

Authorize debiting transactions from accounts without sufficient funds

Eduardo António Leite da Silva Gonçalves

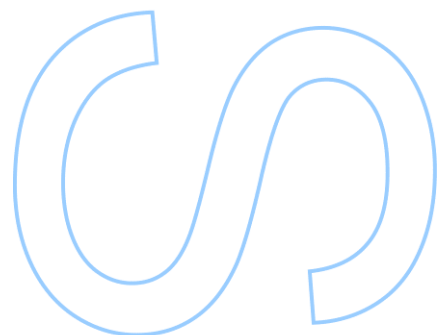
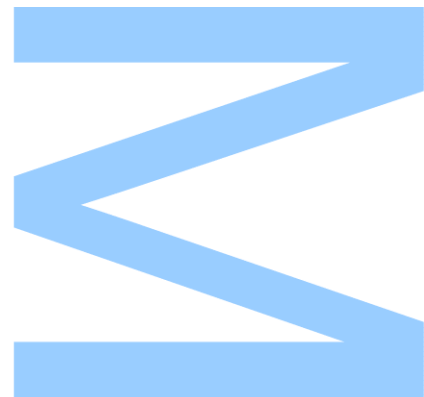
Master in Engineering Physics

Department of Physics and Astronomy

Faculty of Sciences of the University of Porto

Faculty of Engineering of the University of Porto

2025



Authorize debiting transactions from accounts without sufficient funds

Eduardo António Leite da Silva Gonçalves

Project report carried out as part of the Master
in Engineering Physics

Department of Physics and Astronomy

2025

Supervisor

Prof. Dr. João Manuel Viana Parente Lopes, Faculty of Sciences

*I miss the snow of my home.
The calmness of the white
and the soundless stone
that supported me, with no might.*

Luciano Santos

Dedicated to my close family and friends.

Acknowledgments

First, I would like to express my gratitude to my thesis supervisor, João Viana Lopes, whose expertise, guidance and patience were crucial to the successful completion of this work. His mentorship has been truly invaluable and I am truly fortunate to have learned so much under his guidance.

I am also really thankful to my internship supervisor for his insightful contributions in the business context. His experience and perspective have greatly enriched this project and contributed significantly to my professional and personal development.

I would also like to extend my thanks to the incredible team I had the pleasure of working with, as well as the friends I made during my internship. Your warm welcome and genuine fellowship greatly enriched my experience and made this journey so much easier.

To my parents and close family: words cannot adequately express my gratitude for the countless sacrifices you have made on my behalf. Your unwavering support has been my foundation, empowering me to achieve my goals.

Last, but by no means least, I would like to give heartfelt thanks to my girlfriend, Sofia Carvalho. Your constant support, calm presence and affection have been a source of strength and motivation. Thank you for being my anchor throughout this important chapter of my life.

I would also like to express my sincere gratitude to my friends, whose companionship and support during my time outside of work allowed me to truly enjoy and make the most of my final year at university.

This project would not have been possible without the collective support of all those mentioned above. Thank you for making this such an incredible and memorable journey.

Resumo

Este estudo prioriza o desafio estratégico de autorizar transações de débito em contas bancárias com saldo insuficiente, um processo que conjuga o risco de crédito a curto prazo com a exigência de eficiência operacional no setor financeiro. O principal objetivo do projeto consistiu na modernização e automatização do processo designado por **Pay No Pay**, recorrendo a técnicas avançadas de Aprendizagem Automática (Machine Learning) para estimar a probabilidade de regularização de saldo por parte dos clientes.

Para capturar a heterogeneidade dos perfis de clientes bancários, foi adotada uma estratégia de segmentação que os agrupa em três categorias distintas: Clientes Individuais, Empresários em Nome Individual e Empresas. Esta abordagem permitiu a construção de modelos preditivos especializados, ajustados às especificidades comportamentais e financeiras de cada segmento. A fase de pré-processamento dos dados contemplou um estudo cuidadoso das variáveis, a anonimização de dados sensíveis e a definição de uma nova variável alvo, baseada no comportamento real das contas bancárias no período de 30 dias subsequente à transação.

Foram implementados diversos algoritmos de classificação, com recurso ao Optuna para realizar a otimização de hiperparâmetros, e concretizados testes rigorosos ao seu desempenho. Entre os modelos aplicados incluem-se Decision Tree, Random Forest, XGBoost, LightGBM e CatBoost. A avaliação do desempenho dos modelos foi conduzida com recurso a métricas como a AUC-ROC, análises da matriz de confusão e ferramentas de interpretabilidade como os valores SHAP. Os resultados evidenciaram um desempenho superior dos métodos de *Ensemble*, especialmente os baseados em *gradient boosting*, com destaque para o modelo LightGBM, que demonstrou a mais elevada precisão nas previsões efetuadas.

Além disso, foi introduzido o conceito de *Shadow Limit*, uma proposta de extensão operacional pensada para automatizar a aprovação de transações de baixo risco. Os resultados preliminares são promissores e sugerem que este mecanismo pode ter um impacto bastante relevante na eficiência do processo.

Em síntese, este trabalho reforça o potencial da aplicação de soluções baseadas em Aprendizagem Automática em contextos financeiros particularmente exigentes. Os contributos apresentados constituem uma base sólida para o desenvolvimento futuro de sistemas inteligentes de apoio à decisão em gestão de crédito, que combinem escalabilidade, explicabilidade e que se adequem ao comportamento real dos clientes.

Palavras-chave: Aprendizagem Automática; Algoritmos de Classificação; Métodos de Ensemble; Pay No Pay; Decisão Financeira Automatizada; Risco de Crédito.

Abstract

This study addresses the strategic challenge of authorizing debit transactions from accounts with insufficient funds, a critical decision-making process that lies at the intersection of short-term credit risk and operational efficiency in the financial sector. The project aims to modernize and automate the reputed **Pay No Pay** framework by applying advanced Machine Learning (ML) techniques to predict customer repayment likelihood, thus enabling more data-driven, timely and accurate decisions.

To capture the heterogeneous nature of banking clients, a segmentation strategy was adopted, classifying customers into Individuals, Entrepreneurs and Companies. Each group was modeled separately to better reflect their distinct financial behaviors and operational profiles. The data preprocessing phase involved the careful engineering of features, the anonymization of sensitive information and the definition of a new target variable grounded in the actual account balance behavior within a 30-day post-transaction window.

Multiple classification algorithms were implemented using the Optuna framework for hyperparameter optimization, and were then rigorously evaluated. These algorithms included Decision Tree, Random Forest, XGBoost, LightGBM and CatBoost. Performance assessment relied on AUC-ROC, confusion matrix analysis and interpretability tools such as SHAP values. The results revealed that ensemble methods, especially gradient boosting models, with LightGBM standing out as the top performer, consistently outperformed baseline approaches, offering strong predictive accuracy without sacrificing model transparency.

Beyond the core modeling work, this project introduced the concept of a Shadow Limit as an operational extension to automate low-risk transaction approvals. The exploratory results indicate that this mechanism has the potential to significantly enhance the process, while maintaining credit quality and ensuring a positive customer experience.

In conclusion, this work highlights the real potential and benefits of integrating ML solutions into financial decision-making processes that are both high-risk and time-sensitive. It provides a solid starting point for the development of smarter credit management systems, ones that are not only scalable and aligned with real-world client behavior, but also transparent and easy to understand.

Keywords: Machine Learning; Classification Algorithms; Ensemble Methods; Pay No Pay; Automated Financial Decision-Making; Credit Risk.

Contents

List of Figures	vi
List of Tables	vii
List of Abbreviations	viii
1 Introduction	1
1.1 Project Purpose	2
1.2 Final Report and Data Presentation	2
2 Description of the application domain	4
2.1 State of Art	4
2.2 Pay No Pay	7
2.3 Client Segmentation in Pay No Pay	9
3 Data Preprocessing	10
3.1 Dataset Creation	10
3.2 Target Value	12
4 Machine Learning Methods	17
4.1 Decision Tree	17
4.2 Ensemble Methods	20
4.2.1 Bagging	21
4.2.2 Boosting	23
4.2.3 Gradient Boosting	23
4.3 Random Forest	25
4.4 XGBoost	27
4.5 LightGBM	28
4.6 CatBoost	30
5 Model Workflow	32
5.1 Model preparation	32
5.1.1 Hyperparameter Tuning	32
5.1.2 Cross Validation	36
5.2 Model evaluation	38

5.2.1	Confusion Matrix	38
5.2.2	AUC-ROC	40
5.2.3	Decision Threshold	41
5.2.4	Feature Importance and SHAP Analysis	42
6	Experiments and Results	46
6.1	Model Comparison	46
6.1.1	Individuals	46
6.1.2	Entrepreneurs	48
6.1.3	Companies	50
6.2	Model Final Assumptions	52
6.2.1	Individuals	53
6.2.2	Entrepreneurs	58
6.2.3	Companies	60
6.3	Future Improvements	63
7	Extra Work	65
7.1	Shadow Limit	65
7.2	Shadow Limit Evaluation	67
7.3	Final Assumptions	68
8	Conclusion	69
	References	76
A	Confusion Matrix for Individuals	77
B	Confusion Matrix for Entrepreneurs	78
C	Confusion Matrix for Companies	79
D	Beeswarm Plots for Entrepreneurs and Companies	80

List of Figures

3.1	Non-Null Values in the Busin Variables.	12
3.2	New Target Value Revision.	14
3.3	New Target Value Distribution.	15
4.1	Example of Decision Tree model [27].	18
4.2	Flowchart of parallel and sequential ensemble [33].	21
4.3	Flowchart of Bootstrap Sampling [35].	22
4.4	Random Forest Scheme [40].	26
5.1	Visual Difference between Hyperparameter Tuning processes [45].	33
5.2	K - Fold train and test split [49].	37
5.3	ROC-AUC Classification Evaluation Metric [52].	40
6.1	Models ROC Curve for Individuals	48
6.2	Models ROC curve for Entrepreneurs	50
6.3	Models ROC curve for Companies	52
6.4	Mean SHAP Value of LightGBM Individuals model	55
6.5	Beeswarm Plot of LightGBM Individuals model	57
6.6	Mean SHAP Value of LightGBM Entrepreneurs model	60
6.7	Mean SHAP Value of LightGBM Companies model	62
7.1	New Pay No Pay Flow	66
D.1	Beeswarm Plot of LightGBM Entrepreneurs model	80
D.2	Beeswarm Plot of LightGBM Companies model	81

List of Tables

3.1 Dataset across all Client Segmentation.	11
3.2 Score distribution.	16
4.1 Decision Tree Hyperparameters	20
4.2 XGBoost Hyperparameters	28
4.3 CatBoost Hyperparameters	31
5.1 Confusion matrix layout for binary classification	38
6.1 Model Performance for Individuals	46
6.2 Model Performance for Entrepreneurs	49
6.3 Model Performance for Companies	51
6.4 LightGBM Confusion matrix for Individuals	53
6.5 Threshold Study for Individuals	54
6.6 LightGBM Confusion matrix for Entrepreneurs	58
6.7 Threshold Study for Entrepreneurs	59
6.8 LightGBM Confusion matrix for Companies	61
6.9 Threshold Study for Companies	61
7.1 Shadow Limit Evaluation	67
A.1 Decision Tree Confusion Matrix for Individuals	77
A.2 Random Forest Confusion Matrix for Individuals	77
A.3 XGBoost Confusion Matrix for Individuals	77
A.4 CatBoost Confusion Matrix for Individuals	77
B.1 Decision Tree Confusion Matrix for Entrepreneurs	78
B.2 Random Forest Confusion Matrix for Entrepreneurs	78
B.3 XGBoost Confusion Matrix for Entrepreneurs	78
B.4 CatBoost Confusion Matrix for Entrepreneurs	78
C.1 Decision Tree Confusion Matrix for Companies	79
C.2 Random Forest Confusion Matrix for Companies	79
C.3 XGBoost Confusion Matrix for Companies	79
C.4 CatBoost Confusion Matrix for Companies	79

List of Abbreviations

ML	Machine Learning
XGBoost	eXtreme Gradient Boosting
LightGBM	Light Gradient Boosting Machine
CatBoost	Categorical Boosting
TPE	Tree-structured Parzen Estimator
EI	Expected Improvement
TN	True Negative
FN	False Negative
TP	True Positive
FP	False Positive
NPV	Negative Predictive Value
ROC	Receiver Operating Characteristic
AUC	Area Under the ROC Curve
TPR	True Positive Rate
FPR	False Positive Rate
SHAP	SHapley Additive exPlanations

1 Introduction

As currency emerged during the era of great civilizations, the acts of lending, borrowing and safekeeping money became almost inevitable. It is believed that the first banking operations in history were developed in the Phoenician civilization [1]. However, the term Bank was coined by the Romans and referred to the *bench* where currency exchanges took place.

Over time, financial institutions began accepting monetary deposits and issuing a kind of certificate in return. Later, they realized that withdrawals did not always match deposits, meaning that there was always a reserve available for circulation. This realization led to the practice of lending money in exchange for interest [2].

This marked a turning point in financial history, as bankers evolved from mere money changers into influential financial intermediaries. By satisfying the growing demand for liquidity among businesses and individuals, they now play a central role in shaping the modern financial system.

Sheikh, Goel, and Kumar (2020) emphasize that although banks provide a wide range of financial services, their main source of income is obtained from the credit they extend to clients, including not only loans, but also other forms of credit [3]. These credit products generate profit primarily through the interest and fees applied.

Nevertheless, lending credit entails substantial responsibility and risk. Financial institutions do not grant credit indiscriminately, they rigorously evaluate the potential risk of default associated with each applicant before approval.

In this regard, integrating Machine Learning (ML) techniques into credit risk assessment processes has proven to be highly beneficial. These models can process and analyze large volumes of financial and behavioral data to more accurately estimate the likelihood of default, and as a result, financial institutions are equipped to make more informed, consistent and data-driven decisions, reducing exposure to risk [4].

1.1 Project Purpose

The proposed topic for the internship and dissertation development is **Authorize debiting transactions from accounts without sufficient funds**.

The main objective is to apply classification algorithms to support decision-making in these high-risk scenarios. By integrating ML, financial institutions can estimate the likelihood of repayment more accurately, reduce exposure to default and enhance the efficiency of their transaction approval strategies.

In such contexts, making informed and data-driven decisions is crucial to maintain a balance between risk management and customer satisfaction.

This work is based on the **Pay No Pay** decision framework, which is a critical process where financial institutions must evaluate whether to approve or reject transactions in the absence of sufficient funds. To improve the effectiveness and reliability of this process, the study employs advanced and *state-of-the-art* ML techniques, aligned with the latest developments in the field.

1.2 Final Report and Data Presentation

This internship report consists of eight distinct chapters, carefully structured to present the methodologies used, the obtained results and the final assumptions in a clear and systematic way.

Chapter 2 provides a comprehensive overview of the application domain, defining the context and objectives of the project. It begins with a literature review on contemporary approaches to credit risk modeling, with particular attention given to ML and ensemble methods. The chapter then delves into the Pay No Pay decision process, emphasizing its importance and operational challenges. Finally, it introduces the client segmentation strategy, dividing customers into Individuals, Entrepreneurs and Companies, which serves as a critical foundation for the development of the subsequent stages of the work.

Chapter 3 focuses on data preprocessing and exploratory analysis, outlining the creation of customized datasets aligned with the previously identified customer segments. It also introduces the construction of a new target variable to support the modeling phase.

Chapter 4 presents the ML methods applied in the study. It covers Decision Tree, Bagging techniques such as Random Forest, and also the Gradient Boosting algorithms including XGBoost, LightGBM and CatBoost.

Chapter 5 outlines the model development process, including cross-validation, hyperparameter tuning, and evaluation using performance metrics such as the confusion matrix and

AUC-ROC. Additionally, the chapter explores the effects of adjusting the decision threshold and introduces SHAP analysis to enhance model interpretability.

Chapter 6 delves into the work experiments and results, analyzing the comparison of all models across the three customer segments. This chapter also presents the final model assumptions and discusses potential future improvements to enhance model performance.

Chapter 7 presents additional work based on the Shadow Limit concept, designed to pre-approve low-risk transactions for individual clients.

Finally, Chapter 8 outlines the final assumptions, summarizing the key findings of the report and outlining potential directions for further research and development.

To guarantee confidentiality, all the data in this report has been anonymized to maintain strict privacy for both the company associated with this internship (whose name will remain undisclosed) and any relevant information related to it.

2 Description of the application domain

This chapter presents a comprehensive overview of the current state of the art in credit risk modeling, with a particular focus on ML and ensemble methods. It also introduces the specific context of the **Pay No Pay** decision process, outlining its operational challenges and strategic relevance.

Furthermore, the chapter details the client segmentation strategy adopted in this work, which is essential for developing tailored and effective predictive models.

2.1 State of Art

The risk associated with the decision-making process is coupled with the difficulty in distinguishing between creditworthy applicants from unworthy applicants (those who are prone to default, fail to pay their obligations). As the most recent financial crisis resulted in huge losses world wide, financial institutions increased the attention paid to credit risk prediction, and more resources are allocated to credit risk management in order to lend to creditworthy borrowers and protect themselves from potential losses resulting from customer defaults.

To address this challenge, credit scoring has become one of the most widely adopted methodologies for assessing credit risk. The main purpose of a credit scoring model is to classify applicants into one of two broad groups: *good* customers (those who are expected to fulfill their obligation) and *bad* customers (those predicted to default). This binary classification framework forms the backbone of these automated credit decision-making processes (Lee et al, 2002) [5].

Initially, creditworthiness assessments relied on manual evaluation, particularly through the use of the 5Cs framework: character, capital, collateral, capacity and conditions. However, this approach suffered from significant inconsistencies, as the results often varied depending on the individual who performed the evaluation.

As credit demand increased and computational capabilities evolved, more objective and scalable methods were introduced. This technological shift enabled the automation of credit decisions, leading to improved efficiency and consistency. The transition from manual assessments to data-driven credit scoring systems was essential, as it reduce default rates by more than 50% [6].

In this context, the selection of appropriate methodologies for credit risk modeling is crucial and, as such, a comparative analysis of different approaches proves to be highly valuable. Therefore, this subsection provides a detailed analysis of the evolution of applicable ML meth-

ods and their implementation in the context of credit risk assessment.

During the 1980s and 1990s, statistical methodologies gained significant importance in the field of risk prediction. Among them, logistic regression became the dominant technique due to its versatility in handling different types of variables, its simplicity and its high degree of interpretability [7].

However, by the late 1990s, the application of ML techniques began to gain relevance as researchers and practitioners began to look for more accurate and robust models capable of capturing complex and non-linear patterns in data.

Decision trees emerged as a fundamental tool in statistics and data analysis, widely used for both classification and regression tasks. One of the earliest significant contributions to this field was made by Hunt & Stone (1996), where they described early methods for decision tree induction [8].

The continued development of algorithms and supporting software contributed to the global adoption of decision trees in various domains within financial institutions. A major advancement came in 2001, when Leo Breiman introduced the Random Forest technique, which constructs an ensemble (bagging) of decision trees to enhance prediction accuracy and address overfitting issues often present in individual trees [9].

Building upon this foundation, Jerome Friedman later introduced Gradient Boosting Machines, an ensembling technique, in which each successive tree is trained to correct the errors made by the previous ones, resulting in highly accurate and adaptive models capable of capturing subtle relationships in the data [10].

These Ensemble Methods have emerged as a highly effective strategy for enhancing model performance by combining multiple base learners (specially decision tree) to reduce variance, mitigate bias and improve generalization capabilities. Among the most prominent ensemble techniques are the gradient boosting algorithms such as XGBoost (eXtreme Gradient Boosting), LightGBM (Light Gradient Boosting Machine) and CatBoost (Categorical Boosting), besides as already mentioned, Random Forest (based on bagging). These models have gained substantial recognition for their strong predictive performance and computational efficiency, particularly in structured data applications.

These developments laid the groundwork for modern ensemble methods, which continue to play a crucial role in predictive modeling, especially in domains such as decision-making in high-risk scenarios, where accuracy and model reliability are of greatest importance.

According to Li and Chen (2020), ensemble learning methods can deliver superior predictive performance compared to traditional single-model approaches in credit scoring tasks [11].

These methods do not depend on strict statistical assumptions and instead leverage the combined strength of multiple base learners to capture complex patterns in the data. In contrast, traditional statistical methods often rely on assumptions about data distribution and variable relationships, which, if not met, may significantly impair their predictive accuracy. Ensemble models offer a more flexible and robust alternative for credit risk evaluation in financial institutions.

Several related studies provide a foundation for this project by exploring the use of ML in credit risk assessment and financial decision-making. Simão (2023) notes that the application of ML to credit scoring and financial risk modeling remains at an early stage, particularly with respect to ensemble methods [12]. Nevertheless, preliminary results are promising and support continued research and application, given the scale and financial implications for institutions.

Pothumsetty (2020) highlights the transformative impact of ML in financial services, particularly in risk management, fraud detection and credit assessment [13]. The study emphasizes that ML technologies contribute to increased operational efficiency and more accurate decision-making. While their integration continues to evolve, these technologies are already proving valuable, reinforcing the need for further development.

Zhu et al. (2024) present an advanced ensemble framework for credit default prediction using LightGBM, XGBoost and local ensemble techniques [14]. This work demonstrates clear improvements in predictive accuracy and model robustness, further validating the role of gradient boosting methods in financial risk analysis.

Similarly, Yu et al. (2024) develop a hybrid credit risk model that combines LightGBM, XGBoost and TabNet, enhanced by the SMOTEENN technique to address class imbalance [15]. Their approach shows that integrating ensemble learning with resampling strategies can significantly increase the effectiveness and generalizability of credit scoring systems, especially in imbalanced datasets.

A particularly new contribution comes from Alagic et al. (2024), who incorporate mental health data into ML-based loan approval models [16]. Their comparative analysis suggests that including psychosocial variables can significantly improve predictive accuracy, highlighting the potential of more holistic and inclusive data strategies in credit risk assessment.

Kozodoi et al. (2025) focus on a fundamental challenge in ML-based credit scoring: sampling bias [17]. They propose a structured framework for training and evaluating models that mitigates bias and promotes fairer and more reliable outcomes. Their work underscores the importance of methodological rigor and data integrity in financial applications.

Ntamwiza and Bwire (2025) have shown that ensemble ML models can outperform tradi-

tional statistical methods in modeling complex behavioral patterns, even when working in the transportation domain [18]. Their findings, derived from a study on biking preferences in Kigali, reinforce the broader applicability of ensemble approaches in domains such as credit risk modeling, where heterogeneity and nonlinear relationships are common.

