RISK MANAGEMENT MAGAZINE

Vol. 20, Issue 2
May – August 2025

EXCERPT

https://www.aifirm.it/rivista/progetto-editoriale/

Integrating Machine Learning and Rule-Based Systems for Fraud Detection: A case study based on the Logic Learning Machine

Pier Giuseppe Giribone, Giorgio Mantero, Marco Muselli and Damiano Verda

Integrating Machine Learning and Rule-Based Systems for Fraud Detection: A case study based on the Logic Learning Machine

Pier Giuseppe Giribone (University of Genoa, BPER Group); Giorgio Mantero (Rulex Inc.); Marco Muselli (Rulex Inc.), Damiano Verda (Rulex Inc.)

Corresponding Author: Giorgio Mantero (giorgio.mantero@rulex.ai)

Article submitted to double-blind peer review, received on 14st June 2025 and accepted on 13th August 2025

Abstract

Money laundering is one of the most relevant global challenges, with significant repercussions on the economy and international security. Identifying suspicious transactions is a key element in the fight against the phenomenon, but the task is extremely complex due to the constant evolution of the strategies adopted by criminals and the great amount of data to be analyzed daily. This study proposes a hybrid method that integrates Machine Learning models with heuristic rules, with the aim of identifying fraudulent transactions more effectively. The dataset used, SAML, includes millions of bank transactions and presents a strong imbalance between classes (fraudulent vs regular transactions). The entire process was carried out through a self-code platform designed to optimize data management, processing and analysis. The heuristic rules were evaluated using the covering and error metrics and then integrated into the Logic Learning Machine (LLM) task. The effectiveness of the approach was verified by comparing two main configurations: one based exclusively on the use of LLM and the other combining LLM and heuristic rules. The results obtained highlight that the integration of heuristic rules improves the performance of the model, confirming the synergy between Machine Learning and expert knowledge. This study confirms the effectiveness of the hybrid approach and emphasizes the importance of the union between automated analysis and human insight to address the challenges posed by money laundering.

Key Words: Anti-Money Laundering (AML), Transaction Monitoring, Synthetic Dataset, Machine Learning (ML), Heuristic Rules, Logic Learning Machine (LLM)

JEL code: C38, C45, K14, K22

1) Introduction

As described by the United Nations Office on Drugs and Crime (UNODC): “Money laundering is the processing of criminal proceeds to disguise their illegal origin. This process is of critical importance, as it enables the criminal to enjoy these profits without jeopardizing their source” (UNODC, 2021). Money laundering therefore indicates all those processes implemented by criminal organizations in order to disguise the origins of money obtained through illegal activities, such as corruption, drug trafficking, or fraud, to make it appear legitimate. By implementing it, such organizations can integrate illicit funds into the financial system, allowing further investment in illegal operations while still managing to avoid recognition by supervisory bodies.

These illicit activities pose serious concerns for the global economy because they are used to fund criminal activities, they disrupt financial markets and ultimately may also damage the reputation of the financial institutions involved in fraudulent activities.

Although it is clearly not possible to directly measure the extent of money laundering as we usually do with legitimate economic activities, its scale is massive, thus representing a significant threat to global financial systems. The UNODC estimates that money laundering accounts for 2-5% of global GDP annually, i.e. between 800 billion and 2 trillion EUR (UNODC, 2021), thus underscoring the need for robust detection and prevention mechanisms.

Money laundering is a global phenomenon, but data shows that it is more prevalent in specific industries and countries. In particular, the main sectors most affected by it are the real estate, the financial system, the gambling system, the trade of luxury goods, international trades, the construction sector and FinTech. Regarding the most problematic countries, the Financial Action Task Force (on Money Laundering), better known as FATF, identifies the countries with severe weaknesses in measures to combat money laundering and terrorist financing through the "High-risk jurisdictions subject to a call for action" list (blacklist) and the "Jurisdictions under increased monitoring" list (greylist).

Anti-Money Laundering (AML) thus refers to the regulations, policies, and procedures designed to detect and prevent money laundering activities. The main objective of AML agencies is indeed to prevent criminals from using the financial system to conceal the funds arising from their illicit activities. The work of AML professionals is divided into three main areas:

- **Prevention:** This part involves ensuring that governments, companies and financial institutions take all the necessary preventive measures to identify suspicious activities in their work;

- **Monitoring:** This consists of creating monitoring systems to inspect transactions and flag those that may be linked to laundering activities;

- **Reporting and prosecution:** This phase involves reporting any suspicious activity to Financial Intelligence Units (FIUs) in order to take action. In such sense, cooperation between bodies to enable criminal investigation is crucial.

All these procedures are carried out in full compliance with national and international laws (Financial Action Task Force, 2003). Despite advancements, AML efforts face several challenges:

- **Scalability:** The sheer volume of daily financial transactions demands highly efficient AML systems. For instance, international banks process millions of real-time transactions each day, requiring models that balance accuracy and speed for timely fraud detection without compromising system performance;

- **False positives:** AML detection systems often use strict rules and conservative thresholds to avoid missing fraud, but this leads to high false positive rates, driving up costs and causing delays that can harm customer relationships. The key is finding a balance between swiftness and accuracy, with systems that are fast yet precise enough to catch suspicious cases. The choice between highly accurate but slower systems and faster, less precise ones largely depends on the volume of alerts they need to handle;

- **Adaptability:** As criminal techniques evolve, and new fields emerge, such as cryptocurrencies and decentralized finance, AML systems must continuously adapt to keep pace. Relying only on past fraud patterns risks bringing rapid obsolescence, making it harder to detect new illicit behaviors. This requires constant updates to both heuristic rules and data-driven models, which can be costly, both in terms of maintaining high performance models and training operators with deep expertise.

The complex and multifaceted fight against money laundering is a problem that requires robust regulatory frameworks to ensure the resilience of the financial system. To counter the threats posed by it, governments, international organizations and financial institutions developed a wide range of guidelines and regulations over time with the aim of preventing, detecting, and prosecuting illicit activities. In this sense, part of our heuristic rules was drafted in order to align with these international guidelines and incorporate a regulatory perspective.

This study aims to investigate whether the combination of Machine Learning systems with heuristic rules can improve the effectiveness in detecting fraudulent transactions in the AML field. The proposed approach aims to exploit the strengths of data-driven methodologies while integrating specific sectorial expertise. The analysis was conducted using the Rulex Platform, an advanced platform that combines data analytics tools, Machine Learning tasks and heuristic rules management, allowing to efficiently implement and test models. The purpose of this study is to assess the effectiveness of this solution based on the Logic Learning Machine (LLM) algorithm by comparing it with other traditional Machine Learning approaches in order to analyze its performance in detecting odd activities in financial transactions datasets. We have chosen the LLM algorithm because it has proven to be valuable in solving problems in the context of financial and credit risk management. In particular, it has been employed both in an asset allocation context in order to select the optimal weights of a portfolio of ESG assets (Gaggero et al., 2024) and to improve models for predicting the probability of default in a set of U.S. companies (Berretta et al., 2025). In both contexts, it proved to be a reliable supervised Machine Learning technique characterized by a high level of explainability.

2) Literary review

The techniques used in the AML area mainly focus on the implementation of analytical and technological methodologies to detect and prevent money laundering activities. Specifically, there are customer due diligence measures (KYC and customer risk scoring), computing techniques to discover fraudulent patterns (ML algorithms) and finally manual investigation to confirm the flags raised. Traditionally, fraud detection has relied on deterministic rules, often derived from regulations, and subsequent manual checks by operators. These methods, albeit very useful, show severe limitations in terms of scalability and ability to adapt to complex and evolving fraudulent schemes, as well as high costs in terms of training suitable personnel.

In recent years, Machine Learning has emerged and begun to revolutionize fraud identification, allowing specialists to analyze large volumes of data and identify complex patterns that are not easily detectable with static rules (Teradata, 2022) (Nweze et al., 2024).

Machine Learning techniques are broadly categorized into supervised, unsupervised, and semi-supervised approaches, and are applied across several analytical dimensions, including anomaly detection, risk scoring, behavioral modelling and link analysis. Specifically, supervised models rely on labeled datasets distinguishing between normal and suspicious transactions. Notable algorithms include: Support Vector Machines (SVM), which are effective in high-dimensional spaces though computationally intensive on large, imbalanced datasets; decision trees, which are highly interpretable and useful for risk scoring and profiling but prone to overfitting if not appropriately pruned; and Radial Basis Function Networks (RBFN) which offer great adaptability and fast learning, but suffer from the risk of overfitting in cases of low feature diversity. On the contrary, unsupervised techniques cluster data without any prior label, making them especially suitable when suspicious labels are scarce. The most prominent algorithms are clustering techniques like K-means and CLOPE, which are used to group similar patterns of transactions to identify anomalies, and Expectation-Maximization (EM) methodologies to model customer behavior and detect deviations. Semi-supervised approaches are designed in such a way to strike a balance between the need for labeled data and the complexity of fraudulent patterns. The latter combine supervised learning and clustering, often using synthetic data to overcome the problem of class imbalance. Deep learning refers to a subclass of Machine Learning techniques that utilizes models composed of many layers of nonlinear transformations. These networks are capable of learning complex and abstract representations of data, often with performance far superior to other techniques, but at the cost of needing extensive computational resources and lacking interpretability. Another highly developed branch is the one related to graph-based methodologies and Social Network Analysis (SNA). The latter are increasingly used to model relationships among different entities, showing structural patterns for money laundering schemes. It works by building a graph where the nodes represent different entities (e.g. bank accounts) and the arcs represent the relationship between nodes (e.g. money transactions). The objective is to identify specific money laundering schemes like circular-shaped transactions or hub-and-spoke structures, characteristic of layering (Chen et al., 2018).

More recently, literature highlights a growing interest in hybrid methodologies that combine the rule-based approach with Machine Learning models. The latter are gaining more and more ground, as they leverage the strengths of both techniques. These approaches make it possible to improve the explainability of decisions, exploiting the expert knowledge embedded in heuristic rules, while maintaining the flexibility and learning ability of Machine Learning models.

Although literature has demonstrated that a rational integration of heuristic rules with Machine Learning models generally improves the fraud detection process, the creation, but above all the management, of large rulesets can lead to severe operational complexities and difficulties in their interpretability. For this reason, in recent years research focused not only on the combination of the two methodologies, but also on the optimization of such sets of rules, with the aim of reducing their number and their computational cost, obviously without compromising the overall model performance. This particular field of research led to the development of models capable of improving sets of rules using techniques that, once again, combine algorithms and human expertise.

One of the most interesting studies in this field is that related to the development of the RUDOLF system (Milo et al., 2018). In this study the authors try to overcome the classic problem linked to the use of "mining" and Machine Learning techniques for the derivation of rules in the anti-fraud field. Since heuristic rules must necessarily be updated or redefined from time to time to keep up with fraud trends, researchers developed RUDOLF, a system that assists experts in defining and redefining the rules for identifying fraudulent transactions. RUDOLF first tries to uncover illicit instances by generalizing the initial rules proposed, and then it specializes them in order to avoid capturing unhelpful legitimate transactions. The changes are not mandatory but just proposed by the system: the supervisor can consequently accept, modify or reject each suggestion, based on his judgement and experience. This process goes on until the expert obtains the desired ruleset. Modifications to rules made by RUDOLF are associated with a cost-benefit model. This model assumes that every operation performed leads to a cost, but also to an entailed benefit, measured in terms of an increase in the number of frauds captured by the new rule or a decrease in the number of normal transactions captured. Therefore, the system's objective is to modify the existing ruleset so that the cost function is minimized.

In the performance comparison between the baseline version of RUDOLF and RUDOLF^{-}, a variant that automatically refines the ruleset without consulting experts, the authors demonstrated that RUDOLF performs best in various domains, especially in the quality of predictions. This demonstrates the relevance of incorporating the expertise of AML domain specialists in the drafting of anti-fraud rules.

Another relevant contribution in this field is given by the ARMS project (Aparício et al., 2020). The Automated Rules Management System (ARMS) is a technique that optimizes the set of rules used thanks to heuristic search and a loss function defined by the user. Its proposed goal is to minimize the number of rules and alerts, while preserving the initial performance.

Instead of considering and evaluating each rule independently, researchers built a system to manage rules that considers the interactions between rules with different actions and priorities. The latter are needed because transactions may trigger different rules with contradictory actions, creating the need for a stable hierarchy between rules. The ARMS optimization process is basically implemented in two ways, specifically by disabling inefficient rules and by changing rules' priorities. The heuristic methods tested by authors are random search, greedy expansion and genetic programming. The results obtained on two big online datasets show that ARMS was able to remove almost 50% (and 80% in the second case) of initial rules, while maintaining the original performance of the system.

3) Regulatory Framework

The complex and multifaceted fight against money laundering is a problem that requires robust regulatory frameworks to ensure the resilience of the financial system. To counter the threats posed by it, governments, international organizations and financial institutions developed over time a wide range of guidelines and regulations with the aim preventing, detecting, and prosecuting illicit activities.

We now briefly explore the key international standards and regulatory frameworks that underpin AML efforts. These regulations provide a comprehensive framework for addressing financial crimes by establishing obligations for financial institutions and governments, promoting cross-border cooperation and ensuring rule-compliance through rigorous monitoring mechanisms.

3.1) FATF Recommendations (Financial Action Task Force)

Less than a year after its establishment by the G7 summit, the FATF issued in 1990 a report containing the well-known set of "Forty recommendations", later revised in 1996. Five years later, in 2001, the FATF expanded its mandate to also cope with the problem of terrorism financing (AML/CFT Anti-money laundering and Combating the Financing of Terrorism) and it continued to update its agenda over the years.

With its recommendations, the FATF is internationally endorsed as the global standard against money laundering and terrorist financing, as well as the financing of proliferation of weapons of mass destruction.

