Project Title: Credit Card Fraud Detection

Objective: Develop a machine learning model to identify and prevent fraudulent credit card transactions.

Steps:

1. \*\*Data Collection\*\*:

- Gather a comprehensive dataset of credit card transactions. This dataset should ideally contain both legitimate and fraudulent transactions.

- Ensure data privacy and compliance with regulations like GDPR when collecting and handling sensitive financial data.

2. \*\*Data Preprocessing\*\*:

- Clean the data by addressing missing values, duplicates, and outliers.

- Explore the dataset to understand its structure and characteristics.

- Encode categorical features and standardize numerical features for modeling.

3. \*\*Data Splitting\*\*:

- Split the dataset into training, validation, and test sets. Typically, an 80-10-10 or 70-15-15 split is used.

4. \*\*Feature Engineering\*\*:

- Create relevant features that capture transaction behavior, such as transaction frequency, transaction amounts, time of day, and more.

- Consider feature scaling and transformation techniques to improve model performance.

5. \*\*Model Selection\*\*:

- Choose appropriate machine learning algorithms for fraud detection. Common choices include logistic regression, decision trees, random forests, support vector machines, and neural networks.

- Experiment with multiple models to find the one that performs best on the validation set.

6. \*\*Model Training\*\*:

- Train the selected models on the training data.

- Tune hyperparameters to optimize model performance using techniques like grid search or random search.

7. \*\*Model Evaluation\*\*:

- Assess the model’s performance using various metrics, including accuracy, precision, recall, F1-score, and ROC AUC.

- Evaluate the model’s ability to detect fraud while minimizing false positives.

8. \*\*Real-time Monitoring\*\*:

- Implement real-time monitoring of credit card transactions using the trained model. This involves setting up a system that can process and analyze incoming transactions in real-time.

9. \*\*Alerting System\*\*:

- Develop an alerting system that triggers notifications (e.g., emails, SMS alerts) when potentially fraudulent transactions are detected.

10. \*\*Continuous Learning\*\*:

- Implement a feedback loop to continuously update and improve the model. New data and fraud patterns should be incorporated into the model to stay effective against evolving fraud tactics.

11. \*\*Documentation\*\*:

- Maintain thorough documentation of data sources, preprocessing steps, model architecture, and performance metrics.

12. \*\*Privacy and Security\*\*:

- Ensure robust data security measures to protect sensitive customer information throughout the project.

13. \*\*Compliance\*\*:

- Ensure that your project complies with legal and regulatory requirements related to data privacy, such as GDPR or local financial regulations.

14. \*\*Scalability\*\*:

- Design the system to handle a growing volume of transactions as the credit card user base expands.

15. \*\*Deployment\*\*:

- Deploy the model and monitoring system in a production environment, ensuring high availability and scalability.

16. \*\*User Interface\*\* :

- Create a user interface for administrators to visualize and manage detected fraud cases.

17. \*\*Reporting\*\*:

- Generate regular reports summarizing the system’s performance and highlighting any trends or patterns in detected fraud.

Keep the project concise, focusing on the core steps of data collection, preprocessing, modeling, and real-time monitoring to achieve effective credit card fraud detection.

# Flow chart

Credit card fraud detection system

A credit card Fraud Detection System (FDS) is typically composed of a set of five layers of control

