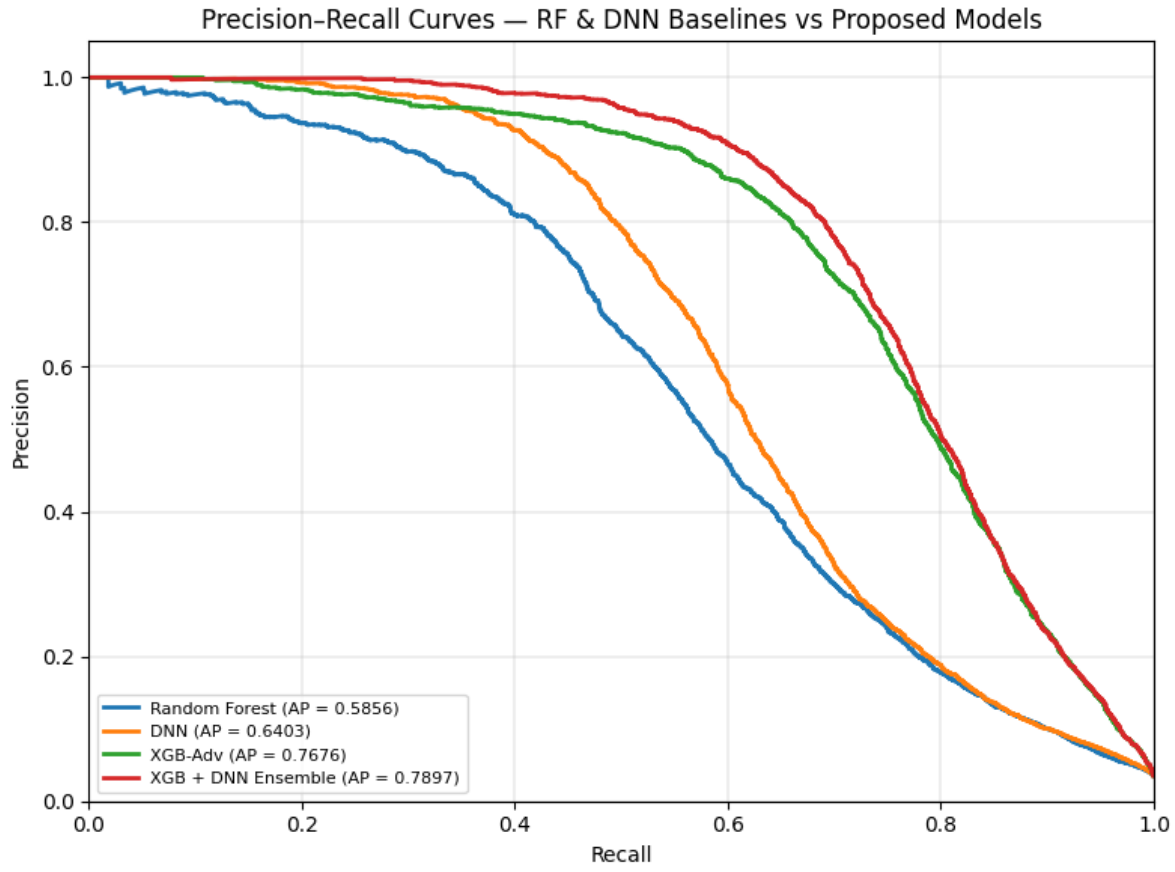

Figure 13: Precision–Recall Curves of Baseline and Proposed Models

Figure 13 presents the Precision–Recall (PR) curves for the same four models. Since PR curves are more informative for imbalanced datasets, they provide an important perspective on how models balance sensitivity and specificity in the presence of class imbalance.

The XGB + DNN Ensemble achieved the highest Average Precision (AP = 0.7897), followed by XGB-Adv (AP = 0.7676). This demonstrates that the proposed models maintain higher precision at varying recall levels, which is particularly important when false positives are costly. Among the baselines, the DNN again outperformed the Random Forest (AP = 0.6403 vs. 0.5856), consistent with the ROC analysis. The proposed models exhibit a noticeably slower decline in precision as recall increases, suggesting superior handling of the positive class and more stable performance under different decision thresholds.

