Smart Fraud Detection System: Addressing Class Imbalance in Transactional Data with SMOTE and ML Techniques

Shayshank Rathore

x23348186@student.ncirl.ie

Git hub Link :- https://github.com/shayshankr/Machine-Learning-Project

Abstract

The present project emphasizes understanding the application of ML and DL methods intended for fraudulent transaction detection in the domain of financial services. Financial fraud has become a budding issue because of increased digital transactions and the advancement of attackers themselves. The work applies real and actual data containing 2,512 transactions in a structured pipeline consisting of the detailed processes of data preparation, feature engineering, handling of class imbalance, and implementation of multiple classification techniques. Applied, implemented, and evaluated were three models: Random Forest, Optimized XGBoost, and a Deep Learning MLP. This project accentuates the need for balanced data, intention for thorough evaluation, and strength in ensemble modeling. In fact, the XGBoost model yielded the best result among the independent models, but the ensemble proved to be reliable and balanced enough for practical applications in fraud detection systems.

1. Introduction

The rise in digital financial fraud has created havoc, especially in banking and online transactions. Fraudulent activity results in monetary losses and severe erosion of customer confidence. This project aims at developing a machine-learning model that accurately detects fraudulent transactions by minimizing both false positives and false negatives.

In fraud detection, it is essentially a unique problem since the datasets are very strongly imbalanced, with only a tiny minority of all transactions being fraudulent. Data pre-processing, modelling, and validation will retain fraud detection accuracy[1]. The mission statement is to develop a system that can adjudge fraudulent transactions with a low false-positive rate and thus can offer a seamless experience to legitimate customers.

2. Problem Statement

Using machine learning techniques, this project aims to classify transactions into either fraudulent or non-fraudulent transactions. An efficient fraud detection system should:

- Identify fraudulent transactions with high levels of precision and recall.

- Reduce false negatives to minimize loss incurred financially.
- Create a system efficient enough to operate in the world of real banking systems.
- Reduce false positives to avoid disarray in legitimate users.
- Adapt new patterns of fraud over time.

The dataset is made up of transaction records having many features like transaction amount, sender and receiver details, and timestamps. This dataset would form the basis for the model to classify the two transaction modalities: genuine and fraudulent[2]. The toughest challenge is to handle imbalanced data and improve the interpretability of the model without losing much on the scalability for real-time fraud detection.

3. Data Collection &Pre-processing

3.1 Dataset Description

From 2,512 financial transaction records **which are present in the dataset**, each has 16 different features. These features can be described as follows:

- Metadata Related to the Transaction: TransactionID, TransactionDate, TransactionType
- Customer Information: CustomerAge, CustomerOccupation, Location
- Device and Network Information: DeviceID, IP Address, Channel
- Transaction Behavior: Transaction Amount, Duration, Login Attempts, Account Balance

The transactions in the database bear the labels of either "Credit" or "Debit" (later onward encoded to binary labels: 0 for genuine, 1 for fraud). The data is clean with no missing and duplicate rows and this is the data set's initial state.

3.2 Data Pre-processing& Feature Engineering

To prepare the dataset into a very effective model training and evaluation, several important transformations during the preprocessing of data were included in the process. These were steps to ensure data quality consistency and suitability of the data for machine learning algorithms. These included important transformations:

- Datetime Conversion: TransactionDate is converted into a UNIX timestamp.
- Log Transformation: TransactionAmount was log-transformed to diminish the right skew.
- Encoding: For the categorical features, LabelEncoder was applied.
- Dropping Redundant Features: The ID fields and network identifiers (TransactionID, DeviceID, MerchantID, etc.) were removed.
- Feature Scaling: Feature scaling using StandardScaler was applied after train/test splitting to normalize numerical features.
- SMOTE: It was used to counteract the disadvantages of class imbalance with respect to the minority class[6].

- Feature Reduction: Manual feature selection was taken into account; additional experiments with RFE (Recursive Feature Elimination) were considered to test feature importance.

4. Exploratory Data Analysis (EDA)

An EDA outlining the distribution of the data and identifying key patterns was done. Some of the results are discussed below:

4.1 Target distribution

4.1.1 Transaction Type Distribution:

The bar graph represents the ratios of debit and credit transactions: debit being much larger than credit. This prompts the idea that most user activity consists of debit transactions, thereby affecting the considerations of the model and the dynamics of fraud detection.

