Credit Card Fraud Detection Technical Report

Final Documentation

Authors:

Kacper Stępniak – Product Manager & AI/ML Engineer

Michalina Sobkiewicz – Tech Lead & AI/ML Engineer

Contents

1. Executive Summary

This report summarizes a 10-week research and development project aimed at developing an effective machine learning model for detecting credit card fraud. The primary technical challenge of the project was the extreme class imbalance (only 0.172% of transactions were fraudulent).

Throughout the project, we analyzed the dataset, performed feature engineering, and tested a range of modeling approaches, including Random Forests, Neural Networks (MLP), and Gradient Boosting algorithms (XGBoost). Experiments covered both data augmentation techniques (SMOTE, Tomek Links) and algorithmic methods (Class Weighting, Focal Loss).

The final solution selected for deployment is the **XGBoost model utilizing the scale_pos_weight parameter**. This model demonstrated the best balance between Precision and Recall, reducing expected operational costs to **26,989 PLN** on the test set, which is a result over 27% better than the competing solution based on Focal Loss.

2. Introduction and Problem Definition

2.1. Business Background

Credit card fraud generates billions in losses for financial institutions annually. Detection systems must operate in real-time and minimize two types of errors:

1. **False Negatives (FN):** Missed fraud – generates direct financial loss.
2. **False Positives (FP):** False alarm – blocks a legitimate transaction, frustrates the customer, and generates administrative costs.

2.2 Project Objectives

The main goal was to create a model that maximizes the detection rate (Recall) while maintaining an acceptable level of precision, thereby minimizing the total business cost (Expected Cost).

3. Data Analysis and Preparation (EDA & Preprocessing)

3.1. Dataset Characteristics

The dataset contained 284,807 transactions made by European cardholders in September 2013. Features V1-V28 were the result of a PCA transformation (due to confidentiality). The only original features were Time (seconds elapsed since the first transaction) and Amount .

A key aspect was the class distribution :

- **Class 0 (Legitimate):** 284,315 (99.828%)
- **Class 1 (Fraud):** 492 (0.172%)

3.2. EDA Findings

- **Amount Analysis:** Fraudulent transactions had a different distribution of amounts, often containing outliers. A decision was made to use RobustScaler instead of standard scaling to minimize the impact of outliers.
- **Time Analysis:** Density plots revealed a cyclical pattern for legitimate transactions (drop during night hours), while fraud was evenly distributed throughout the day. This confirmed the utility of this feature for the model .

3.3 Data Splitting Strategy

To ensure robust model evaluation and hyperparameter tuning, the data was split into three distinct sets:

1. **Training Set (60%):** Used for model learning.
2. **Validation Set (20%):** Used for threshold tuning and monitoring overfitting during training.
3. **Test Set (20%):** Used exclusively for the final evaluation.

Stratification (stratify=y) was applied to maintain class distribution. Scaling (RobustScaler) was fitted exclusively on the **Training Set** to prevent data leakage.

4. Methodology Evolution and Prototyping

Three main modeling phases were tested during the project.

4.1 Phase I: Baseline and Data Augmentation

The first phase focused on balancing the dataset using synthetic methods.

- **Strategy:** A hybrid of SMOTE (oversampling) + Tomek Links (decision boundary cleaning).
- **Baseline Model (Random Forest):** Trained on balanced data, it achieved very good results (Recall 83%, Precision 79%) . It became the benchmark for subsequent experiments.

4.2 Phase II: MLP Neural Networks

A neural network prototype (Multilayer Perceptron) with a 128x64x32 architecture was built.

- **Challenges:** The model showed strong tendencies toward overfitting.
- **Optimization:** Dropout layers and an Early Stopping mechanism were applied.
- **Result:** Despite a higher Recall rate (87%), the model generated over three times as many false alarms (75 FP) as the Random Forest . A decision was made to abandon this path in favor of tree-based models.