The recommendations, last comprehensively revised in 2012 and subsequently updated on specific issues such as virtual assets and beneficial ownership (e.g., 2021), set the minimum standards that countries must implement according to their specific circumstances and regulatory system (FATF, 2025). The FATF provides guidance on the following topics:

Risk-based approach: This means that each country should assess the risks that it faces and take appropriate preventive action in response (KYC-Chain, 2020). Such an approach may also be "scalable", in the sense that riskier instances clearly need more stringent measures and vice versa, so it is a proportionate approach.

Sanctions: The FATF recommends applying member states to implement a "targeted financial sanctions regime" to fully comply with the United Nations Security Council Resolutions (UNSCRs). The latter asks governments to freeze without delay the assets and funds of listed people or entities (or groups of them) that pose terrorist financing risk, and also ensure that no further financial assets are made available to them in the future (KYC-Chain, 2020) (FATF, 2013).

Customer Due Diligence and record-keeping: This principle states that financial institutions should not keep anonymous accounts or with obviously fictitious names. Additionally, it requires financial institutions to undertake customer due diligence measures, like identifying and verifying the identity of their clients (FATF, 2025). Furthermore, agencies are asked to keep records of all relevant information about clients in order to assess the risks posed by potential and current customers.

Reporting of suspicious transactions and compliance Reporting: it is a vital tool for AML and CFT measures to work in an efficient way. If authorities do not receive any reports, it is obvious that finding illicit activities becomes problematic. The FATF strongly recommends that institutions implement a mandatory reporting obligation, regardless of the gravity of the illicit action, also posing great importance on the celerity in sending such reports: the sooner the better (KYC-Chain, 2020) (FATF, 2025).

New technologies: The FATF strongly suggests countries to be aware of how fraudsters may use new arising and disruptive technologies in order to commit crimes, particularly regarding the financial sector. In this sense institutions should not release new products or technological developments unless a prior risk assessment has occurred. This means implementing a sophisticated system to prevent, or at least better manage, any potential risk that could emerge. Regarding virtual assets (e.g. cryptocurrencies), countries should ensure that the service providers are regulated for AML/CFT purposes, registered and adequately monitored (KYC-Chain, 2020).

3.2) European Union Anti-Money Laundering Directives (AMLD)

The evolution of anti-money laundering efforts at European level can be described by analyzing the series of Directives produced. The first commitment dates back to 1991, when the first Directive was adopted to prevent the misuse of the financial system for the purpose of money laundering (European Commission, 2023). This process was undoubtedly influenced by FATF recommendations produced on the same year and driven by the rising international awareness and effort in tackling money laundering. In particular the first AMLD focused on the imposition of due diligence measures for financial institutions and on the establishment of a reporting system for all suspicious transactions. In the following Directives, the European Union kept revising the previous regulatory limitations and expanding the operational framework in order to mitigate the risks related to money laundering. A particularly important step was made with the Third Directive, after the September 11, 2001 terrorist attacks in the US, which stressed the problem of organized terrorism even more and made clear that tighter measures were needed at global level. Relevant advancements were made in January 2020 with the transposition by all member states of the Fifth AML Directive which aimed at expanding the extent of regulation also to virtual currency exchanges, estate agents, art dealers and more (LSEG, 2024) (European Union, 2018). The Sixth Directive came into effect in December 2020 not long after the previous one, and again, some major improvements were made. Specifically, they reached an harmonized definition for "predicate crime" ("cyber crime" and "environmental crime" had been included within the offences); they further expanded the regulatory scope also considering "aiding and abetting" as punishable criminal offences; they extended the criminal liability, so that companies could also be criminally liable for the illicit actions carried out by their employees; and lastly they also made punishments tougher by increasing the sentences for money laundering crimes.
The last Directive (EU) 2024/1640, known as AMLD VI, was adopted in April 2024 and represents a major overhaul of the EU's anti-money laundering (AML) framework. It builds upon and significantly strengthens previous directives (AMLD IV & V) through more centralized oversight, broader scope, and enhanced transparency. AMLD VI is the most ambitious and comprehensive EU AML directive to date. It establishes a new centralized authority (AMLA), mandates greater transparency and access to financial data, introduces tighter controls on crypto and cash, and broadens the regulatory scope to new sectors like football and luxury goods (see paragraph 3.6). It marks a major step toward a fully harmonized and enforceable EU-wide AML framework.

3.3) United Nations Convention Against Corruption (UNCAC)

Adopted in 2003 and entered into force in 2005, the UNCAC (also known as "Merida Convention") had the aim to promote preventing measures against corruption and the criminalization of acts like money laundering, bribery and embezzlement (UNODC, 2000). It is the only legally binding universal tool to contrast corruption. It requires member states to implement a set of anti-corruption measures and promotes both international cooperation and mutual legal assistance in the fight against illicit activities. The convention also plays a relevant role in driving forward both the 2030 Agenda and the Sustainable Development Goals (SDGs), by addressing the widespread problem of corruption (United Nations, 2021).

3.4) Bank Secrecy Act (BSA)

Overseas, the first law to combat money laundering was enacted as early as 1970 by the US Congress with the Bank Secrecy Act, officially known as the Currency and Foreign Transaction Reporting Act (IRS, 2025). Shortly after its passage, the BSA met the indignation of several groups who thought it was unconstitutional, claiming it was violating both the Fourth and Fifth Amendments. For these reasons it remained inactive until the '80s, when financial institutions eventually complied with BSA requirements.
Nowadays BSA mandates financial institutions to maintain records and submit different types of reports based on the problem encountered. It is a cornerstone regulation in the combat against money laundering and other fraudulent activities in the US.

3.5) Recent developments in Italian regulations (UIF)

On July 3, 2025, the Italian Financial Intelligence Unit (UIF) published a consultation paper updating the instructions regarding the reporting of suspicious transactions (SOS), with the aim of enhancing the effectiveness of the anti-money laundering and counter-terrorism financing system. The new text is intended to replace the current regulation dated May 4, 2011, in light of recent regulatory developments and international best practices. UIF seeks to improve the quality of reports by discouraging overly automatic or overly cautious reporting practices, which risk undermining the investigative value of suspicious transaction reports.
The instructions are divided into three main sections. The first part sets out the general principles and operational rules, emphasizing the importance of active cooperation by obligated entities. It clarifies that a report should not be triggered by numerical thresholds or automated criteria alone, but rather by a concrete, documented, and reasoned assessment of objective or subjective anomalies. The analysis process must be thorough, and in some cases, may include the temporary suspension of the suspicious transaction. Additionally, UIF encourages feedback mechanisms to strengthen the overall effectiveness of the reporting system. The second part of the document addresses the organizational and procedural obligations of reporting entities. Each organization or professional must

appoint a person responsible for SOS reporting, who must be independent, competent, and free from conflicts of interest. In smaller settings, such as individual practices, this responsibility may fall onto the professional directly. Entities are required to establish formal procedures for detecting and assessing suspicious activity, even when using IT tools or artificial intelligence algorithms. However, these tools must complement human analysis rather than replace it.

The third part focuses on the technical and operational aspects of submitting reports through the Infostat-UIF portal. It provides guidelines on how to register, compile, submit, amend, or cancel reports, with the goal of streamlining the process and ensuring consistent information flows.

The document has been opened to public consultation for a period of 60 days, ending on September 3, 2025. UIF invites all relevant stakeholders, including court-appointed administrators, to submit comments and suggestions. Particular attention is given to those operating in high-risk contexts, such as the management of seized or confiscated assets.

Overall, UIF's proposed reform marks a decisive step toward a more selective, professional, and effective reporting system. The goal is not to increase the number of reports, but to enhance their quality, favoring thoughtful analysis over automatic or overly cautious reporting. This change requires the active commitment of all obligated entities, who are called upon to exercise sound and responsible judgment in managing money laundering and terrorism financing risks (UIF, 2025).

3.6) Toward centralized AML supervision in Europe

In 2024, the European Union introduced a major reform of its anti-money laundering framework with the approval of a new legislative package, including two Regulations and one Directive. Most notably, Regulation (EU) 2024/1620 established the European Anti-Money Laundering Authority (AMLA), headquartered in Frankfurt. AMLA will directly supervise selected high-risk financial institutions and coordinate national AML/CFT authorities, with the goal of harmonizing and strengthening supervisory practices across the EU (European Union, 2024 [a]). Alongside this institutional innovation, Regulation (EU) 2024/1624 and Directive (EU) 2024/1640 lay out new rules on risk assessment, customer due diligence, and cross-border cooperation. This transition to centralized supervision marks a clear shift toward greater efficiency, consistency, and technological adoption in the fight against financial crime.

The approach proposed in this study, combining Machine Learning techniques with expert-driven heuristic rules, aligns with the European regulatory direction, which increasingly promotes data-driven supervision and enhanced detection capabilities supported by innovation and harmonized methodologies (European Union, 2024 [b]) (European Union, 2024 [c]). Furthermore, the study is particularly timely considering the recent AML regulatory developments proposed by UIF whose documents are discussed by professionals in this field.

4) Description of the methodologies

In this section, we analyze in detail the three predictive models used in the study, namely: Decision tree, Logistic and Logic Learning Machine. Each model takes a different approach to classification, with distinct characteristics that influence both performance and interpretability of the results. The underlying idea is to check how the Logic Learning Machine performs and then compare its results with the ones obtained through the other two standard classification techniques. Therefore, we first describe the working principles of the models considered, analyzing their key features, in such a way as to provide a clear understanding of each algorithm. Beyond describing the models, we also analyze the evaluation metrics used to assess their effectiveness and determine the best configuration.

4.1) Traditional Machine Learning methodologies

Traditional Machine Learning methodologies, such as Classification and Regression Trees (CART) and logistic regression, have long been foundational in data analysis, providing powerful tools for extracting patterns and making predictions.

CART models represent a method for approximating data through a stepwise function, obtained by iteratively subdividing the space of observations, based on specific thresholds applied to the model variables. Each subdivision aims to identify subsets of data with values of the target variable that are as homogeneous as possible. In classification problems, observations are assigned to the most representative class within the group to which they belong. This methodology is often visualized as a decision tree, since the separation rules can be organized in a hierarchical structure. Each subdivision of the dataset can be represented as a node in a binary tree, where the first node, known as the “root”, serves as the starting point, while the terminal nodes, called “leaves”, identify the final subsets of data. To train a decision tree, a dataset is needed where the target variable is known. The algorithm builds the model progressively, applying binary splits to the data, based on optimal cutoffs. In each phase, a threshold is determined for each variable which allows to obtain a subdivision such as to reduce the variance within each subset and, at the same time, increase its difference compared to the other groups. This procedure is repeated iteratively on each new subset, determining, each time, the optimal threshold for the variable considered. The process continues until a stopping criterion is reached, which may be, for example, the impossibility of further improving the separation of the data, the presence of only one element in a subset or the creation of a group composed exclusively of observations belonging to the same class as the target variable.

Once training is complete, the resulting tree can become very complex and detailed, with a large number of splits that make it difficult to interpret and increase the risk of over-fitting. To avoid this problem, a simplification technique known as “pruning” is applied. This process is guided by a loss function which allows to reduce the complexity of the tree by removing less significant branches. Pruning is performed by progressively eliminating branches that contribute the least to the overall performance, aiming to achieve a good balance between model simplicity and predictive accuracy (Lewis, 2000).

Logistic is a supervised algorithm used for binary classification tasks. It is widely employed in areas such as fraud detection, medical diagnosis and engineering (e.g. for predicting the probability of failure of a specific process). It is used to model the probability that a given instance belongs to one of two classes of the binary categorial dependent variable. This method uses the logistic function, which is able to convert real values into an interval between 0 and 1: this ensures that the predicted probabilities are in this range

(Ohno-Machado et al., 2002). The logistic function is of the form: $p(x) = \frac{1}{1+\exp(-(x-\mu)/s)}$ where μ is a location parameter (i.e. the midpoint of the curve, where $p(\mu) = 0.5$) and s is a scalar parameter.

4.2) Logic Learning Machine (LLM)

The Logic Learning Machine (LLM) is a rule-based method alternative to decision trees. In plain words, the LLM transforms the data into a Boolean domain where some Boolean functions (namely one for each output value) are reconstructed starting from a portion of their truth table with a method that is described in the paper of Muselli and Ferrari (Muselli et al., 2011). The method creates a set of intelligible rules through Boolean function synthesis following 4 steps. These steps are:

1. Discretization
2. Latticization or Binarization
3. Positive Boolean function
4. Rule generation

In a classification problem, d-dimensional examples $x \in X \subset \Re^d$ are to be assigned to one of q possible classes, labeled by the values of a categorical output y. Starting from a training set S including n pairs $(x_i, y_i), i = 1, \dots, n$, deriving from previous observation, techniques for solving classification problems have the aim of generating a model $g(x)$, called classifier, that provides the correct answer $y = g(x)$, for most input patterns x. In order to analyze the process, a bi-class toy problem is used, whose training set is shown in Table 1. In this example O_0 represents a normal transaction, whilst O_1 represents a fraudulent transaction.

