SGH x Mastercard Fraud Detection Hackaton
Entrance task report

Ludwik Madej, Maciej Andrzejewski, Piotr Kotłowski

1 Before we start

At the very beginning, we should clarify some assumptions we have made and define the key terms and principles that guide our approach throughout this task. These assumptions are critical for interpreting our methodology and evaluation strategy.

- Definition of fraud - in the context of this dataset, we define fraud as the unauthorized use of someone else's credit card. This refers to situations where the physical card or its associated data was stolen and used without the cardholder's consent.
- Temporal nature of data - we treat the data as a time series. This means that for any given transaction, we only use the historical information that was available up to that point in time. The details of how we enforce this constraint will be discussed further in the section on data splitting.
- Knowledge of fraud labels - we assume that, for previous transactions, we know whether they were fraudulent or not. This assumption is justified by the fact that, in real-world scenarios, fraud is typically reported within a short period after the transaction. Therefore, if no fraud report was made, the transaction can be reasonably assumed to be legitimate.
- No data augmentation - despite the significant class imbalance in the dataset (frauds constitute a small minority of all transactions), we deliberately chose not to apply any oversampling or undersampling techniques. This decision was made to preserve the natural distribution of classes and to avoid introducing artificial patterns that could misrepresent the original data, particularly in a regulatory or production context.
- Synthetic nature of the data - it is important to acknowledge that the dataset is synthetic and does not fully reflect real-world transaction behavior. For example, the distribution of transactions does not vary realistically over time (e.g., there is no observable difference between night and daytime activity), some fields contain implausible patterns (e.g., card creation dates occurring after the first transaction), majority of transactions takes place in the middle of the ocean and the average user travels more than 2000 kilometers between consecutive transactions. Because of this, instead of optimizing purely for performance on this dataset, we focused on designing a solution that mimics how a real-world bank might approach the problem. We believe this mindset results in a more valuable and generalizable solution. This focus will be particularly visible in the feature engineering stage, to which we devoted special attention.

2 Exploratory Data Analysis (EDA)

Due to the synthetic nature of the dataset, we did not observe any meaningful or realistic patterns during exploratory analysis. Most features follow either uniform or normal-like distributions, which limits the interpretability of the data and reduces the potential for discovering useful insights typically found in real-world financial datasets. In particular many features lack clear outliers or natural clusters that might indicate suspicious behavior.

These characteristics significantly reduce the effectiveness of traditional data exploration and make the modeling task more challenging, especially when aiming to extract informative features.

All plots, correlation matrices, and statistical summaries generated during the EDA process are available in the corresponding analysis file.

3 Feature Engineering, Data Split

Our feature engineering process was shaped by the overall modeling strategy, which combines both Point-in-Time (PIT) and Through-the-Cycle (TTC) perspectives. These approaches significantly influenced the way we designed and selected features.

- The PIT approach requires that features for each transaction are based strictly on information available up to that moment in time. In our implementation, this means we do not use the entire transaction history but instead focus on recent behavior - features are typically calculated over the last few transactions or short, recent time windows.
- The TTC perspective, on the other hand, aims to capture long-term behavioral patterns and risk profiles, independent of temporary fluctuations. These features may span longer time periods and help the model generalize across various types of customer behavior.

As a result, our features fall into two main categories:

- Dynamic, time-sensitive features, such as recent transaction frequency, time since last transaction, or changes in transaction amounts - all computed using only data from the past relative to each event.
- Stable, aggregated features, like long-term average transaction amount or historical fraud occurrence ratios, representing more general tendencies.

To stay consistent with our temporal modeling assumptions, we also carefully selected our data splitting strategy. After evaluating various options, we decided to divide the dataset based on time: the training set contains the earliest 80% of transactions (chronologically) and the test set comprises the most recent 20% of transactions.

This chronological split simulates a real-life deployment scenario where models are trained on historical data and applied to future events. It also ensures that features for both training and testing are constructed under realistic, production-like conditions.

We developed a wide range of features, including:

- **Temporal features**: day number, month number, day of the week, hour of transaction, whether the transaction occurred at night, and whether it was during the weekend. We applied *periodic encoding* (sine/cosine) to day, month, and hour to preserve cyclicity.
- **Behavioral reuse indicators**: whether the *device* or *payment method* had been seen before for the given user across their entire history and also separately for legitimate and fraudulent cases.
- **Transaction similarity**: cosine and euclidian similarities between the current transaction and past transactions, using numerical and categorical embedding vectors.
- **Country combinations tracking**: whether the user_country-merchant_country-transaction_country trios occurred before, in total and in last time and percentage of frauds in them.
- **Historical fraud statistics**: number of previous frauds, total number of transactions, whether the last transaction was fraudulent.
- **Amount profiling**: whether the current amount is closer to the mean amount from previous frauds or normal transactions; whether it lies within the confidence interval for legitimate transaction amounts mean.
- **Session duration analysis**: deviation of current session time from historical means, with comparisons between fraud and non-fraud patterns.
- **Geolocation and movement features**: physical distances between consecutive transactions, time between them, plausibility of movement, and z-scores for distance variation.
- **User statistics**: mean, median, and standard deviation of past transaction amounts, with corresponding z-scores to detect anomalies.
- **Transaction frequency metrics**: average and median time between user transactions, and their z-scores.

