UNIVERSITY OF TARTU
Faculty of Science and Technology
Institute of Computer Science
Data Science Curriculum

Annabel Hiiu, Anti Karumaa

# Machine Learning Based Risk Scoring for Money Mule Detection

Master's Exam (15 ECTS)

Supervisors: Kristo Raun, PhD
Anna Martignano, MSc
Alexander Jöhnemark, MSc

Tartu 2025

## Machine Learning Based Risk Scoring for Money Mule Detection

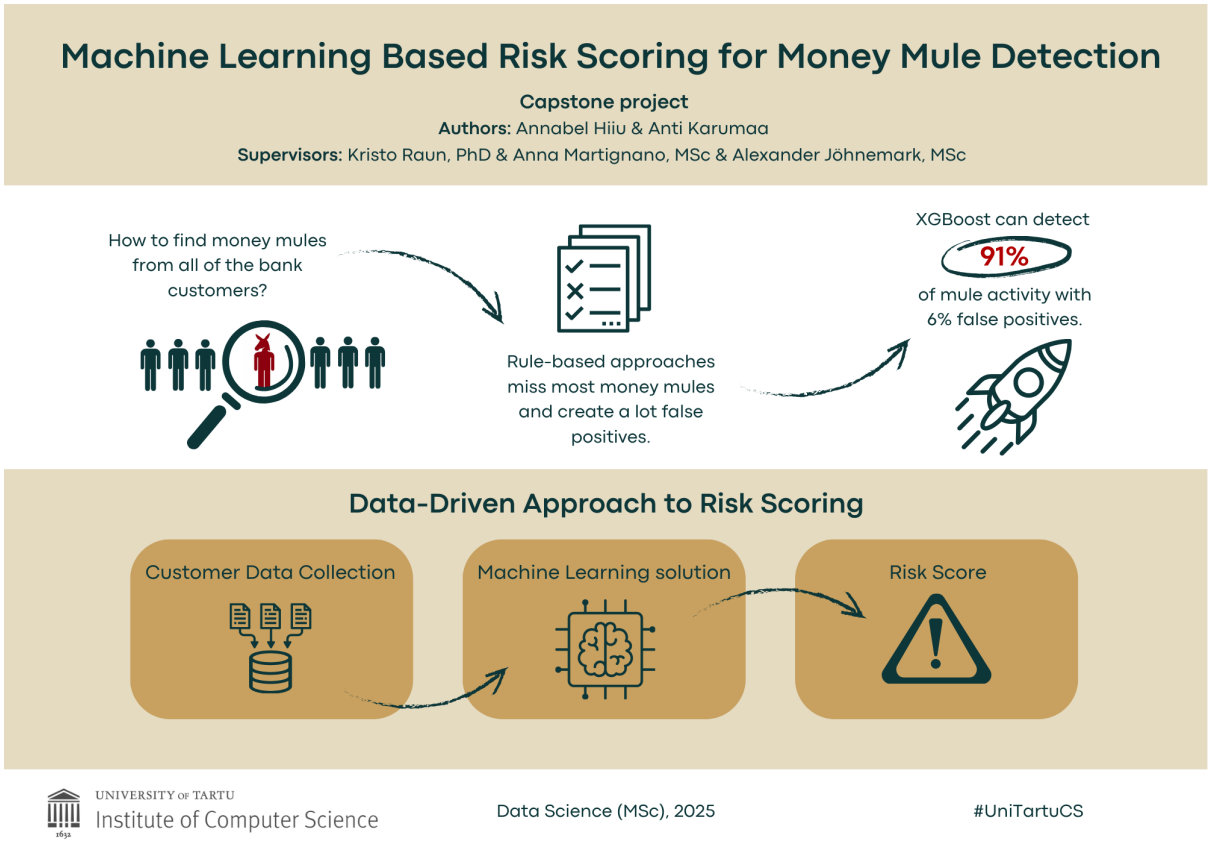
**Abstract:**

Money laundering is a serious problem for the financial systems, as it helps criminals hide illegal funds and damages trust in banks. Money mules are individuals who transfer money on behalf of others, often unknowingly, and play a central role in these activities. Detecting money mules is a major challenge for financial institutions. Traditional detection methods rely on fixed rules that are not flexible enough to keep up with changing criminal tactics. This thesis explores the application of machine learning models for detecting money mules by assigning client-level risk scores based on transaction habits, product usage, and personal information. Multiple models were evaluated, with XGBoost demonstrating the highest area under the precision–recall curve (AUPRC) of 0.1314, indicating strong performance in this highly imbalanced setting. The model shows potential to detect over half of known money mules while maintaining a manageable false positive rate. These findings suggest that integrating machine learning into anti-money laundering systems can improve anti-money laundering efforts, helping banks prevent criminal activity, protect their customers and maintain their reputation.

**Keywords:**
Money Mules, Machine Learning, Risk Scoring, Banking

**CERCS:** P176 - Artificial intelligence, P160 - Statistics, operation research, programming, actuarial mathematics

## Visual Abstract:



## Masinõppel põhinev riskiskoorimine rahamuulade tuvastamiseks

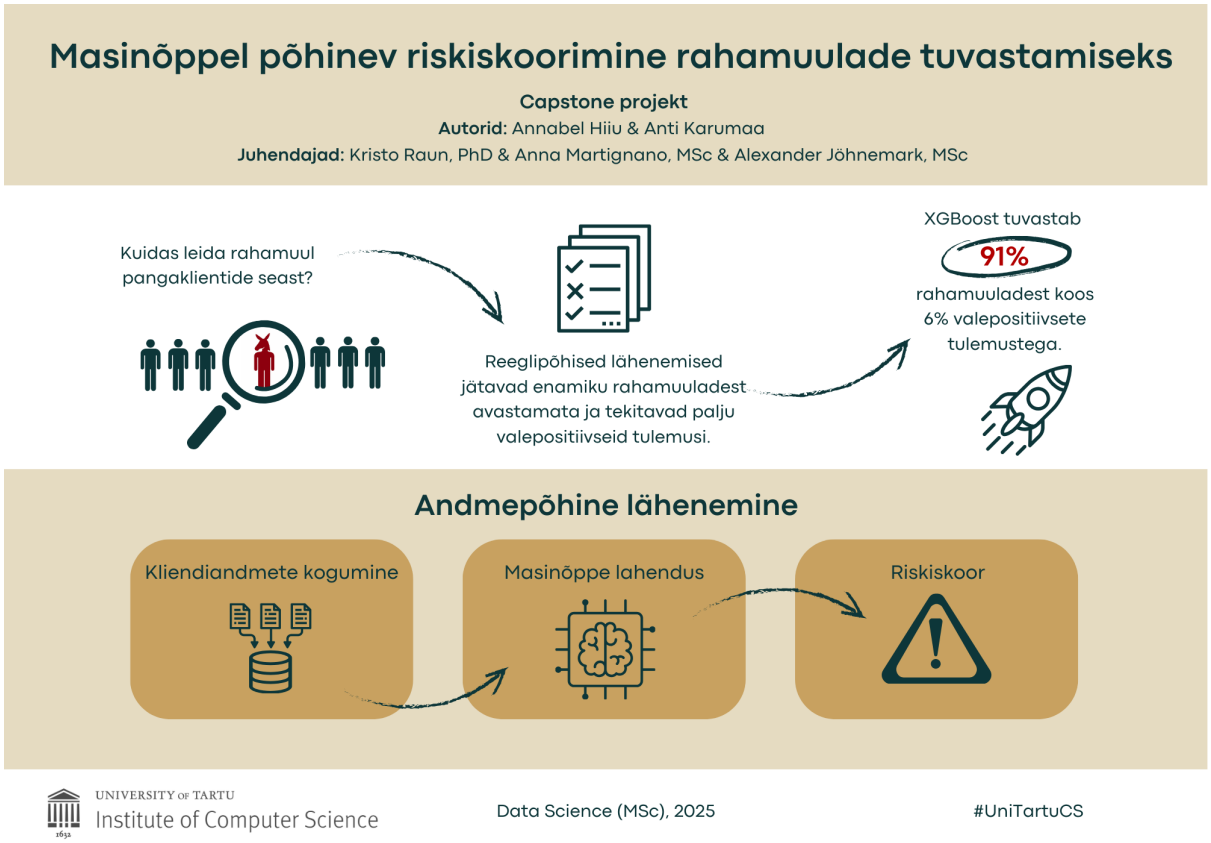
**Lühikokkuvõte:**

Rahapesu on tõsine probleem finantsüsteemides, kuna see aitab kurjategijatel varjata ebaseaduslikke summasid ja kahjustab pankade usaldusväärsust. Rahamuulad on isikud, kes tihti eneselegi teadmata kannavad raha teiste eest ja mängivad rahapesus keskset rolli. Rahamuulade tuvastamine on finantsasutustele suur väljakutse. Traditsioonilised tuvastamismeetodid tuginevad kindlatele reeglitele, mis ei ole piisavalt paindlikud, et kohaneda muutuvate kuritegelike taktikatega. Käesolev magistritöö uurib masinõppe mudelite rakendamist rahamuulade tuvastamiseks, määrates klientidele riskiskoori nende tehinguharjumuste, toodete kasutamise ja isikuandmete põhjal. Hinnati mitmeid mudeleid, millest parima tulemuse andis XGBoost, saavutades täpsuse-saagis kõvera aluse pindala (AUPRC) väärtuseks 0,1314. See on hea tulemus arvestades tasakaalustamata andmestikus. Mudel tuvastas üle poole teadaolevatest rahamuuladest, hoides samal ajal valepositiivsete määrad mõistlikul tasemel. Töö tulemused viitavad, et masinõppe integreerimine rahapesuvastastesse süsteemidesse võib aitada kaasa kuritegude ennetamisele, aidata pankadel kaitsta oma kliente ning säilitada usaldusväärsust.

**Võtmesõnad:**
Rahamuulad, pangandus, riskihindamine

**CERCS:** P176 - Tehisintellekt, P160 Statistika, operatsioonanalüüs, programmeerimine, finants- ja kindlustusmatemaatika

**Visuaalne kokkuvõte:**



## Acknowledgments

We would like to thank our supervisors for their guidance, support, and valuable feedback throughout the development of this thesis. Kristo Raun, our supervisor from the university side, was extremely patient and helpful, always listening and available to us whenever needed. Our weekly meetings in Kärg C at the Delta building were unforgettable, and we will miss those Monday afternoon discussions.

From Swedbank, sincere thanks are extended to Anna Martignano and Alexander Jöhnemark for their technical expertise, whether configuring the compute cluster or advising on machine learning questions, and for making complex banking systems accessible. We would also like to thank the wider Swedbank team, especially stakeholders, for sharing industry knowledge and business logic, which was essential for this thesis to be a solid academical work.

# Contents

# 1. Acronyms and Abbreviations

Here is a full list of all the acronyms and abbreviations used in the thesis.

**AI** Artificial Intelligence

**AML** Anti-Money Laundering

**AMLA** Anti-Money Laundering Authority

**AMLDs** Anti-Money Laundering Directives

**AUROC** Area Under the Receiver Operating Characteristic Curve

**AUPRC** Area Under Precision–Recall Curve

**BSA** Bank Secrecy Act

**CDD** Customer Due Diligence

**CatBoost** Categorical Boosting

**CRISP-DM** Cross-Industry Standard Process for Data Mining

**DBSCAN** Density-Based Spatial Clustering of Applications with Noise

**DT** Decision Tree

**EDD** Enhanced Due Diligence

**EU** European Union

**FATF** Financial Action Task Force

**FBI** The Federal Bureau of Investigation

**FN** False Negative

**FP** False Positive

**GDPR** General Data Protection Regulation

**GDP** Gross Domestic Product

**GBT** Gradient Boosting Tree

**IF** Isolation Forest

**IC3** Internet Crime Complaint Center

| | |
|---|---|
| **KNN** | K-Nearest Neighbours |
| **KYC** | Know Your Customer |
| **KYB** | Know Your Business |
| **KYCC** | Know Your Customer's Customer |
| **LightGBM** | Light Gradient Boosting Machine |
| **LR** | Logistic Regression |
| **LOF** | Local Outlier Factor |
| **MCC** | Matthews Correlation Coefficient |
| **MDP** | Markov Decision Process |
| **ML** | Machine Learning |
| **OCR** | Optical Character Recognition |
| **PCA** | Principal Component Analysis |
| **PEP** | Politically Exposed Person |
| **RF** | Random Forest |
| **SAR** | Suspicious Activity Report |
| **SHAP** | SHapley Additive exPlanations |
| **SMOTE** | Synthetic Minority Over-Sampling Technique |
| **SVM** | Support Vector Machines |
| **TN** | True Negative |
| **TP** | True Positive |
| **XGBoost** | eXtreme Gradient Boosting |

# 2. Introduction

Money laundering is one of the most prevalent financial crimes worldwide, erasing an estimated 2-5% of global Gross Domestic Product (GDP) each year. In 2023, the losses were between 2-5 trillion EUR [1]. Money mules have a important part in money laundering. They are individuals who facilitate the movement of illicitly obtained funds through legitimate financial systems. These individuals, either aware of engaging in illicit activities or not, provide a crucial link in obscuring the origins of illegal money, allowing criminals to evade detection. For financial institutions, detecting and preventing money mule activity is not only a regulatory obligation [2], but also a pressing necessity to safeguard global economic integrity.

Traditional rule-based systems used by banks to monitor suspicious activity often struggle to adapt to increasing criminal activity and their constantly changing approaches. These methods generate high false positive rates and lack the flexibility to respond to evolving tactics. However, the vast amounts of data collected by banks offer a unique opportunity for advanced solutions. Machine learning has emerged as a promising tool, capable of uncovering subtle patterns in data to predict risk more effectively and proactively.

The goal of this thesis is to provide stakeholders with a trained machine learning model as a scoring system that evaluates all customers on a scale from 0 to 100. The score will indicate the likelihood of a customer being a money mule. Multiple machine learning models were evaluated to identify those with the most promising performance for this task. By analysing past patterns, this system provides financial institutions with actionable insights to prevent crimes before they occur, shifting the focus from reactive detection to proactive prevention.

The chapter 3 (Background) provides an overview of what money mules are, and how they are currently detected. The chapter 4 (Overview of Machine Learning Methods) presents potential machine learning methods for money mule detection. The chapter 5 (Methodology) outlines the approach used in this thesis, including data understanding, data preparation, modelling, and evaluation methods. The chapter 6 (Results) presents the outcomes of the model experiments. The chapter 7 (Discussion) interprets these results and reflects on their practical implications, strengths, and limitations. To improve the overall clarity and readability of the thesis, ChatGPT-4o was used as a language assistant to rephrase selected sections and ensure consistency in tone and expression.

# 3. Background

Estimates suggest that almost two trillion US dollars are laundered each year, equivalent to approximately 5% of global GDP [3]. Although exact numbers are hard to find, it is clear that illegal money flow is significant, taking away resources from legitimate businesses and causing serious social and economic problems [4].

Money laundering is the process of making criminally obtained money legitimate and is typically carried out in three stages: placement, layering, and integration [5]. In placement, illegal funds are introduced into the financial system, often in smaller amounts or through cash-intensive businesses. Layering offers a series of complex transactions or transfers to obscure the origin of the funds. Finally, during integration, illicit money is reintroduced into the economy, typically disguised as legal profits or assets.

Given the complexity and scale of these operations, this thesis focuses specifically on the role of money mules who facilitate the movement of illicit funds. The following chapters will analyse how money mules operate, review conventional detection techniques employed by financial institutions, and evaluate the potential of machine learning applications to improve identification and prevention strategies.

## 3.1 Understanding Money Mules

Effective detection of money mules begins with a comprehensive understanding of their behaviours and characteristics. In addition to the confidential information collected by the authors of this thesis through discussions with Swedbank stakeholders, published research has provided valuable insight into common characteristics, behavioural patterns, and typologies associated with money mules. This section reviews the existing literature to provide a better understanding of how money mules operate and how they can be identified.

The Federal Bureau of Investigation (FBI) defines a money mule as an individual who transfers or moves illegally acquired money on behalf of others, often receiving a fee or commission for this service [6]. Criminals use money mules to make money laundering more complex and hide the money trail [7].

Stolen money is first transferred to the accounts of money mules instead of going directly to the criminals, and mules then quickly withdraw or transfer the money [7]. In some cases, the money passes through multiple mules before reaching its final destination. Typically, these mules have

no direct connection to the criminal network, making it harder to trace the criminals behind the operation.

Most money mule operations involve incremental transfers from personal accounts to recipients in jurisdictions with looser anti-money laundering (AML) controls (e.g. parts of Eastern Europe or West Africa) [8]. In this staging phase of money laundering, money mules act as intermediaries who may often be recent immigrants, pensioners, or unemployed persons, and receive illicit payments to redistribute them, thereby camouflaging the money trail. The people who manage these funds usually do not meet the actual beneficiaries. However, their accounts play a crucial role in a complex system of transactions meant to hide the illegal source of the money.

The role of money mules in the laundering process is illustrated in Figure 1.

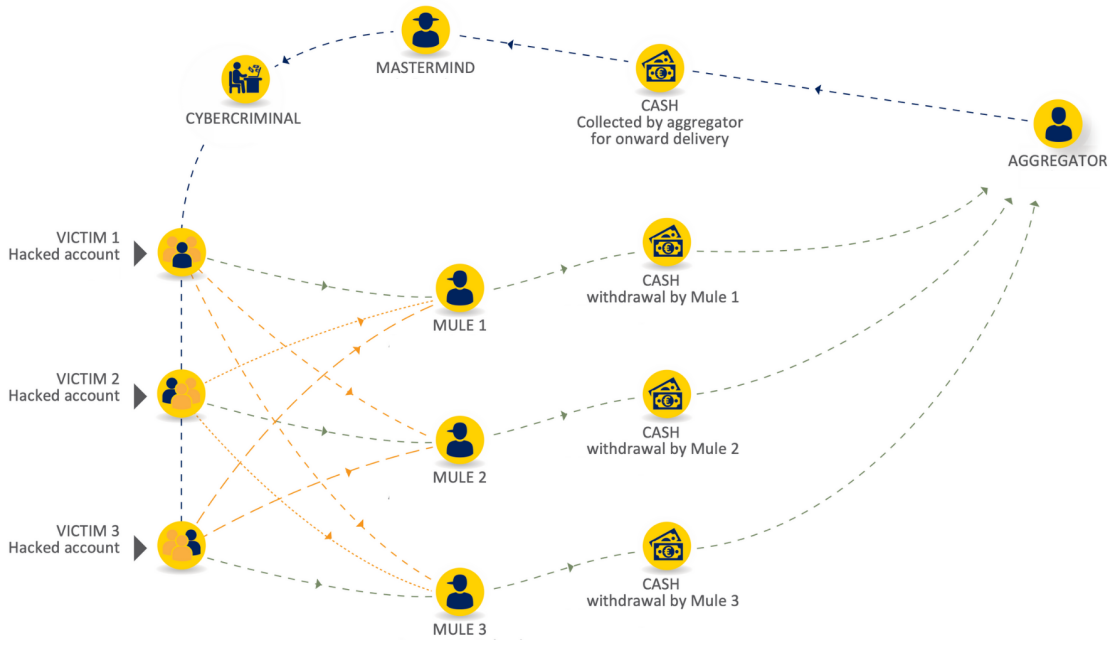



Figure 1. Cycle of money laundering using money mules, adapted from Europol [9]. Country names have been removed.

Some money mules knowingly assist criminals in laundering money to earn easy money for themselves, while others do not know that their actions are illegal [8]. Criminals often exploit financially vulnerable people, including students, unemployed individuals, newcomers to a country, and those facing economic hardship [10]. Many money mules struggle financially, making them easy targets for fraudulent job offers that promise quick and effortless earnings [7].

Recruiters take advantage of financially vulnerable individuals by offering financial rewards in exchange for the use of their bank accounts. Based on Dutch criminal investigations, one

way money mules are recruited is through personal connections, such as friends, classmates, or teammates [7]. Recruiters approach them casually, asking about their financial situation, or directly offering an opportunity to earn easy money with little effort.

Another way to recruit money mules is through online platforms where recruiters post ads on job websites or social media, targeting young users who are more likely to be active on these platforms [7]. Bekkers, Moneva, and Leukfeldt conducted a quasi-experimental study to examine the engagement of young users with money mule recruitment ads on Instagram [11]. They ran three different ads reflecting different recruitment mechanisms on Instagram feeds and stories. The study found that up to 3% of young users interacted with the ads. The higher engagement rates were for messages that promoted “earn fast and lots of money” or “earn money legally”. They noticed that men were more likely to engage with ads and also that engagement rates were higher at night.

Figure 2 illustrates one such fraudulent ad (in Estonian: "Saa 225€ meie pangast!" / "Get 225€ from our bank!" and "Ainult Swedi klientidele" / "Only for Swed clients"). Although this example is a classic fraud attempt with unclear objectives, it demonstrates how criminals might attract unsuspecting individuals to become money mules by offering seemingly legitimate "gifts". Targeting low- or medium-risk customers makes these transactions look routine to financial institutions which reduces the likelihood of detection.

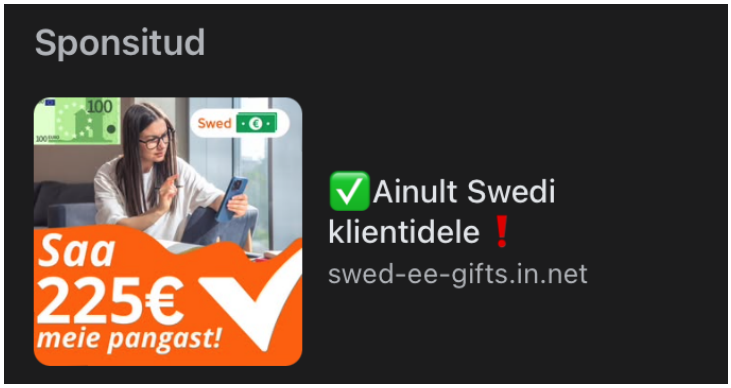



Figure 2. Money mule recruitment ad in Facebook.

According to a 2023 article published by EUROPOL, criminals primarily target individuals under 35 years old, with a growing focus on younger generations between 12 and 21 years old [10]. Younger people are often in need of money and less aware of the risks associated with money muling. However, money muling is not limited to young individuals.

A 2021 article by the UK's fraud prevention service, CIFAS, highlights that people aged 40 to 60 are increasingly involved in money muling [12]. Criminals exploit this age group because their larger transactions are less likely to raise suspicion.

Furthermore, the FBI's Internet Crime Complaint Center (IC3) reported in 2019 that elderly women, particularly widows, are often targeted by romance or confidence fraud [13]. Criminals build trust with their victims and then manipulate them into unknowingly laundering illegally obtained money.

The profile of money mules has become increasingly diverse, including individuals of different ages and genders. The evolving demographic and behavioural diversity of money mules makes it challenging for financial institutions to detect them. The next section discusses traditional methods currently utilized by banks to combat money muling activities.

## 3.2 Traditional Methods of Detecting Money Mules

Money mule schemes are constantly evolving and financial institutions must adapt their traditional AML strategies to keep pace. Financial institutions have long relied on compliance controls and rule-based transaction monitoring to detect money mule activity. Although these methods can catch well-known laundering patterns, they struggle with high volumes of false positives and cannot keep up with new, more subtle mule tactics [14].

This section outlines how regulatory and compliance frameworks underpin customer checks, then examines how rule-based systems flag suspicious flows, and finally shows why these static approaches leave gaps that machine learning methods are poised to fill.

### 3.2.1 Regulatory and Compliance Controls

A fundamental element in the fight against money mule activity is strict adherence to AML regulations, particularly those related to customer verification and monitoring. Financial institutions must comply with international and regional directives, including the Anti-Money Laundering Directives (AMLDs) in the European Union (EU) [2], the Bank Secrecy Act (BSA) in the United States [15], and global guidelines from the Financial Action Task Force (FATF) [16]. In the EU, a new centralized supervisory body, the Anti-Money Laundering Authority (AMLA), was established by Regulation (EU) 2021/557 and is expected to become operational in mid-2025 to ensure the consistent application of the EU AML rules across member states [17].

The key part of these AML rules is **Customer Due Diligence (CDD)**, starting with **Know Your Customer (KYC)** protocols. KYC requires institutions to verify the identity of a customer and

assess their risk profile. This helps banks establish a baseline of expected customer behaviour and detect anomalies that could indicate misuse of services. Initially imposed on banks and other traditional financial institutions, KYC obligations have expanded to fintech firms, virtual asset service providers, and even non-profit organisations in many jurisdictions [18, 19].

During KYC and CDD, institutions collect a variety of information to build a customer risk profile [20]. This typically includes personal details (name, date of birth, address, phone number), scans of identity documents and proof of address, digital footprint data (IP address, device fingerprint, email domain), professional and financial background (occupation, employer, source of funds) and risk indicators such as PEP (politically exposed person) status or sanctions-list hits. Automated systems use these data to run immediately monitoring and sanction checks, and any high- or medium-risk result triggers Enhanced Due Diligence (EDD).

An empirical study by Bashir et al. demonstrated a reduction in money laundering risk within the United Arab Emirates banking sector between 2018 and 2019, attributing this improvement to enhanced CDD, the deployment of advanced IT compliance tools and stricter adherence to FATF standards [21]. Their findings underscore the value of risk-based frameworks and data-driven strategies in strengthening traditional AML controls.

**Know Your Business (KYB)** and **Know Your Customer's Customer (KYCC)** extend due diligence beyond the primary account holder. KYB requires the verification of the legal status and ownership structure of the corporate clients, while KYCC involves understanding and assessing the risks posed by the clients of those corporate clients [16]. These measures are particularly valuable for uncovering multilayered mule networks that seek to obscure the ultimate origin and destination of illicit funds.

Despite these controls, KYC and CDD provide only a static snapshot on onboarding and cannot catch mules who behave normally at first. Fraudsters using fake or synthetic documents can slip past basic OCR (optimal character recognition) or visual inspections. Moreover, some non-bank channels, such as certain crypto exchanges or independent money-transmitters, apply lighter or no KYC, creating gaps that mule networks can exploit.

### 3.2.2 Rule-Based Detection Systems

Many banks utilise rule-based detection systems to identify suspicious transactions indicative of money laundering or money mule activities. These systems rely on predefined rules or thresholds

established by AML compliance experts, which flag transactions that match known suspicious patterns [22].

Typical scenarios that trigger alerts include several small incoming transfers followed by a single large outgoing transfer, transfers between accounts in high-risk countries, sudden activity from previously dormant accounts, or rapid cash withdrawals following unexplained cash deposits [23]. Rule-based systems can also integrate static customer characteristics to identify vulnerable individuals commonly targeted for mule recruitment, such as unemployed persons, students, recent immigrants, or financially distressed clients [22].

One key advantage of rule-based systems is that they allow continuous customer due diligence throughout the relationship, not just at onboarding [24]. They apply a set of predefined rules on an ongoing basis, so any unusual activity can be picked up after the initial checks. When these basic rules prove insufficient, the system can switch on automatic, second-level detection to catch more complex patterns. This layered approach helps firms keep track of risks as they evolve.

However, rule-based systems also exhibit significant drawbacks. Reliant on predefined logic and known patterns, they are rigid and susceptible to missing novel or evolving money mule behaviours [25]. Consequently, banks must regularly calibrate and update these rules to maintain their effectiveness in the face of emerging threats.

Rule-based AML systems generate massive alert volumes. Over 95% of system-generated alerts are closed as false positives in the first phase of review, costing banks billions of dollars in investigation time [26]. Regulators are increasingly encouraging the use of artificial intelligence and machine learning to triage and prioritise alerts, thereby reducing false positives and improving compliance efficiency.

### 3.2.3 Limitations of Traditional Detection Approaches

Traditional AML systems are mainly based on static rule-based logic, which restricts their ability to adapt to evolving money mule tactics. Hard-coded rules require manual updates for each new fraud pattern, a labour-intensive and error-prone process. This rigidity produces high false positive rates, overwhelming investigators with low-value alerts and driving up operational costs [26]. Moreover, applying generic rules uniformly often flags legitimate but unusual customer behaviour as suspicious, harming customer trust and risking reputational damage [27].

These limitations are caused by reliance on complete and accurate data. Inconsistent customer profiles, missing transaction details and siloed databases compromise rule accuracy and hinder detection of subtle patterns [28]. While enriching data with external sources such as sanctions lists, device fingerprints and geolocation can help, integrating and managing these feeds adds complexity and potential inconsistencies [29]. Maintaining detection performance also requires systematic feature engineering, automated versioning and drift monitoring, which extend beyond traditional rule maintenance [30].

Adaptive methods face additional hurdles from regulations. The General Data Protection Regulation (GDPR) enforces strict data minimisation, transparency and restrictions on automated profiling, complicating extensive behavioural analysis [31]. Likewise, the EU's AML Directives and FATF recommendations uphold strong CDD requirements but do not explicitly support data-driven adaptive controls [2, 16]. This regulatory framework slows innovation and encourages reliance on non-transparent rule sets that can raise fairness and discrimination concerns when legitimate customers are subjected to intrusive investigations.

# 4. Overview of Machine Learning Methods

Machine learning (ML) is a subfield of artificial intelligence (AI) which designs algorithms that improve their performance on a task as they are exposed to more data, without explicit reprogramming [32, 33]. Unlike rule-based systems, which operate according to predefined logical conditions and require manual updates to address novel patterns, ML models can uncover complex, non-obvious relationships and adapt as additional data become available [34].

In financial institutions, traditional rule-based approaches often generate a high volume of false positives, demand continual maintenance and struggle to accommodate emerging fraud tactics. In contrast, ML models offer greater flexibility and scalability, learning to generalise from historical transaction records to detect evolving threats. Nevertheless, ML comes with challenges such as algorithmic transparency, potential biases in training data and the need for large, representative datasets along with careful hyperparameter tuning [33].

There is a lot of research on supervised, and unsupervised learning for AML, but studies using reinforcement learning in this area are very few. To keep the focus on the most established and directly relevant methods, reinforcement learning is not covered.

## 4.1 Supervised Learning Models

Supervised learning is a class of ML techniques in which models are trained on labelled datasets comprising input–output pairs. Formally, given a training set

$$\{(x_i, y_i)\}_{i=1}^{N},$$

where each $x_i \in \mathcal{X}$ is a feature vector and $y_i \in \mathcal{Y}$ the corresponding ground-truth label, the goal is to learn a mapping $f : \mathcal{X} \to \mathcal{Y}$ that generalises well to unseen inputs [32, 33].

What distinguishes supervised learning from other paradigms is the availability of explicit supervisory signals during training. Models directly minimise a loss function $L(y, \hat{y})$ over labelled examples by empirical risk minimisation, in contrast to unsupervised methods that seek structure without labels or reinforcement learning which relies on delayed scalar rewards [33].

Advantages of supervised learning include high predictive accuracy when sufficient, representative labels are available, clear validation via hold-out or cross-validation schemes, and interpretability, which is an essential feature in regulated domains such as AML [32]. However, reliance on labelled data can be costly and time-consuming to acquire, models risk overfitting in

high-dimensional spaces with limited samples, and there is a risk of reinforcing biases present in historical labels [33].

A wide range of supervised models has been applied to AML problems. These can be grouped based on their learning strategy and complexity: from simple, interpretable linear classifiers to more advanced tree-based and ensemble methods. The following subsections present each class of models with their respective trade-offs and practical performance in AML scenarios.

### 4.1.1 Linear and Simpler Models

Linear and simpler models are often used as baselines in machine learning due to their interpretability and ease of implementation. While they may lack the capacity to capture complex patterns, their transparency and efficiency make them valuable in AML settings, especially where explainability is required.

**Logistic regression (LR)** is a supervised ML method for binary classification that models the log-odds of an outcome as a linear combination of predictor variables and outputs a probability between 0 and 1, with a default decision threshold of 0.5 to separate classes [35]. Its transparency allows to interpret feature coefficients directly, offering clear insights into how each variable influences the likelihood of fraudulent behaviour. In AML applications, logistic regression provides an understandable risk score for each customer, such as the probability of becoming a money mule, simplifying compliance reporting.

LR has been widely applied as a baseline model for credit card fraud detection due to its simplicity. Elhusseny, Ouf, and Idrees reported an MCC score of 0.761 for LR, highlighting its effectiveness after resampling techniques were applied [36]. Similarly, Itoo, Meenakshi, and Singh found that logistic regression outperformed both Naïve Bayes and K-Nearest Neighbour, achieving up to 95.9% accuracy and strong results across all evaluation metrics [37]. They attributed this performance to LR's ability to model correlated features and its reliance on the conditional data likelihood function. Bhattacharyya et al. also used logistic regression as a benchmark and observed that, despite its simplicity, LR often outperformed more complex models such as SVM, especially when dealing with varying degrees of data imbalance [38].

**K-nearest neighbours (KNN)** assigns each new instance to the class most common among its $K$ closest neighbours in feature space, using distance metrics like Euclidean or cosine distance to measure closeness [35]. This simple, non-parametric method serves as a baseline for identifying patterns in data, often outperforming random predictions. KNN's interpretability stems from

how the neighbourhood composition influences decisions. However, its accuracy can decline in high-dimensional spaces due to the curse of dimensionality, and as the dataset size increases, prediction latency grows linearly. Therefore, choosing the right $K$ and distance measures is essential to balance bias and variance.

KNN algorithm has shown mixed results across studies. Elhusseny et al. found that KNN achieved a high MCC score of 0.793, second only to random forest, and outperformed both LR and SVM in their experiments [36]. The model's ability to classify transactions based on the majority class of nearby instances proved effective on their well-prepared dataset. In contrast, Itoo et al. reported that KNN struggled to distinguish between similar fraud and non-fraud patterns, achieving only 75% accuracy [37]. Despite this lower performance, KNN is a simple and valuable baseline model worth exploring further.