Although ML applications in credit risk modeling have been widely studied, few works have specifically tackled the problem addressed in this project. Among them, two studies are especially pertinent to this research.

The first study, by Joaquim Costa (2008), investigates the construction of an optimal decision model for the retail (Individual) credit segment [19]. This work conducts a comparative analysis of several ML algorithms, including logistic regression, decision trees and neural networks, and explores different classification strategies, using both binary and ternary target definitions. Although the three class approach was considered, current industry practice has largely converged to binary classification frameworks. This shift reflects a preference for greater operational clarity and aligns more closely with real world credit approval processes, which typically require simple accept/reject decisions.

The second study, by Hélia Costa (2012), adopts a target definition closely aligned with the one used in the present work, but its scope is more limited in terms of methodological diversity [20]. This thesis primarily employs model like decision tree and random forest and does not incorporate more recent developments in ensemble learning. Given the considerable evolution of ML techniques over the past decade, particularly in areas such as model tuning and interpretability, there is a clear opportunity to build upon this foundation using a more comprehensive and technically advanced modeling approach.

Building on these prior studies and by applying these advanced ensemble methods, this project aims not only to improve predictive accuracy and model robustness but also to align credit decision models more closely with the practical requirements of financial institutions. In doing so, this work fills a notable gap in the existing literature and sets the stage for future research that connects methodological rigor with real-world financial decision-making.

2.2 Pay No Pay

The credit decision-making process is a complex area that involves human, social, economic and financial factors. One specific aspect of this domain is the decision to approve or reject transactions (such as direct debits, bill payments or checks) on accounts that lack sufficient funds to complete them. In this work, this decision-making process is referred to as **Pay No Pay**.

A fully automated system capable of issuing an immediate approval or rejection for every transaction would present considerable challenges, particularly in managing credit risk. Any decision to approve a transaction that would result in an overdraft (bank account balance falls below zero) must consider not only the likelihood of repayment but also the potential impact on the customer relationship and the bank's profitability through interest charges and service fees.

The ultimate goal is to automate Pay No Pay decisions in a way that minimizes credit losses and maximizes value, both for the bank and the customer. From the financial institution's perspective, allowing overdrafts is only beneficial when customers regularize their account balance in a short period. This regularization, which refers to restoring the account from a negative to a positive balance, allows the bank to collect applicable service fees and reuse those funds for other profitable activities. Conversely, when a customer fails to regularize the overdraft, the bank faces potential losses, as the account remains in default and the extended credit becomes irrecoverable.

The Pay No Pay process therefore represents a short-term credit underwriting decision that must be made promptly, often within minutes or hours of the transaction request. In this context, applying classification algorithms to transactional and account-level data is essential to distinguish between clients likely to regularize and those who pose a higher default risk.

The proposed models will integrate customer segmentation, historical behavioral analysis and transaction-level data to optimize the approval process. The main objective is to increase operational efficiency, reduce credit risk and ensure profitability, all while improving the customer experience through timely and consistent decision-making.

This process currently relies on a tripartite scorecard, in which final decisions fall into one of three categories: approval, rejection or manual review. The objective of this project is to convert the system into a fully automated binary scorecard (with only approval/rejection), thereby eliminating the need for manual intervention and enhancing overall efficiency.

By analyzing this process over the course of a month, the number of transactions reviewed manually by analysts remained relatively low, accounting for less than **5%** of the total transactions. Although this percentage is small, reducing the reliance on manual interventions will help streamline the process and align it more closely with ongoing efforts toward modernization and automation in other operational workflows.

2.3 Client Segmentation in Pay No Pay

To develop more comprehensive and objective models, clients are segmented according to their characteristics. Within this segmentation, it was possible to identify **Individuals** as well as Business Clients, which include both **Entrepreneurs** and **Companies**.

What clearly distinguishes business clients from individual clients is their significantly higher volume of transactions (both debit and credit), together with the wider variability in transaction amounts. Merging all types of client into a single model risks overlooking specific characteristics that are essential for accurate analysis and decision-making.

Throughout this project, maintaining this segmentation has proven to be particularly important, especially given the critical role that each variable plays in the final decision-making process of the models. In the following section, a detailed explanation of all variables considered for each developed model is presented, based on the segmentation outlined.

3 Data Preprocessing

As stated in Section 1, all data has been anonymized. To preserve confidentiality, the specific variables used in constructing the datasets have been partially withheld for the application in the selected models.

Following the development of the client segmentation, it is now possible to define a specific dataset for each segment, based on their characteristics.

All data used in this study corresponds to transactions processed by the current Pay No Pay model over a one month period, which provided a substantial volume of information, considered enough for the development and training of the proposed models.

Accordingly, the selected variables will be introduced, many of which are common across all clients, while others are unique to specific segments.

One of the most important variables used across all datasets (proving to be particularly influential in the modeling process) is the unique **Credit Risk** assigned to each client. This variable represents a weighted calculation that is used to assess the individual risk profile of each client.

3.1 Dataset Creation

Table 3.1 provides a comprehensive overview, that summarizes the variables used across the different datasets created for each client segmentation. These variables are grouped into several categories based on their characteristics. For each group, the table presents the data format (qualitative or quantitative), the number of variables included and specifies whether they are present in the dataset for Individuals, Entrepreneurs or Companies.

Starting with the group labeled as **Flag**, which contains qualitative, flag-type variables that indicate whether a client is subject to account restrictions or, for example, is currently under monitoring, among other attributes.

Then, the variables of **Sociodem** provide an overview of the sociodemographic characteristics of each individual client, which are obviously not applicable in the Companies segment.

Next, additional relevant information is presented at **Inf**, such as the length of the client's and account's antiquity with the financial institution, the presence of any blocked credit or debit amounts and, so on.

Variable	Format	Number of variables	Individuals	Entrepreneurs	Companies
Flag	Qualitative	10	✓	✓	✓
Sociodem	Qualitative	4	✓	✓	✗
Inf	Quantitative	7	✓	✓	✓
Finan	Quantitative	5	✓	✓	✓
Activ	Quantitative	14	✓	✓	✓
Pay	Quantitative	13	✓	✓	✓
Over	Quantitative	3	✓	✓	✓
Busin	Quantitative	15	✗	✓	✓
Extra	Qualitative	3	✗	✓	✓

Table 3.1: Dataset across all Client Segmentation.

Proceeding to group **Finan**, which presents a detailed analysis of specific account inflows, with particular emphasis on salary deposits from the previous month, as well as the median values observed during the preceding three and six months. These inflows are generally regular and tend to occur on consistent dates.

Now, in the **Activ** group, one can find the main account movements, including the number and amounts of both debit and credit transactions, as well as the dates of the most recent transactions.

In the variables of **Pay**, there is information regarding the transaction currently undergoing the Pay No Pay process, including its amount and transaction code. Additionally, it provides historical data related to both the client and the account within this context, with particular emphasis on the number of previous transactions that have entered this process. Finally, the **Over** variables are associated with the client's overdraft history.

Since the primary objective of this internship was to develop a model for individual clients, this group of variables already reported were the first to be created and served as the foundation for the development of the subsequent two datasets (Entrepreneurs and Companies), which include additional variables specific to business clients, the **Busin** and **Extra** variables.

The elements presented in **Busin** offer valuable accounting insights that can enhance model decision-making. However, their influence is significantly more pronounced in the Companies segment.

As illustrated in Figure 3.1, in the Entrepreneur segment only **1.3%** of the data contain this type of information, which poses a considerable challenge in build accurate models for this client segment.

Compared to the Entrepreneur segment, the **Busin** variables for Companies are significantly more complete. As shown in Figure 3.1, **88.7%** of the values are non-null, an encouraging outcome, given that such information is of extreme importance for the robust evaluation and modeling of this client category.

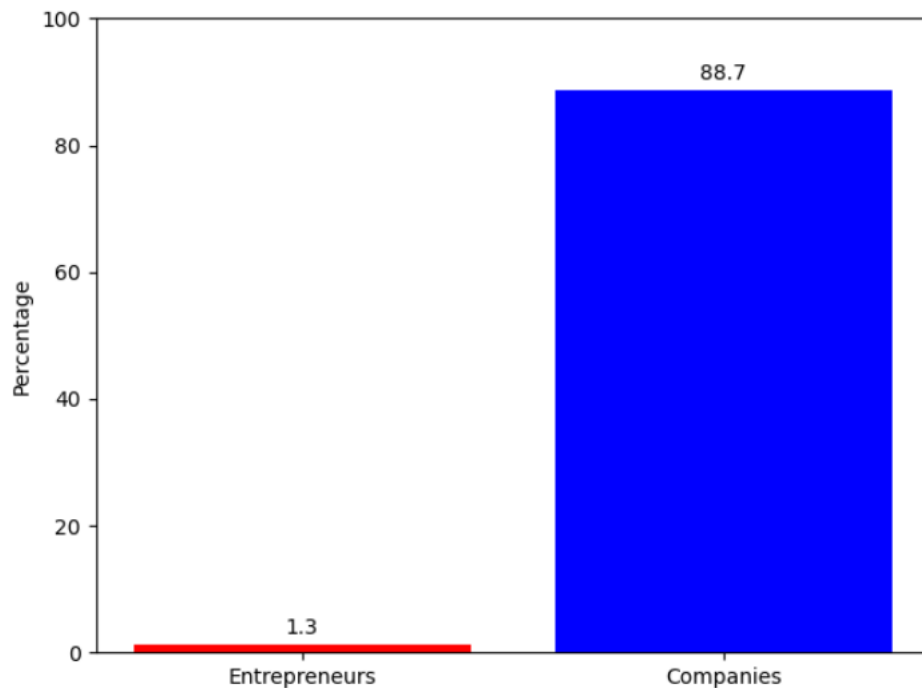


Figure 3.1: Non-Null Values in the **Busin** Variables.

The **Extra** variables provide additional information, including the loan status and its amount, as well as the sanctioned checking account limit.

With the analysis of all datasets now complete, this study can proceed to the definition of the target value (label to approve or reject), an aspect that naturally requires careful consideration.

3.2 Target Value

To define the new target value, the current decision process of the **Pay No Pay** model was reviewed, in particular, the final decisions made each day were collected and evaluated to determine their precision and suitability. This evaluation was based on a 30-day period after the transaction.

For approved transactions, the decision was considered correct if, at any point during the 30-day period, the client's balance became positive, indicating that the client's account was

overdrawn but subsequently covered.

For rejected transactions, the decision was considered incorrect if, at any point during the 30-day period, the client's balance exceeded the value of the denied transaction, indicating that the client had sufficient financial capacity for the transaction to be approved.

Once the decision for each transaction has been redefined, it is essential to consider specific conditions under which approval is categorically prohibited. According to predefined criteria established by this financial institution, any transaction meeting these conditions must be immediately rejected.

After verifying the eligibility for a new approval, the available balance following the acceptance of the transaction is also assessed.

Specifically, the account overdraft must not exceed a threshold α , which varies depending on the client's segment.

- α_1 for Individuals;
- α_2 for Entrepreneurs and Companies.

If all of these conditions are satisfied, the transaction is classified as approved for the purposes of the new model; otherwise, it is labeled as rejected, as illustrated in Figure 3.2.

This method, which involves evaluating past decisions against subsequent customer behavior, is recognized in the field for improving model accuracy and relevance.

Such analyses enables institutions to refine strategies based on historical credit information, ensuring that models remain aligned with actual customer outcomes. Additionally, the paper *Determining the Target Variable in Credit Scoring Models* emphasizes that the definition of the target variable is crucial in the development of credit scoring models [21]. The study explores various definitions and their impact on model performance, underscoring the necessity of a well-considered target variable that reflects true customer behavior.

With the implementation of this new decision logic, many transaction outcomes throughout the dataset will be changed, as illustrated in Figure 3.3.

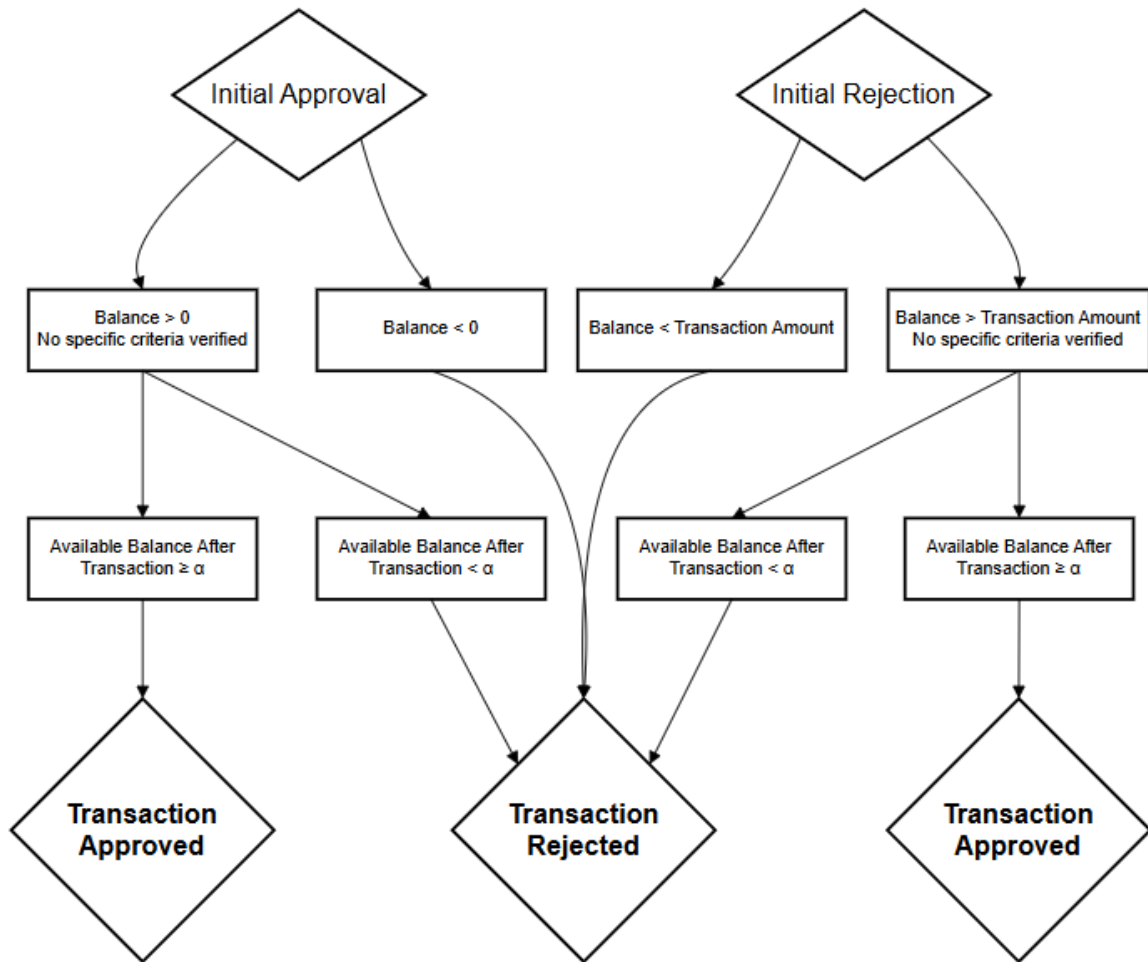


Figure 3.2: New Target Value Revision.

This graph (in Figure 3.3) illustrates the changes observed in both approval and rejection decisions. It presents the percentage of approvals that remained consistent with the updated target revision (**True Approval Rate**), as well as the percentage of rejections that were also consistent (**True Rejected Rate**). The complementary percentage in each case represents the decisions that were now altered by the method described.

As observed, each client type displays distinct patterns in the stability of decisions. For both Individuals and Entrepreneurs, the True Approval Rate and True Rejected Rate show similar behavior, in which there is a high proportion of correctly maintained approvals and a lower proportion of correctly maintained rejections.

In contrast, the pattern for Companies is more balanced, with approval and rejection consistency rates being closer in value. This suggests that, for this client group, the revised model resulted in a comparable number of changes to both approval and rejection decisions.

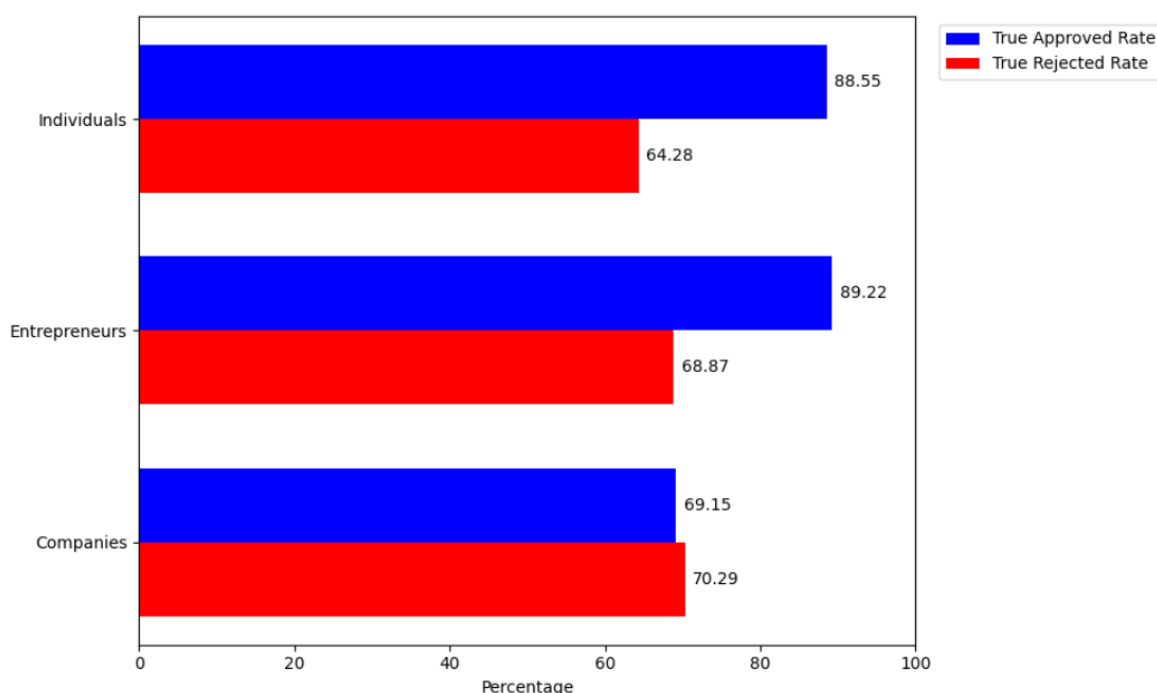


Figure 3.3: New Target Value Distribution.

Logically, the number of approved transactions would change if this new metric was applied. Under this approach, the approval volume would shift from:

- **Individuals:** 25.29% $\Rightarrow$ 49.05%;
- **Entrepreneurs:** 29.03% $\Rightarrow$ 47.95%;
- **Companies:** 44.67% $\Rightarrow$ 46.59%.

Naturally, this would lead to a higher approval rate in the Pay No Pay process moving forward.

However, it is essential to assess the extent to which these new decisions impact the overall process. In other words: **Are the clients being approved under this new approach truly the most suitable ones?**

To support this analysis, an additional variable is introduced, the **Score**, which serves as a proxy for client quality, where higher values indicate lower associated client risk. This variable can be closely linked to the concept of Credit Risk introduced at the beginning of this section. However, the Score will be used solely as a supporting measure to better understand the behavior of this new target value definition.

In order to accomplish this study, the new obtained decisions (for each client segment) are reported in relation to this variable, as presented in Table 3.2.

Value	Score for Individuals	Score for Entrepreneurs	Score for Companies
Old label 1	707.95 $\pm$ 30.04	705.97 $\pm$ 30.29	698.25 $\pm$ 37.75
New label 1	695.97 $\pm$ 32.42	695.36 $\pm$ 32.00	686.65 $\pm$ 35.50
Old label 0	681.32 $\pm$ 37.45	676.79 $\pm$ 37.00	668.93 $\pm$ 38.28
New label 0	680.03 $\pm$ 40.87	675.52 $\pm$ 40.47	679.21 $\pm$ 45.56

Table 3.2: Score distribution.

First, in the Companies segment, the average score for approved transactions (label 1) decreased under the new label definition, from 698.25 to 686.65. This behavior noticed in all segments is expected, as expanding the set of approved transactions typically includes approving clients with lower scores, thus reducing the overall mean. This suggests that the model, when redefined, is more inclusive in its approval decisions, potentially allowing a broader range of clients to be accepted.

Interestingly, the average score for rejected clients (new label 0) increased, more precisely from 668.93 to 679.21, which may imply that some clients who were previously approved (and had higher scores) are now being rejected. Although this may initially appear implausible, it could indicate a shift in the decision boundary that places greater emphasis on specific features not captured by the score alone.

For the Individuals segment, as anticipated with the increase in approvals, the average score for the new (approval) label 1 is lower, reflecting the inclusion of a larger number of clients. Regarding rejections (new label 0), the average score is also slightly lower under the new rejection approach (different from the Companies segment). This outcome suggests that the revised definition of the target variable for this client segment is quite satisfactory, as it continues to distinguish effectively between *good and bad clients*. It is observed a similar pattern in the Entrepreneurs segment, in which the increase in approvals follows a comparable trend, as expected given the notable similarities between these two client profiles.

In summary, the new definition appears to redistribute approval decisions, possibly aiming for a more balanced risk assessment. With the dataset and the redefined target variable now established, the following section will present the theoretical foundations of the ML methods selected for this classification task.

4 Machine Learning Methods

ML is one of the fastest growing fields in computer science, with applications across various domains. It refers to the automated detection of meaningful patterns in data, enabling systems to learn from experience and improve performance without being explicitly programmed.

Among the different paradigms within ML, supervised learning has been selected for this project, which involves training algorithms on labeled data, meaning that the input features are paired with known output labels. This approach facilitates the selection of meaningful features and generally results in higher predictive accuracy when compared to unsupervised methods [22].

Given the nature of the project, analysts must be able to understand and justify the reasoning behind each classification outcome. For this reason, interpretable models like decision tree are favored in these contexts [23].

The models selected for this study were chosen based on their proven performance and relevance in the literature, particularly in classification problems involving tabular financial data. The decision tree algorithm was selected as the foundational model due to its interpretability, low computational cost and widespread use [24]. Building upon this foundation, more advanced models like ensemble methods such as Random Forest (an extension of decision tree using bagging), as well as *state of the art* gradient boosting frameworks like XGBoost, LightGBM and CatBoost are explored.

