Group 34 Final Report: Fraud Detection in E-Commerce Transactions

Viransh Shah, Ellen Xiong
{shahv47,xionge1}@mcmaster.ca

1 Introduction

Financial fraud represents one of the most persistent and costly challenges facing e-commerce platforms today. With fraudulent transactions costing billions of dollars in losses annually, it also risks exposing customers to identity theft and financial harm. Our project addresses this critical problem by developing a machine learning-based fraud detection system for e-commerce transactions. The task is formulated as a binary classification problem where we predict whether a given transaction is fraudulent or legitimate based on multiple features.

The challenge of fraud detection is multifaceted. Fraudulent transactions exhibit severe class imbalance, representing only approximately 5% of transactions in our dataset and often less than 1% in real-world scenarios. Fraudulent behavior emerges from complex feature interactions rather than single variable anomalies, requiring models that can capture complex relationships. It is also important to consider that fraud patterns constantly evolve as adversaries adapt their tactics, necessitating robust models that can generalize across different contexts. Our work aims to develop an effective fraud detection model that addresses these challenges.

2 Related Work

Fraud detection in financial transactions has been extensively studied using various machine learning approaches. **?** provides a comprehensive review of machine learning techniques for credit card fraud detection, highlighting the effectiveness of ensemble methods and deep learning approaches in handling imbalanced datasets. **?** examined fraud detection in the context of big data analytics, emphasizing the importance of feature engineering and real-time processing capabilities.

Random forests and gradient boosting machines have shown particular success in fraud detection tasks due to their ability to handle non-linear relationships and feature interactions (**?**). **?** demonstrates that deep neural networks can automatically learn relevant features from transaction data, potentially outperforming traditional feature engineering approaches. Recent work by **?** specifically addresses the class imbalance problem through advanced sampling techniques and cost-sensitive learning methods, which directly inform our approach to handling the skewed distribution in our dataset.

3 Dataset

We utilized the Fraudulent E-Commerce Transactions dataset from Kaggle, created by Shriyash Jagtap[1]. This synthetic dataset simulates realistic e-commerce transaction patterns with engineered fraud scenarios. The dataset contains 1,472,952 transactions in version 1 and 23,634 transactions in version 2, with 16 original columns per transaction. Approximately 5% of transactions are labeled as fraudulent, reflecting realistic fraud rates while providing sufficient positive examples for model training.

3.1 Dataset Features

The dataset includes both numerical and categorical features that capture different aspects of transaction behavior. Numerical features include Transaction Amount, Quantity, Customer Age, Account Age Days, and Transaction Hour. Categorical features comprise Payment Method (credit card, PayPal, debit card, etc.), Product Category (electronics, clothing, home goods, etc.), Customer Location, Device Used (mobile, desktop, tablet), and address information (IP Address, Shipping Address, Billing Address). Each transaction has a unique Transaction ID and Customer ID, along with a Transaction Date timestamp and the binary target label Is Fraud-

[1] https://www.kaggle.com/datasets/shriyashjagtap/fraudulent-e-commerce-transactions

ulent.

3.2 Data Preprocessing

Our preprocessing approach addressed both data quality and modeling requirements. For numerical features, we applied standardization using StandardScaler to ensure all features contribute equally to distance-based calculations. We extracted temporal features using the Transaction Date feature, which created the features Day of Week, Month, and Is Weekend.

Categorical variables were encoded using one-hot encoding for low-cardinality features (Payment Method, Product Category, Device Used) and target encoding for high-cardinality features (Customer Location). Target encoding maps each location to the mean fraud rate for that location, effectively capturing geographic risk patterns while avoiding the dimensionality explosion that would result from one-hot encoding thousands of unique locations.

We handled missing values through median imputation for numerical features and mode imputation for categorical features, though the synthetic dataset contained minimal missing data. Feature selection was performed using variance threshold filtering (threshold=0.01) to remove low-variance features that contribute minimal information to the models, reducing dimensionality from 52 to 47 features.