**Support vector machines (SVM)** are ML models used for classification and regression tasks [35]. Their primary objective is to create the best decision boundary, known as a hyperplane, that separates different classes of data points by maximising the margin between the boundary and the closest points, called support vectors. These support vectors influence the position and orientation of the boundary. SVMs excel in high-dimensional spaces and can address situations where data is not linearly separable by using kernel functions to project the data into higher dimensions, making separation easier. For multi-class problems, SVM can employ one-vs-rest or one-vs-one strategies. However, careful selection of kernels and tuning of parameters are crucial to balance bias and variance and manage computational complexity.

SVM were also evaluated by Elhusseny et al. who reported a modest MCC score of 0.558 [36]. While SVM achieved competitive accuracy in some settings, it generally underperformed compared to other classifiers in their study. Bhattacharyya et al. explored SVM as an advanced classification technique and noted that its performance decreased with higher data imbalance, particularly in terms of AUROC [38]. Although SVM showed potential for improved fraud capture at deeper ranking positions, its effectiveness was limited without parameter tuning or advanced optimisation. These findings suggest that SVM may require careful configuration to be effective in practical fraud detection scenarios.

### 4.1.2 Tree-Based Models

Tree-based models are used in classification tasks due to their ability to model non-linear relationships and produce interpretable decision rules. In AML contexts, they offer a good

balance between predictive performance and transparency, making them a good choice for both baseline and production systems.

**Decision trees (DT)** are a popular tool for classification tasks, as they can predict either the most common class or provide class probabilities [35]. The model works by guiding each data point through a series of decision rules until it reaches a leaf node, where the final prediction is based on the distribution of training samples in that leaf. While their intuitive structure makes them easy to understand, decision trees can overfit the training data if they grow too complex, which negatively impacts their performance on new data. To create splits in the dataset, decision trees use metrics like entropy and Gini impurity[1]. Entropy measures the disorder of class labels within a node, helping the model select the most informative feature to split on. Gini impurity indicates the likelihood of misclassification, with lower values suggesting purer nodes and better class separation. Splitting continues until certain conditions are met, such as when all samples in a node belong to the same class or when a maximum depth is reached. Overall, while decision trees provide clear, rule-based predictions, they require careful management to prevent overfitting and ensure they generalise well to unseen data.

**Random Forest (RF)** is an ensemble machine learning method that combines multiple decision trees to improve predictive accuracy and reduce overfitting [39]. Each tree in the forest is trained on a different random subset of the data using bagging (bootstrap aggregation), and at each split, only a random subset of features is considered [35]. This randomness promotes diversity among the trees, making the ensemble more robust to noise and less prone to overfitting than individual decision trees. RF perform well with high-dimensional datasets and are relatively insensitive to hyperparameter settings, often achieving strong results with default configurations. Additionally, they offer feature importance scores, helping to identify which variables contribute most to model performance.

RF has been applied in AML and fraud detection tasks due to its robustness, ease of implementation, and strong performance on imbalanced and high-dimensional datasets. Liu et al. demonstrated that a RF ensemble significantly reduced false positives in corporate transaction data, improving the precision–recall balance in real-world AML applications compared to single decision trees [40]. Similarly, Bhattacharyya et al. found RF to be the best-performing model in

[1]Decision Trees Explained: towardsdatascience.com article

their study, outperforming both SVM and LR, particularly in detecting fraud at deeper ranking levels [38]. The model's minimal parameter tuning requirements and reliable default settings made it both effective and practical.

Elhusseny et al. further confirmed RF's strength by reporting the highest MCC score among all evaluated models—initially 0.848, which improved to 0.89 after hyperparameter tuning with Grid Search [36]. The tuned model achieved excellent classification accuracy, misclassifying only 31 instances out of over 71,000 transactions. These results highlight RF's suitability for large-scale, imbalanced fraud detection tasks due to its ensemble structure and resistance to overfitting.

RF has also served effectively as a baseline model in ensemble learning frameworks. In the study by Baisholan et al. it was used alongside LR and DT to benchmark the performance of their proposed stacking-based FraudX AI framework [41]. The baseline models helped illustrate the added value of ensemble techniques over traditional classifiers in credit card fraud detection.

### 4.1.3 Boosted Ensemble Models

Boosted ensemble methods build powerful predictive models by combining many weak learners (typically decision trees) into a single, stronger model. They are especially effective in handling complex patterns and imbalanced datasets, which makes them well suited for AML tasks such as money mule detection.

**Gradient Boosting Trees (GBT)** build an additive ensemble by fitting each successive decision tree to the negative gradient of a chosen loss function with respect to the predictions of the current model, thereby minimising prediction errors in a stage-wise manner [42]. The original formulation by Friedman demonstrated superior performance over single-tree methods in regression and classification tasks.

**eXtreme Gradient Boosting (XGBoost)** is a scalable, distributed gradient boosting framework that extends Friedman's original algorithm with regularised objective functions and a sparsity-aware split-finding method to handle missing values efficiently [43]. By incorporating both Lasso and Ridge regularisation, XGBoost mitigates overfitting, while its block-structured gradient boosting and out-of-core computation enable rapid training on large-scale transaction datasets. Additionally, it prunes unhelpful tree branches after training, helping to maintain model simplicity and generalisability.

Petr Hajek, Mohammad Zoynul Abedin, and Uthayasankar Sivarajah proposed an XGBoost-based fraud detection framework to address the challenges of extreme class imbalance in mobile payment transaction data [44]. They evaluated the performance of XGBoost alongside other supervised learning methods such as KNN, SVM, and RF. XGBoost outperformed all baseline models, demonstrating a balanced performance between precision and recall. In other AML applications, XGBoost has demonstrated significant reductions in false positives and improved detection recall compared to single-tree models, making it a reliable choice when balancing predictive power against computational cost [45, 46].

**Light Gradient Boosting Machine (LightGBM)** is a gradient boosting framework designed for high efficiency and scalability, particularly when working with large datasets and high-dimensional data [47]. It uses a histogram-based decision tree algorithm that converts continuous features into discrete bins, reducing the computational cost of split searches. LightGBM also employs a leaf-wise tree growth strategy, expanding the leaf that produces the greatest loss reduction, which results in fewer trees and faster convergence. Additionally, it prioritises data points with larger gradients for updates, improving training efficiency. Its flexible tree structure allows branches of varying depths, and it offers strong support for categorical features, making it especially effective for complex, real-world machine learning tasks.

In fraud detection benchmarks, LightGBM has consistently outperformed level-wise algorithms in both speed and accuracy, particularly on high-dimensional financial data, although its aggressive leaf-wise growth demands careful parameter tuning to avoid overfitting [48, 49].

**Categorical Boosting (CatBoost)** is a gradient boosting algorithm specifically designed to handle categorical features efficiently and with minimal preprocessing [50]. It introduces ordered boosting and a unique approach to encoding categorical variables by using permutations of the dataset and computing statistics only from preceding observations, which helps prevent target leakage and ensures unbiased gradient estimates. This built-in handling eliminates the need for manual one-hot encoding, simplifying the modelling pipeline. CatBoost also features a distinct tree-growing strategy and techniques to reduce overfitting, making it especially effective and easy to use for datasets with many categorical variables.

Studies in AML contexts reveal that CatBoost's native categorical support and robust regularisation get superior detection rates on transaction streams with numerous discrete attributes. For instance, Moparthi applied CatBoost to real-world banking data, achieving a 5% increase in F1-score and a 3% reduction in false positives compared to random forest and

XGBoost by using regularisation and automatic categorical encoding [51]. Similarly, Aldania et al. demonstrated on an imbalanced credit card dataset that CatBoost outperformed LightGBM, improving recall by 3% while reducing the false-positive rate by 2% [52].

Piotr Florek and Adam Zagdański explored gradient boosting models to evaluate their effectiveness and practicality in diverse classification tasks, particularly under varying data characteristics and computational constraints [49]. Their study found that LightGBM, after hyperparameter tuning, delivered the best performance in terms of accuracy, F1 score, AUC, and efficiency. In contrast, CatBoost and XGBoost performed well even without tuning, making them strong candidates for practical use. CatBoost, in particular, excelled in handling categorical variables but showed high runtime sensitivity to data dimensionality. The authors also noted that XGBoost and LightGBM are well-suited for high-dimensional and large-scale datasets due to their scalability and support for regularisation techniques.

## 4.2 Unsupervised Learning Models

Unsupervised learning techniques aim to uncover patterns in unlabelled data by identifying natural groupings, estimating probability distributions, or simplifying complex structures [53]. These methods such as clustering, density estimation, manifold learning, and anomaly detection do not rely on predefined labels and are particularly useful in exploratory data analysis, data compression, and feature extraction, especially in high-dimensional domains.

Among anomaly detection approaches, **Isolation Forest (IF)** is a tree-based method that isolates anomalies by randomly partitioning the feature space [54]. Anomalies, requiring fewer splits to be isolated, receive higher anomaly scores. Its linear time complexity and low memory usage make it suitable for large-scale, real-time AML pipelines. Abbassi et al. demonstrated its effectiveness in a real-time fraud detection system using Kafka and Spark, achieving high accuracy and low latency [55].

**Local Outlier Factor (LOF)** identifies anomalies based on local density comparisons, flagging data points that are significantly less dense than their neighbours [56]. This makes it effective when fraudulent transactions deviate from their surrounding context but remain embedded in otherwise legitimate clusters.

**Autoencoders** are a neural network-based models that are another unsupervised method for anomaly detection [57]. They learn to reconstruct typical transaction patterns and detect anomalies as those with high reconstruction errors. While powerful on complex, high-

dimensional datasets, autoencoders require significant computational resources and tend to lack interpretability. Fiore et al. applied a denoising autoencoder to credit card data and reduced false positives by 12% compared to Principal Component Analysis (PCA) based methods in a European bank pilot [58].

Despite their strengths, unsupervised methods often suffer from high false-positive rates and are sensitive to hyperparameter choices, such as tree depth in Isolation Forest or the latent space dimension in autoencoders [53]. Moreover, the absence of clear model explanations requires additional tools (like saliency maps or local explanation methods) to justify alerts and support compliance and audit requirements. Therefore, while effective, these models must be carefully validated and calibrated for practical AML deployment.

## 4.3 Stacked Ensemble Models

Model stacking is an ensemble method that improves predictive performance by combining the outputs of multiple base learners through a meta-model, which learns how to best weight their predictions [35]. Stacking combines the complementary strengths of diverse models to enhance accuracy.

Baisholan et al. proposed the FraudX AI framework as a practical application of model stacking for credit card fraud detection [41]. The framework combines random forest and XGBoost by averaging predicted probabilities and optimising thresholds to improve classification performance. It was evaluated on the European credit card dataset with its natural class imbalance and achieved a recall of 95% and an AUCPRC of 97%. FraudX AI outperformed several baseline models, demonstrating that stacking can offer an effective and interpretable solution for real-world fraud detection.

Shanshan Zhu et al. proposed a financial fraud prediction framework based on stacking ensemble learning [59]. The model integrates financial and non-financial indicators from companies and combines the outputs of several base learners (random forest, AdaBoost, XGBoost, LightGBM, and ExtraTrees) using logistic regression as the meta-learner. Base models predictions were used as input features for the final estimator. The stacking model achieved an AUROC of 0.8146 and a recall rate of 82.46%.

In the context of AML systems, stacking addresses key challenges such as data imbalance, high false-positive rates and model instability by integrating heterogeneous classifiers. The

meta-model refines the final output, allowing the system to capture complex patterns and make more reliable decisions across varied and imbalanced datasets.

## 4.4 Model Evaluation

Due to the typically high class imbalance and the significant consequences of prediction errors, evaluating the performance of machine learning models in the context of AML requires careful consideration.

The **confusion matrix** provides a comprehensive overview of classification outcomes. For binary classification it breaks down predictions into four fundamental categories [35]. True Positives (TP) represent the number of positive instances correctly identified by the model. True Negatives (TN) refer to the negative instances accurately classified as such. False Negatives (FN) indicate positive instances that the model failed to detect, and False Positives (FP) represent negative instances incorrectly classified as positive.

Beyond summarising classification outcomes, the confusion matrix enables the calculation of informative performance metrics, such as **precision**, **recall**, and the **F1-score**. Precision reflects the proportion of predicted positive cases that are actually correct, as shown in Equation 1.

$$\text{precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{1}$$

Recall measures the model's ability to identify all actual positive instances, as shown in Equation 2.

$$\text{recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{2}$$

The F1 score provides a harmonic mean of precision and recall, offering a balanced view of performance when both false positives and false negatives are important. It is calculated as shown in Equation 3.

$$\text{F1 score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{3}$$

These metrics have been used in the literature to evaluate models for fraud detection in various domains, including financial transactions [60], credit card fraud [61], and fraudulent investment schemes [62].

Despite their usefulness, confusion-matrix-based metrics can be misleading in highly imbalanced datasets [63]. For example, **accuracy** (Equation 4) can appear deceptively high when the majority class dominates, even if the model fails to detect minority-class instances.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{4}$$

Similarly, the **Area Under the Receiver Operating Characteristic Curve (AUROC)** can give an overly optimistic view of model performance in imbalanced scenarios (Equation 5) [63]. This is because the False-Positive Rate (FPR = $\frac{FP}{FP+TN}$) tends to be low when negative samples are occurring in large amounts, making the model appear better than it actually is at distinguishing between classes.

$$\text{AUROC} = \int_0^1 \frac{TP}{TP + FN} d\left(\frac{FP}{FP + TN}\right) \tag{5}$$

Nevertheless, AUROC remains commonly used and has been applied in studies aiming to detecting money laundering patterns analysing financial transactions [64, 65, 66].

To address the limitations of AUROC in imbalanced contexts, the **Area under Precision Recall Curve** (AUPRC) is preferred because it emphasises how well the model identifies positive cases while considering the trade-off between precision (avoiding false positives) and recall (capturing true positives) [63]. AUPRC represents the area under the curve that plots precision against recall at different classification thresholds. In binary classification, AUPRC focuses only on the performance of the positive class, making it more informative when the dataset is skewed. AUROC considers the performance of both positive and negative classes.

$$\text{AUPRC} = \int_0^1 \frac{TP}{TP + FP} d\left(\frac{TP}{TP + FN}\right) \tag{6}$$

AUPRC is calculated by integrating precision ($\frac{TP}{TP+FP}$) over recall ($\frac{TP}{TP+FN}$) as seen in Equation 6. The model balances well both false positives and false negatives at different decision thresholds. Models in imbalanced datasets often need to be tuned carefully to avoid excessive

false positives or false negatives. AUPRC provides a robust way to compare models based on their ability to correctly identify positive instances without being biased by the majority class. This makes AUPRC particularly suitable for tasks such as credit card fraud detection [61, 67, 68, 41].

In addition to traditional performance metrics, **cost-sensitive metrics** are also used in AML applications. These metrics translate classification errors into monetary terms by assigning costs to false positives (unnecessary investigations) and false negatives (undetected fraudulent activities resulting in financial loss). Cost-sensitive evaluation has been employed in credit card fraud detection [60] and in semi-supervised approaches for identifying suspicious transactions [69].

**SHapley Additive exPlanations (SHAP)** values help explain how machine learning models make predictions [70]. They assign an importance score to each feature, showing how much it contributes to a specific prediction. It is known for having a unique solution that meets desirable properties like consistency and local accuracy. SHAP also brings together several existing explanation methods and improves on them by being more reliable and easier to understand.

Together, these evaluation strategies offer a more comprehensive and realistic assessment of model performance in AML settings, where class imbalance and the consequences of errors necessitate more nuanced approaches than standard accuracy metrics alone.

## 4.5 Overview and Key Takeaways

Each machine learning approach discussed (supervised, unsupervised, and ensemble methods) offers distinct strengths and addresses specific challenges in AML. However, the specific domain of money mule detection remains relatively under-explored in the literature.

Supervised learning techniques provide clear interpretability and robust predictive performance, especially beneficial when historical labelled data is available. These methods allow precise risk scoring, making them ideal for directly predicting the likelihood of a customer becoming a money mule.

Unsupervised anomaly detection methods are valuable for identifying new or previously unseen fraudulent behaviours without the requirement of labelled data. While these approaches are excellent at detecting unusual patterns and emerging threats, their limitations in interpretability and higher false-positive rates may complicate practical implementation.

Ensemble methods are used in machine learning to improve predictive performance, particularly in challenging settings such as imbalanced classification. By combining multiple models, ensembles can reduce variance, mitigate overfitting, and better capture complex patterns in the data. In the context of imbalanced datasets, ensemble approaches are especially valuable as they can help balance sensitivity and specificity by leveraging diverse decision boundaries. As such, ensemble learning is recommended as a strategy to enhance overall model effectiveness.

In practice, the effectiveness depends not only on the diversity of models but also on how their performance is evaluated. Traditional metrics like accuracy are often insufficient in imbalanced settings, as they can obscure poor performance on the minority class. Instead, AUPRC is more appropriate, as this can better reflect a model's ability to identify rare events.

These insights can be used to design and evaluate a machine learning based scoring system for money mule detection. Understanding the strengths of different models, the importance of selecting appropriate evaluation metrics, and the challenges posed by class imbalance provides a foundation for building predictive models.

# 5. Methodology

The empirical side of this thesis aims to provide stakeholders with a trained machine learning model as a scoring system that evaluates all customers on a scale from 0 to 100. The score will indicate the likelihood of a customer being a money mule. When a high-risk transaction is initiated towards an account with a compromised reputation (a customer score exceeding a defined threshold), the system should automatically flag the transaction for review and generate a notification to trigger further follow-up and mitigation actions. This procedure is intended to prevent funds from potential fraud victims from reaching accounts associated with money mules, thereby reducing financial harm. Stakeholders require the model to update scores monthly, based on the customer's profile, behavioural patterns, product usage, and changes in behavioural volume over the preceding six months. The authors suggest retraining the model monthly to adapt to constantly evolving fraud schemes.

The process of developing required model follows the Cross-Industry Standard Process for Data Mining (CRISP-DM) [71] framework, a widely adopted methodology for structuring data science and machine learning projects. CRISP-DM consists of six iterative phases: business understanding, data understanding, data preparation, modelling, evaluation, and deployment. Although the framework was released in 2000, it still remains relevant. In 2021, Schröer, Kruse, and Gómez published a systematic literature review in which they claim that CRISP-DM is the de facto standard in data mining [72]. Also, the structure of methodology chapter follows the CRISP-DM framework. The business understanding phase is not described due to confidentiality, and deployment, is outside the scope of this thesis. The focus of this thesis is solely on identifying which machine learning models and configurations achieve the best predictive performance, leaving full system integration for future work.

## 5.1 Data Understanding

Two separate datasets containing confirmed money mule cases were used for this work. The vast majority of customers in these datasets were private individuals, and in agreement with stakeholders, the authors chose to focus exclusively on private customers. Only rows representing distinct private money mule customers were retained. In cases where multiple entries existed for the same customer, the earliest transaction date was selected. Most of these customers had a valid transaction date available.

To get as many data points as possible from stakeholders, the authors tried to estimate the transaction date for some mules that did not have one. However, the results were not reliable due to a lot of noise affecting the estimation.

Since it is a binary classification task, it is important to clearly define both classes. For legitimate customers (non–money mules), the most recent available six-month period of financial data was used. Ideally, only money mules active during the same period would have been selected, but due to the low number of identified cases, all known money mules from the past few years were included. For each mule, the six-month period preceding their confirmed fraudulent transaction was used. For example, if the transaction occurred on 23 January 2024, data from July 2023 to December 2023 was selected.

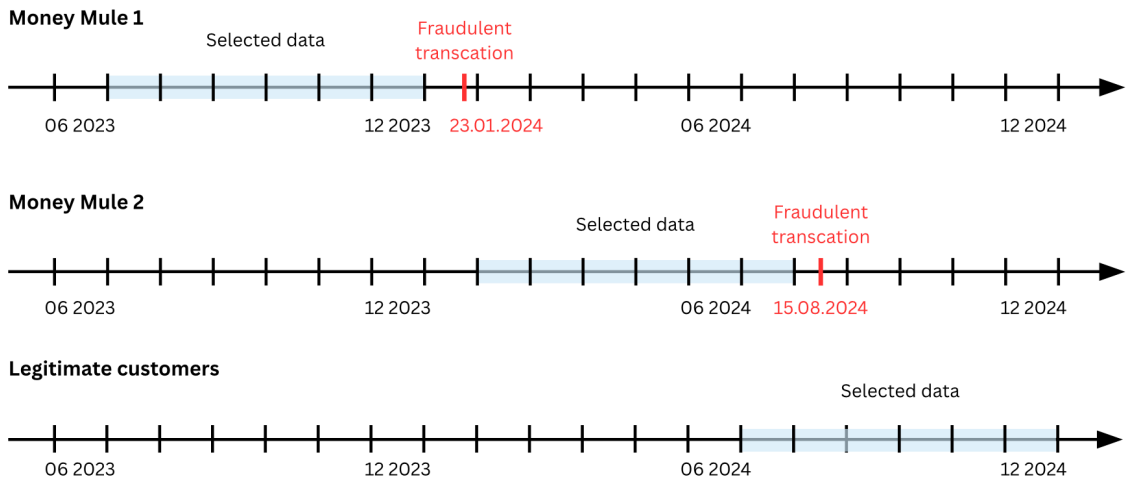



Figure 3. Selected data time-period for money mules and legitimate customers.

The data selection process is illustrated in Figure 3. The first two timelines demonstrate how data was selected for money mules, and the third timeline shows the period from which data was selected for legitimate customers. The transaction dates in this example are illustrative and do not correspond to real individuals. This data selection approach, while necessary due to the limited number of confirmed cases, introduces a potential temporal bias, as the legitimate customer data comes from a fixed recent period, whereas mule data spans multiple years and may reflect changing behavioural patterns over time.

To ensure data quality, only a sample of active legitimate customers with relevant transaction activities was included in the dataset. Inactive customers were excluded to reduce noise and improve model reliability. To prevent overlap between known money mules and legitimate customers, an anti-left join was performed between the legitimate customer dataset and the

money mule dataset. However, as the data only includes confirmed mule cases, it is possible that some undetected money mules remain in the legitimate customer sample. This label noise introduces a potential source of selection bias, but given the limited number of confirmed cases, it was considered an acceptable trade-off. The final dataset remains highly imbalanced, with an approximate ratio of 1:1000 between money mules and legitimate customers.

## 5.2 Data Preparation

To select the features, available financial data in the bank was explored and it was decided to work with four key tables and one reference dataset from the bank's internal systems. These were identified as the most relevant sources for capturing customer profiles and behavioural patterns that can be used to create batch predictions monthly.

Having too many features can spread out data points, making it difficult for models to find clusters or patterns, a problem known as the curse of dimensionality [73]. To address this, exploratory data analysis was conducted to understand the features and their importance related to the research question. Understanding specific banking terminology and abbreviations, often in Swedish, was time-consuming and challenging. The analysis involved checking for missing values, inconsistencies, and outliers to identify issues within the data. Features with no variation were removed, since these do not provide value.

Of the 122 customer profile features, one was retained unchanged, three were converted to numeric values, nine were binarised (indicating whether each attribute applies), seven new flags were added to show if a feature had changed over the past six months, and one feature was derived from a reference table for extra context.

From the 165 customer behaviour features, 67 were selected and aggregated into six-month sums. Of the 89 product usage features, a new feature was created counting the number of products each customer holds and retained one specific usage feature, also summed over six months. From the 39 volume-based behaviour features, 32 were kept and similarly aggregated.

This left 122 features out of the original 415. A correlation matrix revealed no leakage issues. Where pairs of features exceeded 90% mutual correlation, only the one most correlated with the target was kept. Then all features whose absolute correlation with the target was at least 0.1 were selected. The final set comprises 36 numeric features (excluding the label): 10 profile, 23 behaviour, one product-usage and two volume-based. Any missing values were filled with zero.

## 5.3 Modelling

The modelling section follows a five-step pipeline, detailed below. First, engineered features are made into a single vector and, where algorithms are scale-sensitive, standardised. Second, the data are split into training, validation and test sets. The training set is then rebalanced by random sampling or by applying class-specific weights. Third, a range of machine learning models are trained in PySpark, and their hyperparameters are tuned with Optuna on the validation set. Fourth, the best-tuned models are evaluated once on the test set to verify their generalisation performance. Fifth, the models' prediction scores are combined into simple ensembles that blend a high-precision predictor with an high-recall detector.

The models evaluated in this study are logistic regression, k-nearest neighbours, support vector machine with linear kernel (linear SVM), decision tree, random forest, gradient boosted trees, XGBoost, CatBoost, LightGBM, and Isolation Forest. Given the limited research specifically focused on the application of machine learning to money mule detection, a wide range of models was tested to explore their relative performance. Since the study was conducted using labelled data, nine of the models are supervised learning algorithms. In addition, Isolation Forest, an unsupervised anomaly detection model, was included to investigate whether such methods can offer value in highly imbalanced classification tasks.

Due to the large volume of data, PySpark[2], the Python API for Apache Spark, was used for distributed processing. The computational cluster was configured with worker type with 128–640 GB memory, 32–160 cores, 1–5 workers and driver type with 14 GB memory, 4 cores, 1 driver.

When feasible, machine learning models from the PySpark library were utilised, as Spark is optimised for efficiently handling larger datasets. This included models such as logistic regression, linear SVM, decision tree, gradient boosted trees, and random forest. The K-NearestNeighboursClassifier[3] was imported from sklearn.neighbors. For the XGBoost model SparkXGBClassifier[4] was imported from xgboost.spark. LightGBMClassifier[5] and

[2]PySpark Overview: `https://spark.apache.org/docs/latest/api/python/index.html`
[3]KNeighborsClassifier: `https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html`
[4]Distributed XGBoost with PySpark: `https://xgboost.readthedocs.io/en/stable/tutorials/spark_estimator.html`
[5]LightGBM on Apache Spark: `https://microsoft.github.io/SynapseML/docs/Explore%20Algorithms/LightGBM/Overview/`

IsolationForest[6] was sourced from the package Synapse ML, and the CatBoost[7] package was imported from PyPI.

After feature engineering and prior to model training, several preprocessing steps were required to ensure the data was suitable for input into the models. Since different models have different requirements, various methods were applied accordingly. The following provides an overview of the preprocessing techniques used.

**Vector assembler**[8] was employed to merge multiple columns into a single vector column. This step was crucial for models from the PySpark library because it allows different features to be handled together in a single format, enabling efficient processing and analysis in machine learning tasks.

**Standard Scaler**[9] was used because some models, such as logistic regression, k-nearest neighbours, and linear SVM require the input data to be scaled before training. This is because these algorithms are sensitive to the range and magnitude of the features. For example, if one feature represents age (ranging from 0 to 100) and another represents salary (ranging from 0 to 100 000), the model may give undue importance to salary simply because of its larger numerical values. Scaling ensures that each feature contributes equally to the model, regardless of its original range, thereby improving performance and fairness.

Next step in the pipeline is to **split dataset into train, validation and test data** using randomSplit[10] with ratio 0.7/0.15/0.15 (corresponding percentage of money mules in the splits is 12%/12%/13%), and using seed 42 to ensure consistent dataset splitting. However, it was later discovered that the data distribution varied slightly due to difference in usage of caching. Without caching, the results are not reproducible within the PySpark framework due to its distributed nature, where data can be processed on different workers. Since the split differences were minor, these variations were considered acceptable.

---

[6]Multivariate Anomaly Detection with Isolation Forest: `https://microsoft.github.io/SynapseML/docs/Explore%20Algorithms/Anomaly%20Detection/Quickstart%20-%20Isolation%20Forests/`

[7]catboost: `https://pypi.org/project/catboost/`

[8]VectorAssembler: `https://spark.apache.org/docs/latest/api/python/reference/api/pyspark.ml.feature.VectorAssembler.html`

[9]StandardScaler: `https://spark.apache.org/docs/latest/api/python/reference/api/pyspark.mllib.feature.StandardScaler.html`

[10]randomSplit: `https://spark.apache.org/docs/latest/api/python/reference/pyspark.sql/api/pyspark.sql.DataFrame.randomSplit.html`

For machine learning models, such as k-nearest neighbors and CatBoost, data had to be converted from Spark DataFrame to Pandas DataFrame because these models require data in a format compatible with their libraries, which are typically designed to work with Pandas. This conversion process increased computation time due to the additional steps required to transform and transfer data between different formats. Furthermore, since Pandas cannot leverage distributed computation, all processing occurs on the driver node while the workers remain idle.

**Sampling methods** were applied to only the training data after splitting the data into a training, validation and test dataset. For undersampling, random undersampling was used to reduce the size of the majority class (legitimate customers). The undersampling ratio was calculated based on the relative frequencies of the minority and majority classes. A random sample of the majority class was selected to match this ratio. For oversampling, a random oversampling was applied to the minority class (money mules). Each minority instance was replicated multiple times so that the total number of minority instances approximately matched the majority class count. This was done by using the explode function on an array of constants to efficiently duplicate rows.

Also, **adjusting class weights** with no sampling was tried. Class weights do not actually affect how the model goes about making each prediction, only how the model updates its weights during optimisation. This means that the class weight adjustments will impact the loss function the model employs when making its predictions.

Table 1 provides an overview of the methods applied to or tried out on each model. Initially, the plan was to obtain results for all models using various sampling techniques and weight adjustments. However, some approaches were later abandoned as they required excessive computation time and no longer seemed feasible given the business objectives.

Table 1. Preprocessing and sampling methods used across different models.

| Model | Vector Assembler | Scaler | Undersampling | Oversampling | Adjusted Weights |
|---|---|---|---|---|---|
| LR | x | x | x | x | x |
| KNN | - | x | x | - | - |
| Linear SVM | x | x | x | x | x |
| DT | x | - | x | x | x |
| RF | x | - | x | x | x |
| GBT | x | - | x | x | x |
| XGBoost | x | - | x | x | x |
| CatBoost | - | - | x | x | x |
| LightGBM | x | - | x | x | x |
| IF | x | - | x | x | - |

**Optuna** [74] was used to automatically tune hyperparameters in this study. To ensure a fair evaluation, each model was tested for 50 trials, except KNN (10 trials), and CatBoost and XGBoost (30 trials), as they faced Out of Memory errors with longer runs. The Area Under Precision and Recall Curve (AUPRC) metric (discussed in the Section 4.4) was selected as the optimisation metric for Optuna, with the goal of maximising its value. Each model was initially assigned a broad parameter range to explore a wide variety of configurations during hyperparameter tuning.

Train and validation data is used in Optuna trial run where the hyper-parameter tuning is performed to find the best parameters. After this the best parameters are used to train the model again but then evaluated on test data. This approach will show if the model has been overfitting. In this configuration, the test set serves as an independent evaluation to verify that models generalise well beyond the validation data used during hyperparameter tuning. This approach is similar to a train/test split but with an additional validation set for hyperparameter optimization.

AUPRC is sensitive to the number of cases. As validation data has approximately on 1% less money mules than test data, it is expected that the AUPRC on test data is slightly larger. When AUPRC on test data is lower than AUPRC on validation data, this can be good indicator that suggests that the hyperparameter tuning process may led to some degree of overfitting to the validation data.

All supervised models in this study output probability scores. In contrast, Isolation Forest produces an anomaly score, which was normalised using Min-Max scaling to ensure comparability with the probability-based outputs of the other models.

To further improve performance, the predictions of multiple models were combined into an **ensemble model**, with the aim of leveraging their individual strengths to achieve better overall results. The goal is to select one model that is better in detecting money mules (meaning it prioritises high recall) and another that is better at identifying legitimate customers (meaning it prioritises high precision).

There were two approaches to combine the models. The first was to apply different weights for each model's predictions. The weights were applied so that they add up to one. Each model combination was iterated through weights from 0 to 1 with a step of 0.1. Then the combination was iterated through a threshold from 0 to 1 with a step of 0.1, which classified the customer as a money mule or a legitimate customer.

The second approach was to create a rule-based selection. This meant that when the high-precision model's prediction score was over a certain threshold, the high-precision model's score was kept. Otherwise high-recall model's prediction score was taken. Then again, the combination was iterated through a threshold from 0 to 1 with a step of 0.1 that classified the customer as a money mule or a legitimate customer.

## 5.4 Evaluation

The primary metric used in thesis for evaluation is **Area under Precision Recall Curve (AUPRC)**. It is preferred because this emphasises how well the model identifies positive cases while considering the trade-off between precision (avoiding false positives) and recall (capturing true positives). As discussed in Chapter 4.4, in binary classification, it focuses only on the performance of the positive class, making it more informative when the dataset is skewed.

The baseline AUPRC corresponds to the proportion of positive cases, and any model achieving a substantially higher AUPRC demonstrates meaningful predictive power [75]. The dataset used in this thesis exhibits a class imbalance of approximately 1:1000, resulting in a baseline AUPRC of 0.001.

**The confusion matrix** was included as one of the evaluation metrics because it provides easy and more in depth overview of potential outcomes if the model were implemented in real life. The confusion matrix was calculated using MulticlassMetrics[11] from Spark. True Positives (TP) represent the number of money mules correctly identified by the model and True Negatives (TN) the legitimate customers accurately classified as non-mules. Additionally, False Negatives (FN) highlight those money mules that were not detected by the model and False Positives (FP) indicate legitimate customers mistakenly labelled as money mules.

Beyond its initial role in summarising outcomes, the confusion matrix results serve to further explain model performance through calculations of **precision**, **recall**, and **F1-score**. Precision measures how many instances labelled as money mules truly are money mules. Recall assesses how effectively all real money mules are identified by evaluating both correctly classified

[11] confusionMatrix: https://spark.apache.org/docs/latest/api/python/reference/api/pyspark.mllib.evaluation.MulticlassMetrics.html?#pyspark.mllib.evaluation.MulticlassMetrics.confusionMatrix

instances and those missed. The F1-score offers a balanced perspective between recall and precision, managing false positives and negatives.

