|  |  |
| --- | --- |
| **Fraud Detection Using Machine Learning** | Abstract  This project presents a machine learning solution for detecting fraudulent financial transactions. A large, highly imbalanced dataset was pre-processed and engineered to create meaningful features, including balance differences and error balances. Logistic Regression was used as a baseline, while a Random Forest model trained on SMOTE-balanced data achieved near-perfect detection of fraud cases. The final model demonstrates high precision, recall, and ROC-AUC, providing a reliable and deployable system for real-world fraud detection.  Pratyaksh Yadav |

**Fraud Detection Using Machine Learning**

**Submission Information:**

**Task:** Fraud Detection Using Machine Learning

**Date:** 20th August ,2025

**Organization:** Accredian

**Dataset Information:**

* **Source:** Provided dataset for fraud detection task
* **Number of Rows:** ~2.1 million
* **Number of Features:** 14–16 numeric and categorical columns
* **Target Column:** isFraud (1 → fraud, 0 → non-fraud)
* **Class Imbalance:** Fraud is very rare (~0.16%)
* **Key Columns:** Step, type, amount, oldbalanceOrg, newbalanceOrig,  
  oldbalanceDest, newbalanceDest, isFraud, isFlaggedFraud

**Objective:**

* Detect fraudulent transactions accurately.
* Handle class imbalance and rare-event prediction.
* Compare baseline model (Logistic Regression) with an advanced model (Random Forest).

**Introduction:**

Fraud detection in financial transactions is a critical task to prevent monetary losses and maintain security. The challenge lies in detecting rare fraudulent activities within a large volume of legitimate transactions. This project focuses on creating a machine learning pipeline that identifies fraud effectively using data preprocessing, feature engineering, and model training.

**Key Steps Performed:**

* Analyzed the dataset and handled missing or zero balances.
* Engineered features like balance differences (diffOrig, diffDest) and error balances (errorBalanceOrig, errorBalanceDest).
* Transformed features (log\_amount) and encoded categorical variables.
* Split data into training and testing sets with stratification to preserve fraud ratios.
* Standardized numeric features for model consistency.
* Handled class imbalance using SMOTE and/or class weights.
* Trained a baseline Logistic Regression model and evaluated performance.
* Built an advanced Random Forest model for improved fraud detection.
* Saved trained model and scaler for deployment and reproducibility.

**Data Preprocessing:**

Before modelling, it is essential to clean and transform the data into meaningful features. This step ensures that the model can correctly interpret transaction patterns and detect fraudulent behaviour.

* Created binary flags for zero balances: isZeroOldBalance, isZeroNewBalanceDest.
* Calculated differences to capture inconsistencies: diffOrig = oldbalanceOrg - newbalanceOrig - amount, diffDest = oldbalanceDest + amount - newbalanceDest.
* Computed error balances: errorBalanceOrig, errorBalanceDest.
* Applied log transformation to amount → log\_amount.
* Dropped identifiers (nameOrig, nameDest) and encoded type column.

**Train-Test Split and Scaling:**

To evaluate the model properly, we split the dataset into training and test sets, ensuring the distribution of fraud cases is preserved. Standardizing numeric features helps models converge faster and improves performance.

* Split dataset into 70% training and 30% test with stratification on isFraud.
* Standardized numeric features using StandardScaler.

**Handling Class Imbalance:**

Fraud is extremely rare in the dataset. Balancing the classes ensures the model learns to detect fraud effectively without being biased towards non-fraud cases.

* Applied SMOTE to generate synthetic fraud samples for the training set.
* Ensured balanced representation of both fraud and non-fraud classes.

**Logistic Regression (Baseline):**

A baseline model provides a simple, interpretable starting point and helps set expectations for more advanced models.

* Trained with class\_weight="balanced".
* Evaluated using classification report, ROC-AUC, and confusion matrix.
* Performance: High accuracy for non-fraud; moderate recall for fraud, many false positives.
* ROC-AUC: ≈ 0.99.
* Observation: Good baseline but limited for detecting rare frauds.

**Random Forest (Advanced Model):**

To improve detection of rare frauds, an ensemble method like Random Forest captures complex patterns and interactions among features.

* Model Setup: 200 trees, class\_weight="balanced", n\_jobs=-1.
* Training: On SMOTE-balanced data (X\_train\_res, y\_train\_res).
* Evaluation:
  + Confusion Matrix: [[1,906,225, 97], [9, 2,455]]
  + Precision (fraud): 0.96
  + Recall (fraud): 1.00
  + F1-score (fraud): 0.98
  + ROC-AUC: 0.999
* Feature Importance: errorBalanceOrig, diffDest, log\_amount most predictive.
* Observation: Excellent performance; almost all frauds detected with very few false positives.

