UNIVERSITY OF THESSALY

SCHOOL OF ENGINEERING

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

Dynamic Application of Creditworthiness and Risk Management in BNPL Systems using Machine Learning and Financial Analysis

Diploma Thesis

Konstantinos Konstantinidis

Supervisor: Michael Vassilakopoulos

June 2025

UNIVERSITY OF THESSALY

SCHOOL OF ENGINEERING

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

Dynamic Application of Creditworthiness and Risk Management in BNPL Systems using Machine Learning and Financial Analysis

Diploma Thesis

Konstantinos Konstantinidis

Supervisor: Michael Vassilakopoulos

June 2025

ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

Δυναμική Εφαρμογή Πιστοληπτικής Ικανότητας και Διαχείρισης Κινδύνου σε Συστήματα BNPL με χρήση Μηχανικής Μάθησης και Χρηματοοικονομικής Ανάλυσης

Διπλωματική Εργασία

Κωνσταντίνος Κωνσταντινίδης

Επιβλέπων: Μιχαήλ Βασιλακόπουλος

Ιούνιος 2025

Approved by the Examination Committee:

Supervisor **Michael Vassilakopoulos**
Professor, Department of Electrical and Computer Engineering, University of Thessaly

Member **Aspassia Daskalopulu**
Associate Professor, Department of Electrical and Computer Engineering, University of Thessaly

Member **Panagiota Tsompanopoulou**
Associate Professor, Department of Electrical and Computer Engineering, University of Thessaly

DISCLAIMER ON ACADEMIC ETHICS AND INTELLECTUAL PROPERTY RIGHTS

«Being fully aware of the implications of copyright laws, I expressly state that this diploma thesis, as well as the electronic files and source codes developed or modified in the course of this thesis, are solely the product of my personal work and do not infringe any rights of intellectual property, personality and personal data of third parties, do not contain work / contributions of third parties for which the permission of the authors / beneficiaries is required and are not a product of partial or complete plagiarism, while the sources used are limited to the bibliographic references only and meet the rules of scientific citing. The points where I have used ideas, text, files and / or sources of other authors are clearly mentioned in the text with the appropriate citation and the relevant complete reference is included in the bibliographic references section. I also declare that the results of the work have not been used to obtain another degree. I fully, individually and personally undertake all legal and administrative consequences that may arise in the event that it is proven, in the course of time, that this thesis or part of it does not belong to me because it is a product of plagiarism».

The declarant

Konstantinos Konstantinidis

Diploma Thesis

Dynamic Application of Creditworthiness and Risk Management in BNPL Systems using Machine Learning and Financial Analysis

Konstantinos Konstantinidis

Abstract

In today's fast-paced market, businesses and customers are faced with challenging financing options like BNPL(Buy now pay later) systems. BNPL has grown six times in terms of transaction amount and volume from 2019 to 2023 and continues to grow rapidly. BNPL loans also referred to as point-of-sale installment loans are short-term loans usually with 0% interest rates and 3-4 installments. Their usual usage is for low to medium-value online purchases such as clothing, goods, and electronic devices. Despite their comfort and ease of use the underlying risks for companies, customers and providers should not be dismissed. Client-side risk is related to overspending and debt issues but this will not be the topic of this thesis due to modeling inability.

Some of the risks associated with the banking systems and specifically to loan providers are credit risk, market risk, operational risk, and liquidity risk. The effective management of these risks is the key to a more efficient and profitable banking system. This type of management consists of financial models that play a crucial role in decision-making and strategic planning. The advancements in machine learning and artificial intelligence have significantly improved the capability to make this kind of model more precise and robust.

This thesis introduces a comprehensive framework for assessing, modeling, and managing credit risk in the rapidly evolving Buy Now, Pay Later (BNPL) ecosystem. It integrates traditional risk tools—such as credit scoring, Probability of Default (PD), and Expected Credit Loss (ECL)—with advanced machine learning, behavioral analytics, and macroeconomic indicators to support real-time, accurate, and adaptive lending decisions.

The framework operates across three interconnected layers: (i) initial credit assessment using demographic, financial, and behavioral data, (ii) PD estimation through a combination of interpretable models (e.g., logistic regression) and neural networks (e.g., multilayer perceptrons), and (iii) a dynamic adjustment layer that adapts risk profiles over time based on behavioral patterns and economic conditions.

By going beyond static models, this approach supports ongoing retraining and AI-driven predictions, aligning with regulatory standards like IFRS 9. It is particularly suited to BNPL platforms, where traditional credit models struggle with thin-file borrowers and volatile behaviors. Through rigorous testing and stress scenarios, the thesis demonstrates a robust and practical system for modern credit risk management—offering valuable insights for researchers, fintech professionals, and risk practitioners alike.

Keywords:

BNPL, credit risk, fraud risk, machine learning, financial analysis, risk management,financial modeling

Διπλωματική Εργασία

Δυναμική Εφαρμογή Πιστοληπτικής Ικανότητας και Διαχείρισης Κινδύνου σε Συστήματα BNPL με χρήση Μηχανικής Μάθησης και Χρηματοοικονομικής Ανάλυσης

Κωνσταντίνος Κωνσταντινίδης

Περίληψη

Στη σημερινή ταχύρρυθμη αγορά, οι επιχειρήσεις και οι καταναλωτές αντιμετωπίζουν προκλήσεις στις επιλογές χρηματοδότησης, όπως τα συστήματα BNPL (Αγόρασε Τώρα, Πλήρωσε Αργότερα). Τα BNPL έχουν αυξηθεί έξι φορές όσον αφορά τον όγκο και την αξία συναλλαγών από το 2019 έως το 2023 και συνεχίζουν να αναπτύσσονται ραγδαία. Τα δάνεια BNPL, γνωστά και ως δάνεια δόσεων στο σημείο πώλησης, είναι βραχυπρόθεσμα δάνεια, συνήθως με 0% επιτόκιο και 3-4 δόσεις. Χρησιμοποιούνται κυρίως για αγορές χαμηλής έως μεσαίας αξίας στο διαδίκτυο, όπως ρούχα, αγαθά και ηλεκτρονικές συσκευές. Παρά την ευκολία και την άνεση που προσφέρουν, οι υποκείμενοι κίνδυνοι για τις επιχειρήσεις, τους καταναλωτές και τους παρόχους δεν πρέπει να παραβλέπονται. Ο κίνδυνος από την πλευρά των καταναλωτών σχετίζεται με την υπερβολική κατανάλωση και τη συσσώρευση χρέους, αλλά αυτό δεν αποτελεί αντικείμενο της παρούσας διπλωματικής εργασίας λόγω της δυσκολίας μοντελοποίησής του.

Ορισμένοι από τους κινδύνους που σχετίζονται με τα τραπεζικά συστήματα και πιο συγκεκ ριμένα με τους παρόχους δανείων είναι ο πιστωτικός κίνδυνος, ο κίνδυνος αγοράς, ο λειτουργι κός κίνδυνος και ο κίνδυνος ρευστότητας. Η αποτελεσματική διαχείριση αυτών των κινδύνων αποτελεί το κλειδί για ένα πιο αποδοτικό και κερδοφόρο τραπεζικό σύστημα. Αυτή η μορφή διαχείρισης περιλαμβάνει χρηματοοικονομικά μοντέλα που παίζουν κρίσιμο ρόλο στη λήψη αποφάσεων και τον στρατηγικό σχεδιασμό. Οι εξελίξεις στη μηχανική μάθηση και την τεχνητή νοημοσύνη έχουν βελτιώσει σημαντικά τη δυνατότητα δημιουργίας πιο ακριβών και αξιόπιστων μοντέλων αυτού του είδους.

Αυτή η διπλωματική εργασία παρουσιάζει ένα ολοκληρωμένο πλαίσιο για την αξιολόγηση, μοντελοποίηση και διαχείριση του πιστωτικού κινδύνου στο ταχέως εξελισσόμενο οικοσύστημα του ”Αγόρασε Τώρα, Πλήρωσε Αργότερα” (BNPL). Ενσωματώνει παραδοσιακά εργαλεία αξιολόγησης κινδύνου —όπως η πιστοληπτική βαθμολόγηση, η Πιθανότητα Αθέ-

τησης (PD) και η Αναμενόμενη Ζημία (ECL)— με προηγμένες τεχνικές μηχανικής μάθησης, ανάλυση συμπεριφοράς και μακροοικονομικούς δείκτες, για την υποστήριξη ακριβών, σε πραγματικό χρόνο και προσαρμοζόμενων αποφάσεων δανεισμού.

Το προτεινόμενο πλαίσιο λειτουργεί σε τρία αλληλένδετα επίπεδα: (i) αρχική αξιολόγηση πιστοληπτικής ικανότητας βάσει δημογραφικών, οικονομικών και συμπεριφορικών στοιχείων, (ii) εκτίμηση PD μέσω συνδυασμού ερμηνεύσιμων στατιστικών μοντέλων (π.χ. λογιστική παλινδρόμηση) και νευρωνικών δικτύων (π.χ. πολυεπίπεδων perceptrons), και (iii) δυναμική προσαρμογή βάσει αγοραστικών μοτίβων, συμπεριφοράς πληρωμών και μακροοικονομικών δεδομένων.

Ξεπερνώντας τα στατικά μοντέλα, η προσέγγιση υποστηρίζει συνεχή επανακατάρτιση και προβλέψεις μέσω AI, ευθυγραμμισμένες με πρότυπα όπως το IFRS 9. Είναι ιδιαίτερα κατάλληλη για πλατφόρμες BNPL, όπου τα παραδοσιακά μοντέλα δυσκολεύονται να αξιολογήσουν πελάτες με περιορισμένα ιστορικά δεδομένα και ασταθή συμπεριφορά. Μέσα από εκτενή έλεγχο και δοκιμές αντοχής, αποδεικνύεται η αξιοπιστία και πρακτικότητα του συστήματος, καθιστώντας το πολύτιμο εργαλείο για ερευνητές, επαγγελματίες fintech και διαχειριστές κινδύνου που επιδιώκουν να χτίσουν σύγχρονες υποδομές πιστωτικού κινδύνου για το ψηφιακό μέλλον.

Λέξεις-κλειδιά:

BNPL,κίνδυνος αγόρας , πιστωτικός κίνδυνος, τραπεζικό σύστημα, πιθανότητα απάτης, οικονομικά μοντέλα

Table of contents

List of figures

List of tables

Abbreviations

BNPL	Buy Now, Pay Later
PD	Probability of Default
ECL	Expected Credit Loss
EAD	Exposure at Default
LGD	Loss Given Default
RAR	Risk-Adjusted Return
DPD	Days Past Due
SMOTE	Synthetic Minority Over-sampling Technique
MLP	Multilayer Perceptron
EDA	Exploratory Data Analysis
IFRS	International Financial Reporting Standards
GDP	Gross Domestic Product
CPI	Consumer Price Index
DTI	Debt-to-Income
ROC AUC	Receiver Operating Characteristic Area Under Curve
XAI	Explainable Artificial Intelligence

Chapter 1

Introduction

In recent years, the rise of new credit models and digital-first lending experiences has brought rapid transformations to the financial sector(fintech). Among these, **Buy Now, Pay Later (BNPL)** has emerged as a disruptive force. Companies such as Klarna, Affirm, and Afterpay have reshaped the consumer digital-lending environment. Behavioral analytics, seamless integrations, and data-driven risk assessments are the key components of this companies system implementation.

The increasing adoption of this type of financing models in e-commerce presents many challenges in credit risk assessment. The fact that traditional credit scoring models often fail to capture the financial behavior of customers with zero or low financial history, makes innovative approaches (Abbott, 1991) necessary. Traditional credit scoring systems are not implemented in a way that accurately assess risk in short-term, small-value loans. Furthermore, the instant approval model of BNPL system credit requests requires **real-time decision-making**, which introduces the need for robust, flexible, and adaptive credit risk frameworks.

This thesis addresses this problem by proposing a **data-driven BNPL credit risk management system**, capable of estimating the initial purchasing capacity of a user and the probability of his loans defaulting. These estimations will be used to dynamically accept or decline the customer's credit requests and also calculate risk metrics like **ECL(expected credit loss)** and **RaR(Risk-adjusted return)** that will be explained further in this thesis.

1.1 Related Work

Traditional credit risk assessment models in banking have long relied on structured financial data, using tools such as credit scoring systems, logistic regression, and expert-based scorecards. These approaches, grounded in parametric statistical theory, are widely implemented across retail and corporate lending sectors. Financial institutions often utilize hard credit pulls, bureau reports, and fixed rule-based systems to evaluate borrower risk. Such methods are well-established but tend to underperform in emerging credit environments like BNPL, where real-time decision-making and fragmented data are common.

In recent years, there has been a growing interest in machine learning (ML) methods to enhance risk prediction accuracy. Techniques such as decision trees, random forests, and neural networks have been introduced to capture non-linear relationships and complex behavioral patterns. Institutions like Goldman Sachs and J.P. Morgan have deployed hybrid risk systems combining macroeconomic indicators and alternative data sources, enabling dynamic credit scoring and PD (Probability of Default) estimation. However, many of these models are developed for long-term loans and rely heavily on high-quality historical data.

When it comes to BNPL systems, the literature is still developing. Most current applications adapt conventional credit scoring or use proprietary “purchase power” metrics—like Klarna and Affirm—which are not publicly disclosed and often act as black-box systems. These lack transparency and pose challenges for fair credit assessment, especially for thin-file borrowers.