All models were implemented using the Python programming language, which is widely adopted in the scientific community. According to a comparative study of statistical software tools, Python's `scikit-learn` library produced the most accurate results and was recommended for these types of projects [25].

4.1 Decision Tree

A decision tree was developed by Morgan and Sonquist in 1963, and it is a *tree-like* model created by recursively breaking down data step by step asking the best possible questions at each point and stopping once it reaches a clear decision [26]. It's an easy and understandable way to sort data into categories and making predictions.

As presented in Figure 4.1, a decision tree works like a flow chart where:

- **Root Node** is the beginning of this model, with no incoming branches;
- Then, the tree branches out into **Internal Nodes**, or decision nodes, each representing

a point where the dataset is split based on feature evaluations. These nodes iteratively partition the data into increasingly homogeneous groups;

- The process continues until it reaches the **Leaf Nodes**, also known as terminal nodes, which represent the outcome/classification.

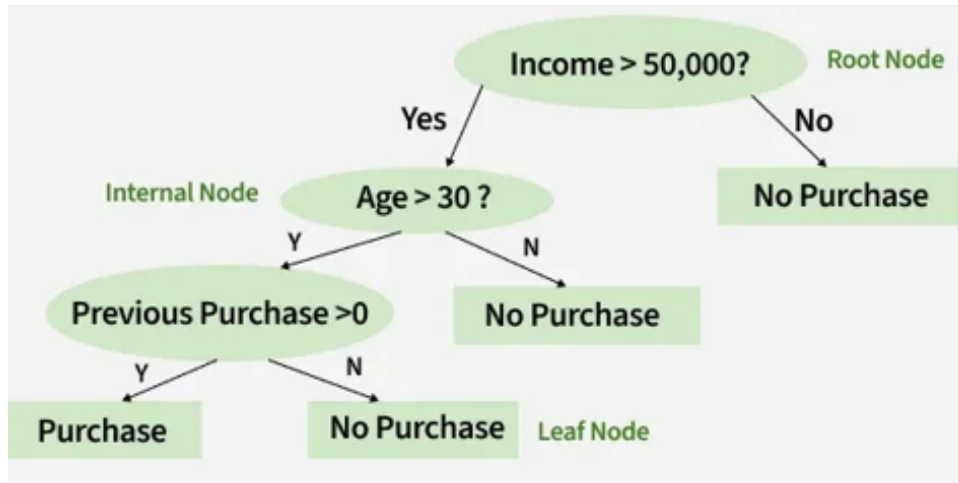


Figure 4.1: Example of Decision Tree model [27].

Let's consider the decision tree illustrated in Figure 4.1 for predicting whether a customer will buy a product based on age, income and previous purchases. Here's how the decision tree really works:

1. Root Node (Income): First Question: "Is the person's income greater than 50,000?"
If Yes → next question;
If No → predict "No Purchase" (leaf node).
2. Internal Node (Age): If the person's income is greater than 50,000, ask: "Is the person's age above 30?"
If Yes → next question;
If No → predict "No Purchase" (leaf node).
3. Internal Node (Previous Purchases):
If the person is above 30 and has made previous purchases → predict "Purchase";
If the person is above 30 and has not made previous purchases → predict "No Purchase".

At the core of decision tree induction lies **Hunt's algorithm**, a recursive partitioning method proposed in the 1960s, which serves as the foundation of decision tree classifiers [28]. The algorithm proceeds as follows:

- At a given node t , if all the records belong to the same class, the node becomes a leaf node labeled with that class;
- Otherwise, the dataset is split into smaller subsets based on an attribute test condition, and the algorithm is applied recursively to each subset.

Decision tree learning employs a **Divide and Conquer** strategy by conducting a greedy search to determine the optimal features and corresponding thresholds that result in the best possible split, which is the split that most effectively separates the data based on class labels. This process of splitting is then repeated in a top-down, recursive manner until all records have been classified under a specific class label.

Therefore, a crucial step in constructing this model is determining the most effective split at each node. This decision is guided by **impurity measures**, which evaluate how well a split separates the data, ideally dividing the data so that each resulting subset contains instances predominantly from a single class. In other words, one subset should mostly contain records from one class, while the other contains records from the other class. The goal is to achieve subsets that are as **pure** as possible, meaning that most (or all) of their elements belong to the same class. Several metrics are used to quantify this level of purity.

- **Entropy and Information Gain:** Entropy measures the level of disorder in a dataset, and Information Gain quantifies how much that disorder is reduced after splitting the data using a specific feature;
- **Gini Index:** This is the most commonly used criterion in modern decision trees. The Gini index measures the probability of incorrectly classifying a randomly chosen element. It is defined as presented in Equation 4.1:

$$\text{GINI}(t) = 1 - \sum_{i=1}^n p_i^2 \quad (4.1)$$

This is calculated based on the proportions of instances that belong to each class within a given node (t). Specifically, for each class i , p_i is computed, which is the proportion of instances in the node that belong to that class. In other words, the number of instances of class i in node t divided by the total number of instances in that node.

Note that a Gini index of 0 represents perfect purity (all instances correspond to the same class), while a value of 0.5 indicates that the node contains an equal mix of both classes, representing maximum impurity.

For this study, the **Gini Index** is selected due to its computational efficiency and strong empirical performance, particularly when applied with the decision tree implementation used in `scikit-learn`, which is based on the CART algorithm (see more in [29]).

Despite being a widely used model with the significant advantage of being highly interpretable and easy to visualize, the decision tree model still presents several limitations, including [30]:

- Prone to overfitting, especially on noisy datasets (tree grows too deep);
- High variance: small changes in data can result in different trees;
- Unstable performance without regularization or pruning.

In order to implement this model effectively, it is necessary to define the following hyperparameters as presented in Table 4.1 (in addition to the split criterion, which in this case is the Gini Index):

Hyperparameter	Description
<code>max_depth</code>	The maximum depth of the decision tree
<code>min_samples_split</code>	The minimum number of samples required to split an internal node
<code>min_samples_leaf</code>	The minimum number of samples required to be at a leaf node
<code>ccp_alpha</code>	Controls the pruning of the decision tree (plays a critical role in mitigating overfitting)

Table 4.1: Decision Tree Hyperparameters

In this study, the decision tree classifier is used as a baseline model to evaluate the performance of more sophisticated methods. The next section introduces the ensemble methods in detail, illustrating how they build upon the decision tree framework to produce more robust models.

4.2 Ensemble Methods

Ensemble learning is a widely adopted ML paradigm that combines multiple base learners to improve predictive performance, rather than relying on a single model [31].

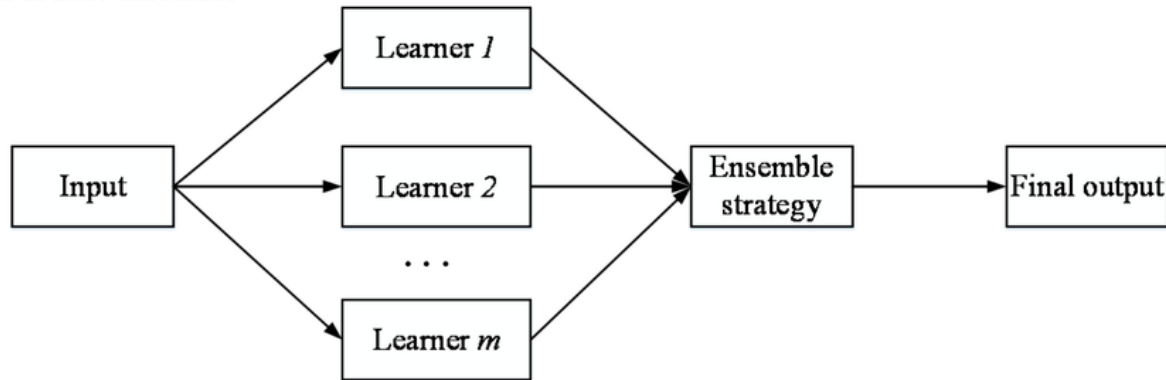
These methods are generally categorized into four main types: bagging, boosting, voting and stacking. In this project, particular emphasis was placed on bagging and boosting techniques, which are among the most commonly used ensemble approaches. Notably, boosting

(and also advanced gradient boosting) received special attention, as it embraces a broader range of models and is expected to yield superior performance [32].

Finally, ensemble methods can be categorized based on the strategy used to generate base learners, as illustrated in Figure 4.2 and described below:

- **Parallel Ensemble Methods — Bagging:** In this approach, all base learners are generated independently and in parallel. The primary motivation is to leverage the independence among learners;
- **Sequential Ensemble Methods — Boosting:** This strategy involves the sequential generation of base learners, where each new model is trained to correct the errors of its predecessors.

Parallel ensemble



Sequential ensemble

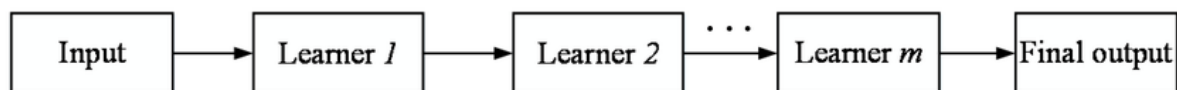


Figure 4.2: Flowchart of parallel and sequential ensemble [33].

In this project, the ensemble methods rely on **Decision Tree as Base Learners**, where their simplicity and interpretability make them a solid foundation for building more complex models.

4.2.1 Bagging

Bagging, short for Bootstrap Aggregating, is a widely used ensemble learning technique designed to improve the stability and accuracy of ML algorithms [34]. It works in the following steps:

1. Generates multiple training datasets using a **Bootstrap Sampling**;

2. Each bootstrapped dataset is used to train an individual base learner (decision tree);
3. All predictions are aggregated (**Model Aggregation**), typically by majority voting for classification tasks.

The process of **Bootstrap Sampling** involves generating multiple datasets by sampling with replacement from an original training set of m instances. Each bootstrap sample also contains m instances, but due to replacement, some examples may be repeated while others may be excluded. As illustrated in Figure 4.3, the original dataset consists of 5 instances, and three bootstrap samples are created, each with 5 instances also. For example, in the first sample, we see two blue animals, one green, and two yellow ones, demonstrating how the total number of instances is preserved, but some are repeated while others are omitted, effectively creating diverse training subsets from a single dataset.

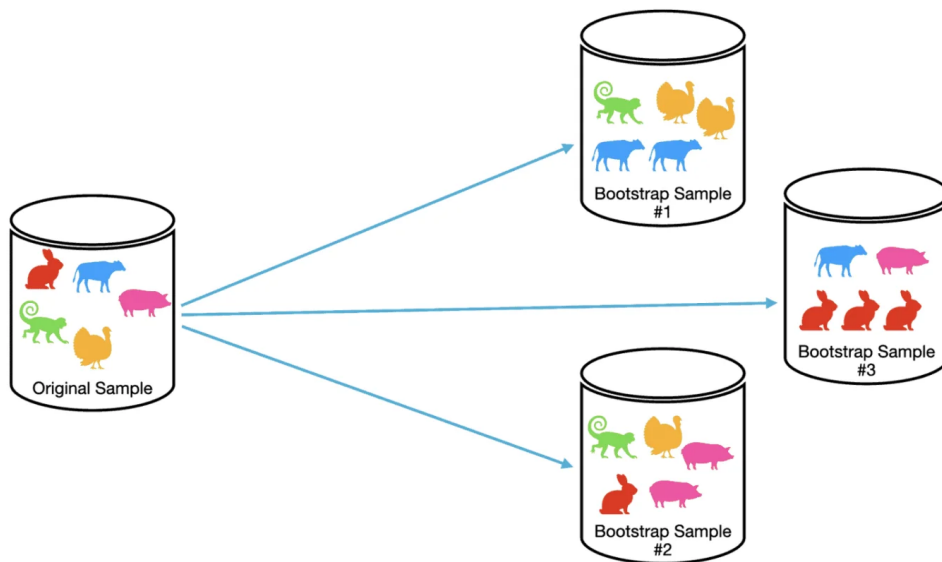


Figure 4.3: Flowchart of Bootstrap Sampling [35].

As previously mentioned, bagging is an example of a parallel ensemble method. The key concept is that base learners (decision tree) are trained independently on different subsets of the dataset, and their outputs are later combined. This parallel approach allows the reduction of variance (how much the model's predictions would change when trained on different data samples) without significantly increasing bias (the error introduced when the model is too simple to capture the underlying patterns).

In the context of this project, **Random Forest** was chosen as the primary model to represent the bagging approach.

4.2.2 Boosting

Boosting is an ensemble technique that aims to transform a set of weak learners, models that perform only slightly better than random guessing, into a single strong learner with significantly enhanced predictive capabilities.

Like bagging, boosting combines the outputs of multiple base learners. However, it differs in two fundamental ways:

- Instead of training models in parallel, boosting **trains them sequentially**;
- Rather than treating each base learner equally, boosting assigns **weights to each learner** based on its performance, giving more influence to those that perform better [36].

Furthermore, this method can be described by the following steps (**where each base learner is a decision tree**):

1. The training process is iterative, where initially, all instances are assigned equal weights;
2. A base learner is then trained on this weighted dataset;
3. After evaluating its performance, the weights are updated: misclassified instances are given higher weights, while correctly classified ones receive lower weights;

This process ensures that each new base learner focuses more on the instances misclassified by the previous ones, effectively learning from their errors. Boosting builds the final predictive model by combining all base learners (decision tree) through a weighted process, resulting in a robust predictive model.

4.2.3 Gradient Boosting

Gradient Boosting is an advanced ensemble technique based on the principles of boosting, but unlike traditional boosting methods that primarily adjust sample weights, Gradient Boosting adopts a **Gradient Descent** optimization strategy to minimize the loss function of the ensemble.

In this approach, each new model is trained to predict the residual errors of the previous ensemble with respect to a given loss function. These residuals indicate how much the current predictions deviate from the actual values, guiding the learning process in the direction of maximum error reduction [10].

In the context of binary classification, the most common loss function used is the **Logistic Loss** (also known as log-loss or deviance loss), which is presented in equation 4.2.

$$\mathcal{L}(y, \hat{p}) = -[y \cdot \log(\hat{p}) + (1 - y) \cdot \log(1 - \hat{p})] \quad (4.2)$$

Where:

- $y \in \{0, 1\}$ is the true class label;
- $\hat{p} = \sigma(F(x)) = \frac{1}{1+e^{-F(x)}}$ is the predicted probability;
- $F(x)$ is the raw score or logit output from the model.

To apply gradient boost, it is necessary to compute the first and second derivatives of the loss with respect to the score $F(x)$, as demonstrated in Equations 4.3 and 4.4. These are achieved by a second-order Taylor approximation.

$$\frac{\partial \mathcal{L}(y_i, F(x_i))}{\partial F(x_i)} = \hat{p}_i - y_i \quad (4.3)$$

$$\frac{\partial^2 \mathcal{L}(y_i, F(x_i))}{\partial F(x_i)^2} = \hat{p}_i(1 - \hat{p}_i) \quad (4.4)$$

At each iteration t , a new model $F_t(x)$ is trained to fit the negative gradient of the loss. The total loss is then approximated using the Taylor expansion, as presented in Equation 4.5 (see more in [37]).

$$\mathcal{L} \approx \sum_{i=1}^n \left[g_i F_t(x_i) + \frac{1}{2} h_i F_t(x_i)^2 \right] \quad (4.5)$$

Where:

- $g_i = \hat{p}_i - y_i$ is the Gradient function;
- $h_i = \hat{p}_i(1 - \hat{p}_i)$ is the Hessian function;
- $F_t(x_i)$ is the prediction from the new base learner.

Taking into account, that $f_t(x)$ is the base learner trained to approximate the negative gradient of the loss function, so that the new base learner is able to correct the errors of the ensemble built.

The core update formula is given by Equation 4.6:

$$F_t(x) = F_{t-1}(x) + \nu \cdot f_t(x) \quad (4.6)$$

Where:

- $F_t(x)$ is the updated prediction function at iteration t ;
- $F_{t-1}(x)$ is the prediction from the previous iteration;
- $\nu \in]0, 1]$ is the learning rate or shrinkage factor, which scales the contribution of each base learner, $f_t(x)$.

The **learning rate** plays a crucial role in controlling the influence of each new added model, where, for example, a smaller learning rate reduces the impact of each individual base learner, requiring more models to converge, but potentially improving generalization and preventing overfitting.

The final predicted class label $\hat{y}$ is then determined by applying a threshold (α) to the predicted probability $\hat{p}$, as presented in Equation 4.7:

$$\hat{y} = \begin{cases} 1 & \text{if } \hat{p}(x) > \alpha \\ 0 & \text{otherwise} \end{cases} \quad (4.7)$$

The core idea of Gradient Boosting is to correct the mistakes of previous models by adding new base learners that predict the errors of the ensemble so far. In each iteration, the algorithm performs a gradient descent step on the loss function in the function space. The output function $F(x)$ is iteratively refined to better fit the data while reducing classification error [38].

The use of the log-loss function ensures that poorly classified examples receive larger updates, allowing the model to focus on the more difficult cases during training. The shrinkage parameter ν helps control overfitting by slowing down the learning process.

In summary, Gradient Boosting represents a powerful enhancement over standard boosting techniques (with the same foundation) by treating the learning task as an optimization problem solved through functional gradient descent. Its success has led to widespread adoption in both academic research and industrial applications, particularly for structured/tabular data [37].

This analysis provides the foundation for understanding how gradient boosting methods (with Decision Tree as base learners) like XGBoost (eXtreme Gradient Boosting), LightGBM (Light Gradient Boosting Machine) and CatBoost (Categorical Boosting) work internally.

4.3 Random Forest

Starting with the chosen bagging model, Random Forest, which is widely recognized as one of the most popular and powerful ensemble learning methods. It was introduced by Breiman in 2001 [9], and it improves the performance of decision trees by training them separately and combining their outputs through majority voting, as illustrated in Figure 4.4. This approach

helps reduce overfitting and improves generalization, while preserving the simplicity and interpretability of individual decision trees.

The Random Forest algorithm extends the basic principles of bagging with a key innovation (Feature Randomness), which consists of:

- **Feature Randomness:** During the construction of each decision tree, a **random subset of features is selected at each split rather than considering all features**. This introduces an additional layer of diversity among the trees in the ensemble, reducing correlation between them, and lowering the overall variance of the model. By forcing individual trees to consider different subsets of features, the model is encouraged to capture a wider range of patterns in the data [39].

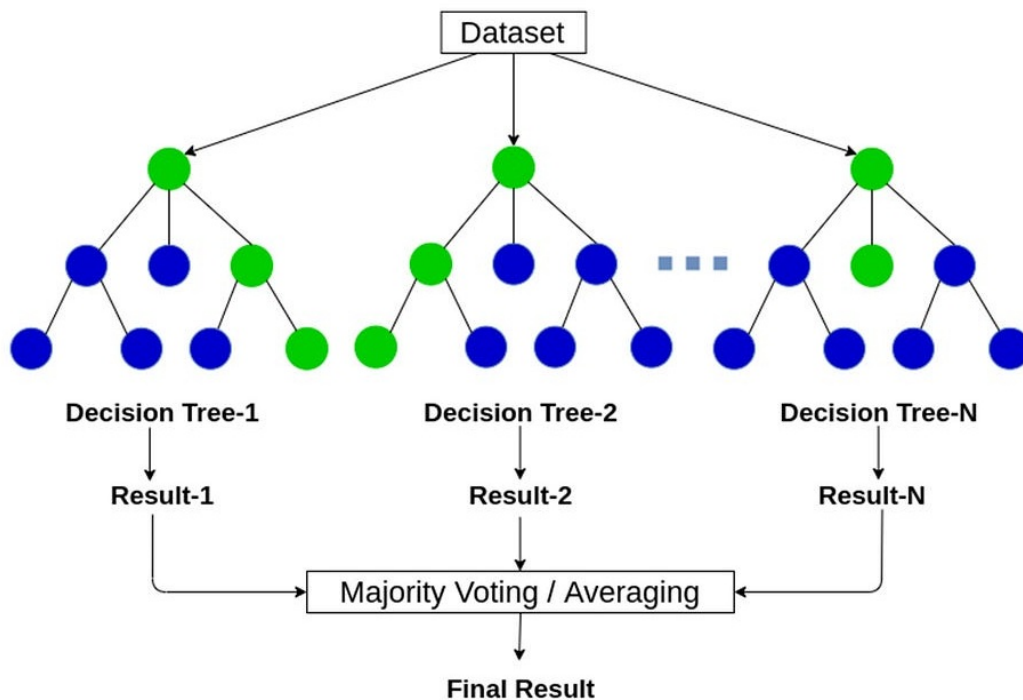


Figure 4.4: Random Forest Scheme [40].

Random Forest benefits from both the low bias of the decision tree and the variance reduction achieved through the bagging process (and the principle of feature randomness). Each tree is trained on a different bootstrap sample of the data, and due to the random feature selection at each split, the model captures a more diverse set of patterns. As a result, even though each individual tree might be a weak learner, their aggregated output through majority voting produces a strong, stable and robust classifier, making Random Forest especially effective for high-dimensional datasets or situations where noise and outliers might reduce the performance of simpler models.

To effectively implement this model, it is necessary to configure the same hyperparameters listed in Table 4.1 for the decision tree model, along with the `n_estimators` hyperparameter, which defines the number of decision tree models included in the ensemble.

In summary, Random Forest offers a solid baseline through its ensemble of randomized decision trees, balancing accuracy, robustness and interpretability. Although it primarily reduces variance, it does not inherently focus on reducing bias. This limitation paves the way for the exploration of boosting techniques, which address this aspect by sequentially building models that correct the mistakes of their predecessors. The next sections present the chosen Gradient Boosting models, which further enhance predictive performance by optimizing the learning process in a more targeted and iterative way.

4.4 XGBoost

Proceeding to the **Gradient Boosting** models, this section begins with an overview of XGBoost, which was developed by Chen and Guestrin in 2016 [37]. It introduces several improvements, such as parallelized tree construction and efficient handling of sparse data, along with advanced regularization techniques. These enhancements can be depicted as follows:

- **Regularization:** Both L1 (Lasso) and L2 (Ridge) regularization are supported, which helps control model complexity and prevent overfitting;
- **Handling Missing Values:** XGBoost is capable of handling missing data internally without the need for imputation;
- **Parallelization:** Tree construction is parallelized, significantly accelerating the training process;
- **Tree Pruning:** Implements a depth-first approach for tree pruning, optimizing the computational resources.

