Tunisian Republic
Ministry of Higher Education and Scientific Research
Carthage University - Engineering School of Statistics and Information Analysis

Graduation Project presented for the obtention of

National Engineering Diploma in Statistics and Information Analysis

Submitted by

Oussema Mzoughi

Modeling Loss Given Default (LGD) Using Machine Learning Techniques

---

Defended on 19/06/2024 in front of the committee composed of:

Mr Ghazi Belmufti	President
Ms Lilia Trabelsi Masmoudi	Examiner
Ms Tasnim Hamdeni	Academic Supervisor
M. Wassim Ben Dely	Company Supervisor
M. Bedis Blaiej	Company Supervisor

A Graduation Project made at

Dedication

To my dear parents, Zouhair Mzoughi and Aida Chebbi, and to my siblings, Yassine and Aya,

At this pivotal moment in my life, I wish to express my deepest gratitude to each of you. Your unwavering support and constant encouragement have been crucial to my academic and personal journey. To my beloved family, you have been my first mentors, my guides, and my role models. Your unconditional love, your sacrifices, and your relentless support have been the pillars of my success. Your ongoing encouragement and faith in me have given me the strength to face challenges and persevere in my studies.

To my wonderful professors, I am profoundly grateful for your expertise, your passion, and your dedication to education. Your teachings, wise counsel, and mentorship have played a fundamental role in my academic formation and development. Your support and encouragement have pushed me to excel and achieve my goals. This end-of-studies project is not only the culmination of my sustained efforts but also a testament to your guidance and support.

I dedicate this project to my family and my professors with deep gratitude and immense respect. To my parents and my professors, you are essential pillars in my life and my success. Your support, your advice, and your love are invaluable treasures. I am honored to have you by my side and thank you from the bottom of my heart for your positive impact on my path.

Thanks

I would like to express my deep appreciation to everyone who provided me with invaluable assistance and unwavering support throughout this project. Their generosity, good spirits, and keen interest in my end-of-studies project have been instrumental in my progress and success.

I am particularly grateful to Mr. Wassim Ben Daly and Mr. Bedis Blaiej for generously agreeing to supervise and guide my project. Despite their busy schedules and numerous responsibilities, they spared no effort in assisting me. I am thankful for their valuable insights, assistance, monitoring, and wise advice. I warmly extend my sincere thanks to the entire team at Quantylix, without exception, for their team spirit and consistent support. Their confidence in me has allowed me to thrive in my roles.

I also wish to express my sincere gratitude to Ms. Tasnim Hamdeni, assistant professor at the School of Statistics and Information Analysis (ESSAI), who supervised this end-of-studies project. I am indebted to her for her valuable guidance and relevant advice that marked my journey.

Further, I would like to thank Ms. Lilia Masmoudi Trabelsi and Mr. Ghazi Belmufti for agreeing to review this work and for the honor of their participation.

Lastly, I wish to acknowledge the entire faculty of ESSAI, who guided me throughout my three years in the engineering program.

Abstract

Ce rapport présente une étude complète sur la Perte en Cas de Défaut (PCD), une mesure cruciale dans la gestion du risque de crédit qui quantifie la perte attendue qu'un prêteur peut subir lorsque l'emprunteur fait défaut, exprimée en pourcentage de l'exposition totale au moment du défaut. Comprendre la PCD est essentiel car elle influence directement les exigences en matière de réserves de capital des institutions financières sous des cadres réglementaires tels que les Accords de Bâle et l'IFRS 9.

Ces dernières années, les méthodes traditionnelles de calcul de la PCD se sont révélées inadéquates en raison de leur capacité limitée à intégrer diverses variables comportementales et financières qui ont un impact significatif sur les récupérations après défaut. Cela a nécessité l'intégration de techniques avancées d'apprentissage automatique (ML) qui offrent une analyse plus nuancée en prenant en compte un éventail plus large de variables prédictives et leurs interactions. Notre étude utilise ces techniques de ML à travers deux approches : la modélisation des scénarios post-défaut où toutes les données disponibles, y compris les changements comportementaux après le défaut, sont utilisées, et les scénarios pré-défaut qui se concentrent exclusivement sur les données disponibles au moment de l'émission du prêt.

Les résultats révèlent des avancées significatives. Le modèle empilé, exploitant les données post-défaut, se distingue par sa précision exceptionnelle. Pour les scénarios pré-défaut, le modèle Random Forest se montre supérieur, indiquant des améliorations notables.

Mots clés— Perte en Cas de Défaut, intelligence artificielle, apprentissage automatique, modèle empilé, forêt aléatoire.

Abstract/Résumé

This report presents a comprehensive study on Loss Given Default (LGD), a crucial metric in credit risk management that quantifies the expected loss a lender may face when a borrower defaults, expressed as a percentage of the total exposure at the time of default. Understanding LGD is vital as it directly influences financial institutions' capital reserve requirements under regulatory frameworks like Basel Accords and IFRS 9.

In recent years, traditional methods of calculating LGD have proven inadequate due to their limited ability to incorporate varied behavioral and financial variables that significantly impact post-default recoveries. This has necessitated the integration of advanced machine learning (ML) techniques that offer a more nuanced analysis by considering a wider range of predictive variables and their interactions. Our study employs these ML techniques through two approaches: modeling post-default scenarios where all available data, including behavioral changes post-default, are used, and pre-default scenarios focusing exclusively on data available at the time of loan issuance. The results reveal significant advances. The stacked model, using post-default data, stands out for its exceptional accuracy. For pre-default scenarios, the Random Forest model is superior, indicating significant improvements

Keywords— Loss Given Default, artificial intelligence, machine learning, stacked model, random forest

Contents

List of Figures

List of Tables

Introduction

In today's regulatory environment, it is crucial for banks to model Loss Given Default (LGD) in alignment with standards such as Basel Accords and IFRS 9. Traditional methods often fall short in their ability to accurately predict LGD due to their limited consideration of diverse behavioral and financial variables. This gap has necessitated the integration of advanced machine learning (ML) techniques, which can provide more reliable predictions by analyzing a broader array of variables and their interactions.

This report explores two distinct approaches to LGD modeling: the post-default approach, which utilizes all available data including post-default behavioral changes, and the pre-default approach, which relies solely on data available at the time of loan issuance. The pre-default approach is particularly challenging due to the limited data available before any default occurs, making accurate predictions more difficult.

The report is structured into four main chapters. The first chapter introduces the host organization, the general framework of the project, and the basic concepts underlying the study.

The second chapter delves into the working mechanisms of each machine learning model employed in the project, the data preprocessing and engineering techniques used, and the performance measures that helped us select the most robust model.

The third chapter is dedicated to exploratory data analysis, detailing the steps of data preprocessing and cleaning, and presenting descriptive statistics and various methods of variable selection used.

Finally, the fourth chapter details the results obtained from the machine learning models and the deployment of these models into a production environment.

Chapter 1

General Project Overview

1.1 Introduction

This first chapter provides an overview of the general framework for the end-of-studies project. It begins by introducing the host organization, followed by an outline of the problem statement and the objectives we aim to achieve. Subsequently, it defines some basic concepts essential for understanding the project. Finally, it describes the methodology of the work.

1.2 The host organization Quantylix

1.2.1 General presentation

Quantylix is a RegTech (a branch of FinTechs) based in Paris, Tunis, Casablanca, and Abidjan. It specializes in quantitative risk management, data modeling, and the development of innovative technological tools in this field. Despite its young age (established in 2016), Quantylix has established itself as a reference in the risk management sector in the MENA region. It has supported over 20 banks from the Top100 African banks and has completed 63 missions in Europe and Africa. It currently has the largest team of Risk/Data engineers and consultants in the region. By the end of 2023, it had over 50 employees.

1.2.2 Areas of expertise

Quantylix's areas of expertise include:

- **Quantitative Risk:** IFRS9 modeling, ICAAP, Stress Testing, and Internal Rating System.
- **Risk Management:** Risk appetite framework, Risk management strategy and policy, operational risk, and Risk Control Self Assessment.
- **Data Management:** Data strategy and governance, Data quality improvement, and implementation of data lakes (big data).
- **Quantitative Finance:** Econometric modeling, Time series analysis, and Vector Autoregressive (VAR) modeling.

1.3 Project Presentation and Scope

1.3.1 Context

Since the adoption of the advanced internal rating-based approach under the Basel framework, financial institutions and regulators have been dealing with the increased complexities in constructing Loss Given Default (LGD) models. Moreover, the introduction of IFRS 9 in January 2018 has compounded these challenges, demanding a reassessment of model development and validation practices to adapt to the new standards.

However, within these challenges, the emergence of machine learning (ML) and sophisticated analytical techniques presents a promising avenue for revolutionizing LGD predictions. This is particularly significant as part of my end-of-studies internship at Quantylix, where I am actively engaged in exploring how these advanced methodologies can be applied to enhance LGD estimation, consequently empowering stakeholders with insights to enhance decision-making processes.

1.3.2 Objectives

The primary objective of this internship is to calculate LGD using traditional methodology while ensuring alignment with regulatory requirements and industry standards. Secondly, the study seeks to model LGD through two distinct approaches: post-default and pre-default. The post-default approach involves leveraging all available data, including financial and behavioral variables alongside post-default information, to provide a comprehensive understanding of borrower behavior following default events. In contrast, the pre-default approach focuses solely on variables known at contract origination, excluding post-default data. Lastly, the study seeks to integrate these LGD models into a production environment, enabling their practical application in financial risk management processes.

1.4 Overview of Basel Accords and IFRS 9

1.4.1 Basel Committee

The Basel Committee, formerly known as the Committee on Banking Regulations and Supervisory Practices, was established by the Group of Ten central bank governors at the end of 1974 in reaction to major disruptions in the global currency and banking markets, chief among them being the failure of the West German bank Bankhaus Herstatt. The Bank for International Settlements in Basel served as the Committee's headquarters when it was established. Its objectives are to raise the bar for international financial supervision standards and encourage regular communication among its member countries on banking supervisory matters. The Committee is primarily known for having issued the historic capital adequacy accords, or Basel I, Basel II, and most recently, Basel III, which established a variety of international guidelines for bank oversight.

1.4.1.1 Basel I

The development of Basel I, formerly known as the Basel Capital Accord, dates back to 1988. It was prompted by the growth of global banks and the growing interconnectivity of financial markets. Authorities in charge of regulations in different countries were concerned that these banks did not

have enough cash on hand. A multi-country financial crisis might perhaps be sparked by the failure of a big bank because of the close interconnectedness of global financial systems at the time.
By 1992, these measures were legally required in the G10 nations, and more than 100 more nations followed suit, albeit with minor adjustments. These regulations set minimum cash reserve requirements for foreign banks in an effort to improve the stability of the financial system.
Additionally, Basel I established a risk-weighting methodology for different asset classes, introducing a methodical approach to credit risk management. Assets were divided into four risk-weight groups

- 0% for risk-free assets (cash, treasury bonds)
- 20% for loans to other banks or securities with the highest credit rating
- 50% for residential mortgages
- 100% for corporate debt

It was mandated that banks with a large global footprint maintain cash buffers equal to 8% of their risk-weighted assets. International banks were advised to spend their cash in less risky ventures.

1.4.1.2 Basel II

The BCBS introduced the BASEL II Accord in 2004, which marked a significant advancement in global banking regulation.It was intended to improve the stability and soundness of the international financial system by offering an all-encompassing framework for risk management.
Three pillars were introduced: supervisory examination of capital adequacy, market discipline, and minimum capital requirements (basically the same as BASEL I). These pillars were designed to help banks enhance their risk management procedures, better evaluate their risks, and take more risk-sensitive approaches to capital allocation.

- **Pillar 1**

Pillar 1 improves Basel I's regulations by taking operational risks into account in addition to credit risks associated with risk-weighted assets (RWA). It requires banks to maintain a minimum capital

adequacy standard of 8% for their RWA.

Furthermore, Basel II provides banks with more advanced techniques to calculate capital requirements based on credit risk, taking into consideration the distinct characteristics and risk profile of various asset classes. The two main techniques are as follows:

- **Standardized approach** : Banks that operate at a lower volume and have a more straightforward control structure should choose the standardized technique. It entails determining a bank debtor's creditworthiness by utilizing credit ratings from outside credit assessment organizations.

- **Internal ratings-based approach** : Banks with more sophisticated risk management systems and more complicated activities should use the internal ratings-based method. When determining capital needs for credit risk based on internal ratings, the IRB uses one of two methods.

- **Foundation Internal Ratings-based approach (FIRB):** Banks under FIRB use their own evaluations of metrics such as the Probability of Default, while the supervisor sets the methods for evaluating other parameters, mainly risk components such as Loss Given Default and Exposure at Default.

- **Advanced Internal Ratings-based approach (AIRB):** When banks use the AIRB approach, they use their own evaluations for all risk components and other metrics.

- **Pillar 2**

Pillar 2 was added because there was a need for better supervision, which was missing in Basel I, especially in evaluating a bank's ability to manage its capital for risks. Under Pillar 2, banks are required to check their capital adequacy themselves, ensuring they can cover all possible risks in their operations. Supervisors need to make sure that banks are using correct methods for these assessments and covering all necessary risks.

- **Internal Capital Adequacy Assessment Process (ICAAP):** Banks must develop plans to maintain the required capital levels and evaluate their capital sufficiency on a regular basis based on their risk profile.

- **Supervisory Review and Evaluation Process (SREP):** Supervisors are required to examine and assess the methods used by banks to determine and maintain their level of capital adequacy and compliance with capital regulations.

- **Capital above the minimum level:** A key part of the Basel II framework is making sure that supervisors enforce banks to keep their capital above the minimum level set by Pillar 1.

- **Supervisor's interventions:** Supervisors must actively intervene in banks' decision-making processes to prevent their capital from dropping below the required minimum.

- **Pillar 3**

By mandating the disclosure of pertinent market information, Pillar 3 focuses on maintaining market discipline. By doing this, financial information users are guaranteed to have the tools necessary to preserve market discipline and make well-informed judgments.

1.4.1.3 Basel III

The Basel Committee on Banking Supervision (BCBS) developed the Basel III accord, a set of financial reforms, to improve banking industry regulation, supervision, and risk management. It was created in reaction to the global financial crisis of 2008 with the intention of strengthening banks' ability to withstand financial strain and raising their level of responsibility and transparency. Basel III is a component of the continuous endeavor to enhance banking regulation, building upon the foundation established by Basel I and II. The primary goal is to stop banks from taking on risks that could be detrimental to the economy.

- **Minimum Capital Requirements**

The minimum capital requirements for banks under Basel III grew from 2% of common equity as a percentage of the bank's risk-weighted assets to 4.5% under Basel II. Moreover, a buffer capital requirement of 2.5% is required, raising the overall minimum to 7%. In times of financial strain, this buffer gives banks some flexibility, but using it could limit their capacity to pay dividends.
Basel III has increased the Tier 1 capital requirement from 4% in Basel II to 6% in Basel III. This additional 1.5% of Tier 1 capital is on top of the 4.5% of Common Equity Tier 1. These regulations went into effect in 2013, and banks have until January 1, 2022, to completely respect by them.

- **Leverage Ratio** :

In addition to risk-based capital requirements, Basel III included a non-risk-based leverage ratio as a safety measure. Leverage ratios for banks must remain at least 3%. By dividing Tier 1 capital by the total consolidated assets of a bank, this ratio is computed.
The leverage ratio for insured bank holding companies and systematically important financial institutions (SIFI) in the US was set by the Federal Reserve at 5% and 6%, respectively.

- **Liquidity Requirements**

Basel III introduced two essential liquidity measures, namely the Net Stable Funding Ratio and the Liquidity Coverage Ratio. In order to comply with the Liquidity Coverage Ratio, banks must maintain sufficient liquid assets to cover any withdrawals during a 30-day financial stress scenario. The ratio was introduced in 2015 and will increase by 10% a year until it is fully implemented in 2019. In order to avoid liquidity mismatches, banks must maintain stable funding sources that exceed their needed stable funding for a year during stress conditions under the Net Stable Funding Ratio (NSFR), which will be put into effect beginning in 2018.

1.4.1.4 Basel IV

Basel IV is essentially the completion of the Basel III reforms. It took over ten years to develop and consists of two main parts. The Basel Committee finalized these reforms in December 2017. Basel IV introduces new rules for handling operational risk, credit risk, and the credit valuation adjustment.

The reform also introduced a cap on model outputs, revised the leverage ratio definition, and applied it to globally significant banks. The initial phase of these rules was set to start on January 1st, 2022. The aim of these changes is to build more resilient banking systems and enhance financial stability by making risk measurement and capital requirements more precise and consistent, especially through stricter loan loss provisions.

Originally, Basel IV was scheduled to start on January 1st, 2022, with gradual implementation until January 1st, 2027. However, due to the pandemic, the Basel Committee on Banking Supervision delayed the start date by a year to January 1st, 2023.

1.4.2 IFRS 9 standard

IFRS 9 is an international financial reporting standard that deals with how financial instruments are accounted for. It sets the rules for classifying, measuring, and recognizing financial assets and liabilities.

Under IFRS 9, there is a strong focus on using a forward-looking approach to estimate credit losses. This requires institutions to consider expected credit losses over the lifetime of financial assets, even if there's no clear evidence of significant credit deterioration yet. As a result, LGD models have to include long-term credit risk scenarios. The standard also makes LGD modeling more complex, as financial institutions need detailed historical data on defaults, recoveries, and collateral values.

1.4.3 Risk parameters

- **Probability of Default (PD)**

Default probability, also known as the probability of default (PD), is the chance that a borrower will fail to make scheduled payments on a debt within a specific time frame, typically one year. This measure is used in various scenarios related to risk management and credit analysis. The PD not only reflects the borrower's specific characteristics but is also influenced by the broader economic conditions.

- **Loss Given Default (LGD)**

LGD as defined by the Basel Committee, refers to the percentage of an exposure that a bank or financial institution is expected to lose in the event that a borrower defaults on their obligations. It accounts for the severity of losses incurred after default, considering factors such as collateral value, recoveries, and other mitigating circumstances.

- **Exposure at Default (EAD)**

Exposure at Default represents the maximum amount that a financial institution might lose if a borrower fails to meet their debt obligations.