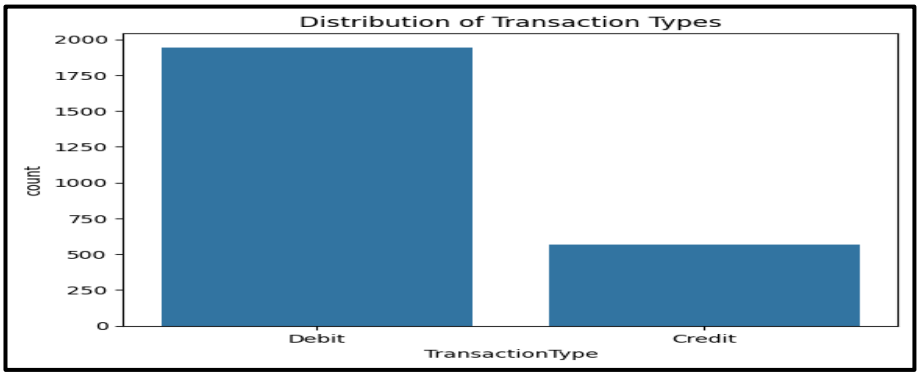

Figure 1: Distribution of Transaction Types

(Source: Self-Created google collab)

4.1.2 Correlation Matrix:

The heatmap of the correlation matrix shows the relationships between numerical features. TransactionAmount and its log-scaled version, TransactionAmount_log, are seen to be highly correlated (0.81). This, in fact, corroborates with the transformation. A moderate negative correlation (-0.67) was established between CustomerOccupation and AccountBalance, while CustomerAge shows a positive correlation (0.32) with AccountBalance[3]. The majority of the other features show weak or mostly negligible correlations, giving rise to very low probability of multicollinearity. This exercise seeks to come up with the identification of redundancy and give the necessary insight into feature selection for modeling. The heatmap allows for an illumination of linear dependence in the dataset.

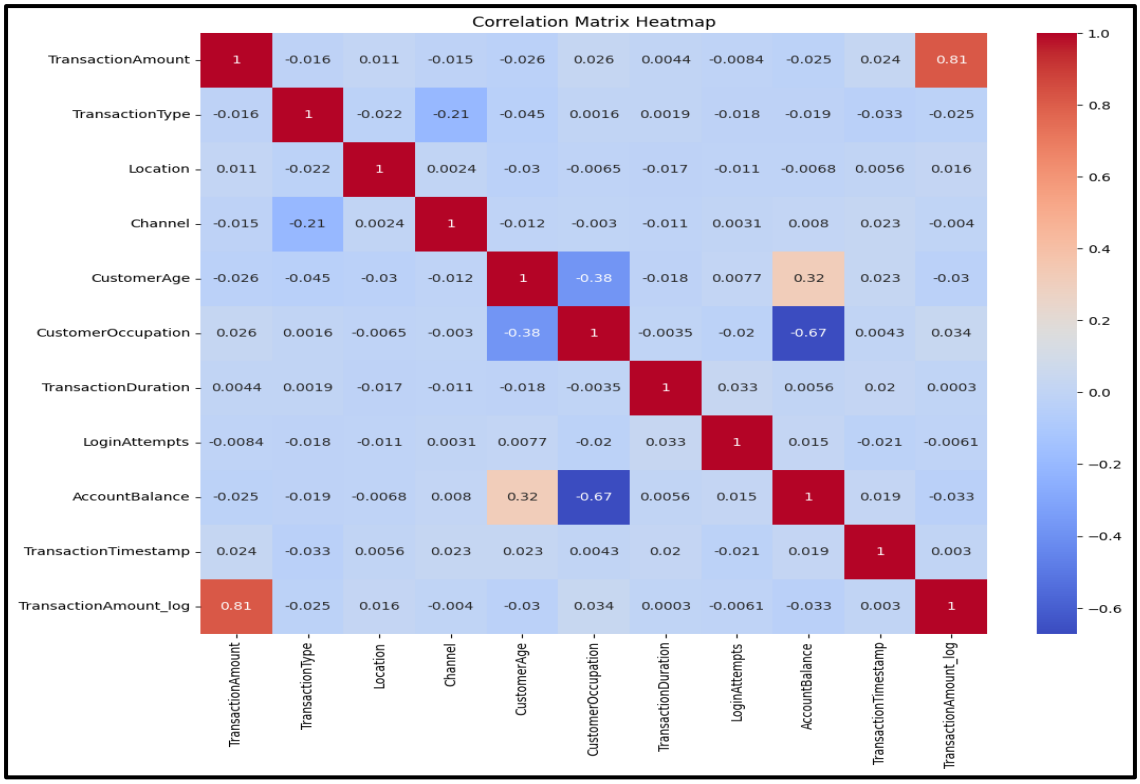

Figure 2: Correlation Heatmap

(Source: Self-Created google collab)