In essence, XGBoost builds upon the standard gradient boosting framework with a strong emphasis on speed and performance. Its ability to handle high-dimensional datasets, robustness to missing data and the use of regularization make this model one of the most widely used and powerful gradient boosting algorithms in practice. In addition, the algorithm supports tree-based models, offering flexibility in terms of model structure depending on the nature of the task. These features, combined with its scalability and efficient memory usage, make XGBoost particularly well suited for large-scale ML problems.

The hyperparameters required for the implementation of this model are described in Table 4.2.

Hyperparameter	Description
<code>n_estimators</code>	Number of boosting rounds (number of trees to be built)
<code>learning_rate</code>	Shrinks the contribution of each tree
<code>max_depth</code>	Maximum depth of each individual tree
<code>min_child_weight</code>	Minimum sum of instance weight (hessian) needed
<code>gamma</code>	Minimum loss reduction required to make a further partition on a leaf node
<code>subsample</code>	Fraction of the training instances to be randomly sampled for growing each tree
<code>colsample_bytree</code>	Fraction of features to be randomly sampled for each tree
<code>reg_alpha</code>	L1 regularization term on weights (Lasso)
<code>reg_lambda</code>	L2 regularization term on weights (Ridge)

Table 4.2: XGBoost Hyperparameters

Overall, the design choices implemented in XGBoost contribute to its excellent performance and widespread adoption in data science competitions and industry applications. However, despite its advantages, it is still fundamentally a gradient boosting algorithm, and as such, shares many similarities with other models in this family. The following sections explore alternative gradient boosting frameworks, such as LightGBM and CatBoost, which introduce different strategies to further improve training speed, memory efficiency and handling of categorical features.

4.5 LightGBM

Moving to the LightGBM model, developed by Microsoft in 2016. It is a high-performance gradient boosting framework designed for speed and efficiency, particularly on large datasets [41]. LightGBM stands out from other gradient boosting algorithms due to the following innovations:

- **Histogram-based Learning:** Continuous features are bucketed into discrete bins, which speeds up training and reduces memory usage;

- **Leaf-wise Tree Growth:** Instead of growing trees level-wise, LightGBM grows trees leaf-wise, choosing the leaf with the maximum loss reduction to split. This often leads to deeper trees and higher accuracy;
- **Gradient-based One-Side Sampling (GOSS):** Instead of using the entire dataset, LightGBM keeps all instances with large gradients (those the model is struggling with) and randomly samples from those with smaller gradients. This allows the model to focus on the most informative examples, maintaining accuracy while significantly speeding up training;
- **Exclusive Feature Bundling (EFB):** LightGBM combines mutually exclusive features (those that rarely take non-zero values simultaneously) into a single feature, allowing lower computational complexity and memory usage while maintaining model accuracy.

LightGBM was specifically designed to address the limitations of traditional gradient boosting algorithms in terms of training speed and memory consumption. Its architecture allows for faster training without sacrificing predictive performance, unlike traditional boosting approaches that rely on level-wise tree growth and exhaustive data usage, LightGBM employs more intelligent data sampling and structure learning strategies. These advantages have led to its popularity in ML competitions and production systems where performance and scalability are critical.

The hyperparameters listed in Table 4.2 (originally used for the XGBoost model) are also applied to this new framework, with the exception of `min_child_weight` and `gamma`, which are replaced in the LightGBM model by `min_child_samples` and `num_leaves`.

The `min_child_samples` hyperparameter sets the minimum number of samples required to form a leaf node, where higher values typically result in more conservative models. Meanwhile, `num_leaves` determines the maximum number of leaves per tree, directly influencing model complexity, where higher values allow the model to learn more intricate patterns, while lower values help prevent overfitting [42].

In summary, LightGBM delivers strong performance through its combination of advanced sampling techniques and optimized tree growth strategies. Its ability to handle large datasets with high speed and low memory usage makes it a preferred choice in many real-world applications. Nonetheless, one of its known limitations is the suboptimal handling of categorical features, often requiring preprocessing steps. To address this challenge, the next model, CatBoost, was specifically designed to incorporate native support for categorical variables.

4.6 CatBoost

CatBoost, developed by Yandex in 2017, is a gradient boosting framework specifically designed to handle categorical features more efficiently [43]. The key characteristics of CatBoost include [44]:

- **Efficient Categorical Feature Handling:** CatBoost applies a unique method called Ordered Target Statistics to convert categorical variables into numerical representations without target leakage;
- **Ordered Boosting:** Unlike traditional boosting that may cause prediction shift, CatBoost uses ordered boosting that builds each tree using a permutation of the dataset, reducing overfitting;
- **Robustness to Overfitting:** The algorithm is regularized and stabilized by design, which improves generalization on unseen data;
- **Multi-class Classification:** CatBoost natively supports multiclass classification and handles imbalanced datasets effectively.

CatBoost builds upon the foundations of gradient boosting by addressing some of its limitations, especially in datasets with a high number of categorical variables. One of its main advantages is that it eliminates the need for extensive preprocessing (or encoding) of categorical data, which can introduce noise or bias if not handled properly. Its implementation of Ordered Target Statistics and Ordered Boosting ensures that information leakage is avoided during training, which significantly improves model robustness and reliability. Moreover, CatBoost is user-friendly, requiring minimal hyperparameter tuning to achieve strong results, making it suitable for both practitioners and researchers.

In Table 4.3, the hyperparameters needed for the implementation of the CatBoost model are presented. In addition to all the mentioned hyperparameters, the `early_stopping` hyperparameter was also employed with a value of 50 across **all gradient boosting models** described so far. This hyperparameter serves to prevent overfitting by automatically identifying the point at which the model's performance on the validation dataset begins to deteriorate, despite continued improvement on the training data, indicating the onset of overfitting.

Hyperparameter	Description
iterations	The maximum number of trees (boosting iterations) to be built
depth	The maximum depth of the trees
learning_rate	The step size shrinkage applied to each tree's contribution
l2_leaf_reg	Coefficient for L2 regularization on leaf weights
bagging_temperature	Controls the intensity of Bayesian bootstrapping, where higher values increase the randomness in data sampling and improve generalization

Table 4.3: CatBoost Hyperparameters

Overall, CatBoost distinguishes itself by combining high performance with simplicity, particularly in datasets rich in categorical features. Its ability to deliver competitive results with minimal preprocessing and its mechanism to prevent overfitting makes it a strong candidate in many real-world applications.

With all the chosen models now introduced and described, the discussion can move on to the workflow required to implement and evaluate them. Only after understanding this process will it be possible to fully comprehend the results obtained when optimizing the Pay No Pay process.

5 Model Workflow

This chapter presents the methodology used for model preparation and validation, followed by the evaluation metrics selected to assess the model's performance.

5.1 Model preparation

To ensure reliable ML models, a structured validation approach was employed. This approach consists of three key steps: initial data partitioning, hyperparameter tuning using cross-validation and final model assessment on a holdout test set.

Initially, the available dataset was split into two distinct subsets:

- **Training Set (75%)** → Used for both model training and hyperparameter tuning;
- **Test Set (25%)** → Reserved exclusively for final model evaluation and never used during training or tuning phases.

To determine the optimal hyperparameters for each model, a K-Fold (K=5) Cross-Validation procedure was employed on the training set within the optimization framework known as **Optuna**. This approach aimed to identify the combination of hyperparameters that maximized a selected evaluation metric.

By leveraging cross-validation rather than a single train-test split, it was possible to achieve a more reliable estimate of model performance.

Once the optimal hyperparameters were identified, the models (with these specific values) were trained using the entire training set. Each model was then evaluated on a separate test set (with 25% of the data), which was kept held out only for this task.

5.1.1 Hyperparameter Tuning

The architecture of a ML model is determined by a set of parameters known as **Hyperparameters**. Consequently, the process of identifying the most effective configuration for a given model is referred to as Hyperparameter Tuning. Naturally, the larger the number of hyperparameters and the wider the search space, the more computationally expensive this process becomes.

There are several strategies available for performing hyperparameter tuning, as illustrated in Figure 5.1, which compares the search processes of Grid Search, Random Search and

Bayesian Optimization. The figure highlights key elements such as the evaluation points explored by each method, the global optimal hyperparameters and the local optimal points in which the algorithm incorrectly assumes it has found the best configuration.

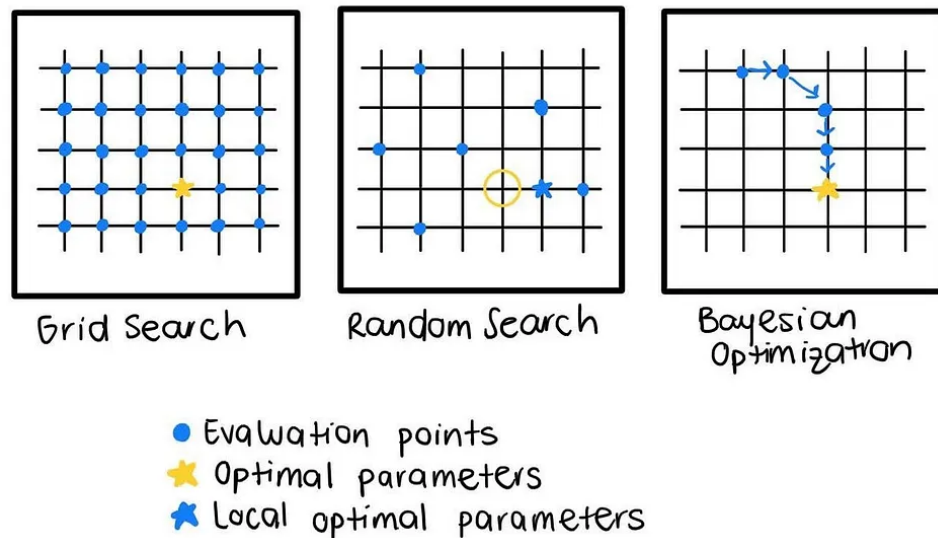


Figure 5.1: Visual Difference between Hyperparameter Tuning processes [45].

Several techniques have been developed to efficiently explore the hyperparameter search space. Below are some of the most commonly used hyperparameter optimization methods:

- **Manual Tuning:** Consists of manually selecting and adjusting hyperparameters based on prior knowledge, domain expertise or intuition. While it can be effective in simple scenarios, it is a long process, that lacks scalability, and is not guaranteed to find optimal configurations;
- **Grid Search:** Exhaustively explores the hyperparameter space by evaluating all possible combinations from a predefined set of values (as one can see with the grid presented in Figure 5.1). It is systematic and easy to implement, but becomes computationally expensive and inefficient, especially when the number of hyperparameters or their range increases;
- **Random Search:** Selects hyperparameter combinations at random from specified distributions. This method often outperforms grid search in high-dimensional spaces, as it is less likely to waste time evaluating less informative regions of the search space (as one can see by the difference between the number of evaluation points from Grid Search and Random Search in Figure 5.1). However, since the selection is purely random, it may converge to local optimal points rather than the true global optimum, a limitation that is

also visually presented in Figure 5.1;

- **Bayesian Optimization:** Builds a probabilistic model of the objective function, typically Gaussian Processes or Tree-structured Parzen Estimators (TPE) to evaluate the performance of hyperparameter configurations. Instead of blindly trying many combinations (see Figure 5.1 for more visual context), it strategically chooses which hyperparameter to evaluate next, by using an acquisition function, which balances exploration (trying out new regions of the search space) and exploitation (focusing on areas that are already known to perform well). After each evaluation, the model updates its understanding of the performance landscape, making future choices smarter;
- **Optuna:** An advanced hyperparameter optimization framework that builds on the Bayesian Optimization principles. It dynamically explores the search space, stops poorly performing trials early to save time (pruning), and applies efficient sampling strategies like TPE to quickly identify promising configurations, significantly reducing computational overhead while improving tuning effectiveness.

Bayesian Optimization is a sequential design strategy for global optimization of black-box models, where instead of optimizing the unknown objective function $f(x)$ directly, it builds a surrogate probabilistic model that approximates $f(x)$, and uses this model to select the next promising point to evaluate. This selection is driven by an acquisition function, with one of the most widely used being the **Expected Improvement (EI)**, which quantifies the expected amount by which a new candidate point will improve over the current best observed value. It takes into account both the predicted performance at that point and the uncertainty of that prediction. By doing so, EI balances exploration (testing points with high uncertainty) and exploitation (focusing on points likely to improve the result), making the search for optimal points more efficient with fewer evaluations.

A widely adopted approach to Bayesian Optimization is the **TPE**, which serves as the core optimization algorithm in **Optuna**. Unlike traditional Gaussian Process methods that model the posterior distribution $p(y|x)$ (representing the probability of an outcome y given hyperparameters x), the TPE instead models the inverse relationship through density estimation: the likelihood $p(x|y)$ and the prior $p(y)$ [46].

Here, x denotes a specific set of hyperparameters, and y corresponds to the value of the objective function when the model is trained and evaluated using x . Also, $p(x|y)$ represents the probability distribution of hyperparameters x conditioned on a certain level of objective value y , and the prior $p(y)$, on the other hand, describes the overall distribution of objective values

across all tested hyperparameters.

To build the surrogate model, the observed pairs $\{(x_i, y_i)\}_{i=1}^n$ are split into two groups based on a quantile threshold y^* . This threshold is chosen such that a fixed fraction γ of the best-performing observations lies below it.

- $l(x) = p(x|y < y^*)$: the likelihood model for good hyperparameter configurations (objective values better than y^*);
- $g(x) = p(x|y \geq y^*)$: the likelihood model for the rest (objective values worse than or equal to y^*).

Both $l(x)$ and $g(x)$ are estimated using non-parametric methods such as Kernel Density Estimation, where $l(x)$ is the density formed with the observations $\{x^{(i)}\}$ such that corresponding loss was less than y^* and $g(x)$ is the density formed by using the remaining observations.

The next point to evaluate, x_{new} , is chosen by maximizing the acquisition function presented in Equation 5.1:

$$x_{\text{new}} = \arg \max_x \frac{l(x)}{g(x)} \quad (5.1)$$

This ratio represents the EI in a sampling-friendly form. More formally, the TPE algorithm optimizes this EI function, which can be described at Equation 5.2.

$$EI(x) = \int_{-\infty}^{y^*} (y^* - y) p(y|x) dy \quad (5.2)$$

This equation expresses the EI as a probabilistic estimate of how much better a new candidate point x is expected to perform compared to the best current observation y^* . It does so by integrating the difference between the best current outcome and all possible outcomes y that are better than y^* , weighted by their likelihood according to the predictive distribution of the surrogate model $p(y|x)$.

By modeling $p(x|y)$ instead of $p(y|x)$, and selecting samples that maximize the likelihood under $l(x)$ while minimizing under $g(x)$, the TPE effectively samples configurations with high EI.

Through a series of simplifications and refinements (see more in [47]), this approach leads to a more tractable and interpretable formulation of EI, as shown in Equation 5.3:

$$EI_{y^*}(x) = \frac{\gamma y^* \ell(x) - \ell(x) \int_{-\infty}^{y^*} p(y) dy}{\gamma \ell(x) + (1 - \gamma) g(x)} \propto \left(\gamma + \frac{g(x)}{\ell(x)} (1 - \gamma) \right)^{-1} \quad (5.3)$$

The advantages of TPE over the traditional Gaussian Process include:

- Better scalability to high-dimensional search spaces;
- Support for conditional search spaces (tree structured);
- Superior handling of categorical and discrete variables;
- Increased sampling efficiency due to non-parametric modeling.

This makes the TPE particularly suited for complex hyperparameter optimization tasks, and of course makes the **Optuna** framework the ideal method for hyperparameter optimization.

In the context of this project, several objective functions were tested during the hyperparameter tuning process with **Optuna**, including accuracy, precision, recall and F1-score. After empirical evaluation across multiple models and datasets, the best overall results were consistently achieved when optimizing for accuracy. Consequently, this metric was adopted as the primary optimization criterion throughout the hyperparameter tuning phase.

5.1.2 Cross Validation

Cross-validation is a widely used technique for assessing the generalization ability of ML models. In this work, the training data was further divided using cross-validation to optimize and validate the model's performance during development. This approach allowed for a robust estimation of how the model performs on unseen data within the training set. After this process, the final evaluation was carried out on a separate test set that was not used in training or validation, ensuring an unbiased assessment of the model's generalization ability.

The primary goal of cross-validation is to mitigate the risk of overfitting, a scenario in which a model performs well on the training data but fails to generalize to new, unseen data.

There are several types of cross-validation techniques that could be employed in this work, each with its own advantages and limitations [48]:

- **Holdout Validation:** The dataset is split into two subsets, where 50% is used for training and the remaining 50% for validation. Although straightforward, this method has the drawback of under utilizing the available data and may produce high variance in the results due to the randomness of the split;
- **Random Subsampling:** Similar to the holdout method, this approach repeatedly splits the data into training and validation sets using different random partitions. Although its theoretical foundation is based on the same principles as bootstrap sampling used in bagging methods, in this context it is applied as part of a cross-validation strategy rather than as a characteristic of a specific model. While random subsampling helps reduce the

variance associated with a single split, it does not ensure that all data points are used equally across training and validation sets;

- **K-Fold Cross-Validation:** The dataset is divided into k equally sized folds. The model is trained on $k - 1$ folds and validated on the remaining fold. This process is repeated k times, with each fold used exactly once as the validation set;
- **Stratified K-Fold Cross-Validation:** A variation of K-Fold that preserves the class distribution within each fold. This is particularly useful for imbalanced classification problems, as it ensures that each fold is representative of the overall dataset in terms of class proportions;
- **LOOCV (Leave-One-Out Cross-Validation):** A special case of K-Fold where k equals the number of observations. Each iteration uses a single data point for validation and the rest for training. While it maximizes the use of data for training, it is computationally expensive and may result in high variance for small datasets;

Given the trade-offs among various cross-validation techniques, **K-Fold Cross-Validation** was selected for this project. It provides a good balance between bias and variance, makes efficient use of the available data and offers stable performance estimates.

Furthermore, it is computationally less demanding than Leave-One-Out Cross-Validation (LOOCV) and more reliable than simple Holdout Validation. For these reasons, it was considered the most suitable approach for validating the models in this project.

A 5-fold cross-validation scheme was chosen for this project. Note that a 10-fold configuration is often considered as ideal, although a value of $k = 5$ was selected to reduce computational time without significantly compromising the reliability of the performance estimates.

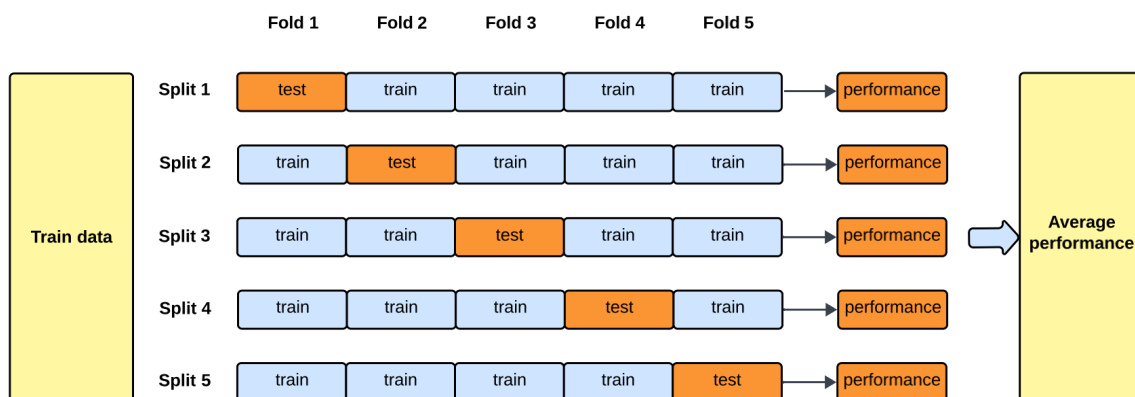


Figure 5.2: K - Fold train and test split [49].

The validation process of this project can be depicted in Figure 5.2 and was implemented as follows:

- The training set was partitioned into five equal-sized subsets (splits);
- In each iteration, one fold of each split was used as the validation set, while the remaining four folds were used for training;
- The model's performance was averaged across the five iterations.

This process was repeated multiple times within the **Optuna** optimization framework, which explored different hyperparameter configurations in order to maximize a predefined evaluation metric.

5.2 Model evaluation

The following section presents the metrics used to evaluate all classification models. It begins with the confusion matrix, from which key performance metrics, such as accuracy, precision, recall, specificity, negative predictive value and the F1-score are reported.

The analysis then focuses on the Receiver Operating Characteristic (ROC) curve and the Area Under the ROC Curve (AUC), which provides a threshold independent measure of a model's discriminative ability. Special attention is given to the role of the classification threshold (cutoff), whose adjustment can significantly influence model behavior.

Finally, the section addresses model interpretability by examining feature importance and introducing SHapley Additive exPlanations (SHAP) Values, offering insight into how individual features contribute to the model's decisions.

5.2.1 Confusion Matrix

The confusion matrix is a commonly used method for evaluating the performance of classification models by comparing predicted outcomes with the true values. Its structure is relatively straightforward, as illustrated in Table 5.1.

		Predicted Category	
		0	1
Real Category	0	TN	FP
	1	FN	TP

Table 5.1: Confusion matrix layout for binary classification

The above matrix can be interpreted as follows:

- **True Negative (TN)**: predicted rejected transaction and it was actually rejected;
- **False Negative (FN)**: predicted rejected transaction and it was actually approved (also known as Type II error);
- **True Positive (TP)**: predicted approved transaction and it was actually approved;
- **False Positive (FP)**: predicted approved transaction and it was actually rejected (also known as Type I error).

Based on the values obtained from the confusion matrix, it is possible to compute several key metrics that are essential for evaluating classification problems [50].