- **Expected Credit Loss (ECL)**

ECL represents the estimated average loss that a financial institution expects to incur over a specified period due to default events on its financial assets. The formula for Expected Credit Loss can be expressed as:

$$\text{ECL} = \text{PD} \times \text{LGD} \times \text{EAD} \tag{1.1}$$

1.4.4 Artificial Intelligence (AI)

Artificial Intelligence (AI) involves creating machines designed to mimic human intelligence and learning capabilities. The primary goal of AI is to build systems capable of performing activities traditionally requiring human intellect, including visual perception, speech recognition, decision-making, and language translation. AI encompasses a diverse range of technologies and methods, including machine learning (ML), natural language processing (NLP), robotics, and expert systems.

1.4.4.1 Machine Learning (ML)

Machine Learning is a branch of AI focused on creating systems that learn from data and improve autonomously over time without explicit programming. ML models use historical data to make predictions about new data. It is used in diverse fields such as consumer behavior prediction, business process optimization, and healthcare diagnostics.

- **Supervised Learning:**

In supervised learning, models are trained using a labeled dataset where each example includes an input vector and a corresponding output label. The model is trained to associate inputs with corresponding outputs and is capable of making predictions on new, unfamiliar data. Examples of applications are spam detection and image recognition.

- **Unsupervised Learning:**

Unsupervised learning involves training models using datasets without pre-defined labels.The objective is to identify the inherent patterns within the data. Common methods include clustering and association. This type of learning is useful for exploratory data analysis, customer segmentation, and market basket analysis.

- **Semi-Supervised Learning:**

Semi-supervised learning uses both labeled and unlabeled data, combining the advantages of supervised and unsupervised learning. When labeled data are hard to come by or expensive to acquire, this method performs well.

- **Reinforcement Learning:**

Through the use of actions and results observation, an agent is taught how to make decisions through reinforcement learning. By mastering activities that maximize a reward, it is used to accomplish certain goals in challenging and uncertain circumstances.

1.4.4.2 Deep Learning (DL)

Deep learning, a subset of machine learning, employs multi-layer neural networks to emulate the complex decision-making abilities of the human brain. It utilizes deep neural networks to recognize patterns and interpret data through machine perception, labeling, and clustering. The network architecture consists of multiple layers of nodes, with each layer processing inputs from previous layers and sending outputs to subsequent layers, similar to the neural processing in the human brain.

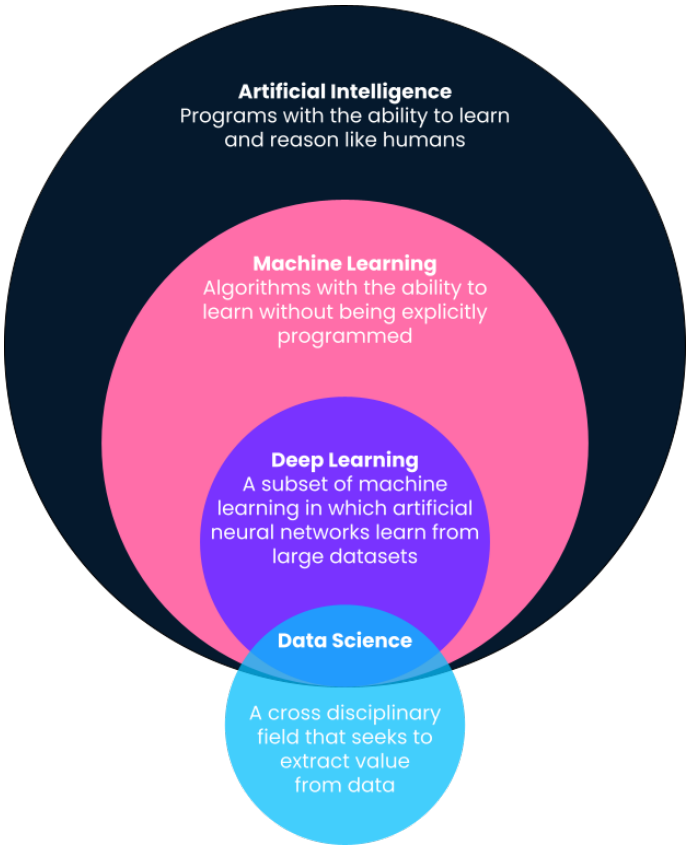

Figure 1.1: Interconnected Hierarchy of Data-Driven Technologies

- **Input Layer:** This is the initial layer in the neural network. It receives input data and forwards it to the next layer (hidden layer) for further processing. Each node in the input layer represents a single feature in the dataset.

- **Hidden Layers:** These are layers between the input and output layers.They execute calculations and relay data from the input layer to the output layer. A neural network can have multiple hidden layers, and each layer can have multiple nodes (neurons). The complexity of the model increases with the number of hidden layers and nodes.

- **Output Layer:** This is the final layer in the neural network. It receives data from the last hidden layer and transforms it into a format suitable for the model's purpose, such as a prediction for regression or classification tasks. The number of nodes in the output layer corresponds to the number of possible output values.

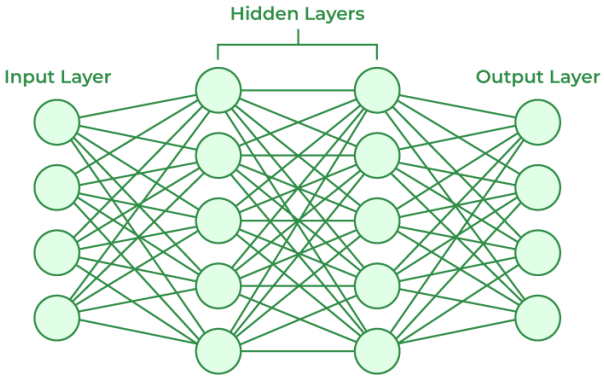

Figure 1.2: Neural Networks Architecture

1.5 Software Environment

We used Jupyter Notebook, an open-source web program, to create and share documents with python codes, equations, visualizations, and narrative text for this assignment. This versatile tool was instrumental in developing the initial stages of our code, experimenting with various models, and performing data analysis.

For the modeling and deployment phases, we employed Python, a powerful and flexible programming language known for its ease of use and wide range of libraries. It includes libraries such as NumPy for numerical data, pandas for data manipulation, and scikit-learn for machine learning, which were essential in building and testing our models.

A key component of this project was deploying the loss given default models using Flask, a lightweight web application framework written in Python. Flask provides tools, libraries, and technologies that enables to build a web server with minimal setup.

We used PyCaret which is an open-source, low-code machine learning library in Python that automates machine learning workflows. It is an end-to-end machine learning and model management tool that simplifies the process of preparing data, selecting models, tuning hyper parameters, and evaluating performance.

1.6 Work Methodology

I this project we employed CRISP-DM methodology, which stands for Cross-Industry Standard Process for Data Mining, is a widely adopted methodology used to guide data mining projects. This structured approach consists of six phases: Business Understanding, Data Understanding, Data Preparation, Modeling, Evaluation, and Deployment. The process begins by establishing clear business objectives, followed by acquiring and understanding the necessary data, preparing it for analysis, and then developing and evaluating models to meet the objectives. The final phase involves deploying the solution into the business environment. CRISP-DM is iterative, allowing for continuous improvements based on feedback and insights gained during the process.

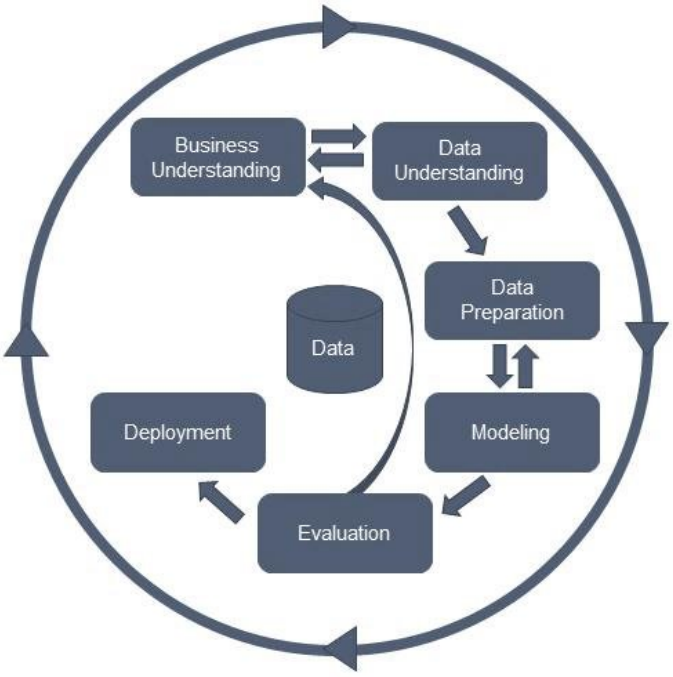

Figure 1.3: CRISP-DM methodology

1.7 Conclusion

During this first chapter, we presented the basic concepts of machine learning and deep learning, as well as the IFRS 9 standard and Basel regulations. We outlined the problem statement and the framework of this project, as well as the objectives we aim to achieve.

Chapter 2

State Of The Art

2.1 Introduction

In this chapter, we present the method for calculating LGD. Next, we discuss variable selection techniques, encoding methods, and the working mechanism of each machine learning model that we will use in the modeling phase. We also introduce the metrics and performance measures that we have adopted to evaluate our models.

2.2 Loss Given Default Calculation

Before embarking on the modeling process, it's crucial to compute the Loss Given Default (LGD). The LGD workout methodology is based on the calculation of the net present value of recovery cash flows, adjusted for associated costs.

The necessary data for LGD workout modeling include:

- Payments : Amount recovered after default
- Recovery costs
- Additional recoveries after default occurrence.

The formula for LGD calculation can be expressed as :

$$\text{LGD} = 1 - \text{RR} \tag{2.1}$$

$$\text{RR} = \frac{\sum_i \text{NPV}(CF_i, t_i; r_i)}{\text{EAD}} - \frac{\sum_i \text{NPV}(CFE_i, t_i; r_i)}{\text{EAD}} \tag{2.2}$$

While the Recovery Rate (RR) is determined by the ratio of the present value of expected cash flows (CF_i) from a loan portfolio to the Exposure at Default (EAD), adjusted for the time value of money. This equation accounts for both the cash inflows (CF_i) and external fees (CFE_i) discounted to their present value using the discount rate (r_i) and time since default (t_i).

2.2.1 LGD parameters

- **Related Defaults**

Two defaults are considered related when the same counterparty defaults once, exits the default status for some time, and then falls into default a second time. If the time period separating the two defaults is quite short, the two defaults are merged and considered as a single default.

In accordance with Article 101 of EBA/GL/2017/16, a related default is defined as follows: 'When the period between the return of the exposure to a non-default status and the subsequent classification as a default exposure is less than nine months.' Therefore, a contract can generate multiple default periods if it alternates between default and recovery over time.

All default periods of the same contract separated by a period of nine months or less are grouped together to form a single default

The new defaults reconstituted from linked defaults assume the initial default date of the first linked default and the final resolution date of the last linked default. The period of recovery to a performing status between these two linked defaults is included in the newly consolidated default period.

- **Discount Rate**

The discount rate is used to convert future cash flows, such as recoveries from a defaulted loan, into their present value. It reflects the risk and the time value of money and determines the current worth of cash expected to be received in the future.
In accordance with Article 143 of EBA/GL/2017/16, discount rates must reflect the average economic conditions suitable for LGD calculations, distinct from downturn conditions. The discount rate for LGD may be based on average rates that reflect normal economic conditions.

$$\text{Discount rate} = \text{Average Interbank Mean Bid Rate} + \text{add-on} \tag{2.3}$$

While add-on is the additional margin to reflect the economic situation.

- **Time To Workout**

In the context of the EBA/GL/2017/16 guidelines, "Time to Workout" is defined as the maximum period during which a bank's recovery process should be completed. Articles 155 and 156 specify that this timeframe should be clearly defined to ensure there are sufficient data to estimate recoveries during this period for ongoing recovery processes.

- **Internal and External Fees**

There are two types of costs, defined as follows:
Direct costs (external): Direct costs should include the costs of outsourced recovery services, legal fees, the cost of hedges or insurance, and any other cost directly attributable to the recovery of a particular exposure.
Indirect costs (internal): Indirect costs should include all costs arising from the implementation of the institution's recovery procedures, the overall cost of outsourced recovery services not included in direct costs, and all other costs related to the recovery of exposures that have defaulted and cannot be directly attributed to recovery on a particular exposure.

According to IFRS 9 standards, only direct recovery costs (external) are included in the calculation of the LGD.

2.3 Feature Selection

Feature selection, a crucial aspect of feature engineering, involves choosing the most significant features for use in machine learning algorithms. This process boosts the accuracy of the model by removing features that are irrelevant or only somewhat relevant. It also minimizes overfitting through model simplification, speeds up the training process by lowering computational demands, and enhances the model's interpretability, making it simpler to comprehend.

2.3.1 Spearman Correlation

The degree and direction of a monotonic link between two variables are measured by the Spearman's correlation coefficient (ρ). Spearman correlation is based on the ranks of the data rather than the actual values, in contrast to Pearson correlation, which assumes a linear relationship between variables. It is calculated as:

$$\rho = 1 - \frac{6 \sum d_i^2}{n(n^2 - 1)} \tag{2.4}$$

where d_i is the difference between the ranks of corresponding values in the two variables, and n is the number of data points.

2.3.2 Cramer V

Cramer's V is a statistic used to quantify the relationship between two categorical variables, producing a value that ranges from 0 to 1. This measure derives from Pearson's Chi-square statistic and serves as an indicator of both the strength and the direction of the association between the variables.

- **Chi-squared Statistic**

Consider a sample of size n consisting of variables A and B distributed across $i = 1, \ldots, r$ rows and $j = 1, \ldots, k$ columns. The frequencies are denoted as n_{ij}, representing the number of occurrences of the pair (A_i, B_j). The chi-squared statistic is computed as follows:

$$\chi^2 = \sum_{i,j} \left(\frac{(n_{ij} - \frac{n_{i.} n_{.j}}{n})^2}{\frac{n_{i.} n_{.j}}{n}} \right) \tag{2.5}$$

where $n_{i.} = \sum_j n_{ij}$ and $n_{.j} = \sum_i n_{ij}$, representing the totals for each row i and column j, respectively.

- **Cramer's V Calculation**

Cramer's V is calculated by taking the square root of the ratio of the chi-squared statistic to the product of the sample size n and the smaller of $(k-1)$ or $(r-1)$:

$$V = \sqrt{\frac{\chi^2/n}{\min(k-1, r-1)}} = \sqrt{\frac{\phi^2}{\min(k-1, r-1)}} \tag{2.6}$$

where:

- ϕ is the phi coefficient.
- χ^2 is based on Pearson's chi-squared test.
- n is the total number of observations.
- k is the number of columns.
- r is the number of rows.

2.3.3 ANOVA Test

ANOVA, or Analysis of Variance, is a statistical method used to test the differences between the means of three or more independent (unrelated) groups. The purpose of ANOVA is to investigate whether the means of several groups are equal or not. To perform this test, we first need to calculate various variances, which are presented as follows :

- **Total Sum of Squares (SST)**

This measures the total variability in the dependent variable:

$$SST = \sum_{i=1}^{n} (Y_i - \overline{Y})^2$$

where Y_i is each individual observation, $\overline{Y}$ is the overall mean of all observations, and n is the total number of observations.

- **Between-Group Sum of Squares (SSB)**

This captures the variability due to the differences between the means of the groups:

$$SSB = \sum_{j=1}^{k} n_j (\overline{Y}_j - \overline{Y})^2$$

where $\overline{Y}_j$ is the mean of group j, n_j is the number of observations in group j, and k is the total number of groups.

- **Within-Group Sum of Squares (SSW)**

This measures the variability within each group:

$$SSW = \sum_{j=1}^{k} \sum_{i=1}^{n_j} (Y_{ij} - \overline{Y}_j)^2$$

where Y_{ij} are the observations within group j.

- **ANOVA Table and F-test**

The mean squares are calculated by dividing the sums of squares by their respective degrees of freedom:

- Degrees of Freedom Between Groups (dfB): $k - 1$

- Degrees of Freedom Within Groups (dfW): $n - k$
- Mean Square Between Groups (MSB): $\frac{SSB}{dfB}$
- Mean Square Within Groups (MSW): $\frac{SSW}{dfW}$

The F-statistic is calculated as:

$$F = \frac{MSB}{MSW}$$

- **Hypothesis Testing**
- Null Hypothesis (H_0): The means of the different groups are equal ($\mu_1 = \mu_2 = \cdots = \mu_k$).
- Alternative Hypothesis (H_A): At least one group mean is different.

If the calculated F-statistic is greater than the critical value from the F-distribution, the null hypothesis is rejected, indicating significant differences among the group means.

2.3.4 SelectKBest

SelectKBest is a feature selection method designed to identify the top K features that are most predictive of a response variable in regression tasks. It uses a scoring function to evaluate each feature's relevance; only the highest scoring K features are retained while the others are discarded. This approach enhances the efficiency and performance of regression models by reducing data dimensionality, which can help minimize overfitting and enhance the interpretability of the model.

Mutual Information (MI) Regression is commonly utilized as a scoring mechanism in SelectKBest for regression tasks, assessing the extent of information one random variable, typically a feature X, imparts about another, usually the target Y. This measurement effectively captures the decrease in uncertainty about Y with known values of X, and is particularly useful for identifying nonlinear relationships.

The mutual information between two continuous random variables X and Y is defined as:

$$I(X;Y) = \int_Y \int_X p(x,y) \log \frac{p(x,y)}{p(x)p(y)} \, dx \, dy$$

where:

- $p(x,y)$ is the joint probability density function of X and Y.
- $p(x)$ and $p(y)$ are the marginal probability density functions of X and Y, respectively.

2.3.5 Sequential Feature Selection

Sequential Feature Selection (SFS) is a methodology used to select the most relevant features from a dataset for specific predictive modeling tasks. It employs a type of greedy algorithm that iteratively adds or removes features to enhance the performance of a predictive model. SFS can operate in two modes: forward selection and backward selection.

How it works:

1. The process begins by initializing the selector with a predictive model, specifying the number of features to select, the scoring metric used for evaluation, and the tolerance for performance improvement.
2. The selector trains the predictive model using the entire set of features.
3. The model's performance is assessed on the training set using the chosen scoring metric.
4. Depending on the selection mode, the feature that most improves the model's cross-validation score is added to the set of selected features, or the feature that least reduces the score is removed. The selection is based on which option provides the most significant improvement in the scoring metric.
5. Steps 2 to 4 are repeated until the desired number of features is selected.

Forward Selection: This approach begins with no initial features and adds features iteratively, selecting at each step the one that most significantly enhances the model's performance.