8. Model Interpretation and Deployment Insights

8.1 Interpretation of SHAP Analysis - XGBoost Classifier

To better understand which features influence the XGBoost model's predictions for fraud detection, SHAP (SHapley Additive exPlanations) values were computed. SHAP helps explain how each feature contributes to the model's decisions, both overall and for individual predictions, making it a valuable tool for understanding complex models.

8.1.1 Global Feature Importance

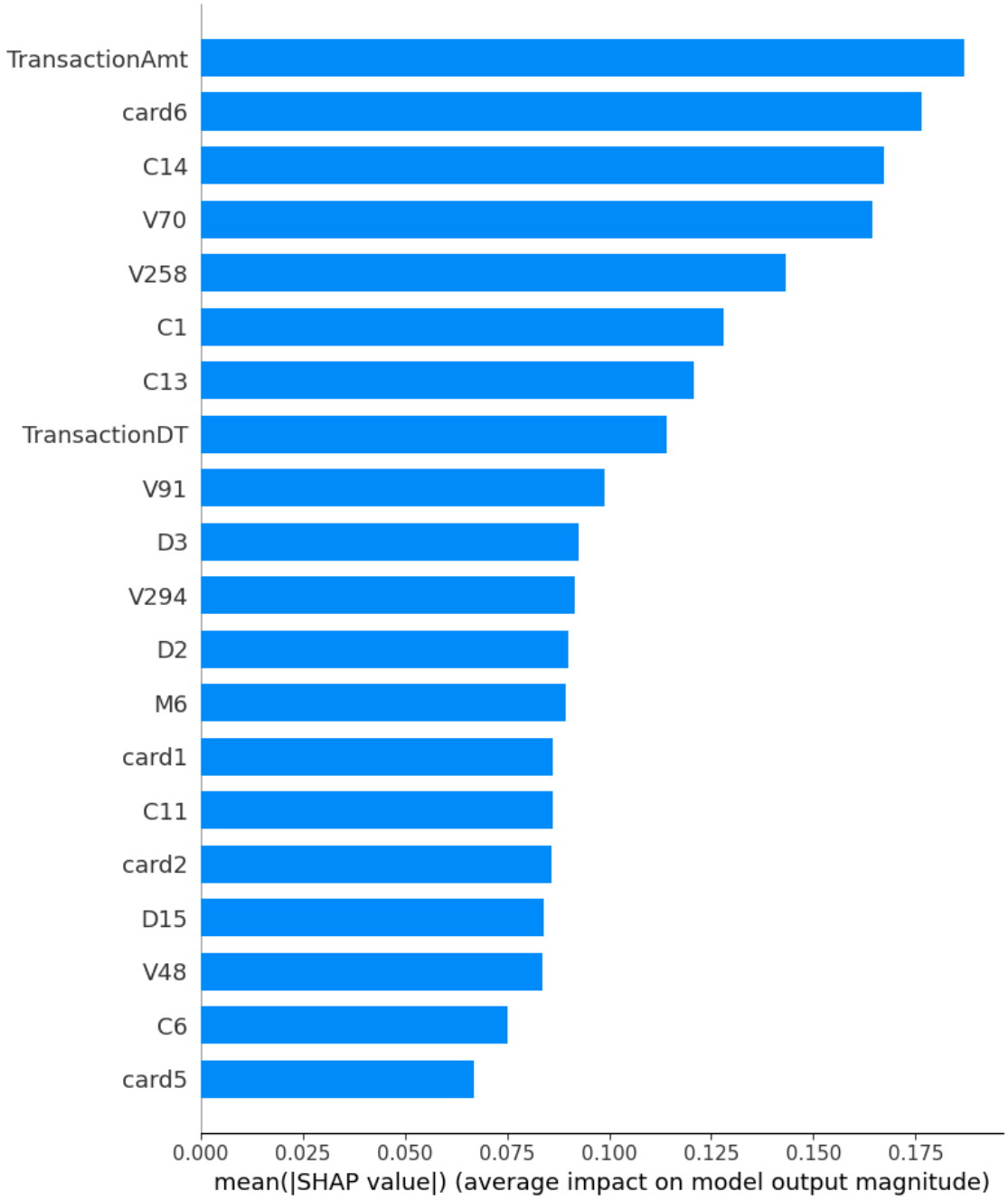

Figure 14: Global Feature Importance - Bar Plot

The SHAP bar plot ranks features based on their average contribution to the model's output across all validation samples. From the plot, the most influential features in predicting fraud include:

- **TransactionAmt** - The transaction amount is the strongest predictor. Larger or unusual amounts tend to raise the model's fraud probability.
- **card6** - Likely reflects card type (e.g., debit, credit, etc.). Certain types may be more associated with fraudulent behavior.
- **C14, C1, C13** - These anonymized features (often linked to customer ID, address match, or transaction count) show strong separation power.
- **V70, V258, V91, V294** - These V* engineered features carry behavioral or device-related information.
- **TransactionDT** – Represents the time of transaction. Time-based patterns can signal fraud trends (e.g., night-time activity).
- **D features** – Often indicate time deltas between various customer activities and help catch suspicious timing anomalies.

Together, these features suggest that the model has learned a combination of transaction size, card/device identity, and temporal behavior to distinguish between fraud and non-fraud transactions.

8.1.2 Local Explanation

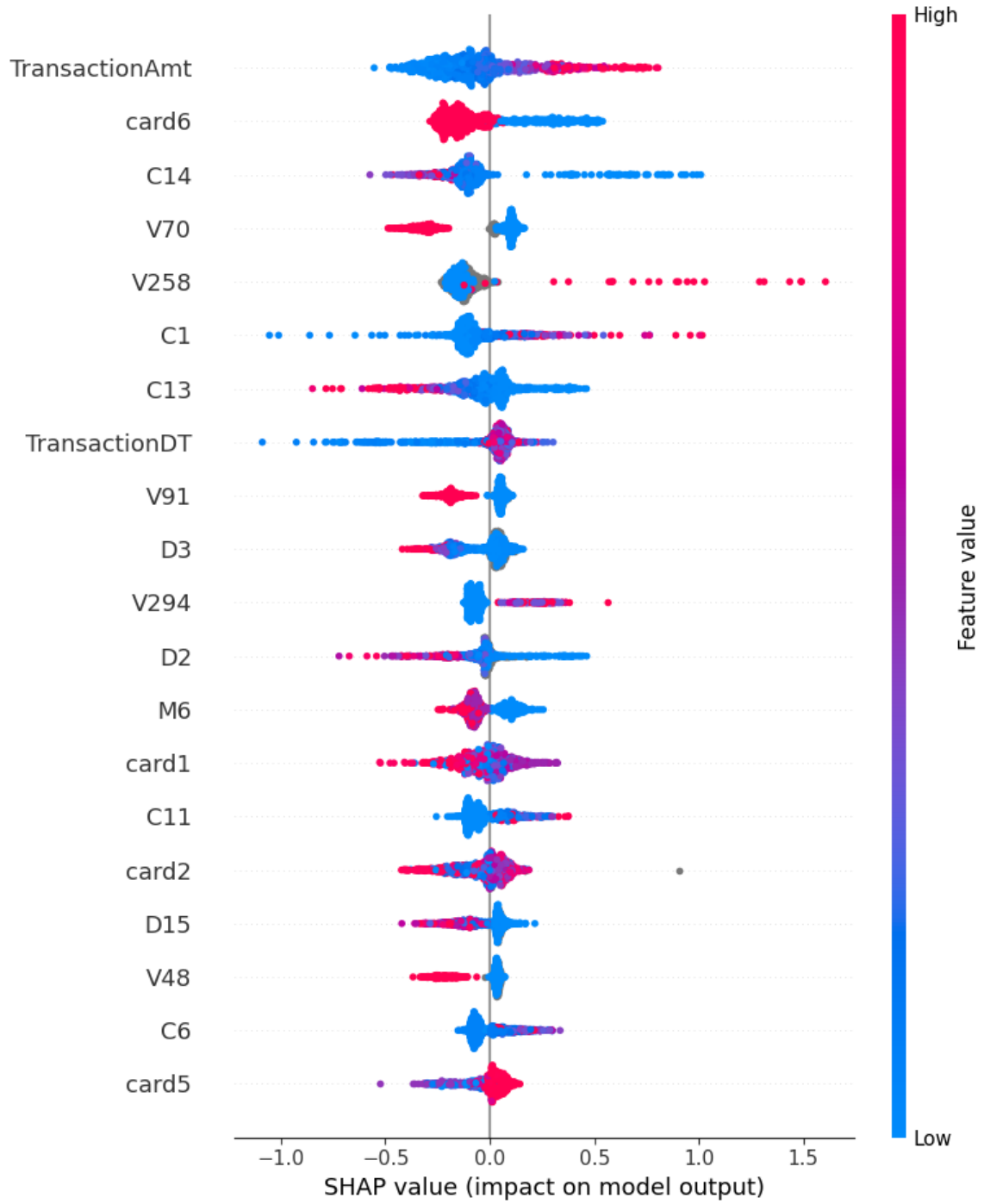

Figure 15: Local Explanation - Beeswarm Plot

The SHAP beeswarm plot gives a closer look at how individual feature values influence the model's decisions for each transaction.

- When a point appears on the right side of the plot (positive SHAP values), it means that particular feature value pushes the model toward predicting fraud. Points on the left (negative values) push the model toward predicting non-fraud.
- For example, higher values of **TransactionAmt** are usually linked to a greater chance of fraud. This is shown by red points clustering on the right side of the plot. In contrast, smaller transaction amounts tend to reduce the model's fraud score.
- Features like `card6` and `C14` have different impacts depending on their specific values, with some categories or ranges increasing the risk of fraud more than others.

This kind of insight helps not only with debugging the model but also with building trust, especially for stakeholders who want to understand the reasoning behind flagged transactions.

Overall, the SHAP analysis reveals that the XGBoost model relies on a meaningful combination of transaction metadata, cardholder identifiers, and temporal features to make its predictions. This transparency ensures that fraud analysts can trust and act upon the model's predictions with confidence.

8.2 Feature Importance Analysis - Random Forest

To better understand how the Random Forest model makes its decisions, feature importance scores from the built-in `"feature_importances_attribute"` were examined. These scores indicate how much each feature contributes to the model's predictions by measuring its impact on reducing uncertainty across all the decision trees.