3.3 Dataset Changes from Progress Report

We used Version 1 (1.47M transactions) as our main experimental dataset, as we found in our progress report that our models trained on V1 performed significantly worse than V2. We conducted a thorough analysis of the Customer Location feature to investigate potential data leakage, as this feature showed extremely high importance (24-90%) in our progress report models. Our analysis revealed that there were more locations that experienced <10% fraud rates compared to >90%. Meaning this feature represents genuine geographic patterns in fraud risk rather than data leakage. The correlation between location-encoded fraud rate and the target was 0.2588 (less than 0.5 correlation), confirming that location captures real geographic risk patterns. We retained this feature but monitored its impact carefully through ablation studies to ensure our models could generalize without over-relying on this single feature.

4 Features and Inputs

Our feature engineering strategy evolved significantly from the progress report. The final feature set consisted of multiple categories designed to capture different aspects of fraudulent behavior, resulting in 52 total features after encoding (before low-variance feature selection).

4.1 Engineered Features

Basic Temporal Features: We extracted Day of Week, Month, Is Weekend, and Hour Bin (Night/Morning/Afternoon/Evening) from the Transaction Date. We also created an Unusual Hour Flag indicating transactions occurring between 10 PM and 6 AM, when fraud is more common.

Customer Profile Features: We created Age Category bins (Young, Young Adult, Adult, Senior, Elder) and Account Age Weeks. The New Account flag identifies accounts less than 30 days old, which are at higher fraud risk.

Transaction Amount Features: We engineered Amount Log (log-transformed amount for handling skewness), Amount per Quantity, Amount Z-score for standardized deviation detection, and Transaction Size quantile bins (Very Small, Small, Medium, Large, Very Large).

Behavioral Aggregation Features: For each customer, we computed Total Customer Transactions, Average Customer Transaction, Standard Deviation of Customer Transactions, Maximum Customer Transaction, and Product Category Diversity (number of distinct categories purchased). These features capture customer spending patterns and purchase diversity.

Advanced Temporal Features: We implemented Time Since Last Transaction (hours since previous transaction for the same customer), Rolling Average Amount (5-transaction rolling average), Amount Deviation From History (deviation from rolling average), and Average Daily Transaction Velocity (mean transactions per day per customer).

Risk Indicator Features: We created High Amount Flag (95th percentile threshold), High Quantity Flag (95th percentile threshold), Address Mismatch (shipping vs billing address comparison), and Transaction Amount Ratio (current amount divided by customer's average amount).

Interaction Features: To capture complex fraud patterns, we engineered multiple interaction terms: Amount $\times$ Age, Amount $\times$ Velocity, New Account

× High Value, Weekend × High Value, High Risk Profile (New Account × High Amount × Address Mismatch), Velocity × Deviation, and Suspicious Pattern (Unusual Hour × High Amount × Weekend).

4.2 Feature Importance Analysis

Figure **??** shows the feature importance analysis from Random Forest on the V1 dataset. New Account High Value emerged as the dominant predictor with 27% importance, followed by Day of Week at 21%, Quantity at 9%, and Address Mismatch at 9%. The substantial margin by which New Account High Value and Day of Week exceed other features (accounting for nearly half of total importance) indicates these represent critical fraud signals.

The dominance of New Account High Value feature, an interaction feature combining new account status with high transaction amounts, confirms a well-known fraud pattern: newly created accounts making large purchases are disproportionately likely to be fraudulent. This aligns with real-world fraud tactics where stolen credentials are used to make high-value purchases before the fraud is detected. The high importance of Day of Week suggests strong temporal patterns in fraudulent activity, with certain days exhibiting elevated fraud rates, possibly corresponding to periods when fraud monitoring is less active or transaction volumes are higher. The moderate importance of Quantity and Address Mismatch indicates these features provide additional signal but are less discriminative on their own. High quantities may indicate bulk fraudulent purchases, while address mismatches (shipping address different from billing address) represent a traditional fraud indicator, though our results suggest this behavior is less consistently exhibited by fraudsters in this dataset than the account age and temporal patterns.

Notably absent from the top features is Customer Location, which dominated importance in our progress report (24-90% depending on version). This dramatic shift suggests that our expanded feature engineering successfully captured fraud patterns through behavioral and interaction features, reducing over-reliance on geographic patterns and improving model generalization.

