Toronto Metropolitan University

Master of Science in Data Science and Analytics
MRP - Literature Review and Exploratory Data Analysis

Optimizing Supervised Machine Learning for Enhanced Transaction Fraud Detection Accuracy

Submitted to:

MRP Supervisor – Dr. Shengkun Xie
MRP Second Reader – Dr. Ceni Babaoglu

Submitted by:

Nguyen Duy Anh Luong (Student ID – 500968520)

June 16, 2025

Abstract

Transaction fraud remains a serious concern for banks, businesses, and consumers, especially as online transactions continue to rise. In this study, I first reviewed several papers (listed in the references) that explore supervised machine learning techniques for detecting fraud. The review focuses on practical strategies for model selection, addressing data imbalance, and improving predictive performance. It also discusses some common challenges such as the lack of fraud cases, anonymous features, and the ongoing trade-off between limiting false positives and increasing fraud detection in real-time systems. Many of the reviewed studies highlight the effectiveness of ensemble models, neural networks, and data sampling techniques in improving overall detection accuracy.

In the second part, I conducted an exploratory data analysis (EDA) on the IEEE-CIS Fraud Detection dataset. Using visual and statistical tools, the EDA revealed patterns in missing data, feature relationships, and the timing of fraudulent transactions. Additional findings include the distribution of transaction amounts by fraud class, the fraud rate across different product categories, and how certain high-value transactions are more closely linked to fraud. One major finding was the significant class imbalance, as fraudulent cases make up only 3.5% of the data, highlighting the importance of using appropriate techniques in future modeling stages.

Literature Review

This literature review focuses on supervised machine learning approaches for transaction fraud detection, especially in real-world scenarios with imbalanced and anonymized data. As online transactions become part of daily life, whether for shopping, paying bills, or transferring money, fraud has become a growing concern for businesses and consumers. Traditional rule-based systems are no longer effective against evolving fraud tactics, leading to increased interest in machine learning methods that learn from historical data.

For my Major Research Project (MRP), I use the IEEE-CIS Fraud Detection dataset, released by Vesta through a Kaggle competition. It includes masked transaction and identity features, adding complexity to preprocessing and modeling. This review explores insights from fifteen peer-reviewed journal and conference papers, examining techniques for handling class imbalance, feature engineering, model selection, and the use of identity-based variables to improve fraud detection performance.

1. Supervised Learning Models in Fraud Detection

I began by exploring 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, the models can make informed predictions on new, unseen transactions.

One of the most interesting approaches I reviewed is the Distributed Deep Neural Network (DDNN) model proposed by Lei et al. (2023), notable for its focus on both accuracy and user privacy [1]. Instead of centralizing all transaction data, their method allows financial institutions to train local models on their own data while only sharing model parameters with a central server. This approach not only protects user privacy but also lowers 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, I explored more interpretable models, such as decision trees and random forests. 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, I examined a large-scale study by Alfaiz and Fati (2022), which evaluated 66 combinations of nine machine learning algorithms and nineteen resampling techniques [3]. Their approach was particularly valuable, as it used a real-world dataset and systematically compared each combination. 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 helped me better understand the strengths and limitations of different supervised learning models. Deep learning approaches like DDNN offer strong performance, particularly when privacy and scalability are important. On the other hand, interpretable models such as random forests and CatBoost can also be highly effective when supported by thoughtful preprocessing and well-chosen data balancing techniques.

2. Imbalanced Data: Core Challenge and Solutions

As I looked deeper into supervised learning models, I quickly realized that class imbalance is 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 leads models to favor legitimate transactions and miss fraud cases. To build a system that works well, it's important to address this imbalance so the model can catch those rare fraud cases without mistakenly flagging too many legitimate ones.

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, their proposed method achieved an F1-score of 85.2% and successfully addressed common oversampling issues like 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, Lahbiss and Chtouki (2024), in their study Credit Card Fraud Detection in Imbalanced Datasets: A Comparative Analysis of Machine Learning Techniques, evaluated traditional, ensemble, and deep learning models using SMOTE and SMOTE-ENN [6]. Their findings showed that Random Forest 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 suggest that combining resampling strategies with strong classifiers can significantly improve fraud detection performance.

Together, these studies make it clear that there isn't a one-size-fits-all solution. Instead, mixing resampling strategies with different models like Random Forest or LSTM seems to be one of the most effective ways to deal with class imbalance in fraud detection.

3. Feature Engineering and Identity-Based Enhancements

After exploring the impact of class imbalance, I started looking into feature engineering. Even the best algorithms won't perform well without meaningful input features. In fraud detection, basic variables like transaction amount or card type offer limited value on their own. The true improvements come from transforming features and adding identity-related data that better reflects user behavior.

One study that helped me understand the value of feature engineering was by Lei et al. (2020), who applied XGBoost to the IEEE-CIS dataset [7]. The authors emphasized careful data cleaning, missing value filling, consistent label encoding, and feature elimination. By combining transaction data with identity fields such as device type and email domain, they significantly improved the model's performance, 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 using Hidden Markov Models (HMMs) [8]. Instead of relying on standard transaction aggregates, their approach modeled sequences using eight combinations based on three binary perspectives. These included whether the sequence was linked to a cardholder or a terminal, whether it was built from transaction amounts or time elapsed, and whether the history was genuine or fraudulent. When the multi-perspective HMM-based features were added to a Random Forest classifier, they led to a 15.1% boost in Precision-Recall AUC. Finally, Bahnsen et al. (2016) proposed a strategy that combined transaction aggregation with periodic behavior modeling [9]. They introduced features based on customer spending patterns and used the von Mises distribution to model typical transaction times. These features led to a 13% increase in savings, reflecting how much more money the system was able to recover or prevent from being lost 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.

4. Ensemble Learning and Hybrid Models

After examining how individual models perform and how feature engineering can improve them, I turned my attention to ensemble learning to explore whether combining models could lead to better results. Ensemble methods combine multiple classifiers to boost accuracy, reduce bias, and better capture the complexity of fraud patterns, which often vary across users and change over time.

One study that stood out was by Carcillo et al. (2021), who introduced a hybrid approach combining unsupervised and supervised learning [10]. The authors first used unsupervised models to generate outlier scores, which were then passed into a supervised classifier to make the final decision. Their system evaluated fraud at multiple levels, such as globally across the dataset, locally for each cardholder, and within clusters of similar customers, allowing the model to catch different types of fraud more effectively. Similarly, Achakzai and Peng (2023) proposed a dynamic ensemble selection method that selects the most effective classifiers for each transaction based on local competence [11]. This 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]. Their method achieved 98.1% accuracy and showed strong performance in both sensitivity and flexibility. By combining logistic regression with the decision-making structure of decision trees, they developed a model that can adapt to evolving fraud tactics. In a related study, Chaurasia et al. (2024) compared several ensemble machine learning models, including Decision Tree, Random Forest, XGBoost, CatBoost, and Gradient Boosting [13]. Using both balanced and imbalanced versions of the European cardholder dataset, they found that XGBoost delivered the best performance in terms of F1-score and recall, making it especially effective at identifying rare fraud cases.

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

5. Comparative Reviews

In addition to studying individual models and techniques, I also looked at broader research that compares multiple machine learning approaches for fraud detection. These comparative reviews gave me a clearer picture of which methods tend to perform well across different scenarios and helped inform my understanding of model selection in practice.

Patel et al. (2024) conducted a comparative review of 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]. Their 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%. While ANN delivered the top performance, the study also highlighted that other models, such as logistic regression and Naive Bayes, still performed well across multiple evaluation metrics.

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

Together, these studies show that while deep learning models often lead in accuracy, traditional models like logistic regression still offer strong, reliable performance. This highlights the importance of choosing models based on both effectiveness and practicality in real-world fraud detection.

6. Conclusion: 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 well 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 together, 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, I will compare several machine learning models, including logistic regression, random forest, artificial neural networks, deep neural networks, and XGBoost, to identify the most effective approach. By applying feature engineering, resampling techniques, and ensemble strategies, the system aims to improve fraud detection accuracy while reducing false positives. The final goal is to build a reliable, scalable, and interpretable model that supports better fraud prevention in real-world financial applications.

Data Description - Exploratory Data Analysis (EDA)

In this section, I explored patterns in fraudulent behavior using a mix of visualizations and statistical analysis. My goal was to find meaningful insights that can guide how I build my models and improve fraud detection accuracy. Each analysis ties back to the research questions I defined earlier, and I focused on identifying trends, anomalies, or potential feature transformations that could help the model detect fraud more effectively.

Data Source

For this project, I used the IEEE-CIS Fraud Detection dataset, 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 supports the development of machine learning models for fraud detection.
Source: `https://www.kaggle.com/competitions/ieee-fraud-detection/data`

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 without fraud labels.

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

Data Constraints and Preprocessing

Before starting the analysis, I had to clean the raw transaction and identity datasets to make them easier to work with. Some column names had hyphens and extra spaces, so I replaced hyphens with underscores and removed any leading or trailing whitespace to avoid issues in Python. Missing values were also a big part of the cleanup. For columns with text or categorical data, I replaced missing values with the word ”missing” so it would be clear later on that something was originally not provided. For numerical columns, I kept missing values as None, which can be handled more carefully during the analysis or modeling phase. This preprocessing step was done 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 later steps in the EDA.