X_1	700	1100	2200	1400	2300	800	1200	2100	2600	2400
X_2	Cheque	Cheque	Cash withdrawal	Cheque	Credit card	Cash withdrawal	Credit card	Cheque	Cheque	ACH
Y	O_0	O_0	O_0	O_0	O_0	O_1	O_1	O_1	O_1	O_1

Table 1: Toy example for describing the LLM working principle

4.2.1) Discretization

In this step, each continuous variable domain is converted into a discrete domain by a mapping. $\psi_j\ X{:}\, X_j \rightarrow I_M$ where X_j is the domain of the j-th variable and $I_M = 1, \dots, M$ is the set of positive integers up to M. The mapping must preserve the ordering of the data. If $x_{ij} \leq x_{kj}$ then $\psi_j(\boldsymbol{x}_i) \leq \psi_j(\boldsymbol{x}_k)$, $\forall j = 1, \dots, d$. One way to describe ψ_j is that it consists of a vector $\boldsymbol{\gamma}_j = (\gamma_{j1}, \dots, \gamma_{jm}, \dots, \gamma_{M_j-1})$ such that:

$$\psi_j(\boldsymbol{x}_i) = \begin{cases} 1, & x_{ij} \leq \gamma_{j1} \\ m, & \gamma_{jm-1} < x_{ij} \leq \gamma_{jm} \\ M_j, & x_{ij} > \gamma_{jM_j-1} \end{cases} \quad (1)$$

There are several strategies for discretization and the simplest one is creating M_j interval having the same length. Let $\boldsymbol{\rho}_j$ be the vector of all the α_j values for input variable j in ascending order $(p_{jl} < p_{jl+1}\ \forall\ l = 1, \dots, \alpha_j)$, then the cutoff γ_{jm} is given by:

$$\gamma_{jm} = p_{j1} + \frac{p_{j\alpha_j} - p_{j1}}{M_j} m \quad (2)$$

This method is referred to as Equal Width discretization. Another approach defines one interval for each value.

4.2.2) Binarization

In this step, each discretized domain is transformed into a binary domain through a mapping $\varphi_j{:}\, I_{M_j} \rightarrow \{0,1\}^{M_j}$, where I_{M_j} is the domain of the j-th variable and $\{0,1\}^{M_j}$ is a string having a bit for each possible value in I_{M_j}. The mapping must maintain the ordering of data: $u < v$ if and only if $\varphi_j(u) < \varphi_j(v)$ where the standard ordering between $\boldsymbol{z}$ and $\boldsymbol{w} \in \{0,1\}^{M_j}$ is defined as follows:

$$\begin{gathered} \boldsymbol{z} < \boldsymbol{w} \text{ if and only if } \begin{cases} \exists\ i & \text{such that } z_i < w_i \\ \forall l\ < i & z_l \leq w_l \end{cases} \\ \boldsymbol{z} \leq \boldsymbol{w} \text{ if and only if } z_i \leq w_i\ \forall\ i = 1, \dots, M_j \end{gathered} \quad (3)$$

if the relation in the equation above holds, then it is said that $\boldsymbol{z}$ covers $\boldsymbol{w}$.

A suitable choice for φ_j is the inverse only-one coding, that for each $k \in I_{M_j}$ creates a string $\boldsymbol{h} \in \{0,1\}^{M_j}$ having all bits equal to 1 except the k-th bit which is set to 0. For example, let $x_{ij} = 3$ with domain I_5, then $\varphi_j(\boldsymbol{x}_i) = 11011$. In this way $\varphi_j(\boldsymbol{x}_i) = \boldsymbol{z}_i$ where $\boldsymbol{z}_i$ is obtained by concatenating $\varphi_j(\boldsymbol{x}_i)$ for $j = 1, \dots, d$. As a result, the new training set is $S' = \{(\boldsymbol{z}_i, y_i)\}_{i=1}^{N}$, with $\boldsymbol{z}_i \in \{0,1\}^B$ where $B = \sum_{j=1}^{d} M_j$. The training set obtained by applying discretization with the single cutoff 1500 for the variable X_1 and subsequent binarization for the toy problem is shown in Table 2.

Z	Y
01 0111	Normal
01 0111	Normal
10 1101	Normal
01 0111	Normal
10 1110	Normal

Z	Y
01 1101	Fraud
01 1110	Fraud
10 0111	Fraud
10 0111	Fraud
10 1011	Fraud

Table 2: Toy example after binarization.

4.2.3) Synthesis of the Boolean function

The training set S', obtained after binarization, can be divided into two different subsets according to the output class: T is the set containing $(\boldsymbol{z}_i, y_i)$ with $y_i = O_1$ whereas F is the set containing the example for which $y_i = O_0$. T and F can be viewed as a portion of the truth table of a Boolean function f that must be reconstructed. Before proceeding with the method description, it is useful to give some definitions and notations.

- Each Boolean function can be written with operators AND, OR, and NOT that constitute the Boolean algebra; if NOT is not considered then a simpler structure, called Boolean lattice, is obtained. From now on, only the Boolean lattice is considered. It can be drawn by positioning $\boldsymbol{z}$ over $\boldsymbol{w}$ if $\boldsymbol{z} > \boldsymbol{w}$ and by linking all the couples $\boldsymbol{z}$, $\boldsymbol{w}$ for which an $\boldsymbol{a}$ does not exist such that $\boldsymbol{w} < \boldsymbol{a} < \boldsymbol{z}$. An example for $\{0,1\}^3$ is shown in Figure 1.b.

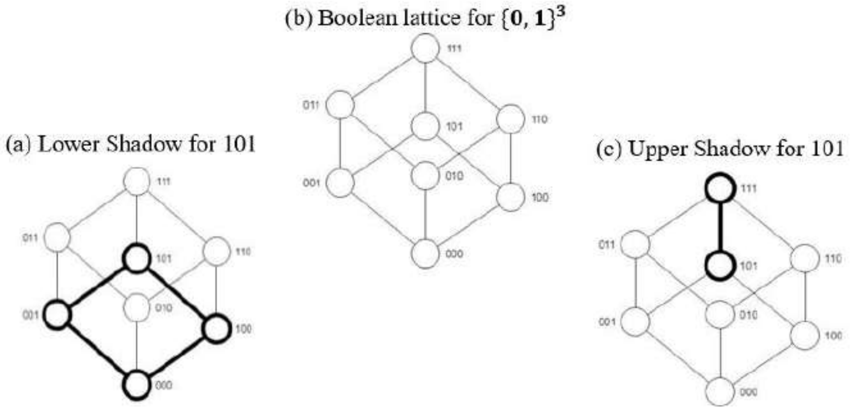
- The sum (OR) and product (AND) of η terms can be denoted as follows:

$$\bigvee_{j=1}^{\eta} z_j = z_1 + z_2 + \dots + z_\eta \quad (4)$$
$$\bigwedge_{j=1}^{\eta} z_j = z_1 \cdot z_2 \cdot \dots \cdot z_\eta = z_1 z_2 \dots z_\eta$$

- A logical product is called an implicant of a function f if the following relation holds: $\bigwedge_{j=1}^{\eta} z_j \le f$, where each element z_j is called literal. The product is called prime implicant if the relation no longer holds when a literal is removed from the implicant.
- The ordering in a Boolean lattice is defined by the equations above; according to this ordering, a Boolean function $f: \{0,1\}^B \to \{0,1\}$ is called positive if $\boldsymbol{z} \le \boldsymbol{w}$ implies $f(\boldsymbol{z}) \le f(\boldsymbol{w})$ for each $\boldsymbol{z}, \boldsymbol{w} \in \{0,1\}^B$.
- A subset $A \subset I_B$ such that for each element $\boldsymbol{z}, \boldsymbol{w} \in A$, an ordering cannot be established (neither $\boldsymbol{z} < \boldsymbol{w}$, nor $\boldsymbol{w} < \boldsymbol{z}$), is called antichain.
- Given $\boldsymbol{a} \in \{0,1\}^B$, then the set $L(\boldsymbol{a}) = \{\boldsymbol{z} \in \{0,1\}^B \mid \boldsymbol{z} \le \boldsymbol{a}\}$ is called lower shadow of $\boldsymbol{a}$, whereas the set $U(\boldsymbol{a}) = \{\boldsymbol{z} \in \{0,1\}^B \mid \boldsymbol{z} \ge \boldsymbol{a}\}$ is called an upper shadow of $\boldsymbol{a}$. The lower and upper shadows for $101 \in \{0,1\}^3$ are shown in Figures 1.a and 1.c.
- Given the subset $T, F \in \{0,1\}^B$, then T is lower separated from F, if there is no element $\boldsymbol{z} \in T$ belonging to the lower shadow of some element of F.
- Given the binary string $\boldsymbol{a}$, if there is a $\boldsymbol{z} \in T$ such that $\boldsymbol{a} \le \boldsymbol{z}$, there is not a $\boldsymbol{w} \in F$ such that $\boldsymbol{a} \le \boldsymbol{w}$, and for each $\boldsymbol{b} < \boldsymbol{a}$, there is $\boldsymbol{w} \in F$ such that $\boldsymbol{b} \le \boldsymbol{w}$, then $\boldsymbol{a}$ is called bottom point for the pair (T, F).
- Every positive Boolean function can be written in its unique, not redundant Positive Disjunctive Normal Form (PDNF) as the sum of its prime implicants: $f(\boldsymbol{z}) = \bigvee_{\boldsymbol{a} \in A} \bigwedge_{j \in P(\boldsymbol{a})} z_j$, where $P(\boldsymbol{a})$ is the subset I_B containing each i such that $a_i = 1$; A is an antichain of $\{0,1\}^B$ and each $\boldsymbol{a}$ is called the minimum true point. For example, the not redundant PDNF $f(\boldsymbol{z}) = z_1 z_3 + z_4$ is obtained from antichain $A = \{1010, 0001\}$.

From these definitions, it follows that a method for finding f must retrieve the set of minimum true points to be used from T and F, in order to represent f in its irredundant PDNF and it follows that the set of all bottom points for (T, F) is an antichain, which elements are candidate minimum true points.
The algorithm employed by LLM to produce implicants is called Shadow Clustering (Muselli et al., 2011). It generates implicants for f through the analysis of the Boolean lattice $\{0,1\}^B$. The algorithm selects a node in the diagram and generates bottom points (T, F)

by descending the diagram: moving down from a node to another node is equivalent to changing a component from 1 to 0 and a bottom point is added to A when any further move down leads to a node belonging to the lower shadow of some $\boldsymbol{w} \in F$.
In particular, the starting node is chosen between the $\boldsymbol{z} \in T \subset \{0,1\}^B$ that do not cover any point $\boldsymbol{a} \in A$ such that $\boldsymbol{a} \leq \boldsymbol{z}$ (in other words the algorithm ends when each element in T covers at least one element in A). Once A has been found, it is possible that it contains redundant elements and consequently, it must be simplified in order to find A^*, from which the PDNF of the positive Boolean function f can be derived.


(b) Boolean lattice for $\{\mathbf{0},\mathbf{1}\}^3$

(a) Lower Shadow for 101

(c) Upper Shadow for 101

Figure 1: Boolean Lattice diagram. Source: Muselli et al., 2011

Different versions of Shadow Clustering exist depending on the choice of the element to be switched from 1 to 0 at each step of the diagram descent. For example, Maximum-covering Shadow Clustering (MSC) at each step changes the i-th element that maximizes the associated potential covering, defined as the number of elements $\boldsymbol{z} \in T$ for which $z_i = 0$.
As concerns the selection of $A^* \subset A$, a possible choice is to subsequently add to A^* the element of A that covers the highest number of points in T that are not covered by any other element of A^*. The application of the Shadow Clustering algorithm (Algorithm 1) to the dataset after binarization shown in Table 2 produces the implicants:

100011 which corresponds to $z_1 \wedge z_5 \wedge z_6$
011100 which corresponds to $z_2 \wedge z_3 \wedge z_4$

Then, the PDNF of the resulting Boolean function is the following: $f(\boldsymbol{z}) = (z_1 \wedge z_5 \wedge z_6) \vee (z_2 \wedge z_3 \wedge z_4)$

Algorithm 1 Shadow Clustering algorithm (bottom-up)

```
Data: $P(x)$
$I = P(x)$;
$A = \emptyset$;
while $I \neq \emptyset$ do
    choose $i \in I$ and remove it from $I$
    if there is $y \in F$ such that $p(I \cup A) \leq y$ then add $i$ to $A$
end
Return $p(A)$.
```

4.2.4) Rule generation

In the last step, each implicant of the positive Boolean function f is transformed into an intelligible rule, where, as said before, a function is generated for each output value, and then the consequent of the rules only depends on f. The transformation considers the coding applied during binarization. In particular, $\boldsymbol{z}$ was obtained by concatenating the results of the mapping $\varphi_j(\boldsymbol{x})$ for each $j = 1, \dots, d$ and consequently it can be split into substring $\boldsymbol{h}_j$ for each attribute, whose bit $z_i \in \boldsymbol{h}_j$ corresponds to a nominal value if X_j is nominal, whereas it corresponds to an interval if X_j is ordered.
For each implicant, a rule in IF − THEN form is generated by adding a condition for each attribute X_j as follows:

- If $z_i = 0$ for each $z_i \in \boldsymbol{h}_j$, then no condition relative to X_j is added to the rule;
- If X_j is nominal, then a condition $X_j \in V$ is added to the rule, where V is the set of values associated with each $z_i \in \boldsymbol{h}_j$ such that $z_i = 0$;
- If X_j is ordered, then a condition $X_j \in V$ is added to the rule, where V is the union of the intervals associated with each $z_i \in \boldsymbol{h}_j$ such that $z_i = 0$.

For the implicant 100011 obtained in the previous step, $\boldsymbol{h}_1 = 10$ leads to the condition $X_1 \in (1700, inf)$ or $X_1 > 1700$ and $\boldsymbol{h}_2 = 0011$ leads to the condition $X_2 \in \{Check, ACH\}$. Then the rule relative to 100011 is: IF $X_1 > 1700$ AND $X_2 \in \{Check, ACH\}$ THEN $Y = O_1$

For the implicant 011100 obtained in the step described previously, $\boldsymbol{h}_1 = 01$ leads to the condition $X_1 \in (-inf, 1700]$ or $X_1 \leq 1700$ and $\boldsymbol{h}_2 = 1100$ leads to the condition $X_2 \in \{Cash\ withdrawal, Credit\ card\}$. Then the rule relative to 011100 is: IF $X_1 \leq 1700$ AND $X_2 \in \{Cash\ withdrawal, Credit\ card\}$ THEN $Y = O_1$

If X_j is ordered, conventionally the upper bound of the interval, if finite, is always included in the condition, whereas the lower bound is excluded. In order to generate the rule for the other class, it is sufficient to label O_0 with 1 and O_1 with 0. In the case of the multiclass problem, it is sufficient to decompose the problem into several bi-class problems for each of the sub-problems the target class is labelled with 1, and all the remaining with 0.