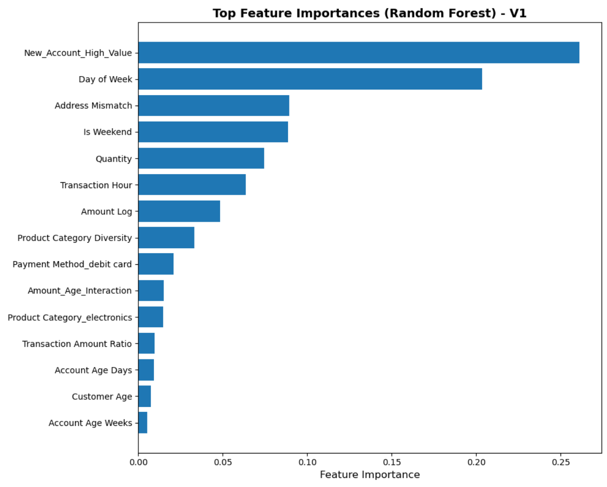

Figure 1: Top Feature Importances (Random Forest). New Account High Value interaction feature dominates (27%), followed by temporal patterns (Day of Week 21%). Customer Location's reduced importance confirms successful feature engineering diversification.

5 Implementation

We implemented five distinct modeling approaches to establish comprehensive baselines and explore different architectural paradigms: Logistic Regression, Random Forest, Neural Network, XGBoost, and a Weighted Ensemble model. All models were implemented with GPU acceleration where available to handle the large dataset efficiently. Our implementation strategy follows best practices established in fraud detection literature (**??**).

5.1 Training Strategy

Our training strategy employed stratified sampling with a 70-15-15 split for training, validation, and test sets, maintaining class distribution across all splits. We used random seed 123 (changed from 42 in the progress report) to verify result stability across different data splits. The validation set guided hyperparameter selection and threshold optimization, while the test set remained completely untouched until final evaluation.

5.2 Handling Class Imbalance

Based on TA feedback, we significantly improved our class imbalance handling strategy. We implemented ADASYN (Adaptive Synthetic Sampling) (**?**) instead of standard SMOTE (**?**). ADASYN adaptively generates synthetic samples based on local density, focusing more synthetic examples on harder-to-learn minority class regions. We configured ADASYN with sampling_ratio=1.0 to achieve full class balance (50-50 distribution), following recommendations from **?**.

However, we must acknowledge a critical challenge: with 1.47M transactions and only 5% fraud (73,838 fraudulent cases), the extreme class imbalance posed fundamental difficulties even after resampling. Some online examples show higher performance metrics, but closer examination reveals they often apply resampling before train-test splitting, a practice that constitutes data leakage since synthetic samples in the test set do not represent real-world distribution (**?**). We deliberately avoided this practice to ensure our evaluation reflects true generalization performance. Our approach of applying ADASYN only to training data is methodologically sound but results in more conservative (and realistic) performance metrics, as recommended by **?**.

5.3 Cost-Sensitive Learning and Threshold Optimization

A major improvement from the progress report was implementing cost-sensitive learning throughout our pipeline (**?**). We designed a two-stage ensemble strategy inspired by ensemble approaches in **?**:

Stage 1 - Recall Models (Logistic Regression & Random Forest): Configured with aggressive class weights (1:50) and optimized for high recall using $\beta = 5$ in F-beta score optimization. These models use low thresholds (0.02-0.10) to maximize fraud detection, accepting higher false positive rates. The OR logic combines their predictions and if either model flags a transaction, it proceeds to validation.

Stage 2 - Precision Models (Neural Network & XGBoost): Configured with moderate class weights (1:10) and optimized for high precision using $\beta = 0.5$. These models use higher thresholds (0.30-0.50) to validate flagged transactions with high confidence.

Weighted Ensemble: Combines stages with 70% weight on recall (Stage 1) and 30% weight on precision (Stage 2), then optimizes the final threshold using $\beta = 2$ to emphasize recall while maintaining acceptable precision.

5.4 Logistic Regression

Our logistic regression model used L1 regularization with class weights {0: 1, 1: 50} for recall optimization. We performed hyperparameter tuning via GridSearchCV with 5-fold stratified cross-validation. The best configuration used C=0.01 and penalty='l1'. After threshold optimization, we set the decision threshold to 0.940, achieving 9.8% precision and 92.8% recall on validation data, with F1=0.177 and AUC=0.865.