Descriptive Statistics

Before performing visual exploration, I reviewed basic statistics 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 in 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 upcoming exploratory data analysis (EDA).

1. Class Imbalance and Baseline Performance

When looking at the dataset for the first time, one of the most obvious things that stands out is the huge 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. That means fraud is not only rare but also dangerous and costly when it happens.

To visualize this, I created a simple bar chart showing the counts for each class. As expected, the legitimate transactions completely dominate the chart. I also added the percentages directly above the bars to make the imbalance more obvious: 96.5% legitimate vs. 3.5% fraudulent.

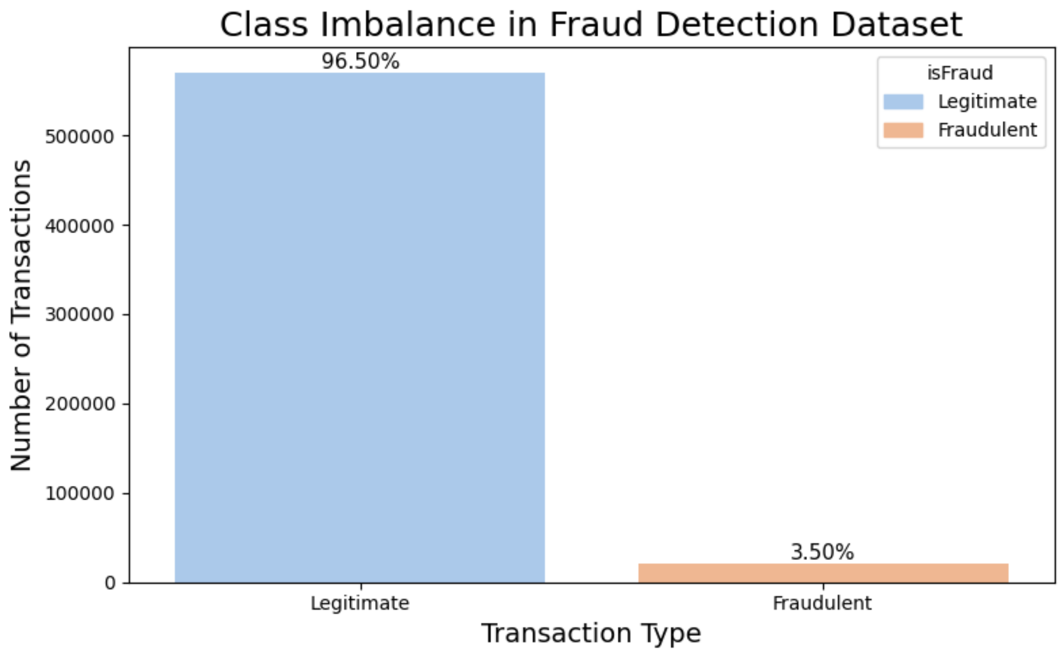

Figure 1: Fraud Distribution by Transaction Type

To show why this is a big deal for machine learning, I tested a very basic model: a dummy classifier that always predicts the majority class (not fraud). As we can see, this model achieved 96.5% accuracy, which sounds great at first, but it completely missed all the fraud cases. The precision, recall, and F1-score were all 0.000, which means that it's completely useless for catching fraud.

I also wanted to put this into a real-world perspective. If we assume that each missed fraud costs the company around $500, then missing 20,663 fraud cases in this dataset could lead to a loss of over $10 million. This is why high accuracy alone isn't enough when dealing with fraud detection.

This EDA confirms that we cannot treat this like a normal classification problem. Fraud detection needs special attention because of how rare and high-impact those fraud cases are. Hence, we need to use techniques like resampling (e.g., SMOTE), class weights, or ensemble models like XGBoost that are better at handling imbalanced data.

2. Transaction Amount and Product Category Patterns

We started by visualizing the distribution of TransactionAmt for fraudulent and legitimate transactions. While both types of transactions mostly occur at low amounts, we saw that fraudulent transactions tend to spread more widely across the amount range, especially between \$100 and \$500. The KDE (Figure 2) plot helped us see this more clearly.

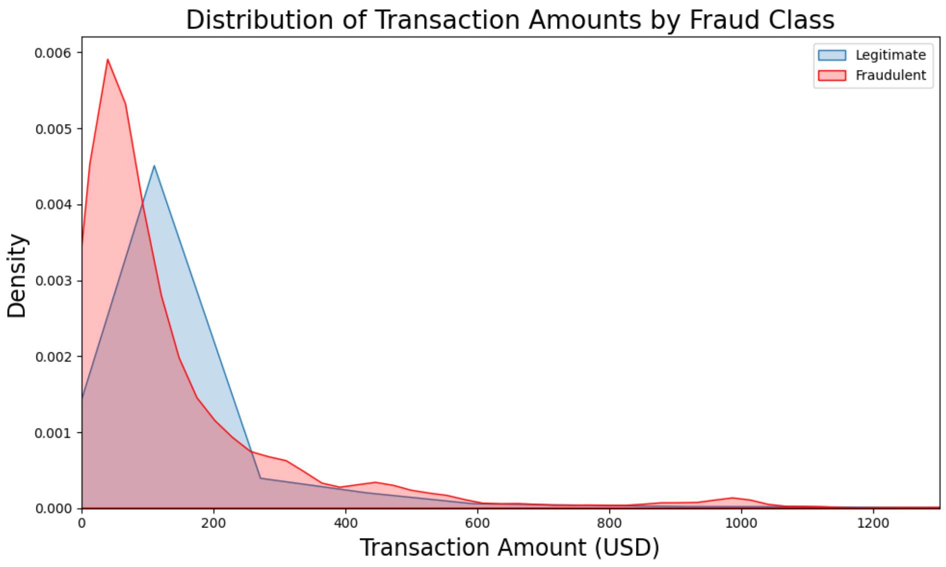

Figure 2: Distribution of Transaction Amounts by Fraud Class

The boxplot (Figure 3) also supports this as fraudulent transactions had more high-value outliers than legitimate ones. This supports the idea that some fraudsters attempt higher-value purchases, possibly hoping to get away with a big transaction.

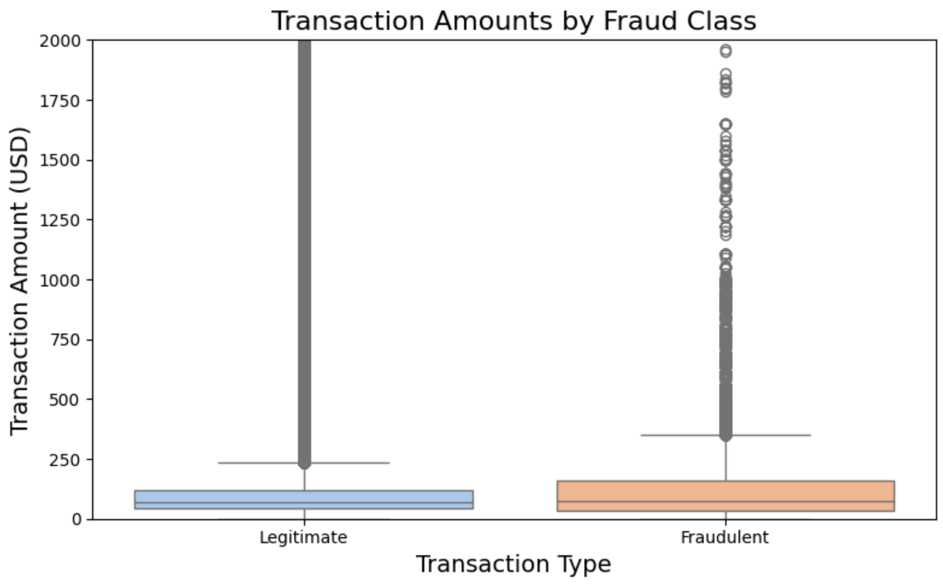

Figure 3: Transaction Amounts by Fraud Class (Boxplot)

Next, we looked at ProductCD (Figure 4), which represents different product or service types. The fraud rate varied a lot by category:

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

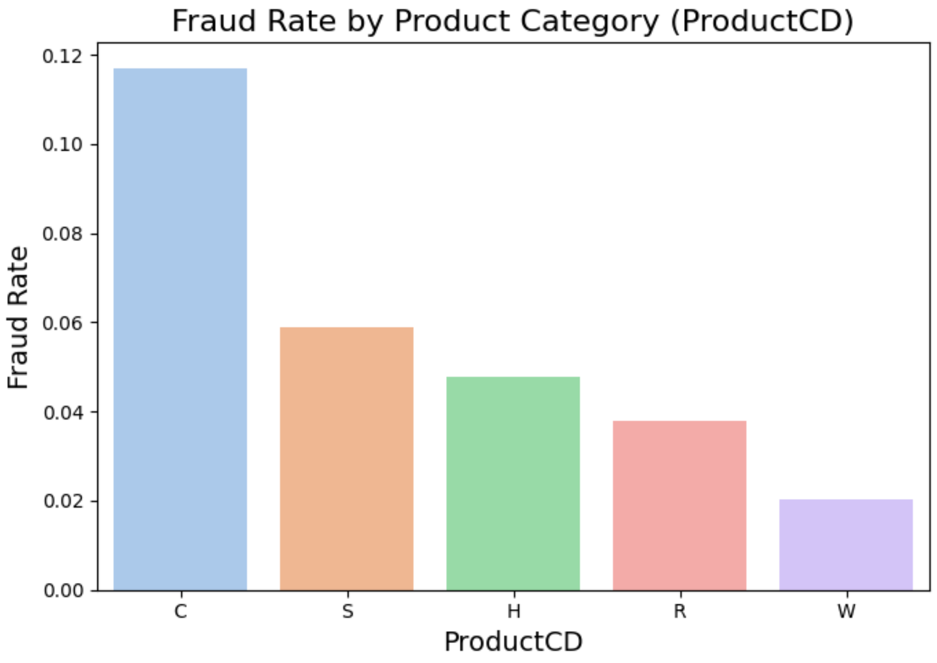

Figure 4: Fraud Rate by Product Category (ProductCD)

This tells us that some products are more targeted by fraud than others, which is really valuable when deciding what features to include in our model.

Then, we created a new feature called amt_over_500_risky_product, 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.

This gives us two important takeaways: Transaction amount and product type do carry useful fraud signals, especially when considered together, and feature engineering can reveal valuable patterns, but it's essential to test those assumptions with data first because not every combination will behave as expected.

These insights will help us fine-tune the features we give our models and guide how we build and improve them. This is especially useful for more advanced models like XGBoost and Deep Neural Networks, which often perform better when they're given smart features to learn from.

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 is a time delta in seconds, we transformed it into more meaningful units: transaction day, hour of day, and a weekend indicator. This helped us explore whether fraud tends to happen more at certain times, which supports RQ5: How does feature engineering affect fraud detection?

We found that the fraud rate by hour of day revealed a clear and meaningful pattern. Fraud was most common in the early morning hours, peaking sharply around 7 AM with a fraud rate over 10%, then gradually dropping through the rest of the day. This suggests that fraudsters may be targeting early hours, possibly to avoid detection during off-peak times.

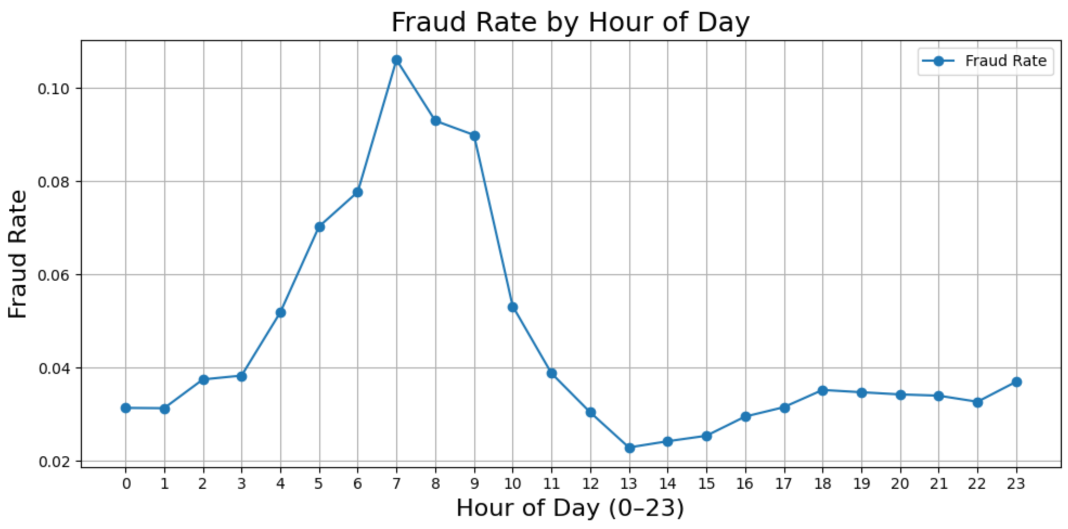

Figure 5: Fraud Rate by Hour of Day

We also looked at fraud by day of the dataset. While the trend was more irregular, it did show fluctuations in fraud rate over time, including several noticeable spikes. This may reflect periodic fraud campaigns or system vulnerabilities that were temporarily exploited.

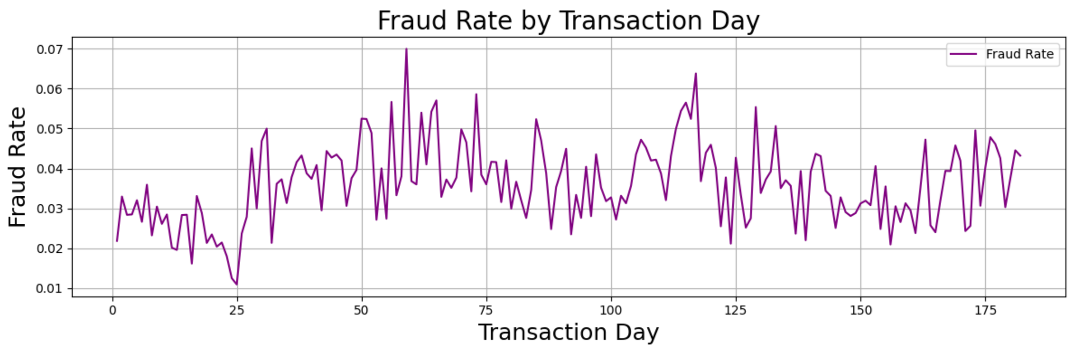

Figure 6: Fraud Rate by Transaction Day

Finally, we compared weekend vs. weekday fraud. Weekdays had a slightly higher fraud rate (3.55%) than weekends (3.38%), though the difference was modest. This could reflect more transaction volume and varied behavior on business days.

These results show that time carries real behavioral signals in fraud detection, and the patterns we uncovered provide strong direction for future feature engineering. For example, features like Transaction_hour or is_weekend may help models learn time-related risks. Hence, it helps us identify what features are worth testing later in supervised models like XGBoost or DNNs.

4. Regional and Geolocation Signals

In this analysis, we explored whether the location-based features addr1 (region) and addr2 (country) show patterns related to fraud. This supports RQ5, which focuses on identifying which features, especially engineered ones, help improve model performance.

We found that the fraud rate varied widely across different addr1 regions, with billing region-465 showing rates above 8%, while others remained much lower. This suggests that regional fraud risk may be influenced by geographic, demographic, or institutional factors.

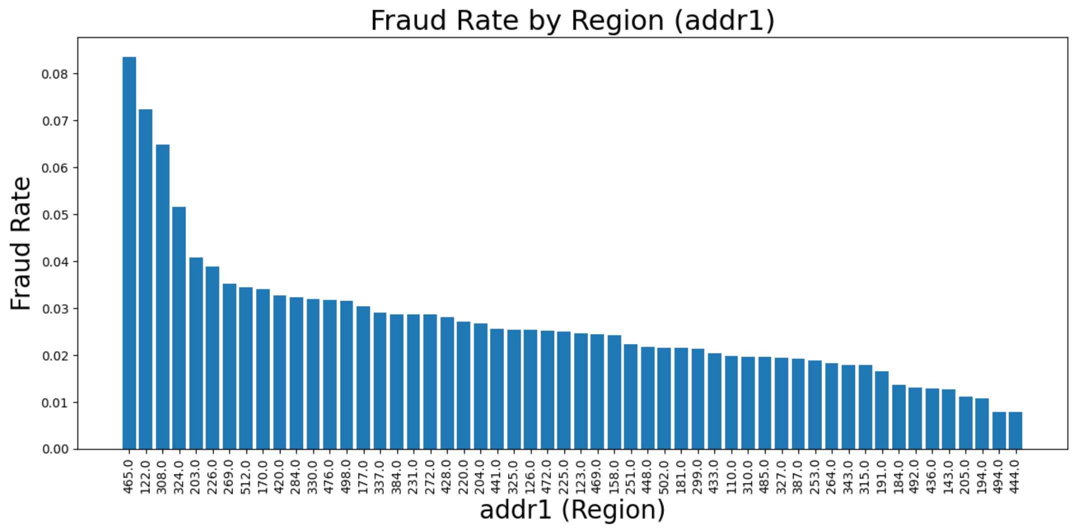

Figure 7: Fraud Rate by Region (addr1)

When examining addr2, we saw a sharp contrast in fraud rates across countries. Transactions from country code 87 (likely representing domestic U.S. cards) had a relatively low fraud rate of 2.4%, while those from other countries, like codes 60 and 96, showed much higher rates, ranging from 8.9% to 13.9%.

