Data Mining Project Report

Strategic Optimization of Credit Card Fraud Detection Using Weighted Deep Learning

Author : Justin Matthew T

Student ID : A11.2023.14877

---

1. Main Objective of Analysis

The primary objective of this project is to develop a robust **Deep Learning model** capable of identifying fraudulent credit card transactions with high sensitivity. In the banking and financial sector, the goal of a fraud detection system is not merely statistical accuracy, but rather the **minimization of Financial Loss**.

Therefore, this analysis focuses on optimizing the **Recall metric** (the ability to catch as many fraud cases as possible). A traditional model that achieves 99.9% accuracy but misses high-value fraud transactions is considered a failure in a business context. The target is to build a model that acts as an effective automated safety layer, blocking suspicious transactions before funds leave the system, while maintaining a manageable level of **False Positives** to ensure customer friction remains within operational limits.

---

2. Data Description & Exploration

Dataset Download : https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud

Google Colab :
https://colab.research.google.com/drive/1peYHGdUBp1gqwQhDb1Xf4SjJpB0Se8vJ#scrollTo=Nfg7dMtTmGww

The analysis utilizes a dataset of European credit card transactions (Source: Kaggle/ULB), covering a period of two days.

- **Data Volume:** The dataset contains **284,807 transaction records**.
- **Feature Engineering:** It includes 30 input features. Features V1 through V28 are the result of a Principal Component Analysis (PCA) transformation to protect user confidentiality. The only non-transformed features are Time and Amount.
- **Key Challenge (Class Imbalance):** The dataset exhibits extreme class imbalance. Fraudulent transactions account for only **0.172% (492 cases)** of the total data. This imbalance poses a significant challenge, as standard models tend to bias towards the majority class (safe transactions).

Preprocessing Actions Taken:

- **Feature Scaling:** The Amount and Time features vary significantly in range compared to the PCA features. To ensure the Neural Network weights converge effectively during Gradient Descent, I applied **Standard Scaling** to normalize these features.

- **Stratified Data Splitting:** The data was split into Training, Validation, and Testing sets using *Stratified Sampling*. This ensures that the rare fraud cases are distributed proportionally across all sets, preventing a scenario where the test set contains no fraud examples.

---

3. Deep Learning Model Variations

To identify the optimal solution, I developed and evaluated three distinct architectural approaches:

- **Model A (Baseline):** A standard **Dense Neural Network** (3 layers). This model serves as a benchmark and was trained without any specific handling for class imbalance.
- **Model B (Optimized Architecture):** An evolution of the baseline, incorporating **Batch Normalization** for training stability and **Dropout Layers (0.3)** to prevent overfitting. This variation tests whether architectural improvements alone can solve the detection problem.
- **Model C (Weighted Neural Network):** This model utilizes the architecture of Model B but integrates a **Cost-Sensitive Learning** technique known as *Class Weighting*. By assigning a significantly higher penalty weight (hundreds of times higher) to misclassifying fraud cases, the model is forced to prioritize the minority class. This effectively aligns the model's loss function with the business reality: missing a fraud is far more costly than flagging a safe transaction.

---

4. Key Findings & Insights

Evaluation on the unseen **Test Set** reveals a critical performance divergence between the Baseline and the Weighted approach.

Visual Performance Comparison:

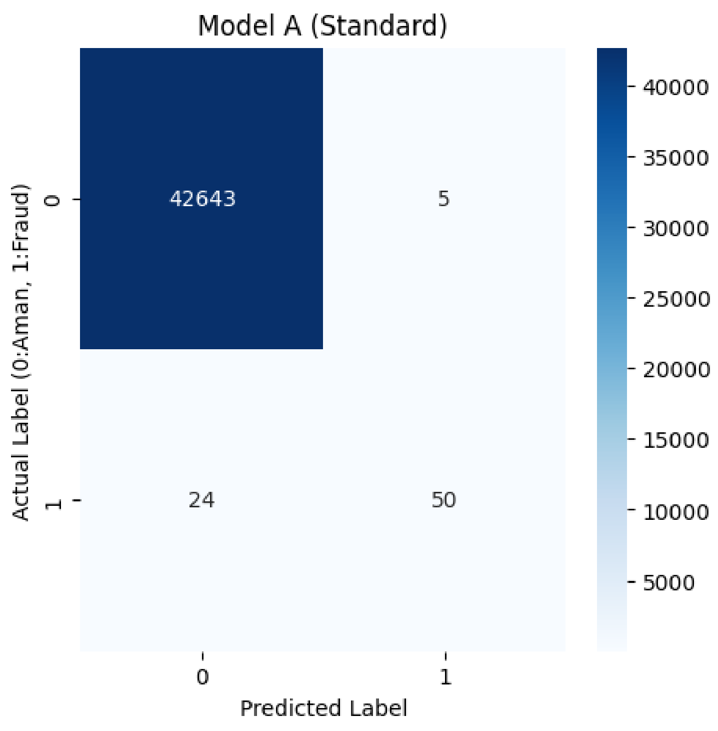

Figure 1: Confusion Matrix - Model A (Baseline)

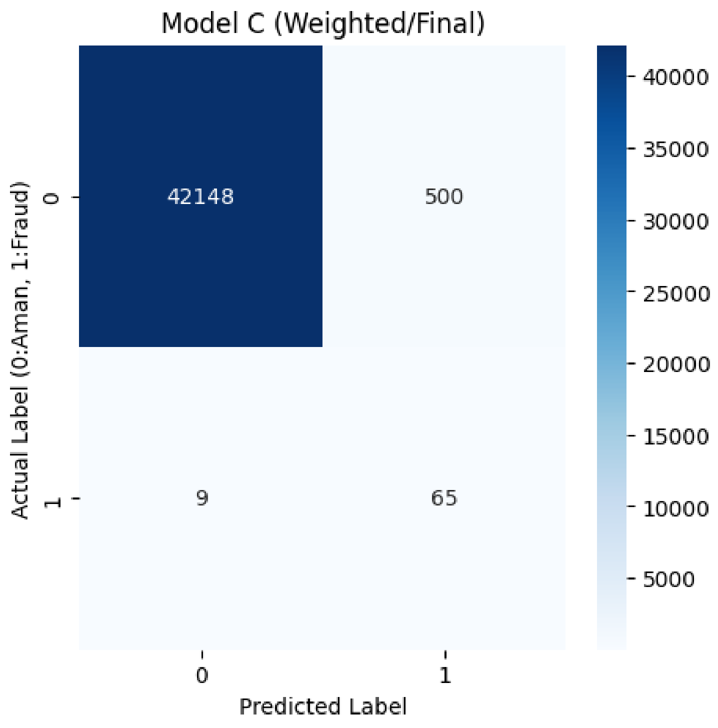

Figure 2: Confusion Matrix - Model C (Weighted/Final)

Analytical Findings:

1. **The Risk of the Baseline (Model A):** As shown in Figure 1, the Baseline model is biased towards the majority class. While it has very few false alarms, it failed to detect **24 fraud cases** (False Negatives). In a production environment, this represents a significant security breach and direct financial liability.
2. **The Effectiveness of Weighting (Model C):** As shown in Figure 2, the Weighted Model demonstrates a drastic improvement in sensitivity. It successfully detected **65 fraud cases**, leaving only 9 undetected. This represents a **~30% improvement in fraud detection** compared to the baseline.
3. **Training Stability:** The Loss Curve indicates that Model C converged well, with Training and Validation losses decreasing in tandem, suggesting the model is robust and not overfitting.