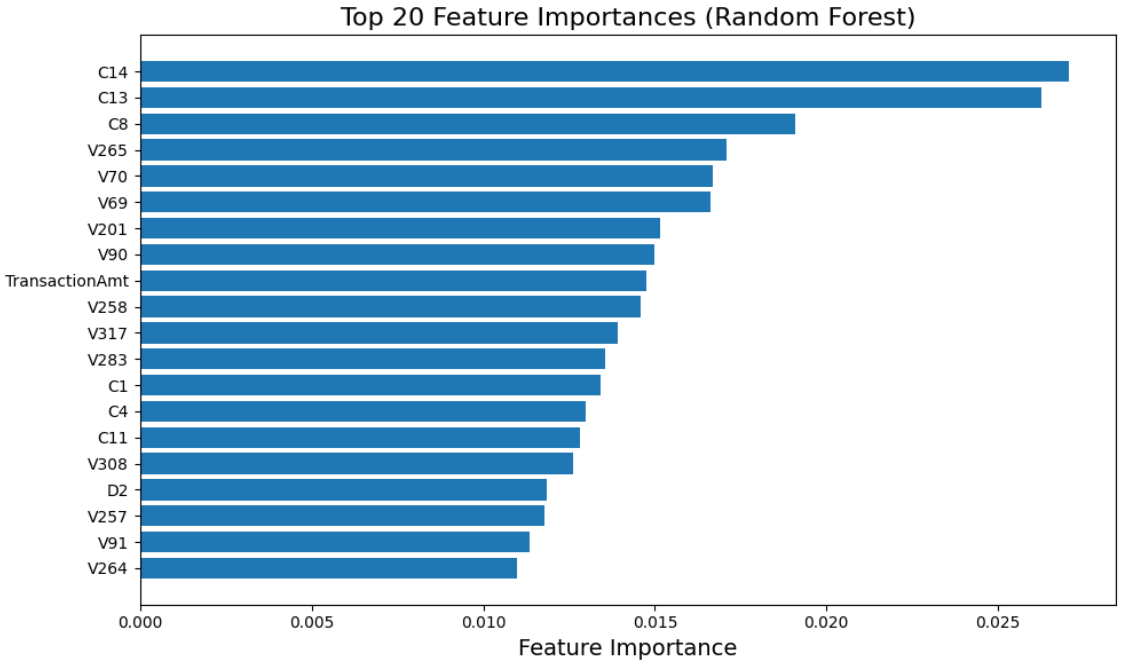

Figure 16: Random Forest - Top 20 Feature Importances

Key Findings:

- Features like **C14**, **C13**, and **C8** had the highest influence on the model's predictions. While these are anonymized, they likely represent patterns tied to customer behavior or transaction structure, making them strong indicators of fraud.
- Variables such as **TransactionAmt** and engineered features like **V258**, **V201**, and **D2** also ranked highly, which aligns with common fraud patterns involving unusual amounts and suspicious timing.
- What stands out is the broad distribution of importance across more than 20 features. This suggests that the Random Forest model draws insights from many different signals, rather than depending on just a few key predictors.

Overall, the feature importance plot provides a quick view of which inputs the Random Forest model relies on most when detecting fraud. While it doesn't offer detailed explanations like SHAP, it's still useful for identifying key features, understanding the model, and communicating results clearly. Features such as **C14**, **V258**, and **TransactionAmt** contribute the most, offering insights that can guide further feature engineering and support ongoing improvements to the fraud detection systems.

8.3 Insights and Recommendations for Real-World De-ployment

Applying the XGBoost and XGBoost + DNN ensemble models into production requires more than just strong validation metrics. To ensure practical success in real-world fraud detection, the following recommendations are worth to consider:

- **Threshold Optimization:** Adjust decision thresholds based on business needs, balancing fraud detection with acceptable false positive rates.
- **Model Monitoring and Retraining:** Continuously track performance metrics and detect data changes. Retrain models regularly to adapt to evolving fraud tactics.
- **Explainability and Compliance:** Use feature importance (for XGBoost) and tools such as SHAP or LIME (for DNN) to provide transparency, ensuring regulatory and operational trust.
- **A/B Testing and ROI Measurement:** Any production launch should be implemented with controlled A/B testing to quantify impacts on fraud loss reduction, customer satisfaction, and operational workload.