Even though both precision and recall are important, in banking, precision takes precedence as false positives results in time spent in due diligence activities for not relevant cases. However, maintaining an appropriate balance is essential, as prioritising precision too much reduces the model's ability to detect actual money mules. Therefore, the optimal trade-off must consider both the cost of missed detections and the burden of excessive false alerts.

In evaluating various machine learning models, it was determined that using a threshold of 0.5 would be optimal. This means that if a customer's probability score for being classified as a money mule reaches 0.5 or higher, they are labelled accordingly. Otherwise, they remain categorised as legitimate customers. Although this fixed threshold offers simplicity and interpretability, it is important to note that the optimal threshold varies across models and should ideally be selected based on the desired balance between recall and precision. This balance is typically determined by domain-specific considerations and falls outside the scope of this thesis.

Even though **accuracy** and **AUROC** can be misleading in the context of highly imbalanced datasets, they were still included in the evaluation to provide a familiar baseline and to support comparison with other studies that commonly report these metrics.

**Cost-Based Validation of Classification Results** is one of the method that could be used for validating the results. The approach is to consider the relative costs of false positives and false negatives, which helps to better understand the practical implications of different model outcomes.

For false positives, the stakeholders had already estimated the cost based on the due diligence. This gives a reasonable approximation of the operational impact of over-alerting. For false negatives, it was more difficult to determine a concrete cost. The authors decided to estimate it by calculating the average amount of money that money mules were able to transfer successfully. In other words, the average financial loss experienced by fraud victims was used as a way to measure the cost of undetected fraud.

According to the Swedish Payment Services Act (2010:751) [76], banks are generally obligated to reimburse customers for unauthorised transactions, unless the customer has acted fraudulently or with gross negligence (Chapter 5, Sections 3–8). If a customer submits a reimbursement

request and the bank determines that the transaction was indeed unauthorised, the customer will be compensated.

Based on these metrics, the cost of a false negative was estimated to be approximately 1300 times higher than that of a false positive. However, this validation should be interpreted with caution, as the assumptions made in the cost estimation process are simplifications and may not capture the full complexity of real-world cases.

# 6. Results

This section presents the results of various machine learning models developed to predict the likelihood of a client being a money mule. The models evaluated in this study include: logistic regression, k-nearest neighbours, support vector machine with linear kernel (linear SVM), decision tree, random forest, gradient boosted trees, XGBoost, CatBoost, LightGBM, and Isolation Forest. Each model was trained on the same preprocessed dataset and underwent hyperparameter tuning using Optuna.

Each model's performance is evaluated based on the time of one Optuna trial run with the best hyperparameters, AUROC, AUPRC, the confusion matrix with threshold 0.5, and metrics that depend on the confusion matrix, such as accuracy, precision, recall, and F1-score. The AUPRC receives the most emphasis, as it is particularly well-suited for evaluating models on highly imbalanced datasets. The confusion matrix is represented in percentages, and the ratio between money mules and legitimate customers is approximately 1:1000. All models are compared on test data to provide unbiased performance estimates. AUPRC on validation data was added to identify when Optuna parameter search might overfitted to the validation data, resulting in parameter configurations that do not generalize as well to unseen data.

For every model, the best results obtained across different sampling techniques are reported. Based on the top-performing sampling approach, an AUPRC score plot and a predictions distribution plot are included. On each predictions distribution plot the red colour refers to money mules and blue/purple to the legitimate customers. For each model, the Appendix includes table with range of given parameters and the Optuna-selected best parameters (Appendix I), and based on the best-performing sampling technique there is additional Optuna plots showing the optimization history (Appendix II), parameter importances (Appendix III), and parallel coordinates (Appendix IV).

## 6.1 Logistic Regression

Table 2 presents the performance of the logistic regression model under different sampling techniques: no sampling, undersampling, oversampling, and adjusted class weights. The model trained without any sampling achieved the highest accuracy (0.9987), but due to the imbalanced dataset, this metric is misleading. When looking at the confusion matrix, it can be seen that no money mules were detected. The confusion matrices were generated using a fixed threshold of 0.5. While this threshold may not be optimal for each model, it provides a general sense of

how the predictions are distributed. As there are no detected money mules in that threshold, it means that prediction scores were leaning more towards the lower end, and there is no good distribution between money mules and legitimate customers.

In contrast, all three balancing strategies improved the model's ability to detect money mules. Undersampling test-set AUPRC was 0.0378. At the fixed 0.5 threshold, this model achieved a recall of 0.8358 (correctly identifying 83.6% of money-mule cases), a precision of 0.0079, and an F1 score of 0.0157. Random oversampling further increases the AUPRC to 0.0396. At the same threshold, it had a precision of 0.0078, recall of 0.8517, and F1 score of 0.0154, closely matching the undersampling results.

Adjusting class weights got the best overall performance. It has the highest AUPRC (0.0402), which is more on the lower side but still better than baseline AUPRC. Adjusting class weights also had the highest AUROC (0.9398), precision (0.0089), recall (0.8623) and F1 score (0.0175). On the threshold 0.5, the adjusted weights model correctly identified 86.2% of the positive class instances while having a relatively low false positive rate (12.3%).

Table 2. Performance metrics for logistic regression with different sampling techniques.

| **Metric** | | **No sampling** | | **Undersampling** | | **Oversampling** | | **Adjusted weights** | |
|---|---|---|---|---|---|---|---|---|---|
| Time (s) | | 43.5 | | 41.2 | | 46.9 | | 84.0 | |
| Accuracy | | 0.9987 | | 0.8664 | | 0.8610 | | 0.8766 | |
| AUROC | | 0.8549 | | 0.9301 | | 0.9319 | | 0.9398 | |
| AUPRC (val) | | 0.0182 | | 0.0351 | | 0.0361 | | 0.0363 | |
| AUPRC (test) | | 0.0238 | | 0.0378 | | 0.0396 | | **0.0402** | |
| Precision | | - | | 0.0079 | | 0.0078 | | 0.0089 | |
| Recall | | 0 | | 0.8358 | | 0.8517 | | 0.8623 | |
| F1 Score | | - | | 0.0157 | | 0.0154 | | 0.0175 | |
| **Confusion Matrix** | | Predicted | | | | | | | |
| | | No | Yes | No | Yes | No | Yes | No | Yes |
| Actual | No | 100% | 0% | 86.6% | 13.4% | 86.1% | 13.9% | 87.7% | 12.3% |
| | Yes | 100% | 0% | 16.4% | 83.6% | 14.8% | 85.2% | 13.8% | 86.2% |

The best performing model here is logistic regression with adjusted weights. In Figure 4 it can be seen that most of the curve lies near the bottom, which means the model prioritises recall at the cost of precision and performs overall poorly. This indicates that even with class weights, the model isn't distinguishing money mules from regular customers effectively.

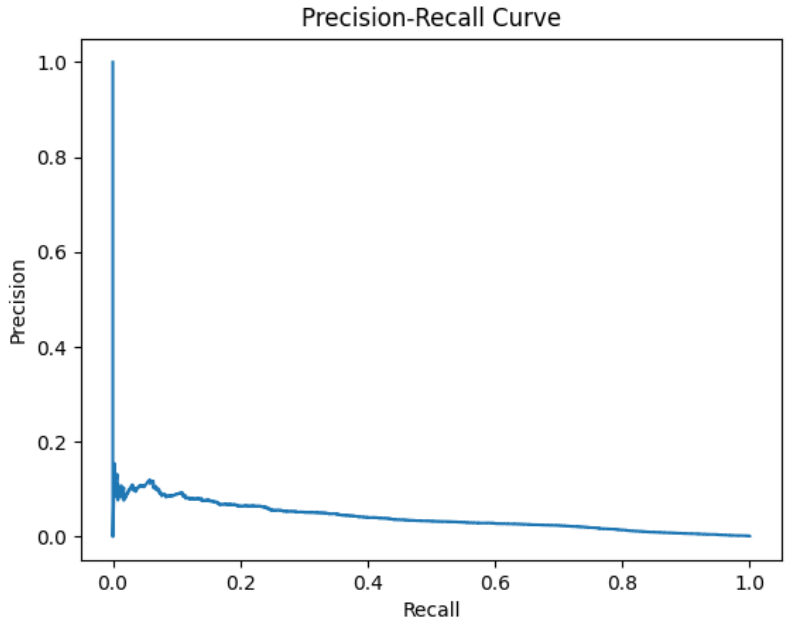



Figure 4. AUPRC plot for logistic regression with adjusted weights.

Figure 5 is a histogram of a predicted probability distribution. It can be seen that most of the predictions are leaning towards 0, meaning the model is confident about these being legitimate customers. Looking at the other end, it can be seen that the model is confident with most of the money mules. However, there is some of spread of red bars across the probability range, especially between 0.3 and 0.9, which means that the model isn't fully confident for some portion of true mules. There is also overlap between blue and red in the range 0.3 to 0.7. This region is where false positives and false negatives are likely to occur. This explains the poor AUPRC, as the model struggles to separate classes.

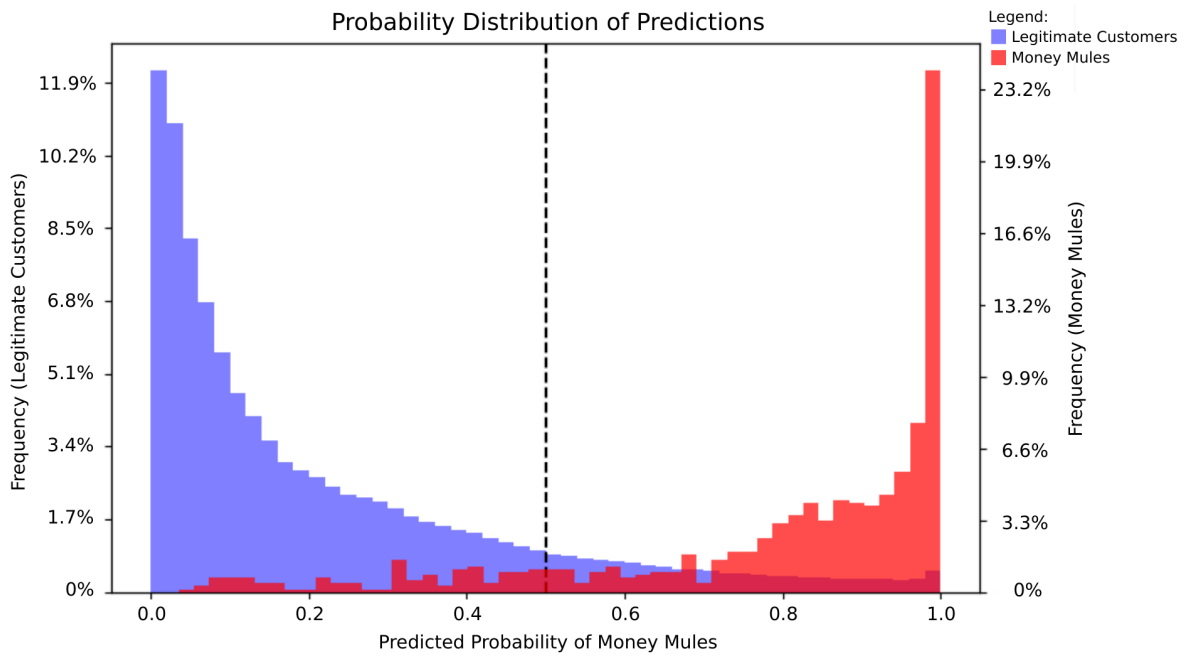



Figure 5. Predicted probability distribution for logistic regression with adjusted weights.

## 6.2 K-Nearest Neighbours

As shown in Table 3, the KNN model trained on an undersampled dataset got an AUPRC of 0.0278. This is a bit lower when compared to the logistic regression model using the same sampling strategy, with AUPRC = 0.0378. This means that KNN is not better at distinguishing the money mules in this imbalanced dataset.

For KNN, only the undersampled model results were recorded. A training attempt without any sampling technique was cancelled due to excessive runtime that exceeded one hour for a single Optuna trial run. This highlights the algorithm's poor scalability on large datasets.

While KNN achieved a relatively high accuracy of 89.66%, this figure is misleading due to the class imbalance and does not reflect the model's ability to capture true positives. The bigger setback is that the model is computationally inefficient. Given both its lower predictive performance and high computational cost, it was decided that KNN would not be explored for further experimentation.

Table 3. Performance metrics for k-nearest neighbours with different sampling techniques.

| **Metric** | | **Undersampling** | |
|---|---|---|---|
| Time (s) | | 324 | |
| Accuracy | | 0.8966 | |
| AUROC | | 0.9410 | |
| AUPRC (val) | | 0.0278 | |
| AUPRC (test) | | **0.0334** | |
| Precision | | 0.0099 | |
| Recall | | 0.8386 | |
| F1 Score | | 0.0195 | |
| **Confusion Matrix** | | Predicted | |
| | | No | Yes |
| Actual | No | 89.7% | 10.3% |
| | Yes | 16.1% | 83.9% |

The shape of the AUPRC curve in Figure 6 indicates that the model returns a high number of false positives when attempting to identify more true positives. While the model achieves perfect precision at very low recall, its performance drops sharply as recall increases, resulting in a low AUPRC. Although KNN can benefit from undersampling, it often struggles with imbalanced datasets due to its sensitivity to noise and the loss of informative majority class samples.

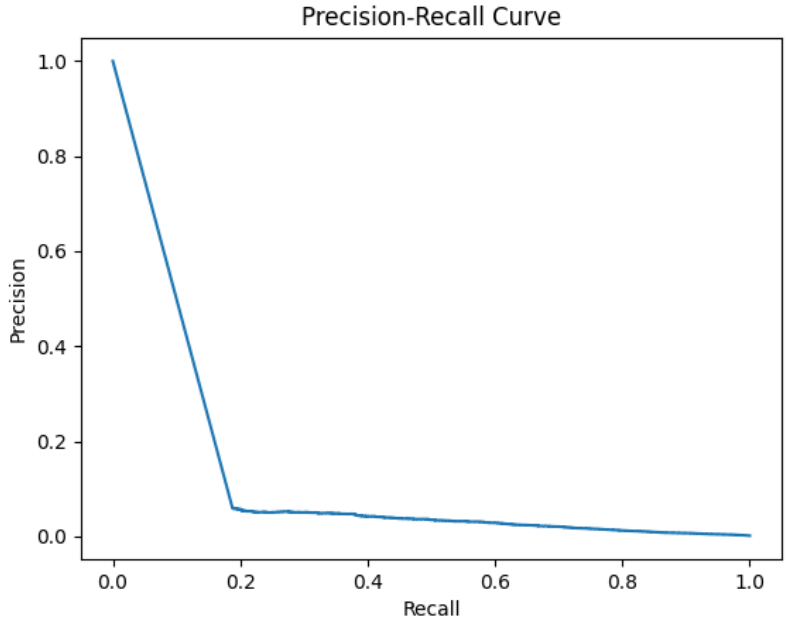



Figure 6. AUPRC plot for k-nearest neighbours with undersampling.

The probability distribution in Figure 7 shows that the model is somewhat confident in classifying money mules and legitimate customers, but not accurate enough to be considered for production use. There is also notable overlap between the two classes in the range of predicted probabilities between 0.2 and 0.8. This overlap is where false positives and false negatives are most likely to occur.

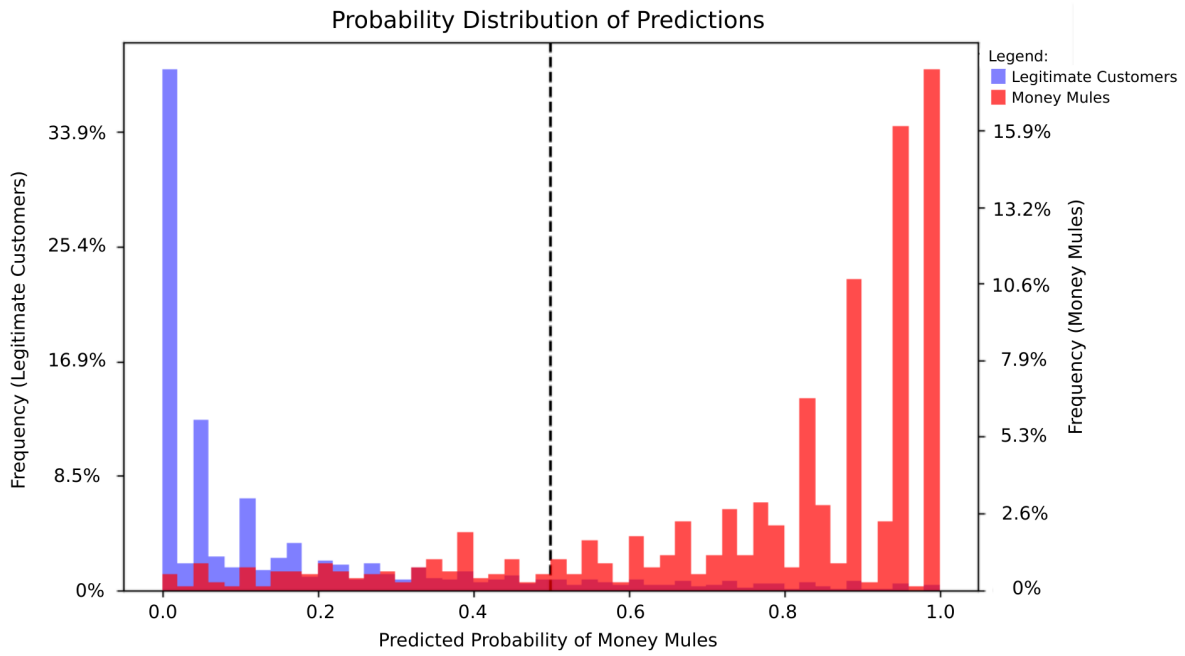



Figure 7. Predicted probability distribution for k-nearest neighbours with undersampling.

## 6.3 Support Vector Machine with Linear Kernel

Table 4 summarises the outcomes of training the linear SVM using four different approaches to address class imbalance. When trained on the dataset without any sampling, the classifier got high overall accuracy (0.9987), but it failed to detect any instances of the positive class at the threshold of 0.5. This indicates that predictions cluster more in the lower corner. As a result, key performance indicators such as recall, precision, and F1 score were either zero or undefined.

In contrast, the resampling techniques or class weight adjustment led to an improvement in the model's ability to recognise positive cases. Among the three balanced strategies, oversampling and adjusted class weights produced the most favourable results across multiple evaluation metrics. Both achieved AUROC values above 0.94, and test AUPRC scores around 0.039. These gains were also reflected in the F1 score (0.0179), driven by high recall (above 86%), but low precision (0.0090).

Models trained with balancing strategies were able to recover a majority of true positive cases, with oversampling very slightly outperforming others in terms of true positive rate (86.5%) and maintaining false positives around 12.1%. From a computational perspective, training times remained within a reasonable range, with the adjusted weights method being the fastest (84 seconds) and the baseline the slowest (108 seconds).

Table 4. Performance metrics for linear SVM with different sampling techniques.

| **Metric** | | **No sampling** | | **Undersampling** | | **Oversampling** | | **Adjusted weights** | |
|---|---|---|---|---|---|---|---|---|---|
| Time (s) | | 108.0 | | 96.0 | | 102.0 | | 84.0 | |
| Accuracy | | 0.9987 | | 0.8775 | | 0.8787 | | 0.8787 | |
| AUROC | | 0.8786 | | 0.9385 | | 0.9410 | | 0.9410 | |
| AUPRC (val) | | 0.0197 | | 0.0328 | | 0.0351 | | 0.0348 | |
| AUPRC (test) | | 0.0234 | | 0.0375 | | **0.0394** | | 0.0392 | |
| Precision | | - | | 0.0089 | | 0.0090 | | 0.0090 | |
| Recall | | 0 | | 0.8570 | | 0.8649 | | 0.8636 | |
| F1 Score | | - | | 0.0175 | | 0.0179 | | 0.0179 | |
| **Confusion Matrix** | | Predicted | | | | | | | |
| | | No | Yes | No | Yes | No | Yes | No | Yes |
| Actual | No | 100% | 0% | 87.8% | 12.2% | 87.9% | 12.1% | 87.9% | 12.1% |
| | Yes | 100% | 0% | 14.3% | 85.7% | 13.5% | 86.5% | 13.6% | 86.4% |

Even though adjusted weight was almost as good, the oversampling strategy was selected as the best configuration for the linear SVM model. The AUPRC curve (Figure 8) is quite at the bottom, meaning the model prioritises recall at the cost of precision.

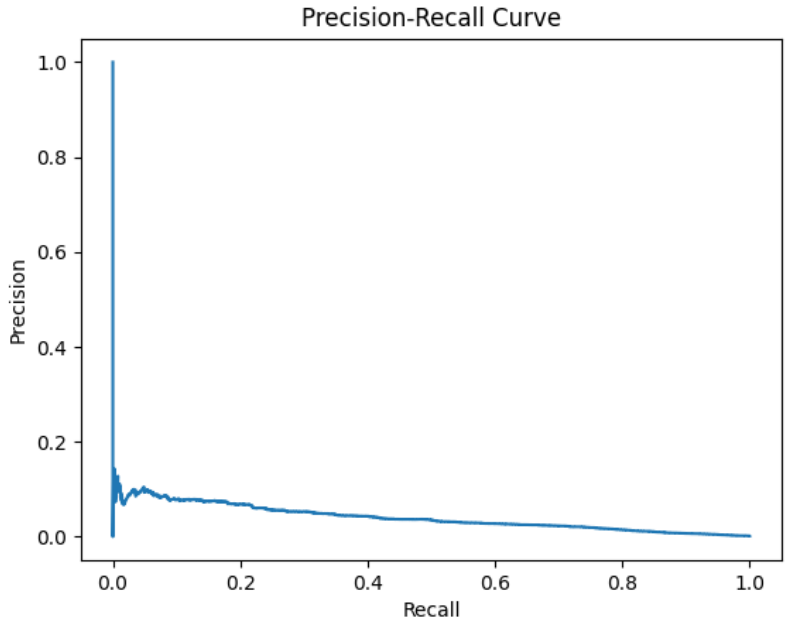



Figure 8. AUPRC plot for linear SVM with oversampling.

On the distribution of predicted probabilities (Figure 9), it can be seen that the distribution of legitimate customers is not really steep, meaning that the model is not very confident in detecting legitimate customers. The mule predictions are better but not quite perfect, as there is still some probabilities in the range of 0.4 to 0.9.

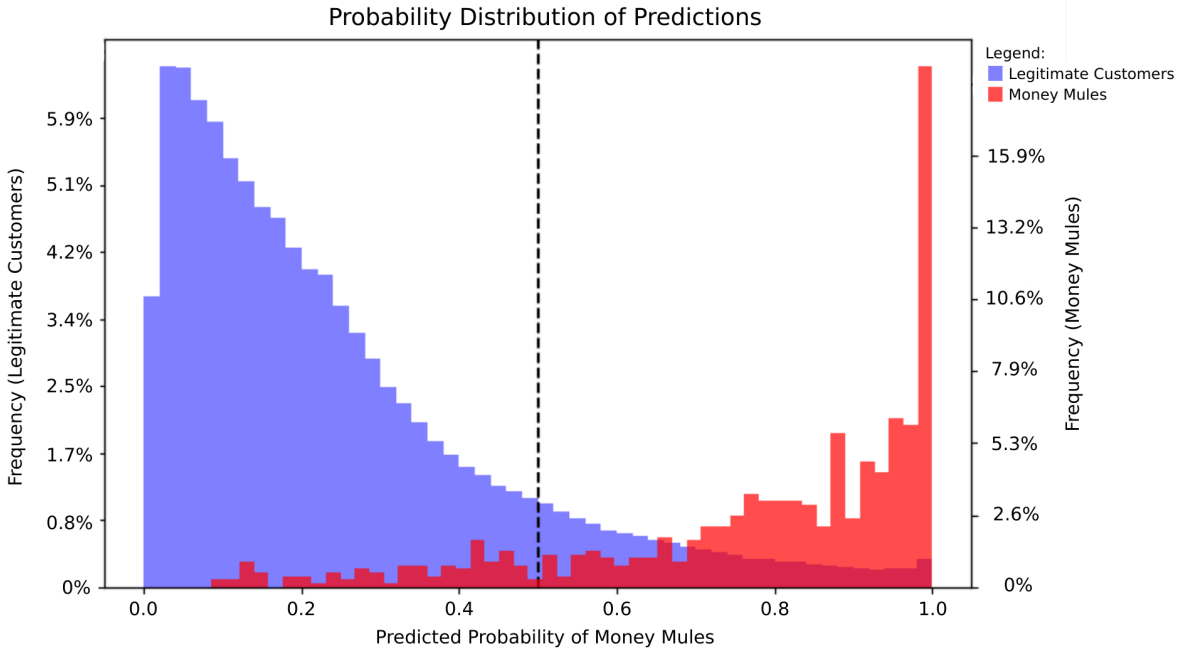



Figure 9. Predicted probability distribution for linear SVM with oversampling.

## 6.4 Decision Tree

Table 5 presents the performance of the decision tree classifier under different sampling strategies. As expected, the model trained on the imbalanced dataset without any sampling achieved a nearly perfect accuracy score (0.9987), but it completely failed to identify any money mules at the threshold of 0.5. This is reflected in a recall of 0, an AUROC of 0.5000 (equivalent to random guessing), and extremely low AUPRC values (0.0013 on the test set), confirming that the model was biased toward the legitimate customers.

The decision tree's performance improved when resampling techniques were applied. Undersampling led to the highest recall (0.8358), indicating strong sensitivity to the money mules, although this came at the cost of a higher false positive rate (15.7%). Oversampling offered a more balanced outcome, increasing both AUROC (0.8122) and test AUPRC (0.0440), with improved precision (0.0138) and F1 score (0.0271). The adjusted class weights approach got the highest precision (0.0168) and F1 score (0.0327), despite having the lowest recall (0.5483) among the resampling methods. This suggests that class weighting helped the model reduce false positives (4.1%), while capturing almost half of the money mules.

All undersampling, oversampling and adjusted weights techniques have slightly lower AUPRC scores on the test set compared to the validation set, which may suggest that the hyperparameter tuning process led to some degree of overfitting to the validation data. In terms of training time, the decision tree remained computationally efficient across all settings, with runtimes ranging from 29.3 to 37.1 seconds. Overall, while the Decision Tree model benefited from all balancing strategies, the trade-offs between precision and recall were more noticeable compared to other classifiers.

Table 5. Performance metrics for decision tree with different sampling techniques.

| **Metric** | **No sampling** | **Undersampling** | **Oversampling** | **Adjusted weights** |
|---|---|---|---|---|
| Time (s) | 31.8 | 29.3 | 37.1 | 35.9 |
| Accuracy | 0.9987 | 0.8430 | 0.9370 | 0.9585 |
| AUROC | 0.5000 | 0.8427 | 0.8122 | 0.7480 |
| AUPRC (val) | 0.0012 | 0.0299 | 0.0467 | 0.0423 |
| AUPRC (test) | 0.0013 | 0.0295 | **0.0440** | 0.0373 |
| Precision | - | 0.0068 | 0.0138 | 0.0168 |
| Recall | 0 | 0.8358 | 0.6861 | 0.5483 |
| F1 Score | - | 0.0134 | 0.0271 | 0.0327 |

| **Confusion Matrix** | | Predicted | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | No | Yes | No | Yes | No | Yes | No | Yes |
| Actual | No | 100% | 0% | 84.3% | 15.7% | 93.7% | 6.3% | 95.9% | 4.1% |
| | Yes | 100% | 0% | 16.4% | 83.6% | 31.4% | 68.6% | 45.2% | 54.8% |

Given its balance between precision and recall metrics and higher AUPRC, the decision tree model trained with oversampling was selected as the best-performing configuration. The AUPRC plot (Figure 10) shows that the model initially achieves perfect precision at very low recall but quickly drops to low precision levels as recall increases. This pattern suggests that the model correctly identifies a small number of positive cases with high confidence, but struggles to maintain precision when attempting to capture more true positives.

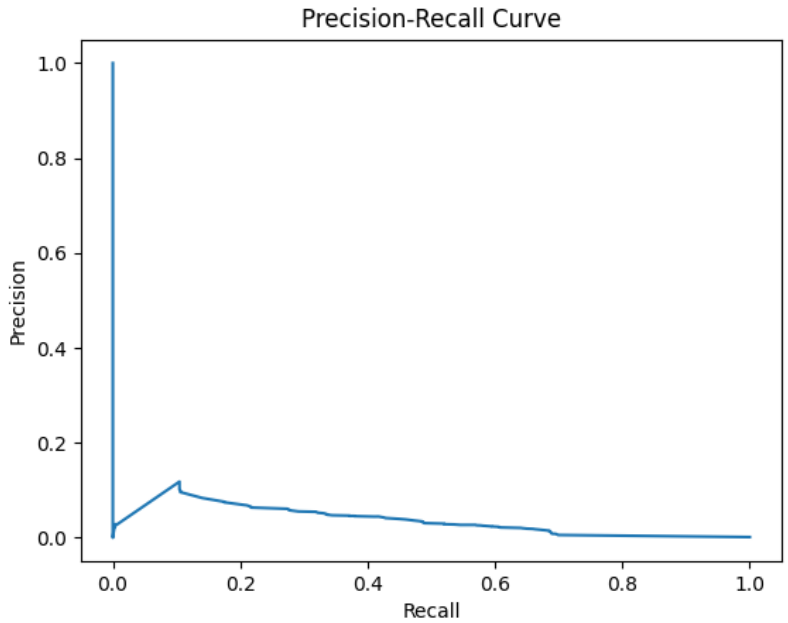



Figure 10. AUPRC plot for decision tree with oversampling.

On the probability distribution Figure 11) most predictions are close to either 0 or 1, indicating that the model is confident in its predictions. The distinct separation between classes can point to potential overfitting. The overlap in lower probability regions and the poor AUPRC suggest that the model lacks generalisation.

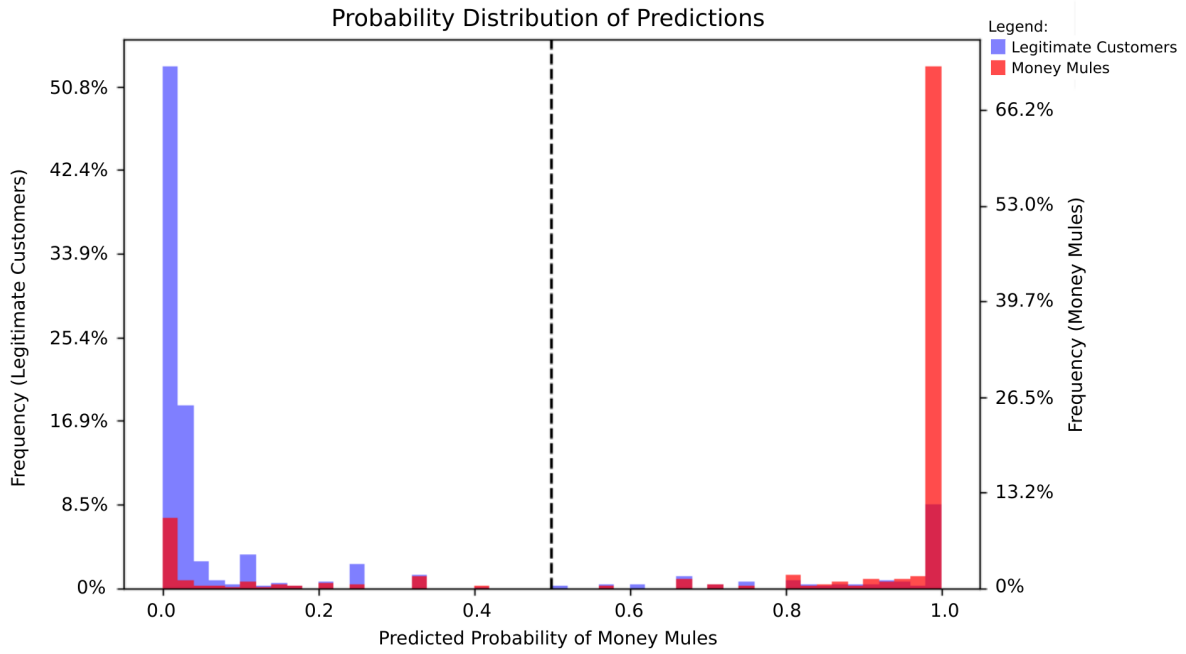



Figure 11. Predicted probability distribution for decision tree with oversampling.

## 6.5 Random Forest

Table 6 displays the performance metrics of the random forest classifier under various sampling strategies. Without any sampling, the model achieved the highest AUPRC on the test set (0.0966), which highlights its strong ranking ability. However, at the default threshold of 0.5, it failed to identify any money mules. This indicates that the classifier is biased toward the legitimate customer when trained on unbalanced data. This result implies that the model's performance could improve at lower thresholds.

Among the three balancing approaches, undersampling got the highest recall (0.8954), correctly identifying nearly 90% of money mules. However, this came at the cost of a relatively high false positive rate (11.5%), which significantly lowered precision (0.0098). Undersampling and oversampling both got lowest AUPRC (0.0649). Oversampling and adjusted class weights reduced false positives to below 1%, which may be a more manageable rate for banking staff. These strategies also improved precision (0.0566 and 0.0814, respectively). Of all four setups, the model with adjusted class weights achieved the highest F1 score (0.1318), indicating the

best balance between precision and recall, despite having the lowest recall among the resampled models (0.3457).