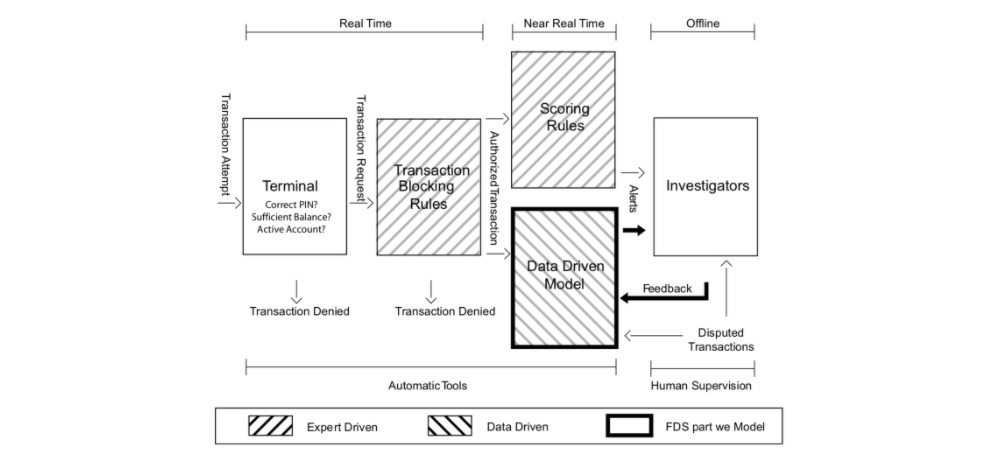


Fig. 1. Diagram illustrating the layers of control in an FDS. Our focus in this book is mostly on the data-driven model, which helps investigators by raising alerts on the most suspicious transactions.

The first two layers (Terminal and Transaction Blocking Rules) are executed in real-time (i.e. within milliseconds and before authorization). The next two layers (Scoring Rules and Data-Driven Model (DDM)), are executed in near real-time to potentially block the card and prevent additional frauds. Finally, the last layer (Investigators) is the only one requiring human intervention and is carried out offline. We describe each of these layers in more detail below.

Terminal

The terminal represents the first control layer in an FDS and performs conventional security checks on all the payment requests [MekterovicBrkicBaranovic18, VVBC+15]. Security checks include controlling the PIN code (possible only in case of cards provided with chip), the number of attempts, the card status (either active or blocked), the balance available, and the expenditure limit. These operations have to be performed in real-time (response has to be provided in a few milliseconds), during which the terminal queries a server of the card-issuing company. Requests that do not pass any of these controls are denied, while the others become transaction requests that are processed by the second layer of control.

Transaction-Blocking Rules

Transaction-blocking rules are if-then (-else) statements meant to block transaction requests that are perceived as frauds [DP15, DPBC+17]. These rules use the information available when the payment Is requested, without analyzing historical records or cardholder profiles. An example of a blocking rule could be: “IF internet transactions AND unsecured website THEN deny the transaction”. In practice, several transaction-blocking rules are simultaneously executed, and transactions firing any of these rules are blocked (though cards are not deactivated). Transaction-blocking rules are manually designed by the investigator and, as such, are expert-driven components of the FDS. To guarantee real-time operations and avoid blocking many genuine transactions, blocking rules should be: i) quick to compute and ii) very precise, namely should raise very few false alarms.

All transactions passing blocking rules are finally authorized. However, the fraud detection activity continues after having enriched transaction data with aggregated features used to contextualize the current purchase with respect to the previous ones and the cardholder profile. These aggregated features include, for instance, the average expenditure, the average number of transactions in the same day, or the location of the previous purchases [DP15, VVBC+15, WHJ+09]. The process of computing aggregated features is referred to as feature engineering or feature augmentation. Augmented features and current transaction data are stacked in a feature vector that is supposed to be informative for determining whether the authorized transaction is fraudulent or genuine. The Scoring Rules and Data-Driven Model (DDM) layers of the FDS operate on this feature vector.

Scoring Rules

Scoring rules are also expert-driven models that are expressed as if-then (-else) statements [CDPLB+18, DPBC+17]. However, these operate on feature vectors and assign a score to each authorized transaction: the larger the score, the more likely the transaction to be a fraud. Scoring rules are manually designed by investigators, which arbitrarily define their associated scores. An example of a scoring rule can be “IF previous transaction in a different continent AND less than one hour from the previous transaction THEN fraud score = 0.95”. Unfortunately, scoring rules can detect only fraudulent strategies that have already been discovered by investigators, and that exhibit patterns involving few components of the feature vectors. Moreover, scoring rules are rather subjective, since different experts design different rules. Finally, they can be incomplete and are difficult to maintain over time.

Data-Driven Model (DDM)

This layer is purely data-driven and adopts a classifier or another statistical model to estimate the probability for each feature vector to be a fraud. This probability is used as the fraud score associated with the authorized transactions. Thus, the data-driven model is trained from a set of labeled transactions and cannot be interpreted or manually modified by investigators. An effective data-driven model is expected to detect fraudulent patterns by simultaneously analyzing multiple components of the feature vector, possibly through nonlinear expressions. Therefore, the DDM is expected to find frauds according to rules that go beyond investigator experience, and that do not necessarily correspond to interpretable rules [Car18, DP15].

Investigators

Investigators are professionals experienced in analyzing credit card transactions and are responsible for the expert-driven layers of the FDS. In particular, investigators design transaction-blocking and scoring rules.