4.2.5) Rule quality and class prediction

The process described in the previous subsection implies that each element $\boldsymbol{x}_i$ of the training set only satisfies rules associated with the output class of $\boldsymbol{x}_i$, but since data are affected by noise, usually it is preferable to admit some errors in order for the model to be able to generalize. In order to permit a fraction of error, the descent of the diagram does not stop when a further move down leads to the lower shadow of some $\boldsymbol{w} \in F$, but still allowing it to go on until a further move leads to a node belonging to the lower shadow of a percentage element $\boldsymbol{w} \in F$ greater than a regularization parameter ε_{max}. Then, it is usual that an element of the training set covers the rule of different classes. When it happens, the output class is established according to the relevance of the rules satisfied by it.

In order to present relevance, the following quantities relative to a rule r in the IF $< premise >$ THEN $< consequence >$ form are introduced:

- $TP(r)$ is the number of training set examples that satisfy both the premise and the consequence of the rule r;
- $FP(r)$ is the number of training set examples that satisfy the premise but do not satisfy the consequence of the rule r;
- $TN(r)$ is the number of training set examples that do not satisfy either the premise or the consequence of the rule r;
- $FN(r)$ is the number of training set examples that do not satisfy the premise and satisfy the consequence of the rule r.

Please note that an example $\boldsymbol{x}_i$ satisfies the premise of the rule r if it satisfies all its premise conditions, whereas $\boldsymbol{x}_i$ does not satisfy the premise of the rule r if it does not satisfy at least one among its premise conditions. Combining these quantities, it is possible to compute quality measures for a rule r:

Covering: $C(r) = \frac{TP(r)}{TP(r)+FN(r)}$ (5)

Error: $E(r) = \frac{FP(r)}{TN(r)+FP(r)}$ (6)

It is evident that the greater the covering, the more relevant the rule is; on the other hand, the smaller the error, the less relevant the rule is. Then, the relevance of a rule r is obtained by combining $C(r)$ and $E(r)$: $R(r) = C(r)(1 - E(r))$.

Once the relevance of the rule is defined, it is possible to use it to compute a score $S(\boldsymbol{x}_i, o)$ for each class o that measures how likely it is that $y_i = o$:

$$S(\boldsymbol{x}_i, o) = \sum_{r \in \mathcal{R}_o^i} R(r) \quad (7)$$

with $\mathcal{R}_o^i = \{r \mid r \in \mathcal{R}, r \leq \boldsymbol{x}_i, O(r) = o\}$, where $\mathcal{R}$ is the complete ruleset, $r \leq \boldsymbol{x}_i$ denotes that $\boldsymbol{x}_i$ satisfies the premise of the rule. On the other hand, to obtain a measure of relevance $R(c)$ for a condition c included in the premise part of a rule r, the rule r' can be considered, obtained by removing that condition from r. Since the premise part of r' is less stringent, we obtain that $E(r') \geq E(r)$ so that the quantity $R(c) = (E(r')–E(r))C(r)$ can be used as a measure of relevance for the condition c of interest. $O(r) = o$ denotes the consequence of r predict class o. Then $\mathcal{R}_o^i$ is the set of rules satisfied by $\boldsymbol{x}_i$ that predict class o. From the scores of each output class, it is possible to define the probability that $y_i = o$:

$$P(o \mid \boldsymbol{x}_i) = \frac{S(\boldsymbol{x}_i, o)}{\sum_{k \in O} S(\boldsymbol{x}_i, k)} \quad (8)$$

Then the selected output is the one that maximizes the output probability: $\tilde{y}_i = \underset{o}{max}\, P(o \mid \boldsymbol{x}_i)$

4.2.6) Feature ranking

For every ordered variable $x_j \in Z$, let us denote with $M_j - 1$ the collection of all the thresholds γ_{jl} involved in the conditions of rules r_k; through these thresholds the domain of the component x_j is subdivided into M_j adjacent intervals $[-\infty,\gamma_{j1}]$, $(\gamma_{j1},\gamma_{j2}]$, …, $(\gamma_{j,l-1},\gamma_{jl}]$, …, $(\gamma_{j\,M_j-1}, +\infty]$. Let us denote with J_{j1}, J_{j2}, …, J_{M_j} these intervals, so that $J_{j1} = [-\infty,\gamma_{j1}]$, $J_{j2} = (\gamma_{j1},\gamma_{j2}]$, etc.

Now, if a rule $r_k \in \mathcal{R}_O = \{r \mid r \in \mathcal{R}, O(r) = o\}$ for the output class o (i.e. whose consequence part is $y = o$) includes a condition c_{kl}, with relevance $R(c_{kl})$, involving the ordered component x_j, the points of m_{kl} of the M_j adjacent intervals verify that condition. For instance, if the condition c_{kl} is $x_j \leq \gamma_{j3}$, the points of the $m_{kl} = 3$ intervals J_{j1}, J_{j2} and J_{j3} satisfy c_{kl}. It is then possible to retrieve a measure of relevance $R_k{}^o(J_{ji})$ for each interval J_{ji}, with respect to the output class o, by looking at the quantities $R(c_{kl})$ of the conditions c_{kl}, that are included in rules r_k, that involve the component x_j, and are verified by points of J_{ji}. In particular, if a condition c_{kl} involving x_j is satisfied by m_{ki} of the M_j adjacent intervals, the relevance quantity that can be attributed to each of these intervals is $R_k{}^o(J_{ji}) = R(c_{kl})/m_{kl}$.

By collecting all the relevancies derived from all the rules $r_k \in \mathcal{R}_O$ including a condition c_{kl} on the component x_j, we can obtain the measure of relevance $R^h(J_{ji})$ of the interval J_{ji} with respect to the output class o:

$$R^o(J_{ji}) = 1 - \prod_{r_k \in \mathcal{R}_O} \left(1 - R_k^o(J_{ji})\right) \quad (9)$$

Starting from Eq. (9) a measure of relevance $R^o(x_j)$ for the component x_j (with respect to o) can be derived by considering the variation of $R^o(J_{ji})$ over the M_j adjacent intervals $J_{j1}, J_{j2}, \ldots, J_{M_J}$. In fact, if $R^o(J_{ji})$ does not change so much in these intervals, then different thresholds are essentially used to determine parts of the input domain where the behavior of the model $g(\boldsymbol{x})$ is similar. This means that the variable x_j has little discriminant power among different classes, but it characterizes the input domain with respect to $g(\boldsymbol{x})$ for the output class o.

A possible way of measuring the variation of a quantity is to consider its standard deviation σ; therefore, we have:

$$R^o(x_j) = M_j\, \sigma_j\left(R^o(J_{ji})\right) \quad (10)$$

where σ_j stands for the standard deviation over the M_j intervals $J_{j1}, J_{j2}, \ldots, J_{M_j}$.

A sign for $R^o(x_j)$, which indicates if the variable x_j is directly (if the sign is positive) or inversely (if the sign is negative) correlated with the output class o, can also be retrieved by looking where higher values of $R^o(J_{ji})$ are located. In particular, if higher values of $R^o(J_{ji})$ occur at higher (resp. lower) i then the variable x_j is directly (resp. inversely) correlated with the output class o.

Hence, a procedure for deriving the sign of $R^o(x_j)$ consists in subdividing the product of (1) in two parts: the first one, denoted with $R^{o-}(J_{ji})$, contains terms $R_k(J_{ji})$ originated by conditions c_{kl} of the form $x_j \leq \gamma_{ji}$, whereas $R^{o+}(J_{ji})$ includes terms $R_k(J_{ji})$ derived by conditions c_{kl} of the kind $x_j > \gamma_{ji}$. As for conditions c_{kl} of the form $\gamma_{ji_1} < x_j \leq \gamma_{ji_2}$ terms $R_k(J_{ji})$ for $i \leq (i_1+i_2)/2$ (resp. $i > (i_1+i_2)/2$) are inserted into $R^{o-}(J_{ji})$ (resp. $R^{o+}(J_{ji})$). With these definitions, the sign of $R^o(x_j)$ becomes negative if $R^{o-}(J_{ji}) < R^{o+}(J_{ji})$ and positive in the opposite case.

If the variable x_j is nominal, then equation (1) can still be used to determine measures of relevance $R^o(J_{ji})$ if $G_j = \{v_{j1}, v_{j2}, \ldots\}$ is the collection of the possible values assumed by x_j and $J_{ji} = \{v_{ji}\}$, for $i = 1, 2, \ldots, |G_j|$. In this case equation (9) becomes:

$$R^o(x_j) = |G_j|\sigma_j\left(R^o(J_{ji})\right) \quad (11)$$

a sign for $R^o(x_j)$ cannot be determined and is therefore always considered as positive.

If the (absolute) maximum over the q output classes of the quantities $R^o(x_j)$ is greater than 1, then all the relevancies $R^o(x_j)$ are normalized to this maximum so that their values lie in the range [0,1]. By averaging the quantities $R^o(J_{ji})$ and $R^o(x_j)$, for $o = 1, \ldots, q$, we can obtain absolute measures of relevance $R(J_{ji})$ and $R(x_j)$ (independent of the output class o) for J_{ji} and for the variable x_j:

$$R(J_{ji}) = \frac{1}{q}\sum_{o=1}^{q} R^o\,(J_{ji}) \quad , \quad R(x_j) = \frac{1}{q}\sum_{o=1}^{q} R^o\,(x_j) \quad (12)$$

In short, as regards Logic Learning Machine models, feature importance can be analyzed based on the generated rules and their frequency and predictive strength. In fact, the Logic Learning Machine does not use coefficients such as the Logistic task or the number of the splits like the Decision tree task, but instead, its feature importance is based on the relevance of the features in the ruleset extracted from the model. By using feature ranking applied to LLM, it is possible to inspect the presence and weight of attributes in the final ruleset. This task provides an analysis of the feature importance by counting how many times a feature appears in the model rules and generates a ranking of features, showing which ones had the greatest impact. The more a feature appears in important rules, the more impact it has on model decisions.

4.3) Evaluation metrics

As previously stated, the objective of this study was to confirm whether the combination of Machine Learning algorithms with heuristic rules could lead to an improvement of the results in the detection of fraudulent transactions. Therefore, our aim was to analyze and find sensible heuristic rules that could somehow bring an improvement both in terms of precision and explicability of the results. In writing and selecting the rules, we therefore tried to combine the information deriving from regulations and from the SAML-D paper, to obtain a general picture of the topic and to summarize this information within our personal ruleset.

In order to evaluate the performance of the heuristic rules, we employed the "covering" and "error" statistics previously analyzed (see Eq. 5 and 6). The covering describes the percentage of samples that are covered by that rule, in a class, compared to the total samples in that class. We want this value to be the highest possible, i.e. ≈ 1. The error, on the contrary, measures the percentage of errors within the covering of the rule, i.e. how often the rule is wrong within the covering. In this case we aim to obtain a small error, i.e. ≈ 0. These statistics were computed to later calculate further metrics that allowed us to select only the best performing rules and group them together.

In this phase, the preliminary step was to compute specific metrics in order to filter and pick only the best rules for each of them. To this aim, we used the error and covering statistics we previously calculated. Specifically, we computed the following metrics:

- **Error/Covering**: this metric simply performs the ratio between the two core statistics. It indicates the proportion of error compared to how much the rule is applicable in the dataset. Since we wanted our rules to maximize the covering and minimize the error, we consequently selected only those rules who scored the lowest results for this metric;

- **Score**: this metric is useful for balancing precision and generalization. Specifically, it "discounts" the covering for the error. The formula to describe it is:

$$Score = Covering \cdot (1 - Error) \quad (13)$$

- **Score (1)**: this is an alternative version of the previous score metric. In this case we reduce the weight of the covering based on the ratio between error and covering. The formula to describe it is:

$$Score_1 = Covering \cdot \left(1 - {}^{Error}\!/_{Covering}\right) \quad (14)$$

The aim at this point was to rank the rules according to the metrics just described, to select only the best performing ones for each of them. Consequently, we filtered the first 10 rules which scored the best and selected them to later complement the classification models in the merging phase between heuristics and Machine Learning.

As an example of one of the best-performing rules identified, we present the "*AMLCheckUAE*" rule. This rule was selected based on its high covering and low error, reflecting its effective application in detecting potentially fraudulent transactions. The rule flags transactions as fraudulent if either the sender's or receiver's bank location is in the United Arab Emirates (UAE), and if the transaction amount exceeds the defined threshold for specific payment types. The logic behind this rule stems from the known risks associated with high-value transactions in particular locations and payment methods. By applying this rule, we are able to identify potentially suspicious transactions and flag them as fraudulent.

In general, the results seem to show that the best rules obtained are quite specific, in the sense that they capture very few fraud cases in the dataset. This is not necessarily a bad thing but this lack in intercepting fraudulent instances may indicate a scarce relevance in terms of improvement of results. In other words, the most prominent heuristic rules we defined are generally very small, that is, they have a negligible weight compared to what classification algorithms can achieve (they are often precise but have little generality).

To compare the employed classification models' performance, we adopted several evaluation metrics. Each of them provides specific information on the prediction quality, allowing for an extensive analysis of the results. This was done to discriminate against the model and the specific parameterization which performed best among all the ones we implemented.

- **AUC**: The AUC (Area Under the Curve) measures the ability of a model to distinguish between two target classes, in this case between being a laundering transaction or a normal one. It is computed exploiting the Receiver Operating Characteristic (ROC) curve, which represents, for different thresholds, the tradeoff between the True Positive Rate (TPR), also known as sensitivity, and the False Positive Rate (FPR). In this study we have implemented it by applying the roc function to the score of the model predictions. The results in terms of AUC vary in the range [0.5, 1], where the left bound indicates a non-discriminating model, while the right one denotes a perfectly discriminating model.