All models have slightly lower AUPRC scores on the test set compared to the validation set, which may suggest that the hyperparameter tuning process led to some degree of overfitting to the validation data. In terms of runtime, the random forest model was more computationally intensive than the other classifiers, particularly for the oversampling configuration (252 seconds). In contrast, undersampling significantly reduced training time (51.1 seconds).

Table 6. Performance metrics for random forest with different sampling techniques.

| **Metric** | | **No sampling** | | **Undersampling** | | **Oversampling** | | **Adjusted weights** | |
|---|---|---|---|---|---|---|---|---|---|
| Time (s) | | 264 | | 51.1 | | 252 | | 234 | |
| Accuracy | | 0.9987 | | 0.8848 | | 0.9903 | | 0.9942 | |
| AUROC | | 0.9539 | | 0.9541 | | 0.9203 | | 0.9241 | |
| AUPRC (val) | | 0.0974 | | 0.0675 | | 0.0748 | | 0.0745 | |
| AUPRC (test) | | **0.0966** | | 0.0649 | | 0.0649 | | 0.0727 | |
| Precision | | - | | 0.0098 | | 0.0566 | | 0.0814 | |
| Recall | | 0 | | 0.8954 | | 0.4199 | | 0.3457 | |
| F1 Score | | - | | 0.0195 | | 0.0998 | | 0.1318 | |
| **Confusion Matrix** | | Predicted | | | | | | | |
| | | No | Yes | No | Yes | No | Yes | No | Yes |
| Actual | No | 100% | 0% | 88.5% | 11.5% | 99.1% | 0.9% | 99.5% | 0.5% |
| | Yes | 100% | 0% | 10.5% | 89.5% | 58.0% | 42.0% | 65.4% | 34.6% |

Selecting the best model is challenging due to the severe class imbalance in the dataset. Since the threshold of 0.5 was used primarily as an example and can be adjusted later depending on stakeholder preferences, model selection was based on the highest AUPRC score on the test set. Under this criterion, the Random Forest model trained without any sampling was chosen as the best-performing model.

The AUPRC curve (Figure 12) shows relatively high precision at very low recall levels, followed by a gradual decline as recall increases. This pattern suggests that the model is able to identify a small number of true positive cases with reasonable confidence, but struggles to maintain precision as it attempts to retrieve more money mules. The sharp drop in precision reflects the challenge posed by the extreme class imbalance, where false positives quickly outnumber true positives.

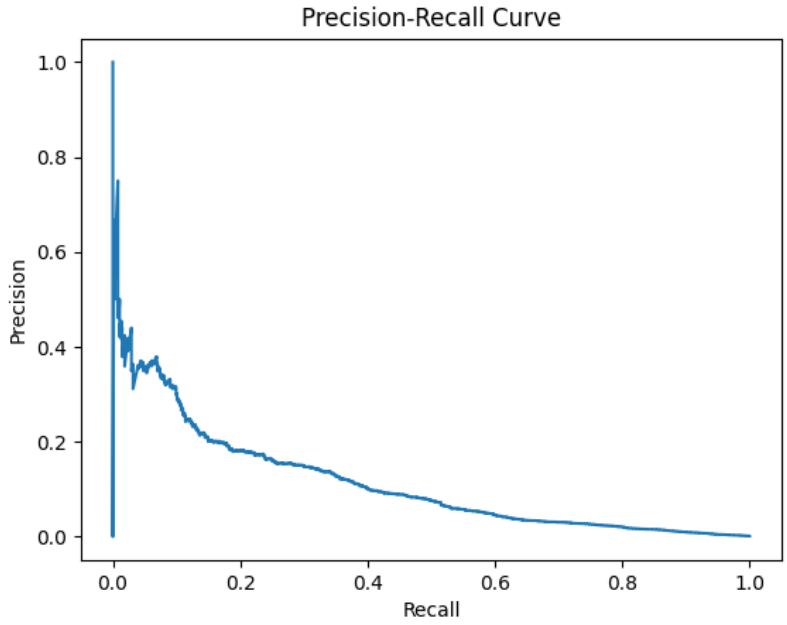



Figure 12. AUPRC plot for random forest without sampling.

The predicted probability distribution (Figure 13) indicates that the vast majority of predictions are concentrated near zero, with very few instances receiving a probability above 0.2. This suggests that while the model struggles to confidently assign high probabilities to the minority class, but based on the higher AUPRC it still produces a ranking that allows for useful discrimination at lower thresholds.

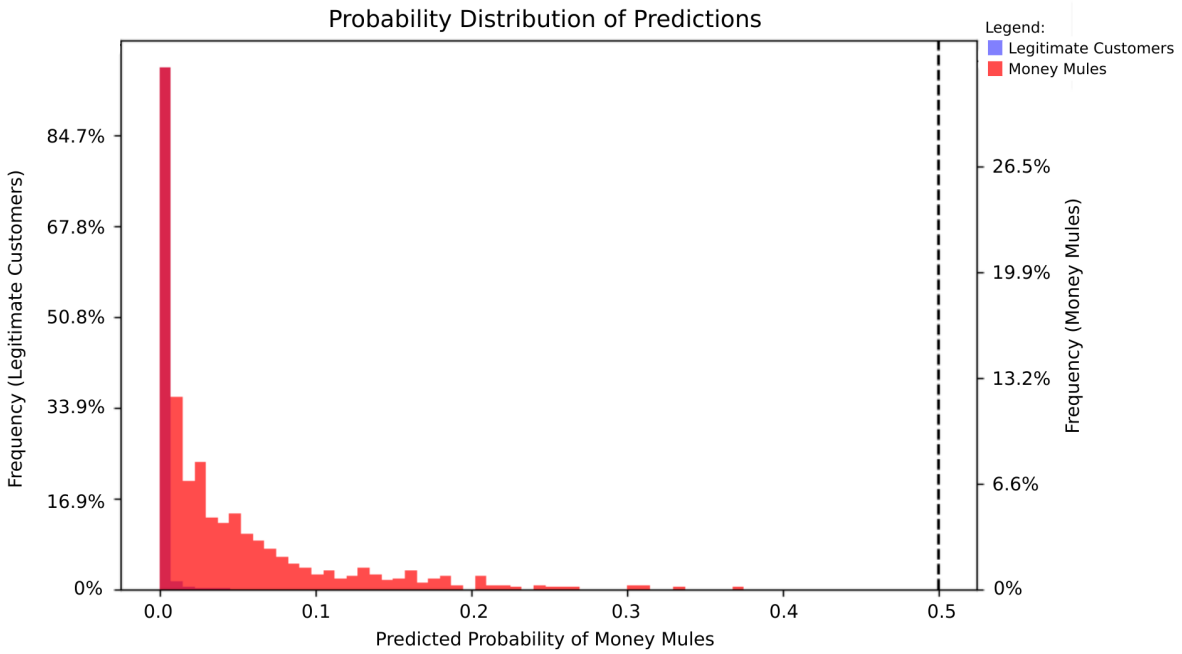



Figure 13. Predicted probability distribution for random forest without sampling.

## 6.6 Gradient Boosted Trees

Table 7 summarises the performance of the gradient boosted trees model across different sampling strategies. Without any balancing, the model produced high overall accuracy (0.9987) and an AUROC of 0.8519. However, these metrics are misleading in the context of severe class imbalance. At a standard decision threshold of 0.5, the model failed to identify any minority class instances, indicating a strong bias toward predicting legitimate (majority class) customers. Despite this, the AUPRC suggests that the model has some capacity to detect minority class cases at lower thresholds, which could be leveraged depending on stakeholder preferences regarding the trade-off between false positives and false negatives.

When balancing techniques were used, model performance improved considerably in identifying the minority class on the decision threshold of 0.5. The undersampling strategy resulted in the highest recall (0.8954) but also introduced a notable number of false positives (11.2%), leading to the lowest AUPRC on the test set (0.0609). Oversampling provided a more balanced outcome, achieving the highest F1 score (0.0361) and the best precision (0.0185), indicating that it enabled the model to identify minority instances more accurately while keeping false positives lower than others.

Interestingly, the adjusted weights approach got the highest recall after undersampling (0.8715), while also improving precision over undersampling. Although its F1 score (0.0246) was not the highest, it offered the best AUROC (0.9555) and the highest AUPRC on the test set (0.0814), making it a strong candidate for scenarios prioritizing both discrimination and ranking ability.

All models with sampling techniques have slightly lower AUPRC scores on the test set compared to the validation set, which may suggest that the hyperparameter tuning process led to some degree of overfitting to the validation data. Training times varied, with oversampling and adjusted weights requiring considerably more time (270 s and 192 s, respectively) compared to the default and undersampled versions. This suggests a trade-off between performance gains and computational efficiency, particularly when oversampling large datasets.

Table 7. Performance metrics for gradieent boosted trees with different sampling techniques.

| **Metric** | **No sampling** | **Undersampling** | **Oversampling** | **Adjusted weights** |
|---|---|---|---|---|
| Time (s) | 57.9 | 90 | 270 | 192 |
| Accuracy | 0.9987 | 0.8878 | 0.9475 | 0.9117 |
| AUROC | 0.8519 | 0.9542 | 0.9354 | 0.9555 |
| AUPRC (val) | 0.0738 | 0.0646 | 0.0851 | 0.0864 |
| AUPRC (test) | 0.0783 | 0.0609 | 0.0778 | **0.0814** |
| Precision | 0 | 0.0101 | 0.0185 | 0.0125 |
| Recall | 0 | 0.8954 | 0.7709 | 0.8715 |
| F1 Score | - | 0.0200 | 0.0361 | 0.0246 |

| **Confusion Matrix** | | Predicted | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | No | Yes | No | Yes | No | Yes | No | Yes |
| Actual | No | 100% | 0% | 88.8% | 11.2% | 94.8% | 5.2% | 91.2% | 8.8% |
| | Yes | 100% | 0% | 10.5% | 89.5% | 22.9% | 77.1% | 12.8% | 87.2% |

Oversampling achieved the best overall balance between precision and recall, resulting in the highest F1 score (0.0361), while also maintaining a strong AUROC and AUPRC. However, adjusted class weights offered the highest AUROC (0.9555) and test AUPRC (0.0814), indicating improved ranking performance and a better ability to assign meaningful probability scores to the minority class. Therefore, adjusted weights demonstrated the strongest overall scoring capability.

The Precision-Recall Curve in Figure 14 shows a steep drop in precision at very low recall values, reflecting the difficulty of identifying true positives in an imbalanced setting. However, the curve maintains moderately higher precision at low recall levels, contributing to the observed AUPRC score.

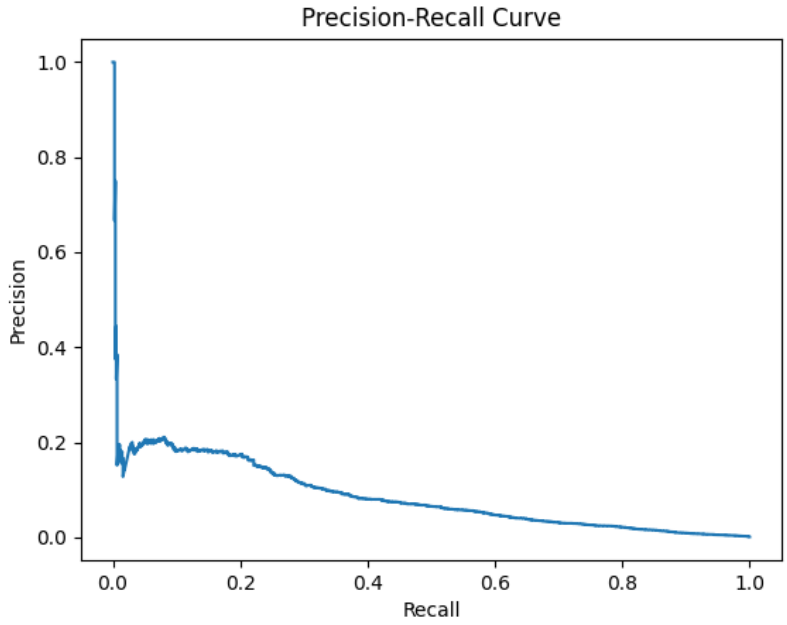



Figure 14. AUPRC plot for gradient boosted trees with adjusted weights.

Figure 15 illustrates the predicted probability distribution for both classes. The model assigns high probabilities close to 0 and 1 for the majority and minority classes, respectively, indicating good separation between classes and a well-calibrated probability output. This separation suggests that the model could perform well with a threshold optimized for the minority class, depending on stakeholder priorities.

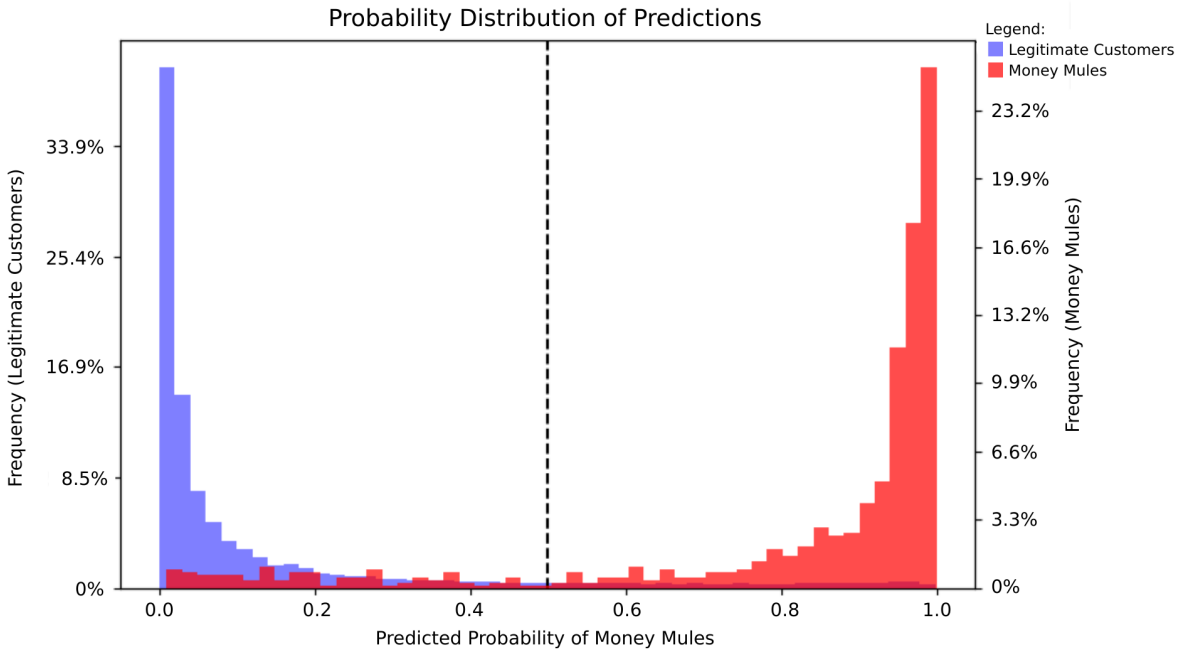



Figure 15. Predicted probability distribution for gradient boosted trees with adjusted weights.

## 6.7 XGBoost

Table 8 reports the performance metrics for the XGBoost model across different sampling methods. Without any sampling, the model achieved the highest accuracy (0.9988) and impressive AUROC (0.9626), but struggled to detect the minority class, managing a recall of only 4.1%. While the precision appeared high (0.4833), this was due to the low number of positive predictions, leading to a modest F1 score of 0.0759. This again demonstrates how strong global metrics can obscure poor minority class performance in imbalanced datasets.

When undersampling was applied, recall increased to 18.3% and the F1 score to 0.2066, with a slight trade-off in precision (0.2367). This setting got a solid AUROC (0.9637) and a relatively high AUPRC on the test set (0.1037), suggesting that XGBoost can still rank instances effectively under this strategy, even with fewer majority class examples.

The oversampling approach resulted in an improvement in recall (90.5%), as well as the highest AUPRC on the test set (0.1314), but at the cost of precision (0.0188). The F1 score was 0.0368, reflecting this imbalance between true and false positives. Nonetheless, the model was highly effective at recovering the positive class, which is desirable in high-risk scenarios because false negatives are costly.

With adjusted class weights, the model achieved the highest recall (91.4%), with slightly lower precision than oversampling, leading to the lowest F1 score (0.0208) among the balanced settings. This technique got a solid AUROC (0.9650) but a lower AUPRC (0.0821), suggesting that although positive examples were identified, they were not ranked as well compared to oversampling.

In terms of efficiency, all models run less than two minutes. The undersampling was the fastest (19 s), while oversampling required the most time (90 s). Interestingly, all XGBoost configurations performed significantly better on the test data than on the validation data in terms of AUPRC. This may indicate that the test set was particularly well aligned with the patterns learned during training, or that the model was able to generalise better than expected. Although some degree of random variation in small minority-class samples cannot be ruled out.

Table 8. Performance metrics for XGBoost with different sampling techniques.

| **Metric** | **No sampling** | **Undersampling** | **Oversampling** | **Adjusted weights** |
|---|---|---|---|---|
| Time (s) | 60 | 19 | 90 | 50 |
| Accuracy | 0.9988 | 0.9983 | 0.9430 | 0.8986 |
| AUROC | 0.9626 | 0.9637 | 0.9759 | 0.9650 |
| AUPRC (val) | 0.1004 | 0.0934 | 0.0895 | 0.0596 |
| AUPRC (test) | 0.1065 | 0.1037 | **0.1314** | 0.0821 |
| Precision | 0.4833 | 0.2367 | 0.0188 | 0.0105 |
| Recall | 0.0412 | 0.1832 | 0.9046 | 0.9140 |
| F1 Score | 0.0759 | 0.2066 | 0.0368 | 0.0208 |

| **Confusion Matrix** | | Predicted | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | No | Yes | No | Yes | No | Yes | No | Yes |
| Actual | No | 100% | 0% | 99.9% | 0.1% | 94.3% | 5.7% | 89.9% | 10.1% |
| | Yes | 95.9% | 4.1% | 81.7% | 18.3% | 9.5% | 90.5% | 8.6% | 91.4% |

Among the evaluated sampling strategies for XGBoost, oversampling demonstrated the most balanced and effective performance for minority class detection. It achieved the highest recall (90.5%) and AUPRC on the test set (0.1314), indicating both strong sensitivity to positive cases and good ranking capability. Although the precision (0.0188) remained low. These findings suggest that oversampling was the most suitable approach for addressing class imbalance in this context.

The Precision-Recall Curve in Figure 16 shows relatively high precision at the initial stages of recall, followed by a gradual decline as more true positives are retrieved. This indicates that the model is capable of identifying some money mule instances with reasonable confidence, although performance diminishes as recall increases.

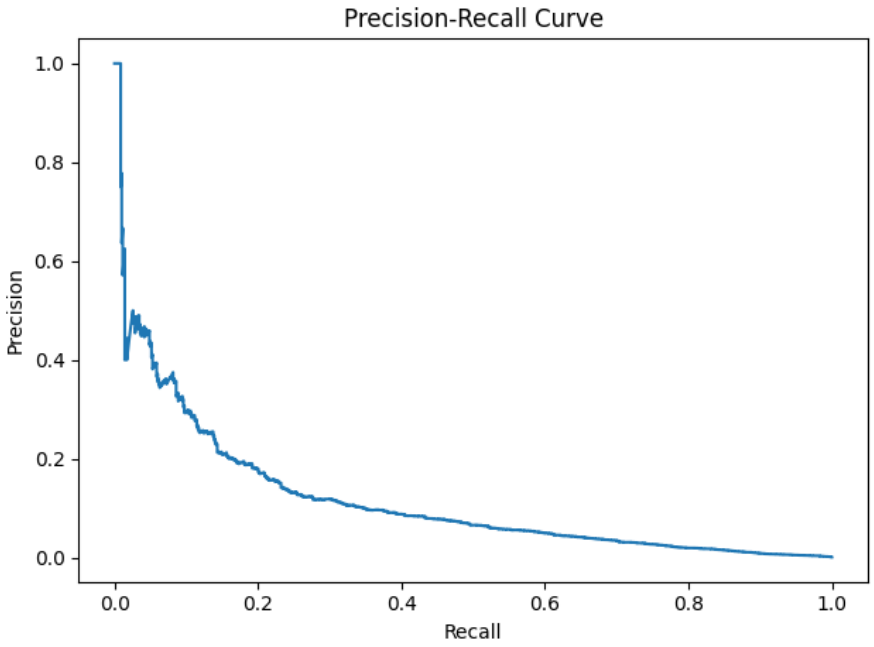



Figure 16. AUPRC plot for XGBoost with oversampling.

The probability distribution in Figure 17 shows a separation between predicted probabilities for the two classes, with legitimate customers predictions clustered near 0 and money mules near 1. This separation suggests that the model assigns probabilities with a confidence, and that oversampling has contributed to more balanced and distinguishable class predictions.

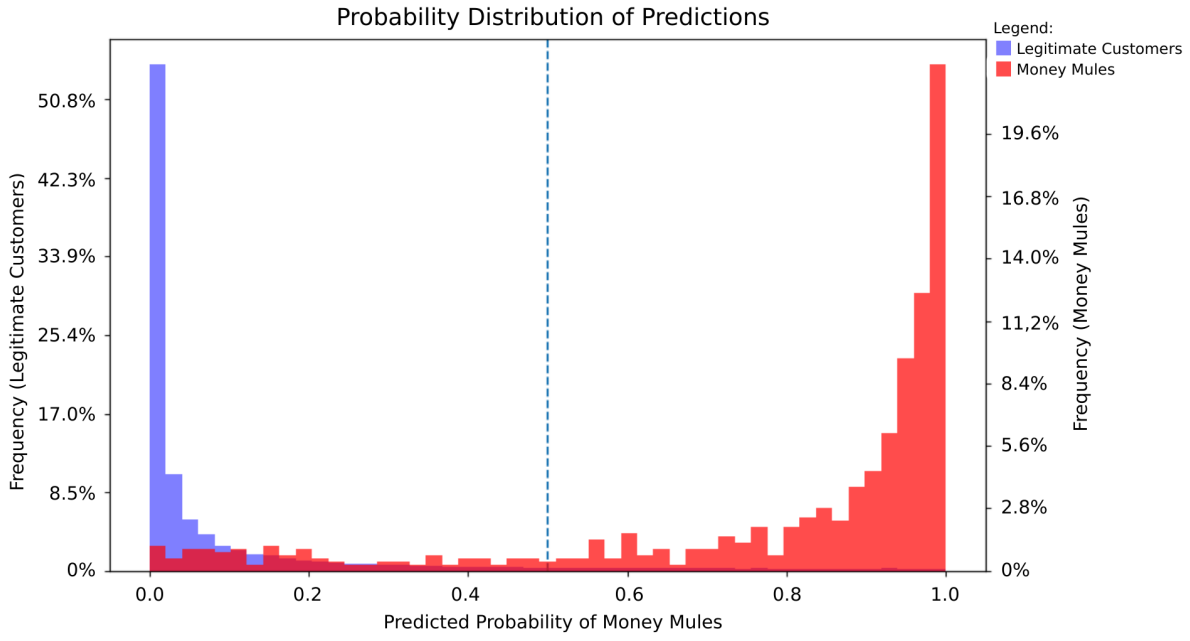



Figure 17. Predicted probability distribution for XGBoost with oversampling.

## 6.8 CatBoost

Table 9 presents the evaluation results for the CatBoost classifier under different sampling strategies. Without any sampling, the model achieved the best AUPRC on test set (0.1229), high accuracy (0.9988) and the highest AUROC (0.9608) among all configurations. However, this came at the cost of poor recall (3.5%) and a low F1 score (0.0644), indicating that the model largely failed to identify money mules, despite strong global metrics. The precision was misleadingly high (0.4807), again due to the model predicting very few positive cases of which most were correct. Furthermore, the distribution of predicted probabilities for this model was heavily skewed toward the lower end, suggesting that a much lower decision threshold would have resulted in more true positives. This implies that while the model ranks examples well, it needs careful threshold tuning.

Applying undersampling improved recall to 90.5%, the highest among all CatBoost variants. However, this came with a drop in precision (0.0101), resulting in a low F1 score (0.0200). While the model could detect money mules more effectively, the AUPRC score slightly decreased to 0.0999, suggesting weaker confidence in ranking predictions. The oversampling strategy achieved a good balance, with a recall of 76.0% and the highest F1 score (0.0447) among all setups. This configuration also provided a good AUPRC on the test set (0.1138), implying better trade-offs between true and false positives than undersampling or weighted training.

Adjusted class weights led to a recall of 79.2%, close to the oversampling result, but with a slightly lower F1 score (0.0390). The AUPRC (0.1141) was better than with oversampling. This indicates good ranking performance, though the classification quality was somewhat lower due to lower precision.

In terms of training time, undersampling was the fastest at 126 seconds, while oversampling was the most computationally expensive (864 seconds). Overall, CatBoost demonstrated good ranking capability under all configurations.

Table 9. Performance metrics for CatBoost with different sampling techniques.

| **Metric** | **No sampling** | **Undersampling** | **Oversampling** | **Adjusted weights** |
|---|---|---|---|---|
| Time (s) | 366 | 126 | 864 | 396 |
| Accuracy | 0.9988 | 0.8910 | 0.9601 | 0.9521 |
| AUROC | 0.9608 | 0.9604 | 0.9474 | 0.9510 |
| AUPRC (val) | 0.1132 | 0.0881 | 0.1050 | 0.1087 |
| AUPRC (test) | **0.1229** | 0.0999 | 0.1138 | 0.1141 |
| Precision | 0.4807 | 0.0101 | 0.0230 | 0.0200 |
| Recall | 0.0345 | 0.9048 | 0.7600 | 0.7917 |
| F1 Score | 0.0644 | 0.0200 | 0.0447 | 0.0390 |

| **Confusion Matrix** | | Predicted | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | No | Yes | No | Yes | No | Yes | No | Yes |
| Actual | No | 100% | 0% | 89.1% | 10.9% | 96.0% | 4.0% | 95.2% | 4.8% |
| | Yes | 96.6% | 3.4% | 9.5% | 90.5% | 24.0% | 76.0% | 20.8% | 79.2% |

Selecting the best model is challenging due to the severe class imbalance in the dataset. Since the threshold of 0.5 was used primarily as an example and can be adjusted later depending on stakeholder preferences, model selection was based on the highest AUPRC score on the test set. Under this criterion, the Catboost model trained without any sampling was chosen as the best-performing model.

The Precision-Recall Curve in Figure 18 shows strong precision at very low recall levels, followed by a gradual decrease as recall increases. This suggests that the model is able to correctly identify a limited number of positive instances with relatively high confidence, but its precision diminishes as it attempts to capture more true positives.

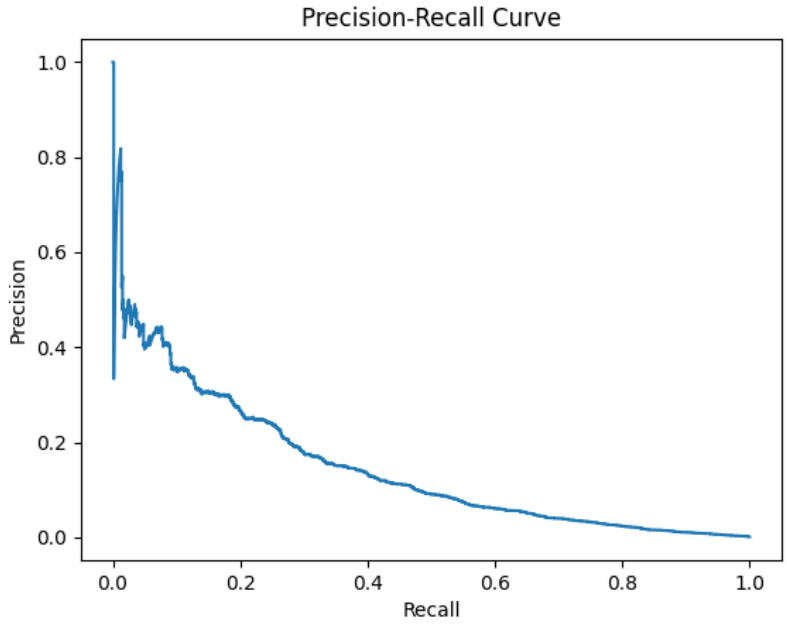



Figure 18. AUPRC plot for CatBoost without sampling.

The predicted probability distribution in Figure 19 reveals a skew toward low predicted probabilities, with most samples clustered near zero. Very few instances are assigned probabilities above 0.5, indicating that the model is conservative in predicting money mules. This distribution reflects the model's tendency to favour the legitimate customers in an imbalanced setting, though it still retains some ranking ability useful for threshold tuning.

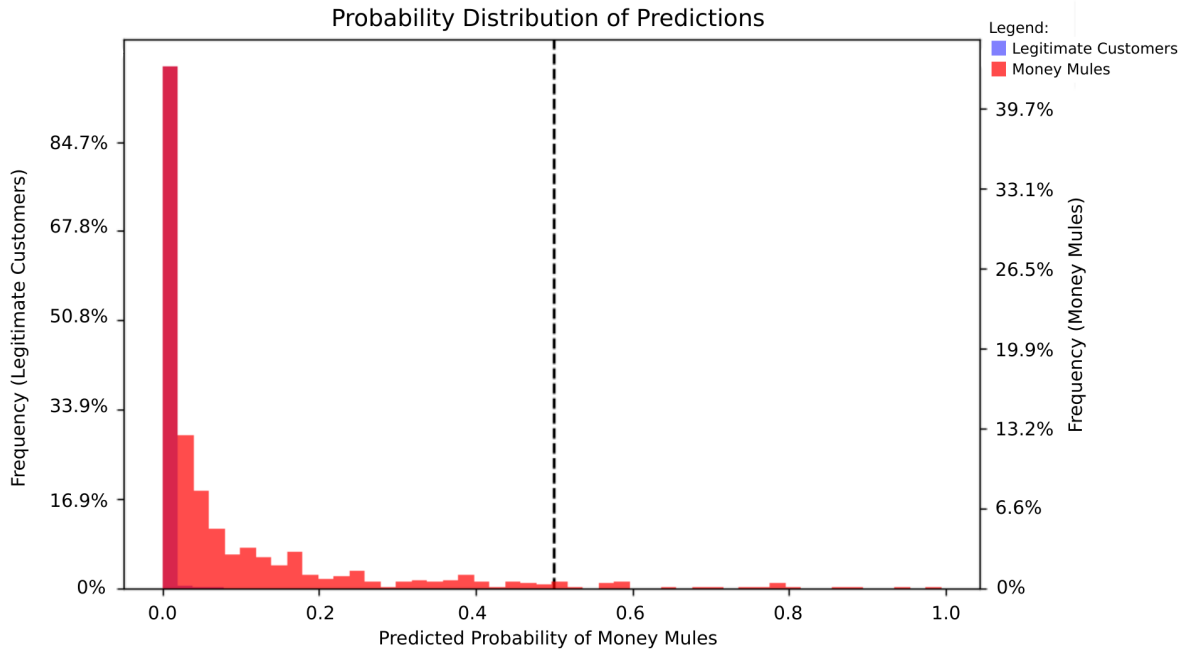



Figure 19. Predicted probability distribution for CatBoost without sampling.

## 6.9 LightGBM

Table 10 presents the performance metrics for the LightGBM classifier under various sampling strategies. The model trained without any sampling achieved high accuracy (0.9987), but at the cost of an almost non-existent recall (0.0013%), indicating that it struggled to detect money mules. Precision was low (0.3333), but this result can be attributed to the fact that very few positive cases were predicted. It also got the highest AUPRC on test data (0.1137) showing that the model ranked positive cases well, but needed lower threshold to predict money mules. While the model's global metrics were good, it performed poorly on threshold of 0.5 in identifying money mules without any sampling technique.

Undersampling boosted recall to 90.9%, the highest among all LightGBM variants. However, this came with a decrease in precision (0.0103), leading to a low F1 score of 0.0204. This suggests that undersampling allowed the model to identify most of the positive cases, but at the expense of false positives, reducing its precision. The AUPRC score on the test set was 0.0828, slightly lower than the no-sampling model, indicating that while the model was better at detecting money mules, it wasn't able to rank positive cases as confidently.

Oversampling provided a balance between recall and precision, with a recall of 78.3% and precision of 0.0266. The F1 score (0.0515) improved compared to undersampling, but was still lower than the no-sampling model. The AUPRC on the test set was 0.0996, slightly better than undersampling, but still showing room for improvement. This suggests that while oversampling improved the model's ability to identify positive cases, it did not outperform undersampling in terms of ranking performance.

Adjusted class weights resulted in a recall of 72.2%, lower than oversampling but higher than the no-sampling model. The precision was slightly higher than with oversampling (0.0323), leading to a better F1 score (0.0618). The AUPRC score was 0.1092, just slightly lower than the no-sampling approach, indicating competitive performance when it comes to ranking positive cases.

Models with undersampling and adjusted weights have slightly lower AUPRC scores on the test set compared to the validation set, which may suggest that the hyperparameter tuning process led to some degree of overfitting to the validation data. In terms of computational efficiency, undersampling was the fastest at 66 seconds, while adjusted weights were the most computationally expensive (210 seconds).