Starting with **Accuracy**, which measures the proportion of correctly classified instances, both positive and negative, relative to the total number of evaluated instances. It can be computed as shown in Equation 5.4:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (5.4)$$

Precision captures the proportion of correctly predicted positive observations out of all predicted positives. It becomes particularly relevant in contexts where minimizing FP is essential (Equation 5.5).

$$Precision = \frac{TP}{TP + FP} \quad (5.5)$$

Recall, also referred to as sensitivity, measures the ability of the model to identify all relevant positive instances (Equation 5.6).

$$Recall = \frac{TP}{TP + FN} \quad (5.6)$$

The **F1-score** represents the harmonic mean of precision and recall, offering a balanced metric that is particularly informative when dealing with class imbalance (Equation 5.7).

$$F1score = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} = \frac{2TP}{2TP + FN + FP} \quad (5.7)$$

Negative Predictive Value (NPV) reflects the proportion of negative predictions that are truly negative, similar to precision but for the negative label (Equation 5.8).

$$NPV = \frac{TN}{TN + FN} \quad (5.8)$$

Specificity evaluates the proportion of actual negatives correctly identified, serving as a counterpart to recall and proving useful in scenarios where avoiding false positives is critical (Equation 5.9).

$$\text{Specificity} = \frac{TN}{TN + FP} \quad (5.9)$$

In summary, the confusion matrix provides the foundational structure for interpreting classification results by breaking down predictions into four meaningful categories (TN, FN, FP and FN). The derived metrics, such as accuracy, precision, recall, F1-score, NPV and specificity offer complementary insights that help assess model performance.

5.2.2 AUC-ROC

The **Receiver Operating Characteristic (ROC) curve** is a widely used method to evaluate the performance of a binary classification model [51]. It provides insight into the model's ability to distinguish between positive and negative classes across various threshold levels, thereby illustrating how effectively the model separates the two labels (target values), as presented in Figure 5.3.

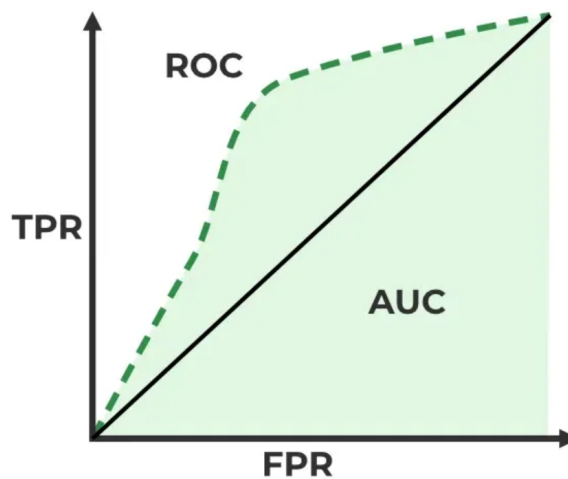


Figure 5.3: ROC-AUC Classification Evaluation Metric [52].

This is a two-dimensional plot of the True Positive Rate (vertical axis) versus the False Positive Rate (horizontal axis), where each axis represents the following:

- **True Positive Rate (TPR):** how often the model correctly predicts the positive cases also known as *Sensitivity* or *Recall*;
- **False Positive Rate (FPR):** how often the model incorrectly predicts a negative case as positive, which equals to $1 - \text{Specificity}$.

The ideal classifier corresponds to a point located in the upper left corner of the ROC space, indicating 100% specificity (no FP) and 100% sensitivity (no FN).

A point positioned along the diagonal line of the ROC curve represents a completely random classifier. This diagonal serves as a reference, dividing the ROC space into two regions: points below the line indicate performance worse than random guessing, while points above it reflect better than random classification performance.

To facilitate the comparison of different classifiers, it is standard practice to compute the **Area Under the ROC Curve (AUC)**. This metric condenses the model's overall performance into a single scalar value, offering an intuitive measure of its discriminative ability.

The AUC ranges from 0 to 1, and can be interpreted as follows:

- **AUC = 1** → The classifier perfectly distinguishes between the positive and negative classes;
- **AUC = 0** → The classifier makes completely incorrect predictions, assigning all instances to the wrong class;
- **AUC = 0.5** → The ROC curve coincides with the diagonal line, indicating that the classifier performs no better than random guessing;
- **AUC between 0.5 and 1** → The classifier demonstrates an increasing ability to distinguish between the two classes, with higher AUC values indicating better performance.

In conclusion, the ROC curve and the AUC score offer a comprehensive view of a model's discriminative capability across varying classification thresholds. By analyzing the trade-off between TP and FP rates, and summarizing performance through a single AUC value, these tools are especially useful when the costs of different types of errors must be analyzed.

5.2.3 Decision Threshold

In addition to evaluating classification performance through standard metrics and visualizations such as the confusion matrix and ROC curve, it is important to reflect on two critical aspects that further enhance the interpretability and performance of a ML model: the decision threshold and model explainability.

In binary classification problems, the default probability threshold (or cutoff) used to classify an instance as positive or negative is typically set at 0.5. That is, if the predicted probability of a given instance belonging to the positive class is greater than or equal to 0.5, the instance is classified as positive; otherwise, it is classified as negative.

However, this default setting does not always align with the objectives or constraints of a specific application. Adjusting the cutoff value allows for a trade-off between precision and recall, depending on which type of error (FP or FN) is more critical in the context:

- **Lower Threshold** – when higher recall is desired, capturing more TP at the cost of more FP is preferable; this also leads to fewer FN, since more positive predictions are made, and potentially fewer TN, as more negatives may be misclassified as positives.
- **Higher Threshold** – when higher precision is desired, minimizing FP is prioritized, even if it results in missing some TP; this typically increases FN, as fewer positive predictions are made, and increases TN, since more negatives are correctly classified.

It is important to highlight that, similar to hyperparameter tuning for optimizing a given evaluation metric, the decision threshold can also be modified to enhance model performance according to different scoring functions.

In this study, the decision threshold is adjusted to maximize selected evaluation metrics (such as Accuracy and F1-score) and a specific *Score Function*.

5.2.4 Feature Importance and SHAP Analysis

The use of ML techniques are often criticized because of their lack of interpretability, and they can be called a **black box**. Understanding not only which variables have an impact on the final decision, but also how it is making the prediction is very important.

To address this issue, feature importance techniques are employed to estimate the relative influence of each input variable on the model's predictions. In tree-based models, such as Decision Trees or Random Forest, feature importance is typically computed based on how much each feature contributes to reducing impurity (like Gini impurity) across all trees in the ensemble. Features that are used more frequently to split the data and lead to greater reductions in impurity are assigned higher importance scores.

These importance scores provide a global overview of which features the model relies on most, thereby enabling analysts to gain insight into the underlying structure of the data and potentially guide feature selection.

However, while useful, this type of feature importance does not offer insight into individual predictions or account for feature interactions and collinearity. This is where model explanation methods like SHAP (SHapley Additive exPlanations) values become valuable.

SHAP (SHapley Additive exPlanations) Value:

SHAP values are a unified, game-theoretic approach to explain the output of any ML model. Based on Shapley values from cooperative game theory, SHAP assigns each feature an importance value for a particular prediction [53].

The idea is to consider each feature as a *player* in a game where the task is to predict the target value. The SHAP value for a feature represents its average contribution to the prediction across all possible subsets of features, which makes it both consistent and locally accurate.

The generalized equation 5.10, gives the formula for the Shapley value of a player i of a p player game.

$$\Phi_i = \sum_{S \subseteq \{1, \dots, p\} \setminus \{i\}} \frac{|S|!(p - |S| - 1)!}{p!} [val(S \cup \{i\}) - val(S)] \quad (5.10)$$

Starting with the summation sign, the expression iterates over all coalitions S that do not include player i . In other words, S represents the set of all possible coalitions to which player i can contribute marginally.

Within the square bracket, the formula calculates the contribution of player i to the coalition S . Specifically, it is the difference between the value of coalition $S \cup \{i\}$ (including player $\{i\}$) and the value of coalition S alone.

Finally, the marginal contributions are weighted and each component of the weighting term serves a specific purpose:

- $|S|!$ $\longrightarrow$ number of ways players can join S before player i ;
- $(p - |S| - 1)!$ $\longrightarrow$ number of ways players can join coalition after player i joins;
- $p!$ $\longrightarrow$ number of ways to form a coalition of p players.

In the numerator of the weighting term lies the number of possible configurations in which the coalition S can be formed. The denominator corresponds to the number of possible arrangements of the full set of p players. Consequently, the weight represents the probability that player i contributes to a coalition of size $|S|$, in a game involving p participants [54].

The rationale behind this approach lies in its comprehensive nature: it considers not only each player's individual contribution but also the alliance that may arise from different groupings, acknowledging that some players may improve each other's value when grouped together. While this method appears to provide a fair allocation, it is important to note that alternative allocation rules might also seem equitable. Therefore, it is necessary to demonstrate that the

Shapley value uniquely satisfies a set of fairness axioms, thereby justifying its use as the principle method for value distribution.

The Shapley value is derived from three fundamental axioms, which serve as a formal definition of fairness in cooperative game theory. While these axioms can be expressed mathematically, a high-level summary is provided below. Any method of value allocation that satisfies these principles may be regarded as fair (see more in [55]).

- **Symmetry:** If two players contribute equally to every possible coalition, they are considered homogeneous. In such cases, both players must receive an identical share of the total value;
- **Null Player Property:** A player who contributes nothing to any coalition, whose marginal contribution is always zero, should receive no share of the total value;
- **Additivity:** When combining two distinct games, a player's total contribution should equal the sum of their contributions in each individual game. This axiom assumes independence between the games.

As previously discussed, Shapley values can be employed to interpret how a ML model arrives at a particular prediction. In this context, the *value of the game* corresponds to the model's output for a specific instance, while the *players* are the feature values associated with that instance.

It is important to note that it is not the features themselves, but their specific values for a given observation, that participate in the game. Nevertheless, for simplicity, these will be referred to as features. **Shapley values are then used to quantify the individual contribution of each feature to the final prediction.**

Each feature's contribution can be computed using the Shapley value formula presented in Equation 5.10, replacing the original value function with the one presented in Equation 5.11. This function is applied to a specific observation x , where S denotes a coalition of feature values, f represents the model's prediction function, and p refers to the total number of features.

$$val_x(S) = \int f(x_1, \dots, x_p) dP_{x \notin S} \quad (5.11)$$

The equation effectively involves performing multiple integrations. To marginalize over a feature, the model's prediction function is integrated with respect to the probability distribution of that feature's values. This process is repeated for all features not included in the coalition S .

More specifically, in the context of this project, the SHAP Python package has become the

standard tool for applying Shapley values in practice. Its extensive adoption is largely due to its efficiency in computing approximate Shapley values at scale.

In this project, the method employed is **TreeSHAP**, which is specifically designed for tree-based models, such as those used throughout this work.

As previously mentioned, these approximation techniques enable the analysis of numerous Shapley values, which can be aggregated in various ways to improve the interpretability of the model. For example, in credit risk modeling, SHAP can help explain why a specific loan application was approved or rejected, attributing the outcome to features such as income, credit score or existing debts [56].

Beyond interpretability tools such as the one now described, the evaluation of classification models relies on a diverse set of metrics and techniques, which provide complementary perspectives on model performance and are essential for a balanced assessment.

Combined with the model preparation process, these evaluation approaches form a solid foundation for the objective and transparent analysis of the developed models. The following chapter builds upon this foundation by applying the discussed concepts to the experimental results, comparing the performance of the different ML algorithms.

6 Experiments and Results

To evaluate the models and determine the most suitable one for this project, a hyperparameter tuning process was conducted using the Optuna framework, combined with 5-Fold Cross-Validation to ensure robustness and avoid overfitting (as reported in Chapter 5).

This process also included an analysis of hyperparameter importance, providing valuable insights into how these values influence model performance and how their optimization can significantly impact the results.

With the dataset structure, methodology and model training phases fully established, the final step is to evaluate each model on the reserved test set to assess their performance.

6.1 Model Comparison

This section provides a detailed comparative analysis of the selected models, incorporating the performance metrics previously discussed. The goal is to identify the most accurate and reliable model for each customer segment, ensuring its relevance for potential deployment in the **Pay No Pay** credit decision process.

6.1.1 Individuals

Starting with the Individuals customer segment, Table 6.1 provides a concise summary of the chosen evaluation metrics for all models.

Evaluation Metric	Decision Tree	Random Forest	XGBoost	LightGBM	CatBoost
accuracy	0.82	0.86	0.85	0.90	0.89
precision	0.81	0.81	0.82	0.88	0.87
recall	0.83	0.93	0.90	0.93	0.92
F1-score	0.82	0.87	0.86	0.91	0.89
NPV	0.83	0.92	0.89	0.93	0.92
specificity	0.81	0.79	0.81	0.88	0.87
AUC	0.896	0.935	0.933	0.965	0.958

Table 6.1: Model Performance for Individuals

Among the five models, **LightGBM** emerges as the top-performing classifier across nearly all metrics. It achieves the highest accuracy (0.90), F1-score (0.91) and AUC (0.965), indicating strong overall classification performance. Its high recall (0.93) and NPV (0.93) further suggest that it performs well in correctly identifying both positive and negative instances, which is critical in minimizing FN and FP.

The only metrics that fall slightly below the ideal, in comparison to the others, are precision (0.88) and specificity (0.88). This suggests that, while the model is highly effective in capturing true cases, it is relatively more prone to producing FP, indicating room for improvement in refining the model's ability to avoid misclassify negative instances as positive.

CatBoost follows closely behind, demonstrating comparable strength with an AUC of 0.958 and an accuracy of 0.89. Although its precision (0.87) is slightly lower than the one of LightGBM (0.88), it maintains a strong recall (0.92) and NPV (0.92), making it a reliable alternative with slightly different trade-offs.

XGBoost and Random Forest both demonstrate reasonably strong performance, though notably below the one obtained with LightGBM.

XGBoost achieves an accuracy of 0.85 and an AUC of 0.933, with balanced precision (0.82) and high recall (0.90). However, its overall performance does not live up to expectations, particularly given its reputation as one of the leading gradient boosting algorithms. These results suggest that, in this context, XGBoost does not represent a particularly competitive alternative.

Random Forest, while achieving high recall (0.93) and NPV (0.92) comparable to LightGBM, falls short in terms of precision (0.81) and specificity (0.79), indicating an even greater propensity for FP. While such a trade-off may be acceptable in scenarios where it is more critical to capture all positive cases, even at the expense of some false alarms, that is not the case in this context, making the model less suitable for the task at hand.

Finally, the **Decision Tree** model shows the lowest overall performance across most metrics, with an accuracy of 0.82 and AUC of 0.896. Despite being the most interpretable model, its relatively lower F1-score (0.82) and specificity (0.81) suggest that it lacks the predictive power and robustness seen in the ensemble approaches.

Furthermore, it is important to note that the AUC metric is particularly valuable when evaluating model performance across varying decision thresholds. To complement this analysis, Figure 6.1 presents a clear graphical comparison of each model behavior across all thresholds through the ROC curve.

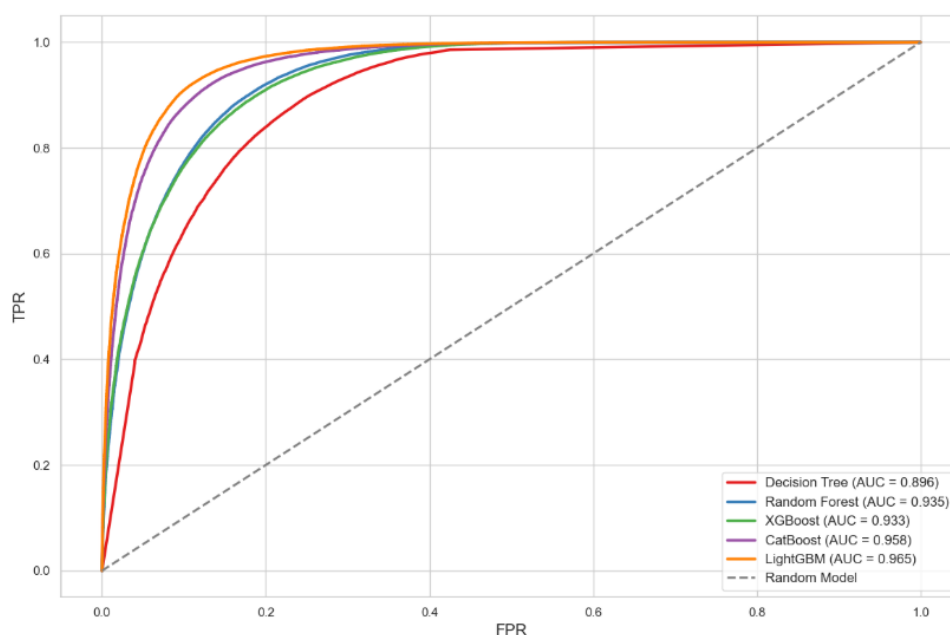


Figure 6.1: Models ROC Curve for Individuals

This visualization not only confirms the overall ranking of the models based on AUC, but also highlights how consistently they differentiate between positive and negative classes. For instance, LightGBM and CatBoost maintain high TPR even at low FPR, as evidenced by the shape of their curves.

The observed logarithmic trend (in Figure 6.1) further emphasizes the model's confidence in classification decisions across threshold variations, offering additional information beyond summary statistics.

In conclusion, while all models exhibit acceptable performance, **LightGBM** consistently outperforms across nearly all evaluation metrics, establishing itself as the most suitable model for the Pay No Pay process in the Individuals segment.

6.1.2 Entrepreneurs

Moving to the Entrepreneurs segment, Table 6.2 provides a concise summary of all evaluation metrics for the models under consideration.

Among the five models, **LightGBM** once again stands out as the top performer, achieving the highest accuracy (0.92), F1-score (0.92) and AUC (0.973). Its high recall (0.94) and NPV (0.95) further underline its effectiveness in correctly identifying both positive and negative outcomes.

The model also maintains strong precision (0.89) and specificity (0.90), suggesting a balanced ability to avoid both FP and FN.

Evaluation Metric	Decision Tree	Random Forest	XGBoost	LightGBM	CatBoost
accuracy	0.84	0.88	0.89	0.92	0.91
precision	0.85	0.83	0.86	0.89	0.89
recall	0.81	0.94	0.92	0.94	0.94
F1-score	0.83	0.88	0.89	0.92	0.91
NPV	0.83	0.94	0.92	0.95	0.94
specificity	0.86	0.82	0.86	0.90	0.89
AUC	0.863	0.949	0.958	0.973	0.969

Table 6.2: Model Performance for Entrepreneurs

Closely following is **CatBoost**, which shows similar performance across nearly all metrics. With an accuracy of 0.91, F1-score of 0.91 and AUC of 0.969, CatBoost represents a highly competitive alternative. Its precision and recall (both 0.89 and 0.94, respectively) are nearly indistinguishable from those of LightGBM.

XGBoost also performs well in this segment, achieving an accuracy of 0.89 and an AUC of 0.958. It demonstrates a favorable balance between precision (0.86) and recall (0.92), resulting in an overall F1-score of 0.89. While these results are respectable, they remain slightly below those of LightGBM and CatBoost.

Random Forest performs in accordance with XGBoost in some respects, particularly in recall (0.94) and NPV (0.94), which are on par with the top models. However, its lower precision (0.83) and specificity (0.82) suggest a greater inclination towards FP.

Finally, the **Decision Tree** model delivers the weakest performance, with the lowest AUC (0.863) and F1-score (0.83). Although it shows moderate strength in precision (0.85) and specificity (0.86), its limited performance across most metrics make this model less suitable for deployment.

As presented in the Individuals section, Figure 6.2 provides a visual complement to the numerical metrics, offering a comprehensive comparison of model performance through ROC curves.

This visualization further confirms the dominance of LightGBM and CatBoost, both of which maintain high TPR even at low FPR, an indication of strong discriminative power. Their curves exhibit a characteristic logarithmic trend, reflecting each model's confidence across varying thresholds and enhancing the analysis beyond aggregate metrics.

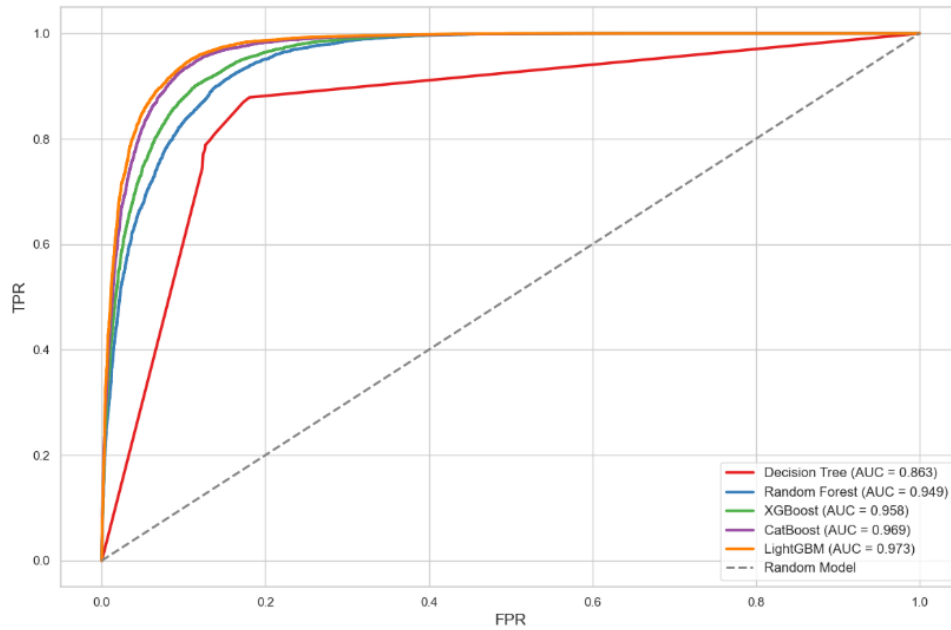


Figure 6.2: Models ROC curve for Entrepreneurs

In contrast, the less smooth and irregular curves of the remaining models show their comparatively weaker performance.

In conclusion, while all models demonstrate adequate predictive capacity in the Entrepreneurs segment, **LightGBM** consistently delivers the strongest and most balanced performance across all evaluation criteria, making it the preferred model. Additionally, CatBoost follows closely and remains a viable alternative, particularly when specific requirements may align with its strengths.

6.1.3 Companies

For the Companies segment, Table 6.3 summarizes the performance of all models across the key evaluation metrics.

In this segment, **LightGBM** once again stands out as the superior model, achieving the highest accuracy (0.95), F1-score (0.95) and AUC (0.988). Its strong precision (0.93), coupled with high recall (0.97), demonstrates excellent performance in capturing TP while maintaining a low FPR.

Similarly, the model maintains exceptional values for NPV (0.97) and specificity (0.93), confirming its balanced and reliable predictive capacity.