- **User-country-merchant triplet statistics**: historical fraud rates over full history and rolling windows of one, two, and three months.

Due to the nature of the task and our treatment of the dataset as a time series, all preprocessing steps were applied **before the train-test split**. At first glance, this may seem to risk data leakage. However we ensured that **no future information** was used when constructing features for any transaction.

Each transaction was processed in chronological order, and all features were calculated using **only the information available up to that point in time**. This means that even though the entire dataset was passed through the preprocessor beforehand, each data point was handled with full temporal isolation.

This approach allows us to simulate a real-time production setting where each transaction must be evaluated based solely on the historical context available at the moment of its occurrence.

4 Feature Selection

Given the high dimensionality of our feature space and the importance of selecting the most informative variables, we implemented a dual-method feature selection strategy using two distinct models: **Random Forest** and **Logistic Regression with Recursive Feature Elimination (RFE)**.

To ensure robustness and interpretability, we selected as our final feature set the **intersection** of features identified by both methods. This intersection represents variables considered relevant both in terms of non-linear relationships (captured by Random Forest) and linear separability (captured by Logistic Regression).

5 Model Selection

Due to the synthetic nature of the dataset, we expected that even sophisticated models might struggle to produce strong and generalizable performance. Patterns in the data are not always realistic, and several variables show artificial distributions that do not correspond to real-world behaviors (as discussed earlier in the report).

Nevertheless, we evaluated a wide range of classification algorithms to compare their relative effectiveness and determine which ones would serve as a strong foundation for further tuning. The following models were tested using default settings: `RandomForestClassifier`, `DecisionTreeClassifier`, `XGBClassifier`, `LGBMClassifier`, `LogisticRegression`, `GaussianNB`, `AdaBoostClassifier` with `DecisionTreeClassifier` as base estimator

After initial comparisons, we selected several promising models for more thorough **hyperparameter tuning**. We applied grid search procedures with the F2-score as the primary optimization objective. The tuned models comprise of `DecisionTreeClassifier`, `XGBClassifier`, `LGBMClassifier`, `RandomForestClassifier`, `AdaBoostClassifier` and `LogisticRegression`.

6 Final Model

After comparing multiple models and considering their performance, interpretability, and suitability for fraud detection, we decided to use a **Voting Classifier** as our final ensemble model. Specifically, we employed a `soft voting` strategy, where the predicted class probabilities from each base model are averaged, and the class with the highest mean probability is chosen as the final prediction.

7 Intepretability

To gain insight into how our model makes decisions and which features influence those decisions most, we employed the SHAP (SHapley Additive exPlanations) library [1]. While we will not go into the technical details of how SHAP works, we chose it deliberately due to its strong theoretical foundation [2] and wide acceptance in both academic and applied machine learning contexts.

Using SHAP allowed us to ensure that the model relies on relevant and meaningful features, not artifacts or noise. It also supported our goal of creating a solution that would not only be performant but also interpretable and responsible - particularly important when detecting fraud in sensitive financial systems.