What sets this thesis apart is its integrated and modular framework tailored specifically for BNPL environments. Unlike most previous research which either focuses on traditional credit or purely technical ML models, this work blends classical credit risk principles (PD, ECL) with dynamic behavioral adjustments, macroeconomic indicators, and hybrid ML architectures. It uses interpretable models (e.g., logistic regression) alongside more flexible ones (e.g., multilayer perceptrons), ensuring both performance and transparency. Additionally, this framework introduces a retrainable structure and full-stack implementation, bridging the gap between theoretical modeling and practical deployment in real-time digital lending systems.

1.2 Objectives

The main objectives of this thesis are as follows:

- To design a framework for **initial credit evaluation** of users based on demographic and financial characteristics. This initial check will also be enriched with country-specific macroeconomic indicators.
- To develop predictive models (**logistic regression**, **MLP neural networks**) that estimate the **probability of default (PD)**.
- To integrate **behavioral analysis** for evaluating creditworthiness based on user behavior.
- To implement key risk management metrics such as **Expected Credit Loss (ECL) and Risk-adjusted return**, that align with modern accounting standards (e.g., IFRS 9).
- All components will be combined into a full-stack application that integrates data science, credit risk theory, and real-world deployment logic.

1.3 Methodology Overview

This project is developed as a **modular credit risk engine**. The proposed methodology consists of the following stages:

1. **Initial Credit Assessment**: The purpose of the first part of this Thesis is to develop a model that classifies recently registered users into limit categories and then apply macroeconomic-indicating changes to formulate the initial purchase power).
2. **PD Estimation**: A hybrid machine learning model that combines logistic regression (with calibration) and MLP models to predict the likelihood of default for each credit request.
3. **Behavioral Analysis**: A framework that detects loan repayment behavior and updates user creditworthiness accordingly.
4. **Risk Metrics**: A Statistics model that calculates metrics like ECL using PD × EAD × LGD and performs stress testing, as defined in regulatory frameworks.

1.4 Contribution and Innovation

This thesis introduces a **hybrid credit risk assessment framework** specifically made for BNPL environments. The core innovations include:

- A scalable **macro-adjusted purchase power model** based on country-specific economic indicators.
- A hybrid-model structure combining **interpretable** (logistic regression) and **non-linear** (MLP with autoencoder) models for efficient PD prediction.
- Integration of **IFRS 9-inspired** metrics into a fintech BNPL context.
- A complete **end-to-end deployment-ready** application that bridges the gap between research and implementation.

1.5 Structure of the Thesis

The remainder of this document is organized as follows:

- **Chapter 2** reviews the BNPL environment, drivers of usage, and statistics.
- **Chapter 3** presents the theoretical background of risks in banking systems and dives deeper into the BNPL system assessment framework.
- In **Chapters 4** the initial credit assessment model and the purchase power formula are developed.
- **Chapter 5** presents the database schema and the integration with the application environment.
- In **Chapter 6** the full-stack application development will be analyzed.
- **Chapter 7** presents the behavioral analysis and the purchase power changes based on user payment behavior.
- In**Chapter 8** the probability of default prediction models will be analyzed and evaluated.

- **Chapter 8** represents the retraining process of the MLP model.
- **Chapter 9** combines the predicted probabilities and application statistics to calculate risk metrics.
- In **Chapter 10** the total application performance will be tested and the results will be rigorously analyzed.
- **Chapter 11** concludes with the future work also potential improvements and extensions will be discussed.

Chapter 2

BNPL and Statistics

Once a niche payment option, Buy Now, Pay Later (BNPL) has quickly evolved into a mainstream financial tool, reshaping how consumers manage short-term spending. Driven by digital convenience and changing credit preferences, BNPL adoption is rising globally—especially among younger generations and underserved borrowers. Understanding how these systems work, who uses them, and what drives their growth reveals deeper insights into the future of consumer finance.

2.1 How it Works

Buy Now, Pay Later (BNPL) is a type of short-term loan that lets shoppers pay for products in 3-4 installments spread over a set period of time. These services are typically used for minor, although expensive purchases like electronic devices or luxury clothing. BNPL loans work like personal loans, mortgages, or car loans. The seller receives the entire purchase amount from the loan provider upfront then you pay off the amount to the BNPL service in smaller installments.

A small down payment, generally around 25% of the purchase amount, may be required at once and then you pay off the rest in the coming weeks or months. For example, if the purchase costs $1,000, you may have to pay $250 immediately during the checkout, then you can pay the remaining $750 in 3 $250 installments spread over five or 10 weeks. Usually, the payment of the installments is an automated procedure made by the providers to decrease the number of defaults **(Consumer Financial Protection Bureau (CFPB))**. Unlike other types

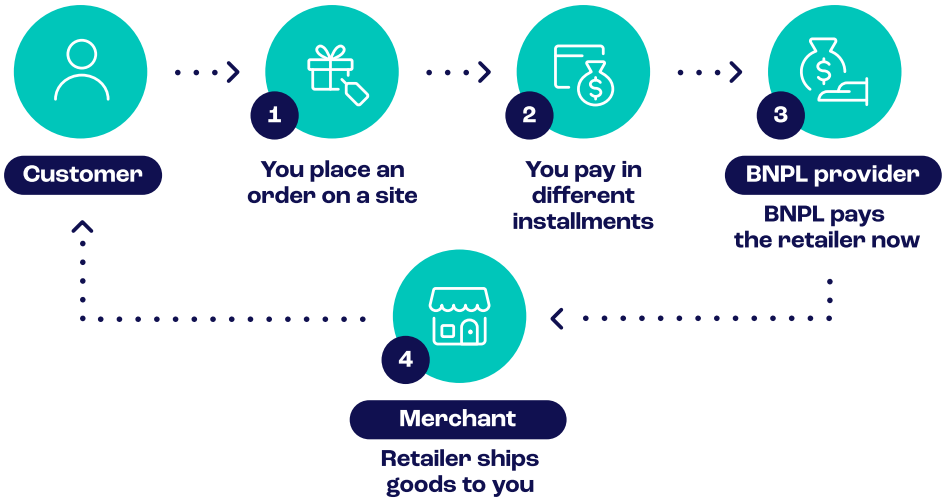

Figure 2.1: BNPL pipeline.

of loans, BNPL loans are typically interest-free and rarely carry other service fees, making them suitable for people on a tight budget. The convenience of this environment is the reason that the BNPL has dramatically changed the world of digital purchases.

2.2 Statistics

2.2.1 Market

The global BNPL payment market is expected to grow by 13.7% annually to reach $560.1 billion in 2025. Global BNPL market experienced dramatic growth during 2021-2024, achieving a **CAGR(Compound annual growth rate)** of 21.7%. This upward trend is expected to continue, with the market forecast to grow at a CAGR of 10.2% during 2025-2030. By the end of 2030, the BNPL sector is projected to expand from its 2024 value of USD 492.8 billion to approximately USD 911.8 billion.

2.2.2 Users

The users of BNPL systems commonly belong to the age bracket of 18-34 years old, and they consist the 65% of total customers. This trend highlights the preference of the user for more convenient payment options. Furthermore, 42% of the the customers use this option at least once a month, the fact that shows the increasing dependence on the system on daily

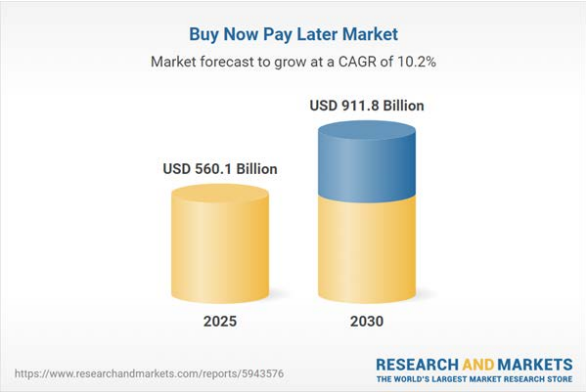

Figure 2.2: BNPL market share

purchases. According to Exploding Topics (2023), 39% of Gen Z (born 1997–2012) planned to use BNPL, showing dominance among younger demographics. In contrast, only 21% of Gen X (born 1965–1980) and 9% of Baby Boomers (born 1946-1964) expressed the intention to use these types of services. Budget consciousness, flexibility, and interest-free installments are some of the reasons that BNPL usage is more common in younger generations. These statistics also showcase that new generations find alternatives to credit cards and traditional banking.

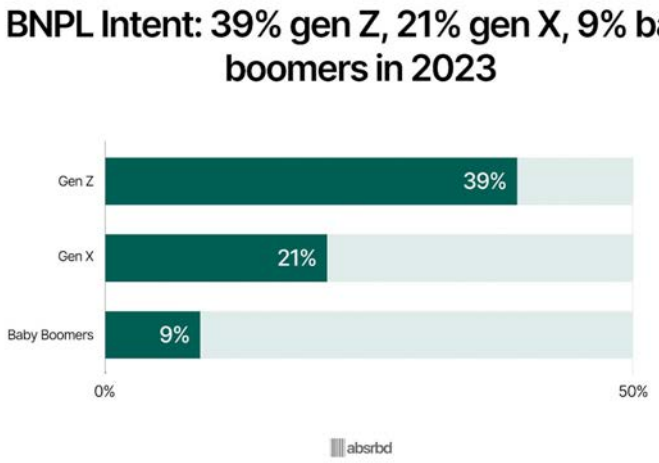

Figure 2.3: Generational disparity

A more comprehensive survey from **Numerator** for the correlation of BNPL users and their demographic characteristics follows the table below: The data reveals significant patterns in BNPL adoption across age groups, income levels, and ethnicities. While Gen Z (5%) and Millennials (27%) show lower representation among current BNPL users compared to

Grouping	Demographic	BNPL Users	Non-Users	Index
Age (Generation)	Gen Z [> 1996]	5%	2%	212
	Millennials [1982-1995]	27%	18%	148
	Gen X [1965-1981]	33%	31%	105
	Boomers+ [< 1965]	36%	48%	73
Income	- $20k	11%	7%	172
	$20k-40k	15%	12%	120
	$40k-60k	15%	14%	104
	$60k-80k	13%	12%	104
	$80k-100k	11%	11%	102
	$100k-125k	8%	13%	64
	$125k +	27%	31%	87
Ethnicity	White/Caucasian	61%	81%	76
	Black or African American	15%	7%	208
	Hispanic/Latino	15%	7%	203
	Asian	7%	4%	180
	Other	2%	1%	202

Figure 2.4: Age-Income-Ethnicity

Gen X (33%) and Baby Boomers+ (36%), their intent to use BNPL (as seen in previous studies) suggests growing future adoption. This paradox may indicate that younger generations are still entering the financial system, while older groups use BNPL for larger purchases.

Income disparities are equally notable: lower-income earners (under 40k, exhibit higher adoption rates of the BNPL systems due to the inability to access credit in the traditional banking sector.

Ethnicity trends highlight disproportionate adoption among African American (15% users vs. 7% non-users) and Hispanic/Latino (15% vs. 7%) communities, both scoring an Index >20 that indicates overrepresentation. This may come from systemic barriers to traditional credit. Meanwhile, White/Caucasian users (61%) dominate in absolute numbers but are underrepresented relative to their population(Index: 7.6).

2.3 Drivers

Recent research by the **BIS(Bank for international settlements)** underscores the variation in the use of BNLP by country. The panel regression analysis examines data from 25 countries and for years 2016-2022 to identify the drivers usage of BNPL per capita. The result

of this analysis is characterized by the equation below:

$$\begin{aligned}\ln(\text{BNPL}_{it}) =& \alpha \ln(\text{Inflation}_{it}) + \beta \ln(\text{Ecommerce}_{it}) \\ &+ \gamma \text{Regulation}_{it} + \delta \ln(\text{BankCostIncome}_{it}) \\ &+ \theta \text{HouseholdDebt}_{it} + \text{Controls}_{it} \\ &+ \text{FE} + \varepsilon_{it}\end{aligned} \tag{2.1}$$

where BNPL_{it} is the number of BNPL app users per capita in country i and quarter t.

(2.2)

Key variables:

- Inflation:high inflation environments (e.g., 2021–2022) increase BNPL demand as households want to smooth consumption amid eroding real incomes.
- E-Commerce Penetration : BIS data shows BNPL usage is 3× higher in countries where e-commerce exceeds 15% of retail sales (e.g., UK, South Korea).
- Regulatory Environment:Uses two proxies from World Bank datasets:
  1. Banking regulation stringency index (Barba Navaretti et al., 2017)
  2. Government effectiveness (consumer protection capacity)

  BIS notes a U-shaped relationship: BNPL grows in both laissez-faire regimes (low barriers) and high-trust regimes (strong consumer safeguards).
- Financial System Controls:
  1. Bank cost-to-income ratio: Measures traditional credit inefficiency (BNPL substitutes where banks are costly).
  2. Household debt/GDP: Tests whether BNPL complements or crowds out existing debt (BIS 2023 finds mixed evidence).
- The model includes Controls_{it} to account for observable country-quarter specific factors (e.g., GDP growth, unemployment rates), fixed effects (FE) to captivate time-invariant unsupervised heterogeneity across countries and periods, and an error term ε_{it} capturing shocks and measurement errors. This specification ensures our estimates

of inflation (α), e-commerce penetration (β), and regulatory effects (γ) are robust to variable bias while isolating the true relationship between these drivers and BNPL adoption.