CatBoost closely follows, with similarly high values across all metrics. It matches LightGBM in both accuracy (0.95) and recall (0.97), while maintaining strong precision (0.92), F1-score (0.94) and AUC (0.985). This places CatBoost as a robust and nearly equivalent alternative, which was observed across all segments.

Evaluation Metric	Decision Tree	Random Forest	XGBoost	LightGBM	CatBoost
accuracy	0.91	0.92	0.94	0.95	0.95
precision	0.87	0.88	0.91	0.93	0.92
recall	0.94	0.97	0.96	0.97	0.97
F1-score	0.90	0.92	0.94	0.95	0.94
NPV	0.94	0.97	0.97	0.97	0.97
specificity	0.88	0.88	0.92	0.93	0.93
AUC	0.948	0.975	0.984	0.988	0.985

Table 6.3: Model Performance for Companies

XGBoost also delivers excellent results, slightly below LightGBM and CatBoost. With an accuracy of 0.94, precision of 0.91, and an AUC of 0.984, it exhibits a high level of discrimination and strong overall reliability. The model's slightly lower specificity (0.92) compared to the top two contenders is critical, and does not significantly detract from its strong performance.

The **Random Forest** model performs well overall, especially in terms of recall (0.97) and NPV (0.97). However, its lower precision (0.88) and specificity (0.88) indicate a higher rate of FP.

Finally, the **Decision Tree** model delivers surprisingly competitive results in this segment with an accuracy of 0.91 and F1-score of 0.90, although it still falls behind the ensemble models. Its lower precision (0.87) and specificity (0.88) indicate that, despite its interpretability, it lacks the classification confidence seen in more advanced algorithms.

As presented in the previous segments, the AUC metric offers an understanding of model performance across all classification thresholds. Figure 6.3 provides a graphical representation of this through the ROC curves for each model.

This visualization confirms the numerical values observed in Table 6.3, clearly distinguishing LightGBM, CatBoost and XGBoost as the most reliable classifiers.

The curves (in Figure 6.3) for these models exhibit a pronounced logarithmic shape, maintaining high TPR even at low FPR. This reflects a high degree of classification confidence and reinforces the overall performance trends.

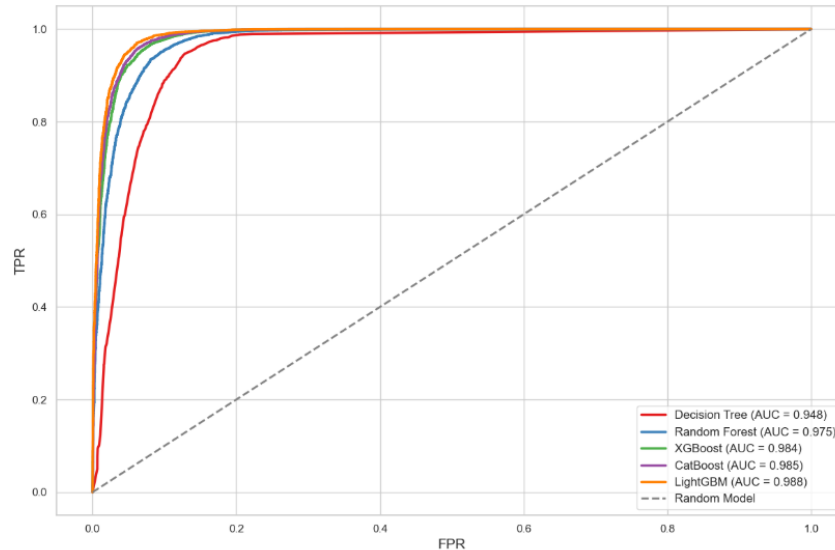


Figure 6.3: Models ROC curve for Companies

In conclusion, **LightGBM** emerges as the leading model for the Companies segment, consistently achieving the best results across all evaluation metrics. CatBoost and XGBoost also deliver exceptional performance, making them strong alternatives depending on deployment needs. Random Forest performs adequately but shows a greater susceptibility to FP, and finally the Decision Tree model remains the most interpretable, although it stays as the least competitive in terms of predictive strength.

As demonstrated across all customer segments, LightGBM consistently delivered the strongest performance, emerging as the most effective model overall. CatBoost closely followed, showing comparable results across most metrics. XGBoost occasionally approached CatBoost's performance but at times fell short of expectations. Random Forest, while generally robust, consistently ranked below XGBoost in terms of evaluation metrics. Lastly, the Decision Tree model showed the weakest performance by a significant margin, clearly falling behind the ensemble methods.

6.2 Model Final Assumptions

Building on the outcome of the previous section, the results are consistent with findings reported in the literature, where several studies have shown that LightGBM is particularly well-suited for credit risk and financial institution applications (for example [14]). Therefore, it is not surprising that LightGBM emerged as the best model for this project, which is known to adapt effectively to the characteristics of such datasets.

Consequently, to optimize the current **Pay No Pay** process as effectively as possible, **LightGBM** is identified as the most suitable model. Therefore, the subsequent analysis will focus

on this model, exploring different classification thresholds to evaluate whether alternative cutoff points can yield more favorable trade-offs between key performance metrics.

Additionally, to enhance interpretability and gain deeper insights into the model's decision-making process, a comprehensive analysis will be performed using **SHAP** values.

6.2.1 Individuals

Starting with a more detailed analysis of the **LightGBM** model for the **Individuals** client segment, one can claim that given the nature of the task, minimizing both FP and FN is essential, as these errors carry greater implications in the current application context. To better understand how the model can be adjusted in order to minimize the impact of these two errors, it is useful to look beyond standard evaluation metrics and first examine the confusion matrix, presented in Table 6.4, presented here as percentages of the total sample, the confusion matrix offers an anonymized, however, insightful overview of the model's performance.

As shown, approximately 44.7% of all samples are TN and 45.8% are TP, while for FP and FN there are only 6.3% and 3.2% samples, respectively.

		Predicted Category	
		0	1
Real Category	0	44.7%	6.3%
	1	3.2%	45.8%

Table 6.4: LightGBM Confusion matrix for Individuals

For an interpretation of the other models within this customer segment and to complement the evaluation metrics summarized in Table 6.1, anonymized confusion matrices for the alternative models are provided in Appendix A.

It is important to note that the confusion matrix above corresponds to a decision threshold (cutoff) of 0.5, which is the standard default and from which the performance metrics in Table 6.1 were derived. However, adjusting this cutoff can influence the balance between FP and FN.

As observed in Table 6.4, the number of FN is lower than the number of FP, which implies that the model tends to approve more cases overall, even with increased risk, as it results in a higher number of incorrect approvals. To develop a model where the opposite holds, where the number of FN exceeds the number of FP, it becomes necessary to increase the decision threshold. As is well established, increasing the cutoff typically reduces FP, although at the cost of raising the number of FN.

To investigate this further, threshold optimization was performed on the LightGBM model to identify cutoff values that maximize the F1-score and accuracy. This threshold tuning is carried out after model training and operates independently of the hyperparameter optimization process. Its primary purpose is to tune the decision boundary to potentially improve the model's performance on some specific metrics.

In addition to this exploration and to address this trade-off more explicitly, one can define a weighted objective function that imposes a greater penalty on FP relative to FN. For example, consider the score function presented in Equation 6.1:

$$\text{Score (threshold)} = -(0.7 \cdot FP + 0.3 \cdot FN) \quad (6.1)$$

By selecting the threshold that minimizes this cost function, one can effectively minimize FP while still accounting for FN with a lesser weight. Additional experiments with varying weight combinations were conducted across all segments, although this weighting scheme consistently obtained the most favorable results.

Given that the LightGBM model exhibits an almost ideal trend in its ROC curve and considering it outperforms other models across evaluation metrics, it serves as the optimal candidate for this threshold variation study. Table 6.5 summarizes the performance of the LightGBM model for the Individuals client segment under various threshold selection strategies. The primary objective of this analysis is to reduce the number of FP, thereby offering an alternative perspective different from the one presented with the standard cutoff.

Cutoff Strategy	Cutoff Value	TN	FP	FN	TP
None	0.5	44.69	6.29	3.25	45.78
F1-score	0.498	44.66	6.32	3.22	45.80
Accuracy	0.556	45.44	5.53	3.95	45.07
Score function	0.722	47.42	3.55	7.23	41.79

Table 6.5: Threshold Study for Individuals

The obtained results reveal that the default cutoff of 0.5 serves as a solid baseline, with 6.29% of FP and a well-balanced distribution of TP (45.78%) and TN (44.69%). Optimizing for F1-score (cutoff = 0.498) basically leads to identical results, as there is almost no difference in the obtained cutoff.

The accuracy optimized cutoff (0.556) offers a meaningful reduction in FP to 5.53%, with

only a minor drop in TP (45.07%) and a slight rise in FN (3.95%), presenting a balanced and practical alternative.

The custom cost function strategy (cutoff = 0.722), while achieving the lowest FP rate at 3.55%, results in a higher FN rate (7.23%) and a reduction in TP to 41.79%, thereby clearly prioritizing the minimization of FP at the expense of recall.

Depending on the specific context and risk tolerance of the financial institution, each of these cutoff strategies allows for a different level of strictness in approval decisions. Among them, the threshold derived from accuracy optimization offers a more balanced compromise, neither overly conservative nor too permissive, and may therefore represent the most appropriate choice when the objective is to avoid extreme trade-offs in either direction.

After analyzing the performance of the best model and its potential to adapt to the evolving economic landscape of the financial institution, it is now essential to conduct a more detailed analysis of the model itself. The goal is to understand why the model makes certain decisions, based on the variables presented in the dataset.

To achieve this, SHAP values serve as an essential tool, enabling a comprehensive understanding of complex/black-box ML models. This level of interpretability is particularly crucial in the financial domain, where transparency and explainability are not only expected but often demanded.

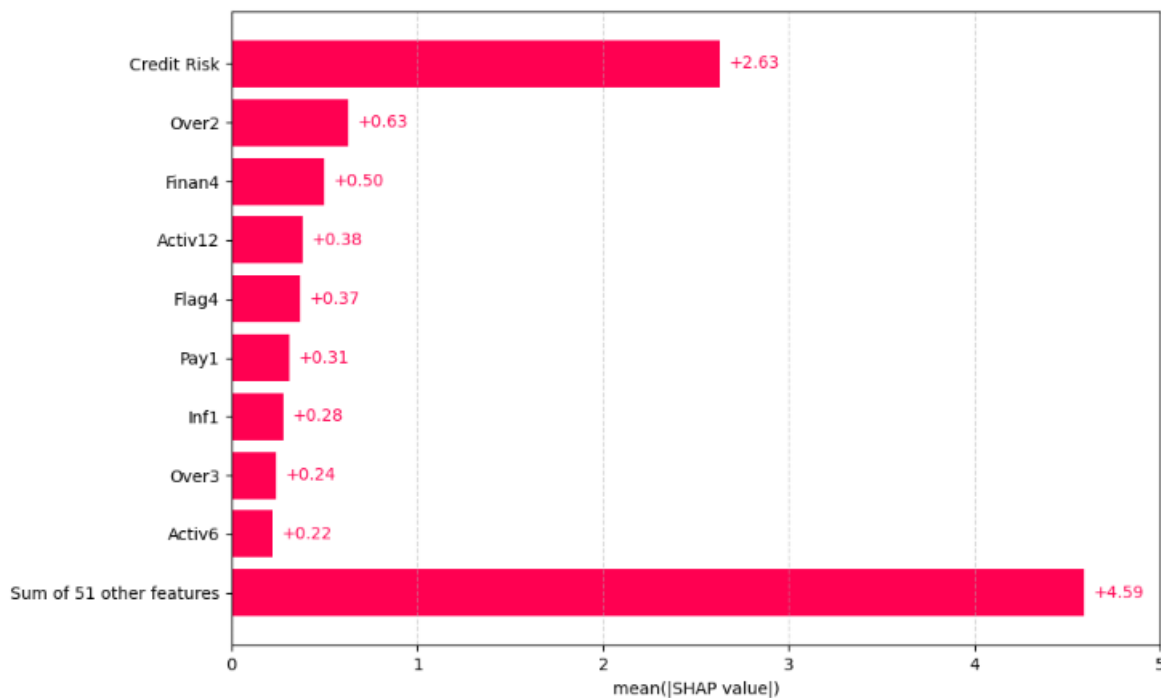


Figure 6.4: Mean SHAP Value of LightGBM Individuals model

The bar chart, presented in Figure 6.4, illustrates the mean absolute SHAP values for the LightGBM model, providing insights into the relative importance of the input features in shaping the model's predictions. Each bar represents the average magnitude of SHAP for a given feature across all predictions, thereby quantifying its overall impact regardless of whether its effect is positive or negative.

By the visualization of 6.4, one can infer that the most dominant feature is obviously *Credit Risk*, which shows the highest mean SHAP value at 2.63, indicating it is by far the most influential variable in the model's decision-making process, with the model relying heavily on this variable to distinguish between outcomes. This result is both expected and consistent with typical patterns observed in credit modeling.

Other features, such as *Over2* (0.63) and *Finan4* (0.50), follow in importance but exhibit considerably lower mean SHAP values compared to *Credit Risk*, indicating a smaller but still meaningful contribution to the model's predictions. Additional variables, including *Activ12* (0.38), *Flag4* (0.37) and *Pay1* (0.31), among others, place a moderate influence, suggesting they may function as complementary indicators or capture correlated patterns within the data.

Notably, the combined contribution of the remaining 51 features is substantial, with a cumulative mean SHAP value of 4.59. This underscores that while many individual features have limited importance, their collective impact can even surpass the value of the most influential variable.

It is important to note that the chart illustrated in 6.4 presents absolute SHAP values, which reflect the magnitude of a feature's importance but not its direction of influence (whether it drives the prediction higher or lower). To explore the directional impact of features, one must consult different types of visualizations, which incorporate both SHAP value signs and feature value gradients.

In this project, a **Beeswarm Plot** was selected to further explore some of the most influential features of the model (see Figure 6.5). This type of visualization, provided by the SHAP package, offers a more detailed and interpretable view of feature impact. In the plot, each point represents an individual instance from the dataset, with color indicating the value of the corresponding feature (red for high values and blue for low values). The x-axis displays the SHAP value (not its absolute value), which reveals the direction of the feature's effect, where positive values indicate that the feature pushes the prediction higher, while negative values suggest it drives the prediction lower.

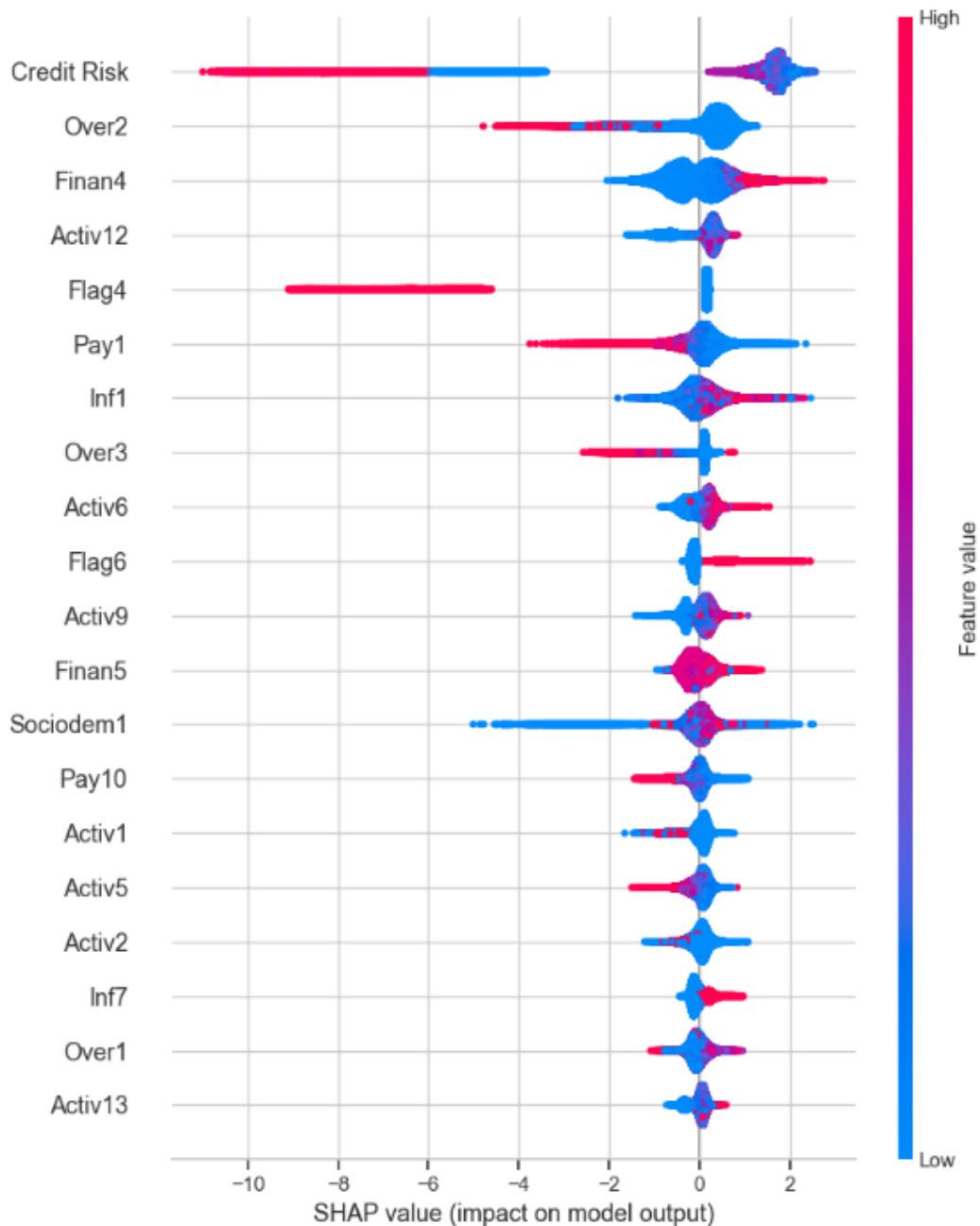


Figure 6.5: Beeswarm Plot of LightGBM Individuals model

Features such as *Credit Risk* display a clear directional trend, where higher values (in red), which correspond to clients with higher risk, and the lowest values (in blue), which represent the non-attributive or neutral categories, are predominantly associated with negative SHAP values. This indicates a tendency to decrease the model's output, which, in the context of this project, will typically result in rejection.

In the case of *Flag4*, higher values are also linked to negative SHAP values, suggesting that this feature contributes consistently toward rejection, regardless of its specific magnitude.

For *Pay1*, one can observe a clear threshold between positive and negative SHAP values.

Higher feature values are concentrated on the negative side of the x-axis, whereas lower feature values appear more frequently on the positive side. This pattern suggests a well-defined threshold effect within the model's decision-making process.

In more complex cases, such as *Finan5*, both red and blue points are widely distributed across both sides of the x-axis. This dispersion suggests that the same feature can exert both positive and negative influences, depending on the context in which it appears, likely reflecting interactions or conditional relationships with other features.

Overall, the Beeswarm plot confirms that the model's behavior is consistent with domain knowledge. For instance, when *risk-related* variables have higher values, the model tends to lower the probability of approval, increasing the likelihood of rejection, which aligns with what is expected in credit risk evaluation.

Additionally, some features exhibit a high dispersion of SHAP values, implying significant variation in their influence across individual cases. This variability often points to nonlinear effects that back up the need for further investigation.

6.2.2 Entrepreneurs

Proceeding with the same analysis for the **LightGBM** model in the **Entrepreneurs** client segment, it is essential to explore the confusion matrix presented in Table 6.6. As shown, approximately 46.71% of all samples are TN, while 45.23% are TP. FP and FN represent 5.37% and 2.69%, respectively.

		Predicted Category	
		0	1
Real Category	0	46.71%	5.37%
	1	2.69%	45.23%

Table 6.6: LightGBM Confusion matrix for Entrepreneurs

For a comprehensive interpretation of the other models within this customer segment and to complement the evaluation metrics summarized in Table 6.2, anonymized confusion matrices for the alternative models are provided in Appendix B.

In this section, the same analysis performed for the Individuals segment is extended to the Entrepreneurs segment, with Table 6.7 presenting the results of the threshold study.

Cutoff Strategy	Cutoff Value	TN	FP	FN	TP
None	0.5	46.71	5.37	2.69	45.23
F1-score	0.471	44.61	5.46	2.57	45.36
Accuracy	0.520	46.79	5.28	2.80	45.13
Score function	0.908	49.17	2.90	6.41	41.52

Table 6.7: Threshold Study for Entrepreneurs

The analysis of Table 6.7 shows that when optimizing for the F1-score (cutoff = 0.471) a slightly increase of FP to 5.46% occurs, while reducing FN to 2.57%. This indicates a minor improvement in recall, with minimal overall impact on model performance compared to the default threshold.

The accuracy optimized threshold (cutoff = 0.520) results in a modest improvement, lowering FP to 5.28% while maintaining balanced TP (45.13%) and TN (46.79%) rates.

The custom cost function strategy (cutoff = 0.908) produces the lowest FP rate at 2.90%, in line with the study's objective to minimize such errors. However, this comes at the cost of increased FN (6.41%) and a reduction in TP to 41.52%, demonstrating a clear shift at the expense of recall.

Overall, the results presented in Table 6.7 underscore the possible trade-offs in threshold selection for the Entrepreneurs segment. As observed for the Individuals segment, the choice of cutoff should be carefully aligned with the institution's risk tolerance and operational objectives. It is also worth noting that more balanced outcomes could potentially be achieved by selecting cutoff values between 0.5 and 0.908.

Now, moving on to the additional framework of SHAP, which enables a comprehensive understanding of complex, black-box ML models (in this case the LightGBM).

The same study presented in the Individuals segment is now introduced, where the bar chart, presented in Figure 6.6, illustrates the mean absolute SHAP values, providing insights into the relative importance of the input features in shaping the model's predictions. Each bar represents the average magnitude of SHAP for a given feature by quantifying its overall impact regardless of whether its effect is positive or negative.

By the visualization of 6.6, one can infer that the most dominant feature is obviously *Credit Risk* (same for the Individual segment), which shows the highest mean SHAP value at 3.87, indicating it is by far the most influential variable.