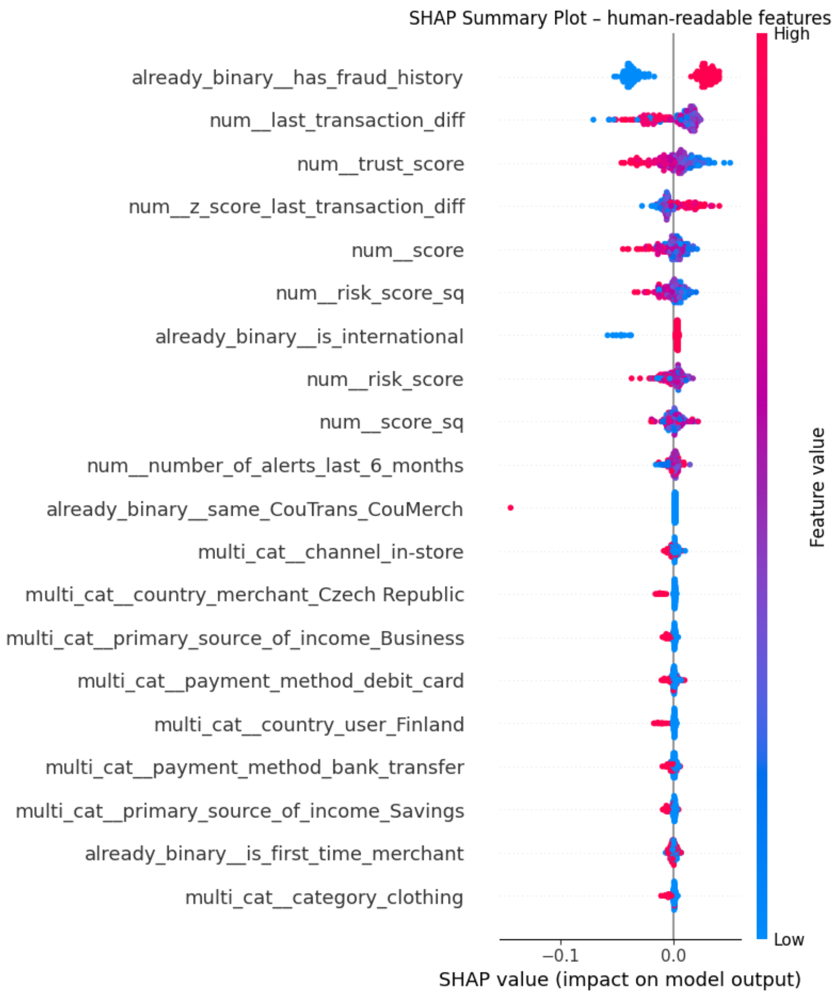

Figure 1: SHAP Summary Plot for selected features

The SHAP summary plot above shows which features most strongly influence the model's predictions. For instance, `has_fraud_history`, `last_transaction_diff`, and `trust_score` are among the most influential indicators. Features such as international transactions and historical similarity of country combinations (`same_CouTrans_CouMerch`) also appear to significantly contribute to the final decision. This aligns with our domain expectations and validates the relevance of our engineered features.

8 Model Performance

As expected due to the synthetic nature of the dataset, the model's performance on the test set is not impressive. The results are summarized in the confusion matrix (Figure 2) and the classification report below.

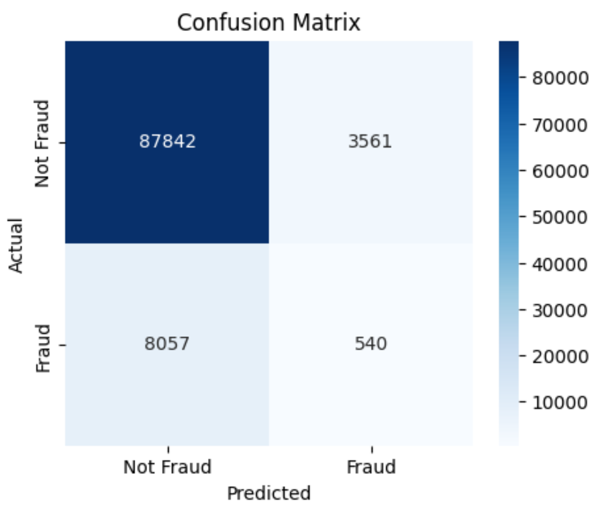

Figure 2: Confusion Matrix of the Final Model

Classification Report:

```
              precision    recall  f1-score   support

           0       0.92      0.96      0.94     91403
           1       0.13      0.06      0.09      8597

    accuracy                           0.88    100000
   macro avg       0.52      0.51      0.51    100000
weighted avg       0.85      0.88      0.86    100000
```

As the metrics show, the model struggles with fraud detection. This is most evident in the low recall (0.06) and precision (0.13) for the fraudulent class. These results confirm our assumption that, given the artificial character of the dataset, the model cannot generalize well.

We strongly believe, however, that on real-world banking data-especially when enriched with high-quality features similar to those we engineered-the model would perform significantly better. Our focus throughout this project was on building a realistic and scalable pipeline inspired by production-level systems, rather than optimizing for a synthetic benchmark.

9 Summary

In this project, we focused on designing a fraud detection that aligns closely with real-world banking systems rather than optimizing purely for performance on a synthetic dataset. Our methodology incorporated both point-in-time (PIT) and through-the-cycle (TTC) modeling assumptions, a careful time-aware data split, and an extensive feature engineering process reflecting user behavior, transaction patterns, and historical context.

Despite the limitations imposed by the artificial nature of the data-which likely impacted model accuracy-we prioritized model realism and interpretability. We believe that, when applied to real transaction data, our approach would prove substantially more effective and valuable in detecting fraudulent activity.

References

[1] *SHAP Documentation.* https://shap.readthedocs.io/en/latest/. Accessed: 2025-05-15.

[2] *Shapley value - Wikipedia.* https://en.wikipedia.org/wiki/Shapley_value. Accessed: 2025-05-15.