Backward Selection: This approach initiates with all available features and methodically discards the least significant ones. This elimination process persists until the target number of features is obtained or until additional exclusions severely compromise the model's efficacy.
The number of features to be selected can be controlled using the `n_features_to_select` parameter. The `tol` parameter sets the tolerance for performance improvement, meaning the selector will only add or remove a feature if it enhances the scoring metric by at least the value specified by `tol`.

2.4 Treatment of categorical variables

The presence of categorical variables within datasets often adds a layer of complexity to the learning process. Indeed, most machine learning models require numerical data as input for effective analysis and prediction. Consequently, there arises the need to devise methodologies for transforming the categories of these variables into numerical values. This transformation is essential for enabling machine learning algorithms to process and extract meaningful patterns from categorical data, thus enhancing the overall performance and accuracy of predictive models.

2.4.1 One Hot Encoder

The one-hot encoding technique processes categorical data by converting it into a numerical format, which is essential for training models. This approach assigns a unique binary vector to each category, ensuring each category within a vector is represented by its own distinct binary value. Specifically, for each categorical variable, the one-hot encoder creates a binary feature for each unique category present in the variable. In these binary vectors, only one feature (or bit) is 'hot' (assigned the value 1) while the others are 'cold' (assigned the value 0), indicating the presence or absence of a particular category.

2.4.2 Label Encoder

a Label Encoder is a preprocessing technique used in machine learning to transform categorical data into numerical labels. It assigns a unique numerical identifier to each category in a categorical

variable. Label encoding is particularly useful for algorithms that require numerical input, such as decision trees and support vector machines.

2.5 Numerical Feature Scaling

Scaling numerical features is a preprocessing technique employed in machine learning to normalize or standardize the range of independent variables or features of a dataset. It involves transforming the numerical values of different features to a similar scale. This process ensures that each feature contributes equally to the model's learning process and optimization, preventing bias toward variables with higher magnitudes.

2.5.1 Robust Scaler

The Robust Scaler is a method used for scaling numerical features in machine learning, particularly effective when dealing with data containing outliers. It scales features using statistics such as the median and interquartile range (IQR). The formula for the Robust Scaler is as follow :

$$X_{\text{scaled}} = \frac{X - \text{median}(X)}{\text{IQR}(X)}$$

While IQR is the Interquartile Range, which is the range between the 1st quartile (25th quantile) and the 3rd quartile (75th quantile).

2.6 Models Overview

Regression models are essential tools in predictive modeling, particularly when predicting numerical targets like loss given default (LGD). These models establish relationships between predictors and the target variable, enabling accurate forecasts of continuous outcomes.Approaches such as Bagging, Boosting alongside with traditional linear and nonlinear regression models, offer adaptable solutions for analyzing complex relationships and making precise predictions.

2.6.1 Boosting Models

Boosting is a supervised machine learning technique that aggregates the predictions of several weak models to create an ensemble predictor. Unlike traditional ensemble methods like bagging or averaging, boosting specifically trains base models sequentially, focusing on samples that were previously misclassified. This approach helps the model prioritize harder-to-classify examples, enabling it to learn from its mistakes and iteratively improve its accuracy.

Algorithm

1. **Initialization of Weights:** Each training example is initially assigned an equal weight.

2. **Training a Weak Learner:** A weak learner is essentially a basic model that performs marginally better than making random guesses. is trained using the weighted training data. An example of a weak learner could be a decision tree with limited depth.

3. **Error Calculation:** The error rate is obtained based on the weak learner's performance over the weighted training data. The error specifically reflects the weighted sum of misclassified examples.

4. **Updating Weights:** Weights for the training examples are adjusted according to their error rates. Examples that were misclassified receive increased weights, whereas correctly classified examples receive reduced weights.

5. **Repetition:** The process from steps 2 to 4 is repeated multiple times. Each cycle involves training a new weak learner on the re-weighted training data.

6. **Combining Weak Learners:** The final model is an aggregation of all the weak learners trained in previous iterations. The influence of each learner on the final decision is proportional to its accuracy, with predictions from each learner being weighted accordingly.

7. **Prediction:** The composite model, now trained and optimized, is used to predict class labels for new instances.

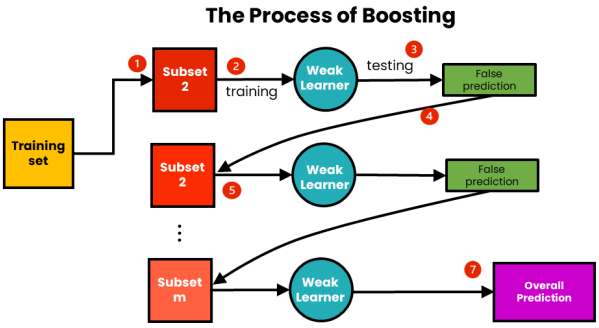

Figure 2.1: The Process of Boosting,

2.6.2 Gradient Boosting

Gradient boosting is renowned for its effectiveness with tabular datasets. It excels in uncovering complex, nonlinear relationships between the target variable and features within your data. Notably, gradient boosting demonstrates remarkable versatility, capable of handling missing values, outliers, and high cardinality categorical features without necessitating any specific preprocessing steps.

2.6.2.1 LightGBM

LightGBM (Light Gradient Boosting Machine) is a high-speed, distributed gradient boosting framework that utilizes decision tree algorithms for various machine learning tasks, including regression and classification. Developed by Microsoft, LightGBM is engineered to be efficient and scalable, offering several benefits: it uses less memory, operates more efficiently, trains faster, and achieves superior accuracy compared to many other machine learning frameworks.

LightGBM employs a histogram-based approach for building its decision trees.Here's how this method works:

- **Data Binning**: Instead of processing continuous values directly, LightGBM organizes data into bins. This is done by creating a histogram of the feature's distribution, where each bin represents a range of values. Data points are then placed into these bins based on their value. During the tree-building process, calculations for gains and splits are performed on these bins rather than on individual data points. This significantly reduces the number of computations needed, as the algorithm only needs to iterate over the bins, not every single data point.

- **Exclusive Feature Bundling (EFB):** To further enhance efficiency, LightGBM includes a mechanism called Exclusive Feature Bundling. This technique groups together features that are mutually exclusive. By bundling these features, LightGBM effectively reduces the dimensionality of the data, which speeds up the training process without significant loss of information.

2.6.2.2 XGBoost

XGBoost (eXtreme Gradient Boosting) is developed by Tianqi Chen, XGBoost is a powerful and adaptable gradient boosting framework that excels in various machine learning applications such as regression, classification, and ranking tasks. Designed for high computational efficiency and superior model performance, XGBoost is renowned for its quick execution speeds and accurate models. The framework employs several sophisticated techniques to enhance its efficiency and effectiveness:

- **Gradient Boosting Framework:** XGBoost is built on the gradient boosting framework, where new models are developed to predict the errors of previous models and are then combined to form the final prediction.

- **Tree Pruning:** Unlike traditional methods that build trees to their maximum size and then trim them back, XGBoost uses a depth-first approach that prunes trees during the growth phase itself, optimizing both speed and performance.

2.6.2.3 CatBoost

Originating from Yandex, CatBoost is a robust, open-source library designed for gradient boosting on decision trees. It supports classification, regression, and ranking tasks. Exceptionally effective at managing large, complex datasets with categorical variables, CatBoost employs a blend of ordered boosting, random permutations, and gradient-based optimization for superior performance. Its main features and techniques include:

- **Categorical Data Support:** CatBoost efficiently processes categorical data directly, without needing extensive preprocessing required by other machine learning models, such as one-hot

encoding. It employs an innovative algorithm to convert categorical variables into numerical data, preserving their informational value.

- **Symmetric Trees:** The library constructs balanced, symmetric trees, which ensures better generalization and reduces the risk of overfitting compared to other growth strategies such as depth-wise or leaf-wise expansion.

- **Ordered Boosting:** CatBoost's implementation of ordered boosting, a permutation-based technique, effectively reduces overfitting and prevents leakage of target information during training, offering unbiased and robust estimates.

2.6.2.4 AdaBoost

AdaBoost (Adaptive Boosting) is a pioneering boosting algorithm developed by Yoav Freund and Robert Schapire. It's a potent ensemble method that amalgamates several weak learners, often decision trees, to create a robust classifier or regressor. Primarily employed for binary classification, adaBoost can also be adapted for multi-class classification and regression tasks.
Key features and techniques of AdaBoost include:

- **Sequential Learning:** adaBoost enhances the performance of its ensemble by training weak learners sequentially. Each new learner focuses on the instances that previous models misclassified by increasing their weights, thereby correcting errors step-by-step.

- **Weak Learners:** AdaBoost generally uses decision stumps, which are one-level decision trees, as weak learners. This choice makes the algorithm both quick and less susceptible to overfitting. Nonetheless, any machine learning algorithm capable of handling weighted data can be employed as a base learner.

- **Error Weighting:** after incorporating each new learner, AdaBoost adjusts the weights of instances that were incorrectly classified to ensure subsequent learners pay more attention to these difficult cases. This iterative reweighting process systematically improves model accuracy by concentrating training on challenging areas of the data.

- **Weighted Voting:** In AdaBoost's final ensemble, each learner casts a vote to determine the output, based on its predictions. AdaBoost aggregates these votes into a weighted sum to produce the final decision, assigning higher weights to more accurate learners, thereby giving them greater influence in the final model.

2.6.3 Bagging Models

This ensemble technique enhances the stability and accuracy of machine learning models by independently training multiple models on random data subsets and then aggregating their predictions using methods such as voting or averaging.

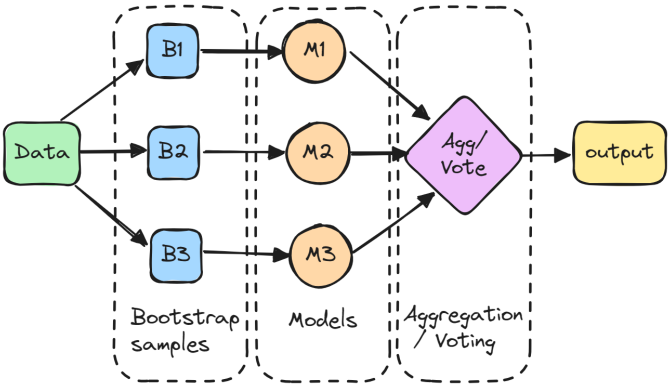

Figure 2.2: The Process of Bagging

Specifically, each model is trained on a bootstrap sample, which is a random subset of the data sampled with replacement. This approach allows the same data points to appear in a sample more than once. Training on diverse bootstrap samples helps reduce the variance of the individual models and mitigates overfitting by exposing different models to various aspects of the dataset.

The predictions from all the models are then aggregated, typically by averaging, to form the final prediction. This aggregation effectively combines the strengths of individual models and diminishes their errors, producing a more robust model.

Before delving into more advanced bagging models like Random Forest and Extra Trees, it is essential to understand decision trees, which serve as the basic building block for these complex methods.

2.6.3.1 Decision Trees

a decision tree is a supervised learning algorithm used for both classification and regression tasks. It represents a series of decision rules inferred from the data features. Essentially, a decision tree splits the data into branches, which represent the decision taken after considering some input features. Each end of the branch that doesn't split further is called a leaf and represents an output (class label in classification, value in regression).

- **Root Node:** This is the top-most decision node that represents the entire dataset, which then gets split based on a feature that results in the best split. The criteria to measure the "best split" depend on the algorithm (e.g., Gini impurity, entropy for classification; variance reduction for regression).

- **Splitting:** The data at each node is split into two or more homogeneous sets based on the most discriminatory features. This process is repeated recursively on each derived subset in a recursive manner called recursive partitioning.

- **Decision Node:** When a subset of the data is split on a particular feature, that condition is called a decision node.

- **Leaf/ Terminal Node:** Nodes that do not split further are called leaves or terminal nodes, representing the predictions or outcomes.

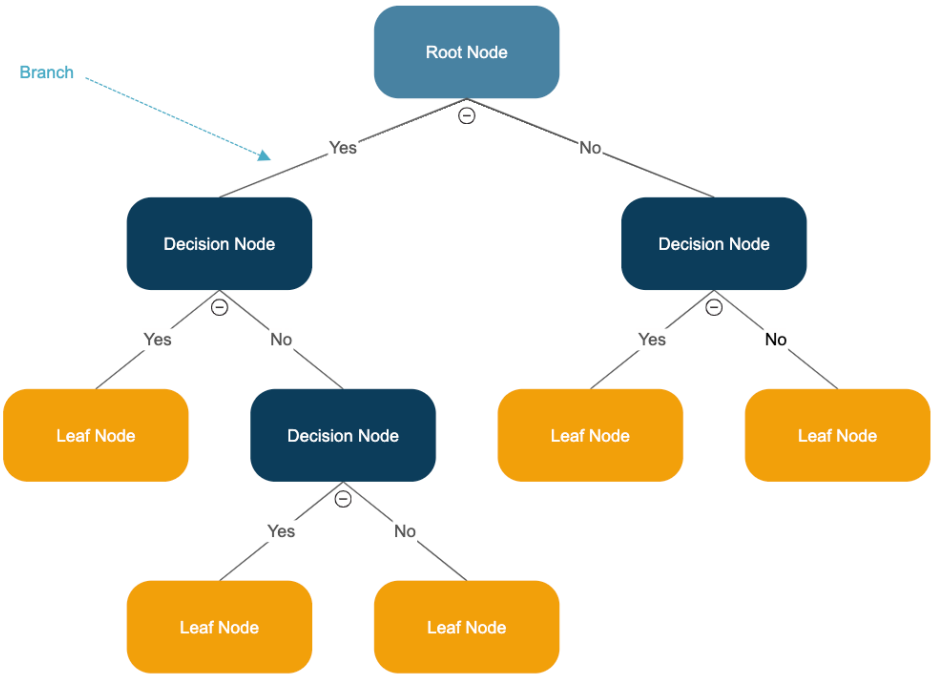

Figure 2.3: Structure of a Decision Tree

Decision Tree Pruning

Decision tree pruning is a technique aimed at reducing the complexity of the decision model and avoiding overfitting, which occurs when a model is too complex, capturing noise instead of reflecting the underlying data distribution. Pruning removes parts of the tree that do not significantly aid in predicting target variables. This can be done either during the tree's growth (pre-pruning) or after the tree has fully developed (post-pruning).

Pre-pruning curtails the development of the tree by imposing constraints such as the minimum number of samples required at a node before it can split, or the maximum depth allowed for the tree.

Post-pruning involves trimming branches from a fully grown tree, typically replacing them with leaf nodes if doing so does not significantly increase the prediction error. Pruning simplifies the decision tree, making the model easier to understand and quicker to execute, and enhances its ability to generalize from new, unseen data.

2.6.3.2 Random Forest

Random Forest is an advanced ensemble learning technique that constructs multiple decision trees and integrates them to produce more accurate and stable predictions. a Random Forest generally outperforms a single decision tree by mitigating the risk of overfitting through the averaging of multiple trees, each of which may individually exhibit high variance.

How Random Forest Works

- **Bootstrap Sampling:** Random Forest begins by generating multiple samples from the original dataset using bootstrap sampling, where each sample is created by randomly selecting data points with replacement.

- **Building Multiple Trees:** A decision tree is then built for each bootstrap sample. Notably, Random Forest introduces an additional layer of randomness during tree construction; it selects a random subset of features to consider when splitting a node, instead of using all features. This approach is known as feature bagging.

- **Aggregation:** Once all trees are trained, predictions from each tree are combined to make a final prediction. For classification tasks, this aggregation is usually performed through majority voting, selecting the class most frequently predicted by the trees. For regression tasks, predictions are averaged.

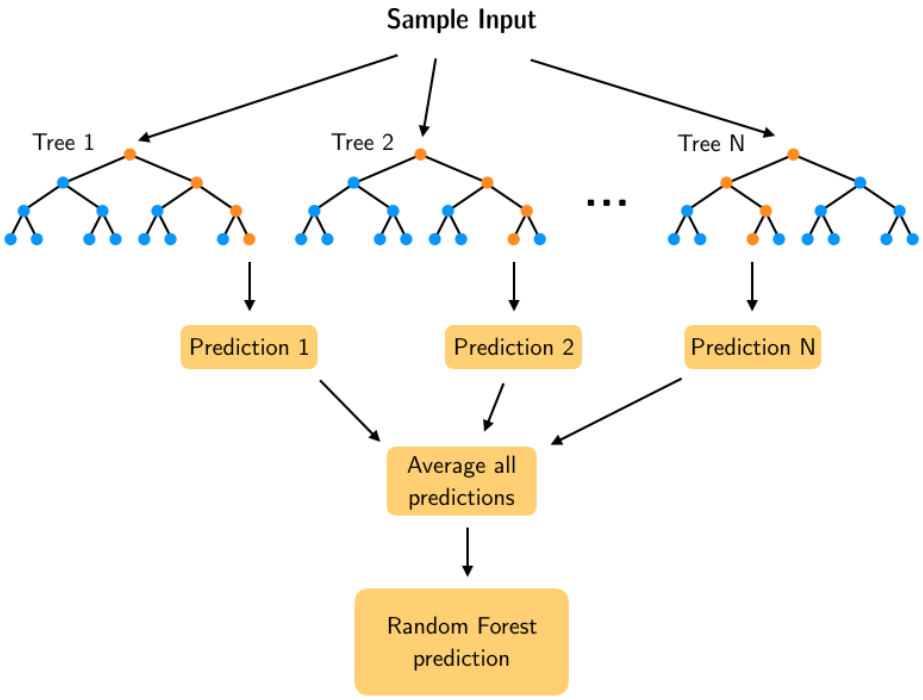

Figure 2.4: Random Forest

2.6.3.3 Extra Trees

Extra Trees builds multiple decision trees and averages their predictions. Unlike other methods, it introduces high randomness in choosing split points during tree construction, which can lead to faster training times and reduced model variance.

While both Extra Trees and Random Forest are powerful ensemble techniques that build on decision trees, they differ significantly in their approach to training.

- **Node Splits:** Extra Trees selects splits randomly within each feature's range and chooses the best random split, while Random Forest seeks the optimal split based on the data.
- **Computational Efficiency:** Extra Trees is faster to train because it skips the calculation for the best split point at each node.

2.6.4 Stacked Model

a stacked model, or stacking, is an advanced ensemble machine learning technique where multiple different models are combined to improve the predictive performance. Stacking involves training a new model to aggregate the predictions of several other models, effectively using those predictions as input for the final decision-making model.

Initially, multiple diverse models, known as base learners, are trained independently on the same dataset. These models can be of various types, including decision trees, support vector machines, or neural networks, providing a mix of predictions.