Investigators are also in charge of controlling alerts raised by the scoring rules and the DDM, to determine whether these correspond to frauds or false alarms [DP15]. In particular, they visualize all the suspicious transactions in a case management tool, where all the information about the transaction is reported, including the assigned scores/probabilities, which in practice indicate how risky each transaction is. Investigators call cardholders and, after having verified, assign the label “genuine” or “fraudulent” to the alerted transaction, and return this information to the FDS. In the following, we refer to these labeled transactions as feedbacks and use the term alert-feedback interaction to describe this mechanism yielding supervised information in a real-world FDS.

Any card that is suspected of fraud is immediately blocked to prevent further fraudulent activities. Typically, investigators check all the recent transactions from a compromised card, which means that each detected fraud can potentially generate more than one feedback, not necessarily corresponding to alerts or frauds. In a real-world FDS, investigators can only check few alerts per day [Car18, DP15, Kri10] as this process can be long and tedious. Therefore, the primary goal of a DDM is to return precise alerts, as investigators might ignore further alerts when too many false alarms are reported.

In recent systems, transactions associated with very high-risk scores can bypass investigators and be directly sent to the cardholder for feedback requests (e.g. by SMS). This addition is interesting for an accelerated processing and a larger feedback bandwidth. However, it should be done with caution to avoid jeopardizing the customer’s trust

Real-world data

The previous section illustrated how simple data preprocessing and supervised learning techniques can be used to design a baseline fraud detection system. The presented results relied on reproducible, but simulated data. Let us now apply the exact same methodology with real-world transaction data. Due to confidentiality reasons, the data used in this section cannot be shared. While the results presented in this section cannot be reproduced, they however provide insights into the performances that would be obtained in a real-world setting.

The dataset used was provided By Worldline, and is similar in nature to the datasets used In the publications referenced on our ResearchGate page – Joint collaboration: MLG ULB and Worldline.

Training and test sets

More specifically, our real-world data are e-commerce transactions for Belgium from 2018. The number of daily transactions is around 400000. The number of daily fraudulent transactions was around 1000, giving a ratio of fraudulent transactions of around 0.25%. The number of daily compromised cards was slightly above 300: each compromised card caused an average of 3 fraudulent transactions per day. Fig. 1 summarises these statistics on a daily basis for the period 2018-07-25 to 2018-08-14.