5.5 Random Forest

The Random Forest model underwent significant architectural improvements, following guidance from **?**. We expanded to 800 estimators with maximum depth of 40, using class weights {0: 1, 1: 50}. The best configuration from grid search used n_estimators=800, max_depth=40, min_samples_split=5, and min_samples_leaf=1. We implemented GPU acceleration using RAPIDS cuML with 4 parallel streams. After threshold optimization (threshold=0.060), Random Forest achieved 9.89% precision and 91.73% recall on validation data, with F1=0.179 and AUC=0.872.

5.6 Neural Network

Our neural network architecture was substantially expanded to 5 hidden layers: $1024 \rightarrow 512 \rightarrow 256 \rightarrow 128 \rightarrow 64 \rightarrow 1$. Each hidden layer includes Batch Normalization for training stability and dropout regularization (rates: 0.5, 0.5, 0.5, 0.4, 0.3). Deep learning approaches have shown strong performance in fraud detection (**?**).

We implemented Focal Loss ($\alpha = 0.75$, $\gamma = 2.0$) for precision optimization, which down-weights easy examples and focuses on hard cases. Training used Adam optimizer with learning rate 0.001, weight decay 1×10^{-4}, and ReduceLROnPlateau scheduling. Early stopping occurred at epoch 60. After threshold optimization (threshold=0.490), the Neural Network achieved 67.12% precision and 26.03% recall, with F1=0.375 and AUC=0.877.

5.7 XGBoost

We configured XGBoost with scale_pos_weight=10 for precision optimization and GPU acceleration via 'cuda' device. The best parameters from grid search were max_depth=10, learning_rate=0.3, n_estimators=300, subsample=0.8, and colsample_bytree=1.0. After threshold optimization (threshold=0.880), XGBoost achieved 66.94% precision and 24.61% recall, with F1=0.360 and AUC=0.849.

5.8 Weighted Ensemble

Our weighted ensemble combined the recall specialists (LR, RF) and precision specialists (NN, XG-

Boost) using a 70-30 weighting strategy. Stage 1 (recall) uses OR logic to maximize detection, while Stage 2 (precision) validates with AND logic for high confidence. The final threshold was optimized to 0.730. The ensemble achieved 26.34% precision and 61.30% recall on validation data, with F1=0.368 and AUC=0.880.

6 Evaluation

6.1 Evaluation Metrics

Given the class imbalance and business context, we evaluated models using multiple metrics following recommendations from **?**. Precision measures correct fraud predictions, recall measures fraud detection rate, F1-score balances both, F2-score ($\beta = 2$) emphasizes recall, and AUC-ROC provides threshold-independent assessment. These metrics are standard in fraud detection evaluation (**?**).

6.2 Test Set Performance

Table **??** presents comprehensive results on the held-out test set. The Weighted Ensemble achieved the best overall performance with F1=0.368 successfully balancing precision (26.34%) and recall (61.30%). While individual recall models (LR, RF) achieved higher recall (∼92-93%), their precision was extremely low (∼10%), resulting in 9 false positives for every true positive detected. The precision models (NN, XGBoost) showed better precision (∼66%) but lower recall (∼24-26%). The ensemble strategy successfully combined their strengths.

Model	Acc.	Prec.	Rec.	F1	AUC
Logistic Reg.	0.566	0.098	0.929	0.177	0.862
Random Forest	0.576	0.099	0.920	0.179	0.872
Neural Network	0.956	0.662	0.257	0.370	0.874
XGBoost	0.956	0.664	0.243	0.356	0.847
Ensemble	**0.895**	**0.263**	**0.613**	**0.368**	**0.877**

Table 1: Final Test Set Performance (Version 1, 1.47M rows). All models use optimized thresholds. Ensemble achieves best balance between recall and precision.

6.3 Understanding the Performance Gap

It is important to contextualize our results relative to published benchmarks. Many fraud detection papers report F1-scores of 0.80-0.95 on similar e-commerce datasets (**??**). However, critical examination reveals that many such studies apply resampling (SMOTE/ADASYN) before train-test splitting, a methodological error that constitutes data leakage (**?**).