The predictions made by these base models are then used as inputs (features) for a second-level model, known as the meta-learner or meta-model. This model is trained to effectively combine the base models' predictions into a final prediction.

2.6.5 Voting Regressor

a Voting Regressor is an ensemble technique used in regression tasks that improves prediction accuracy by averaging the outputs of multiple diverse base models. It involves training various regressors independently, and then combining their predictions through simple or weighted averaging. This method effectively reduces variance and increases the stability and reliability of the predictions, making it especially useful when no single model provides consistently superior results.

2.7 Regression Metrics

2.7.1 Root Mean Squared Error

The Square Root of Mean Squared Error (RMSE) represents the square root of the average of the squared discrepancies. It essentially gauges the standard deviation of these deviations. An advantage of RMSE is that it is presented in the same units as the dependent variable, which simplifies the scale and understanding of errors.

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2}$$

2.7.2 Mean Absolute Error

Mean Absolute Error (MAE) quantifies the average magnitude of the errors in a set of predictions, without considering their direction. It is the mean over the test sample of the absolute differences between prediction and actual observation where all individual differences have equal weight.

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|Y_i - \hat{Y}_i|$$

2.7.3 Mean Absolute Percentage Error

Mean Absolute Percentage Error (MAPE) measures the prediction accuracy as a percentage, averaging the absolute percentage errors between the predicted and actual values across a dataset. A key benefit of MAPE is scale independence facilitates the comparison of model performances across varied scales or measurement units.

$$\text{MAPE} = \frac{100\%}{n}\sum_{i=1}^{n}\left|\frac{Y_i - \hat{Y}_i}{Y_i}\right|$$

2.8 Conclusion

This chapter was dedicated to explaining the LGD calculation method as well as the theoretical aspect of the machine learning models adopted in this project. We also mentioned some techniques for variable selection and encoding of qualitative variables. Next, we will focus on the results of applying these techniques.

Chapter 3

Methodology

3.1 Introduction

In the third chapter, we will describe the database, then we will detail the data cleaning steps, and the last part will be devoted to representing the results of descriptive statistics as well as the variable selection methods that we will use in the modeling.

3.2 Data collection

The data for this study was gathered during an internship project at Quantylix. It comprises information on Egyptian clients who took out loans between 2011 and 2021. The dataset includes variables essential for LGD calculation, such as recoveries and fees, along with other financial and behavioral variables.

3.3 LGD calculation

3.3.1 Default dates inconsistencies

Some loans had overlapping default periods within the same contract. This happens when the end date of one default period is later than the start date of the next period. To ensure accurate analysis

irrelevant rows were removed from the database.

3.3.2 LGD calculation

After addressing the default dates inconsistencies, we proceeded with the calculation of Loss Given Default (LGD). Initially, related defaults were identified and analyzed as explained in section 2.2.1. Following this, Formula 2.1 was employed to calculate the LGD for each client, using recoveries and fees. In accordance with IFRS 9 guidelines, only external fees were considered in this calculation.

3.4 Data base presentation

The database consists of information on Egyptian retail clients. It contains 3,534 observations and 19 variables. These variables include 10 quantitative variables, 6 qualitative variables, and 3 datetime variables. The period of observation covers 5 years.

3.4.1 Quantitative variables

- **Client Age:** The age of the client.
- **Interest Rate:** The annual percentage rate charged on the loan amount.
- **Loan Period:** The total duration of the loan measured in years.
- **Installment Amount:** The amount the client is required to pay at regular intervals.
- **Past Due Amount:** The total amount that is overdue for each default, indicating the unpaid portion of the loan during periods of default.
- **Final DPD (Days Past Due):** The total number of days the client has been past due on payments for each default by the end of the observation period.

3.4.2 Qualitative variables

- **LOAN TYPE DESC**: The type of loan taken by clients, including various categories such as personal loans, car loans, housing loans, and others.
- **Categories/Segments**: Different segments or categories within the loan portfolio, such as secured loans, real estate loans, special nature programs, and others.
- **Main Program**: The primary program or initiative under which the loan is offered, such as secured cash, mortgage CBE initiative, secured auto, and others.
- **Product line**: The specific product line or category of loans, such as cash loans, mortgage loans, and auto loans.
- **PQR Categories/Segments Products:** This variable provides further segmentation or categorization of loan products based on specific criteria or characteristics, such as secured cash loans, mortgage CBE initiative, secured auto loans, and others.
- **SAS Segment**: This variable might represent segments or categories defined by a specific system or software, such as secured, mortgage, auto secured loans, and others.
- **Installement Freq:** This variable indicates the frequency at which loan installments are paid by clients, such as monthly, quarterly, etc.
- **Denoument:** A binary variable indicating whether clients have fulfilled their obligations by returning to pay their loans or not.
- **Security:** A binary variable indicating if the loan is secured by collateral or not.
- **Industry:** Represents the profession the client.
- **Branch Name:** Represents the geographical location of the client

3.4.3 Date times variables

- **Start Default Date :** Default starting date
- **End Default Date :** Default ending date
- **Booking Date :** Relation start date

3.5 Creating new variables

We have introduced two new variables to better understand loan defaults:

- **Time in Default (TID):** Calculated as the duration between the start and end of a default.
- **Time to Default (TTD):** Calculated as the interval from the booking date of the loan to the start of the default.

3.6 Data cleaning

3.6.1 Missing values imputation

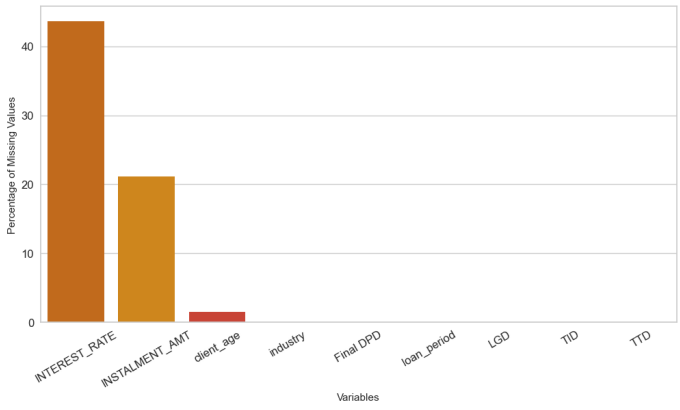

Figure 3.1: Percentatage of missing values in each variable

Figure 4.9 highlights the percentage of missing data in each variable.Almost all variables have no missing values excluding Interest Rate, Installment Amount, and Client Age. The Interest Rate and

Installment Amount variables exhibit significant proportions of missing data, accounting for 44% and 21% respectively. For these financial variables, we've employed a specific formula to estimate missing entries, which has proven effective over the dataset's observations without missing data. The formula used is:

$$M = \frac{P \cdot r \cdot (1+r)^n}{(1+r)^n - 1} \tag{3.1}$$

where M represents the monthly payment, P is the principal loan amount, r is the monthly interest rate, and n is the number of payments. This approach provides a robust means of handling missing data by leveraging predictable financial calculations.

For the variable Client Age, which exhibits a relatively lower missing data percentage of 2%, we have imputed the missing values using the KNN imputer.

3.6.1.1 Outliers detection

To ensure the accuracy and reliability of our analysis, we employed the Interquartile Range (IQR) method to handle outliers. This method involves the following steps:

Interquartile Range (IQR)

- **First Quartile (Q1)**: The value below which 25% of the data points fall.
- **Third Quartile (Q3)**: The value below which 75% of the data points fall.
- **Interquartile Range (IQR)**: The range between the first and third quartiles, representing the middle 50% of the data.
  - **Interquartile Range (IQR)** = Q3 - Q1.
  - **Lower Limit** = Q1 - 1.5 * IQR.
  - **Upper Limit** = Q3 + 1.5 * IQR.

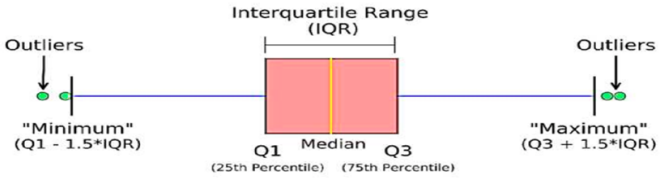

Figure 3.2: Interquartile Range method

In the next step, we examine all the values of the variable. Values that fall below the lower limit or above the upper limit are considered as outliers.

3.6.1.2 Outliers dealing

Outliers can significantly impact the performance of certain machine learning models. To manage them effectively, various techniques are available. In our approach, we employed both outlier removal and capping methods.

- **Outlier Removal:** For each quantitative variable, outliers detected using the interquartile method are eliminated.
- **Outlier Capping:** Values exceeding the upper limit are replaced with the upper limit value, while values below the lower limit are replaced with the lower limit value.

3.7 Grouping categorical variables

For BRANCH Name variable we have created 6 new modalities : 'Northern Egypt', 'Nile Delta' 'Cairo Metropolitan Area', 'Upper Egypt', 'Red Sea Coast' and 'Canal Zone' and changed it's name to 'region'

For industry variable we have created 5 new modalities : 'government public sector', 'pensioners retirees', 'financial sector', 'private sector' and 'special misc'

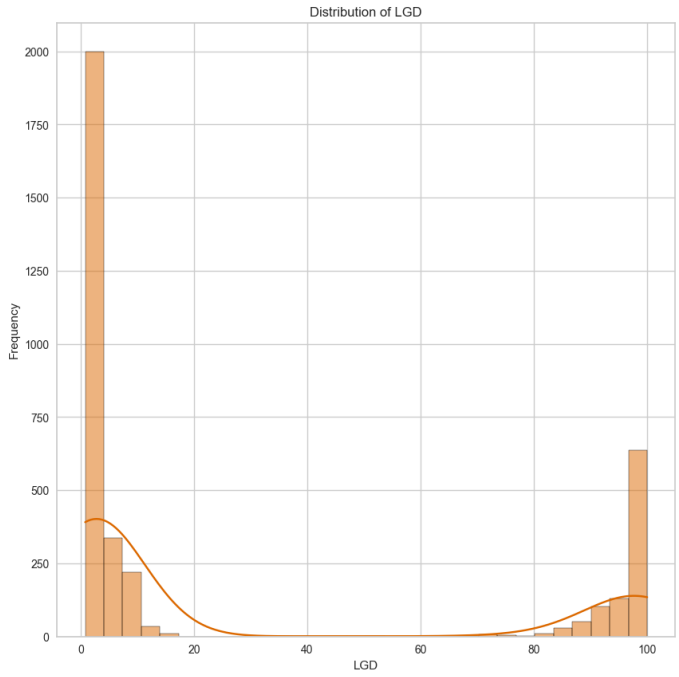

Figure 3.3: Loss Given Default distribution

3.8 Data analysis

3.8.1 Univariate analysis

3.8.1.1 Target analysis

The histogram of Loss Given Default (LGD) values shown in figure 3.2 reveals a bimodal distribution, indicating two distinct peaks. The first and most prominent peak occurs at the lower end of the LGD scale around 0, suggesting that a large proportion of defaults result in little to no loss. The second, smaller peak appears at the upper end around 100, indicating that a significant portion of defaults result in a total loss. Between these extremes, the frequency of LGD values is relatively low, highlighting that moderate losses are less common.

3.8.1.2 Product line analysis

The bar chart shown in figure 3.3 illustrates the distribution of denouement categories across different product lines, showing that Cash Loans dominate the dataset with 81.55%Åuto Loans account for

14.99% representing a significant but smaller portion of defaults, while Mortgages are the least represented, comprising only 3.46% of the total.

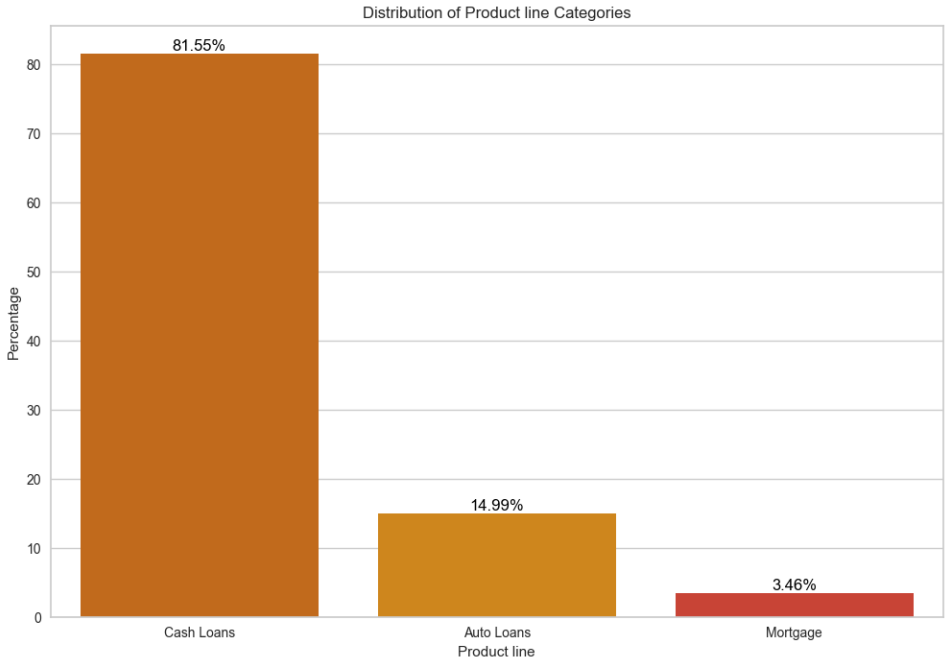

Figure 3.4: Product line distribution

3.8.1.3 Region analysis

The bar chart shown in figure 3.4 displays the distribution of region categories, showing that the Cairo Metropolitan Area has the highest percentage at 47.84% indicating nearly half of the dataset originates from this region. The "Other" category follows with 19.42% .The Nile Delta and Northern Egypt account for 12.71% and 10.70% respectively, highlighting their significant contributions. Upper Egypt represents 6.07% of the total, while the Canal Zone and Red Sea Coast are the least represented, each comprising around 1.6%.

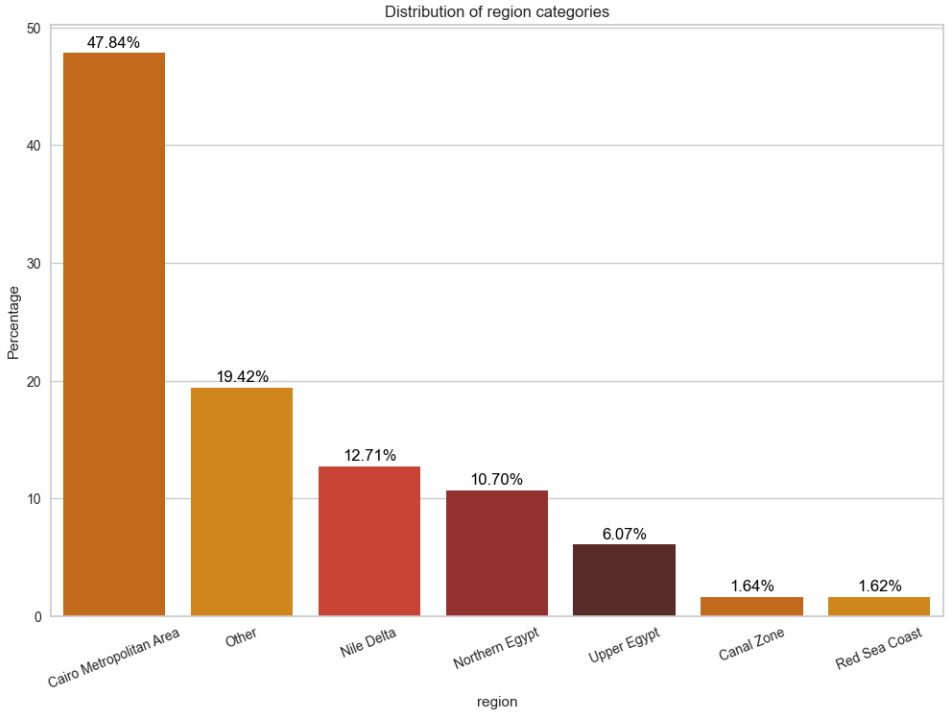

Figure 3.5: Region distribution

3.8.1.4 Profession analysis

The bar chart in figure 3.5 shows the distribution of client professions.It reveals a dominant private sector at 51.46% .Following closely behind is the government and public sector with 40.60% .The remaining workforce, divided between finance, retirees, and others, comprises less than 8%

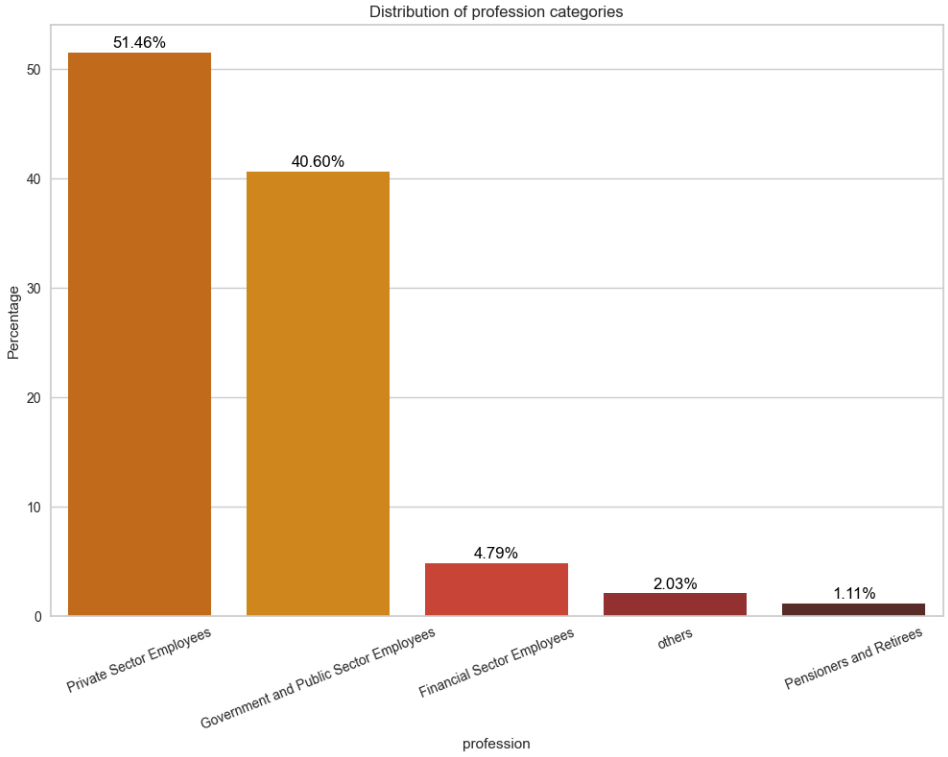

Figure 3.6: Profession distribution

3.8.1.5 Client age analysis

The graph in figure 3.6 shows the distribution of client age.It appears The majority of loans are requested by clients between the ages of 30 and 40.