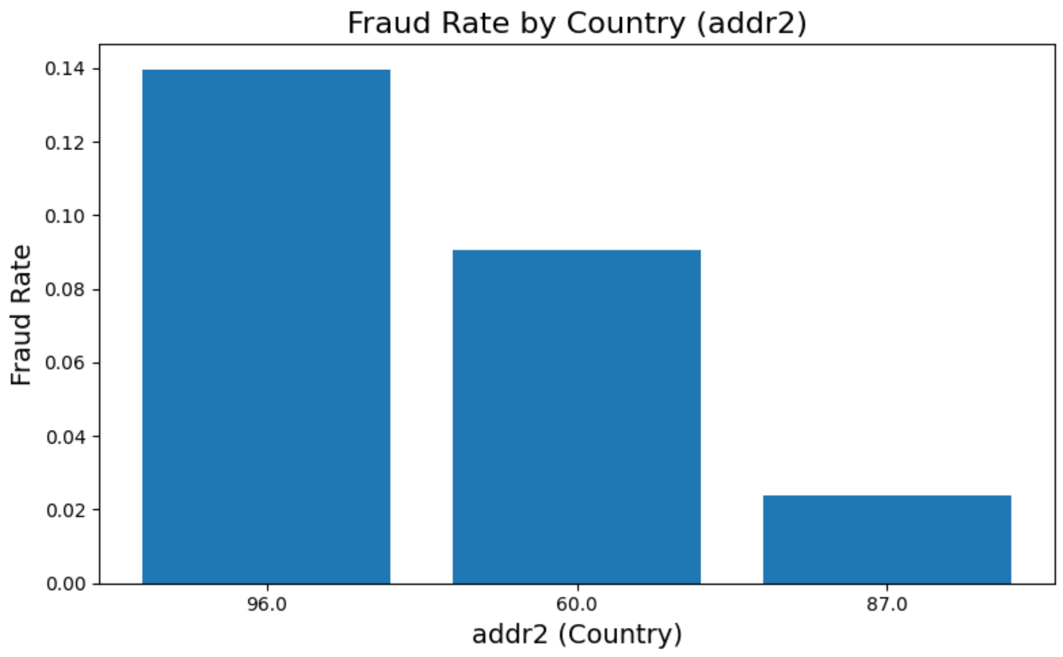

Figure 8: Fraud Rate by Country (addr2)

We also created a new binary feature, foreign_transaction_flag, which labels transactions as foreign if addr2 is not 87. This simple feature showed a very strong signal, with foreign transactions having a fraud rate of 11.7%, nearly five times higher than domestic ones (2.40%).

These results show that location-based fields like addr1 and addr2 provide meaningful insight into fraud risk. EDA results like this help guide the creation of effective 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.

5. Identity-Based Features: Email Domain and Device Type

This analysis focused on identity-related information such as email domain and device type to determine whether these features help detect fraud. This directly supports RQ2 of the project, which evaluates whether merging identity data with transaction data improves model performance.

I found that purchaser email domains showed clear differences in fraud risk. For instance, transactions linked to 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%. On the other hand, domains like yahoo.com and aol.com showed much lower fraud rates, typically around 2.1%. Even domains labeled as anonymous or missing still had non-zero fraud rates in the range of 2.5% to 2.9%, suggesting they carry some signal of risk.

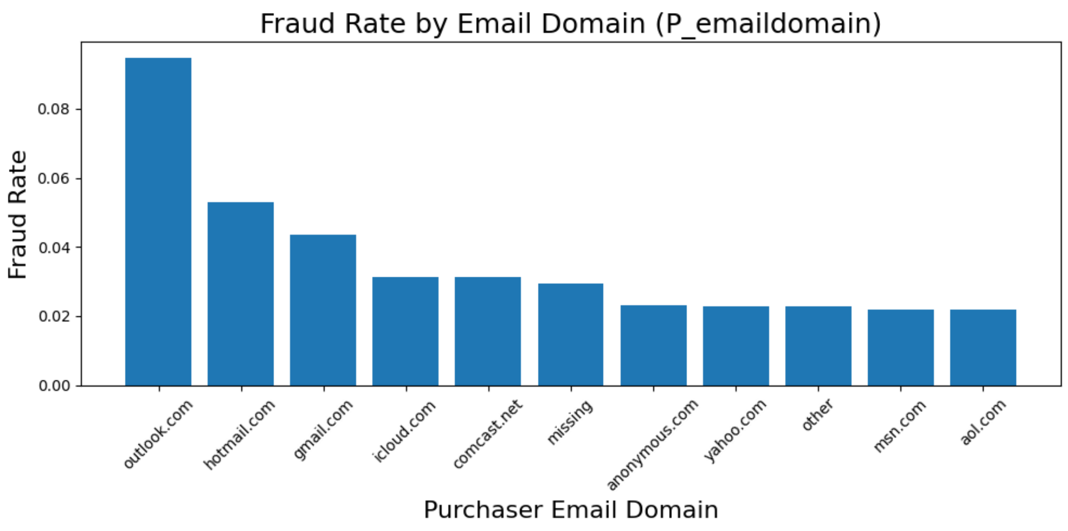

Figure 9: Fraud Rate by Email Domain (P_emaildomain)

Device type also revealed strong fraud patterns. Mobile transactions had the highest fraud rate at 10.1%, followed by desktop at 6.5%. Transactions with missing device type information showed the lowest fraud rate at 3.1%.

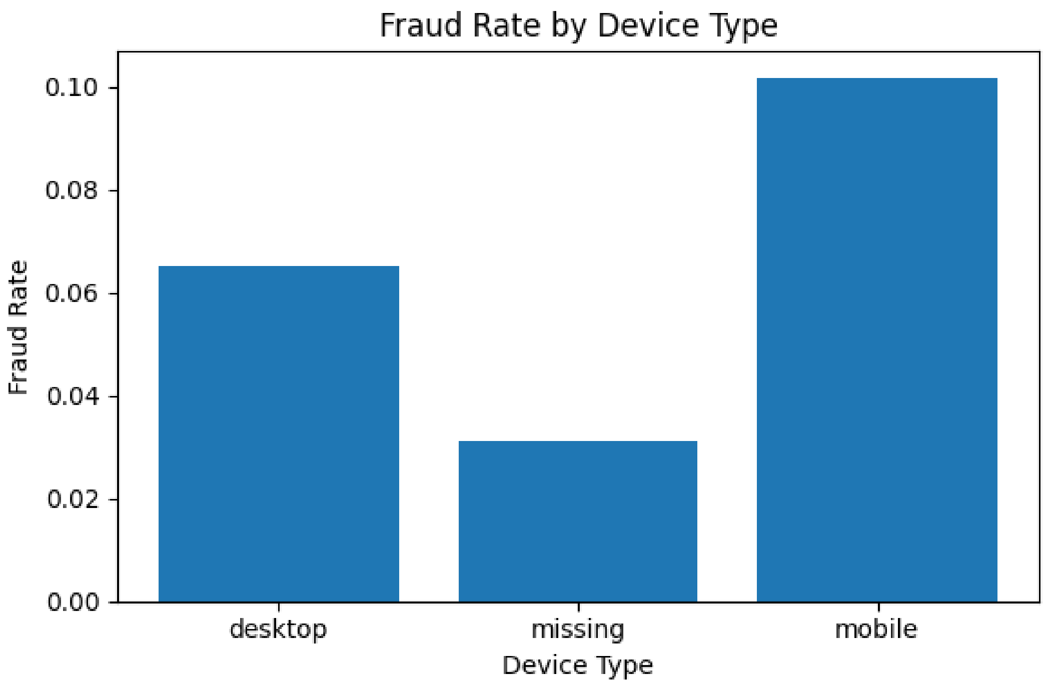

Figure 10: Fraud Rate by Device Type

When I looked at specific devices using the DeviceInfo field, certain Android models like “SM-J700M Build/MMB29K” showed fraud rates above 11%. In contrast, devices running MacOS or browsers like Trident/7.0 had much lower fraud risk, typically under 3%, while Windows and iOS devices fell somewhere in the middle, with fraud rates between 6% and 8%.

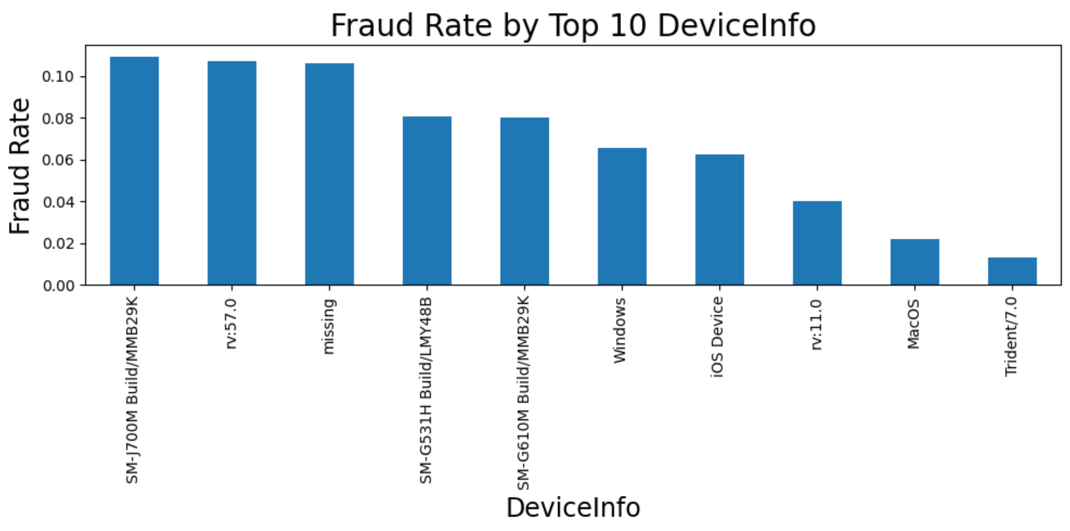

Figure 11: Fraud Rate by Top 10 DeviceInfo Values

These insights support several feature engineering opportunities that can directly feed into model development. For example, we can create a binary feature called is_disposable_email to flag risky or missing email domains, and another called is_mobile_device to highlight mobile-based transactions. Grouping DeviceInfo values into a device_risk_level variable or simplifying email domains into top categories using email_domain_grouped could also help improve model learning by reducing noise and cardinality.