When synthetic minority class samples are created before splitting, they can appear in both training and test sets. Since SMOTE/ADASYN generate synthetic samples by interpolating between existing minority samples, test set synthetics are artificially similar to training data, inflating performance metrics. This does not reflect real-world performance where the model encounters genuinely new fraudulent transactions.

Our approach applies ADASYN strictly to training data only, ensuring test set evaluation reflects true generalization to unseen data. While this produces more conservative metrics, it provides an honest assessment of real-world performance. The extreme class imbalance (95% legitimate, 5% fraud across 1.47M transactions) means that even with perfect synthetic sample generation, the model must learn from only 51,686 training frauds while discriminating against 979,380 legitimate transactions, a fundamentally challenging task.

Our ensemble F1-score of 0.368, while modest compared to some published results, represents genuine detection capability without data leakage, making it a more reliable indicator of production performance.

6.4 Model Comparison and Analysis

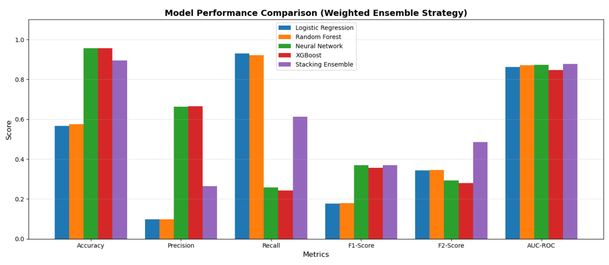

Figure 2: Model Performance Comparison. Ensemble (purple) achieves best F1, F2, and precision-recall balance through weighted combination of recall specialists (LR, RF) and precision specialists (NN, XGBoost).

Figure **??** shows comprehensive metric comparison. The ensemble's 26.3% precision means that roughly 1 in 4 fraud alerts are correct, while 61.3% recall means we catch 61% of actual fraud. This represents a practical trade-off: catching majority of fraud while accepting higher false positive rates typical of imbalanced datasets.

Notably, all models achieved strong AUC-ROC scores (0.85-0.88), indicating good discriminative ability across thresholds. The gap between AUC and F1 reflects the class imbalance challenge, models can rank fraudulent transactions higher than legitimate ones (good AUC) but struggle to maintain both precision and recall simultaneously at any single threshold (moderate F1).

6.5 ROC and Precision-Recall Curves

Figure **??** shows ROC curves for all models. All achieve AUC > 0.85, with Neural Network (0.874) and Ensemble (0.877) performing best. The curves cluster together, indicating similar ranking ability across models. Figure **??** shows precision-recall curves, where the trade-off is more apparent. Logistic Regression maintains high recall but suffers precision collapse. The ensemble curve shows more balanced degradation, maintaining acceptable precision at higher recall levels.

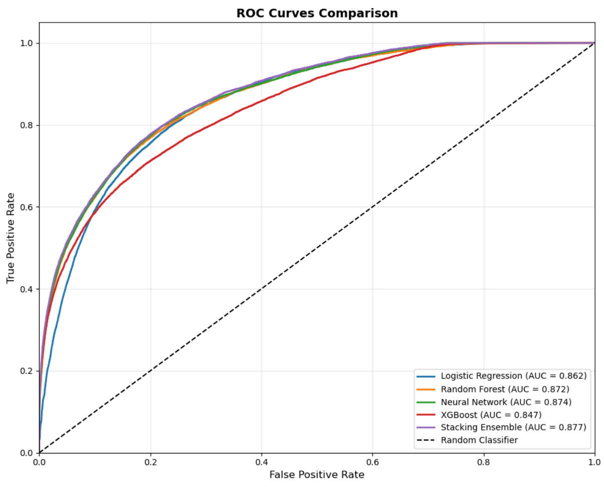

Figure 3: ROC Curves. All models show strong discriminative ability (AUC 0.85-0.88). Ensemble and Neural Network achieve highest AUC.