By following these recommendations, organizations can maximize the value of machine learning-based fraud detection systems, ensuring not only strong detection performance, but also operational efficiency, scalability, and adaptability to evolving fraud tactics.

9. Conclusion

This project set out to explore how machine learning can be used to improve the detection of fraudulent transactions, testing both individual models and hybrid approaches. Several algorithms were evaluated, including Logistic Regression, Random Forest, XGBoost, Artificial Neural Networks, Deep Neural Networks, and an XGBoost + DNN ensemble. The results showed that Enhanced XGBoost and the XGBoost + DNN ensemble delivered the best overall performance. Both achieved strong scores across accuracy, recall, precision, F1-score, ROC-AUC, and PR-AUC, with the ensemble model excelling at balancing precision and recall, a key factor in catching fraud while minimizing false alarms.

Beyond raw performance, the analysis also looked at how the models make decisions. SHAP analysis for XGBoost and feature importance rankings for Random Forest revealed that transaction amount, certain card details, engineered behavioral variables, and time-based features play a major role in detecting fraud. Identity-based features such as device type and email domain were not added, as their inclusion was deemed unnecessary based on the dataset and problem context.

To address the challenge of class imbalance, class weighting was applied, ensuring that the models remained sensitive to fraud cases without overfitting to the majority class. Practical recommendations were outlined for putting these models into real-world use. This includes setting decision thresholds that match business needs, regularly monitoring and retraining the models to keep up with new fraud tactics, and maintaining transparency through explainability tools.

In the end, this research shows that combining the strengths of tree-based models with deep learning can lead to powerful, reliable, and interpretable fraud detection systems. While the results are promising, the next step would be to integrate the models into a live environment and continue refining them with real-time data to ensure they stay effective against ever-changing threats.