Overall, this EDA demonstrates that identity fields are more than just normal information. They contain meaningful behavioral patterns that are clearly tied to fraud risk.

6. High-Correlation Numeric Features

In this part of the analysis, I wanted to find out which numerical features in the dataset are most closely linked to fraud. Since the dataset includes a lot of anonymized variables like the V-series, I used Pearson correlation to measure how each feature relates to the target variable, isFraud. This helps me narrow down which features might be useful when building early models, especially those that rely on numerical inputs like Logistic Regression. It also supports both RQ1 (which features best predict fraud) and RQ5 (how feature selection and engineering impact performance).

The numeric feature with the strongest correlation was V45, with a value of around 0.2818. That is relatively high in fraud detection, especially given the class imbalance. V44 came next at 0.2604, followed closely by V86 and V87, both just above 0.251. Other strong signals included V52 at 0.2395 and V51 at 0.2232. I also saw V40, V39, V38, and V43 showed correlations near or slightly above the 0.20 mark. While these aren't extremely high, they're still meaningful and worth considering during model development.

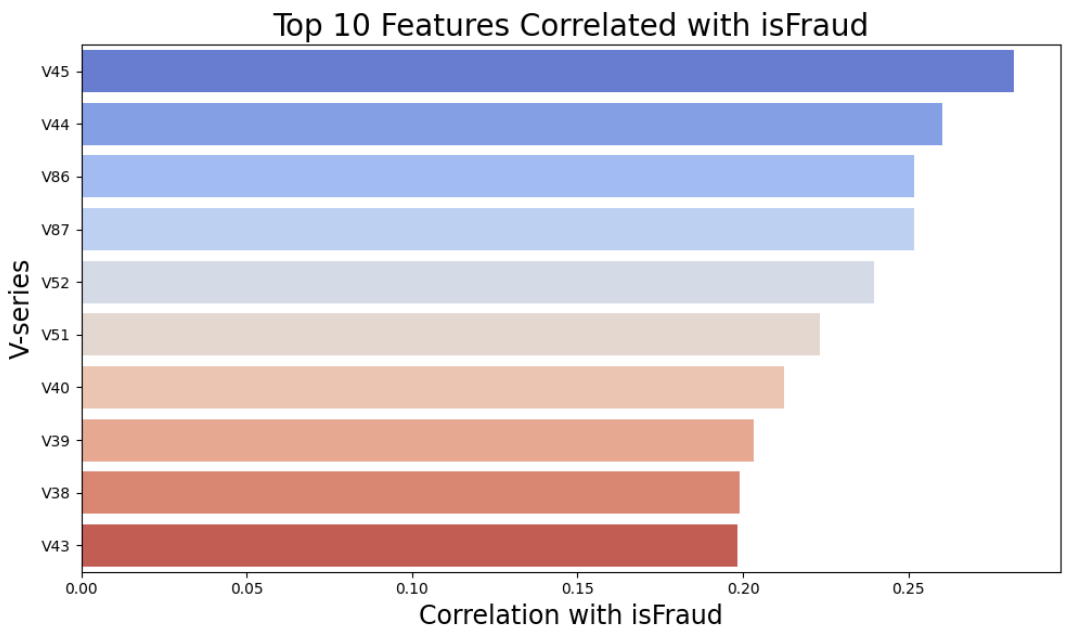

Figure 12: Top 10 Features Correlated with isFraud

To better understand how these numeric features behave, I visualized the distributions of V44 and V45 by fraud class. The plots (Figure 13 and Figure 14) showed very clear separation between fraud and non-fraud transactions. In each case, non-fraudulent transactions were tightly clustered near one, while fraudulent transactions were more spread out. This kind of distributional difference confirms the correlation results and shows that these features are carrying fraud-related signals despite their anonymized names.

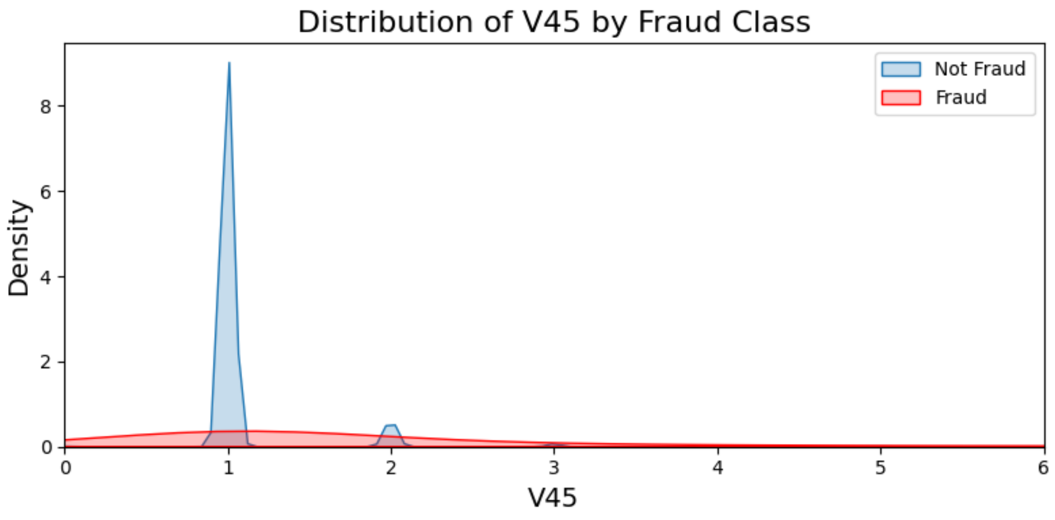

Figure 13: Distribution of V45 by Fraud Class

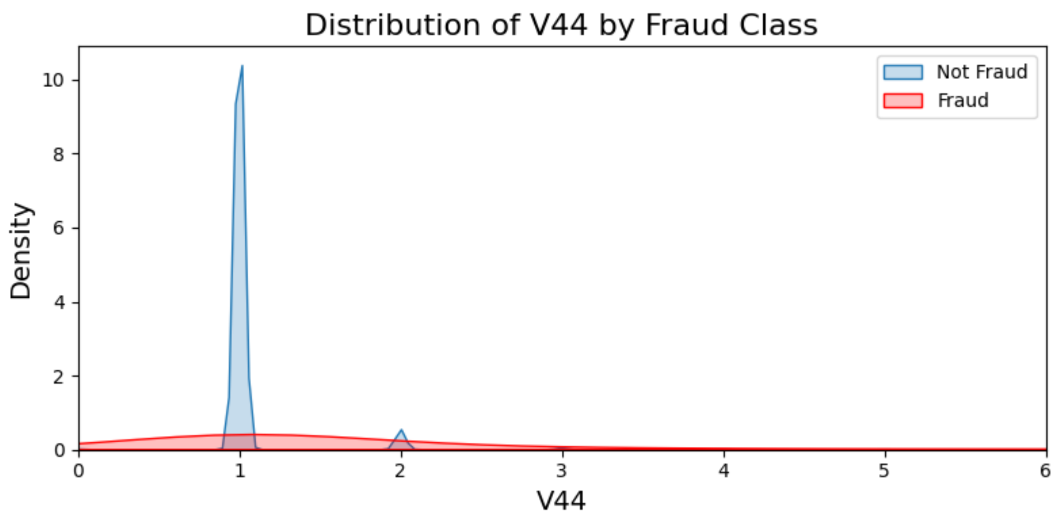

Figure 14: Distribution of V44 by Fraud Class

These insights are especially helpful at this stage because early models like Logistic Regression depend heavily on the quality and relevance of numeric inputs. Based on these findings, I'm planning to experiment with interaction features like V45 × V44, and possibly bin V45 into quantile-based risk tiers to reduce skew and make the feature easier to interpret.

Overall, this step helped me create a focused list of informative numeric features. It simplifies the modeling process and gives me a clearer idea of which variables are most promising for detecting fraud.