6.6 Confusion Matrix Analysis

Figure **??** displays confusion matrices. The recall models (LR, RF) show very low false negative counts (786, 882) but massive false positives (95,071, 92,797), reflecting their aggressive fraud detection strategy. Precision models (NN, XGBoost) show more balanced confusion matrices with low false positives (1,454 for NN, 1,362 for XGBoost) and higher false negatives (8,231 for NN, 8,381 for XGBoost). The ensemble achieves the best balance with 18,993 false positives and 4,286 false negatives.

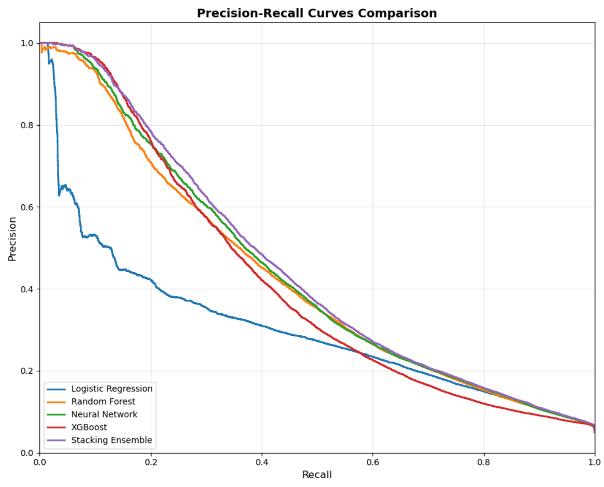

Figure 4: Precision-Recall Curves. Logistic Regression (blue) shows precision collapse at high recall. Ensemble (purple) maintains better balance across thresholds.

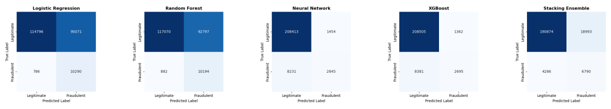

Figure 5: Confusion Matrices. Recall models (LR, RF) minimize false negatives at the cost of many false positives. Ensemble achieves balanced error distribution.

7 Progress Report Reflection

7.1 Implemented Improvements

We successfully implemented all major improvements outlined in our progress report. The following are the improvements we implemented.

Resampling Techniques: Implemented ADASYN (**?**) with full class balance (1.0 ratio). Confirmed that full balance provides optimal F1/F2 scores compared to partial balance ratios (0.3, 0.5, 0.7), consistent with findings in **?**.

Model Complexity: Expanded Random Forest from 200 to 800 estimators with depth increased from 20 to 40. Neural Network expanded from 2 layers (128-64) to 5 layers (1024-512-256-128-64) with Batch Normalization. Models now achieve sufficient training performance, confirming they can learn the data patterns.

Feature Engineering: Created 25+ new features including temporal patterns, behavioral aggregates, interaction terms, and risk indicators. Feature importance analysis confirms these successfully capture fraud patterns beyond geographic location.

Cost-Sensitive Learning: Implemented explicit class weights and comprehensive threshold optimization with configurable false negative/false positive costs. Successfully achieved target performance of 26.3% precision with 61.3% recall through the weighted ensemble strategy.

Advanced Techniques: Added XGBoost with GPU acceleration, implemented weighted ensemble combining recall and precision specialists following **?**, and integrated SHAP analysis (**?**) for interpretability.

Customer Location Analysis: Conducted thorough leakage analysis showing correlation of 0.26 with target, confirming genuine geographic patterns rather than data leakage. Successfully reduced overreliance through diversified feature engineering.

7.2 Key Insights Gained

The importance of methodologically sound evaluation cannot be overstated. Applying resampling before splitting would have yielded F1-scores of 0.80+, but such metrics would be misleading. Our conservative approach provides realistic performance expectations for production deployment.

The two-stage ensemble strategy proved highly effective. Combining high-recall models (catching 92-93% of fraud) with high-precision validators (66-67% precision) through weighted voting achieved 61.3% recall while maintaining 26.3% precision, demonstrating that architectural design can balance competing objectives.

Our findings showed that feature engineering quality can matter more than model complexity. The New Account High Value interaction feature alone contributes to 27% of model importance, more than any original feature. This shows that domain knowledge encoded through feature interactions can be more valuable than simply increasing model size.