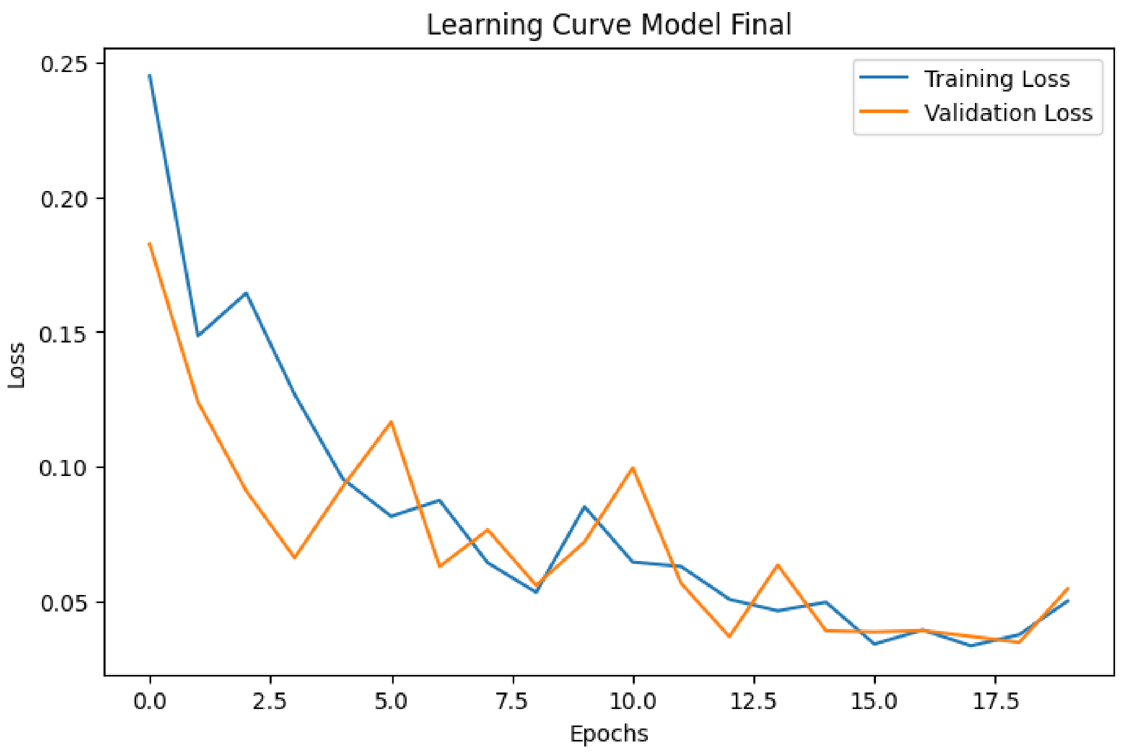

Figure 3: Learning Curve - Final Model

Final Recommendation: I recommend deploying **Model C (Weighted Neural Network)**. While this model generates a higher number of False Positives (~500 transactions), this is a justifiable trade-off. The cost of automating verification (e.g., via SMS or App notification) for these 500 customers is negligible compared to the potential loss from the 15 additional fraud cases that Model C catches but Model A misses.

---

5. Flaws & Plan of Action (Next Steps)

While Model C excels in risk mitigation, there are areas for future refinement:

- **Identified Flaw:** The model produces approximately 500 False Positives. If deployed without filters, this could lead to 500 legitimate transactions being blocked, potentially frustrating customers.

- **Strategic Next Steps:**

  1. **Threshold Moving (Precision-Recall Trade-off):** The current decision boundary is set at the default probability of 0.5. By shifting this threshold to a stricter value (e.g., **0.8** or **0.9**), we can significantly reduce the False Positives without severely impacting the Recall rate. This would fine-tune the balance between security and user experience.
  2. **Sequential Behavior Analysis (LSTM):** The current model treats each transaction as an independent event. Future iterations should implement **Long Short-Term Memory (LSTM)** networks to analyze the *sequence* of a user's transactions. This would allow the system to understand "spending patterns" over time, ensuring that legitimate high-value transactions that fit a user's history are not flagged as anomalies.