References

[1] Y.-T. Lei, C.-Q. Ma, Y.-S. Ren, X.-Q. Chen, S. Narayan, and A. N. Q. Huynh, "A distributed deep neural network model for credit card fraud detection," *Finance Research Letters*, vol. 58, p. 104547, 2023. https://doi.org/10.1016/j.frl.2023.104547

[2] D. Shah and L. K. Sharma, "Credit Card Fraud Detection using Decision Tree and Random Forest," *ITM Web of Conferences*, vol. 53, p. 2012, 2023. https://doi.org/10.1051/itmconf/20235302012

[3] N. S. Alfaiz and S. M. Fati, "Enhanced Credit Card Fraud Detection Model Using Machine Learning," *Electronics*, vol. 11, no. 4, p. 662, 2022. https://doi.org/10.3390/electronics11040662

[4] M. Alamri and M. Ykhlef, "Hybrid Undersampling and Oversampling for Handling Imbalanced Credit Card Data," *IEEE Access*, vol. 12, pp. 14050–14060, 2024. https://doi.org/10.1109/ACCESS.2024.3357091

[5] E. Ileberi, Y. Sun, and Z. Wang, "Performance Evaluation of Machine Learning Methods for Credit Card Fraud Detection Using SMOTE and AdaBoost," *IEEE Access*, vol. 9, pp. 165286–165294, 2021. https://doi.org/10.1109/ACCESS.2021.3134330

[6] M. M. Lahbiss and Y. Chtouki, "Credit Card Fraud Detection in Imbalanced Datasets: A Comparative Analysis of Machine Learning Techniques," *2024 International Conference on Computer and Applications (ICCA)*, pp. 1–6, 2024. https://doi.org/10.1109/ICCA62237.2024.10927865

[7] S. Lei, K. Xu, Y. Huang, and X. Sha, "An Xgboost based system for financial fraud detection," *E3S Web of Conferences*, vol. 214, p. 2042, 2020. https://doi.org/10.1051/e3sconf/202021402042

[8] Y. Lucas et al., "Multiple perspectives HMM-based feature engineering for credit card fraud detection," *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing*, pp. 1359–1361, 2019. https://doi.org/10.1145/3297280.3297586

[9] A. Correa Bahnsen, D. Aouada, A. Stojanovic, and B. Ottersten, "Feature engineering strategies for credit card fraud detection," *Expert Systems with Applications*, vol. 51, pp. 134–142, 2016. https://doi.org/10.1016/j.eswa.2015.12.030

[10] F. Carcillo et al., "Combining unsupervised and supervised learning in credit card fraud detection," *Information Sciences*, vol. 557, pp. 317–331, 2021. https://doi.org/10.1016/j.ins.2019.05.042

[11] M. A. K. Achakzai and J. Peng, "Detecting financial statement fraud using dynamic ensemble machine learning," *International Review of Financial Analysis*, vol. 89, p. 102827, 2023. https://doi.org/10.1016/j.irfa.2023.102827

[12] D. Jahnavi et al., "Robust Hybrid Machine Learning Model for Financial Fraud Detection in Credit Card Transactions," *2024 2nd International Conference on Intelligent Data Communication Technologies and Internet of Things (IDCIoT)*, pp. 680–686, 2024. https://doi.org/10.1109/IDCIoT59759.2024.10467340

[13] S. Chaurasia et al., "Analysis of Ensemble Machine Learning Models for Fraud Detection," *2024 International Conference on Intelligent Systems for Cybersecurity (ISCS)*, pp. 1–6, 2024. https://doi.org/10.1109/ISCS61804.2024.10581076

[14] A. Patel, M. Patel, and P. Patel, "Exploring Supervised Machine Learning Techniques for Detecting Credit Card Fraud: An Investigative Review," *ITM Web of Conferences*, vol. 65, p. 3006, 2024. https://doi.org/10.1051/itmconf/20246503006

[15] Jyoti, K. Bhardwaj, G. Garima, M. Kumar, R. Verma, and D. Kumar, "Machine Learning and Deep Learning for Credit Card Fraud Detection: A Comparative Analysis," *2024 International Conference on Artificial Intelligence and Emerging Technology (Global AI Summit)*, pp. 131–136, 2024. https://doi.org/10.1109/GlobalAISummit62156.2024.10947915