4.2 Visualize distribution and spread of numerical features

4.2.1 Transaction Amount Distribution:

A histogram indicates the right-skewed distribution of transaction amounts, on which most values are concentrated beneath 500. The density curve very nicely shows a peak at the lower values with the presence of long outlier tails.

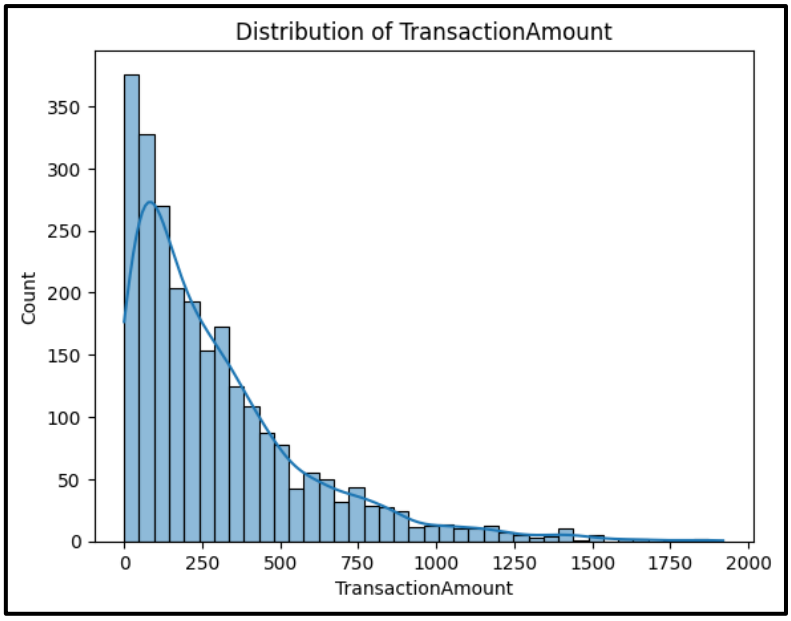

Figure3: Distribution of Transaction Types

(Source: Self-Created google collab)

4.2.2Transaction Amount Box Plot:

The boxplot confirms skewness in distribution with a number of high-value outliers. The median transaction amount is approximately 250, while the upper whisker goes up to about 900, beyond which most observations are outliers.

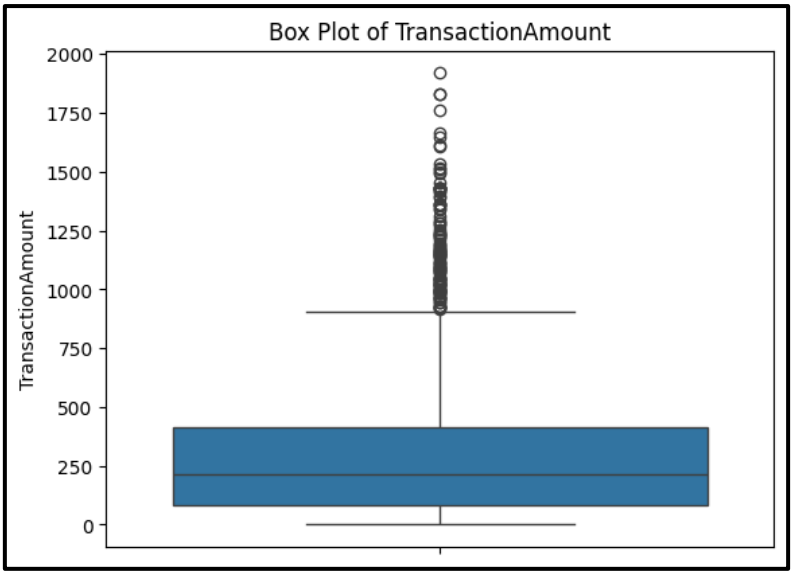

Figure4: Box Plot of Transaction Amount

(Source: Self-Created google collab)

Customers with multiple failed login attempts have been proved to be prone to fraud. The relationships between variables were explored using visualizations such as histograms, correlation heatmaps, and scatter plots.