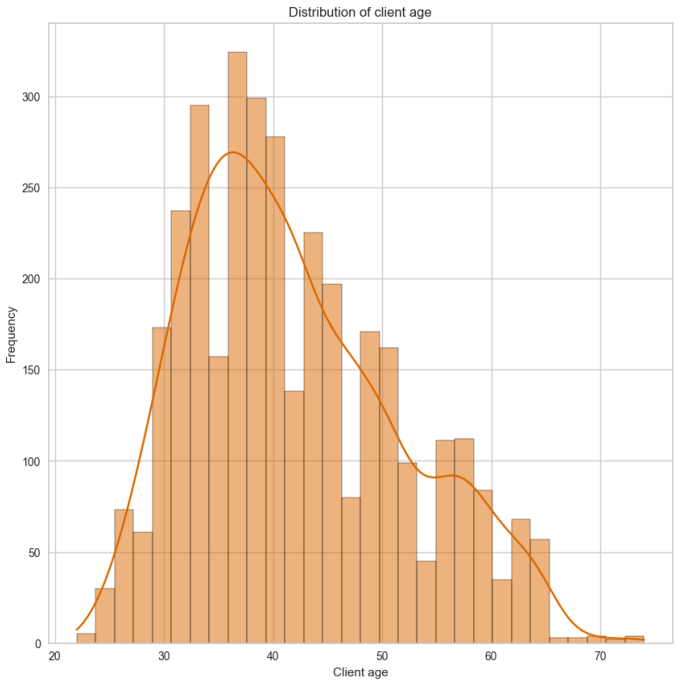

Figure 3.7: Client age distribution

3.8.2 Bivariate analysis

in this section we will analyse the relation between different variable and also with the target. we will start by a graphic analyse then we will move to statistical test and measures.

3.8.2.1 Graphic analyse

The chart in figure 3.7 compares the LGD for two groups of clients: RES and radie ttw. RES stands for clients who defaulted on a loan but then returned to pay their obligations. Raddie ttw stands for clients who defaulted on a loan and remained in default throughout the five-year observation period. RES clients show a noticeably lower LGD compared to radie ttw clients. This is likely because they eventually repay their debts after defaulting, resulting in smaller losses for the bank.

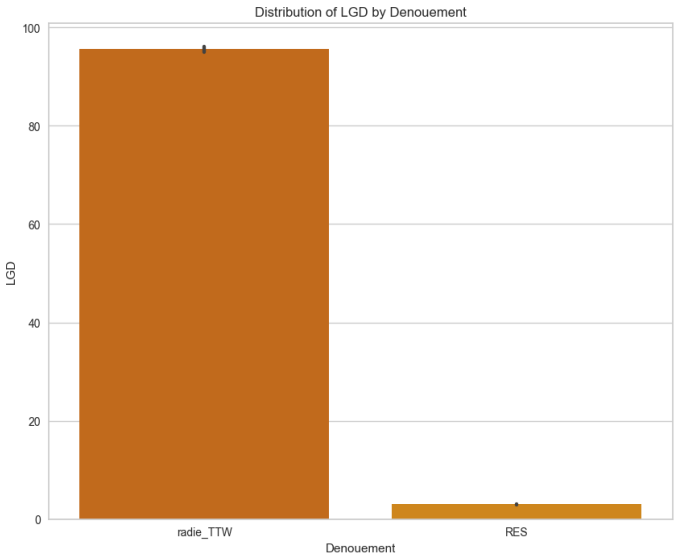

Figure 3.8: Distribution of LGD by Denouement

The graph below in figure 3.8 indicates a significant reduction in average loss for banks when issuing secured loans compared to unsecured loans. This is due to the presence of collateral, which serves as a safety asset that the bank can use in the event of default.

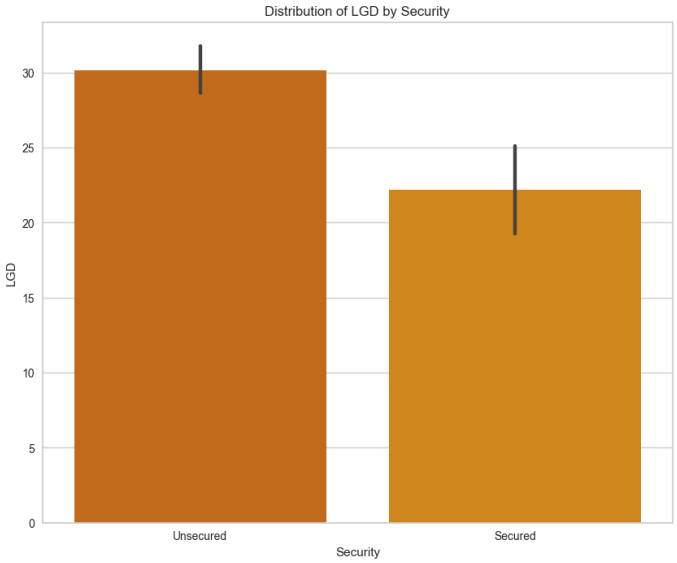

Figure 3.9: Distribution of LGD by Security

3.8.2.2 Correlation matrix

The spearman coefficient, ranging from -1 to 1, reflects the correlation direction between two variables. A value close to 1 indicates a positive correlation, meaning they move together. Conversely,

a value near -1 signifies a negative correlation, where they move in opposite directions. If the coefficient is close to 0, it suggests the two quantitative variables are independent or weakly correlated, implying little to no relationship between their values.

Based on the correlation heat-map shown in figure 3.9 we can observe that :

- There is a high positive correlation between LGD, TID, and Final DPD. In other words, as the length of time a client is in default (TID and Final DPD) increases, the LGD also tends to increase. This makes sense intuitively, as longer defaults are likely to result in larger losses for the bank.

- There is a high positive correlation between the installment amount and the past due amount. This means that these two variables also tend to move together. In other words, clients with larger installment amounts tend to have larger past due amounts.

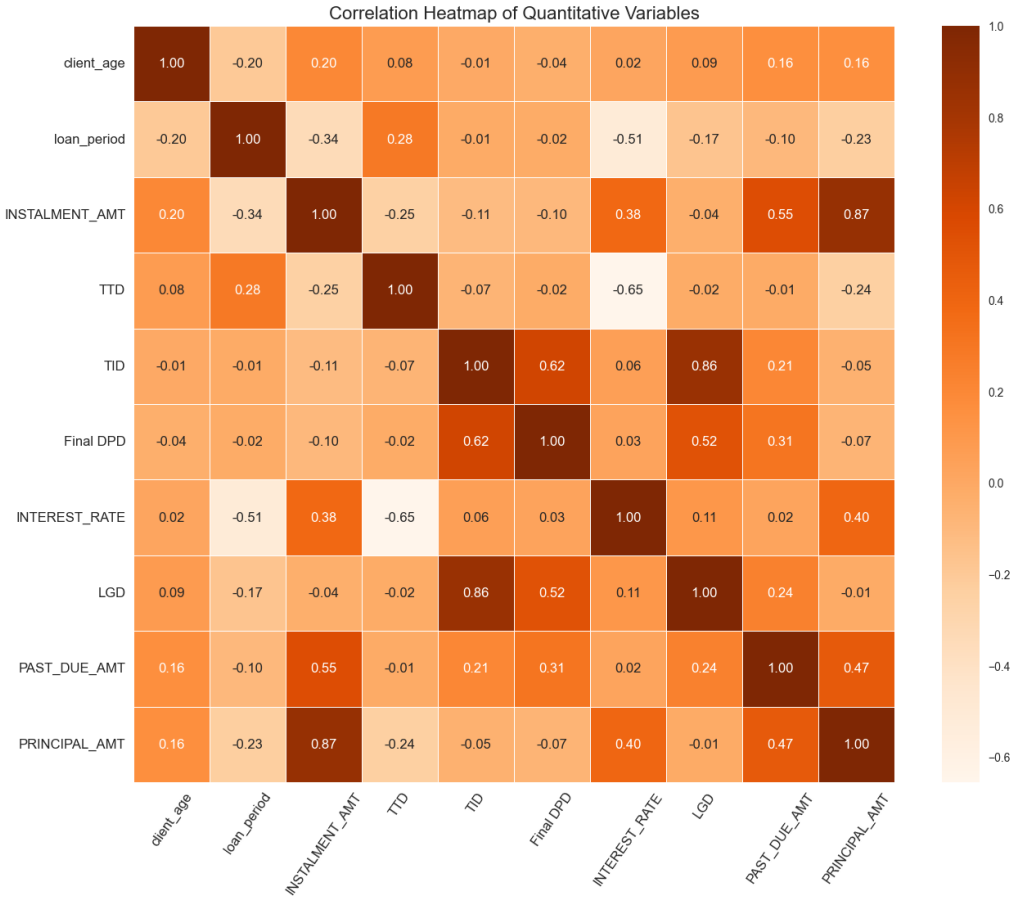

Figure 3.10: Correlation matrix heat-map

3.8.2.3 Cramer distance

The Cramer's distance heatmap reveals a very strong association between several categorical variables describing loan types. These variables, including "LOAN TYPE DESC," "Categories/Segments," and "Main Program," likely all represent the same underlying information but use different labels. To avoid redundancy and enhance clarity in our analysis, we've chosen to focus on the "Product line" variable. This selection offers a clear advantage due to its interpretability with only three categories (cash, auto, and mortgage loans)

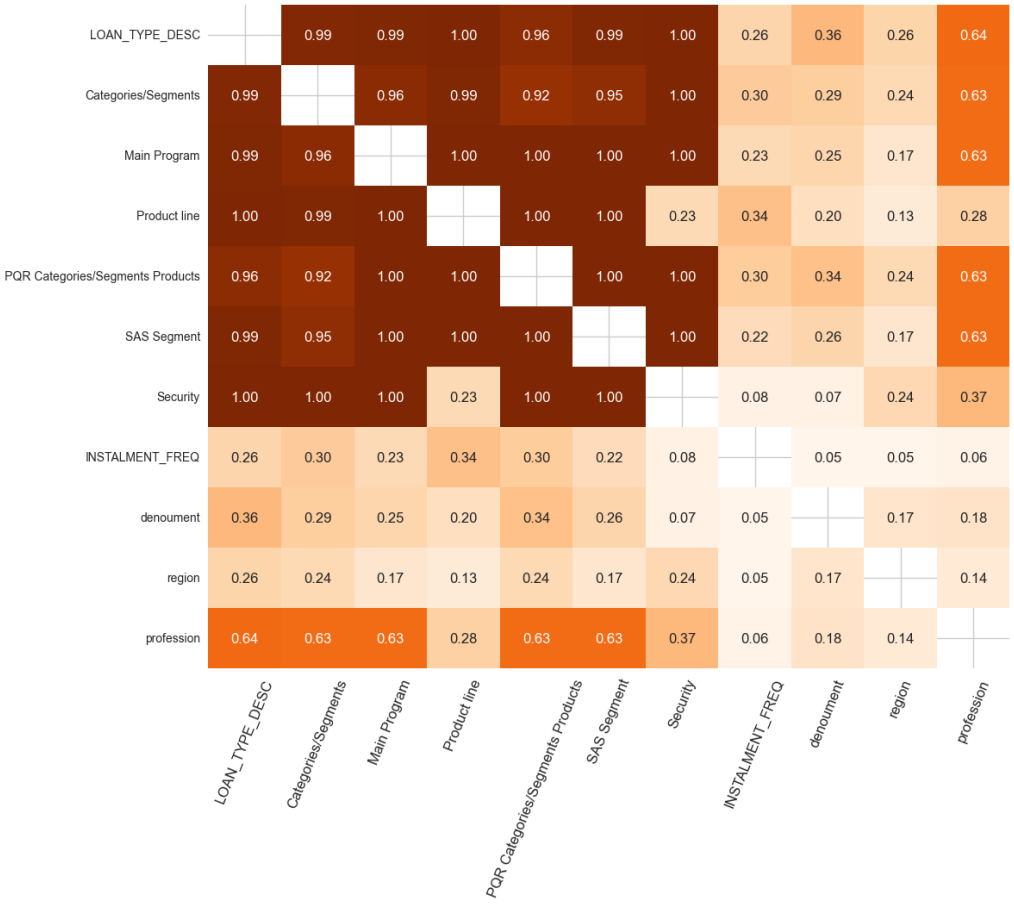

Figure 3.11: Associations between Categorical Variables (Cramer's V)

3.8.2.4 Anova test

We employed the ANOVA Test to evaluate whether the means of the target variable and various quantitative variables were equal .

Categorical Variable	p-value
LOAN TYPE DESC	1.74 e-97
Categories/Segments	8.50 e-630
Main Program	3.26 e-47
Product line	1.02 e-34
PQR Categories/Segments Products	1.08 e-86
SAS Segment	1.07 e-49
Security	5.76 e-06
INSTALMENT FREQ	1.94 e-0.2
NB Defaults	2.35 e-35
denoument	4.65 e-123
region	6.22 e-22
profession	1.86 e-28

Table 3.1: Results of Anova test

Th results shown in table 3.1 revealed statistically significant differences between the means of the groups for all variables (p-value < 0.05). This outcome allows us to reject the null hypothesis, and conclude that there is significant differences in the means between the groups.

3.9 Feature Selection

In this section, we employ two feature selection techniques. First, we use the SelectKBest method to evaluate all features based on mutual information described in section 2.1, for both pre- and post-default variables. Second, we utilize Sequential Feature Selection (SFS), an iterative method that adds or removes features one at a time based on model performance, optimizing the feature sets separately for the two approaches.

3.9.1 SelectKBest

Based on the results shown in Figure 3.11, which highlights the top 15 features based on mutual information score, we can observe that TID, final DPD, and denouement have the highest scores compared to other variables.

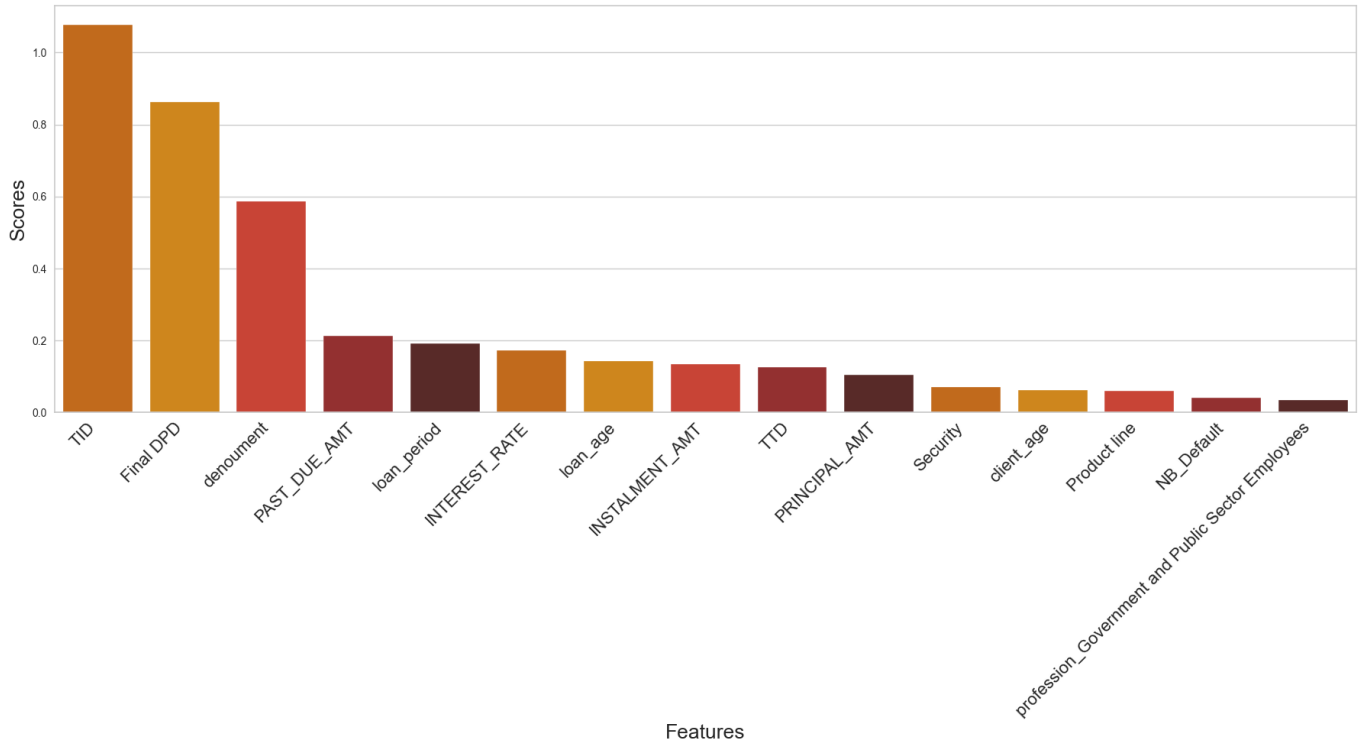

Figure 3.12: Top 15 features based on mutual information

3.9.2 Sequential Fetaure Selection

3.9.2.1 Post-default approach

To identify the most important factors influencing our analysis, we employed a sequential feature selection method (detailed in Section 2.10). This process resulted in the selection of 12 critical variables, which are listed in Table 3.2.

Selected Features	
Principal Amount	**Loan Period**
Interest Rate	**Total In Default (TID)**
Instalment Amount	**Time To Default (TTD)**
Past Due Amount	**Profession**
Final Days Past Due (DPD)	**Region**
Denoument	**Security**

Table 3.2: Selected features by SFS method in the Post-default approach

3.9.2.2 Pre-default approach

In the Pre-default appraoch, 9 variables were selected using sequential feature selection method.

Selected Features	
Principal Amount	**Product Line**
Interest Rate	**Security**
Instalment Amount	**Region**
Client Age	**Profession**
Loan Period	

Table 3.3: Selected features by SFS method in the Pre-default approach

3.10 Conclusion

In this chapter, we have detailed the preprocessing steps of our database, such as the treatment of outliers, and the grouping of modalities of some variables that have a very high number of categories. We have also presented some graphs that describe our dataset and we have described variable selection approaches that we will adopt in the fourth modeling chapter.