Insights:

- Stricter banking regulation may unintentionally spur BNPL growth if it restricts traditional credit access (BIS Bulletin No. 68, 2023).
- As the BIS emphasizes, cross-country BNPL metrics remain inconsistent, urging harmonization (e.g., via FSB or IMF standards).
- Countries with inflation >5% (2021–22) saw 22% faster BNPL adoption.
- 60% of BNPL users in emerging markets lack credit cards (vs. 25% in advanced economies).

Chapter 3

Theoretical Background

This chapter provides a theoretical foundation for the key concepts and methodologies that underpin the analysis conducted in this thesis. It explores the principal risk categories in banking, the mechanics of credit risk assessment, and the evolution of predictive modeling techniques. By establishing this background, the chapter sets the stage for applying these principles to the context of digital lending and Buy Now, Pay Later services.

3.1 Risks in Banking Systems

Banks and financial institutions are exposed to a series of risks that can seriously affect their economic stability, revenue, and their total operation. Based on international standards of the Basel Committee on Banking Supervision(BCBS) these risks are often classified into 4 major categories: credit risk, market risk, liquidity risk, and operational risk. The understanding and effective management of this risk is a mandatory premise for the sustainability of the banking systems, especially in the era of digital transformation and the rise of new types of lending like Buy Now, Pay Later(BNPL).

3.1.1 Credit Risk

Credit risk is considered the risk that needs the most attention among all and is defined as the probability that the borrower will not take care of his obligations according to the pre-negotiated circumstances(Bank for International Settlements,2020,). At the core of credit risk lies the creditworthiness of the customers which can be expressed as :

- Defaults

- Credit deterioration
- Unforeseen loss due to concentrated risk

In the prospect of digital lending (BNPL) credit risk intensifies significantly due to the following factors :

- Lack of credit history for a considerable percentage of customers especially for younger ones or individuals with limited access to traditional banking system
- In contrast with traditional banking there is a lack of personal evaluation and assessment.

3.1.2 Market Risk

Market risks refer to the damages that can occur from dramatic changes in the prices of financial instruments or other market conditions like interest rates, exchange rates, and title prices (European Banking Authority,2021). The main forms are :

- Interest rate risk
- Foreign exchange risk
- Equity, commodity price risk

In the case of the BNPL system the market risk is restricted due to short-term credits with stable cost . Although, in general terms, banks that offer BNPL services can face consequences for interest rate fluctuations or macroeconomic changes that can harm the creditability of the customers.

3.1.3 Liquidity Risk

Liquidity risk refers to the bank's inability to cover its short-term expenses without serious losses(Basel III, 2010). Distinguished in :

- Funding liquidity risk (difficulty in capital collection)
- Market liquidity risk (difficulty in selling assets without loss)

In terms of BNPL, this risk takes place in situations of economic recession where :

- Number of delayed payments can significantly rise.
- Dependence on short-term funding sources can cause liquidity problems.

3.1.4 Operational Risk

Based on Basel II, operational risk is defined as the risk of damage from inadequate or failed internal processes, personnel, systems, or external events(Basel Committee,2004). The most usual cases are :

- Technical system failures.
- Cybersecurity(Data breaches ,fraud).
- Bad implementation of credit scoring models.

The assenting trend of digital lending and BNPL systems has introduced new dimensions in the following risks:

- Credit risk maximizes due to the lack of traditional credit assessment models.
- Liquidity risk becomes sharp in cases of financial instability.
- Operational risk strengthened from the dependence on advanced digital systems.

3.2 Credit Risk in Modern Banking

Credit risk as we mentioned before is considered the most important and timeless risk in the banking industry, with special meaning in the era of digital transformation of financial institutions. Specifically refers to the risk of the borrower not paying his obligations whether it comes from disability or reluctance. The theoretical analysis of credit risk is based on its dual nature: on one hand, as a risk at the borrower’s level, on the other hand as a systemic risk at the economic level.

According to classical financial theory , credit risk is manifested on three dimensions. Firstly , through the probability of default(PD) .Secondly with the exposure at default when the event occurs(Exposure at default) and lastly with the loss in case of a default(Loss given default) that corresponds to the percentage of the credit amount that cannot be obtained back.

Within the framework of the European Union, the European Central Bank (ECB) and the European Banking Authority (EBA) have developed further instructions that aim to regulate the management of credit risk, with particular emphasis on the transparency and robustness of internal assessment models.It is mandatory to mention that this framework responds to the advancement of the digital work and the rise of new types of lending like BNPL systems.

This theoritical framework comprise the base of the specific measurement techniques and management of credit risk, that will be further analyzed in the prospect of the following sections. This thesis will maintain it's focus in trying to regulate credit risk through the PD(Probability of default) and Expected credit losses (ECL).

3.3 ECL and Probability of Default

The concept of Expected Credit Losses (ECL) constitutes a cornerstone of modern credit risk management, particularly following the adoption of the IFRS 9 accounting standard.In contrast with previous standards (IAS 39) where losses were only calculated when they had taken place, IFRS 9 propose the expected loss model where losses are calculated or more accurately predicted before they occur. The ECL is calculated with the following fundamental equation :

$$ECL = PD \times EAD \times LGD \tag{3.1}$$

where all components have been previously discussed.

PD is considered the most uncertain and difficult-to-predict feature. Although the super majority of the models that have been developed from financial institutions have the probability of default as their target feature. Reliable prediction of it requires specialized quantitative models that incorporate both macroeconomic data and borrower-specific microeconomic characteristics, such as income, credit behavior, and employment status. Particularly in digital lending environments (e.g., BNPL systems), the lack of historical credit data necessitates the use of alternative data sources and machine learning techniques to generate accurate PD estimates.

An approach to better analyze credit risk is by grading the loans with the borrower's creditworthiness. A simplified way to do that lies in the table below:

This classification is not merely an accounting exercise but directly impacts:

- Loan pricing (e.g., higher interest rates for subprime borrowers),

Table 3.1: Loan Classification

PD	Creditworthiness	Loan classification
<2%	High creditworthiness	Prime
2-10%	Medium creditworthiness	Near-prime
>10%	Limited or unreliable history	Subprime

- Loss provisioning (Expected Credit Loss calculations under IFRS 9),
- Regulatory capital adequacy requirements (as per Basel III frameworks).

3.4 Predicting it

The development and implementation of predictive models for estimating the Probability of Default (PD) represent one of the most critical domains in modern credit risk management. Within this framework, leading financial institutions have developed specialized methodologies that combine traditional econometric analysis with state-of-the-art artificial intelligence (AI) and machine learning (ML) techniques. Modern credit risk management theory is grounded in two principal schools of thought. On one hand the parametric models such as the classic Merton (1974) model, which applies options theory principles to estimate the probability of default (PD) of corporate borrowers. On the other hand the non-parametric models that use machine learning algorithms that are particularly effective in processing large volumes of unstructured data.

Logistic regression remains the cornerstone of banking practice, particularly for retail portfolios, due to its interpretable results and relative simplicity. However, in recent years, more complex models like decision trees (CART), random forests, and neural networks have gained traction, offering higher predictive accuracy - especially in cases where variable relationships are nonlinear.

In practical applications, several innovative approaches developed by leading global financial institutions deserve mention. Goldman Sachs, through its 'Marquee PD' platform, has implemented a multivariate system incorporating 57 distinct economic indicators, enabling real-time PD estimation updates for corporate clients that reflect current macroeconomic scenarios.

J.P. Morgan, for its part, has developed the 'Athena' credit risk system—a hybrid model integrating three distinct modeling layers capable of processing semi-structured data from diverse sources. This system performs automated credit scoring for over 15,000 corporate entities daily, providing credit analysts with a robust decision-making tool.

Despite the significant benefits offered by modern PD predictive models, their implementation faces several challenges. Stringent regulatory requirements for model transparency and verifiability often conflict with the 'black box' nature characterizing many advanced machine learning models. Additionally, the need for high-quality, comprehensive data remains a major barrier, particularly for smaller financial institutions.

3.5 Credit Score

Professionals in the field of economics and risk management in their effort to enumerate a person's creditworthiness came up with the metric(Credit Score). This systematic approach enabled lenders to evaluate the probability of default (PD) through standardized metrics, facilitating data-driven decision-making across the credit life cycle. Lenders use credit scores to determine who qualifies for a loan, at what interest rate, and what credit limits. Lenders also use credit scores to determine which customers are likely to bring in the most revenue.

FICO® Score system (ranging from 300 to 850) stands as the predominant model in the United States. This scoring spectrum categorizes borrowers into distinct risk tiers:

- scores below 580 indicate poor creditworthiness,
- 580-669 represent fair credit,
- 670-739 denote good credit,
- 740 and above reflect excellent credit standing.

Similar systems have been developed,including VantageScore in the U.S., CIBIL in India, and SCHUFA in Germany, each adapting to regional financial behaviors and regulatory requirements.

The calculation of credit scores incorporates multiple weighted factors that collectively make an approximation of the borrower's reliability. Payment history emerges as the most influential component, contributing approximately 35% to the total score, as it demonstrates

consistent repayment behavior. Credit utilization follows closely at 30%, measuring the ratio of outstanding debt to available credit limits. The remaining score composition considers credit history length (15%), diversity of credit products (10%), and frequency of recent credit inquiries (10%). This multi-factorial approach ensures robust risk assessment while allowing for nuanced differentiation among borrowers.

Methodologically, credit scoring has progressed through distinct generations of modeling techniques. Traditional approaches continue to dominate retail banking applications, with logistic regression maintaining prominence due to its interoperability and regulatory compliance. This method estimates default probability through the logistic function, where predictor variables (income, existing debt, etc.) are weighted by statistically derived coefficients. Complementing this, scorecard systems implement weighted-factor models that translate borrower attributes into point allocations, which are then aggregated into final scores.

Data infrastructure forms the foundation of effective credit scoring systems. Traditional credit bureau data, encompassing repayment histories and outstanding obligations, remains the primary information source. However, the emergence of alternative data - including utility payment records, rental histories, and even psychometric indicators - has expanded scoring possibilities for previously unscorable populations. Macroeconomic factors such as GDP growth and unemployment rates provide contextual adjustments, though their incorporation requires careful handling to avoid cyclical biases.

3.6 Credit Inquiries

A credit inquiry is a request for credit report information from a credit bureau. Credit inquiries are typically made by financial institutions to help them determine whether to approve you for credit. However, credit inquiries can be made for many other reasons as well, such as if you want to check your credit. The two main types of credit inquiries are hard inquiries and soft inquiries.

3.6.1 Soft Pull

Soft inquiries occur when a credit report is accessed for informational or non-lending purposes. These checks do not result from active credit applications and thus have no adverse effect on credit scores. Common scenarios include:

- Personal credit monitoring,
- Pre-approved credit offers,
- Employment background checks,
- Account maintenance reviews.

A key characteristic of soft inquiries is that they remain visible only to the consumer on their credit report and do not influence credit scoring algorithms. This makes them a risk-free method for individuals to track their credit health or for lenders to perform preliminary assessments.

3.6.2 Hard Pull

In contrast, hard inquiries are generated when a lender evaluates a borrower's credit report as part of a formal credit application. These occur in situations such as:

- New credit applications (credit cards, personal loans, mortgages),
- Auto loan financing requests,
- Utility service contracts in regions where providers check creditworthiness.

3.6.3 About BNPL

The specific type of inquiries that BNPL providers use when credit requests take place have not been public, our thesis will use Klarna's customer service-specific statements to make a profile of their policy.**"When a credit check is performed, we verify your identity using the details you provided and we look at information from your credit report to understand your financial behavior and evaluate your creditworthiness"**. Another statement says **" Unlike traditional credit card applications that require a hard credit pull when you apply for a card, the Klarna Card doesn't perform a hard credit inquiry at any point during the application process. This means that your credit scores won't be affected if you get rejected for the card"**. Recent announcements although confirm that Klarna have tightened up their policy saying **"While using Klarna won't necessarily have a negative effect on credit score, missing Klarna repayments will. Klarna will report late**

or missed payments to the credit reference agency, and these will stay on your credit file for six years".

3.7 Purchase Power

Although the information that Klarna uses to estimate an individual's creditworthiness is not disclosed, it is public that they use models that incorporate behavioral analysis and financial information to make this happen. Klarna to approve or reject a credit request uses a metric called purchase power, which we assume is initially made by credit and financial information and then been shaped by the client's behavior within the app. In our thesis, by using machine learning techniques we will try to make the initial credit check of the newly registered customer using a dataset from a credit card application and then use business logic algorithms to dynamically change his creditworthiness. We would like Klarna to have the purchase power as the main metric of our risk management policy. The meaning of adjusting a number that has currency values in a person's creditworthiness is captured in the statement that **"Purchase power is the estimation of the amount of money you can spend on the app with really low risk of defaulting and without losing your financial stability** ". This metric although is not calculated only from the person's creditworthiness. Below are some of the main factors that affect a user's purchase power:

1. Financial information
    - Credit history and score. Users with high scores are given higher limits
    - Income and financial stability. High income and employment status strongly affect purchase power.
    - Debt. Debt alone or more commonly debt to income ratio(DTI) is really critical to a user's ability to repay obligations.
2. Behavioral factors
    - Users who pay their installments with no latency are assigned higher limits.
    - Users that are frequently late to their obligations are receiving lower limits regardless of other information.
    - New users usually have low limits.