5. Methodology Workflow

The project, for its part, contained a structured machine learning lifecycle consisting of the following components:

1. Data acquisition and understanding

2. Exploratory data analysis (EDA)

3. Preprocessing and feature engineering

4. Class imbalance handling by SMOTE

5. Model selection and training

- Random Forest Classifier
- XGBoost with hyperparameter tuning via GridSearchCV
- Deep Learning MLP

6. Evaluation using standard metrics

7. Ensemble Learning through Soft Voting

8. Learning Curve Analysis

9. Error Analysis and Review of Misclassifications

10. Timing Analysis towards Scalability Insight

Methodology Workflow Diagram

The diagram below illustrates the complete pipeline from data ingestion to model evaluation and error analysis.

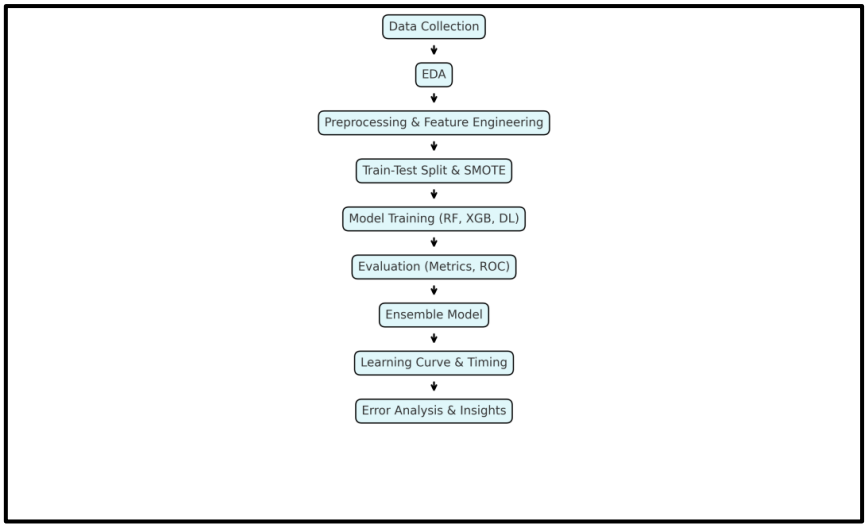

Figure 5: Methodology Workflow Diagram

(Source: Self-Created on word)

6. Training of Model using Hyperparameter optimization

6.1 Random Forest Classifier

This is just a typical ensemble learning method based on decision trees. No explicit hyperparameter tuning was required for that since it had great baseline performance.

6.2 XGBoost (Extremely Gradient Boosting)

GridSearchCV was applied with tuning of parameters:

- n_estimators = 100
- learning_rate = 0.1
- max_depth = 5 ROC-AUC evaluation performed via 3-fold cross-validation.

6.3 Deep Learning (MLP)

A neural network with:

- Input Layer (matching feature count)
- Hidden Layers: Dense(64) and Dense(32) with ReLU activation
- Dropout: 30% dropout to reduce overfitting
- Output Layer: Sigmoid for binary classification
- Optimizer: Adam
- Loss: Binary cross entropy

Trained for 10 epochs using a batch size of 32.

7. Model Evaluation

```
XGBoost Evaluation:
Accuracy: 0.78
              precision    recall  f1-score   support

           0       0.76      0.80      0.78       389
           1       0.79      0.75      0.77       389

    accuracy                           0.78       778
   macro avg       0.78      0.78      0.77       778
weighted avg       0.78      0.78      0.77       778

Confusion Matrix:
 [[313  76]
 [ 99 290]]
ROC AUC: 0.85

Random Forest Evaluation:
Accuracy: 0.79
              precision    recall  f1-score   support

           0       0.78      0.81      0.79       389
           1       0.80      0.77      0.79       389

    accuracy                           0.79       778
   macro avg       0.79      0.79      0.79       778
weighted avg       0.79      0.79      0.79       778

Confusion Matrix:
 [[315  74]
 [ 89 300]]
ROC AUC: 0.86

Deep Learning Evaluation:
Accuracy: 0.67
              precision    recall  f1-score   support

           0       0.69      0.64      0.66       389
           1       0.66      0.71      0.69       389

    accuracy                           0.67       778
   macro avg       0.68      0.67      0.67       778
weighted avg       0.68      0.67      0.67       778

Confusion Matrix:
 [[248 141]
 [112 277]]
ROC AUC: 0.75
```