- **Precision**: The Confusion matrix shows the number and percentage value of correctly and incorrectly classified observations. In this context, the precision statistic measures the proportion of correct positive predictions with respect to the total number of positive predictions made by the model. In mathematical terms, this is defined as:

$$Precision = \frac{TP}{TP+FP} \quad (15)$$

  High values of this metric indicate that the model is effective at reducing false positives, meaning it rarely misclassifies normal transactions as fraudulent. Optimizing precision is crucial in fraud-detection analysis, mainly because false alarms generate a cost for the agency that controls suspicious cases. Therefore, the main goal is to reduce the number of false positives and improve the overall effectiveness of the model.

- **Youden J statistic**: This statistic measures a classification model's ability to discern between two classes. In mathematical terms, it is described as:

$$J = TPR + TNR - 1 \quad (16)$$

It combines the model sensibility, captured by the true positive rate, with its specificity, indicated by the true negative rate, into a synthetic index. Youden's J statistic has proven to be a very useful index for two main reasons. First of all because it is not affected by class imbalance which, as we know, is a problem affecting SAML datasets, and second because it is only valid for binary classification problems. Its value ranges between [-1, 1], where J=0 indicates that the model has no discriminating ability (it has the same behavior as the random case); J=1 denotes a perfect classifying model which commits no errors; whilst negative values signal rare pathological cases where the model performs worse than the random case.

4.4) SAML Dataset at a glance

To address the analyzed problem, the dataset used was the Synthetic AML Dataset (SAML-D), available on Kaggle (Oztas et. al, 2023). SAML-D incorporates 12 features and 28 typologies of transactions (see Appendix B), split between 11 normal and 17 suspicious, making it one of the most comprehensive synthetic AML datasets available. These typologies have been selected based on existing datasets, academic literature, and interviews with AML specialists.

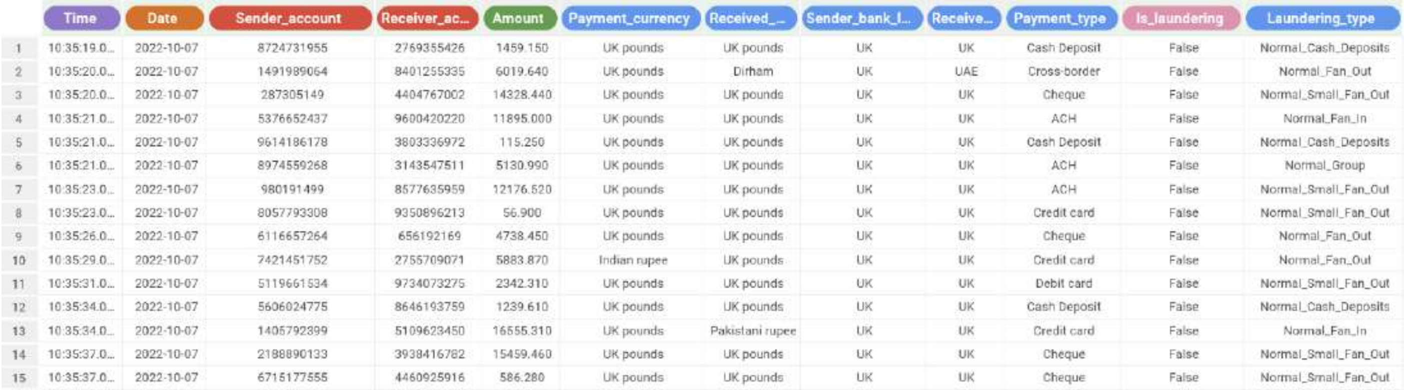
For the construction of SAML dataset certain rules and filters were implemented. In particular, the generation process of both "normal" and "suspicious" transactions involved two methods: the agent-based approach and the typology-based approach. This implies that the dataset includes prior assumptions about what constitutes normal and suspicious behaviors. For instance, fraudulent transactions are generated exploiting specific typologies like "structuring" or "deposit and send" which are characterized by specific patterns.

As a result, given the artificially encoded structure of these typologies, Machine Learning models trained on this dataset could partially learn to recognize these specific patterns rather than truly uncover novel laundering strategies. Consequently, this could limit the model's ability to generalize when applied to real-world data, which could feature new laundering behaviors not covered by the typologies simulated in the generation process (Oztas et. al, 2023).

In order to add complexity and realism to the data, observations include innovative features such as the geographic location of accounts, which also contains high-risk countries in the AML field, and high-risk payment types. In this sense, complexity and realism are also achieved by making fraudulent accounts carry out a wide range of money laundering types in addition to normal transactions (Oztas et al., 2023). Lastly, since the dataset was created with a focus on the United Kingdom, the prevalence of its observations, 99.72% if considering both inwards and outwards transactions, is therefore located in the UK.

The dataset comprises 9,504,852 transactions, of which 0.1039% are suspicious, thus showing a great class imbalance. This generally poses serious concerns in classification problems as the Machine Learning model implemented could develop bias towards the majority class, in this case non-fraud transactions. In other words, this can lead to a model that poorly learns the minority class, the one we are interested in, because it has few examples in the dataset.

When working with heavily unbalanced datasets, the absence of corrective actions can lead to sub-optimal results or misinterpretation of them. In particular, high false negative rates are likely to be obtained, as models tend to favor the majority class, partially or totally ignoring the minority one. This can make some common metrics, such as accuracy, unrepresentative of the model's real predictive capacity. Therefore, in these cases, it is essential to be aware of this issue and adopt appropriate metrics that take into account the imbalance. Moreover, it is also crucial to consider the impact of such an imbalance when comparing different models or between different parameterizations to avoid misleading conclusions based on poorly informed indicators.



	Time	Date	Sender_account	Receiver_ac...	Amount	Payment_currency	Received_...	Sender_bank_l...	Receive...	Payment_type	Is_laundering	Laundering_type
1	10:35:19.0...	2022-10-07	8724731955	2769355426	1459.150	UK pounds	UK pounds	UK	UK	Cash Deposit	False	Normal_Cash_Deposits
2	10:35:20.0...	2022-10-07	1491989064	8401255335	6019.640	UK pounds	Dirham	UK	UAE	Cross-border	False	Normal_Fan_Out
3	10:35:20.0...	2022-10-07	287305149	4404767002	14328.440	UK pounds	UK pounds	UK	UK	Cheque	False	Normal_Small_Fan_Out
4	10:35:21.0...	2022-10-07	5376652437	9600420220	11895.000	UK pounds	UK pounds	UK	UK	ACH	False	Normal_Fan_In
5	10:35:21.0...	2022-10-07	9614186178	3803336972	115.250	UK pounds	UK pounds	UK	UK	Cash Deposit	False	Normal_Cash_Deposits
6	10:35:21.0...	2022-10-07	8974559268	3143547511	5130.990	UK pounds	UK pounds	UK	UK	ACH	False	Normal_Group
7	10:35:23.0...	2022-10-07	980191499	8577635959	12176.520	UK pounds	UK pounds	UK	UK	ACH	False	Normal_Small_Fan_Out
8	10:35:23.0...	2022-10-07	8057793308	9350896213	56.900	UK pounds	UK pounds	UK	UK	Credit card	False	Normal_Small_Fan_Out
9	10:35:26.0...	2022-10-07	6116657264	656192169	4738.450	UK pounds	UK pounds	UK	UK	Cheque	False	Normal_Fan_Out
10	10:35:29.0...	2022-10-07	7421451752	2755709071	5883.870	Indian rupee	UK pounds	UK	UK	Credit card	False	Normal_Fan_Out
11	10:35:31.0...	2022-10-07	5119661534	9734073275	2342.310	UK pounds	UK pounds	UK	UK	Debit card	False	Normal_Small_Fan_Out
12	10:35:34.0...	2022-10-07	5606024775	8646193759	1239.610	UK pounds	UK pounds	UK	UK	Cash Deposit	False	Normal_Cash_Deposits
13	10:35:34.0...	2022-10-07	1405792399	5109623450	16555.310	UK pounds	Pakistani rupee	UK	UK	Credit card	False	Normal_Fan_In
14	10:35:37.0...	2022-10-07	2188890133	3938416782	15459.460	UK pounds	UK pounds	UK	UK	Cheque	False	Normal_Small_Fan_Out
15	10:35:37.0...	2022-10-07	6715177555	4460925916	586.280	UK pounds	UK pounds	UK	UK	Cheque	False	Normal_Small_Fan_Out

Figure 2: SAML-D at a glance.

An ongoing challenge in the AML framework is comparing the results of different Machine Learning algorithms, since the relative experiments are often conducted using datasets with distinct characteristics (Oztas et al., 2022). Thus, the main objective of the researchers who created the SAML-D dataset was to address this challenge by providing peer researchers with a challenging benchmark for evaluating classification models and enabling consistent results comparison, consequently supporting more meaningful analysis. Furthermore, SAML-D also aims to overcome the lack of data for AML analyses, mostly due to legal and privacy limitations that severely limit researchers' possibilities (Jullum et al., 2020).

4.5) Implementation and comparison of classification models

In this section, we describe the practical application of the models in the project, with particular attention to their general functioning and the reasons that guided their implementation. In this sense, after finding the best configuration, the main objective was to evaluate the performance of the Logic Learning Machine task in two different scenarios:

- **Pure Machine Learning:** In this setup, the Logic Learning Machine task is used considering exclusively the initial core attributes of SAML-D, without the integration of additional features based on heuristic rules (this type of models is called "Pure"). The underlying idea was to set a sort of benchmark for this type of classification so that we could compare the results coming from other configurations of the LLM and check whether we obtained any improvements.

- **Combination of Machine Learning and heuristic:** In this case the Logic Learning Machine task is used on the "enriched" dataset we created by adding the features derived from the application of heuristic rules. The aim was to check and consequently gauge potential improvements in the accuracy and interpretability of the results, provided by the introduction of the heuristic rules. This procedure is carried out for two different model variants: the first containing all the heuristic rules we defined (the so-called "All" models), and the second one only containing the set of best rules we previously selected (the so-called "Best Rules" models). This was done to verify whether the Logic Learning Machine task worked better by using all the available information or only by providing it with a part of the total, i.e. the qualitatively better information.

In parallel to this analysis, we also proposed an additional study, using both the Decision tree and the Logistic classification tasks, following the same process proposed for the Logic Learning Machine. This was done primarily to obtain an accurate reference benchmark to compare the results and reach a more comprehensive view of the phenomenon.

In the integration phase between the Logic Learning Machine and the heuristic ruleset, the solution adopted consists in including the attributes derived from the heuristic rules among the input features, allowing the model to independently manage their information content. Subsequently, after the forecast phase, we manually intervene on the prediction, setting the predicted value as "fraudulent" every time a reasonable number of heuristic rules occurs, regardless of the result produced by LLM. This operation is handled by a module that assigns the value "1" every time one of the selected heuristic rules applies. If the sum of such applications exceeds a predefined threshold, the module sets the score value of the observation equal to "1"; otherwise, it retains its original value. Basically, we assign absolute priority to heuristic rules, considering their weight higher than those extracted by the model. However, this forcing process only occurs for a small subset of rules with an extremely low error (≤0.001), which makes them almost flawless. This means that, although these rules rarely fire in the dataset, when they do, we consider them foolproof.

In parallel with this strategy, we also analyzed the binarization of the model. In fact, the standard cutoff of 0.5 for the score may not be optimal, especially in a fraud detection context with highly unbalanced classes. In these cases, the model generally tends to assign lower scores to most observations, leading the default threshold to ignore many frauds and generate a high number of false negatives. To improve the separation between classes, we therefore adopted a new threshold calculated in a data-driven way, exploiting the Youden index, which identifies the optimal balance point between recall and specificity. Once we obtained the new cutoff, we then re-binarized the model using this optimal threshold and computed the AUC metric and the confusion matrix to evaluate the models.

4.6) Self-coding development - Rulex Platform

The so-called self-code platforms are programs that combine the visual approach of no-code programs, i.e. a straightforward WYSIWYG (What You See Is What You Get) interface, with the possibility of developing complex projects thanks to the fact that the platform automatically writes the underlying code. The Rulex Platform is a clear example of a self-code platform, as it offers both a simple and intuitive drag and drop interface and optimized code for every operation it performs in its tasks.

Each executed action, such as data transformation, model creation and custom calculations, is traced and saved in an interactive history table and every step made can be saved, re-executed, undone or deleted. Each single operation can be inspected, making the code behind it visible. In the Rulex Platform, operations are carried out by linking together different blocks which perform specific operations following the direction of the flow. This allows to create a clear and organized succession of computations.

While being a low-code platform, the Rulex Platform offers many tools for implementing advanced customization, thanks to its built-in functions and formulas that can be directly configured by the user. Each individual function, formula and customizable parameter can be examined inside the specific task, allowing the user to access the relative documentation and better comprehend the tool being used. Another key feature of the Rulex Platform is that it includes Machine Learning tasks such as the Logic Learning Machine task, which allows to implement in depth analysis and evaluate models without the need of writing complex programming algorithms.

In short, the Rulex Platform is a versatile user-friendly tool that allows anyone to work with data and perform in-depth analysis, regardless of his initial set of programming skills. The combination of its intuitive interface and wide range of internal tasks makes it a valid tool both for business and academic contexts.

4.7) Flow in the Rulex Platform

In this section we briefly analyze the structure of the flow, describing its main components. A synthetic diagram of the flow is provided in Figure 3.