3. Regulatory and Exogenous Factors

   - Regulatory requirements. In some countries (e.g., Germany, and Australia), BNPL services are subject to credit checks, which may limit purchasing power.
   - Macroeconomic factors Inflation, GDP, and unemployment rates are the main macroeconomic factors that affect:

     (α') User's ability to repay.

     (β') The policy of BNPL companies for granting credits.

Chapter 4

Initial Credit Check

As we introduced in our previous chapters, the first step of the credit assessment of users is to estimate their initial purchase power. It is often the most complicated part of the process as most of the BNPL system's clients have limited banking history. In this chapter we will try to compute this value through machine learning techniques and algorithms.

4.1 Dataset

Due to the lack of data on BNPL systems and the absence of knowledge regarding how these systems perform the initial credit check, we will use a dataset related to credit card applications, which has the credit card limit as the target feature. Based on this, we will perform the initial classification of users. We will perform a "soft credit check" so our challenge is to try to predict the initial creditworthiness of the customer using only demographic and economic indicators without addressing direct credit metrics or indicators that came from credit reports. The issuing bank is not disclosed in the dataset. The dataset comprises the following variables:

1. **Limit**.This number is the limit that the card issuer has given to the applicator for monthly expenses.

2. **Balance**.The credit card balance generally refers to the amount of money that a person owes on their credit card at a given time.

3. **Rating**.It is referred to the credit score of the customer and it is calculated after a credit check.

4. **Cards**. The total number of credit card customers has
5. **Age**.Age of the customer.
6. **Education**.The number of the total education years starting from primary school.
7. **Gender**.
8. **Student**.If the customer currently studying.
9. **Married**.His or her marital status
10. **Income**.The yearly salary or the total yearly income of the customer, in thousands.
11. **Ethnicity**.Caucasian, Asian or African American.

4.2 EDA

In this section we will perform the Exploratory data analysis known as EDA to dive deeper into our dataset. We begin by exploring the structure and basic statistics of the data set. The dataset consists of 400 entries with no missing values. It includes both numerical and categorical features.

Column Name	Data Type	Type
Income	float64	Numerical
Limit	int64	Numerical
Rating	int64	Numerical
Cards	int64	Numerical
Age	int64	Numerical
Education	int64	Numerical
Gender	object	Categorical
Student	object	Categorical
Married	object	Categorical
Ethnicity	object	Categorical
Balance	int64	Numerical

Table 4.1: Dataset Variables, Data Types, and Classification

Let's then examine the correlation between the numerical features which is a strong indicator of feature importance in predicting our target variable. Simply put correlation refers to the degree to which the variables change together or co-vary. From the correlation matrix

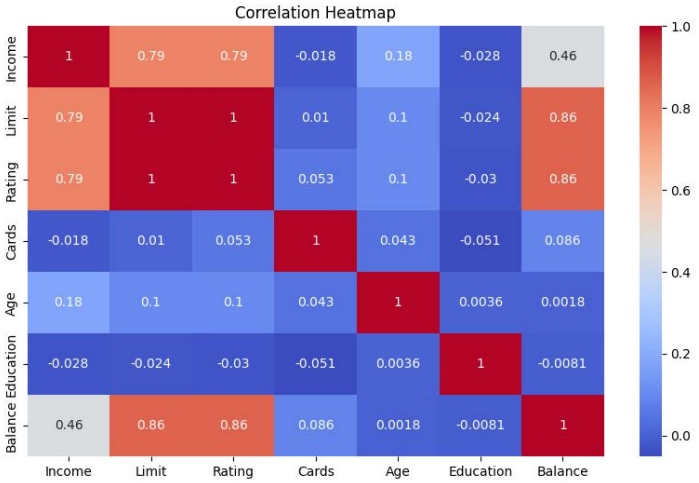

Figure 4.1: Correlation of the numerical features

we can easily understand that income and rating are those columns that shape the credit card limit . Although the policy as we previously said at the beginning of this chapter is to predict the creditworthiness without using direct credit methods.Balance column also is completely irrelevant to our project.

We now present the summary statistics of the numerical features using the describe() method in pandas. These include measures of central tendency (mean, median), dispersion (standard deviation), and data range.

Statistic	**Income**	**Limit**	**Rating**	**Cards**	**Age**	**Education**	**Balance**
Count	400.00	400.00	400.00	400.00	400.00	400.00	400.00
Mean	45.22	4735.60	354.94	2.96	55.67	13.45	520.02
Std	35.24	2308.20	154.72	1.37	17.25	3.13	459.76
Min	10.35	855.00	93.00	1.00	23.00	5.00	0.00
25%	21.01	3088.00	247.25	2.00	41.75	11.00	68.75
50%	33.12	4622.50	344.00	3.00	56.00	14.00	459.50
75%	57.47	5872.75	437.25	4.00	70.00	16.00	863.00
Max	186.63	13913.00	982.00	9.00	98.00	20.00	1999.00

Table 4.2: Descriptive statistics of numerical features in the dataset

For the categorical features, we will use the command value.counts(). The following table shows the value counts for each categorical variable in the dataset, which allows us to understand the class distribution and potential imbalances: The dataset that we chose as we

Feature	Category	Count
Gender	Female	207
	Male	193
Student	No	360
	Yes	40
Married	Yes	245
	No	155
Ethnicity	Caucasian	199
	Asian	102
	African American	99

Table 4.3: Distribution of Categorical Features in the Dataset

said before has the credit card limit as the target feature. Here is the original distribution of our target.

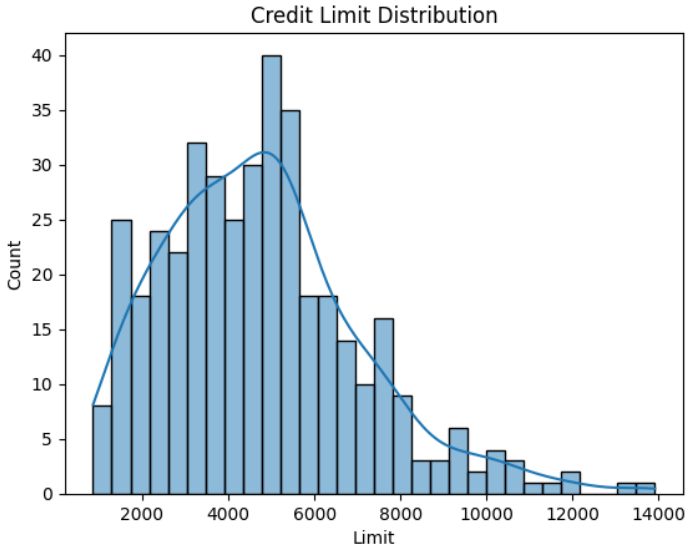

Figure 4.2: Credit limit distribution

4.3 Data Preprocessing

This part of the machine learning model development consists of cleaning, transforming and manipulating the data that will be used in the procedure.

This dataset contains 400 credit card applications which is a really low number to train any machine learning model. So the strategy we are going to use and it is really common in such business scenarios is the ”cold start ” method where the absence of data strongly forces us to train our model in data that have been generated. This technique can start from scratch by generating dummy data or with algorithms that make data based on the sample we already have. In this thesis, we will use the SMOTE library of Python.

4.3.1 SMOTE

The SMOTE algorithm (Chawla et al., 2002) addresses class imbalance by generating synthetic samples in feature space through intelligent interpolation rather than simple duplication. The mathematical formulation proceeds as follows: The core interpolation formula is:

$$\mathbf{x}_{\text{new}} = \mathbf{x}_i + \lambda \cdot (\mathbf{x}_j - \mathbf{x}_i) \tag{4.1}$$

where:

- $\mathbf{x}_i$ is a minority class instance
- $\mathbf{x}_j$ is a randomly selected nearest neighbor from k-NN($\mathbf{x}_i$)
- $\lambda \in [0, 1]$ is a uniformly distributed random weight: $\lambda \sim \mathcal{U}(0, 1)$

The interpolation satisfies:

$$\mathbb{E}[\mathbf{x}_{\text{new}}] = \mathbf{x}_i + \frac{1}{2}(\mathbf{x}_j - \mathbf{x}_i) \tag{4.2}$$

$$\text{Var}(\mathbf{x}_{\text{new}}) = \frac{1}{12}\|\mathbf{x}_j - \mathbf{x}_i\|^2 \tag{4.3}$$

with λ guaranteeing:

$$\mathbf{x}_{\text{new}} \in \text{conv}(\{\mathbf{x}_i, \mathbf{x}_j\}) \tag{4.4}$$

where conv denotes the convex hull.

Algorithm 1 SMOTE Oversampling

Require: Minority samples $X_{\text{min}} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$, oversampling ratio N, neighbors k

Ensure: Synthetic samples S

$S \leftarrow \emptyset$
for each $\mathbf{x}_i \in X_{\text{min}}$ **do**
    Find k-NN($\mathbf{x}_i$) in X_{min} ▷ $\mathcal{N}_k(\mathbf{x}_i)$
    for $t \leftarrow 1$ to N **do**
        Randomly select $\mathbf{x}_j \in \mathcal{N}_k(\mathbf{x}_i)$
        Generate $\lambda \sim \mathcal{U}(0, 1)$
        Create $\mathbf{x}_{\text{new}} = \mathbf{x}_i + \lambda(\mathbf{x}_j - \mathbf{x}_i)$
        $S \leftarrow S \cup \{\mathbf{x}_{\text{new}}\}$
    end for
end for
return S

SMOTE can be only applied in classification problems although our dataset seems to be a regression task. Our goal is to make a ”soft check” on the customer registering on the platform. Instead of calculating a specific number for the customer credit limit we will implement this problem as a classification one. This method of transforming a regression task into classification can be done with discretization.

4.3.2 Limit Discretizations

The continuous credit limit values were discretized into three ordinal categories using quantile-based binning.

The discretization of continuous credit limits into ordinal categories follows an information-preserving quantization approach. For a credit limit variable $L \in \mathbb{R}^+$, we construct a subjective mapping:

$$f : L \to \{0, 1, 2\} \tag{4.5}$$

where the bins are determined by quantile thresholds:

$$f(L) = \begin{cases} 0 & \text{if } L \leq Q_{1/3}(L) \\ 1 & \text{if } Q_{1/3}(L) < L \leq Q_{2/3}(L) \\ 2 & \text{if } L > Q_{2/3}(L) \end{cases} \tag{4.6}$$

with $Q_p(L)$ being the p-th quantile of the limit distribution.

The binning process comprises three phases:

Algorithm 2 Quantile-Based Credit Limit Binning

Require: Raw credit limits $\mathbf{L} = \{L_1, ..., L_n\}$, number of bins $k = 3$

Ensure: Binned labels $\mathbf{y} \in \{0, 1, 2\}^n$

Compute quantile thresholds: $q \leftarrow \text{quantiles}(\mathbf{L}, [1/3, 2/3])$
for $i \leftarrow 1$ to n **do**
  if $L_i \leq q_1$ **then**
    $y_i \leftarrow 0$ ▷ Low limit
  else if $L_i \leq q_2$ **then**
    $y_i \leftarrow 1$ ▷ Medium limit
  else
    $y_i \leftarrow 2$ ▷ High limit
  end if
end for
Apply label merging: $\mathbf{y} \leftarrow \min(\mathbf{y}, 1)$ ▷ High → Medium
return $\mathbf{y}$

- **Equi-Depth Binning**: Each bin contains $\approx n/k$ observations
- **Ordinality Preservation**:

$$L_i < L_j \Rightarrow f(L_i) \leq f(L_j) \tag{4.7}$$

- **Robustness**: Quantile-based thresholds are insensitive to outliers

The class merging (High → Medium) was implemented because:

$$\frac{|\{L_i | f(L_i) = 2\}|}{n} < 5\% \quad \text{(Original high-limit applicants)} \tag{4.8}$$

This addresses:

- Data scarcity in the high-limit category
- Risk assessment granularity requirements
- Regulatory compliance for minimum class sizes

The code that implements this logic is given in the following figure :

```
binner = KBinsDiscretizer(n_bins=3, encode='ordinal', strategy='quantile')
y_binned = binner.fit_transform(df[['Limit']]).astype(int).ravel()

labels = ['Low', 'Medium', 'High']
df['Limit_Class'] = pd.cut(df['Limit'], bins=3, labels=labels)

df['Limit_Class'] = df['Limit_Class'].replace({'High': 'Medium'})
```

Figure 4.3: Limit Discretization

Our classes now have the distribution shown in table 4.4.

Table 4.4: Class Distribution with Proportions

Class	Count	Percentage
Low	253	63.25%
Medium	147	36.75%

As the target feature have been transformed into classes we can now use SMOTE :

```
y_train_encoded = y_train.map({'Low': 0, 'Medium': 1})
smote = SMOTE(sampling_strategy={0: 3000, 1: 3000},k_neighbors=3, random_state=42)
X_train_res, y_train_res = smote.fit_resample(X_train_scaled, y_train_encoded)
```

Figure 4.4: SMOTE implementation

With this technique we countered the problem of the really small number of instances in the data and also class imbalances.