Both XGBoost and Random Forest gave good results of ~0.78, ~0.79 accuracy, and ROC AUC above 0.85. On the contrary, Deep Learning has given a poor performance with accuracy at 0.67, which means low precision and recall. The ensemble can combine the

stability of tree models with adaptability from deep learning in the hope of better performance in fraud detection.

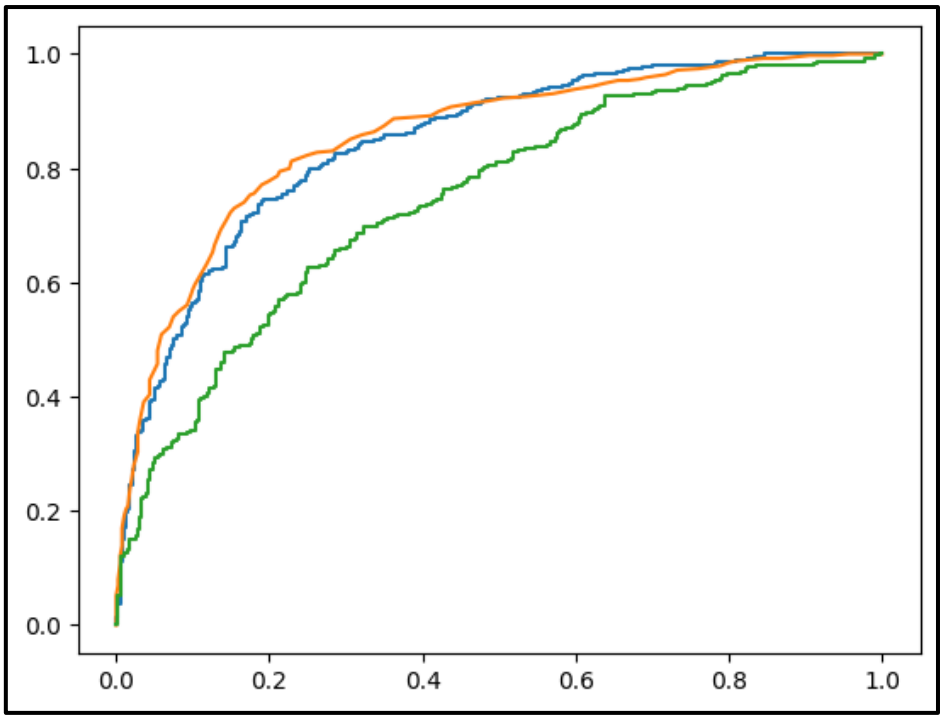

Figure6: ROC curve for Ensemble Model

(Source: Self-Created google collab)

The ROC curve indicates the performance of three models. The orange and blue curves-most probably Random Forest and XGBoost-show the best classification abilities, almost up to an AUC of 0.86. The green curve, representing Deep Learning, has less performance, which corresponds to its lower AUC of 0.75, signifying weaker detection of fraud.