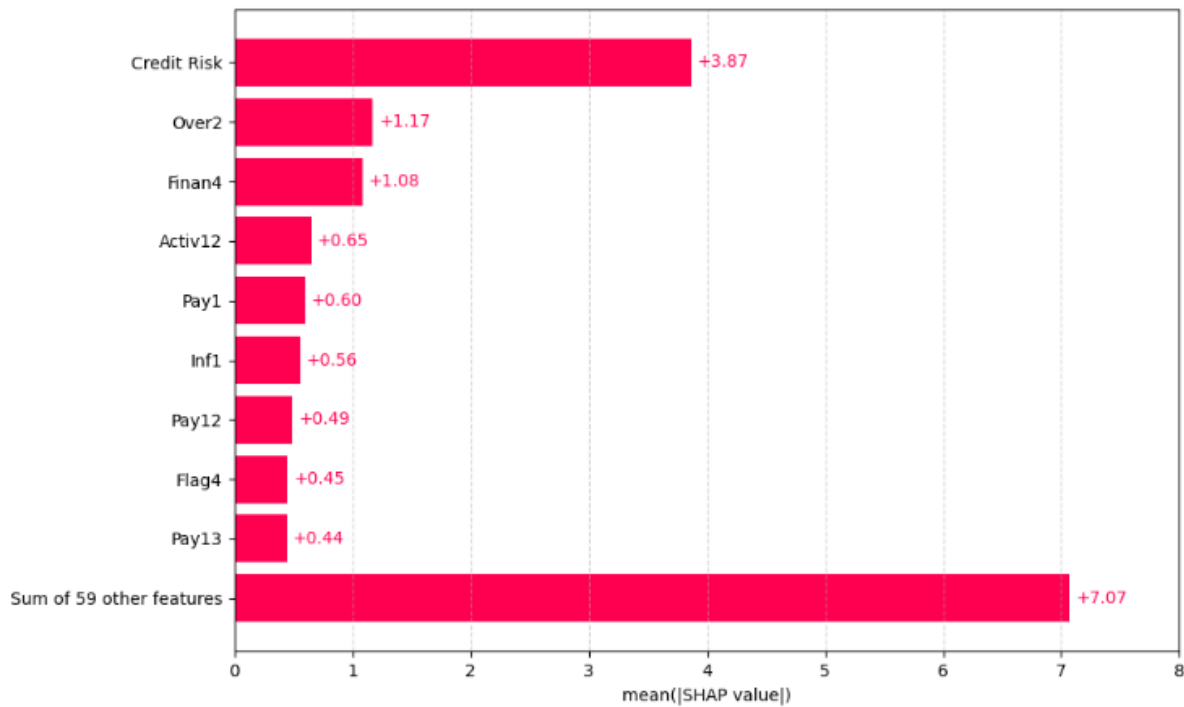


Figure 6.6: Mean SHAP Value of LightGBM Entrepreneurs model

Other features, such as *Over2* (1.17) and *Finan4* (1.08), follow in importance but exhibit considerably lower mean SHAP values compared to *Credit Risk*, indicating a smaller but still meaningful contribution to the model's predictions. Additional variables, including *Activ12* (0.65), *Pay1* (0.60), and *Inf1* (0.56), among others also appear with some relevance in the decision-making process.

Moreover, the combined contribution of the remaining 59 features is considerable, with a cumulative mean SHAP value of 7.07. This highlights the importance of all other features in the dataset.

The behavior observed in this plot closely resembles that presented in Figure 6.4 for the Individuals segment, with only a few different variables emerging as more prominent.

In addition to this more concise graph regarding the capability of SHAP values to interpret a model and the influence of selected variables on decision-making, the Beeswarm Plot for this Client Segment is presented in Appendix D.1. This plot illustrates the impact of feature values on the probability of rejecting or approving a transaction. After reviewing the analysis presented in the Individual segments, one can fully understand the plot shown.

6.2.3 Companies

Finally, focusing on the **Companies** client segment, Table 6.8 presents the confusion matrix for the **LightGBM** model. As illustrated, approximately 49.77% of all samples are correctly

identified as TN, while 45.45% are TP. In addition, FP and FN report the values of 3.48% and 1.30%, respectively.

		Predicted Category	
		0	1
Real Category	0	49.77%	3.48%
	1	1.30%	45.45%

Table 6.8: LightGBM Confusion matrix for Companies

Similar to the other segments, all confusion matrices for the alternative models, which complement the evaluation metrics presented in Table 6.3, can be found in Appendix C.

The detailed analysis of the cutoff variation is now extended to this client segment, as presented in Table 6.9.

Cutoff Strategy	Cutoff Value	TN	FP	FN	TP
None	0.5	49.77	3.48	1.30	45.44
F1-score/Accuracy	0.550	49.92	3.32	1.38	45.38
Score function	0.895	50.92	2.34	2.65	44.09

Table 6.9: Threshold Study for Companies

Table 6.9 presents the performance of the model for the Companies segment under different cutoff strategies. In this case, the default cutoff (0.5) serves as a strong baseline, with a very low FP (3.48%) and FN (1.30%) rates, and of course a very balanced values for TP (45.44%) and TN (49.77%).

Optimizing for either F1-score or accuracy (cutoff = 0.550) leads to a modest improvement, reducing FP to 3.32% while only slightly increasing FN to 1.38%, indicating improvement for the model precision with minimal compromise to recall.

The custom score function (cutoff = 0.895), which prioritizes reducing FP, is the most effective in minimizing it down to just 2.34%. However, this results in an expected increase in FN to 2.65%, and a reduction in TP to 44.09%.

In summary, the analysis of Table 6.9 underscores the value of cutoff optimization in aligning model performance with specific operational objectives. For the Companies segment, both the default and the accuracy/F1-score thresholds lead to very balanced results, maintaining high levels of precision and recall. However, the custom cost function stands out by significantly

reducing FP and helping to stabilize the balance between FP and FN. So one can conclude that this optimization emerges as the most suitable option when the goal is to minimize risk without incurring extreme trade-offs in either direction.

Now, turning to the extended SHAP framework, which will facilitate a comprehensive interpretation of the complex LightGBM model.

The same analytical approach applied to the other two segments is now presented for the Companies segment. The bar chart in Figure 6.7 displays the mean absolute SHAP values, offering insights into the relative importance of each input feature in shaping the model's predictions.

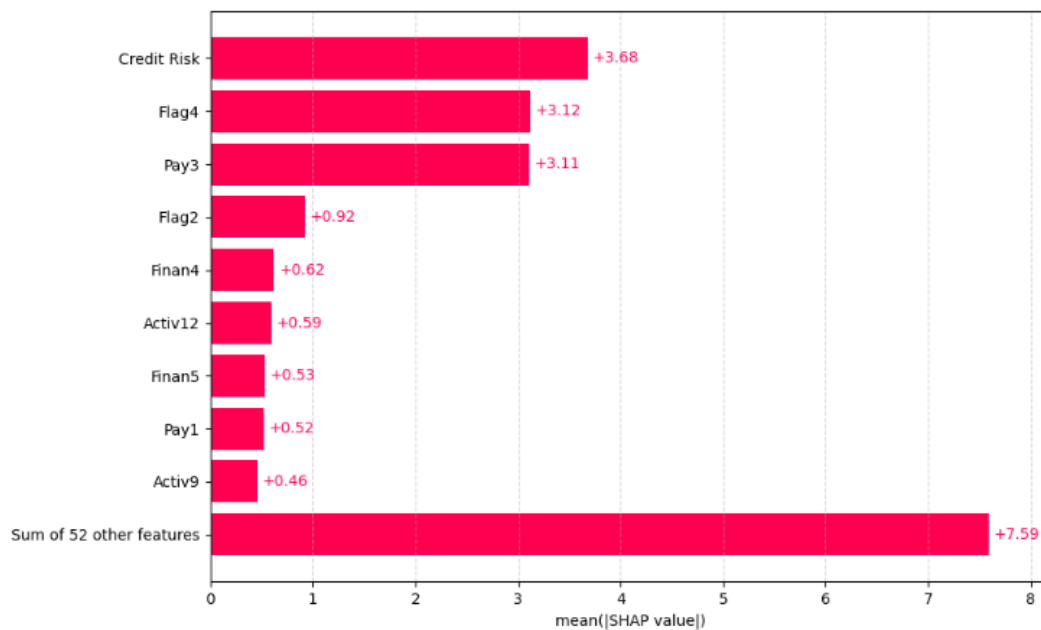


Figure 6.7: Mean SHAP Value of LightGBM Companies model

From the visualization of Figure 6.7, it is evident that the most dominant feature is *Credit Risk*, which exhibits the highest mean SHAP value at 3.68. This indicates that it is by far the most influential variable in the model's decision-making process, as it already was for the Individual and Entrepreneurs segments.

The two features *Flag4* (3.12) and *Pay3* (3.11), follow closely in importance with some meaningful contribution to the model's predictions. Additional variables, including *Flag2* (0.92), *Finan4* (0.62) and *Activ12* (0.59), among others, paly a moderate influence.

Notably, the combined contribution of the remaining 52 features is substantial, with a cumulative mean SHAP value of 7.59. Among these variables are several *business-related* features, which also have a significant impact on the decision-making process.

Complementing this summary chart, the Beeswarm Plot for the Companies segment is pro-

vided in Appendix D.2. With the fundamental analysis already established in the Individuals segment, this plot can now be more fully understood in context.

Throughout this section, a study was conducted on the impact of varying the cutoff threshold on LightGBM's performance across all client segments. The goal was to demonstrate the model's flexibility in adapting to the financial institution's specific priorities, whether that means being more strict about FP, minimizing FN, or even achieving a balance between the two.

Additionally, an interpretability analysis was carried out to better understand which variables most influence the LightGBM model's decisions (with the help of SHAP). The results show that the model's decision-making aligns well with expectations in the context of credit risk modeling.

6.3 Future Improvements

While the current study has demonstrated the strong performance of the LightGBM model across all client segments, several improvements remain open for future enhancement and operational refinement.

First, the current segmentation of customers into Individuals, Entrepreneurs and Companies could be further improved. Within these broad groups, especially in the Company segment, important subcategories such as small, medium and large companies may exhibit significantly different credit behavior. The analysis presented in Chapter 3 highlighted that the Company segment posed the greatest challenge in distinguishing between good and bad clients. This suggests that a more granular segmentation strategy could provide more homogeneous groups, in order to learn more distinct patterns and improve classification accuracy.

Secondly, although the models were rigorously validated using a 5-Fold (Cross-Validation) technique, a live pilot test phase would provide critical insights into *real-world* behavior. Deploying the selected models in a controlled environment and monitoring their impact over a defined period would help uncover potential issues not captured during training (and testing) and would help validate model assumptions under operational restraint. This phase would also allow the financial institution to more accurately evaluate the business impact of the chosen models.

Moreover, future work could focus exclusively on the LightGBM model, which has proven to be the most effective across all segments. By narrowing the scope to a single model, additional fine-tuning (with, for example, more cutoff analysis) could be achieved to further analyze FP and FN.

Finally, interpretability remains an essential factor in model deployment. Integrating more explainability tools (besides SHAP) would help clarify the reason behind each decision, improv-

ing stakeholder trust, and acquiring agreement with some regulatory standards.

In conclusion, future work should focus on refining segmentation, validating model performance through real-world testing, concentrating optimization efforts on the LightGBM model and incorporating more explainability tools. These improvements would significantly contribute to a credit decision system more aligned with the financial institution's objectives.

7 Extra Work

In addition to the main modeling work presented in the previous sections, this chapter presents the results of additional work carried out after the completion of the process described throughout this report. This study was initiated at the request of the financial institution, which desires to further simplify and improve the **Pay No Pay** process. Although considered extra work due to its slight deviation from the main objective of improving the current model of the process, it still contributes to that goal by approaching it from a different perspective.

This chapter extends the analysis previously presented, with the goal of streamlining the decision-making process. Specifically, it investigates how to reduce the number of low-risk transactions that require model assessment, while still maintaining control over (potentially) all approvals. This complementary analysis provides valuable information for improving both operational efficiency and risk management practices.

To guide this exploration, the following question must be posed: ***Is it possible to determine a monetary threshold value to automatically approve certain transactions, so that not all transactions would need to go through the obtained model?***

To achieve this, it is necessary to determine monetary threshold values down to which transactions can be automatically approved. The aim is to ensure that only a small number of transactions that the model would otherwise classify as rejected are accepted, while still retaining a meaningful share of transactions that the model would approve. This carefully calibrated monetary limit will be referred to as the **Shadow Limit**.

7.1 Shadow Limit

Following this concept, the new flow for the Pay No Pay process is shown in Figure 7.1, which incorporates the Shadow Limit concept. In this flow, a preliminary analysis is performed based on the Shadow Limit, which can lead to an automatic approval or a final evaluation by the model obtained.

This study was conducted with the focus on the best model, the **LightGBM**, which considering that this model had the best results, it makes sense to base the analysis on it. Furthermore, it is important to note that this approach is most applicable to **Individuals**, as the average transaction amounts in this process tend to be lower compared to other cases and also exhibit a more linear behavior.

Before presenting the process to achieve the shadow limit, it is essential to note that only clients, which the financial institution considers eligible for a decision without requiring model

intervention are analyzed. To identify these cases, a set of specific criteria was applied, even more strict than those previously outlined for defining the new target variable in Chapter 3.

If any of these conditions are met, the client is immediately excluded from the Shadow Limit. This early exclusion helps to eliminate cases that could potentially distort the analysis, ensuring a more accurate and reliable study.

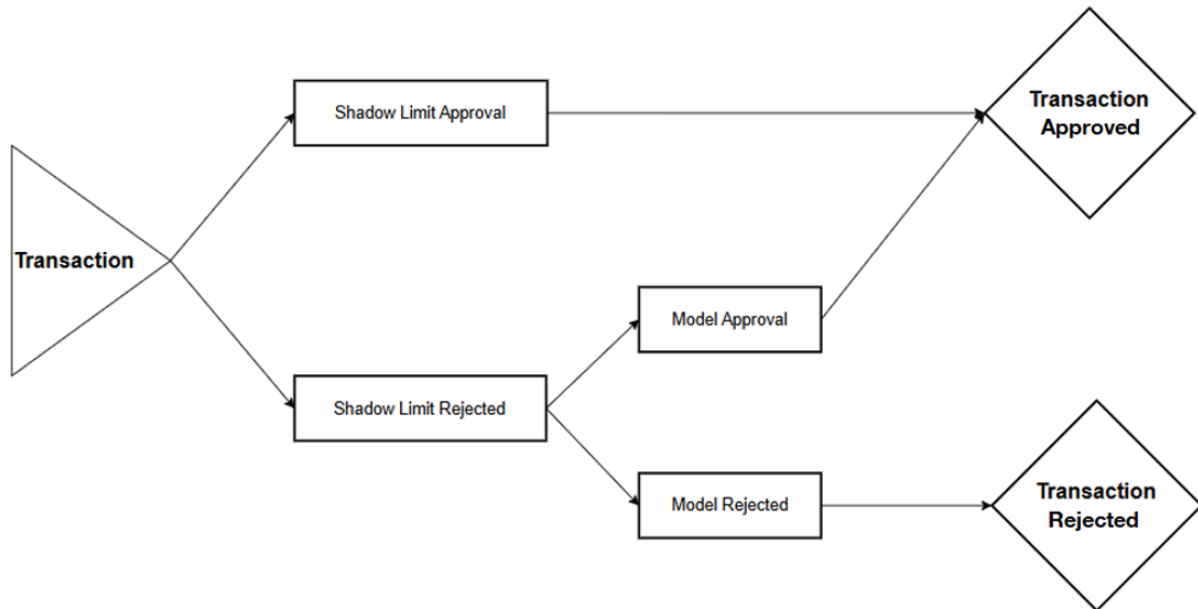


Figure 7.1: **New Pay No Pay Flow**

In this part of the analysis, and based on Table 3.1, the variable *Pay1*, associated with the transaction amount, was examined in relation to the *Pay6* features, which offered a detailed and specific segmentation of the transactions involved in the process.

The shadow limit can be calculated using Equation 7.1:

$$\text{Shadow Limit}_t = Q_p \{ \text{Pay1}_i \mid \text{Pay6}_i = t \wedge \hat{y}_i = 1 \} \quad (7.1)$$

Where for each type of transaction t , the Shadow Limit is defined as the p -th quantile Q_p of the transaction amounts Pay1_i , computed over all **Approved** transactions ($\hat{y}_i = 1$) where the feature Pay6_i is equal to t .

For instance, setting $p = 0.5$ results in a Shadow Limit equal to the median of the approved transaction amounts for type t , capturing the central tendency of those transactions.

This specific threshold reflects the typical transaction behavior within each *Pay6* feature, where by varying the chosen percentile, different Shadow Limit values can be obtained. Therefore, the following section presents the most beneficial results in this study.

The following extra work, as previously justified, will be applied solely to the Individ-

uals client segmentation.

7.2 Shadow Limit Evaluation

Moving on to the results obtained through this approach, the Shadow Limit values for each *Pay6* variable are presented in Table 7.1, where the percentage of transactions correctly and incorrectly approved by the Shadow Limit is also shown, relative to the total number of transactions within each variable of *Pay6* that entered the **Pay No Pay** process and is eligible to fall under this new process.

Note that the exact value of each Shadow Limit is kept undisclosed in this analysis.

Variable <i>Pay6</i>	TP approved (%)	TN approved (%)
α_1	51.21	25.06
α_2	58.51	15.96
α_3	70.79	6.74
α_4	58.63	17.73
α_5	57.29	18.12
α_6	67.27	8.39
α_7	62.77	12.80
α_8	60.89	10.98
α_9	59.55	15.01
α_{10}	61.79	13.01
α_{11}	64.57	8.15
α_{12}	71.19	5.01
α_{13}	71.67	5.55
α_{14}	52.88	17.66

Table 7.1: Shadow Limit Evaluation

The variation of the p -th quantile was studied considering the numerical values of the Pay No Pay transactions from the perspective of the financial institution. However, since the primary objective here is to demonstrate the quality of the results, this value will consistently be set to 0.8.

Also, only the features with a significant number of transactions are included in these results, ensuring that the analysis focuses on cases with the greatest potential impact.

The first observation of Table 7.1 shows the consistent performance of the shadow limit rule across most variables in terms of **true positive approval**, with several values exceeding 60%. Notably, variables α_3 , α_{12} , and α_{13} show particularly strong results, each with over 70% of positive cases correctly approved by the Shadow Limit.

Conversely, the **true negative approval** rate remains low across most features. This indicates a cautious behavior of the rule, as it avoids incorrectly approving cases that should

remain rejected, evident in variables such as α_3 , α_{12} , and α_{13} , where true negative approval stays below 7%.

The fixed p -th quantile of 0.8, selected based on the observed transaction distributions, allows the rule to cover a significant share of valid approvals without introducing excessive risk through false (true negative) approvals.

From the analysis above, it is evident that the Shadow Limit strategy can provide a meaningful contribution to the **Pay No Pay** process by automatically approving a considerable portion of transactions deemed acceptable by the predictive model.

The motivation behind this study lies in the potential to define a transaction threshold, **Shadow Limit**, in which approvals can be made without the need of more complex decision-making systems. Although it is not possible to obtain this approach for all types of transactions due to heterogeneity and volume limitations, it does suggest that targeted automation is possible.

7.3 Final Assumptions

While this study was limited to the Individuals client segment, due to their low transaction values and more consistent behavior, the concept of the Shadow Limit shows promising potential. Applying this approach to business clients is currently not possible given the significantly higher transaction values, which introduce greater variability and complexity.

To extend this methodology to other client segmentations and improve overall accuracy, future work should focus on the development of regression models, which would enable a more precise definition of the Shadow Limit, moving beyond the fixed percentile approach. Such models could incorporate more specific features to dynamically adapt approval criteria, improving both accuracy and operational efficiency.

Additionally, integrating the Shadow Limit mechanism directly into the decision-making pipeline could further optimize the process. By filtering out low-risk transactions before they reach the predictive models, one can reduce their workload and upgrade their focus on more complex cases. This would not only optimize the use of existing models, such as the LightGBM approach used in this study, but also lead the way for their refinement, targeting only the most challenging decisions.

Furthermore, this auxiliary method demonstrates that a portion of transactions can be reliably approved without invoking the full predictive model. For those that cannot, the model continues to serve as a robust and informed decision tool. Together, these strategies offer a more scalable, adaptive and intelligent credit decision system.

8 Conclusion

This project has provided a comprehensive investigation of the application of ML techniques for improving the Pay No Pay decision-making process. The main challenge was the authorization of debit transactions on accounts with insufficient funds, an operationally time sensitive task that balances risk exposure with customer satisfaction. The proposed solution integrates advanced classification models into the existing process to replace manual assessments with reliable automation.

A crucial aspect of this work was the client segmentation into three categories: Individuals, Entrepreneurs and Companies, based on their distinct financial behaviors and data characteristics. This segmentation enabled the construction of specialized models for each group, allowing the algorithms to better capture the refinements of each client profile.

Another major contribution was the definition of a revised target variable, which analyzed the accuracy of past approval and rejection decisions using a 30-day window to observe customer behavior after the transaction. This redefinition not only added robustness and temporal depth to the training dataset but also aligned with minimizing credit loss while maximizing value recovery.

Several classification algorithms were implemented using the Optuna framework for hyperparameter tuning, including Decision Tree, Random Forest, XGBoost, LightGBM and CatBoost. Among these, gradient boosting methods, particularly **LightGBM**, consistently demonstrated superior predictive performance across all client segments, achieving an optimal trade-off between accuracy and computational efficiency.

The inclusion of SHAP analysis further enhanced model transparency, making it possible to identify and justify the contribution of individual features to model predictions, which is an essential requirement in highly regulated domains such as this.

To further refine model performance and applicability, a dedicated study was conducted on the variation of the decision threshold (cutoff). This analysis enabled a deeper understanding of the trade-offs between sensitivity and specificity, allowing for the tuning of classification boundaries according to business priorities. By adjusting the cutoff, financial institutions can better calibrate their tolerance for FP or FN, enhancing control over risk exposure while aligning with future objectives.

In addition to core modeling, this project introduced and evaluated an **Extra Work** known as the Shadow Limit, which is a pre-authorization mechanism for transactions with lower associated risk. This concept, if implemented effectively, can streamline transaction processing by

reducing *unnecessary* rejections (that were initial made by the obtained model), contributing to an increase in operational efficiency.

Looking ahead, several opportunities appear for extending and refining this work, both in terms of modeling strategy and operational integration (for the main modeling work, and also for the extra work presented).