4.3 Phase III: Advanced Gradient Models

The final phase focused on the **XGBoost** algorithm, abandoning data modification (SMOTE) in favor of internal mechanisms for handling imbalance. Two strategies were compared:

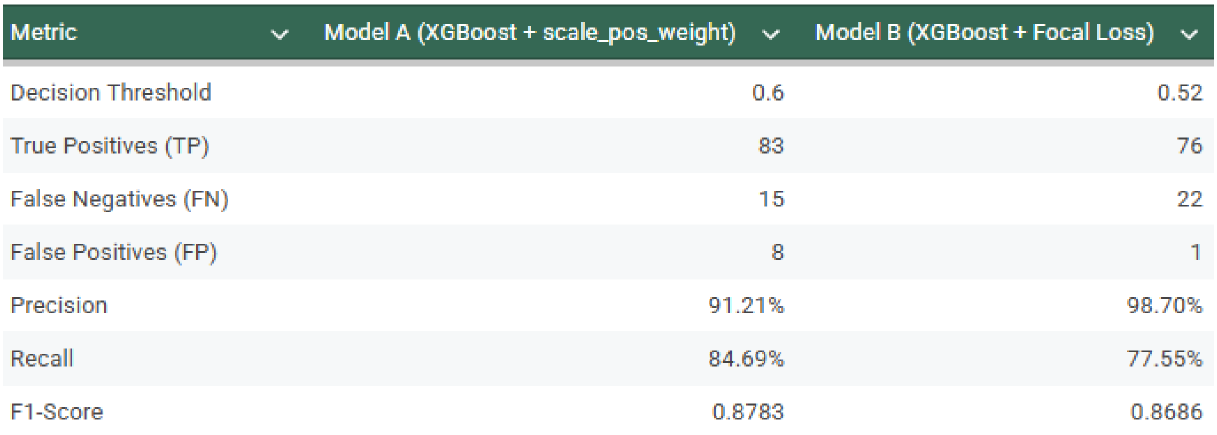
1. **Approach A (Class Weighting):** Setting the scale_pos_weight parameter to approx. 577 (ratio of negative to positive samples). This forces the model to treat an error on the minority class as 578 times more costly .
2. **Approach B (Focal Loss):** Implementation of a custom loss function that dynamically reduces the weight of easy examples, focusing learning on hard cases

5. Final Results and Head-to-Head Comparison

The final evaluation (Head-to-Head) was conducted on the untouched test set (56,962 transactions).

5.1 Quantitative Results

The table below presents key metrics for the two final candidates:



Metric	Model A (XGBoost + scale_pos_weight)	Model B (XGBoost + Focal Loss)
Decision Threshold	0.6	0.52
True Positives (TP)	83	76
False Negatives (FN)	15	22
False Positives (FP)	8	1
Precision	91.21%	98.70%
Recall	84.69%	77.55%
F1-Score	0.8783	0.8686

Quality Metrics for Classifiers:

Metric	Train	Valid	Test
TP	295	76	83
FP	0	7	8
TN	170,588	56,856	56,856
FN	0	23	15
Precision	1.0	0.92	0.91
Recall (TPR)	1.0	0.77	0.85
Specificity (TNR)	1.0	0.99	0.99
FPR	0.0	0.0001	0.0001
FNR	0.0	0.23	0.15
NPV	1.0	0.99	0.99
Error Rate	0	0.0005	0.0004

Metric	Train	Valid	Test
Accuracy	1.0	0.99	0.99
Balanced Accuracy	1.0	0.88	0.92
MCC	1.0	0.84	0.88
F1 Score	1.0	0.84	0.88
G-mean	1.0	0.88	0.92
AUC PR	1.0	0.83	0.87
AUC ROC	1.0	0.97	0.98
Brier Score	0.0	0.0005	0.0004
ECE	0.0001	0.0004	0.0004
MCE	0.32	0.71	0.43
Expected Cost	0 PLN	**38,797.35PLN**	**25,302.62 PLN**

5.2 Economic Analysis (Expected Cost)

Assuming business parameters (the cost of missing a fraud is significantly higher than the cost of verifying a false alarm), the expected cost for both models was calculated:

- **Model A Cost:** 25,302 PLN
- **Model B Cost:** 37,110 PLN

Although Model B is nearly flawless in avoiding false alarms (only 1 FP), its "cautiousness" results in missing 6 additional frauds compared to Model A. These misses generate a loss of over 10,000 PLN on a single test sample

6. Conclusions and Recommendations

6.1. Technical Conclusions

1. In the case of extremely imbalanced data, algorithmic methods (class weighting) in XGBoost proved more effective and stable than oversampling techniques (SMOTE) or custom loss functions (Focal Loss).
2. Neural networks (MLP) applied to tabular data with such characteristics would require significantly more complex engineering to match the performance of gradient boosting models.

6.2 Final Decision

We recommend deploying **Model A (XGBoost with scale_pos_weight)**. This model ensures optimal security (detecting approx. 85% of frauds) with minimal burden on the verification department (only 8 false alarms per 56,856 transactions). It is the most cost-effective solution and is ready for production implementation.