7. Categorical Feature Patterns

In this part of the analysis, I focused on three features that could reveal important fraud signals: card4, card5, and M6. The card1 to card6 fields contain payment card details, such as card type, card category, issuing bank, and possibly even the cardholder's country. Out of these, I selected card4 and card5 for deeper analysis, along with M6, which reflects whether the billing address matches the cardholder's information.

Starting with card4, which represents the card network (like Visa, Mastercard, or Discover), I noticed that fraud rates varied depending on the network. Discover had the highest fraud rate at around 7.8%, while Visa, Mastercard, and American Express were all between 3% and 3.5%. This pattern suggests that certain card networks may be more frequently targeted in fraudulent transactions—possibly due to differences in verification systems or fraud prevention measures.

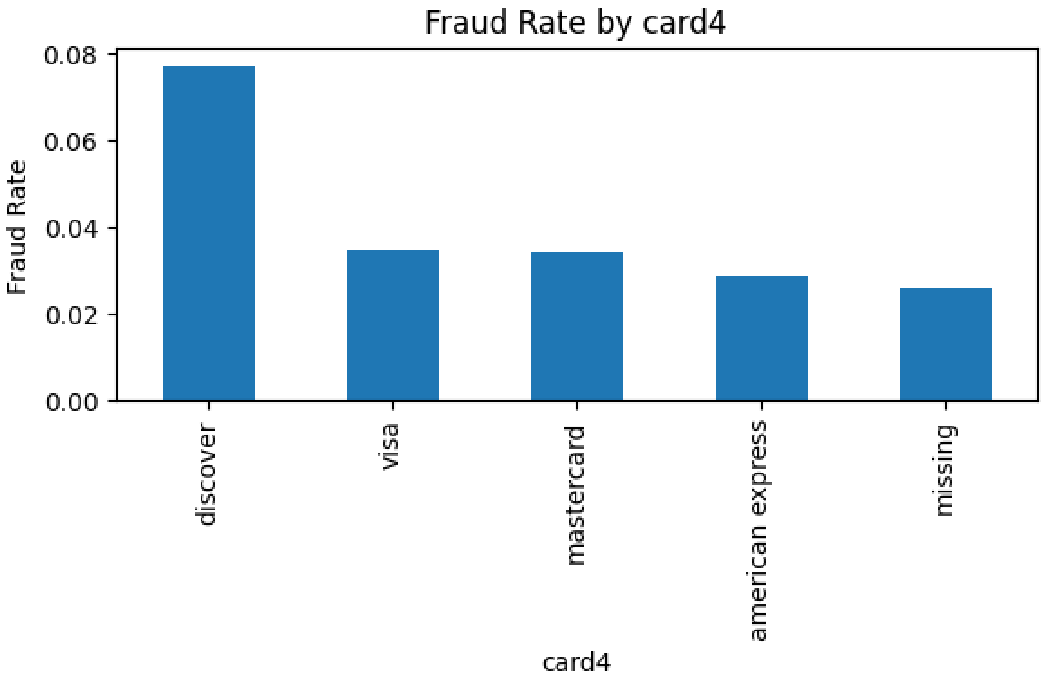

Figure 15: Fraud Rate by card4

For card5, which is technically numeric but behaves like a categorical feature, I limited the analysis to the ten most common values to keep things focused. One particular value, 137, stood out with a fraud rate above 14%. Others ranged between 2% and 9%. Even though the exact meanings of these values are anonymized, I believe they might reflect different banks or card tiers. Given this, I'm considering engineering a binary feature to flag high-risk card5 values or grouping them based on their observed fraud rates.

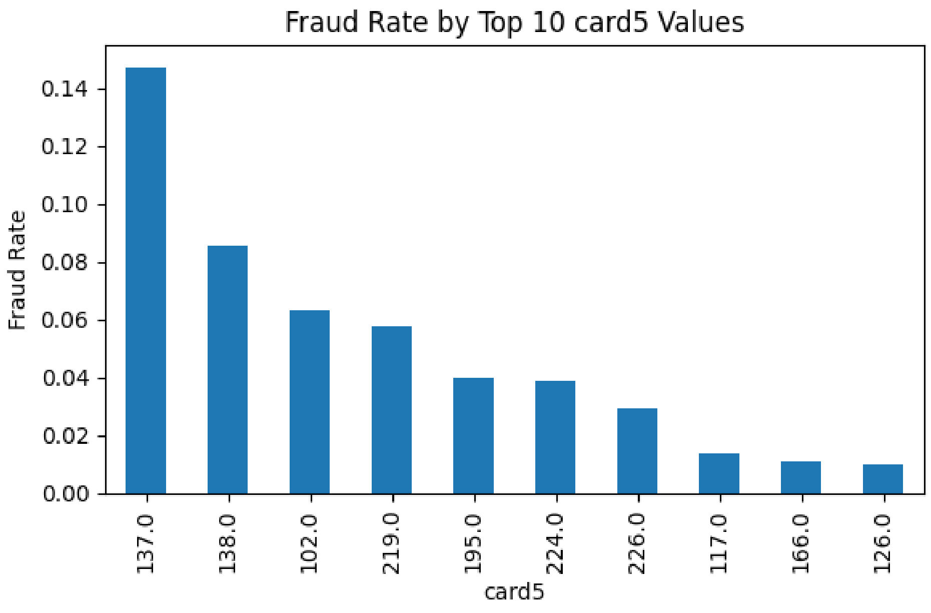

Figure 16: Fraud Rate by Top 10 card5 Values

Finally, I looked at M6, which shows whether the billing address matches the one on file. This feature was more straightforward. When M6 = T (True match), the fraud rate was just 1.7%. It increased to 2.4% for F (False match), and jumped to 7.1% when the value was missing. This makes M6 a strong candidate for binary feature engineering. For example, I could create a new flag like "M6_is_missing" to capture the elevated risk from incomplete data.

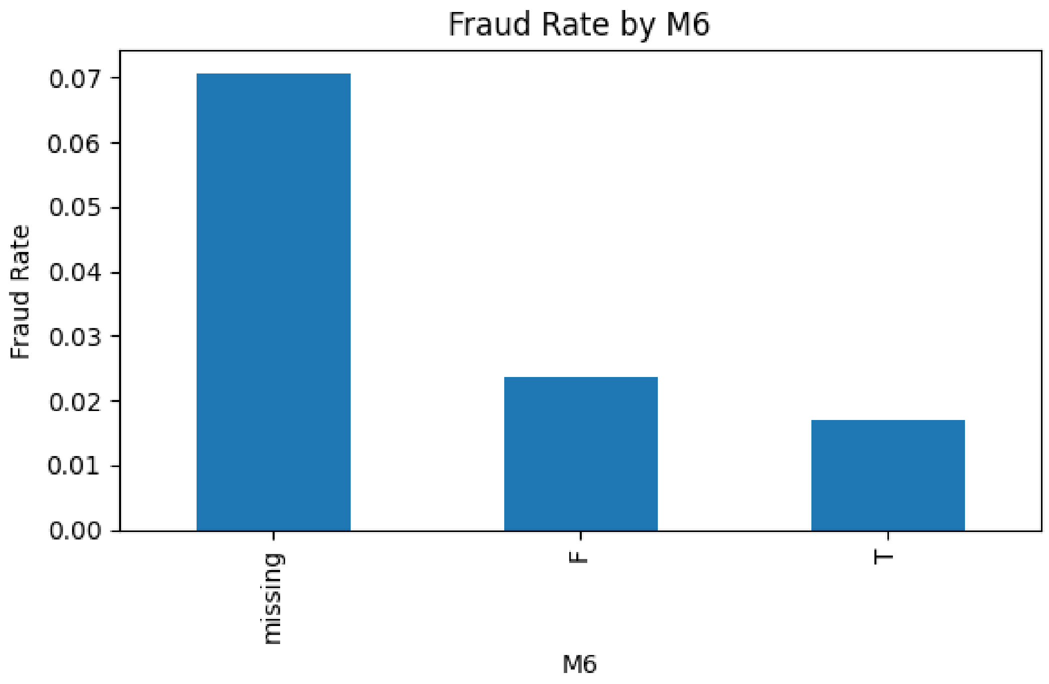

Figure 17: Fraud Rate by M6

Overall, these three features are easy to interpret and quite useful for modeling. They support not just exploratory insights, but also give clear directions for creating binary indicators, groupings, or one-hot encodings that can help supervised models like XGBoost and DNNs capture subtle fraud patterns more effectively.

8. Digital Fingerprinting from Identity Data

In this part of the analysis, I focused on a few key features from the identity dataset to explore whether digital fingerprinting data holds predictive value for fraud detection. Specifically, I looked at id_02 (a numerical identity score), id_30 (operating system), and id_31 (browser). These fields represent device- or session-level metadata collected by Vesta and its partners and are likely used to flag suspicious behavior patterns.

The distribution of id_02 by fraud class showed some subtle but useful separation. Most legitimate transactions were tightly concentrated around lower values, while fraudulent transactions were more broadly distributed, particularly in the higher range. Although not a dramatic split, this density shift still suggests that id_02 may reflect behavioral differences or identity scores that align with fraud risk.