Chapter 4

Results

4.1 Introduction

This chapter is divided into two main parts. In the first part, we will present the best model results following the optimization of hyperparameters for both the post-default and pre-default approaches. The second part will cover the deployment of these models in a production environment.

4.2 Modeling results

We have implemented several regression models using different approaches. The models that attained the best results were Random Forest, CatBoost, LightGBM, Stacking, and Voting Regressors. For the encoding of categorical variables, we ultimately used label encoding, which performed slightly better than one-hot encoding.

For each model we will first present the definitions of its key hyper parameters, followed by a table showcasing the optimized results achieved through hyper parameter tuning using GridSearch.

4.2.1 Post-default approach results

4.2.1.1 Random Forest

- **Bootstrap:** This parameter determines whether or not bootstrap samples are used when building trees. Bootstrap sampling involves randomly sampling the dataset with replacement, which means some data points may be sampled multiple times while others may not be included at all.

- **Max Depth:** It determines the maximum depth of the decision trees in the forest. A deeper tree may capture more intricate relationships in the data, but it also increases the risk of overfitting.

- **Min Samples Leaf:** This parameter specifies the minimum number of samples required to be at a leaf node. A node will not be split further if it leaves fewer samples than the value specified by this parameter.

- **Min Samples Split:** It sets the minimum number of samples required to split an internal node. If the number of samples at a node is less than this value, the node will not be split.

- **N Estimators:** This parameter defines the number of trees in the random forest. Increasing the number of trees typically improves the performance of the model, but it also increases computation time.

Table 4.1 below presents the optimized hyperparameters values, while Figure 4.1 illustrates the most important features in the Random Forest model.

	bootstrap	max depth	min leaf	min split	n estimators
Random Forest	True	10	2	2	100

Table 4.1: Hyperparameter Results for Random Forest Model

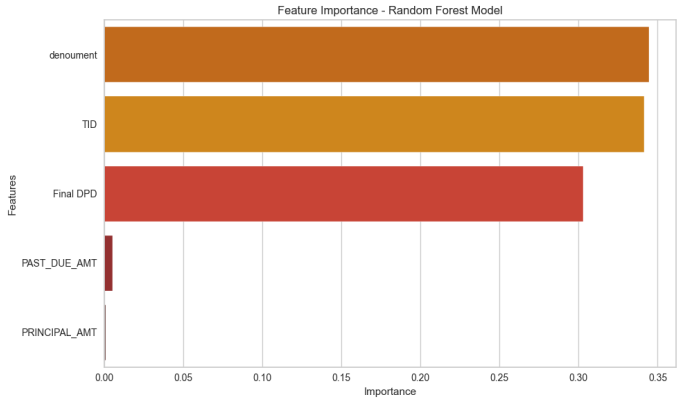

Figure 4.1: Feature importance for Random Forest Model

According to this graph, we observe that the most important variables in constructing our random forest model are primarily the variables : denoument, Time In Default (TID) and Final DPD.

4.2.1.2 Catboost

- **Depth:** This parameter determines the depth of the trees in the ensemble. Deeper trees can capture more complex patterns in the data but may also lead to overfitting.
- **Iterations:** It specifies the number of boosting iterations (trees) to be built during the training process. Each iteration focuses on correcting the errors made by the previous trees.
- **L2 Leaf Regularization:** This parameter controls L2 regularization applied to the leaf weights. Regularization helps prevent overfitting by penalizing large weights in the model.
- **Learning Rate:**It determines the step size at each iteration while moving towards the minimum of the loss function. A lower learning rate typically leads to more stable convergence but may require more iterations to reach the optimum.

The optimized hyperparameters values are presented in Table 4.2, while Figure 4.2 illustrates the most important features in the CatBoost model.

	depth	iterations	l2 leaf reg	learning rate
CatBoost model	6	300	1	0.1

Table 4.2: Hyperparameter Results for CatBoost Model

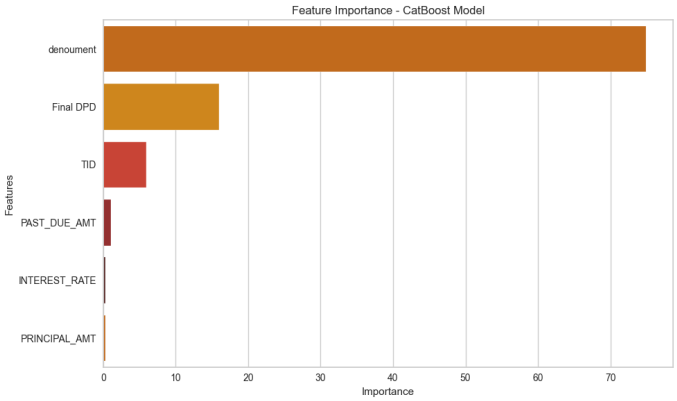

Figure 4.2: Feature importance for CatBoost Model

According to this graph, we observe that the most important variables in CatBoost model are primarily the variables : denoument, Time In Default (TID) and Final DPD.

4.2.1.3 LightGBM

- **Boosting Type:** This parameter determines the type of boosting algorithm to use. LightGBM supports various boosting types such as Gradient Boosting Decision Tree (GBDT), Random Forest, and Gradient-based One-Side Sampling (GOSS).

- **Learning Rate:** It controls the step size at each iteration while moving towards the minimum of the loss function. A smaller learning rate generally leads to more precise models but requires more iterations to converge.

- **N Estimators:** This parameter defines the number of boosting iterations (trees) to be built during the training process. Each iteration focuses on correcting the errors made by the previous trees.

- **Num Leaves:** It specifies the maximum number of leaves per tree. Increasing this parameter allows the model to capture more complex patterns in the data but may also lead to overfitting if not properly controlled.

Table 4.3 below presents the optimized hyperparameters values, while Figure 4.3 illustrates the most important features in the LightGBM model.

	boosting type	learning rate	n estimators	num leaves
LightGBM model	gbdt	0.1	300	50

Table 4.3: Hyperparameter Results for LightGBM Model

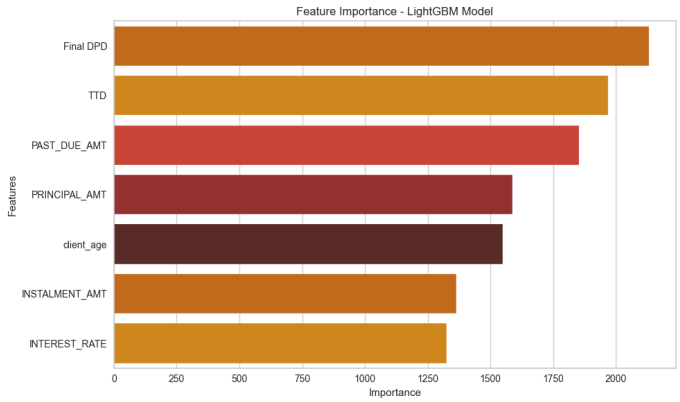

Figure 4.3: Feature importance for LightGBM Model

According to this graph, we observe that the most important variables in CatBoost model are primarily the variables : Past due amount, Time In Default (TID), Final DPD, client age, Instalment amount,Principal amount and Interest rate.

4.2.1.4 Stacked Model

For the stacked model, we utilized Random Forest, CatBoost, and LightGBM as base estimators, with Linear Regression serving as the meta-learner. The results are shown in Table 4.4.

4.2.1.5 Voting Model

In the voting model, we combined Random Forest, CatBoost, and LightGBM as base estimators. The results are shown in Table 4.4.

4.2.1.6 Results Comparison

	MAE	**MAPE**	**RMSE**
Voting	**1.43**	0.31	**3.73**
Stacked	1.45	0.32	3.73
Random Forest	1.46	0.28	3.9
CatBoost	1.51	0.32	3.83
LightGBM	1.54	**0.27**	3.83

Table 4.4: Metrics results for final models

The performance metrics indicate that the Voting model performs the best overall, with the lowest MAE (1.43) and RMSE (3.73), and a MAPE (0.31). The Stacked model follows closely, also showing strong performance with an MAE of 1.45 and RMSE of 3.73, though with a slightly higher MAPE of 0.32. Among the individual models, Random Forest performs moderately well but has the highest RMSE (3.90), while CatBoost and LightGBM exhibit similar performance, with higher MAE values but LightGBM achieving the lowest MAPE (0.27). These results highlight that ensemble methods (Voting and Stacked) generally provide better accuracy and robustness compared to individual models.

4.2.2 Pre-default approach results

4.2.2.1 Random Forest

Table 4.5 below presents the optimized hyperparameters values, while Figure 4.4 illustrates features importance in the Random Forest model.

	bootstrap	max depth	min leaf	min split	n estimators
Random Forest	True	20	2	2	200

Table 4.5: Hyperparameter Results for Random Forest Model

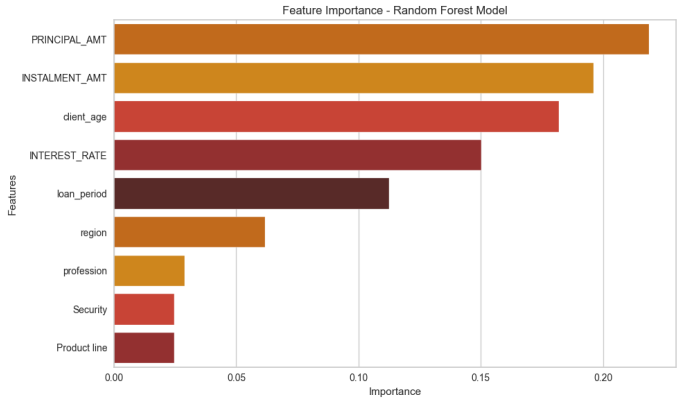

Figure 4.4: Feature importance for Random Forest model

According to this graph, we observe that the most important variables in Random Forest model are primarily the variables : Principal amount, Instalment amount, client age and Interest rate.

4.2.2.2 Catboost

Table 4.6 below presents the optimized hyperparameters values, while Figure 4.5 illustrates features importance in the CatBoost model.

	depth	iterations	l2 leaf reg	learning rate
CatBoost model	6	300	3	0.1

Table 4.6: Hyperparameter Results for CatBoost Model

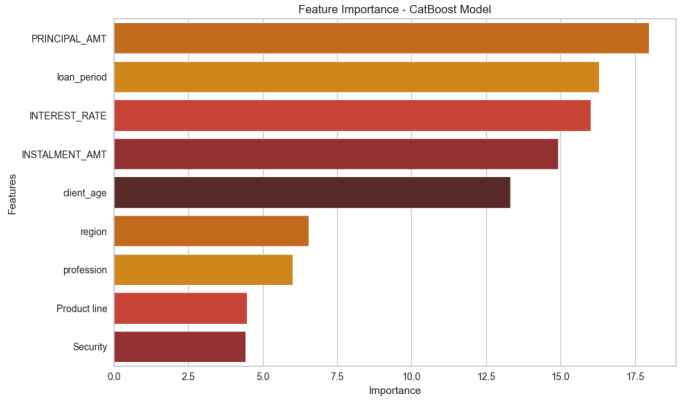

Figure 4.5: Feature importance for CatBoost model

According to this graph, we observe that the most important variables in CatBoost model are primarily the variables : Principal amount,Loan period, Instalment amount, client age and Interest

rate.

4.2.2.3 LightGBM

Table 4.7 below presents the optimized hyperparameters values, while Figure 4.6 illustrates features importance in the LightGBM model.

	boosting type	learning rate	n estimators	num leaves
LightGBM model	dart	0.1	300	30

Table 4.7: Hyperparameter Results for LightGBM Model

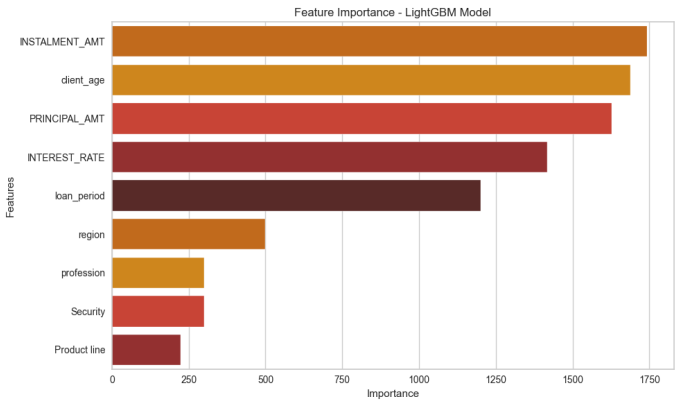

Figure 4.6: Feature importance for LightGBM model

According to this graph, we observe that the most important variables in LightGBM model are primarily the variables : Instalment amount, client age,Principal amount, Interest rate and Loan period.

4.2.2.4 Stacked Model

For the stacked model, we utilized Random Forest, CatBoost, and LightGBM as base estimators, with Linear Regression serving as the meta-learner. The results are shown in Table 4.8.

4.2.2.5 Voting Model

In the voting model, we combined Random Forest, CatBoost, and LightGBM as base estimators. The results are shown in Table 4.8.

4.2.2.6 Results Comparison

The table 4.12 below presents the performance metrics Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE), and Root Mean Squared Error (RMSE) for each model.

	MAE	**MAPE**	**RMSE**
Stacked	26.67	8.31	**35.03**
Voting	26.93	8.44	35.19
Random Forest	**26.49**	8.55	35.12
CatBoost	28.43	9.06	36.32
LightGBM	26.94	**8.24**	35.46

Table 4.8: Metrics results for final models

The results reveal that the Random Forest model achieves the lowest MAE of 26.49 among all models. Following closely, the Stacked model has an MAE of 26.67. The Voting model maintains a similar level of performance with an MAE of 26.93. CatBoost exhibits the highest errors across all metrics, with an MAE of 28.43. These findings suggest that ensemble methods, particularly the Stacked and Voting models, offer improved accuracy and reliability compared to individual models.

4.3 Deployment

Deploying machine learning models from the development environment to real-world applications is a critical step that bridges the gap between theoretical knowledge and practical utility. While model training and evaluation are essential components of the machine learning pipeline, deployment enables these models to be integrated into production systems, where they can be used to make predictions and automate tasks.

In our project we decided to use Flask for this task. For the frontend part, we utilized HTML, CSS, and JavaScript to enhance the presentation and interactivity of the web pages.

The principal page of the API, illustrated in Figure 4.7, is organized into four main sections. The first section provides a tool for directly calculating Loss Given Default (LGD) using conventional formulas. The next two sections Pre-Default and Post-Default approaches are dedicated to enhancing

the capabilities of this API through models training and applying these models to predict LGD. Finally, the dashboard, which constitutes the last section, offers a comprehensive view through various interactive graphs and charts, enabling users to visualize and analyze trends and patterns within the data effectively.

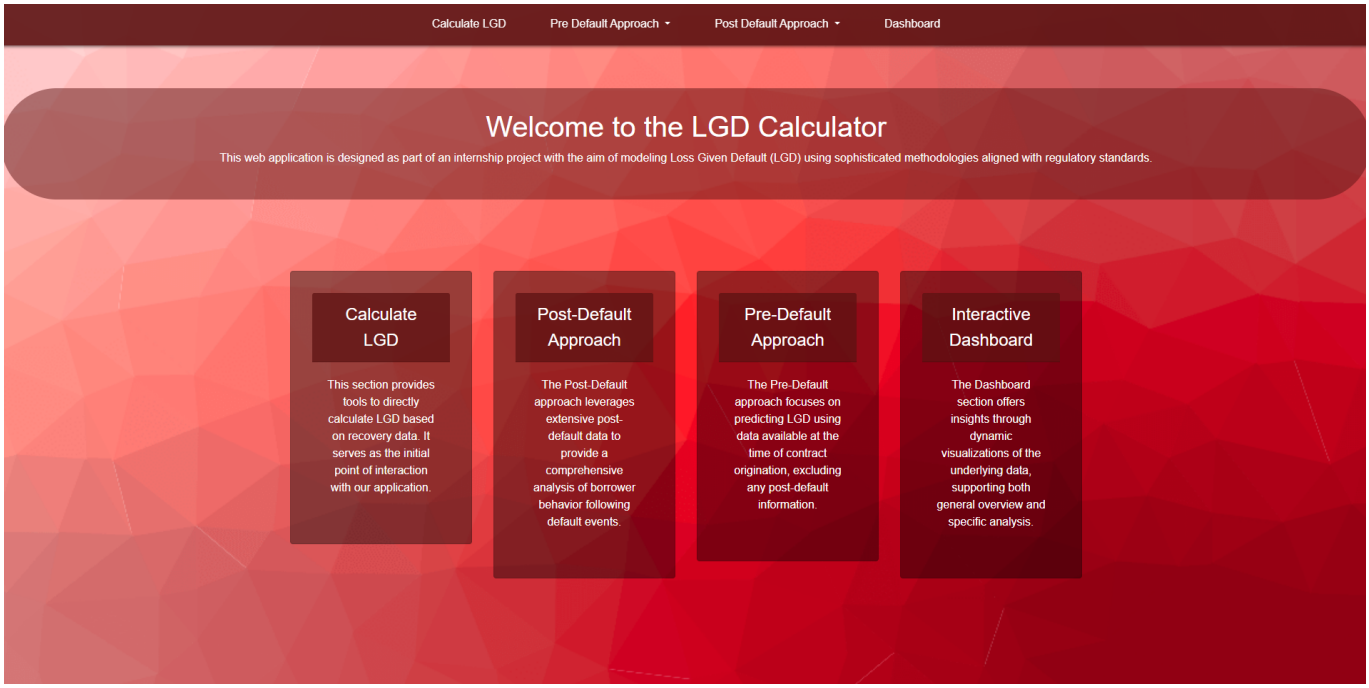

Figure 4.7: Main page

4.3.1 Loss Given Default Calculation

The LGD calculation page allows users to input data and obtain LGD predictions. In Figure 4.8, users input parameters such as the number of recoveries, Exposure at Default (EAD), and discount rate. After tapping 'Calculate,' the API calculates LGD, as shown in Figure 4.9.