Table 10. Performance metrics for LightGBM with different sampling techniques.

| **Metric** | **No sampling** | **Undersampling** | **Oversampling** | **Adjusted weights** |
|---|---|---|---|---|
| Time (s) | 90 | 66 | 144 | 210 |
| Accuracy | 0.9987 | 0.8881 | 0.9632 | 0.9720 |
| AUROC | 0.9585 | 0.9587 | 0.9572 | 0.9593 |
| AUPRC (val) | 0.1131 | 0.0883 | 0.0962 | 0.1113 |
| AUPRC (test) | **0.1137** | 0.0828 | 0.0996 | 0.1092 |
| Precision | 0.3333 | 0.0103 | 0.0266 | 0.0323 |
| Recall | 0.0013 | 0.9099 | 0.7828 | 0.7219 |
| F1 Score | 0.0026 | 0.0204 | 0.0515 | 0.0618 |

| **Confusion Matrix** | | Predicted | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | No | Yes | No | Yes | No | Yes | No | Yes |
| Actual | No | 100% | 0% | 88.8% | 11.2% | 96.3% | 3.7% | 97.2% | 2.8% |
| | Yes | 99.9% | 0.1% | 9.0% | 91.0% | 21.7% | 78.3% | 27.8% | 72.2% |

Selecting the best model is challenging due to the severe class imbalance in the dataset. Since the threshold of 0.5 was used primarily as an example and can be adjusted later depending on stakeholder preferences, model selection was based on the highest AUPRC score on the test set. Under this criterion, the LightGBM model trained without any sampling was chosen as the best-performing model.

The Precision-Recall Curve (Figure 20) indicates that the model maintains relatively high precision at very low recall values, but performance becomes progressively worse as recall increases. This suggests that while the model can identify a small number of true positives with confidence, it quickly begins to misclassify as it attempts to capture more positive instances.

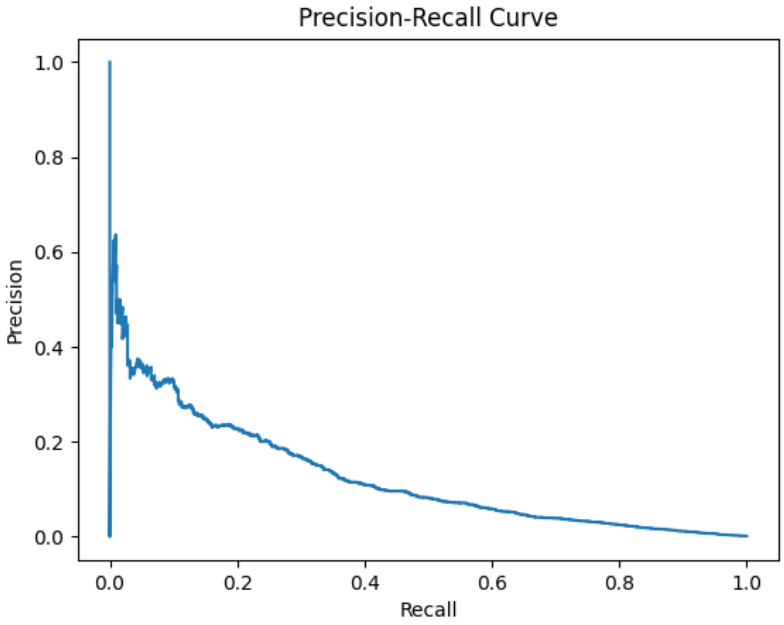



Figure 20. AUPRC plot for LightGBM without sampling.

The predicted probability distribution (Figure 21) confirms this behaviour, showing that most predicted probabilities are heavily concentrated near zero. The right tail, representing higher predicted probabilities for the money mules, is an indication that the model is cautious and tends to assign low probabilities to positive class instances. This conservative behaviour reflects the influence of class imbalance but still supports the model's usefulness in ranking tasks, particularly when paired with threshold tuning.

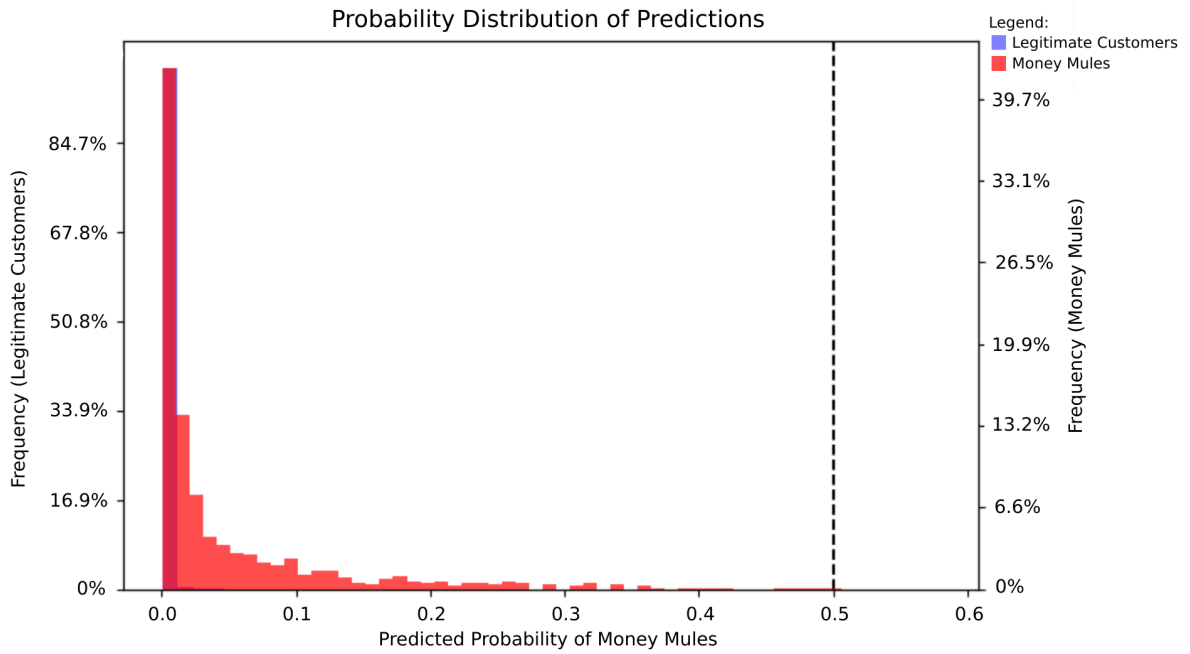



Figure 21. Predicted probability distribution for LightGBM without sampling.

## 6.10 Isolation Forest

Table 11 shows the evaluation of the Isolation Forest model under three different sampling strategies. With Isolation forest it is important to keep in mind that it outputs anomaly score instead of a probability or rawPrediction column like classifiers do. Do normalise the scores Min-Max scailing was used.

Without any sampling, the model achieved moderate recall (38.5%) and a low precision (0.0079), resulting in a low F1 score (0.0154). Despite this, it got a relatively high AUROC (0.8095). AUPRC scores on the test data (0.0099) is slightly lower compared to the validation data (0.0129), which may suggest that the hyperparameter tuning process led to some degree of overfitting to the validation data.

Undersampling significantly improved recall to 72.2%, but precision dropped even further to 0.0044, causing the F1 score to fall to 0.0087. This configuration also had the highest recall and AUROC (0.8329) among all variants, indicating the model became more sensitive to detecting anomalies but at the cost of a large number of false positives. The AUPRC on the test set (0.0125) was the highest of the three configurations, reflecting better ranking performance under this setup.

Oversampling, on the other hand, got the lowest recall (6.6%) and a modest precision (0.0126), resulting in the highest F1 score (0.0212). While this version showed the best precision, it missed nearly all positive cases, that indicates it is ineffective for anomaly detection. Also the test data AUPRC and AUROC values were lower than in undersampling version.

While none of the sampling methods resulted in strong classification performance with Isolation Forest, undersampling achied the highest AUPRC on the test set (0.0125). It was also the most computationally efficient, taking almost one minute less than others. These results suggest that although Isolation Forest is fast and unsupervised, it may not be suitable as a stand-alone solution for money mule detection in highly imbalanced datasets. However, it could still play a valuable role as an anomaly detection component within a hybrid model setup, complementing supervised methods with its ability to flag rare, outlying behaviour.

Table 11. Performance metrics for Isolation Forest with different sampling techniques.

| **Metric** | | **No sampling** | | **Undersampling** | | **Oversampling** | |
|---|---|---|---|---|---|---|---|
| Time (s) | | 120 | | 60 | | 135 | |
| Accuracy | | 0.9418 | | 0.8046 | | 0.9928 | |
| AUROC | | 0.8095 | | 0.8329 | | 0.7208 | |
| AUPRC (val) | | 0.0129 | | 0.0042 | | 0.0043 | |
| AUPRC (test) | | 0.0099 | | **0.0125** | | 0.0085 | |
| Precision | | 0.0079 | | 0.0044 | | 0.0126 | |
| Recall | | 0.3854 | | 0.7221 | | 0.0659 | |
| F1 Score | | 0.0154 | | 0.0087 | | 0.0212 | |
| **Confusion Matrix** | | Predicted | | | | | |
| | | No | Yes | No | Yes | No | Yes |
| Actual | No | 94.2% | 5.8% | 80.5% | 19.5% | 99.4% | 0.6% |
| | Yes | 61.5% | 38.5% | 27.8% | 72.2% | 93.4% | 6.6% |

The Precision-Recall Curve in Figure 22 shows low precision across almost all recall levels, indicating a poor ability to correctly identify money mules. This is also supported by the flat shape of the curve, suggesting that the model struggles to distinguish between the two classes regardless of threshold.

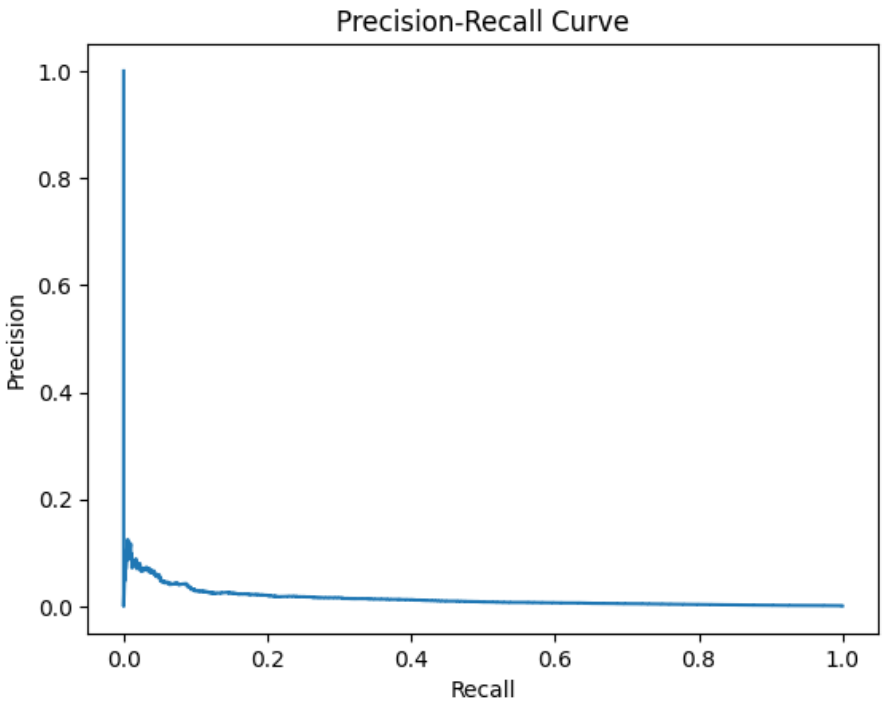



Figure 22. AUPRC plot for Isolation Forest with undersampling.

The predicted probability distribution in Figure 23 shows overlap between the predicted probabilities for both classes. Unlike more discriminative models, the scores here are spread across the probability range without clear separation, suggesting weak signal extraction from the input features.

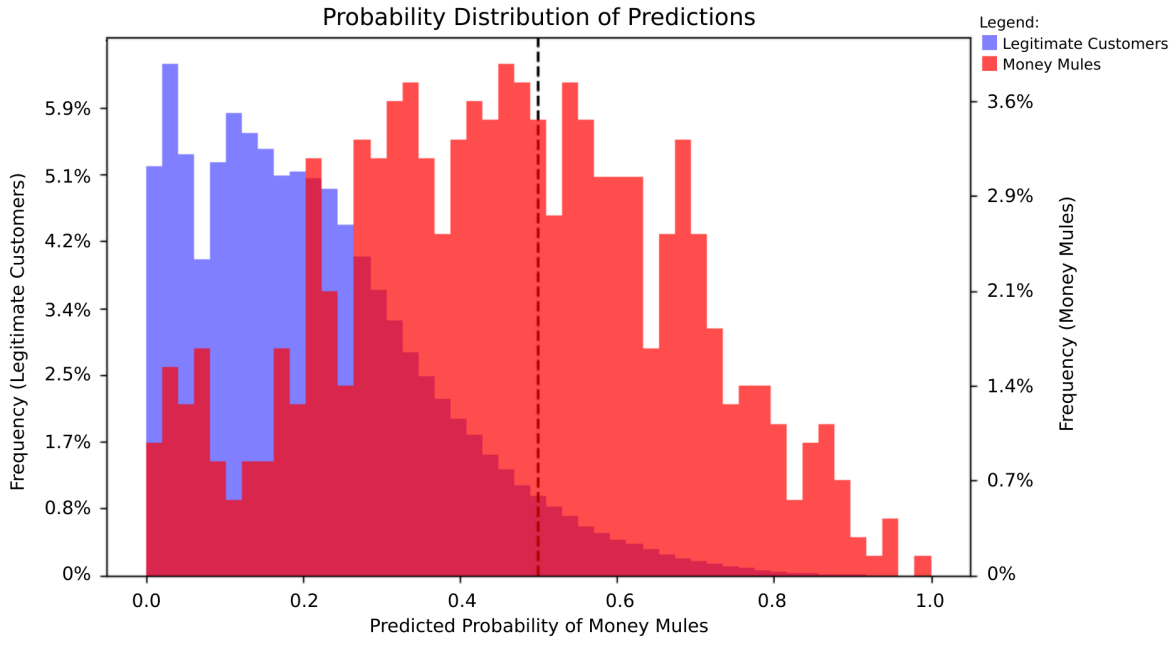



Figure 23. Predicted probability distribution for Isolation Forest with undersampling.

The predicted probability distribution in Figure 23 shows overlap between the predicted probabilities for both classes. Unlike more discriminative models, the scores here are spread across the probability range without clear separation, suggesting weak signal extraction from the input features.

## 6.11 Weighted and Rule-Based Stacked Ensemble Models

To explore the possibilities even further, it was decided to create an ensemble model. Combining prediction scores from multiple models can improve performance by leveraging their complementary strengths and reducing individual model biases. The goal was to select one model that is better in detecting money mules (meaning it prioritises high recall) and another that is better at identifying legitimate customers (meaning it prioritises high precision). So the models were chosen from the five models with the highest recall (Table 12) and the five with the highest precision (Table 13).

Table 12. Top 5 models based on recall.

| **Metric** | | **XGBoost (weights)** | | **LGBM (under)** | | **CatBoost (under)** | | **XGBoost (over)** | | **GBT (under)** | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Time (s) | | 50 | | 66 | | 126 | | 90 | | 90 | |
| Accuracy | | 0.8986 | | 0.8881 | | 0.8910 | | 0.9430 | | 0.8878 | |
| AUC ROC | | 0.9650 | | 0.9587 | | 0.9604 | | 0.9759 | | 0.9542 | |
| AUPRC (val) | | 0.0596 | | 0.0883 | | 0.0881 | | 0.0895 | | 0.0646 | |
| AUPRC (test) | | 0.0822 | | 0.0828 | | 0.0999 | | 0.1314 | | 0.0609 | |
| Precision | | 0.0105 | | 0.0103 | | 0.0101 | | 0.0188 | | 0.0101 | |
| **Recall** | | **0.9140** | | **0.9099** | | **0.9048** | | **0.9046** | | **0.8954** | |
| F1 Score | | 0.0208 | | 0.0204 | | 0.0200 | | 0.0368 | | 0.0200 | |
| **Confusion Matrix** | | Predicted | | | | | | | | | |
| | | No | Yes | No | Yes | No | Yes | No | Yes | No | Yes |
| Actual | No | 89.9% | 10.1% | 88.8% | 11.2% | 89.1% | 10.9% | 94.3% | 5.7% | 88.5% | 11.5% |
| | Yes | 8.6% | 91.4% | 9.0% | 91.0% | 9.5% | 90.5% | 9.5% | 90.5% | 10.5% | 89.5% |

From the top 5 models based on recall, LightGBM with undersampling and XGBoost with oversampling were selected for the ensemble to represent the high-recall models. Even though XGBoost with the adjusted weights had better recall performance (0.9140), XGBoost with oversampling version was chosen because because it had highest AUPRC (0.1314). CatBoost was excluded because converting the Spark DataFrame to Pandas significantly increased runtime and caused part of the minority class to be lost, likely due to memory or format-handling issues. So this meant that LightGBM with undersampling had both second best recall (0.9099) and second best AUPRC (0.0828).

Table 13. Top 5 models based on precision.

| **Metric** | | **XGBoost (none)** | | **CatBoost (none)** | | **LGBM (none)** | | **XGBoost (under)** | | **RF (weights)** | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Time (s) | | 60 | | 366 | | 90 | | 19 | | 234 | |
| Accuracy | | 0.9988 | | 0.9988 | | 0.9987 | | 0.9983 | | 0.9942 | |
| AUC ROC | | 0.9626 | | 0.9608 | | 0.9585 | | 0.9637 | | 0.9241 | |
| AUPRC (val) | | 0.1004 | | 0.1132 | | 0.1131 | | 0.0934 | | 0.0745 | |
| AUPRC (test) | | 0.1065 | | 0.1229 | | 0.1137 | | 0.1037 | | 0.0727 | |
| **Precision** | | **0.4833** | | **0.4808** | | **0.3333** | | **0.2367** | | **0.0814** | |
| Recall | | 0.0412 | | 0.0345 | | 0.0013 | | 0.1832 | | 0.3457 | |
| F1 Score | | 0.0759 | | 0.0644 | | 0.0026 | | 0.2066 | | 0.1318 | |
| **Confusion Matrix** | | Predicted | | | | | | | | | |
| | | No | Yes | No | Yes | No | Yes | No | Yes | No | Yes |
| Actual | No | 100.0% | 0.0% | 100.0% | 0.0% | 100.0% | 0.0% | 99.9% | 0.1% | 99.5% | 0.5% |
| | Yes | 95.9% | 4.1% | 96.6% | 3.4% | 99.9% | 0.1% | 81.7% | 18.3% | 65.4% | 34.6% |

In terms of precision, the XGBoost model without sampling and the random forest with adjusted weights were selected for the ensemble to represent the more high-precision model. The XGBoost model without sampling had the best precision (0.4833). Once again CatBoost was dismissed. This made the XGBoost model without sampling with second best AUPRC (0.1065). The Random Forest model with adjusted weights was included despite its slightly lower precision, as it achieved a higher recall than others. Although LightGBM with no sampling delivered better AUPRC compared to XGBoost without sampling, the latter's better recall performance justified its selection given the emphasis on both precision and recall.

Both weighted ensemble and rule-based ensemble approaches were explored. This meant eight possible combinations:

1. LightGBM with undersampling + XGBoost without sampling + weights approach
2. LightGBM with undersampling + XGBoost without sampling + rule approach
3. LightGBM with undersampling + random forest with adjusted weights + weights approach
4. LightGBM with undersampling + random forest with adjusted weights + rule approach
5. XGBoost with oversampling + XGBoost without sampling + weights approach
6. XGBoost with oversampling + XGBoost without sampling + rule approach
7. XGBoost with oversampling + random forest with adjusted weights + weights approach
8. XGBoost with oversampling + random forest with adjusted weights + rule approach

With eight different model combinations, 11 weight or rule values, and 11 thresholds, this resulted in a total of 968 unique configurations. For each model combination, the configuration that got the highest AUPRC is reported. The AUPRC metric captures the trade-off between precision and recall across all thresholds, making it especially suitable for this task. Unlike the evaluation of individual models, which relied on fixed threshold of 0.5, this approach allows for a more flexible and better comparison.

The highest AUPRC achieved was 0.1222. This result was obtained using the weighted ensemble approach, where the high-recall model (XGBoost with oversampling) contributed 20% to the final prediction, and the high-precision (XGBoost without sampling) contributed 80%. Performance metrics across different thresholds are shown in Table 14, with thresholds ranging from 0 to 1.

In the banking domain, it is generally more important to maintain high precision, even if it results in lower recall. While identifying a higher number of money mules is beneficial, it is essential to limit false positives as they can impose a significant strain on investigative resources. This concern is particularly relevant in the context of a highly imbalanced dataset. For example, here a false positive rate of 7.0% would already lead to an impractical volume of manual investigations. Among the tested thresholds, a value of 0.2 provides the most balanced result. It identifies slightly more than half of the money mules while keeping the false positive rate below 1%, which is a more manageable and realistic outcome for deployment.

Table 14. Model performance and confusion matrix percentages at different thresholds for XGBoost with oversampling, XGBoost without sampling and weighted ensemble approach.

| Threshold | Precision | Recall | F1 score | TP% | FP% | TN% | FN% |
|---|---|---|---|---|---|---|---|
| **0.0** | 0.0012 | 1.0000 | 0.0023 | 100.0% | 100.0% | 0.0% | 0.0% |
| **0.1** | 0.0140 | 0.8494 | 0.0275 | 84.9% | 7.0% | 93.0% | 15.1% |
| **0.2** | 0.0682 | 0.5088 | 0.1203 | 50.9% | 0.8% | 99.2% | 49.1% |
| **0.3** | 0.2575 | 0.2266 | 0.2411 | 22.7% | 0.1% | 99.9% | 77.3% |
| **0.4** | 0.3775 | 0.1126 | 0.1734 | 11.3% | 0.0% | 100.0% | 88.7% |
| **0.5** | 0.4800 | 0.0702 | 0.1224 | 7.0% | 0.0% | 100.0% | 93.0% |
| **0.6** | 0.5745 | 0.0395 | 0.0739 | 3.9% | 0.0% | 100.0% | 96.1% |
| **0.7** | 0.4783 | 0.0161 | 0.0311 | 1.6% | 0.0% | 100.0% | 98.4% |
| **0.8** | 0.6667 | 0.0058 | 0.0116 | 0.6% | 0.0% | 100.0% | 99.4% |
| **0.9** | 1.0000 | 0.0044 | 0.0087 | 0.4% | 0.0% | 100.0% | 99.6% |
| **1.0** | 0.0000 | 0.0000 | – | 0.0% | 0.0% | 100.0% | 100.0% |

The second-highest AUPRC was 0.1219, achieved using the rule-based ensemble approach, where the high-recall model (XGBoost with oversampling) contributed 0% and the high-precision model (XGBoost without sampling) contributed 100% to the final prediction. This corresponds to the stand-alone performance of XGBoost without sampling. This result highlights that increased model complexity does not always lead to better performance. The corresponding performance metrics across various thresholds are provided in Table 15, with the best results observed at a threshold of 0.1. At this threshold, 27.3% of money mules were correctly identified, while the false positive rate remained as low as 0.1%, which is considered operationally manageable.

It is important to note that this AUPRC score (0.1219) differs slightly from the result reported for XGBoost with undersampling in Section 6.7 (0.1065). Although the same hyperparameters were used, the models were retrained for the ensemble approach, and minor variation in results can occur due to inherent randomness in the training process.

Table 15. Model performance and confusion matrix percentages at different thresholds for XGBoost without sampling.

| Threshold | Precision | Recall | F1 score | TP% | FP% | TN% | FN% |
|---|---|---|---|---|---|---|---|
| **0.0** | 0.0012 | 1.0000 | 0.0023 | 100.0% | 100.0% | 0.0% | 0.0% |
| **0.1** | 0.2130 | 0.2734 | 0.2394 | 27.3% | 0.1% | 99.9% | 72.7% |
| **0.2** | 0.3386 | 0.1579 | 0.2154 | 15.8% | 0.0% | 100.0% | 84.2% |
| **0.3** | 0.3896 | 0.0877 | 0.1432 | 8.8% | 0.0% | 100.0% | 91.2% |
| **0.4** | 0.5172 | 0.0658 | 0.1167 | 6.6% | 0.0% | 100.0% | 93.4% |
| **0.5** | 0.5745 | 0.0395 | 0.0739 | 3.9% | 0.0% | 100.0% | 96.1% |
| **0.6** | 0.5714 | 0.0234 | 0.0449 | 2.3% | 0.0% | 100.0% | 97.7% |
| **0.7** | 0.7000 | 0.0102 | 0.0202 | 1.0% | 0.0% | 100.0% | 99.0% |
| **0.8** | 0.6000 | 0.0044 | 0.0087 | 0.4% | 0.0% | 100.0% | 99.6% |
| **0.9** | 1.0000 | 0.0029 | 0.0058 | 0.3% | 0.0% | 100.0% | 99.7% |
| **1.0** | 0.0000 | 0.0000 | – | 0.0% | 0.0% | 100.0% | 100.0% |

The same AUPRC score of 0.1219 was achieved using both the weighted and rule-based ensemble approaches when LightGBM with undersampling (high-recall model) contributed 0% and XGBoost without sampling (high-precision model) contributed 100% to the final prediction. This result further supports the effectiveness of XGBoost without sampling, as it produced strong performance independently, without contributions from other models. These results are consistent with those presented in Table 15.

Both the rule-based and weighted ensemble approaches involving XGBoost with oversampling and random forest with adjusetd weights got an third best AUPRC score of 0.0984. In both cases, XGBoost with oversampling was the only contributor to the final score. This emphasises the reliability of XGBoost in the context of money mule detection. The corresponding performance metrics are presented in Table 16, where the best results were observed at a threshold of 0.9. At this threshold, 56.6% of money mules were detected, while the false positive rate remained close to 1%.

Again it is important to note that this AUPRC score (0.0984) differs from the result reported for XGBoost with oversampling in Section 6.7 (0.1314). The same hyperparameters were used, but the models were retrained again for the ensemble approach, and minor variation in results occur due to inherent randomness in the training process.

Table 16. Model performance and confusion matrix percentages at different thresholds for XGBoost with oversampling.

| Threshold | Precision | Recall | F1 score | TP% | FP% | TN% | FN% |
|---|---|---|---|---|---|---|---|
| **0.0** | 0.0012 | 1.0000 | 0.0023 | 100.0% | 100.0% | 0.0% | 0.0% |
| **0.1** | 0.0045 | 0.9444 | 0.0089 | 94.4% | 24.5% | 75.5% | 5.6% |
| **0.2** | 0.0068 | 0.9181 | 0.0135 | 91.8% | 15.5% | 84.5% | 8.2% |
| **0.3** | 0.0089 | 0.8918 | 0.0177 | 89.2% | 11.5% | 88.5% | 10.8% |
| **0.4** | 0.0113 | 0.8670 | 0.0223 | 86.7% | 8.8% | 91.2% | 13.3% |
| **0.5** | 0.0142 | 0.8436 | 0.0278 | 84.4% | 6.8% | 93.2% | 15.6% |
| **0.6** | 0.0175 | 0.7968 | 0.0342 | 79.7% | 5.2% | 94.8% | 20.3% |
| **0.7** | 0.0225 | 0.7485 | 0.0437 | 74.9% | 3.8% | 96.2% | 25.1% |
| **0.8** | 0.0311 | 0.6813 | 0.0595 | 68.1% | 2.5% | 97.5% | 31.9% |
| **0.9** | 0.0529 | 0.5658 | 0.0968 | 56.6% | 1.2% | 98.8% | 43.4% |
| **1.0** | 0.0000 | 0.0000 | – | 0.0% | 0.0% | 100.0% | 100.0% |

For the combination of LightGBM with undersampling as the high-recall model and random forest with adjusted class weights as the high-precision model, both ensemble approaches achieved their best performance when LightGBM contributed 90% and random forest 10% to the final prediction. The corresponding results are presented in Tables 17 and 18.

For the weighted ensemble approach, the optimal threshold was 0.9, at which 45.8% of money mules were correctly identified, and the false positive rate remained at 0.8%. The resulting AUPRC was 0.0775. Similarly, the rule-based ensemble approach also achieved its best performance at threshold 0.9, detecting 63.6% of money mules. However, this came at

the cost of a higher false positive rate of 2.1%, which may be considered too high for practical implementation in a real-world banking environment. The corresponding AUPRC was 0.0765.

Table 17. Model performance and confusion matrix percentages at different thresholds for LightGBM with undersampling, random forest with adjusted weights, and weighted ensemble approach.

| Threshold | Precision | Recall | F1 score | TP% | FP% | TN% | FN% |
|---|---|---|---|---|---|---|---|
| **0.0** | 0.0012 | 1.0000 | 0.0023 | 100.0% | 100.0% | 0.0% | 0.0% |
| **0.1** | 0.0031 | 0.9708 | 0.0061 | 97.1% | 36.6% | 63.4% | 2.9% |
| **0.2** | 0.0047 | 0.9503 | 0.0094 | 95.0% | 23.2% | 76.8% | 5.0% |
| **0.3** | 0.0062 | 0.9254 | 0.0123 | 92.5% | 17.2% | 82.8% | 7.5% |
| **0.4** | 0.0078 | 0.8947 | 0.0154 | 89.5% | 13.2% | 86.8% | 10.5% |
| **0.5** | 0.0099 | 0.8640 | 0.0195 | 86.4% | 10.1% | 89.9% | 13.6% |
| **0.6** | 0.0127 | 0.8246 | 0.0251 | 82.5% | 7.4% | 92.6% | 17.5% |
| **0.7** | 0.0176 | 0.7719 | 0.0343 | 77.2% | 5.0% | 95.0% | 22.8% |
| **0.8** | 0.0280 | 0.6842 | 0.0537 | 68.4% | 2.8% | 97.2% | 31.6% |
| **0.9** | 0.0648 | 0.4576 | 0.1135 | 45.8% | 0.8% | 99.2% | 54.2% |
| **1.0** | 0.0000 | 0.0000 | – | 0.0% | 0.0% | 100.0% | 100.0% |

Table 18. Model performance and confusion matrix percentages at different thresholds for LightGBM with undersampling, random forest with adjusted weights, and rule-based ensemble approach.

| Threshold | Precision | Recall | F1 score | TP% | FP% | TN% | FN% |
|---|---|---|---|---|---|---|---|
| **0.0** | 0.0012 | 1.0000 | 0.0023 | 100.0% | 100.0% | 0.0% | 0.0% |
| **0.1** | 0.0030 | 0.9781 | 0.0059 | 97.8% | 38.1% | 61.9% | 2.2% |
| **0.2** | 0.0045 | 0.9518 | 0.0089 | 95.2% | 24.5% | 75.5% | 4.8% |
| **0.3** | 0.0058 | 0.9342 | 0.0116 | 93.4% | 18.5% | 81.5% | 6.6% |
| **0.4** | 0.0072 | 0.9079 | 0.0143 | 90.8% | 14.5% | 85.5% | 9.2% |
| **0.5** | 0.0089 | 0.8787 | 0.0176 | 87.9% | 11.3% | 88.7% | 12.1% |
| **0.6** | 0.0112 | 0.8509 | 0.0221 | 85.1% | 8.7% | 91.3% | 14.9% |
| **0.7** | 0.0142 | 0.7982 | 0.0279 | 79.8% | 6.4% | 93.6% | 20.2% |
| **0.8** | 0.0203 | 0.7558 | 0.0396 | 75.6% | 4.2% | 95.8% | 24.4% |
| **0.9** | 0.0342 | 0.6360 | 0.0650 | 63.6% | 2.1% | 97.9% | 36.4% |
| **1.0** | 0.0000 | 0.0000 | – | 0.0% | 0.0% | 100.0% | 100.0% |

# 7. Discussion

This section interprets the results presented in the previous chapters, highlighting the most effective models and sampling strategies for detecting money mule accounts in an imbalanced dataset. The findings are analysed both in terms of predictive performance and practical value for real-world anti-money laundering (AML) efforts in financial institutions. The discussion also examines the strengths and limitations of the applied methodology, the quality and scope of the data, and the trade-offs encountered when optimising for imbalanced classification problems.

## 7.1 Interpretation of Results

Table 19 summarises the best-performing configuration for each algorithm. XGBoost combined with random oversampling achieved the highest test-set AUPRC (0.1314), recovered over 90% of known mules, and limited the false-positive rate to 5.7%. This result highlights the strength of gradient-boosted ensembles in imbalanced classification tasks. CatBoost and LightGBM also achieved high AUPRC scores when trained on the original dataset, but their recalls remained low, as most predicted probabilities fell below the 0.5 threshold.

Table 19. Best results from each model and ensemble model based on test data.

| Model | Sampling | Time (s) | AUPRC (val) | AUPRC (test) | Precision | Recall | F1 Score | TP (%) | FP (%) |
|---|---|---|---|---|---|---|---|---|---|
| LR | Weights | 84 | 0.0363 | 0.0402 | 0.0089 | 0.8623 | 0.0175 | 86.2 | 13.8 |
| KNN | Under | 324 | 0.0278 | 0.0334 | 0.0099 | 0.8386 | 0.0195 | 83.9 | 10.3 |
| Linear SVM | Over | 102 | 0.0351 | 0.0394 | 0.0090 | 0.8649 | 0.0179 | 86.5 | 12.1 |
| DT | Over | 37 | 0.0467 | 0.0440 | 0.0138 | 0.6861 | 0.0271 | 68.6 | 6.3 |
| RF | None | 264 | 0.0974 | 0.0966 | - | 0.0000 | - | 0.0 | 0.0 |
| GBT | Weights | 192 | 0.0864 | 0.0814 | 0.0125 | 0.8715 | 0.0246 | 87.2 | 8.8 |
| XGBoost | Over | 90 | 0.0895 | **0.1314** | 0.0188 | 0.9046 | 0.0368 | 90.5 | 5.7 |
| CatBoost | None | 366 | 0.1132 | 0.1229 | 0.4807 | 0.0345 | 0.0644 | 3.4 | 0.0 |
| LightGBM | None | 90 | 0.1131 | 0.1137 | 0.3333 | 0.0013 | 0.0026 | 0.1 | 0.0 |
| IF | Under | 135 | 0.0042 | 0.0125 | 0.0044 | 0.7221 | 0.0087 | 72.2 | 19.5 |

Traditional models such as logistic regression, KNN, linear SVM, and decision tree achieved moderate AUPRC values (0.0402, 0.0334, 0.0394, and 0.0440) that were higher than baseline AUPRC (0.001). Logistic regression and linear SVM both had recalls above 86%. Although they lack the discriminative power of ensemble models, their simplicity and interpretability make them suitable for use cases where transparency is important. Random forest achieved a relatively high AUPRC (0.0966) without any sampling but failed to recall any mules under the evaluated threshold.