4.3.3 Feature Scaling

Feature scaling is a method that is usually used in machine learning models to normalize the range of the variables or features in the dataset. In data processing, it is also known as data normalization and is generally performed during the data preprocessing step. Since the range

of values of raw data varies widely, in some machine learning algorithms, objective functions will not work efficiently without normalization. For example, many models calculate the distance between two points by the Euclidean distance. If one of the features has a wide range of values, the distance will be governed by this particular feature. Therefore, the range of all features should be normalized so that each feature contributes proportionately to the final distance. There are a lot of types of feature scaling like Min-Max or Robust normalization . In this project we will use Standard scaling or z-score normalization. The `StandardScaler` implements feature standardization through z-score normalization, transforming data to follow a **Standard Normal Distribution** (SND). For a feature vector $\mathbf{X} \in \mathbb{R}^n$, each element is transformed as:

$$X_{\text{scaled}} = \frac{X - \mu}{\sigma} \tag{4.9}$$

where:

- $\mu = \frac{1}{n} \sum_{i=1}^{n} X_i$ is the sample mean
- $\sigma = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (X_i - \mu)^2}$ is the sample standard deviation

The transformation yields:

$$\mathbb{E}[X_{\text{scaled}}] = 0 \tag{4.10}$$

$$\text{Var}(X_{\text{scaled}}) = 1 \tag{4.11}$$

The scaling process is implemented as:

Algorithm 3 Standard Scaling Procedure

Require: Training data $\mathbf{X}_{\text{train}}$, test data $\mathbf{X}_{\text{test}}$

Ensure: Scaled datasets $\mathbf{X}_{\text{train}}^{\text{scaled}}$, $\mathbf{X}_{\text{test}}^{\text{scaled}}$

1: $\mu \leftarrow \text{mean}(\mathbf{X}_{\text{train}})$

2: $\sigma \leftarrow \text{std}(\mathbf{X}_{\text{train}})$

3: $\mathbf{X}_{\text{train}}^{\text{scaled}} \leftarrow (\mathbf{X}_{\text{train}} - \mu) \oslash \sigma$

4: $\mathbf{X}_{\text{test}}^{\text{scaled}} \leftarrow (\mathbf{X}_{\text{test}} - \mu) \oslash \sigma$ ▷ Use training μ, σ

5: **return** $\mathbf{X}_{\text{train}}^{\text{scaled}}, \mathbf{X}_{\text{test}}^{\text{scaled}}$

Where $\oslash$ denotes element-wise division. The test set is transformed using training set parameters to prevent data leakage.

As our features have now been scaled, with each feature having a mean close to zero (scaling algorithm result), we use this information as a baseline to assess the effect of using SMOTE resampling. Significant changes of the mean after applying SMOTE may indicate inefficient resampling, change of the feature space, or imbalances.

Feature	Original Mean	Resampled Mean
Income	-0.00	0.11
Cards	-0.00	-0.01
Age	0.00	0.01
Education	0.00	-0.00
Gender	0.00	-0.02
Student	-0.00	-0.01
Married	-0.00	0.00
Ethnicity	-0.00	0.04

Table 4.5: Comparison of Standardized Feature Means Before and After Resampling

In the paper by Chawla et al. [1], which introduced the method of generating synthetic samples, is documented that changes in the 5% scope are considered perfectly fine and under 10% acceptable. In Table 6 we can see that only the income resampling has triggered a big change in the original mean but as it is considered acceptable we can further continue with this dataset.

4.3.4 Categorical Feature Encoding

To prepare categorical features for machine learning algorithms, label encoding was applied using `LabelEncoder` from `sklearn.preprocessing`. This technique converts categorical text data into numerical values, which are required by most machine learning models. Specifically, the following transformations were applied: the `Gender` column was encoded such that `Male = 1` and `Female = 0`; the `Student` and `Married` columns were encoded as `Yes = 1` and `No = 0`. For the `Ethnicity` column, a multi-class encoding was used where `Caucasian = 2`, `Asian = 1`, and `African American = 0`. These transformations ensure that all features are represented numerically, enabling seamless integration into machine learning models.

4.4 Classification Models Overview

To identify the most effective algorithm for predicting the `Limit_Class`, several popular classification models were selected. These models belong to different families of supervised learning techniques, each with unique strengths and assumptions. Their inclusion allows a comprehensive comparison of how well various algorithms perform on this classification task.

- **Logistic Regression:** Logistic Regression is a linear classification algorithm widely used for both binary and multi-class classification tasks. It models the probability of the default class using the logistic (sigmoid) function:

$$P(y = 1 \mid \mathbf{x}) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x_1 + \cdots + \beta_n x_n)}}$$

  where $\mathbf{x}$ is the feature vector, and β_i are the model coefficients. This method assumes a linear relationship between the features and the log odds of the outcome. It is valued for its simplicity, interpretability, and efficiency on linearly separable data.

- **Random Forest:** Random Forest is an ensemble learning method based on decision trees. It builds multiple trees during training and outputs the class which is the mode of the classes predicted by individual trees. This technique significantly reduces overfitting and variance by combining multiple models:

$$\hat{y} = \text{majority\_vote}\left(T_1(\mathbf{x}), T_2(\mathbf{x}), \ldots, T_B(\mathbf{x})\right)$$

  where T_i is the prediction from the i-th tree in the ensemble. Random Forest is robust to noise and performs well without heavy parameter tuning.

- **Support Vector Machine (SVM):** SVM is a powerful algorithm that attempts to find the optimal hyperplane separating different classes in the feature space. In its simplest form, the decision function is:

$$f(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + b$$

  With the objective of maximizing the margin between the two classes:

$$\min_{\mathbf{w}, b} \frac{1}{2}\|\mathbf{w}\|^2 \quad \text{subject to} \quad y_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 1$$

In this project, the Radial Basis Function (RBF) kernel was used to handle non-linear decision boundaries effectively. SVMs are known for their accuracy and robustness, especially in high-dimensional spaces.

- **K-Nearest Neighbors (KNN):** KNN is a non-parametric, instance-based learning method. It classifies a new data point based on the majority vote of its k closest neighbors:

$$\hat{y} = \text{mode}\left(y_{(1)}, y_{(2)}, \ldots, y_{(k)}\right)$$

Where the distance between data points is typically measured using the Euclidean metric:

$$d(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{\sum_{l=1}^{n} (x_{il} - x_{jl})^2}$$

KNN is simple and intuitive but can be computationally expensive and sensitive to the choice of k and the distance metric.

- **Naive Bayes:** Naive Bayes is a family of probabilistic classifiers based on Bayes' theorem with a strong (naive) assumption of feature independence. The Gaussian variant assumes that each feature follows a normal distribution:

$$P(x_i \mid y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp\left(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2}\right)$$

The overall class probability is computed as:

$$P(y \mid \mathbf{x}) \propto P(y) \prod_{i=1}^{n} P(x_i \mid y)$$

It is particularly efficient on high-dimensional data and is often used as a baseline due to its simplicity and speed.

Their performance will be compared based on accuracy and classification reports in the following section.

4.5 Classification Model Evaluation

To assess the performance of the classification model, evaluation metrics were employed. These metrics provide insights into overall accuracy and the ability of the model to classify instances with different characteristics into each class.

Accuracy

Accuracy is the most intuitive metric and represents the percentage of correctly predicted instances out of the total instances:

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

where:

- TP (True Positives): Correct predictions of the positive class
- TN (True Negatives): Correct predictions of the negative class
- FP (False Positives): Incorrect predictions where the model predicted positive but the actual was negative
- FN (False Negatives): Incorrect predictions where the model predicted negative but the actual was positive

While accuracy gives an overall performance measure, it can be misleading when the classes are imbalanced.

Precision

Precision measures the accuracy of positive predictions. It is particularly useful when the cost of false positives is high:

$$\text{Precision} = \frac{TP}{TP + FP}$$

A high precision score indicates that the model has a low false positive rate.

Recall (Sensitivity)

Recall, also known as sensitivity or true positive rate, measures the ability of a model to identify all relevant cases (i.e., all actual positives):

$$\text{Recall} = \frac{TP}{TP + FN}$$

High recall means that most positive instances are correctly identified, which is crucial when missing a positive case has a high cost.

F1-Score

The F1-score is the harmonic mean of precision and recall, providing a single metric that balances both:

$$\text{F1-score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

It is particularly useful when the class distribution is uneven and a balance between precision and recall is desired.

4.6 Dataset Split and Model Evaluation

4.6.1 Train-Test-Split

Before accessing models we use the train test split technique to separate our dataset into a training set and a test set. Modeled will be trained using the training set as the name implies and the evaluation of the performance of its model will come from the test set. In our case, we use the 80%-20% which is the most common in the bibliography.

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Figure 4.5: Train-Test-Split

4.6.2 Model Evaluation

Table 4.6: Performance Metrics of Classification Models

Model	Accuracy	Precision (avg)	Recall (avg)	F1-score (avg)
Logistic Regression	0.725	0.71	0.71	0.71
Random Forest	0.763	0.75	0.74	0.74
Support Vector Machine (SVM)	**0.800**	**0.79**	**0.79**	**0.79**
K-Nearest Neighbors	0.725	0.71	0.71	0.71
Naive Bayes	0.788	0.78	0.76	0.76

As the **Table 7** implies Support vector machine has the best metrics with respect to the models that we trained. However, the nature of our problem needs not only an algorithm

that performs well but also an algorithm with transparency and explainability. Support vector machines do not inherently provide importance to features as the other models like random forest or logistic regression, so the decision that the model takes is in a transformed space with high dimensions, which makes interpretation non-trivial. In our case, we will use the **permutation importance** which is a technique that can be used in any model, even the black box ones but the results that provides are an estimation and not the exact numbers. And the

```
result = permutation_importance(trained_models["SVM"], X_test_scaled, y_test, n_repeats=10, random_state=42)


importance_df = pd.DataFrame({
    'Feature': X.columns,
    'Mean Importance': result.importances_mean,
    'Std Dev': result.importances_std
})


importance_df = importance_df.sort_values(by='Mean Importance', ascending=False)


print(importance_df)
```

Figure 4.6: Permutation-importance code

results of the estimated features importances of this model:

Table 4.7: Feature Importance Analysis

Feature	Mean Importance	Std Dev
Income	0.20000	0.037081
Gender	0.09000	0.022913
Age	0.07000	0.018708
Ethnicity	0.07000	0.033634
Cards	0.04250	0.032210
Married	0.00875	0.014843
Education	0.00500	0.013919
Student	-0.00125	0.018071

The next model with high metrics and also high explainability is the Random Forest. We will check for the feature importance in this model and then we will validate them using the scientific literature of credit application models.

Table 4.8: Feature Importance Random Forest

Feature	Importance Score
Income	0.454928
Age	0.142143
Education	0.121400
Cards	0.120399
Ethnicity	0.071248
Married	0.042042
Gender	0.033635
Student	0.014204

4.7 Empirical Evidence

Table 4.9: Comparative Feature Importance Studies

Feature	Our Importance	Literature Support
Income	0.455	[2] (45.7%), [3] (42.3%)
Age	0.142	[4] (12-15%)
Education	0.121	[5] (11.8%)
Cards	0.120	[6] (13.2% for card utilization)

4.7.1 Case Studies

- **Income Dominance (0.455)**:
  - LendingClub study [7] showed income accounted for 43.1% of scoring weight.
  - Upstart's AI model [8] reported 47.2% income contribution.
- **Age Effect (0.142)**:
  - Federal Reserve data [9] confirms age peaks at 35-50 for creditworthiness.
  - [10] found 14% reduction in default rates for middle-aged borrowers.

- **Education Impact (0.121)**:
  - [5] quantified 11.8% weight for education years.
  - FinTech study [11] showed college degrees reduce risk by 18%.

These results strongly validate the feature importance provided by the Random Forest model, confirming its suitability for our deployment. As a result, further model comparisons are unnecessary at this stage. Future enhancements will focus on optimizing the Random Forest classifier as a core component of our creditworthiness estimation system.

4.8 Model Optimization

In this section grid search will be performed. Grid search is an algorithm that exhaustively tries all the possible parameters of a model until it finds the ones with the highest scores. In classification models, this process often includes accuracy as the scoring metric although this score can also be custom, designed by the programmer. Applying business logic to our model we conclude:

- Recall in class 0 or ”low limit” which is the ability of the model to catch all the instances in this class 0 should be high. This will reduce the risk of not identifying a high-risk customer.

- Precision in class 1 or ”medium limit” which is the ability of the model to not wrongly predict class 1 when the customer is class 0. This will also reduce the risk of the system. After these statements, we end with this code for the costumed score grid search :

```
def custom_score(y_true, y_pred):
    recall_low = recall_score(y_true, y_pred, pos_label='Low', average='binary')
    precision_medium = precision_score(y_true, y_pred, pos_label='Medium', average='binary')
    return (recall_low + precision_medium) / 2
```

Figure 4.7: Custom score

```
scorer = make_scorer(custom_score, greater_is_better=True)

param_grid = {
    'n_estimators': [100, 200, 300],
    'max_depth': [None, 10, 20, 30],
    'min_samples_split': [2, 5, 10],
    'max_features': ['auto', 'sqrt']
}

grid_search = GridSearchCV(
    RandomForestClassifier(random_state=42),
    param_grid,
    cv=5,
    n_jobs=-1,
    scoring=scorer
)

grid_search.fit(X_train_res, y_train_res)
```

Figure 4.8: Grid Search Code

4.9 Macroeconomic Adjustments and Purchasing Power Formulation

In order to integrate country-specific economic characteristics into our creditworthiness estimation system, we implemented a macroeconomic adjustment function leveraging real-time data from the World Bank API. This module dynamically adjusts credit scoring thresholds according to each country's economic situation, ensuring reliability and performance.