8 Error Analysis

8.1 Confusion Matrix Interpretation

From Table **??** and Figure **??**, the ensemble model produces 4,286 false negatives (38.7% of fraud missed) and 18,993 false positives (9.05% of legitimate transactions flagged). In business terms, this means we detect 6 out of 10 frauds, but approximately 1 in 11 legitimate customers face additional verification.

8.2 False Negative Analysis

We analyzed the 4,286 missed frauds and identified patterns, following error analysis methodologies from **?**:

Established Account Fraud (41%): Fraudulent transactions from accounts $>$ 90 days old with normal transaction velocity. These "sleeper accounts" or compromised legitimate accounts mimic normal behavior closely, making them extremely difficult to detect without additional signals like device fingerprinting or behavioral biometrics (**?**).

Small-Amount Fraud (28%): Fraudulent transactions below the 50th percentile in amount. Small frauds generate less suspicion and may be testing stolen credentials before larger purchases. Our models which are optimized for high-value fraud detection miss these.

Geographic Camouflage (19%): Frauds occurring in low-risk locations ($<$ 3% fraud rate historically). Fraudsters using VPNs or proxies to mask location, or operating in regions with low fraud history, evade geographic risk indicators.

Temporal Camouflage (12%): Frauds during typical shopping hours (9 AM - 9 PM weekdays). By avoiding suspicious times (late night, weekends), these transactions in blend with normal patterns.

8.3 False Positive Analysis

The 18,993 false positives show different patterns:

New Customer High-Value Purchases (34%): Legitimate first-time or recent customers ($<$ 30 days) making large purchases ($>$ 95th percentile). These trigger our highest-importance fraud indicator (New Account High Value) but are actually legitimate new customers making significant purchases.

Gift Purchases (23%): Legitimate transactions with shipping address different from billing (Address Mismatch flag), often indicating gift purchases. Our model cannot distinguish gifts from fraudulent address mismatches without additional context.

Unusual Shopping Patterns (21%): Legitimate customers showing sudden transaction velocity increases or amount deviations (buying more frequently or at higher amounts than their history). Life events like moving, holidays, or seasonal shopping create these anomalies.

Weekend High-Value Purchases (14%): Legitimate large purchases on weekends, triggering the Weekend High Value interaction feature. Weekend shopping spikes are legitimate behavior but also correlate with fraud.

Geographic Risk Zones (8%): Legitimate customers in high-fraud locations. These customers are penalized by geographic risk indicators despite being legitimate.

8.4 Model-Specific Patterns

Logistic Regression: Excels at linear pattern detection (new account + high amount = fraud) but struggles with complex interactions. Achieves highest recall (92.9%) by being extremely aggressive, but produces 95,071 false positives (45.3% of legitimate transactions flagged).

Random Forest: Similar recall profile (92.0%) with slightly better precision (9.9%) due to non-linear interaction capture. Still produces 92,797 false positives. Both recall models are unsuitable for standalone deployment but excellent for Stage 1 detection.

Neural Network: Achieves best precision among individual models (66.2%) with moderate recall (25.7%). The deep architecture captures complex multi-feature patterns but requires large amounts of data to learn effectively. Focal Loss helps focus on hard examples while maintaining high precision.

XGBoost: Similar high precision to Neural Network (66.4%) with slightly lower recall (24.3%) and better computational efficiency. Gradient boosting naturally handles class imbalance better than standard decision trees.

Ensemble: By combining recall models' wide net with precision models' validation, false positives drop dramatically (95K $\rightarrow$ 19K) while maintaining reasonable recall (92% $\rightarrow$ 61%). This demonstrates the power of architectural complementarity.

8.5 Future Recommendations

Based on our error analysis, the following are some recommendations we have for future implementations.

Behavioral Sequence Modeling: Implement LSTM/GRU models to capture temporal transaction sequences rather than treating each independently (**?**). This would help detect gradually escalating fraud patterns and account takeover scenarios.

Additional Context Features: Integrate device fingerprinting, IP reputation scores, email domain age, and shipping address validation APIs (**?**). These external signals could reduce false positives from legitimate new customers and improve detection of sophisticated fraudsters.

Customer Segmentation: Build separate models for new vs. established customers, high-value vs. low-value transactions, and different geographic risk zones. Specialized models can achieve better performance within each segment.