**Model Export:**

Saving the model and preprocessing objects ensures reproducibility and allows deployment in production environments.

* Saved Random Forest model: random\_forest\_fraud\_model.pkl.
* Saved scaler: scaler.pkl.
* Ensures reproducibility and deployment readiness.

**Conclusion:**

The modelling workflow demonstrates the importance of preprocessing, feature engineering, and handling imbalance for fraud detection.

* Logistic Regression provided a baseline but missed some fraud cases.
* Random Forest greatly improved detection metrics with near-perfect recall and ROC-AUC.
* Feature engineering (balance errors, differences, log amount) was critical for high performance.
* The trained model and scaler are ready for deployment or further evaluation.

**Questions Asked:**

**1. Data cleaning including missing values, outliers, and multi-collinearity**

In this project, the dataset was first examined for missing values and anomalies. Transactions where sender or receiver balances were zero were flagged for investigation, and features were engineered to capture these irregularities. Outliers in the amount column were addressed using a log transformation to reduce skewness and make patterns more detectable by the model. Additionally, redundant or irrelevant features such as account identifiers were removed, and categorical variables were encoded for modelling. While multi-collinearity was considered, feature engineering focused on creating meaningful, independent variables to minimize redundancy and improve the predictive power of the model.

**2. Describe your fraud detection model in elaboration**

Two models were developed to detect fraud. The first was a Logistic Regression model, which served as a baseline. It used class weighting to handle the imbalance in fraud cases and provided initial insights into feature importance. To improve detection, a Random Forest model was trained on SMOTE-balanced data, allowing the model to learn from synthetic fraud cases and better capture complex patterns. The Random Forest model, with 200 decision trees, was able to account for nonlinear relationships and interactions between features, resulting in superior performance. Both models were evaluated using classification metrics and ROC-AUC to measure how well they distinguish fraud from non-fraud transactions.

**3. How did you select variables to be included in the model?**

Variable selection was driven by domain understanding and exploratory analysis. Features capturing transaction anomalies, such as differences between old and new balances for both sender and receiver, were created to highlight unusual activities. Binary indicators for zero balances were included to flag suspicious cases, and a log transformation of the transaction amount was applied to handle skewed distributions. Categorical variables, like transaction type, were encoded numerically. This approach ensured that the model included features with clear relevance to detecting fraud while avoiding irrelevant or redundant variables.

**4. Demonstrate the performance of the model by using best set of tools**

The models were evaluated using a combination of metrics. Logistic Regression achieved reasonable accuracy but struggled with the rare fraud cases, producing some false positives. The Random Forest model performed exceptionally well on the test set, detecting nearly all fraud cases while maintaining very few false positives. Key metrics included a precision of 0.96, recall of 1.00, F1-score of 0.98 for fraud, and an ROC-AUC of 0.999. Confusion matrices and classification reports were used to visually and numerically assess the model’s performance, confirming its reliability for real-world deployment

**5. What are the key factors that predict fraudulent customer?**

Feature importance analysis from the Random Forest model highlighted several key predictors of fraud. The difference and error balances for the sender and receiver (errorBalanceOrig, diffDest) were most influential, indicating unusual changes in account balances. High transaction amounts, captured through log\_amount, were also a strong indicator. These features collectively allowed the model to differentiate between normal and suspicious transactions effectively.

**6. Do these factors make sense? If yes, How? If not, How not?**

Yes, these factors make sense from a financial perspective. Fraudulent transactions often involve unusual discrepancies in account balances, such as sending more money than is available or sudden drops in balances. Large transaction amounts are also more likely to attract fraudulent activity. Therefore, the identified features are consistent with real-world patterns of fraudulent behaviour, which validates the model’s focus on these indicators.

**7. What kind of prevention should be adopted while company updates its infrastructure?**

When updating transaction infrastructure, companies should implement real-time monitoring and anomaly detection systems to catch suspicious activity as it occurs. Automated alerts for unusual balance changes, periodic model retraining, and rigorous testing of new system features can prevent fraud from slipping through gaps in the infrastructure. Additionally, security protocols should be regularly reviewed to maintain robust defences against both known and emerging fraud tactics.

**8. Assuming these actions have been implemented, how would you determine if they work?**

The effectiveness of these preventive measures can be assessed by monitoring key performance metrics over time. Metrics such as recall and precision for fraud detection, the false positive rate, and the number of successfully intercepted fraudulent transactions should be tracked. Comparing these metrics before and after infrastructure updates will indicate whether the implemented measures are reducing fraud and improving system reliability. Periodic audits and validation of model predictions against actual outcomes will provide further confirmation of success.