[16] A. A. Taha and S. J. Malebary, "An intelligent approach to credit card fraud detection using an optimized light gradient boosting machine," *IEEE Access*, vol. 8, pp. 25579–25587, 2020. https://doi.org/10.1109/ACCESS.2020.2971354

A Model Configurations & Interpretability Methods

This appendix documents the detailed configurations, architectures, and interpretability methods used in the experiments, enabling reproducibility without including performance metrics (which are reported in the Results section).

A.1 XGBoost Classifier

Preprocessing:

- Transaction and identity datasets merged on `TransactionID`.
- Categorical features label-encoded.
- Missing categorical values replaced with "missing", numerical NaNs left intact.

Hyperparameter	**Value**
Number of Estimators ($n_{estimators}$)	100
Maximum Depth (max_depth)	6
Learning Rate ($learning\_rate$)	0.1
Scale Positive Weight ($scale\_pos\_weight$)	27.58
Subsample Ratio ($subsample$)	0.8
Column Subsample Ratio ($colsample\_bytree$)	0.8
Random State ($random\_state$)	42

Table 3: Hyperparameters for the XGBoost Classifier

A.2 Enhanced XGBoost Classifier

Preprocessing:

- Same as XGBoost.

Hyperparameter	**Value**
Number of Estimators ($n_{estimators}$)	1200
Maximum Depth (max_depth)	9
Learning Rate ($learning\_rate$)	0.015
Scale Positive Weight ($scale\_pos\_weight$)	27.58
Subsample Ratio ($subsample$)	0.92
Column Subsample Ratio ($colsample\_bytree$)	0.92
Gamma (γ)	3
Minimum Child Weight (min_child_weight)	4
Regularization Alpha (reg_α)	4
Regularization Lambda (reg_λ)	8

Table 4: Hyperparameters for the Enhanced XGBoost Classifier

A.3 Random Forest Classifier

Preprocessing: Same as XGBoost.

Hyperparameter	Value
Number of Estimators ($n_{estimators}$)	100
Maximum Depth (max_depth)	15
Class Weight ($class\_weight$)	balanced
Number of Jobs (n_jobs)	-1
Random State ($random\_state$)	42

Table 5: Hyperparameters for the Random Forest Classifier

A.4 Logistic Regression

Preprocessing:

- Same as above, plus missing numerical values imputed with median.

Hyperparameter	Value
Solver ($solver$)	liblinear
Class Weight ($class\_weight$)	balanced
Maximum Iterations (max_iter)	200
Random State ($random\_state$)	42

Table 6: Hyperparameters for the Logistic Regression Model

A.5 Artificial Neural Network (ANN) – Shallow 2-layer

Architecture:

- Input: Scaled numerical + label-encoded categorical features.
- Dense(128), ReLU, Dropout(0.3)
- Dense(64), ReLU, Dropout(0.3)
- Dense(1), Sigmoid

Training Parameter	Value
Loss Function	Binary Cross-Entropy
Optimizer	Adam ($lr = 0.001$)
Batch Size	512
Epochs	20
Early Stopping	Yes (on validation loss)

Table 7: Training Parameters for the ANN

A.6 Deep Neural Network (DNN) – Deeper 3+ layers

Architecture:

- Input: Preprocessed numerical features.
- Dense(512), LeakyReLU, BatchNorm, Dropout(0.4)
- Dense(256), LeakyReLU, BatchNorm, Dropout(0.4)
- Dense(128), LeakyReLU, BatchNorm, Dropout(0.3)
- Dense(1), Sigmoid

Training Parameter	Value
Loss Function	Focal Loss ($\gamma = 2.0, \alpha = 0.25$)
Optimizer	Adam ($lr = 0.001$)
Batch Size	1024
Epochs	20
Probability Threshold	0.3

Table 8: Training Parameters for the DNN

A.7 XGBoost + DNN Ensemble

XGBoost Submodel

- Same preprocessing as Enhanced XGBoost.
- Tuned parameters identical to those in Table 4.
- Trained directly on raw (non-scaled) tabular features.