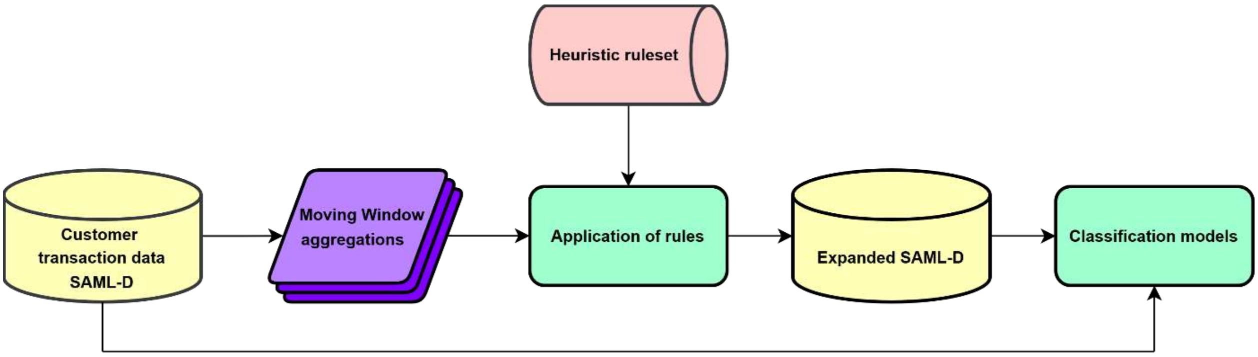


Figure 3: Schematic flow diagram.

After importing the dataset, we implemented the moving window aggregations. The moving window task performs data aggregations over defined time intervals, leveraging measures such as minimum, maximum, mean, and median. This operation can be applied to present, past, or future time frames, with the latter two achieved by shifting the window backward or forward by a fixed time delta. In this study, we employed both present and past aggregations to enable comparative analysis and identify potential suspicious differences in accounts' behaviors. The aggregated data generated through this process was stored in new attributes, which were subsequently used in the heuristic rule application phase. This step was conducted using the rule engine task in the Rulex Platform, which combines the information coming from a dataset with a predefined set of rules. The latter produces new attributes containing the flags raised by each individual rule, whenever a fraudulent pattern is detected.

The rule application phase served a dual purpose: first, to evaluate the heuristic ruleset by computing metrics such as covering, error, and their related performance indicators; second, to generate an expanded version of the original dataset, incorporating the outputs of the rule evaluation. This enhanced dataset was then used as input for the subsequent classification tasks. For consistency and to assess the added value of the heuristic ruleset, the same classification procedure was also applied to the original dataset, enabling a direct comparison of the results. A simplified representation of the flow developed in the Rulex Platform regarding the implementation of the three classification algorithms is provided in Figure 4. As described, the same was performed both on the original SAML dataset and on its enriched version containing heuristic.

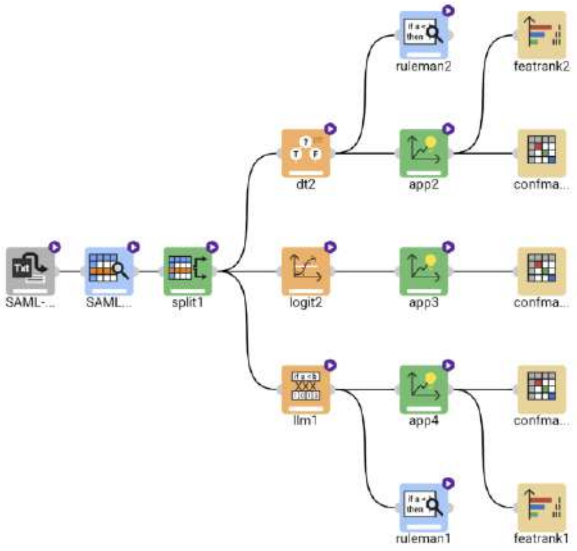


Figure 4: Simplified representation of the Decision Tree, Logistic and Logic Learning Machine classification flow.

5) Results

In this section, we present the results obtained in the various configurations of the models analyzed previously. Table 3 reports the performance of the baseline "Pure" models, providing a reference point for comparison with subsequent ones. In particular, the AUC values on the training and test sets (denoted with "$AUC_{training}$" and "AUC_{test}" respectively) and the computation times for the training and testing phases (denoted with "$t_{training}$" and "t_{test}" respectively and measured in seconds) are examined and compared.

	$AUC_{training}$	AUC_{test}	$t_{training}$	t_{test}
LLM Pure	0.786	0.773	5125	205
Logistic Pure	0.761	0.770	297	325
DT Pure	0.687	0.677	627	15

Table 3: "Pure" models results comparison.

Table 4 shows the results of the models integrating the heuristic rules. The models marked with (*) indicate those in which the heuristic rules have been forced and the binarization has been updated, based on the optimal values obtained through the Youden index.

	$AUC_{training}$	AUC_{test}	$t_{training}$	t_{test}
LLM All	0.861	0.841	19261	161
LLM All (*)	0.861	0.840	19261	161
Logistic All	0.830	0.828	524	306
DT All	0.687	0.677	635	16
LLM Best Rules	0.787	0.778	11281	124
LLM Best Rules (*)	0.787	0.778	11281	124
Logistic Best Rules	0.761	0.770	138	289
DT Best Rules	0.687	0.677	591	14

Table 4: "All", "Best rules" and "(*)" variants models results comparison.

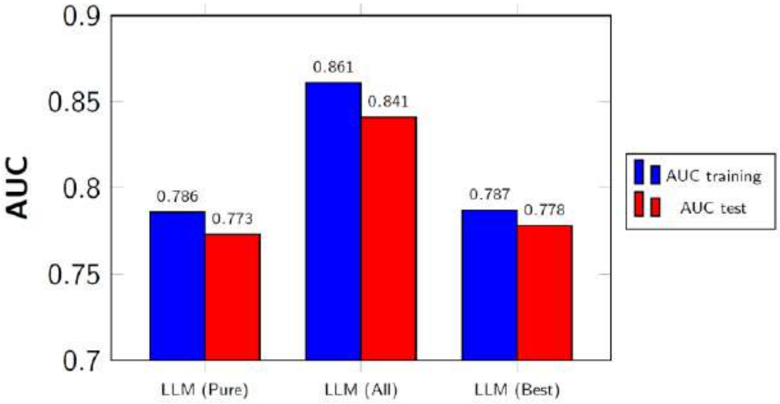


Figure 5: LLM "Pure", "All Rules" and "Best Rules" models AUC comparison.

In this context, it is also particularly useful to analyze ROC curves, as they represent the visual and conceptual counterpart of the AUC. Studying these two elements together provides a more complete view of the discriminative capacity of the model.

The AUC, in fact, makes it possible to assess the goodness-of-fit of the predictive score in a global manner, as it considers the performance of the model for each possible cutoff.

In contrast, point metrics such as the Youden J index focus on a specific cutoff, providing a 'snapshot' of performance at that point, but neglecting the overall behavior of the model.

The latter measures the vertical distance from the random classification line (the bisector starting at point (0,0)) and is therefore a useful tool to identify the optimal balance point between sensitivity and specificity.

In particular, maximizing it allows us to identify the most diagnostically effective decision threshold.

Figures 6, 7 and 8 show the ROC curves on the test set for the models presented in Figure 5.

The first thing that catches the eye when analyzing the results is the inferiority of the Decision tree models compared to the others, in terms of AUC.

However, this outcome was somewhat predictable, considering the intrinsic limitations of the methodology, particularly in contexts characterized by strong class imbalance, such as the one analyzed here.

In such settings, the Decision tree tends to perform poorly, as the algorithm is heavily influenced by the initial root-split, which in turn conditions all the subsequent splits, inevitably compromising the model's ability to effectively detect the minority class (i.e. fraudulent cases).

As a result, Decision tree models fail to produce truly informative trees, substantially returning the same output across all three configurations: "Pure", "All", and "Best Rules".

This highlights a clear limitation, making this methodology not particularly useful for a meaningful comparison with the other methods.

In short, given the significant performance gap, to allow a fairer and more reasonable comparison, the subsequent comparative analysis will focus primarily on the Logistic and Logic Learning Machine models.

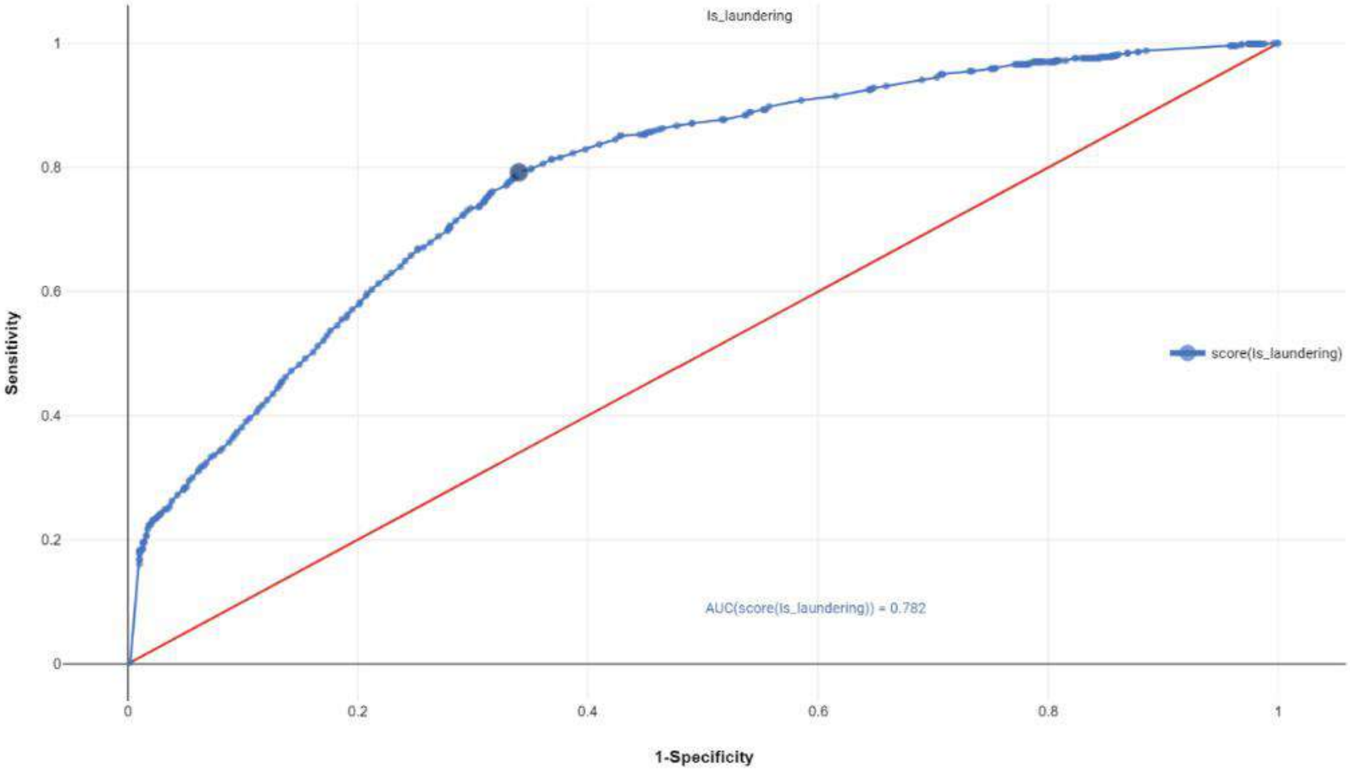


Figure 6: LLM "Pure" ROC curve.

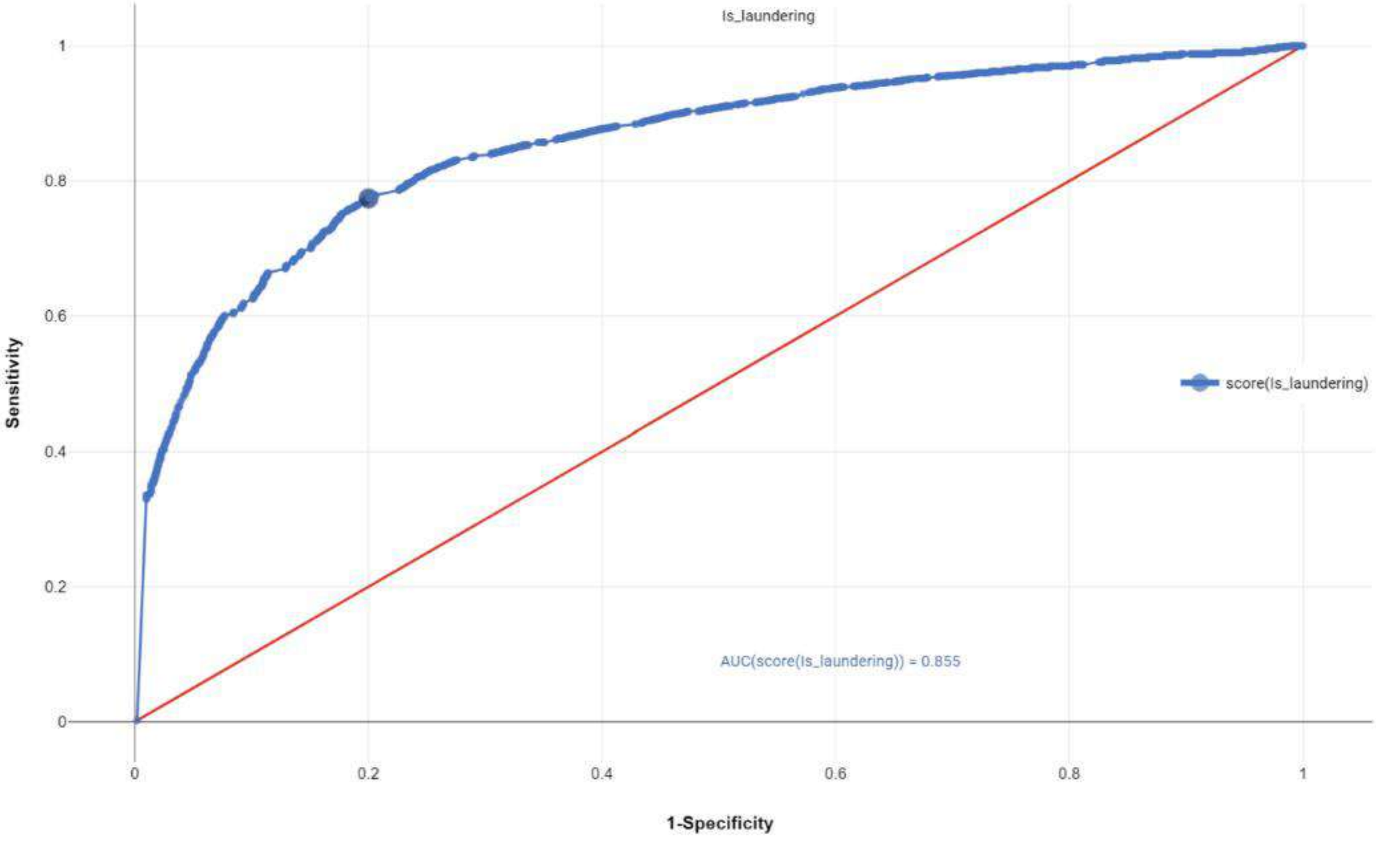


Figure 7: LLM "All" ROC curve.