The choice of sampling strategy played a key role in model performance. Both oversampling and class-weighting consistently improved recall. Undersampling led to faster training due to

reduced dataset size but at the cost of lower recall. Oversampling required more computation in some cases but produced the best trade-off between minority-class sensitivity and ranking performance, as demonstrated by XGBoost.

AUPRC is an appropriate evaluation metric in highly imbalanced settings, as it reflects the ranking quality across thresholds. Tree-based ensembles performed best overall. XGBoost and CatBoost produced the two highest AUPRC values on the test set. In contrast, models such as logistic regression, linear SVM, and KNN got lower AUPRC values despite high recall, indicating limited ability to capture complex, nonlinear behavioural patterns.

For random forest, CatBoost, and LightGBM, most predicted probabilities were concentrated at the lower end of the scale. This distribution supports performance at low thresholds but implies lack of stability, as small threshold adjustments can increase false positives or negatives. The decision tree, random forest, and gradient boosted trees exhibited slightly lower AUPRC values on test data than on validation data, suggesting mild overfitting.

Training times ranged from 37 seconds (decision tree) to 6 minutes (CatBoost). Spark-native models (GBT, RF) scaled efficiently with data volume, while CatBoost incurred additional overhead due to Pandas conversion and memory usage. Although all training durations are manageable in a monthly re-training scenario, CatBoost would benefit from memory-optimised infrastructure.

Isolation Forest was included to assess the potential of unsupervised learning. Its low AUPRC (0.0125) and high false-positive rate (19.5%) indicate it is not suitable for stand-alone blocking. However, its recall of 72% suggests possible use as an auxiliary signal within a broader ensemble or investigation pipeline.

In conclusion, it is recommended to keep the focus on XGBoost as the main scoring model for detecting money mules. It achieved the highest AUPRC on the test set, indicating strong ranking performance in an imbalanced setting.

To assess whether model performance could be improved further, ensemble approaches were explored using both weighted and rule-based combinations of high-recall and high-precision models. A total of 968 ensemble configurations were evaluated. The best result (AUPRC 0.1222) was achieved by a weighted ensemble of XGBoost with oversampling and XGBoost without sampling. While this was slightly below the best stand-alone result (AUPRC 0.1314), the ensemble provided greater flexibility across thresholds. For instance, at a threshold of 0.2, it

correctly identified over half of the money mules while keeping the false positive rate under 1%, offering a more practical balance for real-world use.

The ensemble experiments reinforced the conclusion that XGBoost is the most effective model for this task. The top ensemble configurations all relied on XGBoost variants. Rather than improving overall accuracy, the ensemble exercise ended up providing more information how XGBoost behaves across different thresholds. This insight is particularly valuable for threshold tuning, as it directly affects both the total number of predicted positive cases and the precision of true positive predictions.

## 7.2 Practical Implications

XGBoost with oversampling stands out as the most suitable model for deployment. At the standard threshold of 0.5, it would correctly flag approximately 90% of known mule accounts while generating alerts for fewer than 6% of legitimate customers. Ensemble experiments further demonstrated that by adjusting the threshold to 0.2, the ensemble model can still detect over half of all mules while keeping the false positive rate below 1% that could generate less operation costs.

It is suggested that the model be run in batches to generate scores, as it relies on data collected over a longer period. The resulting score can then be integrated into downstream systems to enrich existing fraud detection solutions, but it is not intended to replace or trigger immediate actions on its own.

## 7.3 Strengths and Limitations

One of the key strengths of this work is its contribution to an under-explored area. While machine learning has been widely applied to fraud detection, there are relatively few studies focused specifically on money mule identification. This study helps fill that gap by thoroughly evaluating multiple machine learning models and sampling strategies in this context, providing a comprehensive analysis.

However, several limitations must be acknowledged. First, the model was trained and evaluated on a single snapshot of data, which does not account for concept drift. Changes in mule profiles or behaviours over time could degrade performance. Regular re-training with rolling time-window validation could address this limitation. Second, only confirmed mule cases were labelled as positive, while the negative class may include undetected mules. This label noise can distort model calibration and inflate precision. Third, random under- and oversampling

methods introduce risks; random oversampling may lead to the memorisation of noise, and random undersampling might discard useful majority-class examples. More robust sampling methods or synthetic data generation should be explored in future work.

Additional limitations include the very limited availability of data, restricting the depth and accuracy of the analysis. Access to larger and more diverse datasets would likely improve model performance. The implementation of more thorough cross-validation strategies would also enhance the robustness of the model evaluation. Furthermore, the current approach to threshold tuning and score calibration is relatively basic, indicating significant potential for future improvements in these areas.

While XGBoost delivered strong results, its complexity poses interpretability challenges. Techniques such as SHAP values could provide post-hoc explanations and improve transparency. Although ensemble models offered flexibility in threshold tuning, their AUPRC did not consistently outperform the best single model, emphasising the importance of a careful cost–benefit analysis when increasing model complexity.

It is important to note that AUPRC scores are sensitive to data characteristics, and with limited training data and small numbers of confirmed mules, minor changes in training processes can create noticeable variations in model outcomes. This fluidity must be considered when interpreting results.

Finally, the use of historical mule data across several years combined with recent legitimate customer data may introduce temporal bias. A better distribution of legitimate customer data across multiple years could more accurately represent seasonality effects. Behavioural patterns and fraud tactics evolve, and models may inadvertently capture outdated patterns. This trade-off was accepted to increase the positive sample size but should be revisited in future evaluations.

# 8. Teamwork

Collaboration between the authors went well. Meetings were held regularly and increased in frequency as the submission deadline approached, culminating in daily sessions. Each meeting served to define and prioritise the tasks for the coming week or, nearer the deadline, for the following days. Both authors contributed equally to model training and the overall drafting of the thesis. In practice, one author concentrated more on the writing of the theoretical framework while the other focused on the empirical implementation. However, they continuously reviewed and critiqued each other's work to ensure coherence and consistency throughout.

# 9. Conclusion

The primary goal of this thesis was to design and evaluate a machine learning based scoring system that predicts the likelihood that a bank customer is acting as a money mule. This kind of system is needed because rule-based methods can't keep up with fast-changing fraud tactics, and undetected mule activity can cause financial and reputational damage. The model assigns each customer a risk score from 0 to 100, helping the bank detect suspicious payments.

Following the CRISP-DM framework, dataset of confirmed mule accounts and a large sample of legitimate customers was used. Careful feature selection and reduction steps cut the original 415 variables down to 36 informative predictors. Ten machine learning models were trained, with hyperparameter optimisation performed using Optuna. Class imbalance was addressed through random undersampling, oversampling, and class-weighting. Model quality was measured with the area under the precision–recall curve (AUPRC), which is well-suited for imbalanced datasets.

The experimental results show that gradient-boosted decision-tree ensembles outperform simpler linear or distance-based classifiers. In particular, XGBoost trained on the oversampled dataset achieved the highest AUPRC of 0.1314, recalling 90.5% of known mules while restricting the false positives rate to 5.7% at a 0.5 threshold. CatBoost and LightGBM also achieved high AUPRC scores, though their prediction scores were generally skewed toward the lower end. A weighted and rule-based ensembles confirmed that XGBoost dominate. The best ensemble (AUPRC = 0.1222) detected half of the money mules while keeping false positives in manageable rate.

The study faces some limitations. First, the positive class is small and scattered across time, so label noise and concept drift remain concerns. Second, random resampling may discard useful information or propagate noise. Third, although explainability was outside the thesis scope, deploying the model would benefit from SHAP-based insight into feature contributions.

In summary, this thesis shows that XGBoost and other gradient-boosted tree models work well for detecting money mule accounts. XGBoost gave the best results, and ensemble tests confirmed its reliability and helped explain how it behaves at different thresholds. Machine learning can improve current anti-money laundering systems by helping to spot high-risk accounts and keep up with changing fraud patterns. With regular updates, better explanations, and real-time use, this approach can make money mule detection more accurate and efficient.

# References


[1] United Nations Office on Drugs and Crime. Money Laundering;. [Accessed 17.05.2025]. https://www.unodc.org/unodc/en/money-laundering/overview.html.

[2] European Parliament and Council of the European Union. Directive (EU) 2015/849 of 20 May 2015 on the prevention of the use of the financial system for the purposes of money laundering or terrorist financing;. [Accessed 21.04.2025]. https://eur-lex.europa.eu/eli/dir/2015/849/oj.

[3] Europol. Money laundering; n.d. [Accessed 15.12.2024]. https://www.europol.europa.eu/crime-areas/economic-crime/money-laundering.

[4] Publications Office of the European Union. Money laundering. Publications Office of the European Union; 2021. [Accessed 22.04.2025]. https://eur-lex.europa.eu/EN/legal-content/glossary/money-laundering.html.

[5] Raza MS, Zhan Q, Rubab S. Role of money mules in money laundering and financial crimes a discussion through case studies. Journal of Financial Crime. 2020 Jun;27(3):911–931. Available from: http://dx.doi.org/10.1108/JFC-02-2020-0028.

[6] Federal Bureau of Investigation. Money Mules;. [Accessed 10.03.2025]. https://www.fbi.gov/how-we-can-help-you/scams-and-safety/common-frauds-and-scams/money-mules.

[7] Leukfeldt R, Kleemans ERE. In: Cybercrime, money mules and situational crime prevention: Recruitment, motives and involvement mechanisms. Routledge; 2019. p. 75–89. Available from: http://dx.doi.org/10.4324/9781351176194-5.

[8] Hulsse R. The Money Mule: Its Discursive Construction and the Implications. Vanderbilt Journal of Transnational Law. 2017 Oct;50(4):1007-32. Available from: https://scholarship.law.vanderbilt.edu/vjtl/vol50/iss4/5.

[9] European Police Office. Why is Cash Still King? A Strategic Report on the Use of Cash by Criminal Groups as a Facilitator for Money Laundering. The Hague, The Netherlands: European Police Office (Europol); 2015. QL-07-14-083-EN-N. Available from: https://www.europol.europa.eu/sites/default/files/documents/europolcik%20%281%29.pdf.

[10] Europol. Money Muling: Public Awareness and Prevention Guides; 2023. [Accessed 10.03.2025]. https://www.europol.europa.eu/operations-services-and-innovation/public-awareness-and-prevention-guides/money-muling.

[11] Bekkers LMJ, Moneva A, Leukfeldt ER. Understanding cybercrime involvement: a quasi-experiment on engagement with money mule recruitment ads on Instagram. Journal of Experimental Criminology. 2022 Nov;20(2):375–394. Available from: http://dx.doi.org/10.1007/s11292-022-09537-7.

[12] Cifas Press Team. Latest fraud statistics reveal middle-aged mules targeted by online money laundering gangs; 2021. [Accessed 16.05.2025]. https://www.cifas.org.uk/newsroom/middle-aged-mules.

[13] Internet Crime Complaint Center (IC3). Cyber Actors Use Online Dating Sites To Conduct Confidence/Romance Fraud And Recruit Money Mules; 2019. [Accessed 16.05.2025]. https://www.ic3.gov/PSA/2019/PSA190805.

[14] Bellomarini L, Laurenza E, Sallinger E. Rule-based Anti-Money Laundering in Financial Intelligence Units: Experience and Vision. CEUR Workshop Proceedings. 2020;2644. Available from: https://ceur-ws.org/Vol-2644/paper40.pdf.

[15] United States Congress. Bank Secrecy Act; 1970. 12 U.S.C. §1829b. Available from: https://www.fincen.gov/resources/statutes-regulations/bank-secrecy-act.

[16] Financial Action Task Force. The FATF Recommendations; 2012. As amended February 2025. Available from: https://www.fatf-gafi.org/content/dam/fatf-gafi/recommendations/FATF%20Recommendations%202012.pdf.

[17] Authority for Anti-Money Laundering and Countering the Financing of Terrorism (AMLA). About AMLA; 2024. Established by Regulation (EU) 2021/557; operational mid-2025. Available from: https://www.amla.europa.eu/about-amla_en.

[18] Chen J. Know your client (KYC): What it means and compliance requirements. Investopedia; 2024. "[Accessed 14.05.2025]". Available from: https://www.investopedia.com/terms/k/knowyourclient.asp.

[19] Financial Action Task Force. Best Practices on Combating the Abuse of Non-Profit Organisations; 2023. Amendments to FATF Recommendation 8. Available from: https:

//www.fatf-gafi.org/en/publications/Financialinclusionandnpoissues/Bpp-combating-abuse-npo.html.

[20] European Banking Authority. Final Report on Guidelines on Customer Due Diligence and the Factors Credit and Financial Institutions Should Consider When Assessing the ML/TF Risk Associated With Individual Business Relationships and Occasional Transactions. European Banking Authority; 2021. EBA/GL/2021/02. Available from: https://www.eba.europa.eu/sites/default/files/document_library/Publications/Guidelines/2023/EBA-GL-2023-03/1061654/Guidelines%20ML%20TF%20Risk%20Factors_conslidated.pdf.pdf.

[21] Bashir R, Rajeev R, Shatarah A, Bashir N. A Risk Score Analysis Related to Money Laundering in Financial Institutions Across Nations. In: 2020 8th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO). IEEE; 2020. p. 940–943. Available from: http://dx.doi.org/10.1109/ICRITO48877.2020.9197900.

[22] Esoimeme E. How Banks Can Detect Money Mules in the Time of COVID-19. Financial Regulation International. 2020 June;23(5):24. Special issue. Available from: https://ssrn.com/abstract=3513558.

[23] Financial Action Task Force (FATF). Anti-money Laundering and Terrorist Financing Measures and Financial Inclusion: With a Supplement on Customer Due Diligence. Paris, France: Financial Action Task Force; 2017. Available from: https://www.fatf-gafi.org/media/fatf/content/images/Updated-2017-FATF-2013-Guidance.pdf.

[24] Ai L. "Rule-based but risk-oriented" approach for combating money laundering in Chinese financial sectors. Journal of Money Laundering Control. 2012 May;15(2):198–209. Available from: http://dx.doi.org/10.1108/13685201211218225.

[25] Financial Action Task Force. Opportunities and Challenges of New Technologies for AML/CFT. FATF; 2021. Available from: https://www.fatf-gafi.org/content/dam/fatf-gafi/guidance/Opportunities-Challenges-of-New-Technologies-for-AML-CFT.pdf.

[26] Global Investigations Review (Forensic Risk Alliance). Anti-Money Laundering Trends and Challenges; 2020. Lexology. Available from: https://www.lexology.com/library/detail.aspx?g=381280e4-7b9a-4a9c-beac-cda860430dee.

[27] UK Financial Conduct Authority. Proceeds of Fraud – Detecting and Preventing Money Mules; 2023. Static rule-based systems yield high false positives and miss evolving mule fraud patterns. Available from: https://www.fca.org.uk/publications/multi-firm-reviews/proceeds-fraud-detecting-preventing-money-mules.

[28] Ngai E, Hu Y, Wong Y, Chen Y, Sun X. The application of data mining techniques in financial fraud detection: A classification framework and an academic review of literature. Decision Support Systems. 2011;50(3):559-69.

[29] Oztas B, Yildiz S, Kan G. Transaction monitoring in anti-money laundering: A qualitative analysis and points of view from industry. Future Generation Computer Systems. 2024;159:161-71.

[30] Dal Pozzolo A, Boracchi G, Caelen O, Alippi C, Bontempi G. Credit Card Fraud Detection: A Realistic Modeling and a Novel Learning Strategy. IEEE Transactions on Neural Networks and Learning Systems. 2018 Aug;29(8):3784–3797. Available from: http://dx.doi.org/10.1109/TNNLS.2017.2736643.

[31] European Parliament and Council of the European Union. Regulation (EU) 2016/679 of the European Parliament and of the Council (General Data Protection Regulation); 2016. OfficialJournaloftheEuropeanUnion.

[32] Mitchell TM. Machine Learning. New York, NY: McGraw-Hill; 1997.

[33] Bishop CM. Pattern Recognition and Machine Learning. New York, NY: Springer; 2006.

[34] Domingos P. A few useful things to know about machine learning. Communications of the ACM. 2012 Oct;55(10):78–87. Available from: http://dx.doi.org/10.1145/2347736.2347755.

[35] Sügis E, Tampuu A, Aljanaki A, Fišel M, Kull M. Praktiline andmeteadus: Kõrgkooliõpik. Saul E, editor. Tartu: Tartu Ülikooli arvutiteaduse instituut; 2024. Uuendatud jaanuaris 2025. Available from: https://courses.cs.ut.ee/t/andmeteadus/.

[36] Elhusseny NS, Ouf AM Shimaa Mohamed andIdrees. Credit Card Fraud Detection Using Machine Learning Techniques. Future Computing and Informatics Journal. 2022 Jun;7(1):13–31. Available from: http://dx.doi.org/10.54623/fue.fcij.7.1.2.

[37] Itoo F, Meenakshi, Singh S. Comparison and analysis of logistic regression, Naïve Bayes and KNN machine learning algorithms for credit card fraud detection. International Journal

of Information Technology. 2021;13:1503-11. Available from: https://doi.org/10.1007/s41870-020-00430-y.

[38] Bhattacharyya S, Jha S, Tharakunnel K, Westland JC. Data mining for credit card fraud: A comparative study. Decision Support Systems. 2011 Feb;50(3):602–613. Available from: http://dx.doi.org/10.1016/j.dss.2010.08.008.

[39] Breiman L. Random Forests. Machine Learning. 2001;45(1):5-32.

[40] Liu C, Chan Y, Alam Kazmi SH, Fu H. Financial Fraud Detection Model: Based on Random Forest. International Journal of Economics and Finance. 2015 Jun;7(7). Available from: http://dx.doi.org/10.5539/ijef.v7n7p178.

[41] Baisholan N, Dietz JE, Gnatyuk S, Turdalyuly M, Matson ET, Baisholanova K. FraudX AI: An Interpretable Machine Learning Framework for Credit Card Fraud Detection on Imbalanced Datasets. Computers. 2025 Mar;14(4):120. Available from: http://dx.doi.org/10.3390/computers14040120.

[42] Friedman JH. Greedy Function Approximation: A Gradient Boosting Machine. Annals of Statistics. 2001;29(5):1189-232.

[43] Chen T, Guestrin C. XGBoost: A Scalable Tree Boosting System. CoRR. 2016;abs/1603.02754. Available from: http://arxiv.org/abs/1603.02754.

[44] Hajek P, Abedin MZ, Sivarajah U. Fraud Detection in Mobile Payment Systems using an XGBoost-based Framework. Information Systems Frontiers. 2022 Oct;25(5):1985–2003. Available from: http://dx.doi.org/10.1007/s10796-022-10346-6.

[45] Bakry AN, Alsharkawy AS, Farag MS, Raslan KR. Automatic suppression of false positive alerts in anti-money laundering systems using machine learning. The Journal of Supercomputing. 2023 Oct;80(5):6264–6284. Available from: http://dx.doi.org/10.1007/s11227-023-05708-z.

[46] Bynagari NB, Ahmed AAA. Anti-Money Laundering Recognition through the Gradient Boosting Classifier. Academy of Accounting and Financial Studies Journal. 2021;25(5):1-11. Available from: https://zenodo.org/record/5523917.

[47] Ke G, Meng Q, Finley T, Wang T, Chen W, Ma W, et al. LightGBM: A Highly Efficient Gradient Boosting Decision Tree. In: Advances in Neural Information Processing Systems. vol. 30; 2017. p. 3146-54.

[48] Huang K. An Optimized LightGBM Model for Fraud Detection. Journal of Physics: Conference Series. 2020 Nov;1651(1):012111. Available from: http://dx.doi.org/10.1088/1742-6596/1651/1/012111.

[49] Florek P, Zagdanski A. Benchmarking state-of-the-art gradient boosting algorithms for classification; 2023. Available from: https://arxiv.org/abs/2305.17094.

[50] Prokhorenkova L, Gusev G, Vorobev A, Dorogush AV, Gulin A. CatBoost: Unbiased Boosting with Categorical Features. In: Advances in Neural Information Processing Systems. vol. 31; 2018. p. 6638-48.

[51] Moparthi NR. Fraud Detection in Banking Data by Machine Learning Techniques. Expert Systems with Applications. 2024;200:117032.

[52] Aldania A, Patel R, Singh N. Comparative Performance of Gradient Boosting Algorithms on Imbalanced Financial Data. Journal of Financial Data Science. 2023;5(2):45-59.

[53] Chandola V, Banerjee A, Kumar V. Anomaly detection: A survey. ACM Computing Surveys. 2009 Jul;41(3):1–58. Available from: http://dx.doi.org/10.1145/1541880.1541882.

[54] Liu FT, Ting KM, Zhou ZH. Isolation Forest. In: 2008 Eighth IEEE International Conference on Data Mining. IEEE; 2008. p. 413-22.

[55] Hanae A, Abdellah B, Saida E, Youssef G. End-to-End Real-time Architecture for Fraud Detection in Online Digital Transactions. International Journal of Advanced Computer Science and Applications. 2023;14(6). Available from: http://dx.doi.org/10.14569/IJACSA.2023.0140680.

[56] Breunig MM, Kriegel HP, Ng RT, Sander J. LOF: Identifying Density-Based Local Outliers. In: Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data; 2000. p. 93-104.

[57] Sakurada M, Yairi T. Anomaly Detection Using Autoencoders with Nonlinear Dimensionality Reduction. In: Proceedings of the MLSDA 2014 2nd Workshop on Machine Learning for Sensory Data Analysis. MLSDA'14. ACM; 2014. p. 4–11. Available from: http://dx.doi.org/10.1145/2689746.2689747.

[58] Fiore U, De Santis A, Perla F, Zanetti P, Palmieri F. Using generative adversarial networks for improving classification effectiveness in credit card fraud detection. Information

Sciences. 2019 Apr;479:448–455. Available from: http://dx.doi.org/10.1016/j.ins.2017.12.030.

[59] Zhu S, Wu H, Ngai EWT, Ren J, He D, Ma T, et al. A Financial Fraud Prediction Framework Based on Stacking Ensemble Learning. Systems. 2024 Dec;12(12):588. Available from: http://dx.doi.org/10.3390/systems12120588.

[60] Hamza C, Lylia A, Nadine C, Nicolas C. Semi-supervised Method to Detect Fraudulent Transactions and Identify Fraud Types while Minimizing Mounting Costs. International Journal of Advanced Computer Science and Applications. 2023;14(2). Available from: http://dx.doi.org/10.14569/IJACSA.2023.0140298.

[61] Hashemi SK, Mirtaheri SL, Greco S. Fraud Detection in Banking Data by Machine Learning Techniques. IEEE Access. 2023;11:3034–3043. Available from: http://dx.doi.org/10.1109/ACCESS.2022.3232287.

[62] Bartoletti M, Pes B, Serusi S. Data Mining for Detecting Bitcoin Ponzi Schemes. In: 2018 Crypto Valley Conference on Blockchain Technology (CVCBT). IEEE; 2018. p. 75–84. Available from: http://dx.doi.org/10.1109/CVCBT.2018.00014.

[63] Saito T, Rehmsmeier M. The Precision-Recall Plot Is More Informative than the ROC Plot When Evaluating Binary Classifiers on Imbalanced Datasets. PLOS ONE. 2015 Mar;10(3):e0118432. Available from: http://dx.doi.org/10.1371/journal.pone.0118432.

[64] Jullum M, Løland A, Huseby RB, Ånonsen G, Lorentzen J. Detecting money laundering transactions with machine learning. Journal of Money Laundering Control. 2020 Jan;23(1):173–186. Available from: http://dx.doi.org/10.1108/JMLC-07-2019-0055.

[65] Mytnyk B, Tkachyk O, Shakhovska N, Fedushko S, Syerov Y. Application of Artificial Intelligence for Fraudulent Banking Operations Recognition. Big Data and Cognitive Computing. 2023 May;7(2):93. Available from: http://dx.doi.org/10.3390/bdcc7020093.

[66] Tatulli MP, Paladini T, D'Onghia M, Carminati M, Zanero S. In: HAMLET: A Transformer Based Approach for Money Laundering Detection. Springer Nature Switzerland; 2023. p. 234–250. Available from: http://dx.doi.org/10.1007/978-3-031-34671-2_17.

[67] Leevy JL, Hancock J, Khoshgoftaar TM, Abdollah Zadeh A. Investigating the effectiveness of one-class and binary classification for fraud detection. Journal of Big Data. 2023 Oct;10(1). Available from: http://dx.doi.org/10.1186/s40537-023-00825-1.

[68] Kennedy RKL, Villanustre F, Khoshgoftaar TM, Salekshahrezaee Z. Synthesizing class labels for highly imbalanced credit card fraud detection data. Journal of Big Data. 2024 Mar;11(1). Available from: http://dx.doi.org/10.1186/s40537-024-00897-7.

[69] Bahnsen AC, Stojanovic A, Aouada D, Ottersten B. Cost Sensitive Credit Card Fraud Detection Using Bayes Minimum Risk. In: 2013 12th International Conference on Machine Learning and Applications. IEEE; 2013. p. 333–338. Available from: http://dx.doi.org/10.1109/ICMLA.2013.68.

[70] Lundberg SM, Lee S. A Unified Approach to Interpreting Model Predictions. In: Advances in Neural Information Processing Systems. vol. 30; 2017. p. 4765-74. Available from: https://arxiv.org/abs/1705.07874.

[71] Wirth R, Hipp J. CRISP-DM: Towards a Standard Process Model for Data Mining. In: Proceedings of the 4th International Conference on the Practical Applications of Knowledge Discovery and Data Mining. vol. 1. London, UK; 2000. p. 29-39.

[72] Schröer C, Kruse F, Gómez JM. A Systematic Literature Review on Applying CRISP-DM Process Model. Procedia Computer Science. 2021;181:526–534. Available from: http://dx.doi.org/10.1016/j.procs.2021.01.199.

[73] Verleysen M, François D. In: The Curse of Dimensionality in Data Mining and Time Series Prediction. Springer Berlin Heidelberg; 2005. p. 758–770. Available from: http://dx.doi.org/10.1007/11494669_93.

[74] Akiba T, Sano S, Yanase T, Ohta T, Koyama M. Optuna: A Next-generation Hyperparameter Optimization Framework. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining; 2019. .

[75] Davis J, Goadrich M. The relationship between Precision-Recall and ROC curves. In: Proceedings of the 23rd international conference on Machine learning - ICML '06. ICML '06. ACM Press; 2006. p. 233–240. Available from: http://dx.doi.org/10.1145/1143844.1143874.

[76] Sveriges Riksdag. Lag (2010:751) om betaltjänster; 2010. [Accessed 13.05.2025]. https://www.riksdagen.se/sv/dokument-och-lagar/dokument/svensk-forfattningssamling/lag-2010751-om-betaltjanster_sfs-2010-751/.

# Appendix

## I Best hyper-parameters and the parameter range

Table 20. Optimal hyperparameter values and their search ranges for logistic regression under different sampling strategies.

| Parameter | Range | No sampling | Undersampling | Oversampling | Adjusted weights |
|---|---|---|---|---|---|
| elasticNetParam | [0.0, 1.0] | 0.0033 | 0.4992 | 0.3964 | 0.5484 |
| fitIntercept | {True, False} | False | False | True | False |
| maxIter | [10, 100] | 66 | 44 | 93 | 100 |
| regParam | [0.0, 1.0] | 0.3299 | 0.0443 | 0.0468 | 0.0045 |
| tol | [1e-6, 1e-3] | 1.5111e-05 | 1.6809e-05 | 1.4750e-06 | 9.8469e-04 |

Table 21. Optimal hyperparameter values and their search ranges for k-nearest neighbours under different sampling strategies.

| Parameter | Range | Undersampling |
|---|---|---|
| n_neighbors | [1, 20] | 18 |
| weights | {uniform, distance} | distance |
| algorithm | {auto, ball_tree, kd_tree, brute} | ball_tree |
| leaf_size | [10, 50] | 36 |
| p | [1, 2] | 1 |

Table 22. Optimal hyperparameter values and their search ranges for linear SVM under different sampling strategies.

| Parameter | Range | No Sampling | Undersampling | Oversampling | Adjusted weights |
|---|---|---|---|---|---|
| fitIntercept | {True, False} | False | False | True | True |
| maxIter | [10, 100] | 89 | 100 | 83 | 61 |
| regParam | [0.0, 1.0] | 0.6718 | 0.0526 | 0.0006 | 0.0010 |
| tol | [1e-6, 1e-3] | 1.0436e-04 | 2.2140e-04 | 1.5513e-04 | 9.7418e-06 |

Table 23. Optimal hyperparameter values and their search ranges for decision tree under different sampling strategies.

| Parameter | Range | No Sampling | Undersampling | Oversampling | Adjusted weights |
|---|---|---|---|---|---|
| impurity | {gini, entropy} | gini | entropy | gini | gini |
| maxDepth | [1, 20] | 4 | 17 | 15 | 19 |
| minInstancesPerNode | [1, 20] | 17 | 3 | 14 | 2 |

Table 24. Optimal hyperparameter values and their search ranges for random forest under different sampling strategies.

| Parameter | Range | No Sampling | Undersampling | Oversampling | Adjusted weights |
|---|---|---|---|---|---|
| numTrees | [10, 100] | 97 | 55 | 42 | 73 |
| maxDepth | [3, 20] | 15 | 20 | 20 | 18 |
| minInstancesPerNode | [1, 15] | 4 | 6 | 14 | 15 |
| maxBins | [32, 64] | 39 | 49 | 63 | 35 |
| subsamplingRate | [0.5, 1.0] | 0.6556 | 0.5487 | 0.5257 | 0.5010 |
| minInfoGain | [0.0, 0.01] | 3.6552e-05 | 6.3614e-03 | 2.9120e-03 | 2.6255e-03 |
| featureSubsetStrategy | {auto, sqrt, log2} | log2 | auto | auto | sqrt |
| impurity | {gini, entropy} | entropy | entropy | entropy | entropy |

Table 25. Optimal hyperparameter values and their search ranges for gradient boosted trees under different sampling strategies.

| Parameter | Range | No Sampling | Undersampling | Oversampling | Adjusted weights |
|---|---|---|---|---|---|
| maxBins | [32, 64] | 50 | 45 | 34 | 51 |
| maxDepth | [3, 20] | 7 | 3 | 8 | 4 |
| maxIter | [10, 70] | 13 | 67 | 65 | 66 |
| minInfoGain | [0.0, 0.01] | 1.1804e-05 | 9.1227e-04 | 6.3721e-04 | 6.7983e-05 |
| minInstancesPerNode | [1, 15] | 9 | 11 | 13 | 9 |
| stepSize | [0.05, 0.3] | 0.2421 | 0.1567 | 0.1618 | 0.2979 |
| subsamplingRate | [0.5, 1.0] | 0.6635 | 0.6542 | 0.6668 | 0.7993 |
| featureSubsetStrategy | {auto, sqrt, log2} | sqrt | auto | auto | auto |

Table 26. Optimal hyperparameter values and their search ranges for XGBoost under different sampling strategies.

| Parameter | Range | No Sampling | Undersampling | Oversampling | Adjusted Weights |
|---|---|---|---|---|---|
| num_round | [60, 300] | 267 | 106 | 133 | 284 |
| max_depth | [3, 10] | 10 | 4 | 5 | 3 |
| eta | [0.1, 0.35] (log) | 0.1094 | 0.1783 | 0.2227 | 0.1140 |
| gamma | [3.0, 5.0] | 3.5030 | 3.2848 | 4.5157 | 3.6124 |
| min_child_weight | [1, 10] | 10 | 9 | 9 | 6 |
| subsample | [0.5, 0.9] | 0.8566 | 0.8155 | 0.5502 | 0.8640 |
| colsample_bytree | [0.8, 1.0] | 0.8904 | 0.8382 | 0.8295 | 0.8426 |
| reg_alpha | [0.4, 1.0] | 0.6595 | 0.6116 | 0.6821 | 0.6402 |
| reg_lambda | [0.0, 0.5] | 0.3429 | 0.2631 | 0.1063 | 0.1905 |

Table 27. Optimal hyperparameter values and their search ranges for Catboost under different sampling strategies.

| Parameter | Range | No Sampling | Undersampling | Oversampling | Adjusted weights |
|---|---|---|---|---|---|
| Bagging_temperature | [0.0, 1.0] | 0.6116 | 0.1148 | 0.0093 | 0.5568 |
| border_count | [1, 255] | 28 | 112 | 247 | 113 |
| Depth | [4, 10] | 4 | 9 | 6 | 4 |
| L2_leaf_reg | [1e-5, 10] | 1.5029 | 3.9043 | 5.0964 | 4.4521 |
| Learning_rate | [0.01, 0.3] | 0.1609 | 0.0102 | 0.0584 | 0.1466 |

Table 28. Optimal hyperparameter values and their search ranges for LightGBM under different sampling strategies.

| Parameter | Range | No Sampling | Undersampling | Oversampling | Adjusted weights |
|---|---|---|---|---|---|
| BaggingFraction | [0.5, 1.0] | 0.6095 | 0.7298 | 0.8847 | 0.8428 |
| BaggingFreq | [1, 10] | 1 | 9 | 1 | 9 |
| FeatureFraction | [0.5, 1.0] | 0.7984 | 0.6256 | 0.7984 | 0.6547 |
| LambdaL1 | [0.0, 1.0] | 0.1734 | 0.8280 | 0.5536 | 0.2457 |
| LambdaL2 | [0.0, 1.0] | 0.6630 | 0.9432 | 0.3336 | 0.7251 |
| LearningRate | [0.01, 0.3] | 0.0141 | 0.1136 | 0.1013 | 0.1658 |
| MaxDepth | [-1, 20] | 20 | 4 | 15 | 19 |
| MinDataInLeaf | [20, 100] | 100 | 97 | 40 | 42 |
| NumLeaves | [31, 256] | 93 | 92 | 97 | 170 |

Table 29. Optimal hyperparameter values and their search ranges for Isolation Forest under different sampling strategies.

| Parameter | Range | No Sampling | Undersampling | Oversampling |
|---|---|---|---|---|
| num_estimators | [50, 200] | 71 | 148 | 107 |
| max_samples | [64, 512] | 66 | 67 | 325 |
| max_features | [0.1, 1.0] | 0.7319 | 0.1474 | 0.1533 |
| contamination | [0.01, 0.2] | 0.0579 | 0.1987 | 0.0366 |
| bootstrap | {True, False} | False | False | True |

# II Optuna optimisation history plots

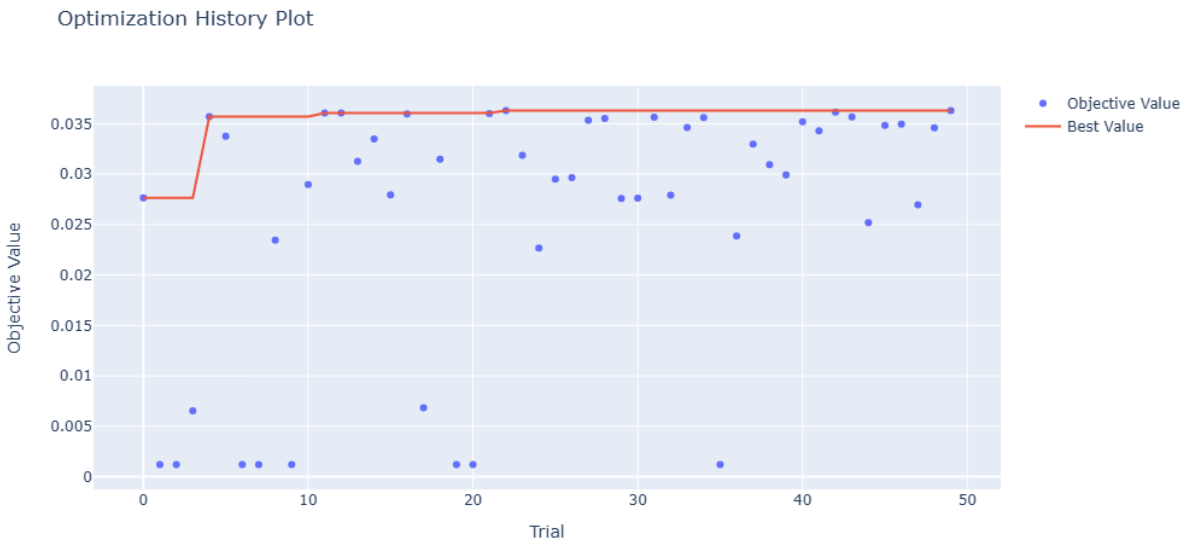



Figure 24. Logistic regression with adjusted weights optimization history plot.