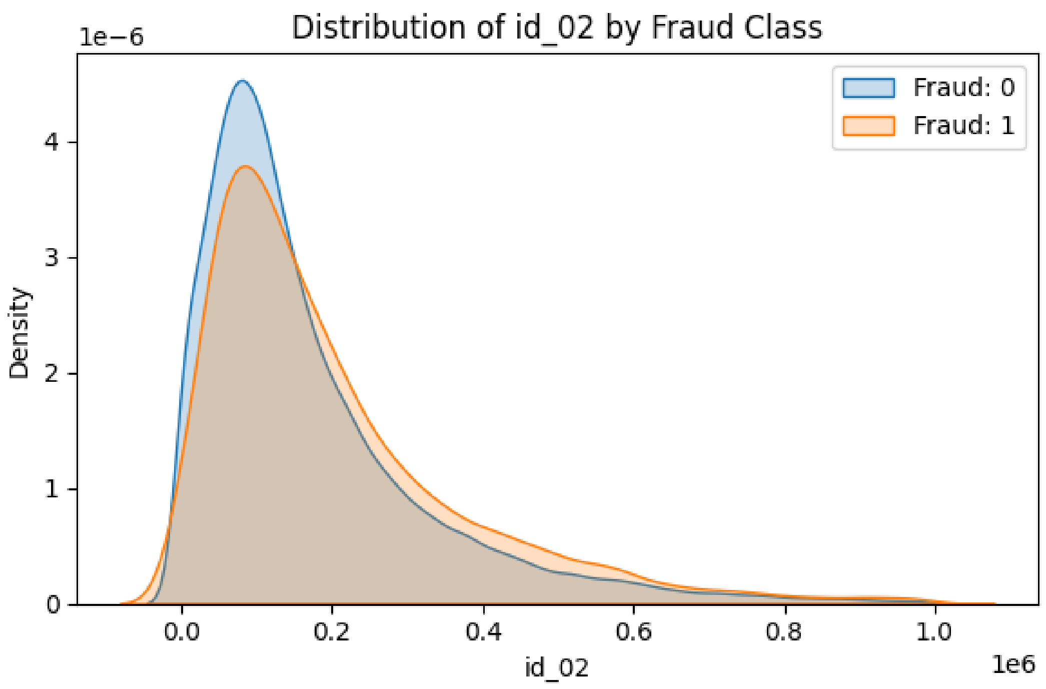

Figure 18: Distribution of id_02 by Fraud Class

For id_30, which captures the operating system, I found that the fraud rate was especially high for missing values, peaking around 11.5%. This suggests that when the system fails to detect or record the OS, the likelihood of fraud increases. Among detected values, older or mobile systems like Windows 8.1, iOS 11.3.0, and Android 7.0 showed moderately elevated fraud rates, which may reflect outdated or less secure environments.

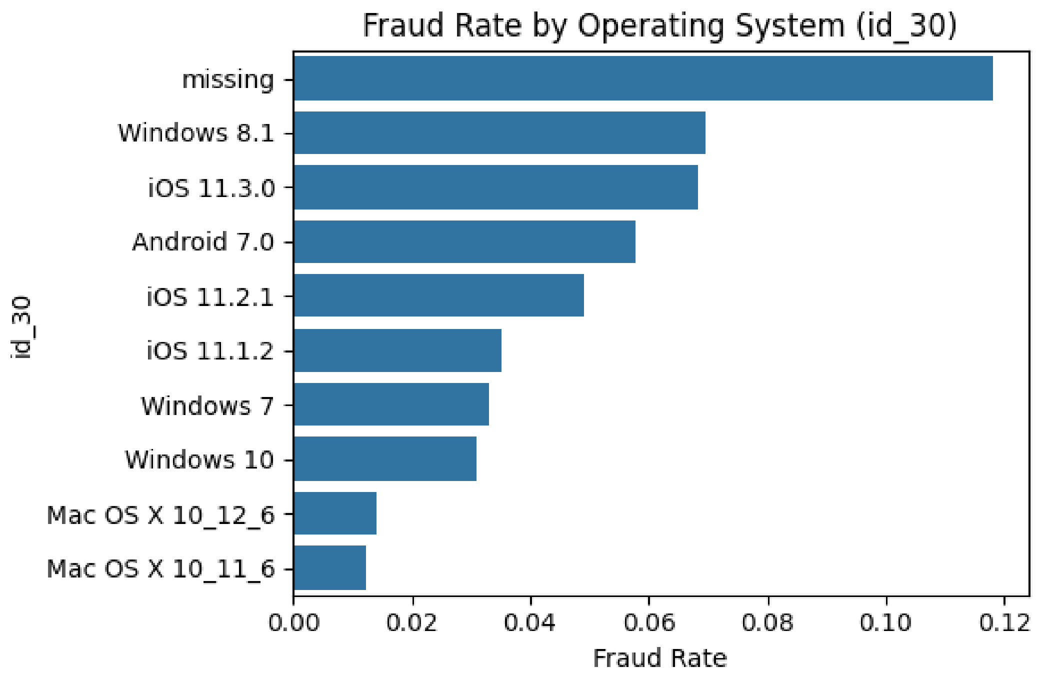

Figure 19: Fraud Rate by Operating System (id_30)

The pattern for id_31, the browser field, was even more striking. Generic browser labels like "chrome generic" and "mobile safari generic" had the highest fraud rates, with "chrome generic" surpassing 17%. This suggests that fraudsters may be using tools or techniques that mask their browser identity. In contrast, more specific or traditional browser labels like "ie 11.0 for desktop" had much lower fraud rates, under 3%.

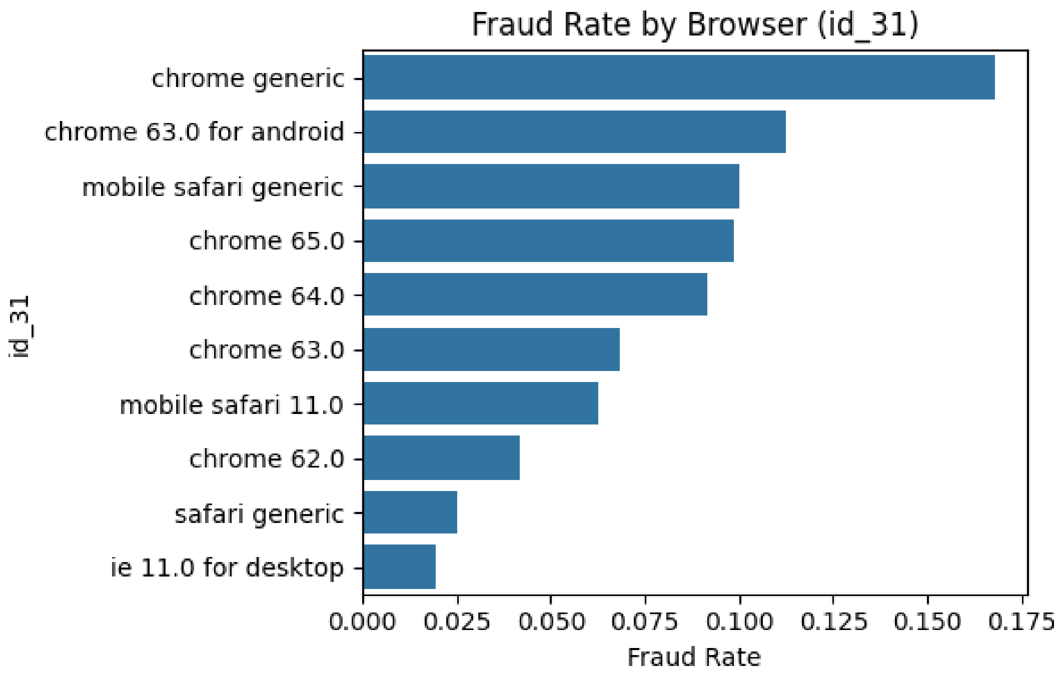

Figure 20: Fraud Rate by Browser (id_31)

These findings provide a strong argument for merging the identity table with transaction data. They also support creating new features like id_30_missing_flag or grouping browser values into "generic" versus "specific" categories. Identity-based digital fingerprinting clearly adds another layer of fraud signal that could improve model performance, especially for complex models like deep learning or XGBoost.

9. Spatial and Behavioral Risk Indicators

In this part of the analysis, I explored two distance-based features, dist1 and dist2, along with one count-based feature, C13, to check for fraud-related patterns. These features may capture location mismatches or behavioral outliers, even though the exact meaning of C13 is masked.

For dist1, fraud rates increased steadily with distance and peaked in the 200 to 500 mile range. This suggests that mid-range location gaps might be more suspicious than very short or very long ones. Beyond that, the fraud rate declined slightly.