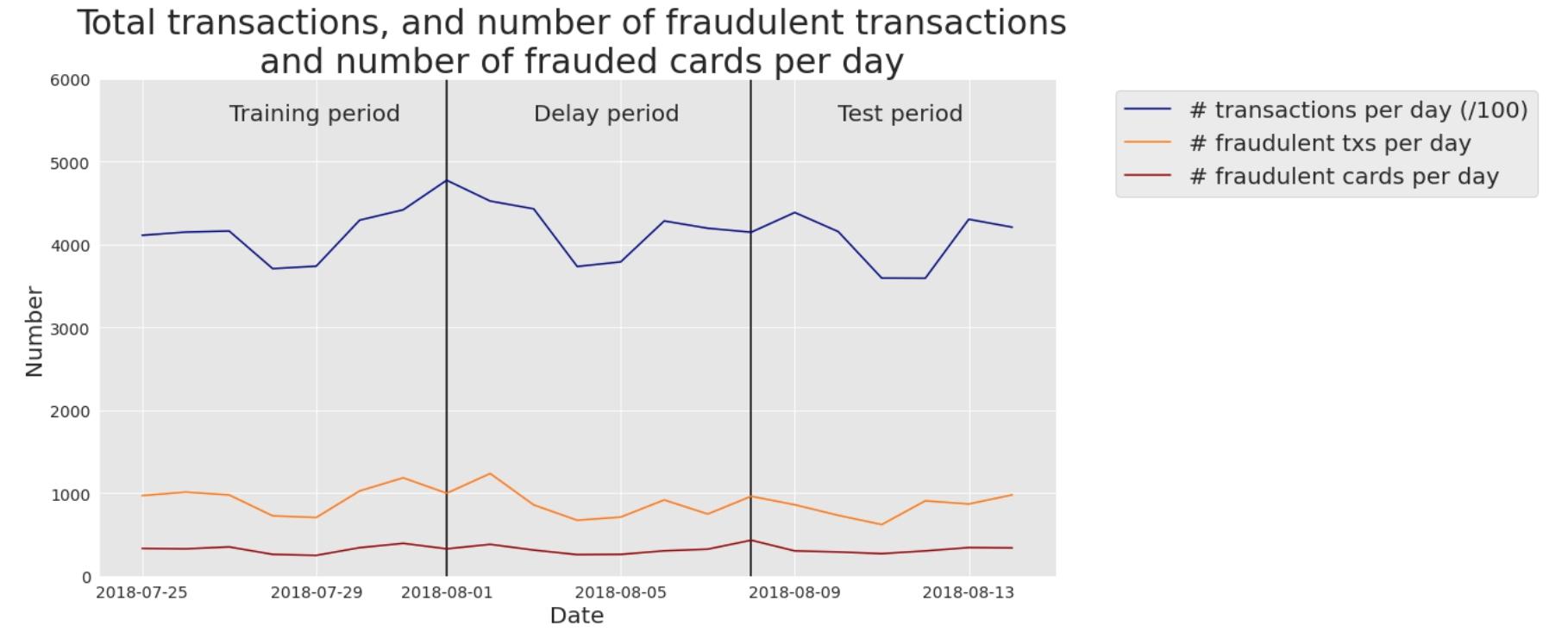


Fig. 1. Real-world transaction data. Number of transactions per day (/100), fraudulent transactions per day, and compromised cards per day. Period: 2018-07-25 to 2018-08-14.

Model training: Decision tree

We first trained a decision tree of depth two. The resulting tree is given in Fig. 2. It is worth noting that the fraud detection rules uncovered by the decision tree differ from the ones obtained on simulated data. The main fraud pattern uncovered by the tree (blue leaf at the bottom right) consists of transactions that occurred on a recently compromised terminal (TERMINAL\_ID\_RISK\_30DAY\_WINDOW>0.075), and for which the customer had realized a large number of transactions in the previous 24 hours (CUSTOMER\_ID\_NB\_TX\_1DAY\_WINDOW>13.5). This rule actually relates to a common fraud behavior: Fraudsters usually try to carry out as many transactions as they can when they manage to compromise a credit card, thus increasing the number of transactions carried out in the last day.