One promising enhancement is the refinement of client segmentation. While the current division into Individuals, Entrepreneurs and Companies has enabled model specialization, further specification, particularly within the Company segment, could significantly improve predictive performance.

Another critical step for future work involves validating model performance through a pilot test phase, which would provide critical insights into real-world behavior and, of course, could improve even more the number of FP and FN, uncovering potential discrepancies not captured during offline testing, and assessing the actual business impact of the new decisions. Although this pilot was not yet in place during the internship, the financial institution began planning its implementation at the end of this study. The goal is to assess, based on real outcomes, not just offline metrics, whether the models presented represent the most viable future approach for managing these transactions and enhancing the overall decision process.

As such, with further testing and given the consistent outperformance of the LightGBM model across all client segments, future optimization efforts could focus exclusively on this algorithm. This would allow for more exhaustive fine-tuning and potentially even better results.

Model explainability remains a keystone for regulatory compliance and stakeholder trust. While SHAP was effectively used in this study, incorporating additional interpretability techniques could provide further clarity regarding feature influence and decision logic.

In parallel, the Shadow Limit concept introduced in this project offered an additional layer of optimization. Although it was applied only to Individual clients, due to their lower transaction amounts and more stable behavior, it presents promising potential for future extension. For business clients, where transactions are significantly larger and more volatile, applying a fixed percentile threshold is impractical. Therefore, in such cases, the development of regression models could enable a dynamic definition of each specific Shadow Limit.

Together, these improvements form a roadmap for building a more robust, transparent and aligned credit decision system.

In summary, this project demonstrates that the application of ML can significantly enhance credit decision-making systems. The methodological and practical insights offered here form a strong foundation for the implementation of interpretable solutions in the financial sector. As

the industry continues to evolve toward automation, initiatives like this underscore the crucial role of data science (and of course ML) in shaping the future of risk management.

Finally, on a personal note, this internship was not only a professional challenge, it was a truly enriching experience. It pushed me to grow in ways I did not expect, as it contributed significantly to my personal and professional growth. I got to sharpen my ML skills, tackle real-world problems, and see how all the theory learned in class actually plays out on environments like the financial sector. It also gave me a clearer sense of what kind of work excites me and confirmed that data science is definitely the path I want to follow. Overall, it was an incredible journey that I will never forget.

References

- [1] Glyn Davies. *A History of Money: From Ancient Times to the Present Day*. University of Wales Press, 4 edition, 2010.
- [2] Charles P. Kindleberger. *A Financial History of Western Europe*. Oxford University Press, 1993.
- [3] Abdalla Mahgoub. Optimizing bank loan approval with binary classification method and deep learning model. *Open Journal of Business and Management*, 12(3):1999–2017, May 2024.
- [4] P. Maheswari and CH. V. Narayana. Predictions of loan defaulter - a data science perspective. In *2020 5th International Conference on Computing, Communication and Security (ICCCS)*, pages 1–4, 2020.
- [5] Tian-Shyug Lee, Chih-Chou Chiu, Chi-Jie Lu, and I-Fei Chen. Credit scoring using the hybrid neural discriminant technique. *Expert Systems with Applications*, 23(3):245–254, 2002.
- [6] Xiang Li and Yuanchun Zhong. An overview of personal credit scoring: Techniques and future work. *International Journal of Intelligence Science*, 2(4A):1–6, 2012.
- [7] Ana I Marqués, Vicente García, and José Salvador Sánchez. Exploring the behaviour of base classifiers in credit scoring ensembles. *Expert systems with Applications*, 39(11):10244–10250, 2012.
- [8] J. Ross Quinlan. Learning decision tree classifiers. *ACM Computing Surveys (CSUR)*, 28(1):71–72, 1996.
- [9] Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.
- [10] J. H. Friedman. Greedy function approximation: A gradient boosting machine. *The Annals of Statistics 2001*, 2001.
- [11] Yiheng Li and Weidong Chen. A comparative performance assessment of ensemble learning for credit scoring. *Mathematics*, 8(10), 2020.
- [12] Sofia Beatriz Santos Simão. Machine learning applied to credit risk assessment: Prediction of loan defaults. Master’s thesis, NOVA Information Management School (NOVA IMS),

January 2023. Dissertation presented as the partial requirement for obtaining a Master's degree in Data Science and Advanced Analytics, specialization in Data Science.

- [13] Rohan Pothumsetty. Implementation of artificial intelligence and machine learning in financial services. *International Research Journal of Engineering and Technology (IRJET)*, 7(3):3186–3190, 2020.
- [14] Mengran Zhu, Ye Zhang, Yulu Gong, Kaijuan Xing, Xu Yan, and Jintong Song. Ensemble methodology: innovations in credit default prediction using lightgbm, xgboost, and localensemble, 2024.
- [15] Chang Yu, Yixin Jin, Qianwen Xing, Ye Zhang, Shaobo Guo, and Shuchen Meng. Advanced user credit risk prediction model using lightgbm, xgboost and tabnet with smoteenn. In *2024 IEEE 6th International Conference on Power, Intelligent Computing and Systems (ICPICS)*, pages 876–883, 2024.
- [16] Adnan Alagic, Natasa Zivic, Esad Kadusic, Dzenan Hamzic, Narcisa Hadzajlic, Mejra Dizdarevic, and Elmedin Selmanovic. Machine learning for an enhanced credit risk analysis: A comparative study of loan approval prediction models integrating mental health data. *Machine Learning and Knowledge Extraction*, 6(1):53–77, 2024.
- [17] Nikita Kozodoi, Stefan Lessmann, Morteza Alamgir, Luis Moreira-Matias, and Konstantinos Papakonstantinou. Fighting sampling bias: A framework for training and evaluating credit scoring models. *European Journal of Operational Research*, 324(2):616–628, 2025.
- [18] Jean Marie Vianney Ntamwiza and Hannibal Bwire. Predicting biking preferences in kigali city: A comparative study of traditional statistical models and ensemble machine learning models. *Transport Economics and Management*, 3:78–91, 2025.
- [19] Joaquim P. Costa. A tripartite scorecard for the pay/no pay decision-making in the retail banking industry. In *Frontiers in Artificial Intelligence and Applications*, page n. pag. IOS Press, 2008. Print.
- [20] Hélia Sofia da Rocha Monteiro da Costa. Estudo comparativo de abordagens ao problema de débito de transações bancárias em contas com saldo insuficiente no segmento de negócios. Dissertação de mestrado, Faculdade de Ciências da Universidade do Porto, Porto, Portugal, 2012. Accessed on 2025-06-03. Licensed under CC BY-NC 4.0.
- [21] Martin Rezac. Determining the target variable in credit scoring models. *The GSTF Journal of Mathematics, Statistics and Operations Research*, 2, 05 2013.

-
- [22] Aslesha Mohanty. Artificial intelligence in credit scoring: Enhancing financial inclusion opportunities. *International Research Journal of Engineering and Technology (IRJET)*, 04 2025.
 - [23] Giovani Valdrighi, Athyrson M. Ribeiro, Jansen S. B. Pereira, Vitoria Guardieiro, Arthur Hendricks, Décio Miranda Filho, Juan David Nieto Garcia, Felipe F. Bocca, Thalita B. Veronese, Lucas Wanner, and Marcos Medeiros Raimundo. Best practices for responsible machine learning in credit scoring, 2024.
 - [24] Pedro Silva and Jaime Cardoso. Differential scorecards for binary and ordinal data. *Intelligent Data Analysis*, 19:1391–1408, 11 2015.
 - [25] Chelsey Hill, Lanqing Du, Marina Johnson, and B. McCullough. Comparing programming languages for data analytics: Accuracy of estimation in python and r. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 14, 02 2024.
 - [26] Adnan Alagic, Natasa Zivic, Esad Kadusic, Dzenan Hamzic, Narcisa Hadzajlic, Mejra Dizdarevic, and Elmedin Selmanovic. Machine learning for an enhanced credit risk analysis: A comparative study of loan approval prediction models integrating mental health data. *Machine Learning and Knowledge Extraction*, 6(1):53–77, 2024.
 - [27] GeeksforGeeks. Decision tree introduction with example, 2021. Accessed: 2025-06-20.
 - [28] Pang-Ning Tan, Michael Steinbach, Vipin Kumar, and Michael Adeyeye Oshin. *Introduction to Data Mining*. Addison-Wesley, 05 2005.
 - [29] Mingsheng Liu and Ling Peng. Predicting entrepreneurial decisions using artificial intelligence within the digital economy context: A cart algorithm. In *Proceedings of the 2024 International Conference on Digital Society and Artificial Intelligence*, DSAI '24, page 429–433, New York, NY, USA, 2024. Association for Computing Machinery.
 - [30] IBM. Decision trees: What they are and how they work, 2024. Accessed: 2025-05-21.
 - [31] Kristína Machová, Miroslav Puszta, František Barčák, and Peter Bednár. A comparison of the bagging and the boosting methods using the decision trees classifiers. *Computer Science and Information Systems*, 3(2):57–72, 2006.
 - [32] Zhi-Hua Zhou. *Ensemble Methods: Foundations and Algorithms*, volume 14. CRC Press, 06 2012.

-
- [33] Yiheng Li and Weidong Chen. A comparative performance assessment of ensemble learning for credit scoring. *Mathematics*, 8:1756, 10 2020.
 - [34] Pelin Yildirim Taser. Application of bagging and boosting approaches using decision tree-based algorithms in diabetes risk prediction. In *Proceedings*, volume 74, page 6. MDPI, 2021.
 - [35] Inside Learning Machines. Implement the bootstrap method in python from scratch, 2021. Accessed: 2025-05-28.
 - [36] Javeria Sarwar, Saud Ahmed Khan, Muhammad Azmat, and Faridoon Khan. An application of hybrid bagging-boosting decision trees ensemble model for riverine flood susceptibility mapping and regional risk delineation. *Water Resources Management*, 39(2):547–577, 2025.
 - [37] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, page 785–794. ACM, August 2016.
 - [38] Jerome H. Friedman. Stochastic gradient boosting. *Computational Statistics Data Analysis*, 38(4):367–378, 2002. Nonlinear Methods and Data Mining.
 - [39] Adele Cutler, D Richard Cutler, and John R Stevens. Random forests. *Ensemble machine learning: Methods and applications*, pages 157–175, 2012.
 - [40] Fernanda Sampaio. Random forest. Medium, May 2024. Accessed: 2025-07-02.
 - [41] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.
 - [42] Bo Gao and Vipin Balyan. Construction of a financial default risk prediction model based on the lightgbm algorithm. *Journal of Intelligent Systems*, 31(1):767–779, 2022.
 - [43] Fangrong Zhou, Hao Pan, Zhenyu Gao, Xuyong Huang, Guochao Qian, Yu Zhu, and Feng Xiao. Fire prediction based on catboost algorithm. *Mathematical Problems in Engineering*, 2021(1):1929137, 2021.

-
- [44] Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. Catboost: unbiased boosting with categorical features. *Advances in Neural Information Processing Systems*, 31, 2018.
- [45] Yezi Zhu. Learn bayesian optimisation in 5 mins. <https://medium.com/@yezi.zhu/learn-bayesian-optimisation-in-5-mins-3315ce4a4e17>, 2020. Accessed: 2025-05-21.
- [46] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework, 2019.
- [47] James Bergstra, R. Bardenet, Balázs Kégl, and Y. Bengio. Algorithms for hyper-parameter optimization. In *Algorithms for Hyper-Parameter Optimization*, 2011.
- [48] Daniel Berrar et al. Cross-validation., 2019.
- [49] Marco Peixeiro. How to cross-validation with time series data, 2020. Accessed: 2025-05-21.
- [50] Azine Said. Confusion matrix as performance measure for corner detectors. *Journal of Advanced Research in Applied Sciences and Engineering Technology*, 29(1):256–265, 2022.
- [51] Francisco Melo. *Area under the ROC Curve*, pages 38–39. Springer New York, New York, NY, 2013.
- [52] GeeksforGeeks. Auc - roc curve in machine learning. <https://www.geeksforgeeks.org/auc-roc-curve/>, 2021. Accessed: 2025-05-21.
- [53] Robert I Hamilton and Panagiotis N Papadopoulos. Using shap values and machine learning to understand trends in the transient stability limit. *IEEE Transactions on Power Systems*, 39(1):1384–1397, 2023.
- [54] Scott Lundberg and Su-In Lee. A unified approach to interpreting model predictions, 2017.
- [55] Eyal Winter. The shapley value. *Handbook of game theory with economic applications*, 3:2025–2054, 2002.
- [56] Hendrik Andries du Toit, Willem Daniël Schutte, and Helgard Raubenheimer. Shapley values as an interpretability technique in credit scoring. *Journal of Risk Model Validation*, 2024.

A Confusion Matrix for Individuals

		Predicted Category	
		0	1
Real Category	0	41.22%	9.77%
	1	8.38%	40.63%

Table A.1: Decision Tree Confusion Matrix for Individuals

		Predicted Category	
		0	1
Real Category	0	40.27%	10.42%
	1	3.61%	45.16%

Table A.2: Random Forest Confusion Matrix for Individuals

		Predicted Category	
		0	1
Real Category	0	41.32%	9.66%
	1	4.89%	44.13%

Table A.3: XGBoost Confusion Matrix for Individuals

		Predicted Category	
		0	1
Real Category	0	44.22%	6.76%
	1	3.93%	45.09%

Table A.4: CatBoost Confusion Matrix for Individuals

B Confusion Matrix for Entrepreneurs

		Predicted Category	
		0	1
Real Category	0	45.03%	7.05%
	1	8.99%	38.93%

Table B.1: Decision Tree Confusion Matrix for Entrepreneurs

		Predicted Category	
		0	1
Real Category	0	42.70%	9.37%
	1	2.95%	44.98%

Table B.2: Random Forest Confusion Matrix for Entrepreneurs

		Predicted Category	
		0	1
Real Category	0	44.86%	7.22%
	1	3.79%	44.13%

Table B.3: XGBoost Confusion Matrix for Entrepreneurs

		Predicted Category	
		0	1
Real Category	0	46.35%	5.71%
	1	2.83%	45.10%

Table B.4: CatBoost Confusion Matrix for Entrepreneurs

C Confusion Matrix for Companies

		Predicted Category	
		0	1
Real Category	0	46.63%	6.63%
	1	2.79%	43.95%

Table C.1: Decision Tree Confusion Matrix for Companies

		Predicted Category	
		0	1
Real Category	0	46.96%	6.30%
	1	1.55%	45.18%

Table C.2: Random Forest Confusion Matrix for Companies

		Predicted Category	
		0	1
Real Category	0	49.05%	4.22%
	1	1.64%	45.09%

Table C.3: XGBoost Confusion Matrix for Companies

		Predicted Category	
		0	1
Real Category	0	49.47%	3.78%
	1	1.60%	45.14%

Table C.4: CatBoost Confusion Matrix for Companies

D Beeswarm Plots for Entrepreneurs and Companies

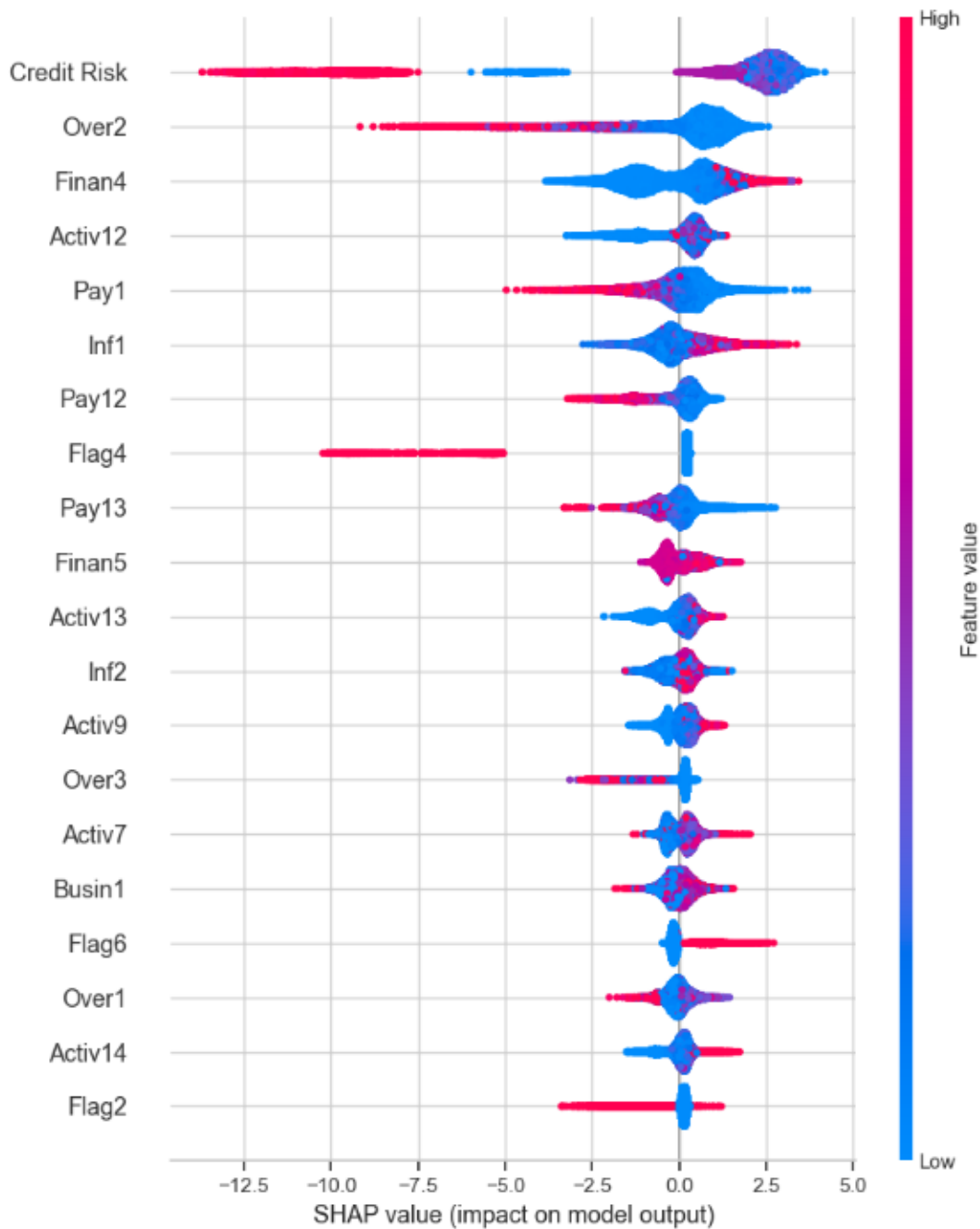


Figure D.1: Beeswarm Plot of LightGBM Entrepreneurs model

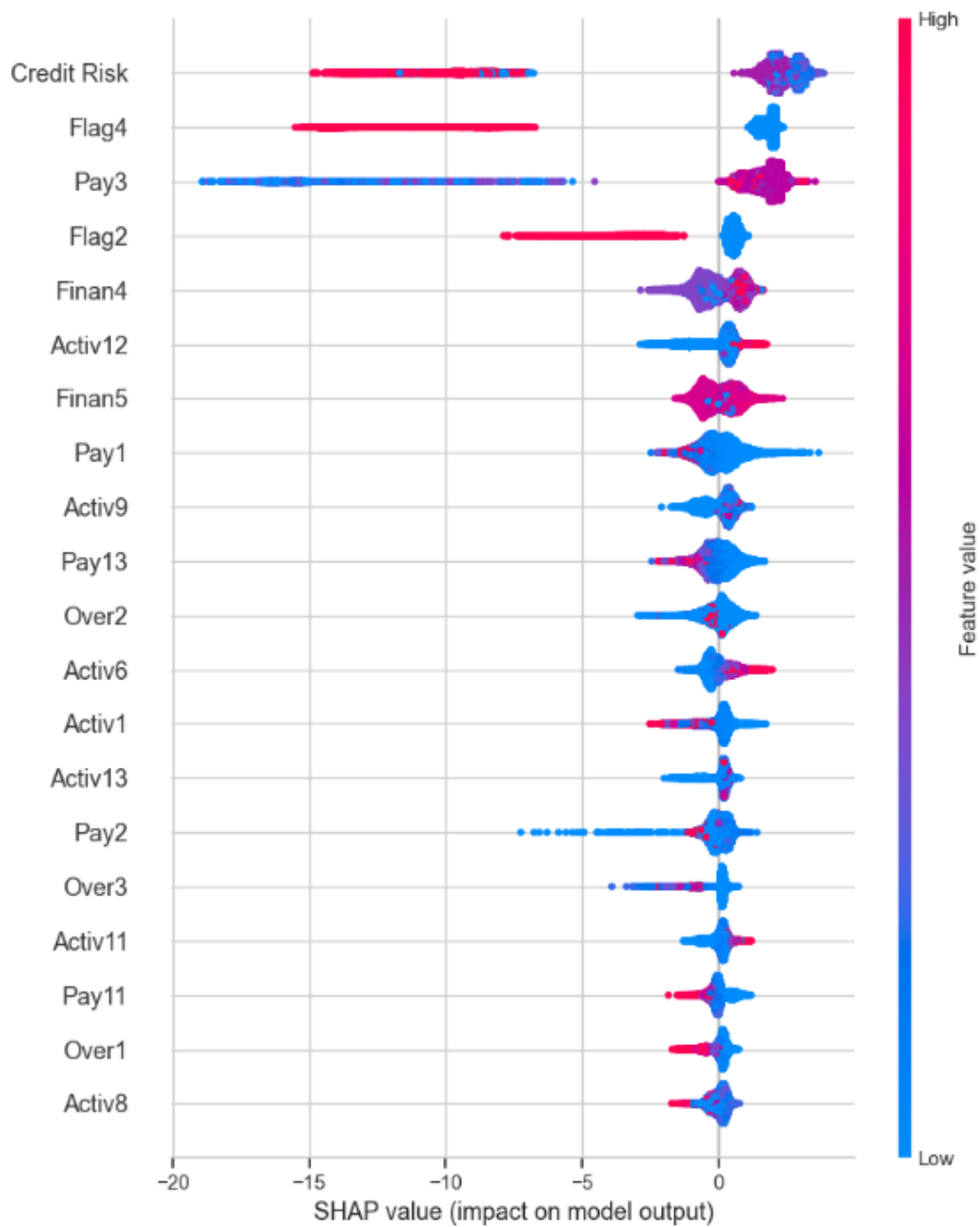


Figure D.2: Beeswarm Plot of LightGBM Companies model