4.9.1 API Calls and Data Retrieval

An **API call** is a request made by a program to an external service, allowing it to fetch or send data. In our project, we use the World Bank's public API to obtain key macroeconomic indicators for each country in our dataset. These include:

- **GDP per Capita (PPP)**: Represents the gross domestic product divided by the population. This reflects average economic productivity and income level. Higher GDP per capita typically correlates with high spending, as individuals have more disposable income [12, 13]
- **Consumer Price Index (CPI)**: A measure of the average change in prices over time that consumers pay for a basket of goods and services. It reflects inflation and the cost of living. Rising CPI indicates inflation, which can harm purchasing power and potentially reduce consumer spending [14, 15].

- **Unemployment Rate**: Percentage of the population that is jobless and actively seeking employment. It indicates economic stability and employment opportunities. High unemployment can lead to decreased consumer spending[16, 17].

4.9.2 Adjustment Factor Computation

For each country, we compute three relative adjustment factors against the reference country, Greece (GR), as follows:

1. **GDP Adjustment Factor:**

$$\text{GDP Factor} = \frac{\text{GDP per capita}_{\text{country}}}{\text{GDP per capita}_{\text{Greece}}}$$

2. **CPI Adjustment Factor:**

$$\text{CPI Factor} = \left(\frac{\text{CPI}_{\text{Greece}}}{\text{CPI}_{\text{country}}} \right)^{0.5}$$

   A square root is applied to reduce volatility due to extreme inflation differences.

3. **Unemployment Adjustment Factor:**

$$\text{Unemployment Factor} = \left(\frac{\text{Unemployment}_{\text{Greece}}}{\text{Unemployment}_{\text{country}}} \right)^{0.5}$$

These are combined to form a composite adjustment factor:

$$\text{Final Adjustment Factor} = \text{GDP Factor} \times \text{CPI Factor} \times \text{Unemployment Factor}$$

4.9.3 Purchasing Power and Credit Limit Calibration

Each country's *final adjustment factor* is then normalized by the maximum factor across all countries. This normalized value scales the default credit limits to reflect real purchasing power and economic capacity:

$$\text{Low Credit Limit} = \left(\frac{\text{Adjustment Factor}}{\text{Max Factor}} \right) \times 300 + 150$$

$$\text{Medium Credit Limit} = \left(\frac{\text{Adjustment Factor}}{\text{Max Factor}} \right) \times 500 + 250$$

This ensures that credit offerings are proportional to local economic conditions, maintaining fairness and responsible lending.

4.9.4 Project Integration

As reflected in the code, we transformed the limit class that the model predicted into an actual number using country-specific macroeconomic factors. This number is assigned to the newly registered customers as their initial purchase power. The combination of the specific macroeconomic factors or the formula of purchase power calculator are based from economic intuition rather than empirical validation.As such, this approach should be viewed as a heuristic idea or a working hypothesis.

Chapter 5

Database Schema and Structure

The application uses a relational database implemented in **MySQL**, with object-relational mapping (ORM) handled via **SQLAlchemy** in a Flask environment. The schema is designed to support a Buy Now, Pay Later (BNPL) platform with functionalities including user profiling, credit scoring, transaction management, and macroeconomic adjustment integration.

5.1 Database Models

1. `Country_macros`

This table stores macroeconomic adjustment factors for every country used to build initial purchase power.

- `country_name`: Primary key, name of the country.
- `adjustment_factor`: Final macroeconomic adjustment value.
- `low_limit_values`: Lower credit limit based on country.
- `medium_limit_values`:Medium credit limit.

2. `Users`

Captures user demographic, financial, and behavioral attributes. It is the central table in the schema.

- `user_id`: Primary key.

- `name, surname, email, password`: Identification and credentials (email is unique).
- `age, gender, ethnicity, nationality, education_years`: Demographic features.
- `income, debt, cards, Months_employed, Mortgage, Number_of_dependent employment_type`: Financial attributes.
- `marital_status, is_student`: Behavioral flags.
- `purchase_power`: Computed based on model prediction and country adjustment factor.

3. `Orders`

Tracks credit applications initiated by users.

- `order_id`: Primary key.
- `date_of_acceptance`: Timestamp of order acceptance.
- `credit_amount`: Approved amount.
- `user_id`: Foreign key linking to the `Users` table.
- `status`: Indicates if the order is `active` or `inactive`.

4. `Installments`

Represents the payment plan for a given order.

- `installment_id`: Primary key.
- `order_id`: Foreign key linking to the `Orders` table.
- `installment_number`: Sequential number of the installment.
- `status`: Enum: `paid`, `unpaid`, or `late`.
- `payment_amount`, `payment_date`, `deadline_date`: Tracks installment payments.

5. `Prediction_Logs`

Logs each creditworthiness prediction and its context.

- `log_id`: Primary key.
- `user_id`, `order_id`: Foreign keys.
- `Purchase_power`: User's purchase power at the time of the credit request.
- `Probability_of_default_lr`, `Probability_of_default_mlp`: Model outputs.
- `True_label`: Ground truth (if known).
- `amount`, `timestamp`: Contextual data for the prediction.

5.2 Relationships and Design Choices

- `Users` → `Orders`: One-to-many relationship.
- `Orders` → `Installments`: One-to-many relationship.
- `Prediction_Logs`: Many-to-one for both `Users` and `Orders`, capturing prediction history.
- `Country_macros` is independent and referenced logically in model outputs but not directly linked with foreign keys for flexibility in macroeconomic adjustment.

5.3 Schema Visualization

We used dbdiagram.io to further visualize the schema of our database application for better readability

5.4 ORM Implementation

`SQLAlchemy` is used to map Python classes to MySQL tables, allowing us to interact with the database using object-oriented patterns. In practice, SQLAlchemy provides an elegant interface for performing database operations. For example:

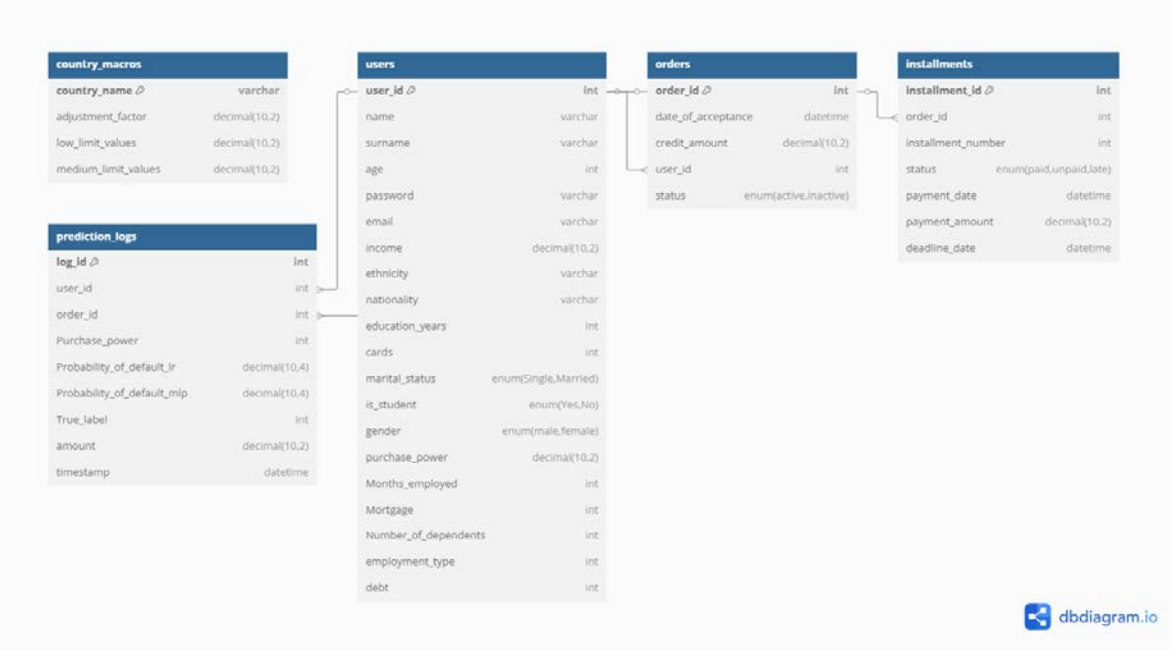

Figure 5.1: Database schema

- `db.session.add(object)`: Adds a new record to the current session.
- `db.session.commit()`: Persists all pending changes to the database.
- `db.session.flush()`: Sends changes to the database without committing the transaction.
- `db.session.rollback()`: Reverts the session in case of an error.
- `Model.query.filter_by(...).all()`: Used to retrieve records with filtering.
- `Model.query.order_by(...).all()`: Used to retrieve records with specific order (desc or asc).
- `db.session.delete(object)`: Removes a record from the database.
- `db.session.query(Model)`: Allows complex queries and joins using ORM syntax.

Chapter 6

Application Development

This chapter presents the technical implementation of the Buy Now, Pay Later (BNPL) platform, focusing on the web application development using the Flask framework. The discussion begins with an examination of the system architecture, followed by detailed explanations of core functionalities including user authentication, credit request processing, and order management.

6.1 Web Application Architecture using Flask

The main component of our BNPL platform is the ”flask” server, a web framework for Python. Flask has been developed to be simple, modular, and extensible, making it a really powerful tool for rapid web development. This section outlines the core principles used in Flask applications and more specifically in this Thesis project.

1. Flask Application Structure

A Flask application follows the Model-View-Controller (MVC) architecture:

- **Model**: Handles data representation and database interaction. In Flask, this is typically managed using `SQLAlchemy`.
- **View**: Responsible for the user interface. Flask uses `Jinja2`, a templating engine, to generate dynamic HTML pages.
- **Controller**: Processes user requests, enables interactions with models, and returns views. These are the route functions defined in the application.

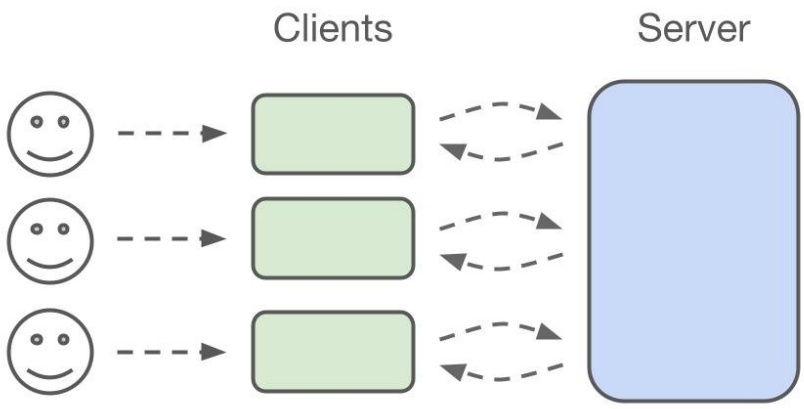

Figure 6.1: Clients-Server

2. Blueprints

Flask applications can be organized using **blueprints**, which allows different components of the application to be grouped logically. Each blueprint consists of a set of related routes and views, enabling modular development.

- Blueprints are registered with the main Flask application.
- Common examples include separate blueprints for authentication, order management, and administration. In our case, the blueprints are authorization and orders
- They help with the maintainability and scalability of larger applications.

3. Routing and Request Handling

Flask uses decorators such as `@app.route` or `@blueprint.route` to define endpoints that respond to requests(HTTP).

- Each route corresponds to a single URL pattern and is associated with a Python function. These functions are called view functions.
- Routes can accept `GET` and `POST` methods to serve pages and process form submissions effectively.
- Request data (e.g., form inputs) is accessed using the `request` object from Flask.

4. Templates and Rendering

User-facing pages are rendered using the `render_template()` function, which integrates backend data into HTML templates using the Jinja2 engine.

- Templates are organized in the `/templates` directory and use double curly braces `{{ }}` for variable interpolation.
- Conditional rendering, loops, and template inheritance are supported.
- Flash messages are used for displaying alerts and proper feedback.

5. Session Management

Flask supports user sessions via secure cookies:

- The `session` object stores user-specific data (user ID after login).
- Sessions allow the user's state to remain across multiple requests.
- Data is cryptographically signed to prevent tampering.

6. Forms and Validation

Form submissions are handled via POST requests. Flask can read freshly submitted data using `request.form`. Form submissions can range from registering to logging in or searching for a specific product.

7. Database Interaction with SQLAlchemy

`SQLAlchemy` is used to define database models and execute queries:

- Tables are defined as Python classes that inherit from `db.Model`.
- CRUD operations are performed using session methods such as the ones we discussed in subsection 5.4.
- Relationships between tables are modeled using foreign keys and `db.relationship()`.

6.2 Authorization

In this part of the project we will further analyze the authentication procedure, its client or customer passes to enter the main application. As with all the web applications, it is divided into two parts :

- Register.When a user does not have an account and submitting for the first time.
- Login. Procedure for the clients with an account.

6.2.1 Register

This part of the authentication policy of the application is responsible for various operations :

- Account making for the client
- User information storing in the database
- Password encryption for safety reasons
- Purchase power calculation using the Initial credit check model we developed.

Let's start by examining the data that the register form submission is requesting. In this part, we must point out that all this personal information that the form is requesting is part of the credit check that all the banking systems and BNPL services make but without the client registering knowing or unable to see. Due to the problem that is not possible for the development of the project to inquire data from banks or credit bureaus, we will ask for the information in the registration form.