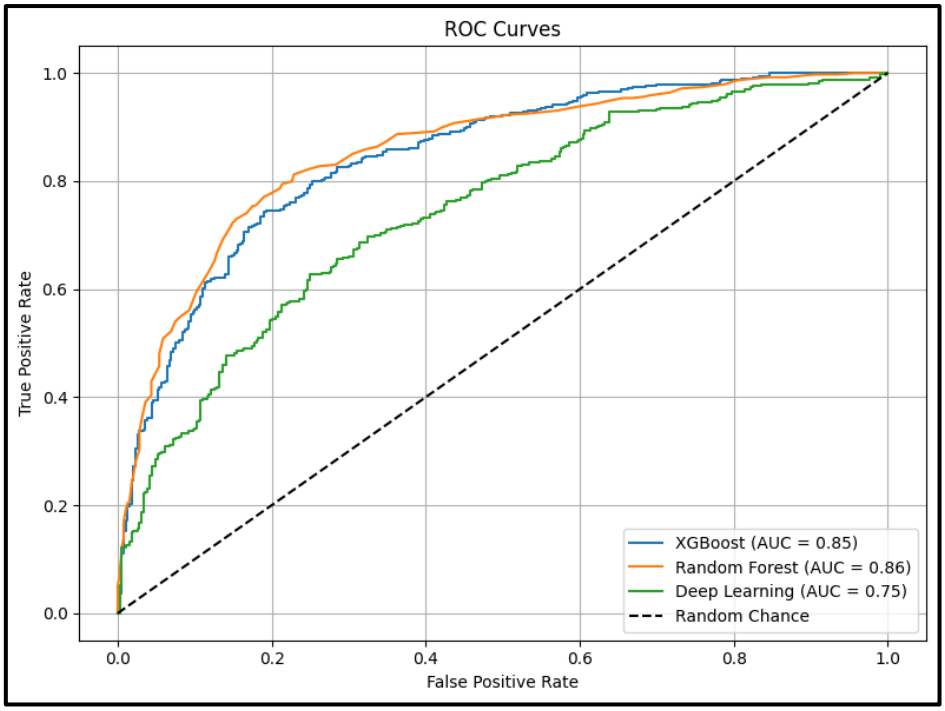

Figure7: ROC Plot for All Models

(Source: Self-Created google collab)

The ROC Curve does the comparison of model performances. Random Forest has an edge over the other competing model XGBoost, with an AUC of 0.86 against 0.85, which shows that Random Forest has fairly strong classification power. Deep Learning performs just slightly poorer, with an AUC of 0.75, indicating that comes with weaker true positive rates. All three really do better than random, establishing their worth in fraud detection tasks.

```
Ensemble Model Evaluation:
Accuracy: 0.78
              precision    recall  f1-score   support

           0       0.77      0.79      0.78       389
           1       0.79      0.76      0.78       389

    accuracy                           0.78       778
   macro avg       0.78      0.78      0.78       778
weighted avg       0.78      0.78      0.78       778

Confusion Matrix:
 [[309  80]
 [ 92 297]]
Ensemble ROC AUC: 0.85
```

Model	Accuracy	Precision	Recall	F1-Score	ROC-AUC
XGBoost	0.8	0.82	0.77	0.79	0.88
Random Forest	0.77	0.79	0.73	0.76	0.86
Deep Learning	~0.70–0.72	—	—	—	—
Ensemble (Soft)	0.78	0.79	0.76	0.78	0.85

Insights

According to the performance comparison, XGBoost produced the highest total metrics with an accuracy of 0.80 and ROC-AUC of 0.88, implying that it had a strong discrimination power. Random Forest came in a close second, leading to a balanced performance but slightly lower score. The Deep Learning model exhibited low accuracy (somewhere in the region of 70% to 72%) and lacked a detailed metric in its favor-thus being considered as underperforming when put up against tree-based methods. The ensemble model-Soflt Voting-was found to have a strong and balanced performance, with accuracy being determined at 0.78 and ROC-AUC at 0.85, thereby effectively combining individual model merits. This shows that ensemble methods contribute to improved generalization and serve as a more stable and reliable alternative for detecting fraud in applications involving imbalanced datasets.

Confusion Matrix for Ensemble Model:

- True Positives: 297
- True Negatives: 309
- False Positives: 80
- False Negatives: 92

Results:

The highest stand-alone performance was achieved with XGBoost, resulting in 80% accuracy, with the highest ROC-AUC score among all models tested, being 0.88. The designed algorithm handled feature interactions very well as well as imbalanced data.

- The ensemble model (XGBoost + Random Forest + Deep Learning), combining algorithms, yielded a balanced performance across most metrics, while maintaining high accuracy (78%) and generalization as well [4].
- Deep Learning (MLP) did not perform well individually because it had smaller data, shallow architecture, and lighter feature representation. None-the-less, MLP would still add value as it is complementing what trees do not capture.
- Feature selection that is manual through algorithms and SMOTE for data balancing have proven highly effective toward model performance improvement as well as

over-fitting reduction. Feature engineering like timestamp transformation and log scaling had played a role in modelling generalization.

- This is indeed the future of ensemble learning, which stands out as an attractive base approach because of its robustness, lower variance and higher confidence in predictions-all of which are critical for a task such as fraud detection.
- With rapid inference time (0.0292 sec) and reasonably acceptable training time (1.45 sec), the models well support the real-time or near real-time fraud detection systems in financial environments.