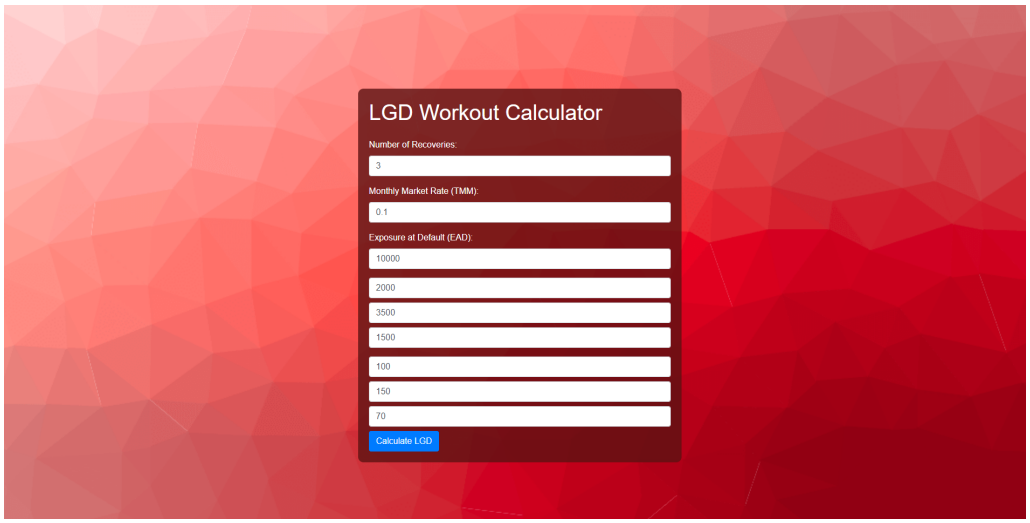

Figure 4.8: LGD calculation page

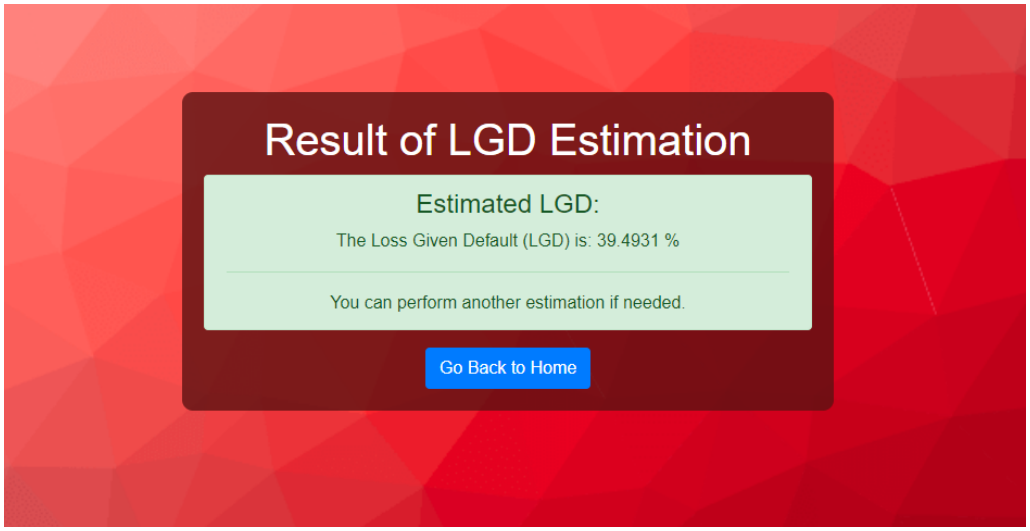

Figure 4.9: LGD calculation result page

4.3.2 Pre-Default Approach

4.3.2.1 Model training

This section focuses on training machine learning models. Users can import the database containing the required variables, as shown in Figure 4.10. Subsequently, a table displaying the performance metrics for each model is generated, as illustrated in Figure 4.11.

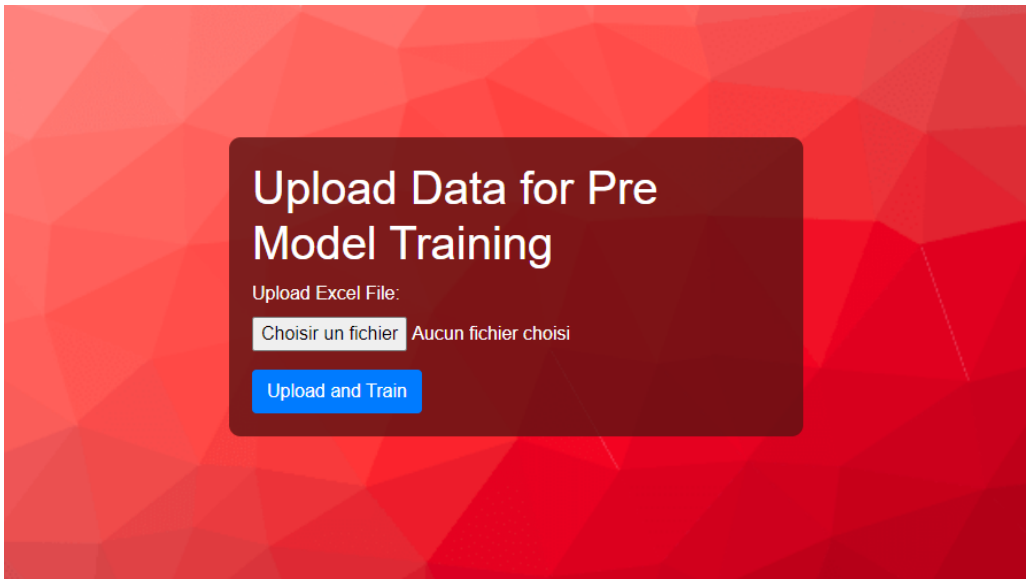

Figure 4.10: Pre-Default Approach models training page

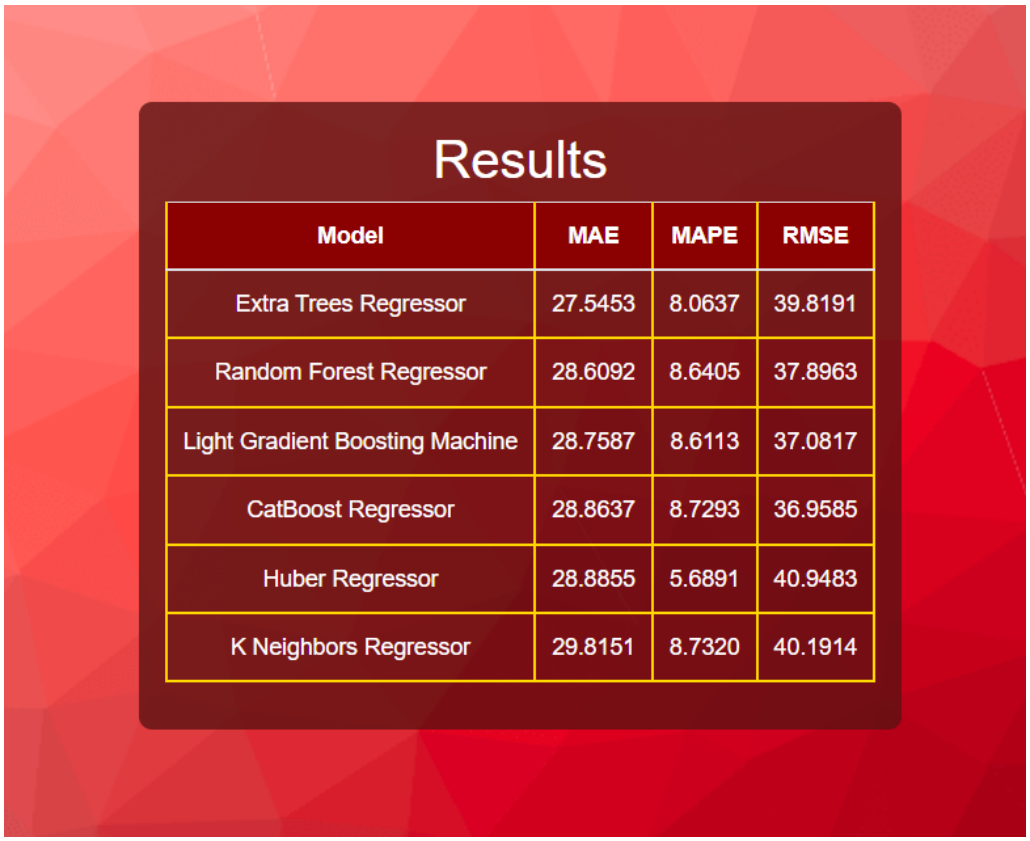

Figure 4.11: Pre-Default Approach models results page

4.3.2.2 Loss Given Default Prediction

The page enables users to input relevant data and receive predictions for Loss Given Default (LGD). Users can enter specific variables including age, profession, region, and various loan characteristics. Upon submitting the necessary inputs, the API dynamically predict the target value, as demonstrated in Figure 4.12

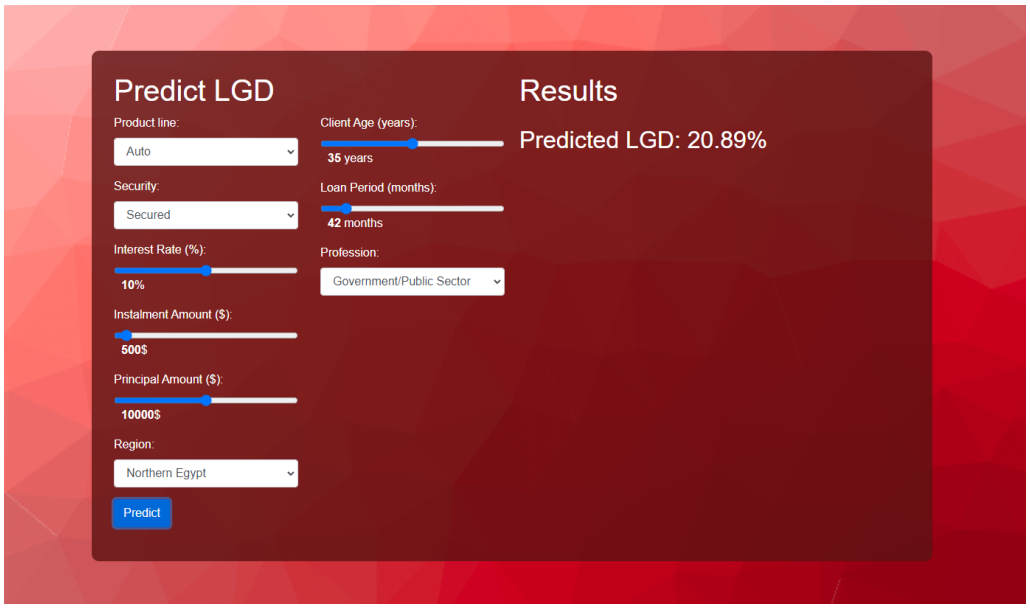

Figure 4.12: Loss Given Default prediction page

4.3.3 Post-Default Approach

4.3.3.1 Model training

This segment is dedicated to the training of machine learning models within the Post-Default framework. Users have the ability to upload a database that includes all necessary variables, as displayed in Figure 4.13. Following the data import, the system automatically produces a table that outlines performance metrics for each model, showcased in Figure 4.12.

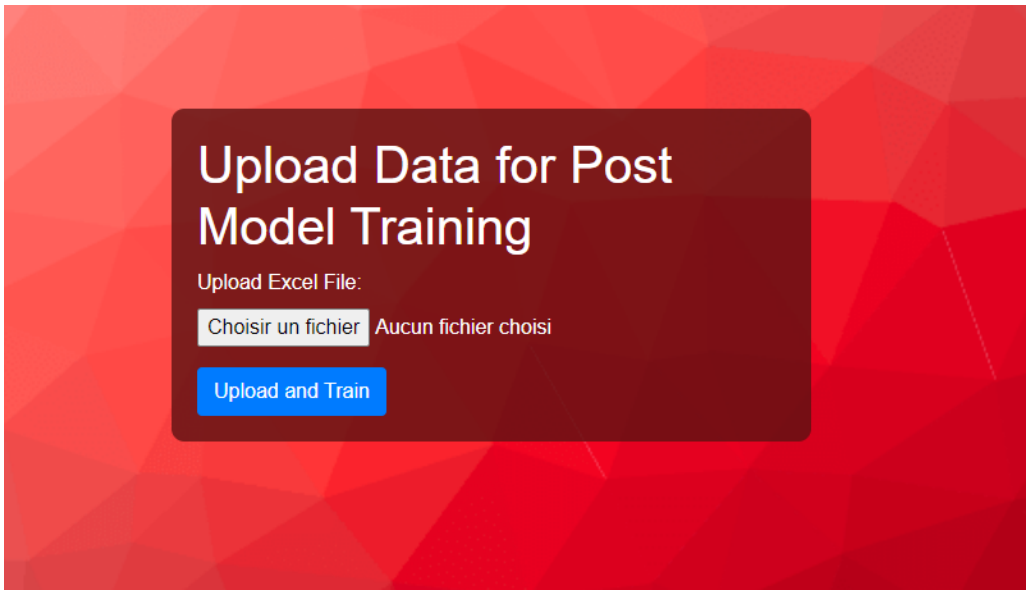

Figure 4.13: Post-Default Approach models training page

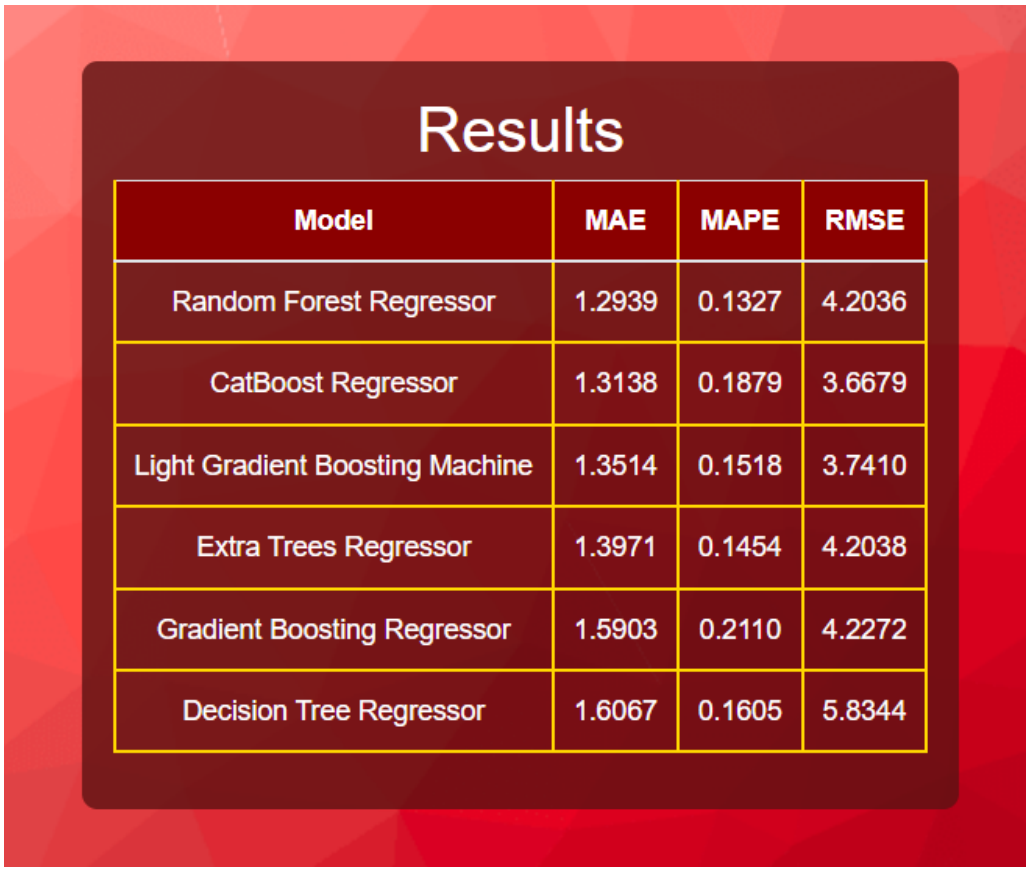

Figure 4.14: Post-Default Approach models results page

4.3.3.2 Loss Given Default Prediction

The page enables users to input relevant data and receive predictions for Loss Given Default (LGD). . Upon submitting the necessary inputs, the API dynamically predict the target value, as demonstrated in Figure 4.15.