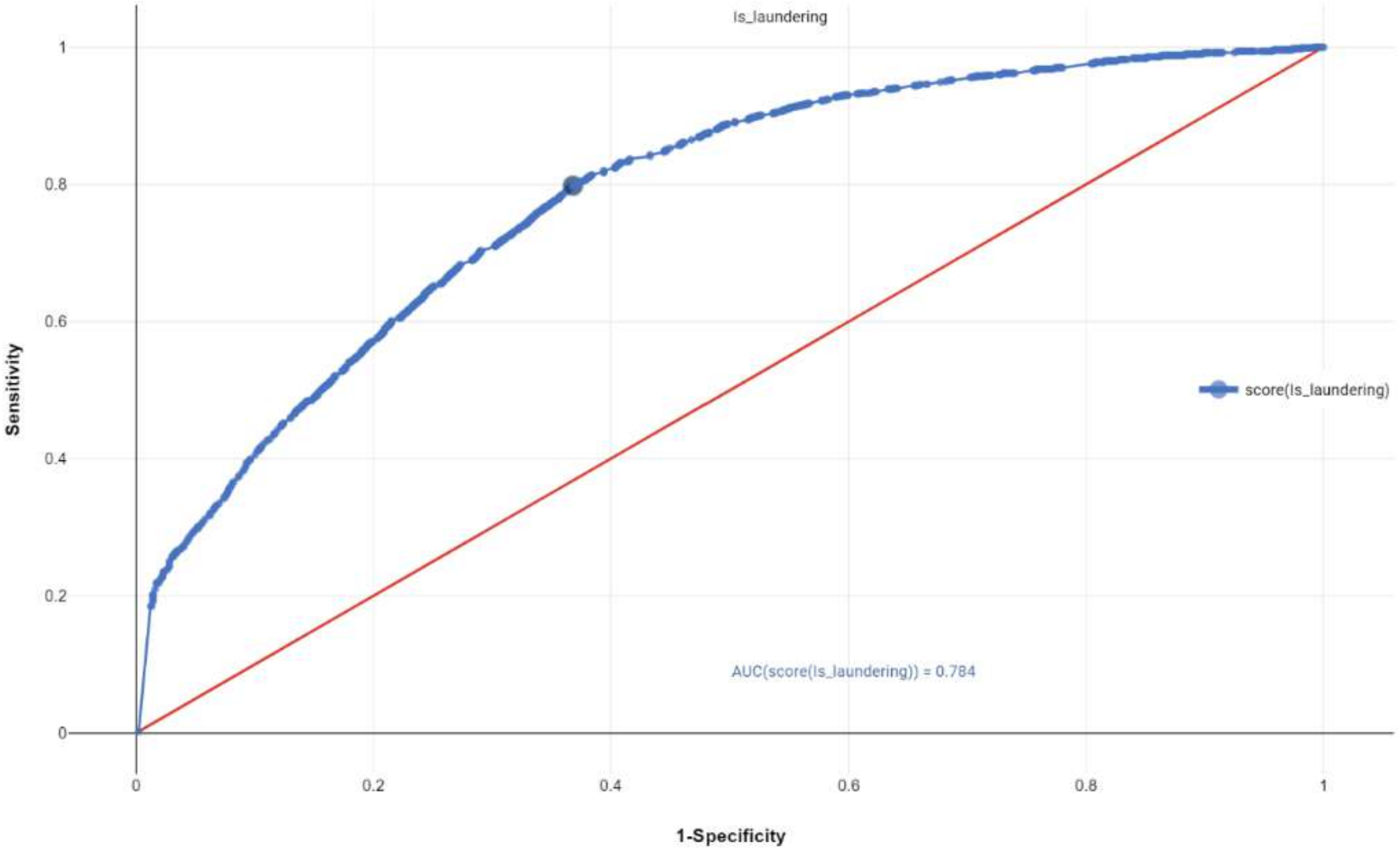


Figure 8: LLM "Best Rules" ROC curve.

5.1) Confusion matrices

To correctly interpret the results obtained from the confusion matrices, it is essential to remember the starting point of the original dataset. The latter, as already highlighted, presents a strong class imbalance, with only 0.1% of fraudulent cases.
Consequently, apparently low precision values are not necessarily disappointing, but instead they represent a significant improvement compared to random classification. For example, a precision of 1%, which may seem unsatisfactory at first sight, actually indicates a model ten times more precise than a random classification.
In general, the results obtained for precision and recall statistics tend to be extreme in opposite directions: either very low precision with high recall is recorded, or the opposite occurs. This imbalance makes both metrics uninformative for a significant descriptive analysis. For a more balanced evaluation, it is more appropriate to consider a synthetic indicator such as Youden's J statistic, which measures the discriminating capacity of the model, without privileging a single aspect, as precision and recall do. This allows for a more reliable evaluation of the overall effectiveness of the model, avoiding misleading interpretations due to unilaterally optimized metrics. A summary of the results is reported in Table 5:

	Youden's J statistic
LLM Pure	0.151
Logistic Pure	0.005
LLM All	0.318
LLM All (*)	0.575
Logistic All	0.006
LLM Best Rules	0.173
LLM Best Rules (*)	0.430
Logistic Best Rules	0.005

Table 5: "Pure", "All", "Best rules" and "(*)" variants Youden's J statistic results comparison.

From the results reported in Table 5, it clearly emerges that the Logic Learning Machine models' configurations obtain the best performance. In particular, the LLM All (*) model stands out with a value of 0.575, showing a good discriminating capacity and demonstrating to be the most effective model in this setting, also considering the strong results previously observed in terms of AUC. In this case, the combination of the Logic Learning Machine and heuristic rules resulted in the generation of 243 rules. The best five rules, in terms of covering, are reported in Table 6. These rules are defined by the following conditions: as can be observed, many of these rules contain conditions regarding the transaction amount and payment types, suggesting the high informational value of these attributes for detecting suspicious behaviors. Some rules also contain conditions that exploit heuristic attributes regarding historical aggregation (e.g. "PayType1D1M" in rule #4), indicating that also the temporal evolution of transactions plays a relevant role in the identification of anomalies.

	Cond. #1	**Cond. #2**	**Cond. #3**	**Cond. #4**	**Covering**
Rule #1	"Amount" > 2033.765	"Payment_currency" in [UK pounds]	"Payment_type" in [Cash Deposit]	"CurPairs1M1M H" in [No]	11.206
Rule #2	"Amount" > 2564.305	"Payment_type" in [Cash Deposit]			9.122
Rule #3	"Payment_type" in [Cash Withdrawal]	"PayType1W1W H" in [No]			8.664
Rule #4	"Amount" > 177.865	"Payment_type" in [Cash Deposit, Cash Withdrawal]	"PayType1D1M" in [Yes]		7.732
Rule #5	"Amount" <= 16434.655	"Payment_currency" in [UK pounds]	"Payment_type" in [Cheque, Credit card, Debit card]	"PayType1M1M H" in [No]	6.83

Table 6: LLM "All (*)" first best rules in terms of covering.

6) Conclusions

To choose the most suitable model, it is essential to adopt an objective-based approach, that is, to identify the most appropriate trade-off between two opposite scenarios, depending on the specific needs of the case. On the one hand, if the main objective is to identify fraudulent cases with maximum precision, it is appropriate to adopt models characterized by high levels of precision. However, this strategy involves an inevitable trade-off: a reduction in recall, with the risk of labeling many fraudulent cases as false negatives. This approach is particularly suitable when the system is unable to handle a large number of reports and must therefore prioritize the quality of identifications over quantity.
In real case scenarios, however, the main problem is not false negatives, which represent a necessary trade-off, but rather false positives, which can generate high costs and inefficiencies, without leading to an actual improvement in fraud detection.
On the other hand, if you want to preserve recall, i.e. maintain the maximum possible number of fraud reports, you need to opt for more balanced models. This choice allows you to intercept a greater number of illicit activities but also involves an increase in operational costs and potential inconvenience for customers. In our analysis, the model that has demonstrated the most balanced performance, and is therefore the most suitable choice for this scenario, is LLM All (*).

In short, the selection of the most appropriate model essentially depends on two factors that must be carefully balanced: the operational costs related to the management of reports and the number of frauds actually detected and reported.
This paper confirmed the results reported in the literature, demonstrating how the integration between Machine Learning and heuristic rules can significantly improve the predictive performance of classification models. In this specific case, with the adopted parameterizations, Logic Learning Machine models proved to be the best choice for fraud detection, clearly outperforming the Decision tree and Logistic regression algorithms. Among these, the most balanced and performing model was found to be LLM All (*), as it best combined the advantages of heuristic information with the benefits deriving from rule forcing and re-binarization.
Although the results obtained are very promising, there is still room for improvement. Specifically, a fundamental evolution concerns the continuous updating of the ruleset, introducing new rules to counter the evolution of money laundering techniques and guaranteeing a constantly effective detection system. Finally, a crucial step will be to replicate the analysis on new compatible transaction datasets. The money laundering problem is highly dependent on the quality and variety of available data, which implies that the performance of a model can vary significantly based on the information used for training and testing. Testing models on different datasets would allow to obtain more stable results and to conduct a more solid and coherent analysis.
Further developments could also explore the adoption of alternative model configurations specifically designed to maximize precision. while preserving an acceptable level of recall. By accurately analyzing and selecting the right value for the "confidence" parameter of the models, it is indeed possible to boost precision metric and control the trade-off with recall, tailoring the model's behavior to operational constraints. Moreover, the fine-tuning of heuristic rules through the Rule Enhancer module, a Rulex tool capable of automatically adjusting thresholds and nominal values based on a user-defined metric, may become a key tool for optimizing model performance according to specific goals. Finally, following recent calls for greater attention to safe machine learning (Giudici, 2024), future work could investigate the adoption of alternative model configurations specifically designed not only to enhance precision, but also to ensure robustness, transparency, and the integration of safety-oriented evaluation criteria.

References

[1] Aparício, David and Barata, Ricardo and Bravo, João and Ascensão, João Tiago and Bizarro, Pedro (2020). Arms: Automated rules management system for fraud detection. In arXiv. Retrieved from: https://arxiv.org/abs/2002.06075 (accessed 13th June 2025).
[2] Berretta S., Fusaro M., Giribone P. G., Muselli M., Tropiano F., Verda D. (2025) – "Enhancing the explainability of the default probability model using the Logic Learning Machine: a comparison between native "white boxes" Machine Learning techniques" – International Journal of Financial Engineering. Online Ready. https://doi.org/10.1142/S2424786325500057.
[3] Chen, Zhiyuan and Van Khoa, Le Dinh and Teoh, Ee Na and Nazir, Amril and Karuppiah, Ettikan Kandasamy and Lam, Kim Sim (2018). Machine Learning techniques for anti-money laundering (AML) solutions in suspicious transaction detection: a review. Knowledge and Information Systems, 57, pp. 245-285.
[4] European Commission (2023). Anti-money laundering and countering the financing of terrorism at EU level. In finance.ec.europa.eu. Retrieved from: https://finance.ec.europa.eu/financial-crime/anti-money-laundering-and-countering-financing-terrorism-eu-level_en (accessed 13th June 2025).
[5] European Union (2018). Directive (EU) 2018/843 of the European Parliament and of the Council. In eur-lex.europa.eu. Retrieved from: https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:32018L0843 (accessed 13th June 2025).
[6a] European Union (2024). Regulation (EU) 2024/1620 of the European Parliament and of the Council. In eur-lex.europa.eu. Retrieved from: https://eur-lex.europa.eu/eli/reg/2024/1620/oj/eng (accessed 13th June 2025).
[6b] European Union (2024). Regulation (EU) 2024/1624 of the European Parliament and of the Council. In eur-ex.europa.eu. Retrieved from: https://eur-lex.europa.eu/eli/reg/2024/1624/oj/eng (accessed 13th June 2025).
[6c] European Union (2024). Regulation (EU) 2024/1640 of the European Parliament and of the Council. In eur-lex.europa.eu. Retrieved from: https://eur-lex.europa.eu/eli/dir/2024/1640/oj/eng (accessed 13th June 2025).
[7] Financial Action Task Force (FATF) (2025). The FATF Recommendations. In fatf-gafi.org. Retrieved from: https://www.fatf-gafi.org/content/dam/fatf-gafi/recommendations/FATF%20Recommendations%202012.pdf (accessed 20th November 2025).
[8] Financial Action Task Force (FATF) (2013). Targeted financial sanctions related to terrorism and terrorist financing (Recommendation 6). In fatf-gafi.org. Retrieved from: https://www.fatf-gafi.org/content/dam/fatf-gafi/guidance/BPP-Fin-Sanctions-TF-R6.pdf.coredownload.pdf (accessed 13th June 2025).
[9] Gaggero G., Giribone P. G., Muselli M., Ünal E., Verda D. (2024) – "Portfolio optimization and risk management through Hierarchical Risk Parity and Logic Learning Machine: a case study applied to the Turkish stock market – Risk Management Magazine Vol. 19, N. 1.
[10] Giudici, P. (2024). Safe machine learning. Statistics, Vol. 58(3), 473-477. https://doi.org/10.1080/02331888.2024.2361481
[11] International Revenue Service (IRS) (2025). Bank Secrecy Act. In irs.gov. Retrieved from: https://www.irs.gov/businesses/small-businesses-self-employed/bank-secrecy-act (accessed 13th June 2025).
[12] Jullum, Martin and Løland, Anders and Huseby, Ragnar Bang and Ånonsen, Geir and Lorentzen, Johannes (2020). Detecting money laundering transactions with machine learning. Journal of Money Laundering Control, Vol. 23 (1), pp. 173-186.
[13] KYC-CHAIN (2020). An overview of the FATF Recommendations, 2020. In kyc-chain.com. Retrieved from: https://kyc-chain.com/an-overview-of-the-fatf-recommendations/ (accessed 13th June 2025).
[14] Lewis, Roger J. (2000). An Introduction to Classification and Regression Tree (CART) Analysis. Conference proceedings of the Annual Meeting of the Society for Academic Medicine in San Francisco, California.
[15] London Stock Exchange Group (LSEG) (2024). EU Anti-money laundering directives. In lseg.com. Retrieved from: https://www.lseg.com/en/risk-intelligence/financial-crime-risk-management/eu-anti-money-laundering-directive (accessed 13th June 2025).
[16] Milo, Tova and Novgorodov, Slava and Tan, Wang-Chiew (2016). Rudolf: interactive rule refinement system for fraud detection. Proceedings of the VLDB, Vol. 9(13), pp. 1465-1468.
[17] Muselli, Marco (2005). Switching Neural Networks: A New Connectionist Model for Classification. Neural Nets, pp. 23-30.