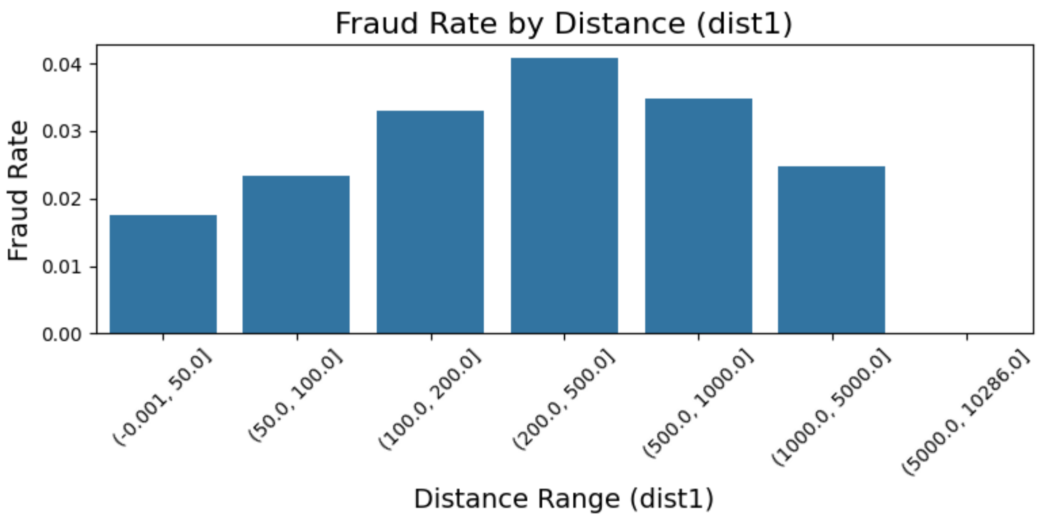

Figure 21: Fraud Rate by Distance Range (dist1)

Dist2 showed an even stronger signal. Fraud was more common in the 100 to 1000 mile range and jumped sharply for distances above 5000 miles, where the fraud rate exceeded 13%. This could reflect the use of spoofed IP addresses or proxies or VPN that introduce large mismatches between billing and access locations.

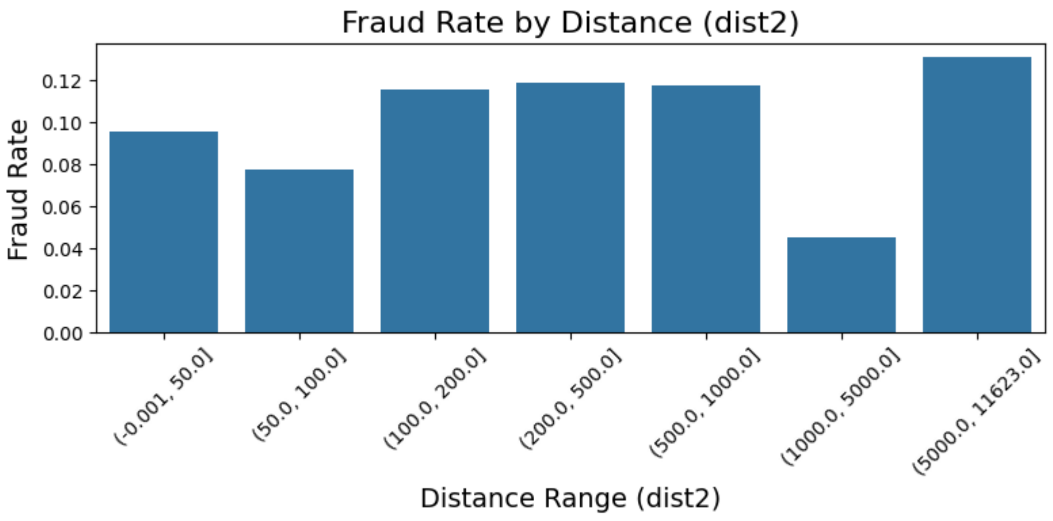

Figure 22: Fraud Rate by Distance Range (dist2)

C13 showed an inverse trend compared to typical count features. Fraud was highest when the value was low and decreased as the count increased. Although we do not know exactly what C13 measures, this pattern suggests that limited historical usage or rare associations may be riskier than more frequent or established ones.

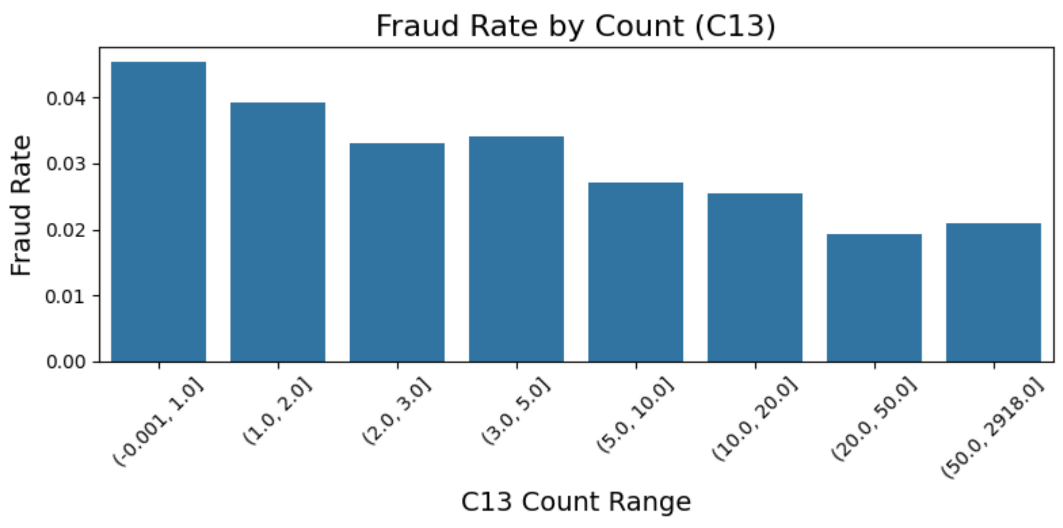

Figure 23: Fraud Rate by Count Range (C13)

Overall, creating simple flags like dist2_over_5000 or C13_low may help models capture these non-obvious patterns more directly. Binning or transforming these features can also reduce noise and improve the model's ability to learn useful fraud signals.

Project Approach

I started this project by reviewing fifteen peer-reviewed journal and conference papers on fraud detection. I wanted to get an overview of how supervised machine learning is being used and what challenges researchers often face. These challenges include extreme class imbalance, anonymized features, and the trade-off between catching fraud and avoiding false positives. Exploring these studies helped me decide the direction of my project by highlighting effective strategies such as ensemble learning, identity feature engineering, and data balancing methods.

After getting the general background, I then performed an exploratory data analysis (EDA) on the IEEE-CIS Fraud Detection dataset. This step allowed me to better understand the structure, quality, and key features of the data, especially the missing values and how fraud cases are distributed. One of the first things I had to do was clean the dataset by handling missing values and standardizing column names to make the data easier to work with in Python.

Overall, the literature review and EDA have given me a strong foundation for building and evaluating machine learning models in the next phase. Below is a diagram that outlines my overall approach from research to modeling.

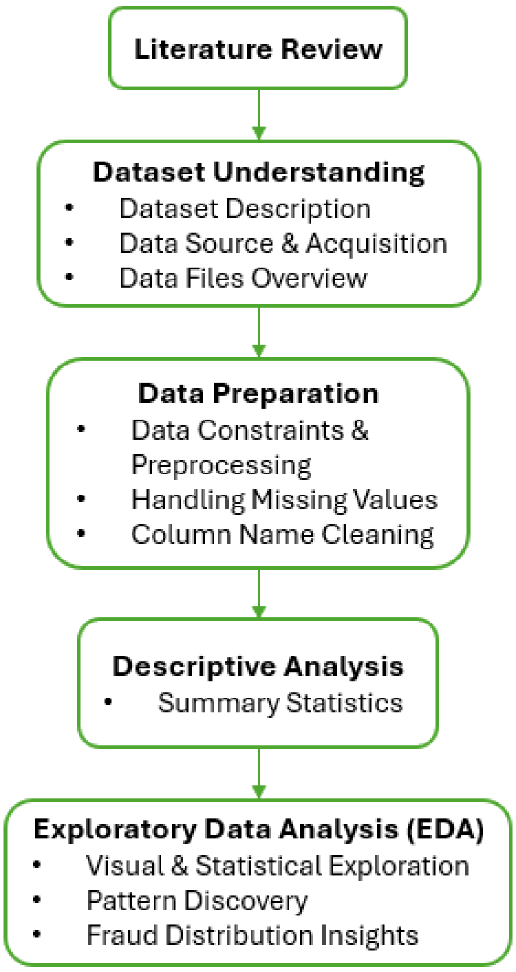
Literature Review
Dataset Understanding
• Dataset Description
• Data Source & Acquisition
• Data Files Overview
Data Preparation
• Data Constraints & Preprocessing
• Handling Missing Values
• Column Name Cleaning
Descriptive Analysis
• Summary Statistics
Exploratory Data Analysis (EDA)
• Visual & Statistical Exploration
• Pattern Discovery
• Fraud Distribution Insights

Figure 24: Overall Project Methodology

GitHub Repository

All codes, visualizations, and results for this Literature Review and EDA stage are available at the following GitHub repository:
Link: `https://github.com/nguyenduyanhluong/TMU-MRP-2025/tree/main`

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