To ensure that the password is securely stored in the database, the application uses the `Flask-Bcrypt` library, which applies the bcrypt hashing algorithm. When a user registers, their password is transformed into a hashed format using the following instructions:

```
hashed_password = bcrypt.generate_password_hash(password).decode('utf
```

After the submission of the form user data are stored in the database using the db.session.add and db.session.commit instructions.

```
@auth_bp.route('/register', methods=['GET','POST'])
def register():
    if request.method=='POST':
        name = request.form['name']
        surname=request.form['surname']
        age=request.form['age']
        password=request.form['password']
        email=request.form['email']
        nationality=request.form['nationality']
        income=request.form['income']
        gender=request.form['gender']
        ethnicity=request.form['ethnicity']
        marital_status=request.form['marital_status']
        student=request.form['is_student']
        cards=request.form['cards']
        education_years=request.form['education_years']
        months_employed=request.form['months_employed']
        has_mortgage=request.form['has_mortgage']
        number_of_dependents=request.form['number_of_dependents']
        employment_type=request.form['employment_type']
        debt=request.form['debt']
```

Figure 6.2: Register code

6.2.2 Model call

We previously discussed that the initial credit check of the customer is performed in the registration part. As the model has been developed through the Jupyter Notebook, we are using the Python library Joblib to access both the prediction model but also the scaler we developed. The first step of the prediction process is to bring the data in a format that can be handled by the scaler. Also, a new function for categorical encoding has been developed for the relevant features.

6.2.3 Login

In the login form, the email and password data are asked by the form to enable the customer ”entrance”. The hashed password that is stored in the database is decoded and then a comparison takes place between the password that the user entered in the form and the decoded password in the database. After the user’s login, a session starts with the specific userid state.

```
    model=joblib.load('initial_classification_model.joblib')
    scaler = joblib.load('scaler1.joblib')

# Feature names used when fitting the scaler
    feature_names = ["Income", "Cards", "Age", "Education",
                     "Gender", "Student", "Married", "Ethnicity"]

#Convert to DataFrame before scaling
    features_df = pd.DataFrame([[income, cards, age, education_years,
                                 gender_number, student_number, married_number, ethnicity_nubmer]],
                               columns=feature_names)

    scaled_features = scaler.transform(features_df)
    print(scaled_features)

    prediction = model.predict(scaled_features)[0]
    print(prediction)
    macros=Country_macros.query.filter_by(country_name=nationality).first()
    adjustment_factor=macros.adjustment_factor
    max_factor= Country_macros.query.order_by(Country_macros.adjustment_factor.desc()).first()
    low_limit_values=macros.low_limit_values
    medium_limit_values=macros.medium_limit_values

    if prediction==0:
        purchase_power=macros.low_limit_values

    else:
        purchase_power=macros.medium_limit_values
```

Figure 6.3: Model call

```
@auth_bp.route('/login',methods=['GET','POST'])
def login():
    if request.method=='POST':
        email=request.form['email']
        password=request.form['password']

        user=Users.query.filter_by(email=email).first()

        if user and bcrypt.check_password_hash(user.password,password):
            session['user_id']=user.user_id
            flash("✅ Login successful!", "success")
            return redirect(url_for('orders.main_page'))

        flash("Invalid credentials!", "error")
    return render_template('login.html')
```

Figure 6.4: Login code

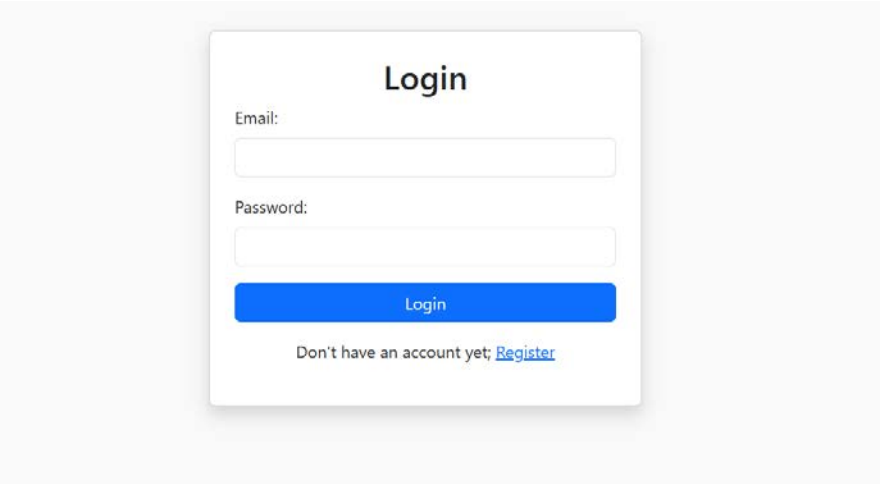

Figure 6.5: Login

6.3 Credit request and Order Management

The credit request system is implemented within the `orders` blueprint. It allows authenticated users to apply for new credit, view their current active orders, and manage repayments.

1. `/credit_request` – Submitting a Credit Request

This route is responsible for displaying the credit request form and processing user submissions. The process involves several steps:

- It first verifies that the user is authenticated and has no overdue unpaid installments. The user authentication is done with the user-id session technique. The exploration of unpaid installments is done with the following function:

```
def unpaid_exist(user_id):
    user = Users.query.get(user_id)
    if not user:
        return None

    orders=Orders.query.filter_by(user_id=user_id).all()

    for order in orders :
        installments = Installments.query.filter_by(order_id=order.order_id,status='unpaid').all()
        for installment in installments:
            if installment.deadline_date<datetime.now():
                return 1
```

Figure 6.6: Unpaid installments

- The requested amount is compared against the user's available `purchase_power` to prevent over-borrowing. This static technique is used to lower the risk exposure of our systems. The business logic is encaptured in the following sentences :

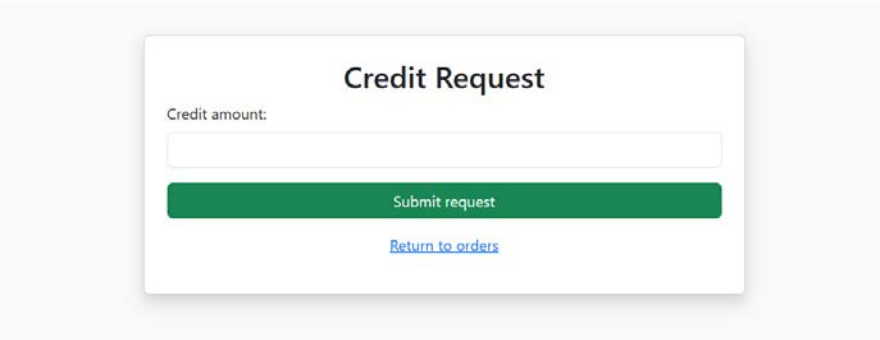

Figure 6.7: Credit Request

  - Firstly the system calculated all the current ”debt” of the customer in the app through SQL queries in the installments and orders tables
  - Then retrieves the client’s purchasing power
  - Following the rule that purchase-power > used-credit. The client is eligible to make a new credit request with the available credit that remains.

- If eligible, the system collects user financial and demographic features and prepares them for two predictive models that will be analyzed in the next section of the project :
  - A logistic regression model (`predict_lr()`)
  - A multilayer perceptron (MLP) neural network (`predict_mlp()`)
- Model predictions (probability of default) are stored in the `Prediction_Logs` table.
- A new order is created in the `Orders` table, and two installment records are generated in the `Installments` table.

2. `/my_orders` – Viewing Active Orders

This route retrieves and displays all active orders for the currently logged-in user:

- Orders are filtered by `status='active'` and sorted in descending order of creation date.
- For each order, the system counts unpaid installments and displays the number beside the order summary.

```python
@orders_bp.route('/main_page')
def main_page ():
    return render_template('main_page.html')
Windsurf: Refactor | Explain | Generate Docstring
@orders_bp.route('/credit_request', methods=['GET', 'POST'])
def request_credit ():
    if 'user_id' not in session:
        return redirect(url_for('auth.login'))
    user_id=session['user_id']

    if unpaid_exist(user_id):
        flash("❌ You have unpaid overdue installments. Please pay them before requesting new credit.", "danger")
        return redirect(url_for('orders.my_orders'))

    if request.method == 'POST':
        amount=float(request.form['amount'])
        user_id=session['user_id']
        user_purchase_power=Users.query.filter_by(user_id=user_id).first().purchase_power
        user_active_orders=Orders.query.filter_by(user_id=user_id,status='active').all()
        used_purchase_amount=0
        for order in user_active_orders:
            used_purchase_amount += sum(installment.payment_amount for installment in Installments.query.filter(Installments.order_id == order.order_id,
                                                                                                                 Installments.status == 'unpaid').all())

        free_purchase_amount=user_purchase_power-used_purchase_amount
        if amount>free_purchase_amount:
            flash(f"❌ Try to pay your unpaid installments and come back again", "danger")
            return redirect(url_for('orders.request_credit'))
        else:
            flash(f"✅ Your offer is  {amount:.2f}€", "success")
```

Figure 6.8: Credit Request Screenshot

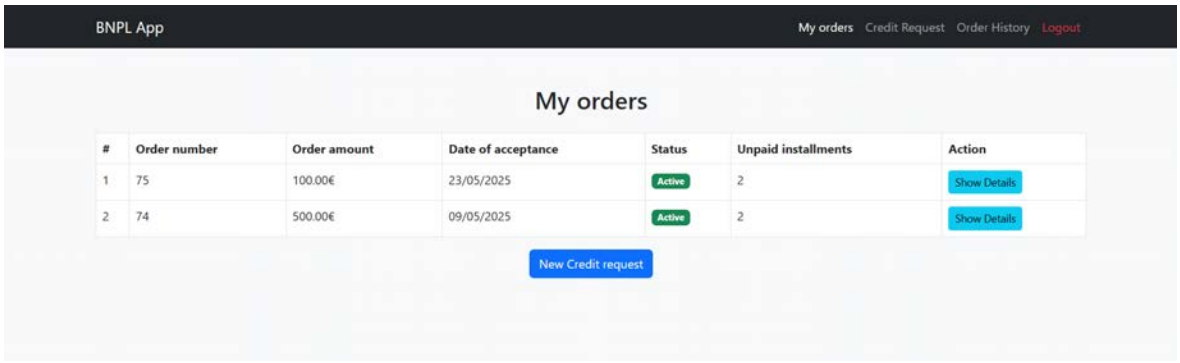

Figure 6.9: Active orders

3. `/order_history` – Viewing All Past Orders

Users can view their complete credit history, including inactive orders. This route displays all orders regardless of their payment status.

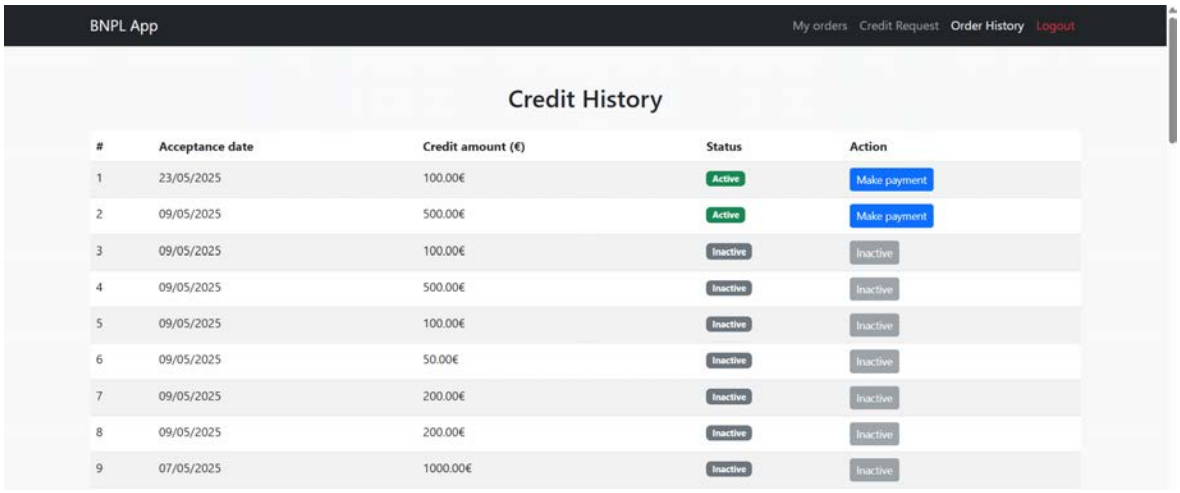

Figure 6.10: Credit History

4. `/my_orders/<order_id>` – Installment Payment Handling

This dynamic route allows users to make installment payments on a specific order:

- The system displays all unpaid or late installments and provides options to pay one or all at once.
- When a payment is submitted:
  - The `Installments` record is updated with the payment date and status.
  - If the total debt is cleared, the order status is changed to `inactive`.
  - If the customer paid the first installment, a new prediction is generated to re-evaluate the user's creditworthiness and adjust their `purchase_power`.
- The updated information is committed to the database using SQLAlchemy's `db.session.commit` method.