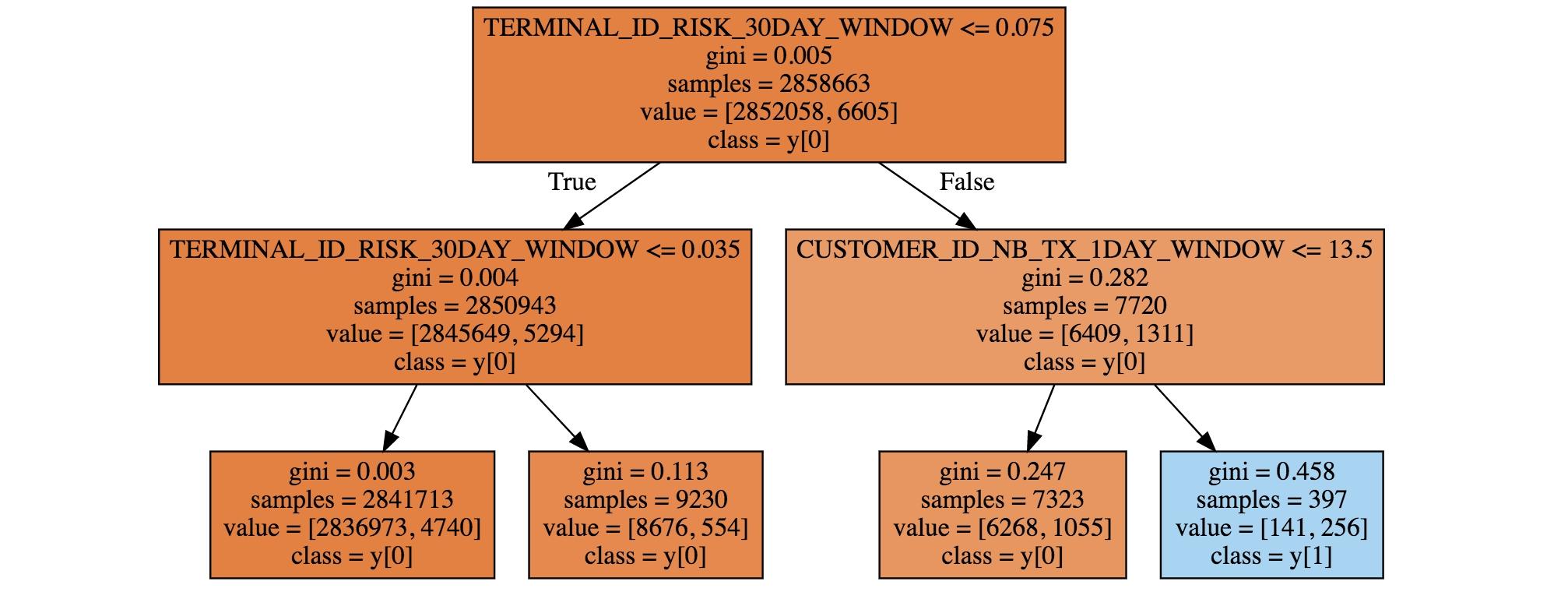


Fig. 2. Decision tree of depth two, obtained using one week of real-world data.

. Performances using standard prediction models

We then ran the same five training algorithms as in the previous section: Logistic regression, decision trees of depth two and unlimited depth, random forest, and XGBoost. The performance results on the test set are reported in Fig. 3.

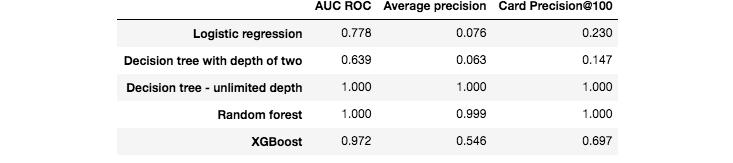


Fig. 3. Fraud detection performances obtained with five standard machine learning algorithms on the test set.

The AUC ROC of all the models is higher than 0.5, meaning that they all perform better than a random classifier. For the three performance metrics, the worst model is the decision tree with unlimited depth (AUC ROC close to 0.5, meaning that it performs not much better than a random classifier). The best models are XGBoost and logistic regression, depending on which performance metric is chosen. For AUC ROC and AP, XGBoost provides the best performances. For CP@100, the best performance is provided by the logistic regression model.

It must be no’ed that these results are preliminary, and only reflect performances obtained on a small subset of the data, without any tuning of the model parameters. Still, it is remarkable that the detection rates of these baseline models, in particular for XGBoost and logistic regression, are well above those of a random classifier. Recalling that the proportion of fraudulent transactions is only 0.25%, the card precision@100 of a random classifier should be around 0.0025 (that is, less than one compromised card detected per day). A simple logistic regression model boosts the CP@100 to 0.15, meaning that on average, this classifier allows correctly detecting, every day, 15 compromised cards out of the 100 most suspicious cards identified by the prediction model.

Besides the prediction performances on the test set, we also computed the prediction performances on the training set. They are reported in Fig. 4. It is worth noting that two models provide perfect predictions: The decision tree with unlimited depth, and the random forest (AUC ROC of 1). These results were also observed for the synthetic data (see previous section, Performances using standard prediction models), and reflect the overfitting phenomenon. The models the least sensitive to overfitting are the logistic regression model, and the decision tree with depth 2. It makes sense that the training metrics of tree-based models with unlimited depth (Random Forest, Decision Tree) achieve perfect detection because, during training, these models will split data recursively until obtention of pure leaves.



Fig. 4. Fraud detection performances obtained with five standard machine learning algorithms on the training set.

We finally report the execution times, for training and predictions, of these five predictions models. The results are presented in Fig. 5. It is worth noting that the execution times are much higher than with the simulated dataset. In particular, the random forest and XGBoost models were run using a server with 20 cores. Their training execution times are therefore close to 100 times longer than logistic regression. XGBoost also has an implementation optimized for GPU that could still speed up training by one order of magnitude.

In conclusion,

credit card fraud detection using data science is a crucial and effective approach to protect financial transactions and prevent fraudulent activities. By leveraging machine learning algorithms and data analysis techniques, it's possible to detect and mitigate fraudulent transactions in real-time, providing a safer environment for both consumers and financial institutions. Continuous model improvement, data updates, and collaboration between data scientists and security experts are essential for staying ahead of evolving fraud tactics and ensuring the security of credit card transactions.