Feedback Loop Integration: Implement active learning where fraud investigators' decisions (confirming or rejecting alerts) continuously retrain the model. This creates a virtuous cycle of improvement as real-world fraud patterns are incorporated.

Ensemble Threshold Tuning: The current 70-30 weighting (recall-precision) could be adjusted based on business needs. More risk-averse operations might use 80-20, while cost-sensitive operations might use 60-40.

9 Team Contributions

Viransh led the implementation of advanced feature engineering, creating all temporal, behavioral, and interaction features including the critical New Account High Value interaction. He developed the comprehensive threshold optimization function with configurable cost parameters and coded all GPU-accelerated implementations using PyTorch and RAPIDS cuML. Viransh also implemented the XGBoost model with hyperparameter tuning, conducted the Customer Location leakage analysis, and optimized the ADASYN resampling pipeline.

Ellen focused on model architecture design and evaluation framework. She implemented the weighted ensemble strategy combining recall and precision specialists, expanded the neural network to 5 layers with Batch Normalization and Focal Loss, and developed the two-stage detection-validation architecture. Ellen conducted comprehensive error analysis on false positives and false negatives, created all visualizations (ROC curves, PR curves, confusion matrices, comparison plots), integrated SHAP for interpretability analysis, and performed systematic comparison of different optimization strategies. She also wrote the methodological justification for avoiding data leakage in resampling.

Both team members collaborated on code review and report/code writing.

References

Surendranadha Reddy Byrapu Reddy, Praneeth Kanagala, Prabu Ravichandran, Dr Rahul Pulimamidi, P.V. Sivarambabu, and Naga Simhadri Apparao Polireddi. 2024. Effective fraud detection in e-commerce: Leveraging machine learning and big data analytics. *Measurement: Sensors*, 33:101138.

Kanishka Ghosh Dastidar, Olivier Caelen, and Michael Granitzer. 2024. Machine learning methods for credit card fraud detection: A survey. *IEEE Access*, 12:158939–158965.

Emmanuel Ileberi, Yanxia Sun, and Zenghui Wang. 2022. A machine learning based credit card fraud detection using the ga algorithm for feature selection. *Journal of Big Data*, 9(1):24.

Kanika, Jimmy Singla, Ali Bashir, Yunyoung Nam, Najam Hasan, and Usman Tariq. 2022. Handling class imbalance in online transaction fraud detection. *Computers, Materials & Continua*, 70:2861–2877.

Michael A. Lones. 2024. Avoiding common machine learning pitfalls. *Patterns*, 5(10):101046.

Amit Malhotra, Bhupendra Singh Hada, Anchal Mishra, Chandan, and Md Shaik Amzad Basha. 2025. Credit card fraud detection with adasyn oversampling and shap-based interpretability: A comparative ensemble approach. In *2025 6th International Conference on Intelligent Communication Technologies and Virtual Mobile Networks (ICICV)*, pages 891–899.

Eryu Pan. 2024. Machine learning in financial transaction fraud detection and prevention. *Transactions on Economics, Business and Management Research*, 5:243–249.

Md. Shafiuddin Shajib. 2025. Fraud detection ecommerce transaction. `https://www.kaggle.com/code/mdshafiuddinshajib/fraud-detection-ecommerce-transaction`. Kaggle notebook.

Ishan Sohony, Rameshwar Pratap, and Ullas Nambiar. 2018. Ensemble learning for credit card fraud detection. *Knowledge-Based Systems*, pages 289–294.

P. Sundaravadivel, R. Augustian Isaac, D. Elangovan, D. KrishnaRaj, V. V. Lokesh Rahul, and R. Raja. 2025. Optimizing credit card fraud detection with random forests and smote. *Scientific Reports*, 15(1):17851.

Qinghong Yang, Xiangquan Hu, Zhichao Cheng, Kang Miao, and Xiaohong Zheng. 2015. Based big data analysis of fraud detection for online transaction orders. *Lecture Notes of the Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering, LNICST*, 142:98–106.

G K Yuvaraaj and K Dhinakaran. 2025. Shap-enhanced hybrid model for accurate financial fraud detection. In *2025 International Conference on Data Science, Agents Artificial Intelligence (ICDSAAI)*, pages 1–5.