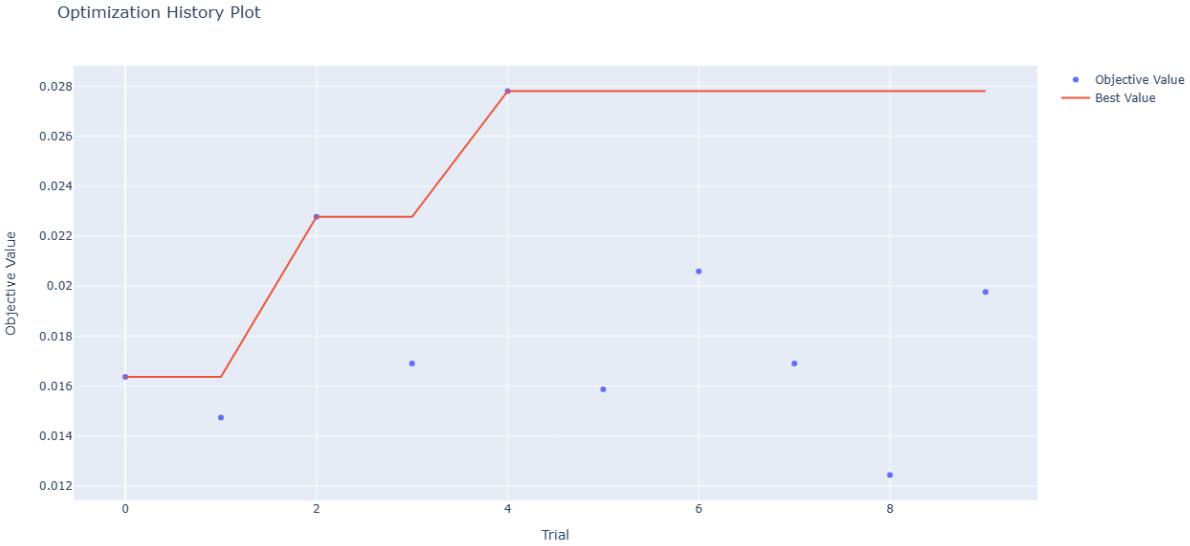



Figure 25. K-nearest neighbours with undersampling optimization history plot.

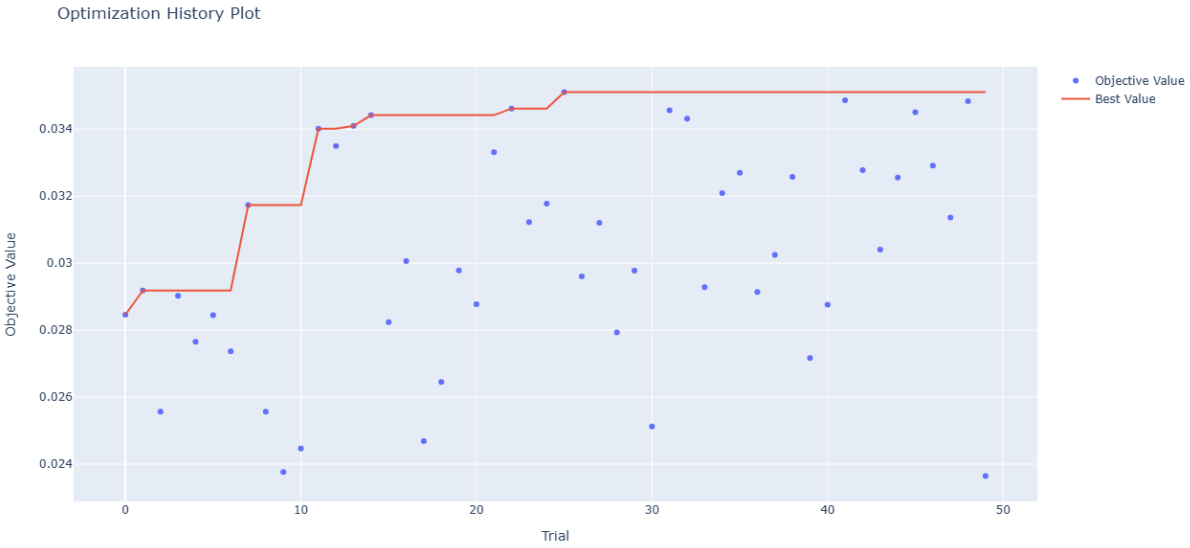



Figure 26. Linear SVM with oversampling optimization history plot.

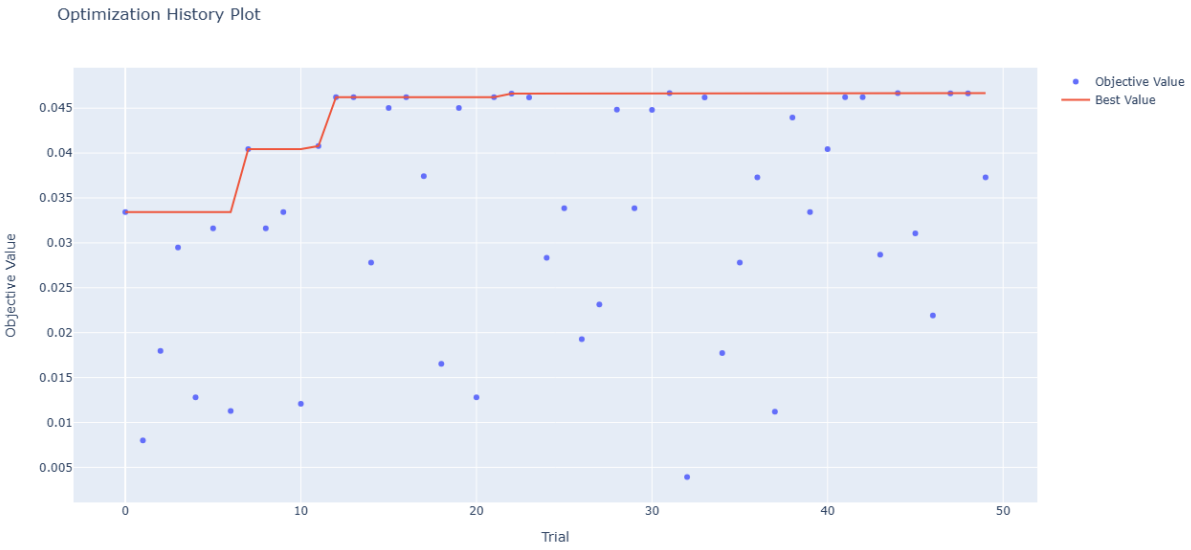



Figure 27. Decision tree with oversampling optimization history plot.

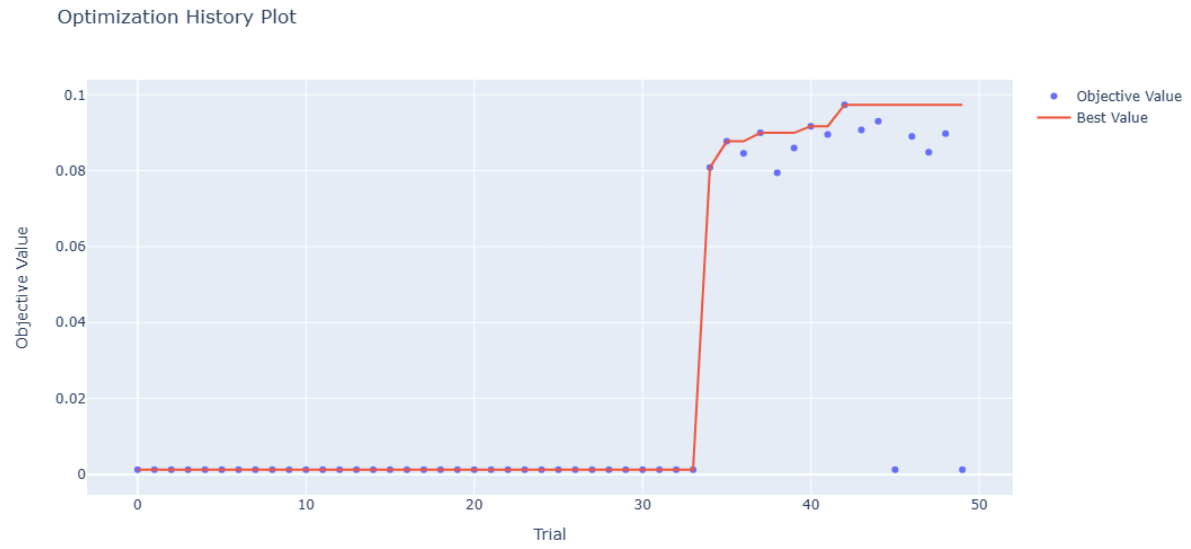



Figure 28. Random forest without sampling optimization history plot.

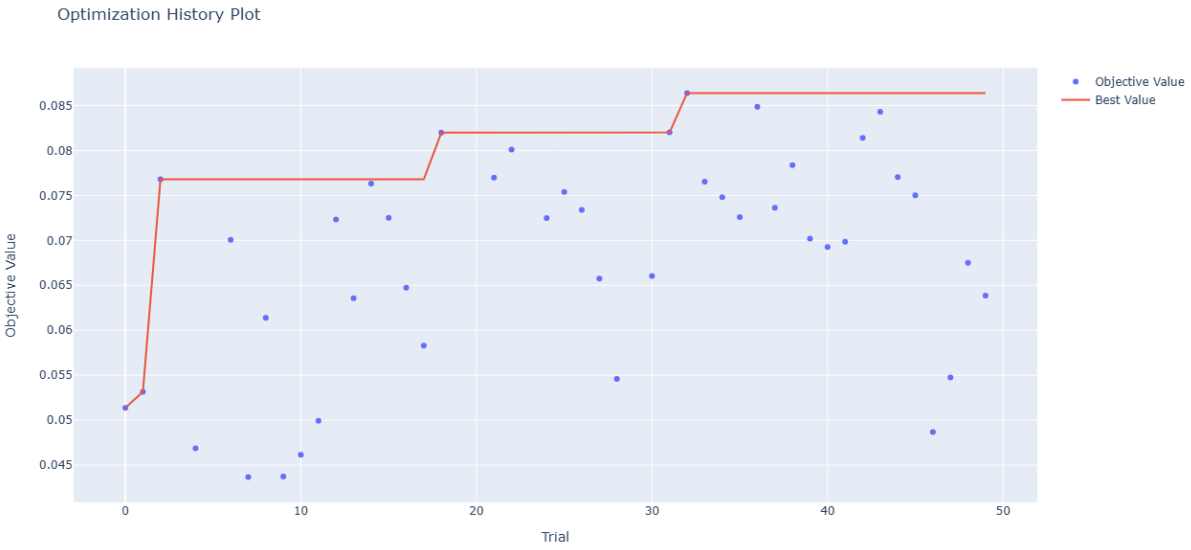



Figure 29. Gradient boosted trees with adjusted weights optimization history plot.

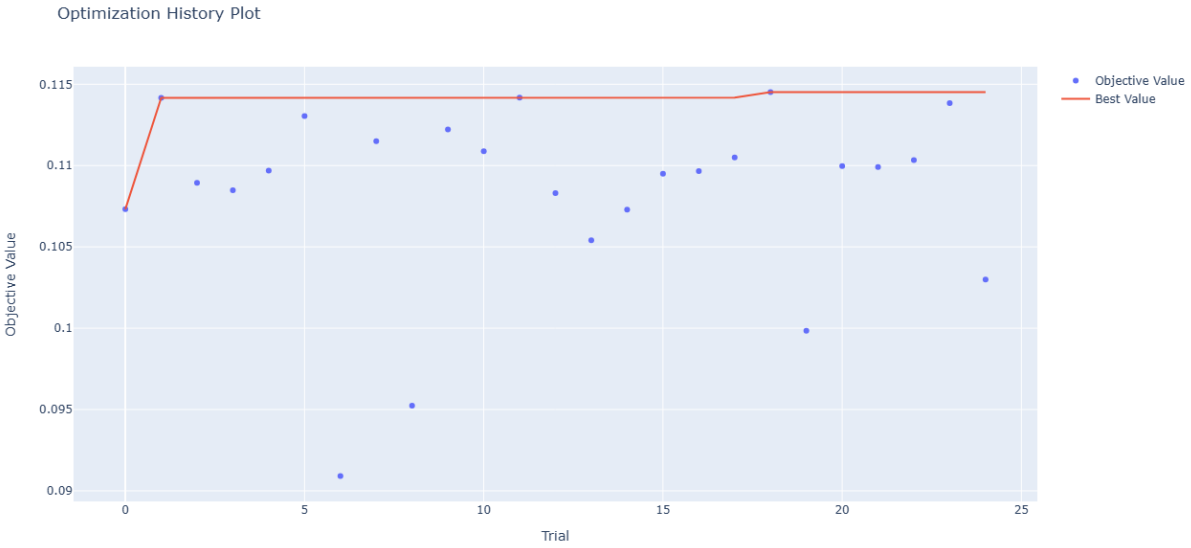



Figure 30. XGBoost with oversampling optimization history plot.

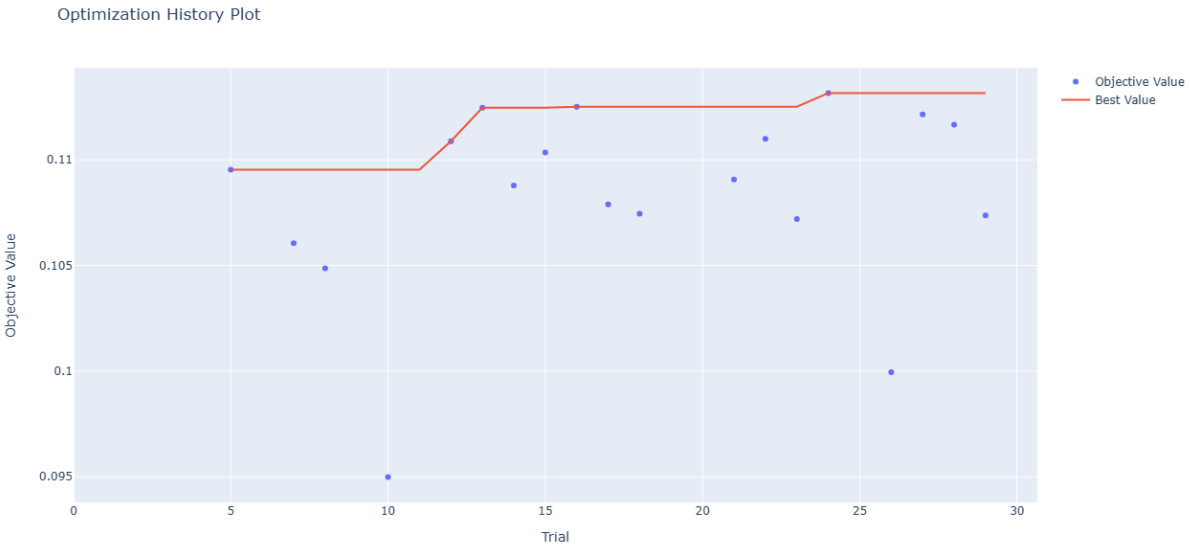



Figure 31. CatBoost without sampling optimization history plot.

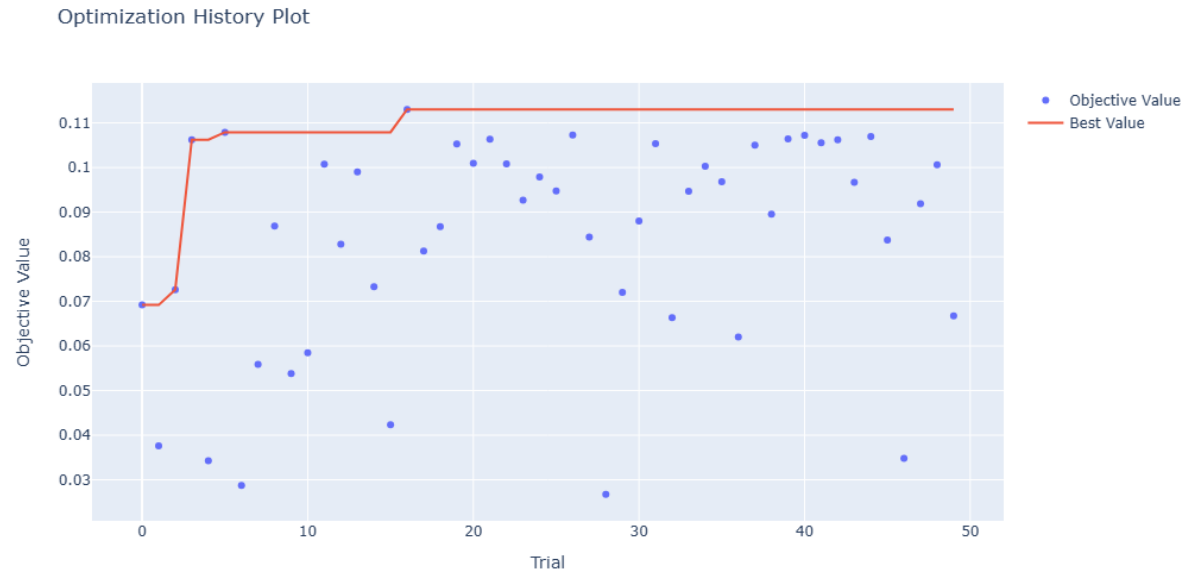



Figure 32. LightGBM without sampling optimization history plot.

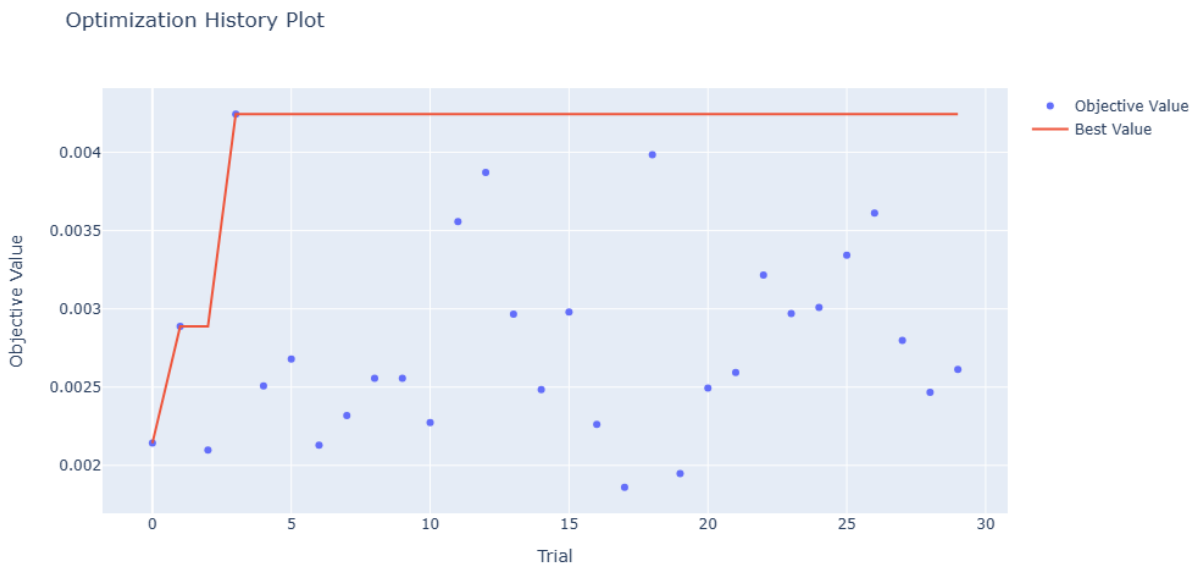



Figure 33. Isolation Forest with undersampling optimization history plot.

# III Optuna parameters importance plots

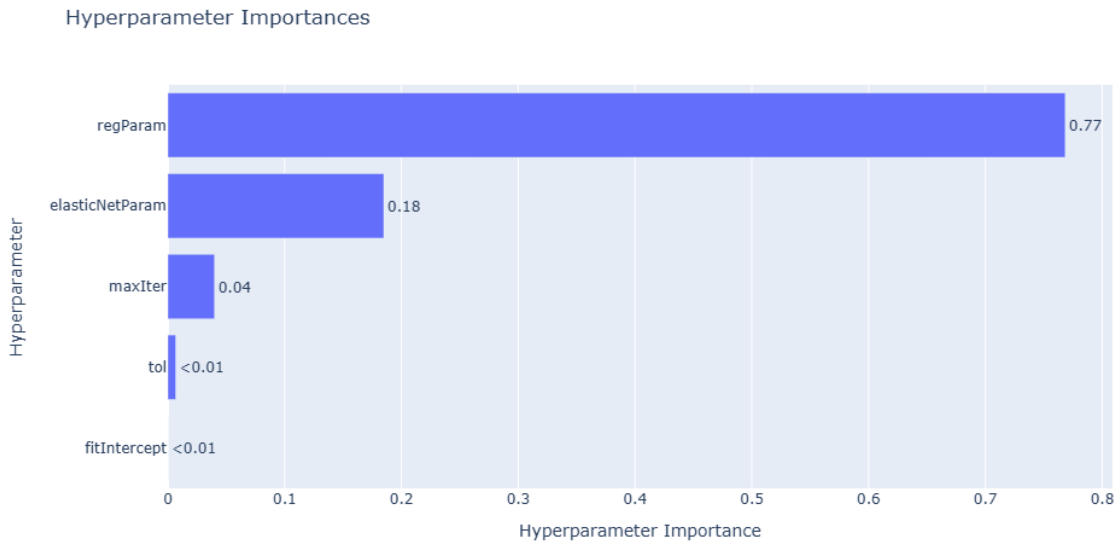



Figure 34. Logistic regression with adjusted weights parameters importance plot.

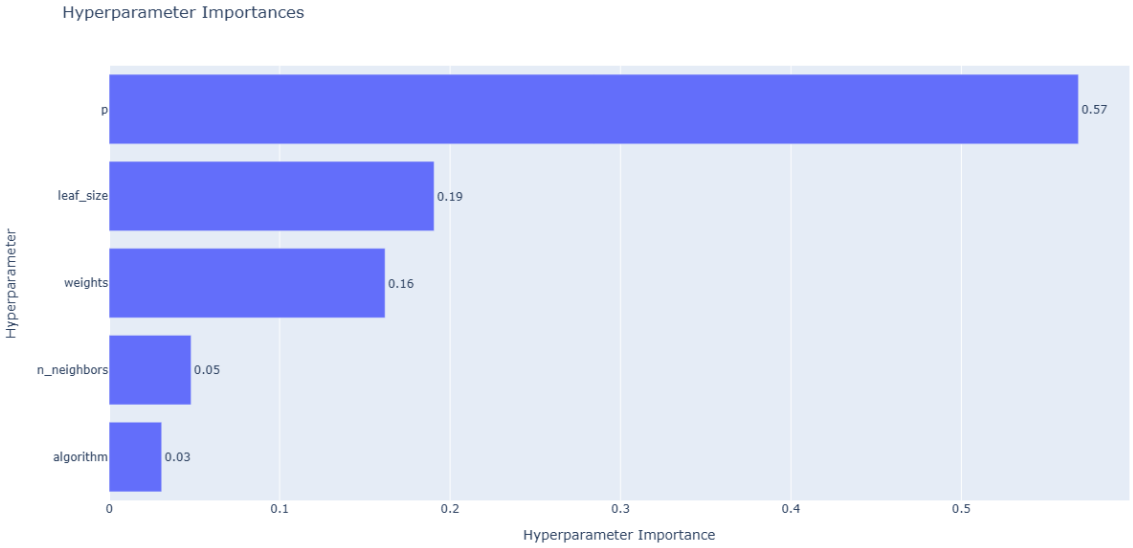



Figure 35. K-nearest neighbours with undersampling parameters importance plot.

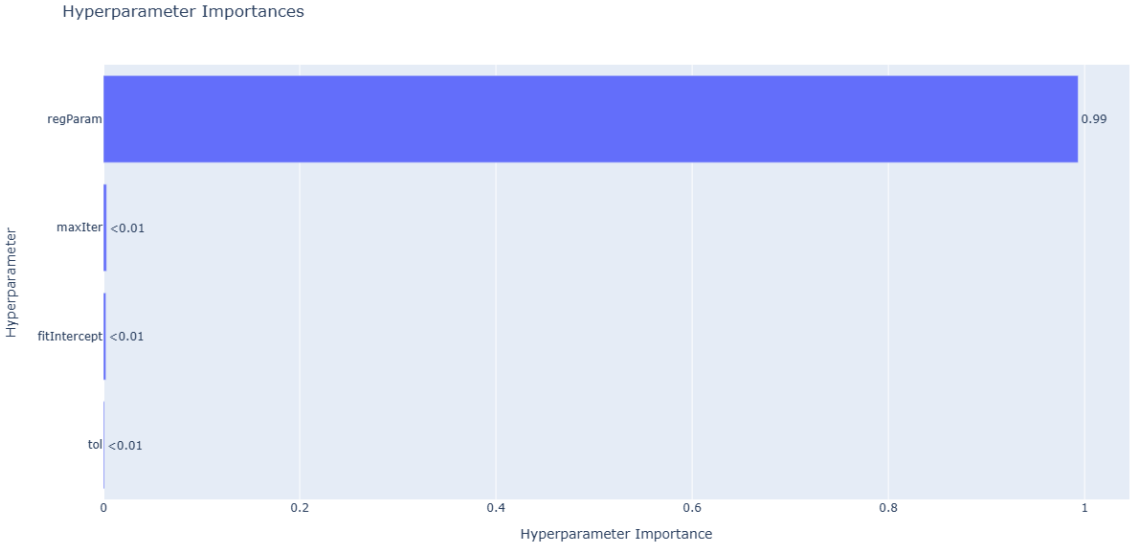



Figure 36. Linear SVM with oversampling parameters importance plot.

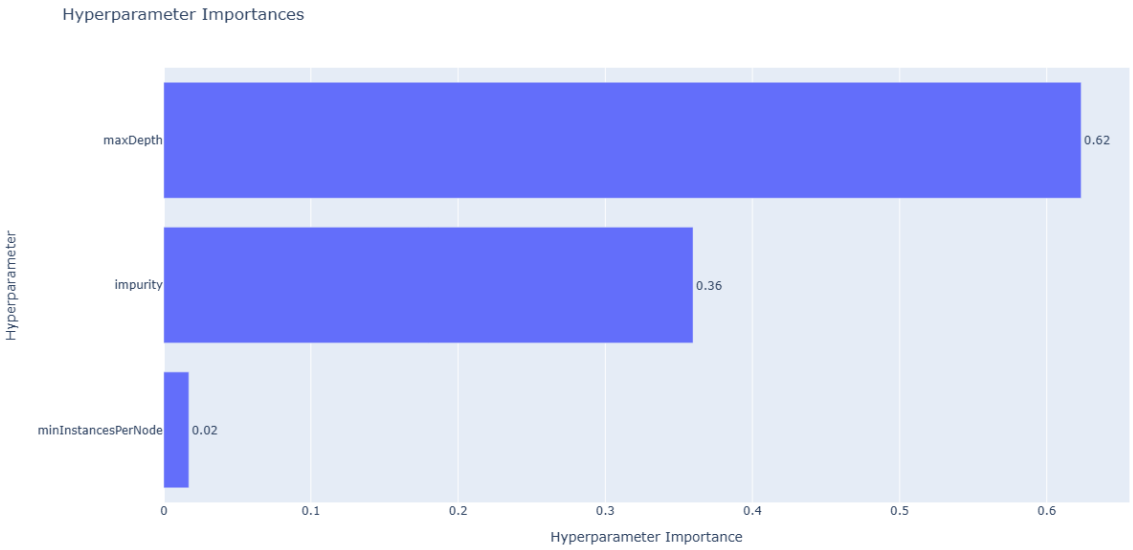



Figure 37. Decision tree with oversampling parameters importance plot.

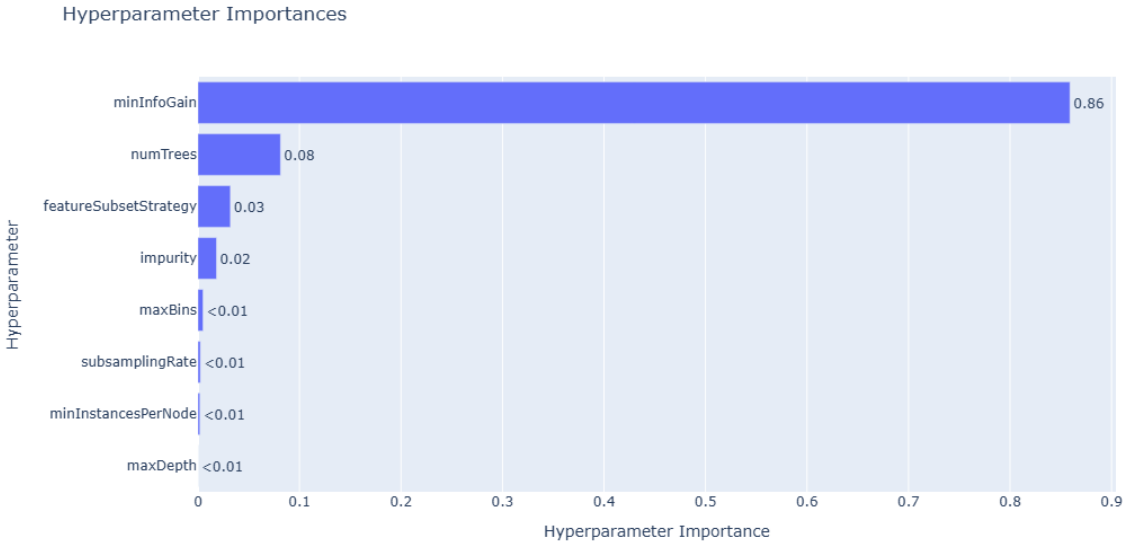



Figure 38. Random forest without sampling parameters importance plot.