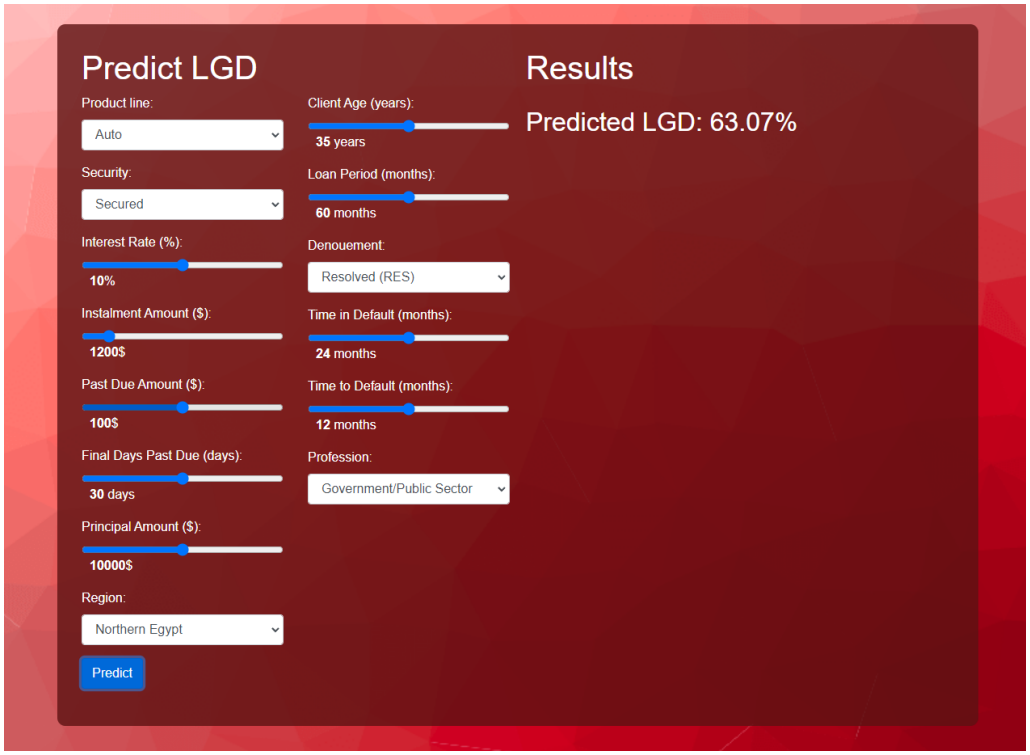

Figure 4.15: Loss Given Default prediction page

4.3.4 Dashboard

The dashboard section of the API is an intricate assembly developed using Dash and Plotly, designed to offer users an insightful overview of the database through a series of detailed visualizations and graphs as shown in figure 4.16

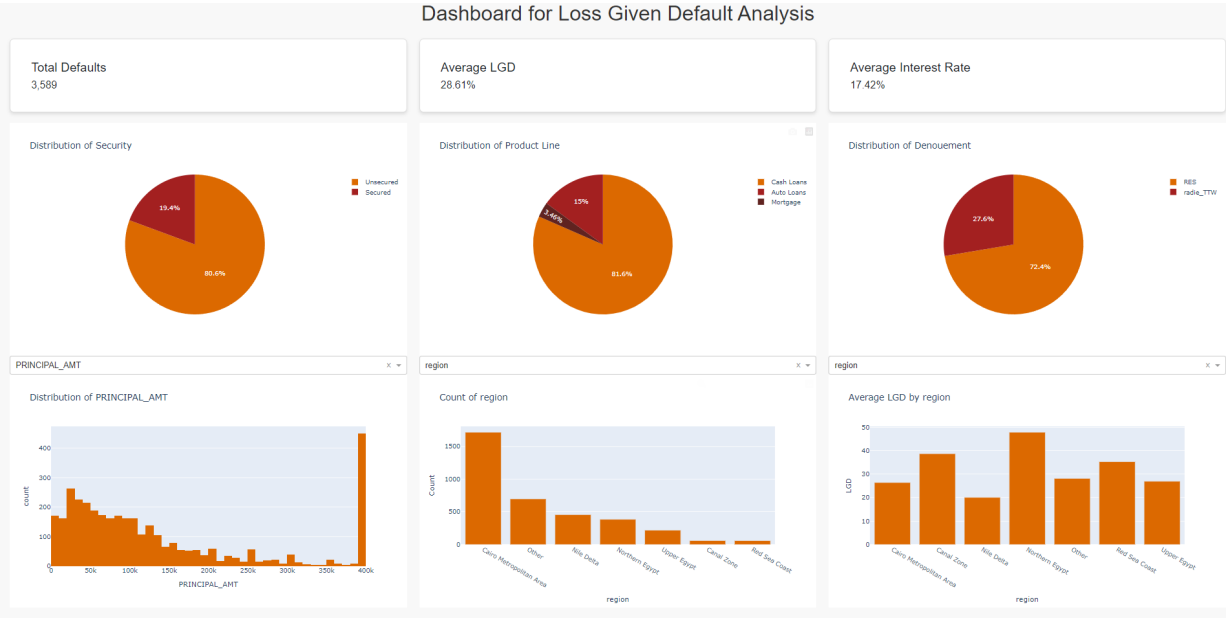

Figure 4.16: Dashboard

4.4 Conclusion

In this chapter, we presented the metrics, hyperparameters, and feature importance results for both the post-default and pre-default approaches. Also, we introduced a Flask API for the deployment task.

Conclusion

In conclusion, this project has systematically addressed the challenges and advancements in modeling Loss Given Default (LGD) for financial institutions. The necessity for accurate LGD modeling, driven by stringent regulatory requirements such as the Basel Accords and IFRS 9, was the starting point of our study. Recognizing the limitations of traditional methods, we integrated advanced machine learning (ML) techniques to improve the precision of LGD predictions by considering a wider range of predictive variables and their interactions.

A significant portion of the project was dedicated to exploratory data analysis, where we preprocessed and cleaned the data, conducted descriptive statistical analyses, and employed various methods for variable selection. This groundwork was essential for building accurate predictive models.

In the modeling phase, we implemented two distinct approaches: the post-default approach, which used all available data including behavioral changes post-default, and the pre-default approach, which focused solely on data available at the time of loan issuance. Despite the challenges posed by the limited data in the pre-default approach, we successfully identified the Random Forest model as the best performer for pre-default scenarios, while the stacked model excelled in post-default scenarios.

Finally, we deployed these models using Flask, making sure that the advanced machine learning techniques can be used in real-world financial risk management.

Annexe

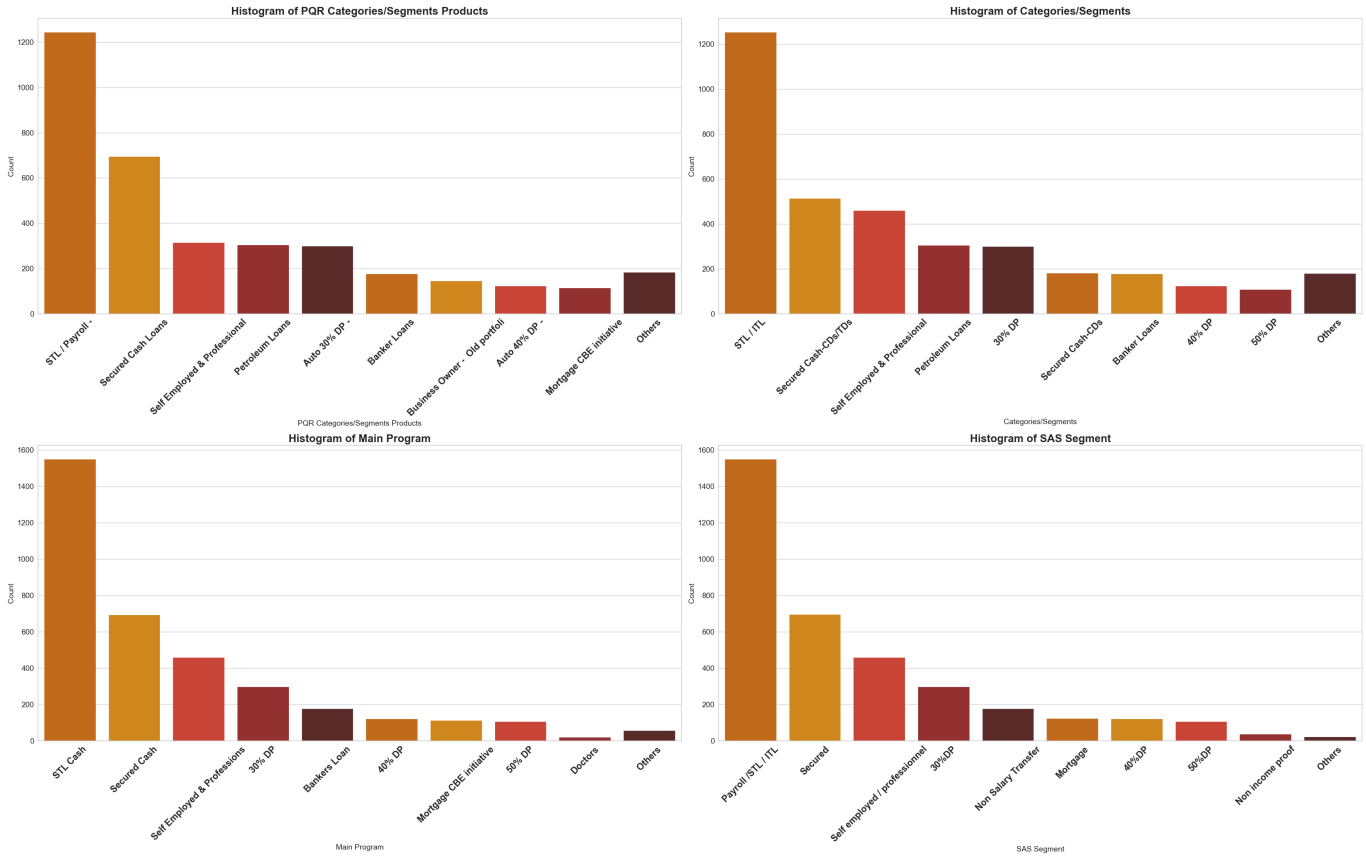

Figure 4.17: Distribution of categorical variables

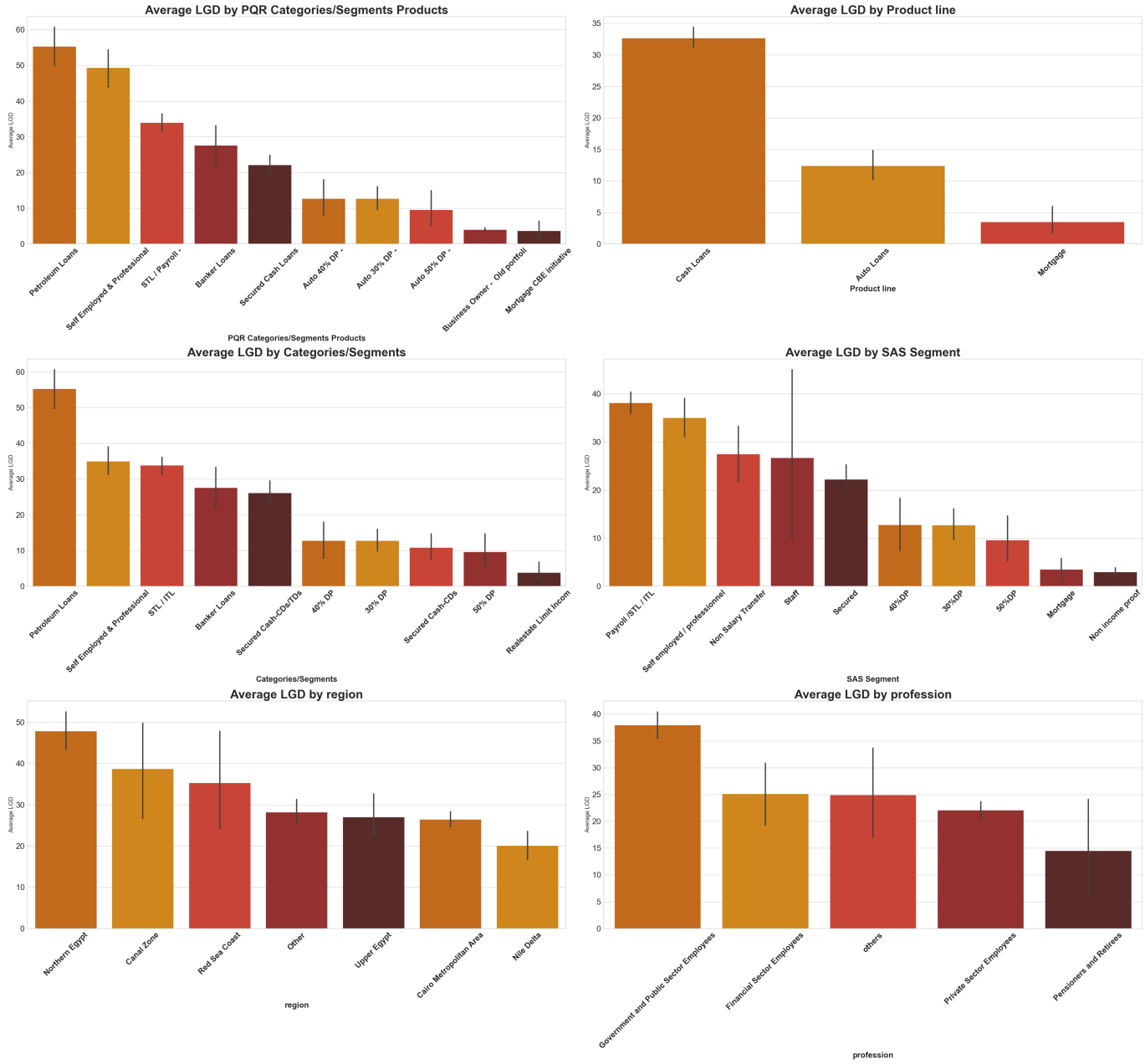

Figure 4.18: Average LGD per categorical variable

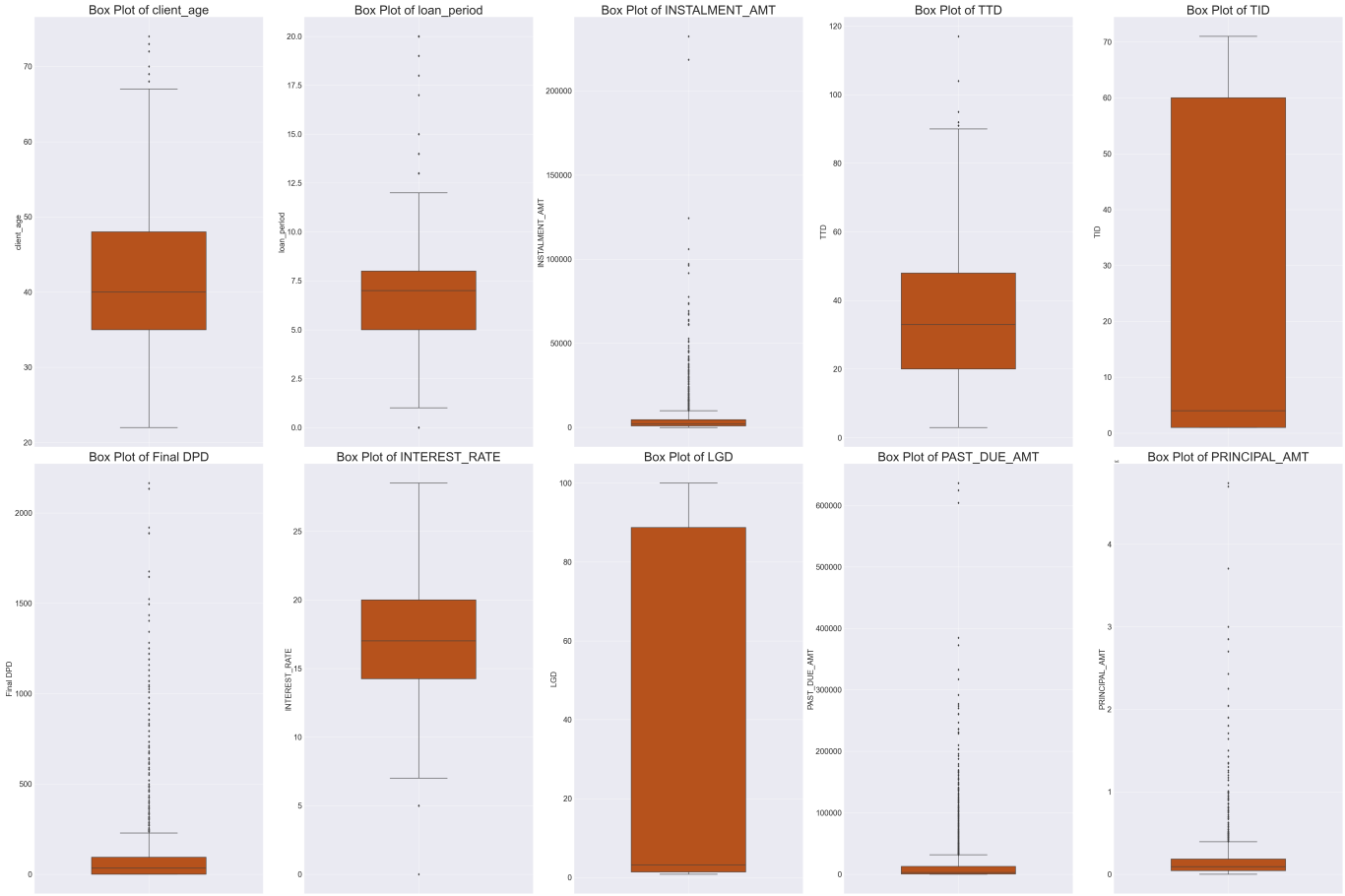

Figure 4.19: Numerical variables box plot

Bibliography

[1] Exploring basel committee's framework: A comprehensive guide. Retrieved from https://fastercapital.com/content/Exploring-Basel-Committee-s-Framework--A-Comprehensive-Guide.html#Introduction-to-Basel-Committees-Framework.

[2] History of the basel committee. https://www.bis.org/bcbs/history.htm.

[3] M. Ali. Pycaret: An open-source, low-code machine learning library in python. https://github.com/pycaret/pycaret, 2020.

[4] L. Allen and A. Saunders. A comparative analysis of current credit risk models. *Journal of Banking and Finance*, 28(1):59–117, 2004.

[5] E. Alpaydin. *Introduction to Machine Learning*. MIT Press, 4 edition, 2020.

[6] T. Bellini. *IFRS 9 and CECL Credit Risk Modelling and Validation*. London, 2018.

[7] G. Biau and E. Scornet. A random forest guided tour. *TEST*, 25(2):197–227, 2016. This comprehensive review on random forests, a popular bagging method, provides insights into the theoretical and practical aspects of the technique, suitable for a modern understanding of bagging in machine learning.

[8] M. Grinberg. *Flask Web Development: Developing Web Applications with Python*. O'Reilly Media, 2 edition, 2018.

[9] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, New York, 2009.

[10] M. J. Jacobs. Loss given default of high loan-to-value residential mortgages. *Journal of Banking and Finance*, 33(5):788–799, 2009.

[11] M. Kuhn and K. Johnson. *Applied Predictive Modeling*. Springer, New York, NY, 2013.

[12] G. Loterman, I. Brown, D. Martens, C. Mues, and B. Baesens. Benchmarking regression algorithms for loss given default modeling. *International Journal of Forecasting*, 28:161–170, 2012.

[13] P. E. McKight and J. Najab. Kruskal-wallis test. In N. J. Salkind, editor, *Encyclopedia of Research Design*, volume 1, pages 715–719. SAGE Publications, Inc., Thousand Oaks, CA, 2010.

[14] F. Pedregosa et al. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[15] M. Qi and X. Zhao. Comparison of modeling methods for loss given default. *Journal of Banking and Finance*, 35(11):2842–2855, 2011.

[16] J. Temim. The IFRS 9 impairment model and its interaction with the Basel framework. Retrieved from `https://www.moodysanalytics.com/risk-perspectives-magazine/convergence-risk-finance-accounting-cecl/spotlight-cecl/ifrs-9-impairment-model-interaction-with-the-basel-framework`. Accessed: November, 2022.

[17] K. M. Ting and I. H. Witten. Stacked generalization: when does it work? *International Journal of Pattern Recognition and Artificial Intelligence*, 22(3):445–455, 2007.

[18] D. P. Tsomocos et al. Predicting loss given default (lgd) for residential mortgage loans: A two-stage model and empirical evidence for uk bank data. *Financial Markets, Institutions & Instruments*, 13(2):123–143, 2004.

[19] Z.-H. Zhou. Ensemble methods: Foundations and algorithms. *CRC Press*, 2012. This book provides a detailed survey of ensemble methods, including boosting, offering insights into the theoretical developments and practical implementations that have shaped the field since Schapire's work.