[18] Muselli, Marco and Ferrari, Enrico (2011). Coupling Logical Analysis of Data and Shadow Clustering for Partially Defined Positive Boolean Function Reconstruction. IEEE Transactions on Knowledge and Data Engineering,Vol. 23, pp. 37-50.

[19] Nweze, Michael and Avickson, Eli and Ekechukwu, Gerald (2024). The Role of AI and Machine Learning in Fraud Detection: Enhancing Risk Management in Corporate Finance. International Journal of Research Publication and Reviews, Vol. 5.

[20] Ohno-Machado, Lucila and Stephan, Dreiseitl (2002). Logistic regression and artificial neural network classification models: a methodology review. Journal of Biomedical Informatics, Vol. 35(5–6), pp. 352-359.

[21] Oztas, Berkan and Cetinkaya, Deniz and Adedoyin, Festus and Budka, Marcin (2022). Enhancing Transaction Monitoring Controls to Detect Money Laundering Using Machine Learning. 2022 IEEE International Conference on e-Business Engineering (ICEBE), pp. 26-28.

[22] Oztas, Berkan and Cetinkaya, Deniz and Adedoyin, Festus and Budka, Marcin and Dogan, Huseyin and Aksu, Gokhan (2023). Enhancing Anti-Money Laundering: Development of a Synthetic Transaction Monitoring Dataset. 2023 IEEE International Conference on e-Business Engineering (ICEBE), pp. 47-54. Dataset Retrieved from: https://www.kaggle.com/datasets/berkanoztas/synthetic-transaction-monitoring-dataset-aml (accessed: 20 November 2025).

[23] Teradata. (n.d.). Fraud Detection with Machine Learning. In teradata.de. Retrieved from: https://www.teradata.de/insights/ai-and-machine-learning/fraud-detection-machine-learning (accessed 13th June 2025).

[24] UIF (Unità di Informazione Finanziaria per l'Italia) istruzioni per la rilevazione di operazioni sospette (2025). Retrieved from: https://uif.bancaditalia.it/normativa/norm-antiricic/Istruzioni_UIF_rilevazione_e_segnalazione_operazioni_sospette.pdf (accessed: 29th July 2025).

[25] United Nations (2021). Learn about UNCAC. In unodc.org. Retrieved from: https://www.unodc.org/corruption/en/uncac/learn-about-uncac.html (accessed 13th June 2025).

[26] United Nations Office on Drugs and Crime (UNODC) (n.d.). Overview of Money Laundering. In unodc.org. Retrieved from: https://www.unodc.org/unodc/en/money-laundering/overview.html (accessed 13th June 2025).

[27] United Nations Office on Drugs and Crime (UNODC) (2000). United Nations Convention Against Corruption. In unodc.org. Retrieved from: https://www.unodc.org/documents/treaties/UNCAC/Publications/Convention/08-50026_E.pdf (accessed 13th June 2025).

Appendix A – Table of letters and symbols

$Y \in \{O_0, O_1, \dots, O_{q-1}\}$	*Explanatory categorical variable with q classes (q= 2 is a bi-class problem)*
q	*Number of classes of Y*
$O_0, O_1, \dots, O_{q-1}$	*Labels of the classes of Y*
$X = X_1, \dots, X_j, \dots, X_d$	*Features*
d	*Number of features*
$y_i\ , i = 1, \dots, n$	*Output of sample i*
$x_{ij} \quad , i = 1, \dots, N_{row}\ \ , j = 1, \dots, d$	*Dataset dimension [n, d]*
$S = \{(x_i, y_i)\}_{i=1}^{n}$	*Training set*
$g(x)$	*Model*
ψ_j	*Mapping from continuous domain to discrete domain of j-th variable*
M	*Number of discretization intervals*
$I_M = 1, \dots, M$	Set of positive integers up to M
$\boldsymbol{\gamma}_j = (\gamma_{j1}, \dots, \gamma_{jm}, \dots, \gamma_{jM_j-1})$	$(\mathrm{M_j} - 1)$ *cutoffs of variable* $\mathrm{X_j}$
M_j	*Number of discretization intervals of variable* X_j
$\boldsymbol{\rho}_j = \left(p_{jl}, \qquad \forall\, l = 1, \dots, \alpha_j\right)$	the vector of all the α_j values for input variable j in ascending order
α_j	*Number of distinct values of* X_j
φ_j	Mapping for transformation of discretized domain into a binary domain
u, v	$u < v$ *if and only if* $\varphi_j(u) < \varphi_j(v)$
$\boldsymbol{z}, \boldsymbol{w} \in \{0,1\}^{M_j}$	Elements of a string having a bit for each possible value in I_{M_j}
$\boldsymbol{z}_i \in \{0,1\}^B\ ;\ \varphi_j(\boldsymbol{x}_i) = \boldsymbol{z}_i$	*Binarization of x*
$\boldsymbol{z}_i$	Obtained by concatenating $\varphi_j(\mathbf{x}_i)$ for $j = 1, \dots, d$.
$B = \sum_{j=1}^{d} M_j$	Sum of the number of discretization intervals for $j = 1, \dots, d$.
$S' = \{(\boldsymbol{z}_i, y_i)\}_{i=1}^{N}$	*Binarized Training Set*
N	*Number of rows in training set*
$\boldsymbol{h} \in \{0,1\}^{M_j}$	Having all bits equal to 1 except the k-th bit which is set to 0
f	Boolean function
T	T is the set containing $(\mathbf{z}_i, y_i)$ with $y_i = 1$
F	F is the set containing $(\mathbf{z}_i, y_i)$ with $y_i = 0$
η	Number of terms in definitions of AND OR
$\mathbf{a} \in \{0,1\}^B$	Vector of zeros or ones of length “B”
$A \subset I_B$	Antichain
A^*	Simplified A, $A^* \subset A$
$P(\boldsymbol{a})$	The subset I_B containing each i such that $a_i = 1$
$U(\boldsymbol{a})$	Upper shadow of **a,**
$L(\boldsymbol{a})$	Lower shadow of **a,**
$\boldsymbol{h}_j$	**z** was obtained by concatenating the results of the mapping $\varphi_j(\mathbf{x})$ and it can be split into substring $\mathbf{h}_j$ for each attribute
V	The set of values associated with each z_i
r	*Rule*
$C(r)$	*Covering of rule r*
$E(r)$	*Error of rule r*
$TP(r), FP(r),\ TN(r),\ FN(r)$	True positive, false positive, True negative, and false negative for rule **r**
ε_{max}	Regularization parameter
$R(r)$	Relevance
$\mathcal{R}$	Complete ruleset
$\mathcal{R}_o = \{r \mid r \in \mathcal{R}, O(r) = o$	Set of rules that predict class o
$\mathcal{R}_o^i = \{r \mid r \in \mathcal{R}, r \le \boldsymbol{x}_i, O(r) = o\}$	Set of rules satisfied by $\mathbf{x}_i$ that predict class o

$S(\boldsymbol{x}_i, o)$	Score for each class o that measures how likely it is that $y_i = o$
$P(o \mid \boldsymbol{x}_i)$	Probability that $y_i = o$ given x_i
$\tilde{y}_i$	The selected output is the one that maximizes the output probability
c	Condition
r'	Obtained by removing condition c from r
$J_{j1}, J_{j2}, \ldots, J_{M_j}$	Intervals $J_{j1} = [-\infty,\gamma_{j1}]$, $J_{j2} = (\gamma_{j1},\gamma_{j2}]$, etc.
$G_j = \{v_{j1}, v_{j2}, \ldots\}$	The collection of the possible values assumed by x_j

Appendix B – SAML-D dataset

B.1) Core features

Time: the precise time of each individual transaction. It is standardized to "UCT+00:00" convention;

Date: it gives information about the transaction date. This, in addition to the **Time** feature, is an essential feature for tracking transaction chronology;

Sender_account: it contains the information about the sending account ID;

Receiver_account: it contains the information about the receiving account ID. In addition to the **Sender_account** feature, it helps uncover behavioural patterns and complex banking connections;

Amount: it indicates transaction values to identify suspicious activities. This value is already standardized to £ (UK pounds);

Payment_currency: it gives information about the currency used to make the payment;

Received_currency: it gives information about the currency received as payment. Both this and **Payment_currency** generally align with the location feature of the account, meaning that they conform with the prevalent currency of that specific country, but several mismatched instances are also present to add complexity;

Sender_bank_location: it contains information relating to the country from which money is sent;

Receiver_bank_location: it contains the information relating to the country where the money is being received. Together with **Sender_bank_location** it helps pinpointing high-risk regions for AML such as Mexico, Morocco and the UAE. The same account may carry out transactions from different countries, meaning that this information is not static across different instances;

Payment_type: it specifies the typology of settlement carried out by the sender account, each involving different levels of risk. It includes various methods like credit card, debit card, cash, ACH transfers, cross-border, and cheque;

Is_laundering: this feature is a binary indicator differentiating "normal" from "suspicious" transactions;

Laundering_type: it further describes the typology of the transaction, classifying both "normal" and "suspicious" instances. It offers deeper insights into prevalent or high-risk typologies of transactions.

B.2) Payment typologies

Credit card;

Debit card;

Cash withdrawal;

Cash deposit;

Automated clearing house (ACH) transfers;

Cross-border;

Cheque.

B.3) Laundering typologies

Cash withdrawal: it involves withdrawing illicit funds in cash from a financial institution. It is used to move money out of the formal financial system and into the physical world, making it more difficult to trace. Cash withdrawals are often part of larger schemes, such as layering or integration in the money laundering process;

Behavioural change 1: the behavioural change 1 and 2 typologies adopt the same structure as the normal group typology. However, in behavioural change 1, the main account deviates from its usual patterns and transacts with new accounts;

Behavioural change 2: in contrast, under the Behavioural Change 2 typology, the main account transacts with new accounts in high-risk locations;

Structuring: it involves breaking large amounts of money into smaller transactions that fall below reporting thresholds;

Smurfing: this is a particular form of structuring where multiple individuals (called "smurfs") are employed to make numerous small deposits or transactions that are below the reporting thresholds;

Fan out: when a single source of illicit funds is distributed to multiple accounts or entities;

Fan in: when multiple smaller amounts from different accounts or entities are funneled into a single account;

Layered fan in: it is a multi-layer scheme which involves multiple sender accounts transacting with a fewer number of receiver accounts, which ultimately transact with a single receiver account in a sort of funnel-shaped pattern;

Layered fan out: this version mirrors the previous structure but with transactions flowing in the reverse direction;

Scatter-gather: when funds are first scattered across multiple accounts or entities (scatter phase) and then later reassembled into one or more accounts (gather phase);

Gather-scatter: it works in the exact opposite way;

Cycle: when funds are moved in a circular manner through a series of transactions that ultimately return the funds to the original account or entity;

Bipartite: when funds are transferred back and forth between two distinct groups of accounts in a manner that disguises the money's origin and destination;

Stacked bipartite: in its "stacked" version, multiple layers or tiers of entities are involved;

Over-invoicing: this technique exploits commercial transactions, e.g. transfer prices, to overestimate the value of goods and services provided to foreign affiliated partners in order to shift the taxable income to high-tax or low-tax jurisdictions;

Deposit-send: this typology refers to a situation where an account first deposits cash into the bank and then within a short period of time sends it to another account. The transaction amount is generally below the reporting threshold limit, with the second transaction having an increased chance of being sent to a high-risk country. It is considered suspicious due to the rapid movement of funds and potentially facilitating terrorism finance.

Single large transaction: as the name suggests, this type of payment involves sending a large sum of money in a single installment. This type of payment appears to be suspicious, especially in cases where a customer typically makes modest transactions and suddenly completes a single large transaction, which can be a sign of suspicious activity, particularly if there is no plausible economic justification.