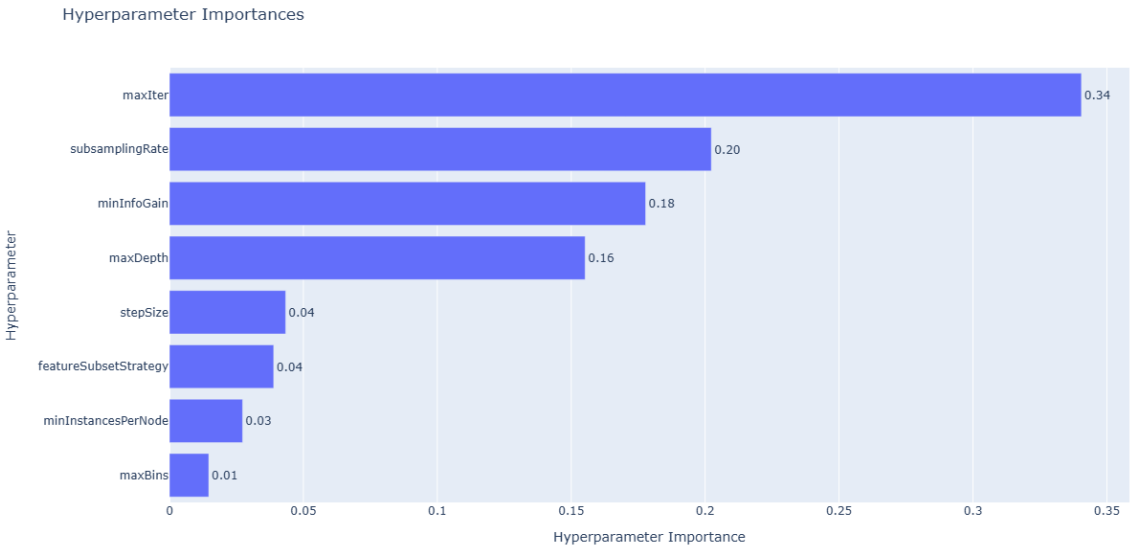



Figure 39. Gradient boosted trees with adjusted weights parameters importance plot.

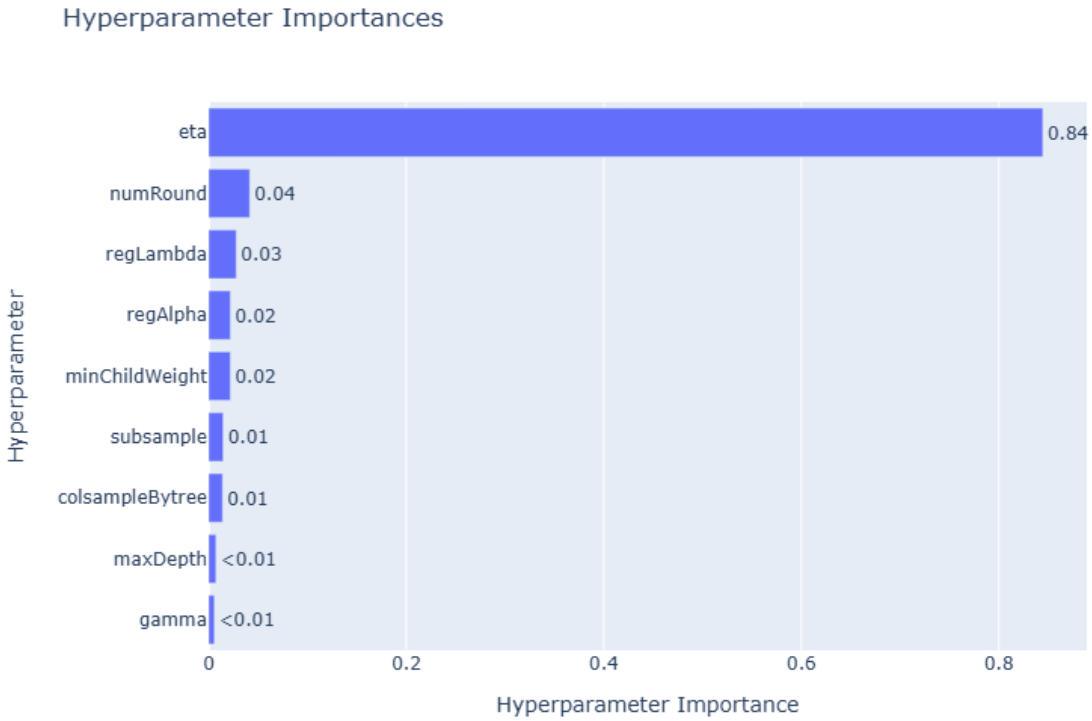



Figure 40. XGBoost with oversampling parameters importance plot.

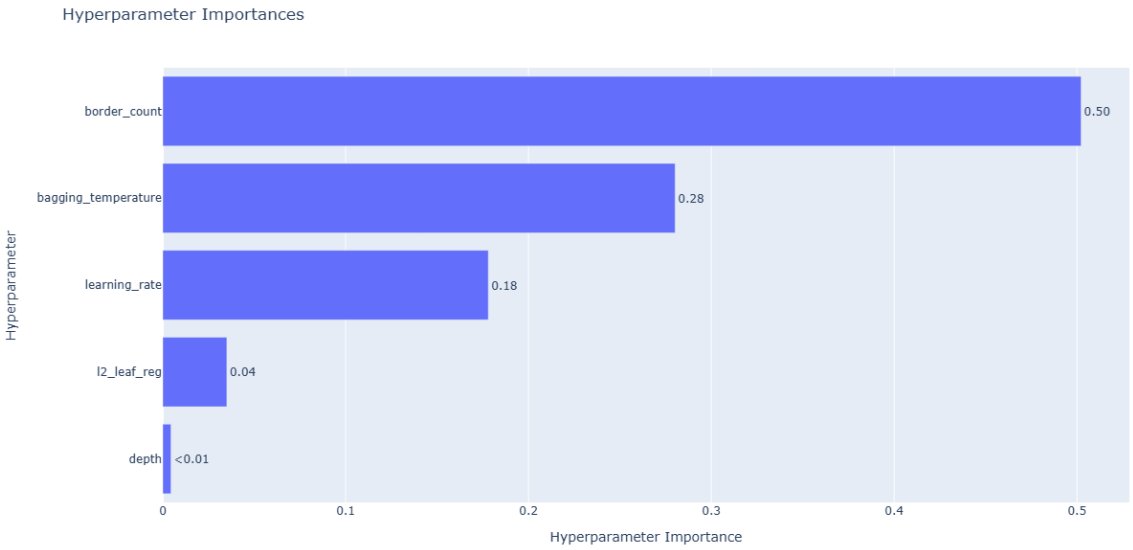



Figure 41. CatBoost without sampling parameters importance plot.

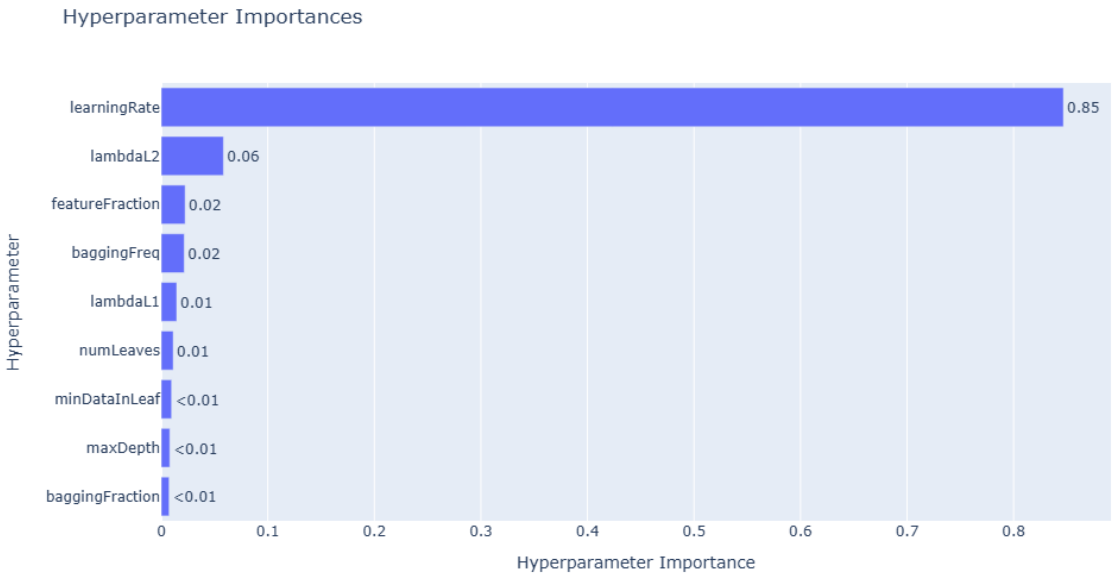



Figure 42. LightGBM without sampling parameters importance plot.

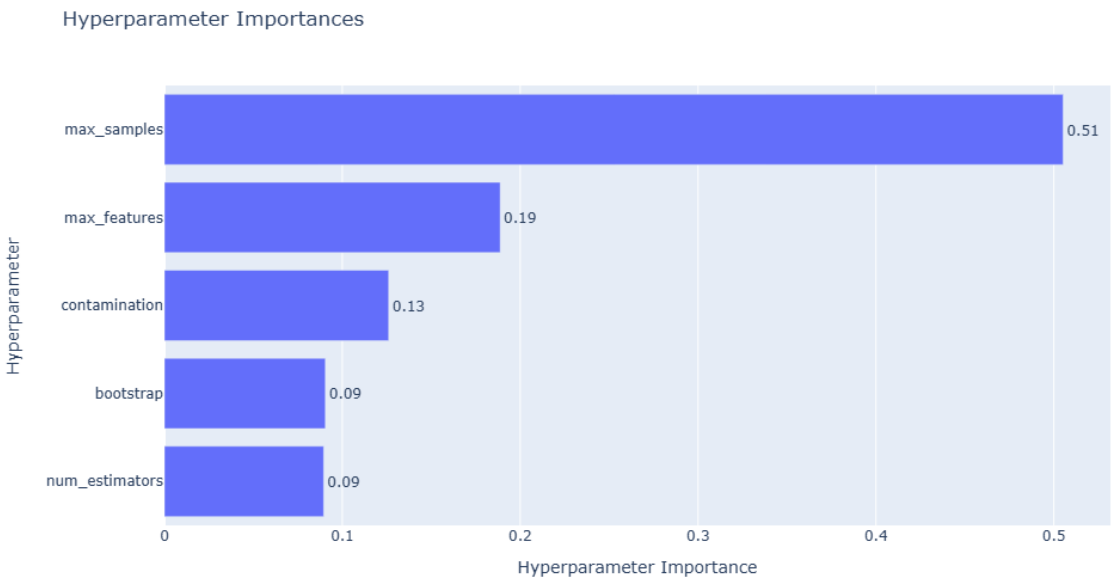



Figure 43. Isolation Forest with undersampling parameters importance plot.

# IV Optuna parallel coordinates plots

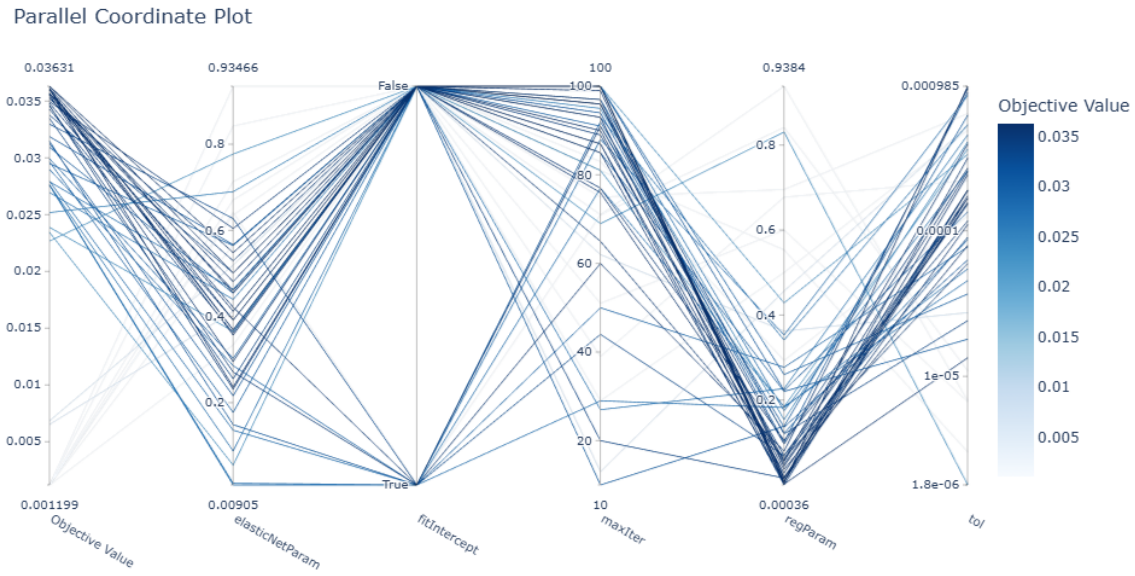



Figure 44. Logistic regression with adjusted weights parallel coordinate plot.

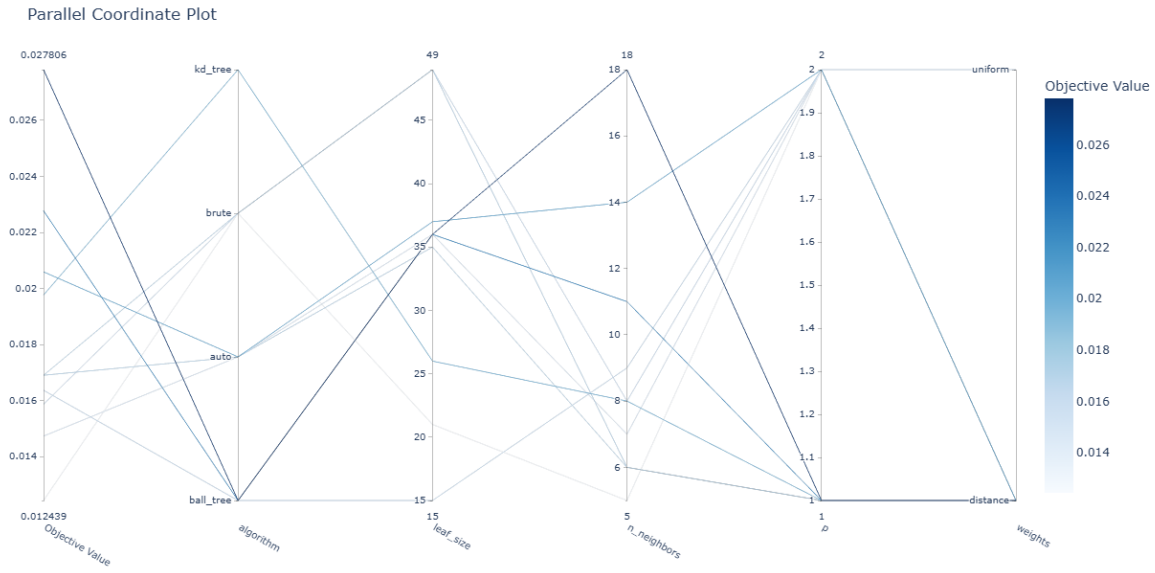



Figure 45. K-nearest neighbours with undersampling parallel coordinate plot.

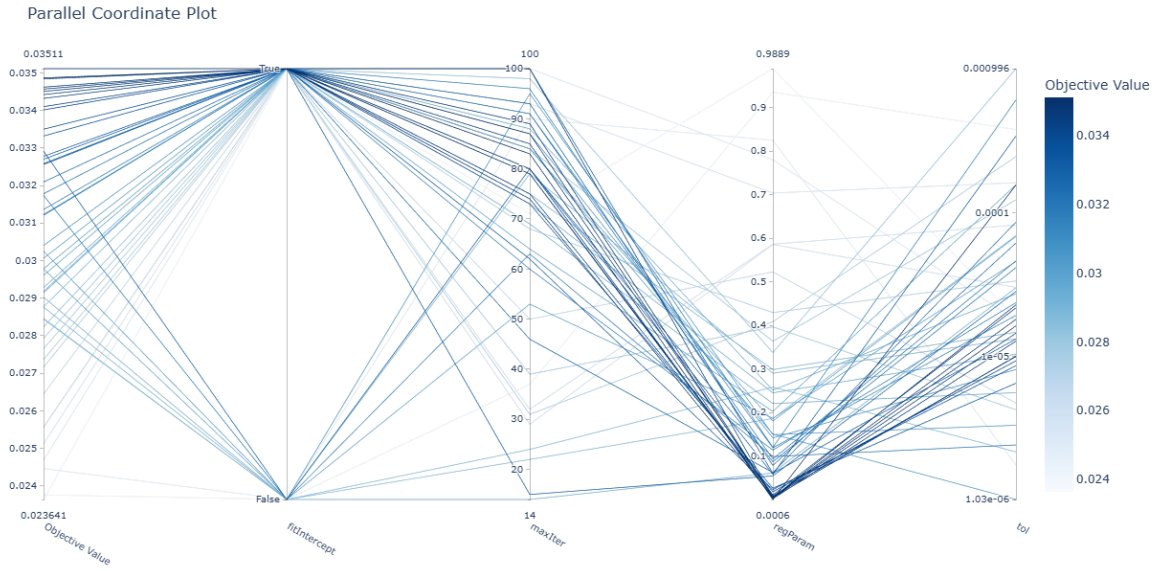



Figure 46. Linear SVM with oversampling parallel coordinate plot.

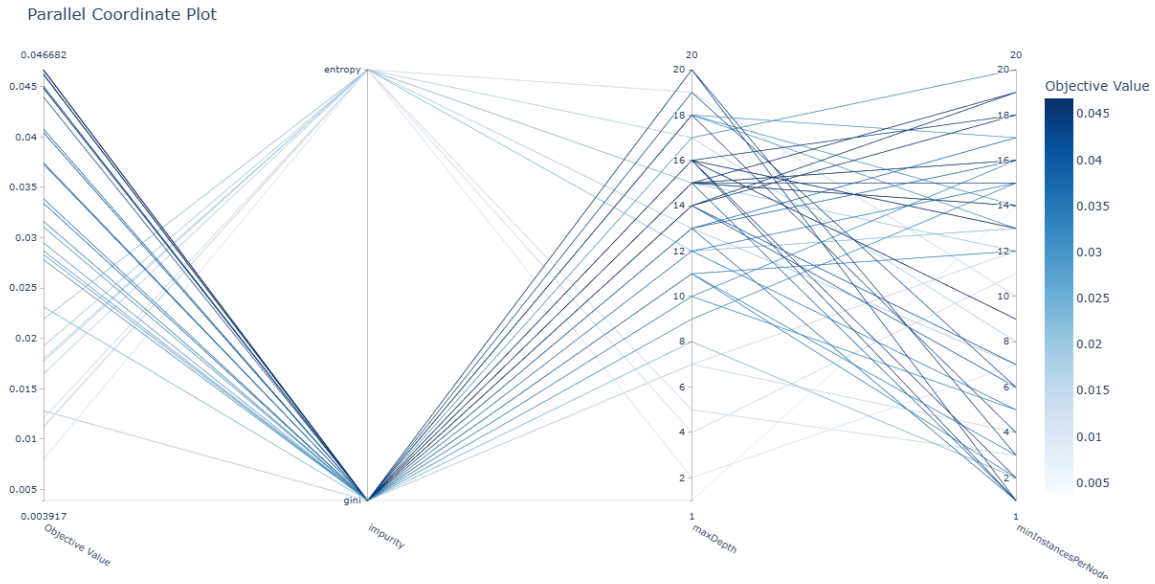



Figure 47. Decision tree with oversampling parallel coordinate plot.

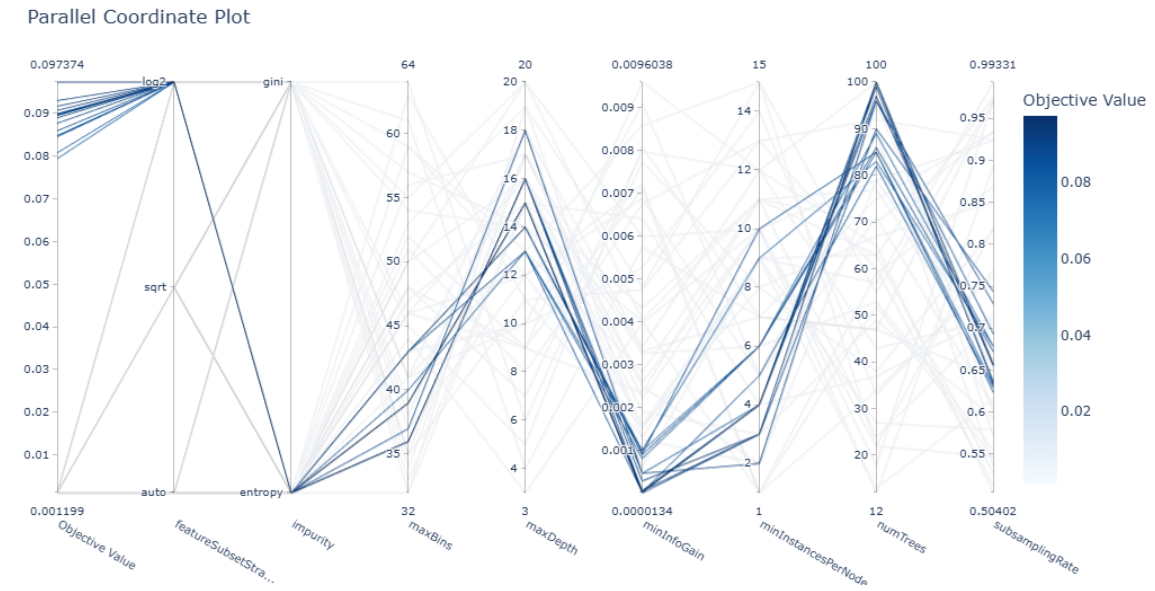



Figure 48. Random forest without sampling) parallel coordinate plot.

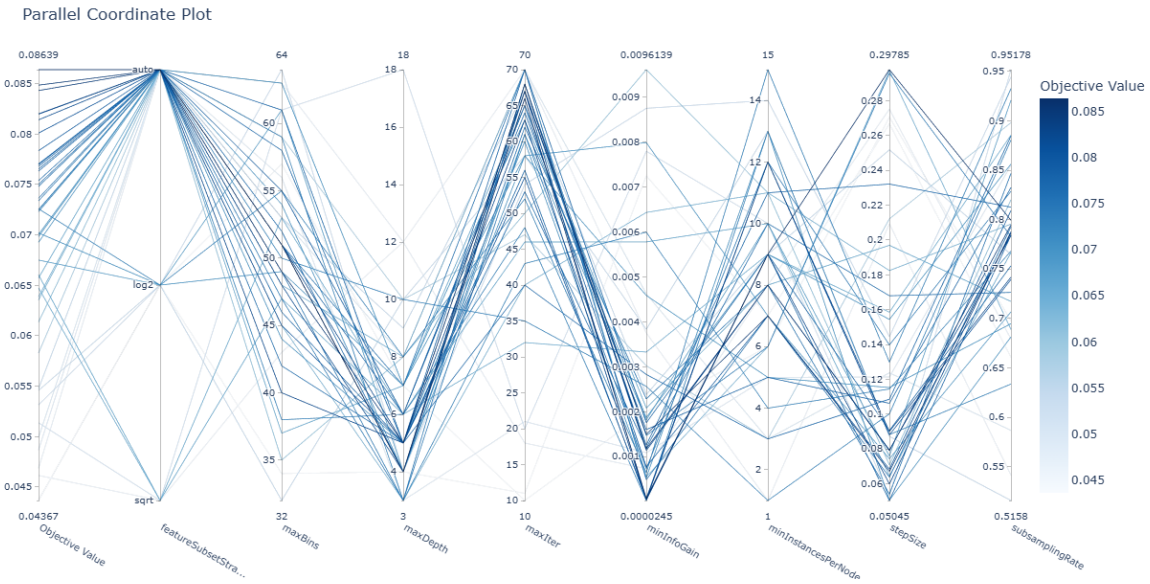



Figure 49. Gradient boosted trees with adjusted weights parallel coordinate plot.

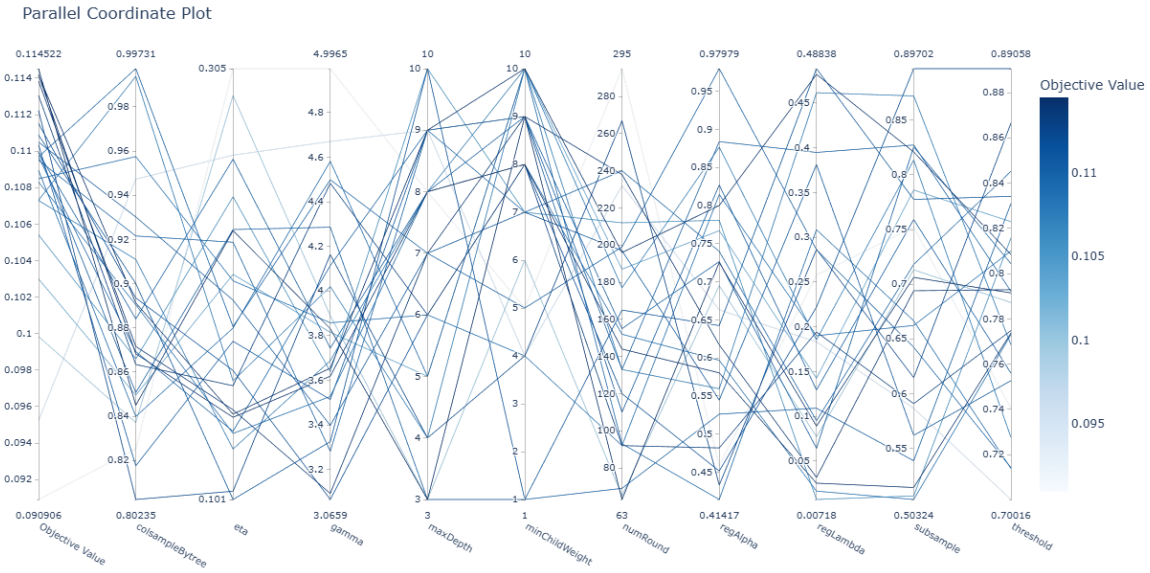



Figure 50. XGBoost with oversampling parallel coordinate plot.

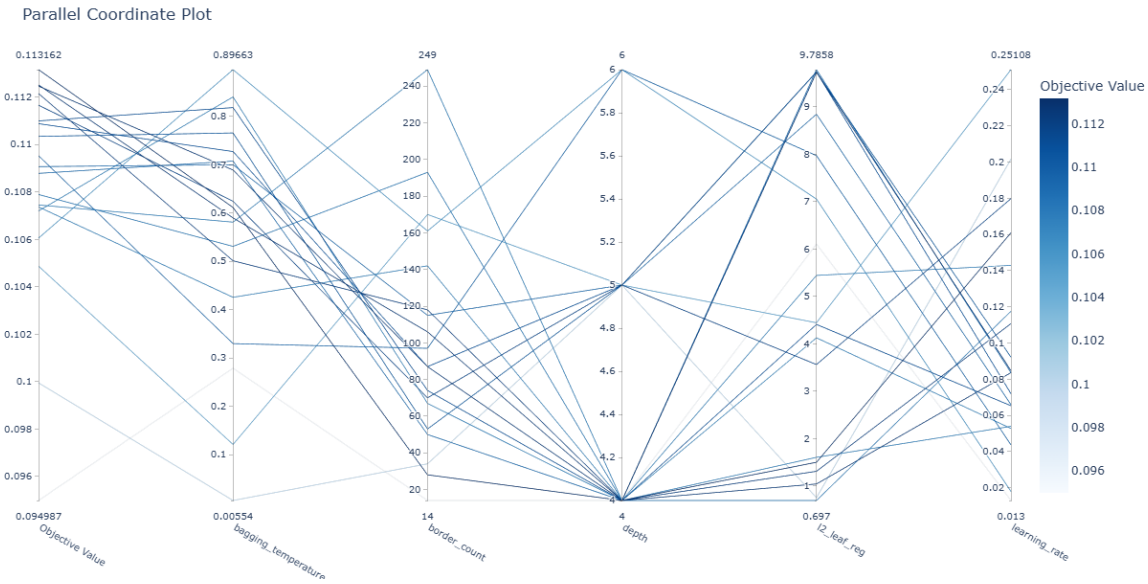



Figure 51. CatBoost without sampling parallel coordinate plot.

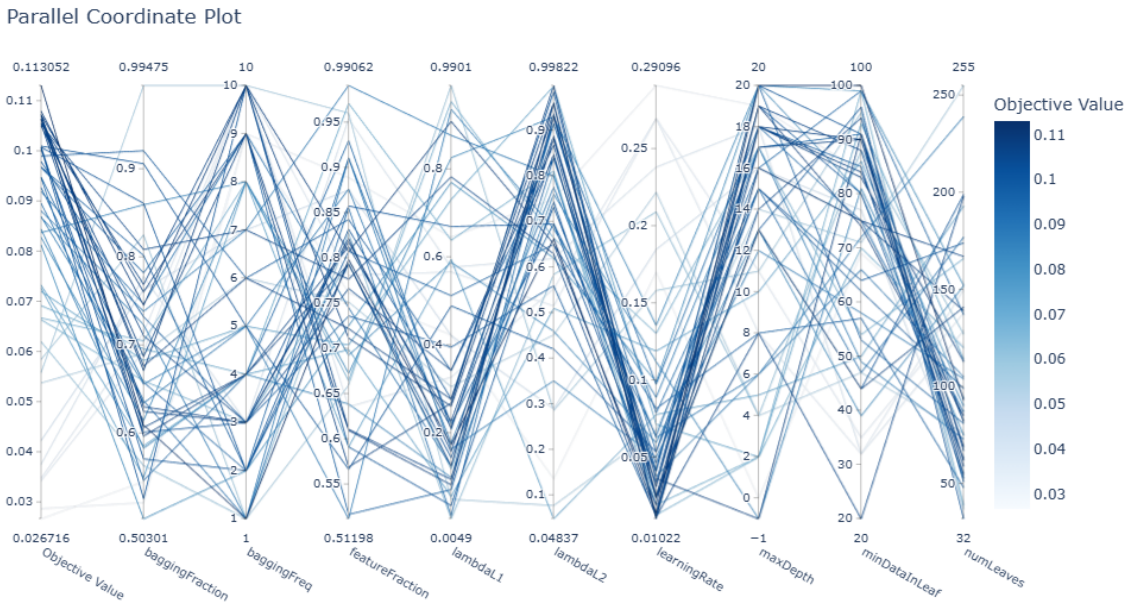



Figure 52. LightGBM without sampling parallel coordinate plot.

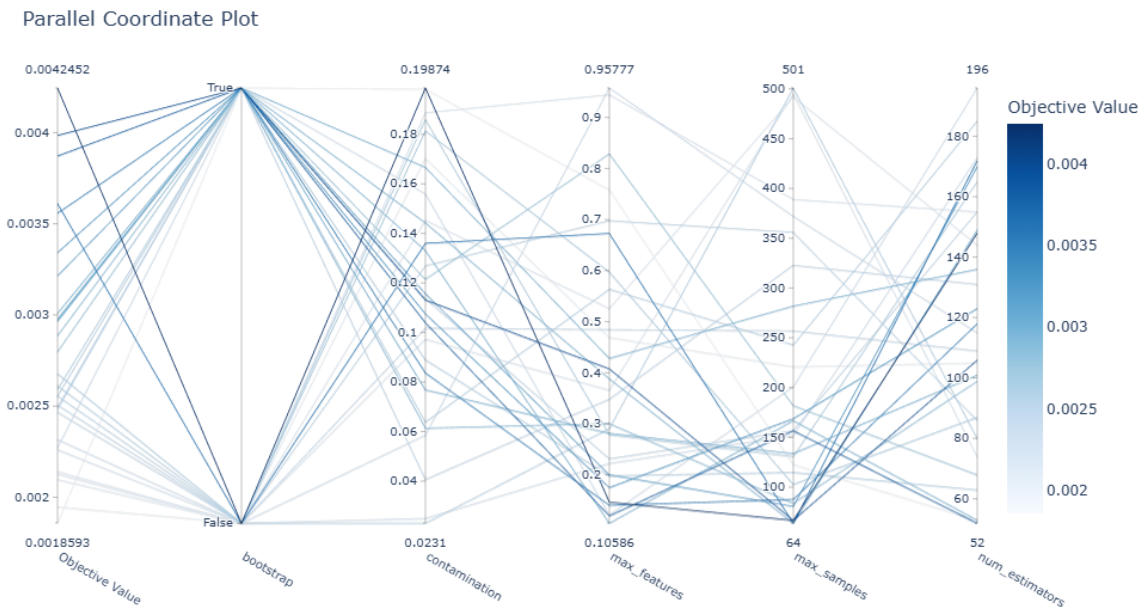



Figure 53. Isolation Forest with undersampling parallel coordinate plot.

# V Licence

## Non-exclusive licence to reproduce thesis and make thesis public

We, **Annabel Hiiu and Anti Karumaa**,

1. herewith grant the University of Tartu a free permit (non-exclusive licence) to

   reproduce, for the purpose of preservation, including for adding to the DSpace digital archives until the expiry of the term of copyright,

   **Machine Learning Based Risk Scoring for Money Mule Detection**,

   supervised by Kristo Raun, Anna Martignano and Alexander Jöhnemark.

2. We grant the University of Tartu a permit to make the work specified in p. 1 available to the public via the web environment of the University of Tartu, including via the DSpace digital archives, under the Creative Commons licence CC BY NC ND 3.0, which allows, by giving appropriate credit to the author, to reproduce, distribute the work and communicate it to the public, and prohibits the creation of derivative works and any commercial use of the work until the expiry of the term of copyright.

3. We am aware of the fact that the author retains the rights specified in p. 1 and 2.

4. We certify that granting the non-exclusive licence does not infringe other persons' intellectual property rights or rights arising from the personal data protection legislation.

Annabel Hiiu and Anti Karumaa

***19.05.2025***