DNN Submodel

Architecture:

- Dense(1024) $\rightarrow$ BatchNorm $\rightarrow$ Dropout(0.4)
- Dense(512) $\rightarrow$ BatchNorm $\rightarrow$ Dropout(0.3)
- Dense(128) $\rightarrow$ BatchNorm $\rightarrow$ Dropout(0.2)
- Output: Dense(1, activation=`sigmoid`)

Training Parameter	Value
Loss Function	Binary Focal Loss ($\gamma = 2.0, \alpha = 0.25$)
Optimizer	Adam ($learning\_rate = 0.0005$)
Batch Size	Not specified
Early Stopping	Yes (on validation loss)
Input Features	Scaled numerical + encoded categorical
Class Weights	Applied

Table 9: Training Parameters for the DNN Submodel

Ensemble Strategy

- Weighted average of probabilities: $ensemble\_probs = 0.6 \times xgb\_probs + 0.4 \times dnn\_probs$
- Threshold optimized via precision-recall curve to maximize F1-score.

A.8 Model Interpretation Methods

XGBoost:

- SHAP analysis for global and local feature importance.
- Key features: `TransactionAmt`, `card6`, anonymized features `C14`, `C1`, `C13`, `V70`, `V258`, `V91`, `V294`, `TransactionDT`, and various D features.

Random Forest:

- Built-in `feature_importances_attribute` for top-20 most influential predictors.
- Key features: `C14`, `C13`, `C8`, `TransactionAmt`, and engineered features such as `V258`, `V201`, `D2`.

B Confusion Matrices

This appendix presents the confusion matrices for all evaluated models. Each table shows the number of transactions in each classification outcome: **True Negatives** (Actual Non-Fraud, Predicted Non-Fraud), **False Positives** (Actual Non-Fraud, Predicted Fraud), **False Negatives** (Actual Fraud, Predicted Non-Fraud), and **True Positives** (Actual Fraud, Predicted Fraud).

B.1 Logistic Regression (LR)

Confusion Matrix:

	Predicted: Non-Fraud	**Predicted: Fraud**
Actual: Non-Fraud	88,579	25,396
Actual: Fraud	1,296	2,837

Table 10: Logistic Regression - Confusion Matrix

B.2 Random Forest (RF)

Confusion Matrix:

	Predicted: Non-Fraud	**Predicted: Fraud**
Actual: Non-Fraud	108,314	5,661
Actual: Fraud	1,323	2,810

Table 11: Random Forest - Confusion Matrix

B.3 XGBoost (XGB)

Confusion Matrix:

	Predicted: Non-Fraud	**Predicted: Fraud**
Actual: Non-Fraud	102,368	11,607
Actual: Fraud	803	3,330

Table 12: XGBoost - Confusion Matrix

B.4 Artificial Neural Network (ANN)

Confusion Matrix:

	Predicted: Non-Fraud	**Predicted: Fraud**
Actual: Non-Fraud	113,801	174
Actual: Fraud	2,337	1,796

Table 13: Artificial Neural Network - Confusion Matrix

B.5 Deep Neural Network (DNN)

Confusion Matrix:

	Predicted: Non-Fraud	**Predicted: Fraud**
Actual: Non-Fraud	113,498	477
Actual: Fraud	2,016	2,117

Table 14: Deep Neural Network - Confusion Matrix

B.6 Enhanced XGBoost

Confusion Matrix:

	Predicted: Non-Fraud	**Predicted: Fraud**
Actual: Non-Fraud	109,145	4,830
Actual: Fraud	709	3,424

Table 15: Enhanced XGBoost - Confusion Matrix

B.7 XGBoost + DNN Ensemble

Confusion Matrix:

	Predicted: Non-Fraud	**Predicted: Fraud**
Actual: Non-Fraud	113,155	820
Actual: Fraud	1,243	2,890

Table 16: XGBoost + DNN Ensemble - Confusion Matrix

C GitHub Repository

All reports, codes, and visualizations for this MRP Project are available at the following GitHub repository:
Link: `https://github.com/nguyenduyanhluong/TMU-MRP-2025`