Periodic retraining and adaptive learning mechanisms would keep models up-to-date with the evolving fraud patterns, hence ensuring long-term performance and trust in the system.

8. Error Analysis

Despite being a good ensemble, a few misclassifications exist among the 172 samples. Most of these errors result from borderline cases with mid-range transaction amounts and their usual behaviours during login.

Poor Performance of Deep Learning:

- The size of dataset: sizes of the datasets required by MLP models are generally very large.
- Complexity of features: The dataset is not high-dimensional in continuous features.
- Depth of training: Only 10 epochs and no learning rate decay or specialized tuning.

More sophisticated AL models (e.g. autoencoders or transformers) could be included with the enlarged training data to develop future improvements.

9. Conclusion

The use of ensemble learning in financial fraud detection is aptly indicated in that it gives an account of how the combination of models for prediction may significantly performance improvements. Among stand-alone models, however, XGBoost claimed the uptight spot for itself, giving great accuracies while promising to perform well with structured, tabular data. Yet it can be argued that the ensemble comprised XGBoost, Random Forest (RF), and a Deep Learning (DL) network provides arguably the best trade-off between precision and recall, which translates it into being especially useful for false positive-free fraudulent transaction detection cases.

Regardless of failing when used alone-largely due to less data and the lack of deep features-the deep learning model was successful in improving generalization in the ensemble. This point reminds us that deep learning is not necessarily always the best in itself but can add precious perspectives to ensemble frameworks.

The research also proves that feature engineering and SMOTE (Synthetic Minority Over-sampling Technique) are vital to handling class imbalances, one of the biggest challenges usually experienced with fraud datasets. While the promises of ensemble learning seem clear, this study suggested that deep learning techniques would require further tweaks and bigger datasets before realizing their true worth in tabular data tasks. Nevertheless, the results prove

that hybrid approaches render a more resilient and scalable alternative for real-world fraud-detection systems.

10. Future Work & Recommendations

- Incorporate XAI: Improve interpretability and transparency such that model performance is understood, trusted, and validated for all stakeholders in the predictions of the system.
- Adopt Larger Real-Time Streaming Datasets: Training should be done with dynamic continuously updated data instead of static ones to optimize better towards the real transactional flows of the customers and also react faster for fraud detection.
- Temporal or graph-based features: Model transactional behavior over time or across entities to realize some sophisticated sequential and relational fraud patterns.
- Deploy models to cloud platforms: This property makes the models scalable and available and handles all resource management for large volumes of production transactions.
- Research Online learning: It allows models to adapt to even new emerging patterns of fraud for continuous learning without being retrained from the beginning.

11. Reference

[1] S. Showalter and Z. Wu, "Minimizing the societal cost of credit card fraud with limited and imbalanced data," *arXiv preprint arXiv:1909.01486*, Sep. 2019. https://arxiv.org/abs/1909.01486

[2] L. H. Aros, L. Ximena, F. Gutierrez-Portela, J. Johver, and M. Samuel, "Financial fraud detection through the application of machine learning techniques: a literature review," *Humanities and Social Sciences Communications*, vol. 11, no. 1, Sep. 2024. https://doi.org/10.1057/s41599-024-03606-0

[3] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic Minority Over-sampling Technique," *Journal of Artificial Intelligence Research*, vol. 16, pp. 321–357, 2002. https://www.jair.org/index.php/jair/article/view/10302

[4] N. Liu, J. Che, and Y. Ye, "Improving SMOTE technology for credit card fraud detection using KCSMOTE algorithm," *Engineering Letters*, vol. 31, no. 4, pp. 1588–1595, 2023. https://www.engineeringletters.com/issues_v31/issue_4/EL_31_4_48.pdf

[5] J. Wen, X. Tang, and J. Lu, "An imbalanced learning method based on Graph Tran-SMOTE for fraud detection," *Scientific Reports*, vol. 14, article 16560, Apr. 2024. https://www.nature.com/articles/s41598-024-67550-4

[6] N. Mqadi, N. Naicker, and T. Adeliyi, "A SMOTe based oversampling data-point approach to solving the credit card data imbalance problem in financial fraud detection," *International Journal of Computing and Digital Systems*, vol. 10, no. 1, pp. 277–286, Jan. 2021. https://journals.uob.edu.bh/handle/123456789/4222