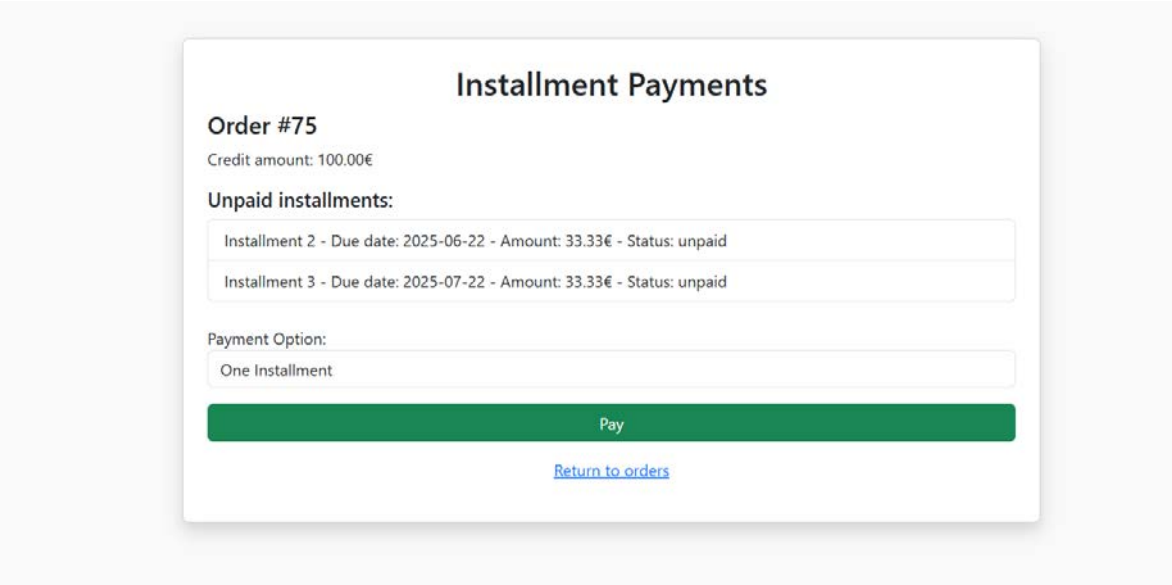

Figure 6.11: Installment Payment

Chapter 7

Behavioral Analysis and Purchase Power Updater

In this chapter we will explore with more details the logic behind the acceptance of credit request and the purchase power formulation. As was previously discussed, BNPL systems usually use the economic technique named ’slow and grow’.The ’slow and grow’ approach incrementally adjusts credit limits based on repayment behavior, reducing initial risk exposure and rewarding reliable users with higher purchasing power over time.” This aligns with Klarna’s ’Pay Later’ model, where initial limits are conservative but expand with consistent on-time payments ”(Klarna Annual Report, 2023).”

7.1 Behavioral Patterns

The policy of credit acceptances and purchase power changes is given by the following rules respectively:

- Installments that are past due, meaning the client didn’t respect his obligations, will block the next transactions until the payments are completed.
- No late fees are applied in case of delayed payments.
- Every installment is considered late until 30 days of delay, then it will be categorized as defaulted and seriously harm the user’s creditworthiness. Defaulted payments indicate really problematic behavior and are seriously considered.
- Number of delayed payments affect purchase power

- Latency duration also affects purchase power.
- On-time payments further enhance client's creditworthiness and purchase power.
- Late payments are categorized into four distinct tiers

Table 7.1: DPD Tiers and Risk Characterization

DPD Range	Tier	Risk Characterization
0-4 days	1	Minimal Risk (Grace Period)
5-10 days	2	Moderate Risk (Early Warning)
11-30 days	3	High Risk (Behavioral Concern)
30+ days	4	Critical Risk (Probable Default)

Before analyzing how these rules shape the user's credit limits, we must also address why late payments are such a critical metric in banking systems. When borrowers miss payment deadlines, it directly signals potential cash flow problems, reduced financial stability, or willingness to default, all of which elevate the lender's risk exposure.

- **Liquidity problems**.Banking systems rely on predictable cash flows to meet their financial obligations, including depositor withdrawals and interbank lending. When payments are delayed:
  1. Short-term funding gaps emerge.
  2. Asset-liability mismatches amplify (e.g., using short-term deposits to fund long-term loans).
  3. Contagion risk increases as banks reduce lending to preserve liquidity.
- **Operational and Systemic Risks**
  1. **Concentration risk**: Clustered late payments in specific sectors (e.g., real estate) can expose overexposed banks.
  2. **Feedback loops**: As banks tighten credit standards in response, marginal borrowers face refinancing crises.
  3. **Macroeconomic amplification**: The 2008 crisis showed how mortgage loan payment delinquencies can propagate through secularization chains.

7.2 Purchase Power Formula

To algorithmically shape the rules that were mentioned in the previous section the following code was constructed. The dynamic purchase power update mechanism evaluates user payment behavior through two key metrics: frequency of late payments and cumulative latency duration. The algorithm operates as follows:

```
weight_of_general_latency=1
weight_of_duration_of_latency=1
total_payments=0
total_late_payments=0
days_late=0
for order in orders :
    installments = Installments.query.filter_by(order_id=order.order_id,status='paid').all()
    for installment in installments:
        if installment.payment_date>installment.deadline_date:
            total_late_payments+=1
            total_payments+=1
            days_late+=(installment.payment_date-installment.deadline_date).days
        else:
            total_payments+=1


if total_payments>0:
    indicator1=total_late_payments/total_payments if total_payments>0 else 0
    indicator2=days_late/(total_payments*30) if total_payments>0 else 0


return float(amount)*(1-indicator1*weight_of_general_latency) * (1-indicator2*weight_of_duration_of_latency)
```

Figure 7.1: Purchase power updater

Key features:

- **Behavioral Metrics**: Combines both occurrence ($indicator1$) and severity ($indicator2$) of late payments
- **Penalty**: Past behavior is as important as the current installment. User with bad credit history need a bigger recovery phase to raise their creditworthiness.
- **Normalization**: Days late are scaled by 30-day periods for comparable units

Unlike traditional credit scoring models that evaluate individual transactions or payments, this algorithm assesses total repayment behavior - adjusting credit limits holistically based on the user's complete payment responsibility across all orders. This approach:

- Captures Reliability - Evaluates the borrower's overall responsibility rather than isolated events.

- Reduces Volatility - Prevents over-reaction to single late payments while maintaining accountability.
- Aligns with BNPL Economics - Reflects the portfolio-based risk management required for high-volume micro-credit

Chapter 8

Probability of Default

This term will be seen in our thesis as PD and is one of the most common metrics in credit reports and risk management. It describes an estimation of the likelihood that the borrower will be unable to meet his obligations. This metric has a lot of value as the feature cash flow and revenue of the financial institutions are totally dependent on PD. The efficiency however of the prediction is restricted by a lot of complex reasons:

- User's with no or limited history,
- Unexpected events,
- Market fluctuations,
- Model inability to capture all the parameters,
- Model biases,
- Behavioral changes
- Rapid change of borrower's economic stability,
- 'Black swan' events

This complexity explains why credit risk modeling has remained a cornerstone of financial research and practice for decades. It requires a combination of statistical rigor, domain expertise, and continuous evaluation of the models. Although the majority of this models are for traditional banking loans and corporate creditworthiness estimation , we will try to implement a PD prediction model for short-term loans and by using only machine-learning audit and no human intervention.

8.1 Dataset

The dataset that will be used for the development of the model is from traditional banking loans with the the outcome of the event as the target feature. Class ”0” for loans that were normally paid and class ”1” for the defaulted ones. As the target feature is the outcome of the loan and we want to predict the probability of default, we will transform this problem as the probability of the loan been classified in class ”1”.

Dataset Overview

- **LoanID**: Unique identifier for each loan application.
- **Age**: Age of the applicant (in years).
- **Income**: Annual income (in dollars).
- **LoanAmount**: Amount of the loan requested (in dollars).
- **CreditScore**: Credit score (300–850).
- **MonthsEmployed**: Employment duration (in months).
- **NumCreditLines**: Number of active credit lines.
- **InterestRate**: Loan interest rate (%).
- **LoanTerm**: Loan duration (in months).
- **DTIRatio**: Debt-to-Income ratio.
- **Education**: Highest education level.
- **EmploymentType**: Type of employment.
- **MaritalStatus**: Marital status.
- **HasMortgage**: Mortgage status (Yes/No).
- **HasDependents**: Dependents status (Yes/No).
- **LoanPurpose**: Purpose of the loan.
- **HasCoSigner**: Co-signer presence (Yes/No).
- **Default**: Loan default status (1/0).

8.2 EDA

Before proceeding to model development, it is essential to gain a thorough understanding of the dataset through Exploratory Data Analysis (EDA).

8.2.1 Statistics

For the numerical features, we will check for the most basic statistic metrics using the df.describe method.

Table 8.1: Summary Statistics for Key Numerical Features

Statistic	Age	Income	LoanAmt	CreditScore	MonthsEmp	CreditLines	Interest	LoanTerm	DTIRatio
Count	255347	255347	255347	255347	255347	255347	255347	255347	255347
Mean	43.50	82499.30	127578.87	574.26	59.54	2.50	13.49	36.03	0.50
Std	14.99	38963.01	70840.71	158.90	34.64	1.12	6.64	16.97	0.23
Min	18.00	15000.00	5000.00	300.00	0.00	1.00	2.00	12.00	0.10
25%	31.00	48825.50	66156.00	437.00	30.00	2.00	7.77	24.00	0.30
50%	43.00	82466.00	127556.00	574.00	60.00	2.00	13.46	36.00	0.50
75%	56.00	116219.00	188985.00	712.00	90.00	3.00	19.25	48.00	0.70
Max	69.00	149999.00	249999.00	849.00	119.00	4.00	25.00	60.00	0.90

For the categorical features we will check for the different values it’s feature takes in the dataset to further investigate it.

Table 8.2: Distinct Values for Categorical Features

Feature	Distinct Values
LoanPurpose	Other, Auto, Business, Home, Education
Education	Bachelor’s, Master’s, High School, PhD
EmploymentType	Full-time, Unemployed, Self-employed, Part-time
MaritalStatus	Divorced, Married, Single
HasMortgage	Yes, No
HasDependents	Yes, No
HasCoSigner	Yes, No

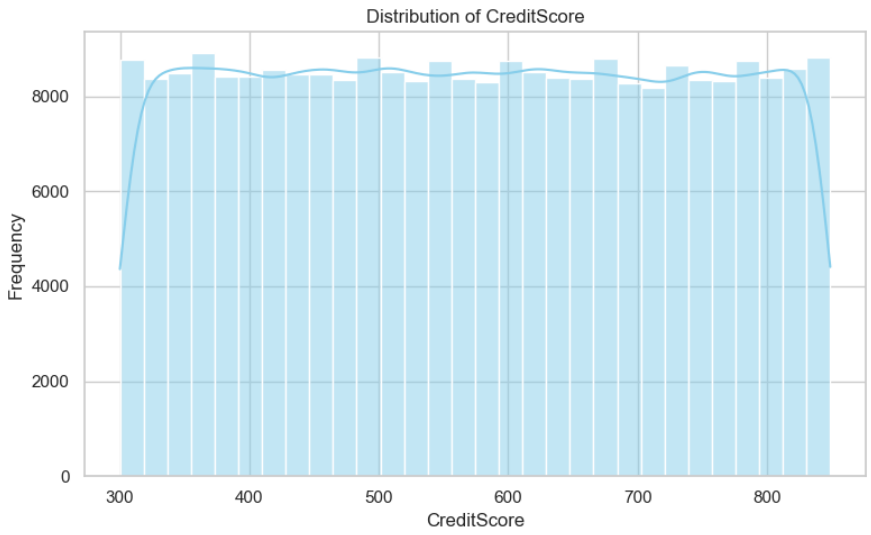

Figure 8.1: Credit score distribution

8.2.2 Feature Distribution

In figure 8.1 we can see that credit score has near perfect uniformal distribution.

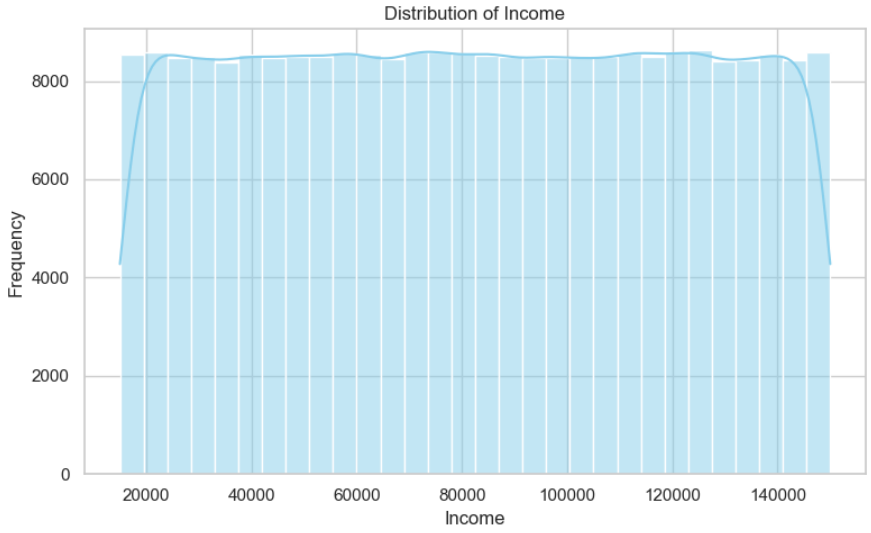

Figure 8.2: Income distribution

Both the most of the columns in the dataset.

Contrary to many financial datasets where features are often normally distributed, our exploratory analysis revealed that several key variables — including `Income`, `LoanAmount`, and `CreditScore` — have near -perfect uniform distributions.

A uniform distribution implies that values are spread relatively evenly across their range, with no dominant mode or strong central tendency. This may be the result of:

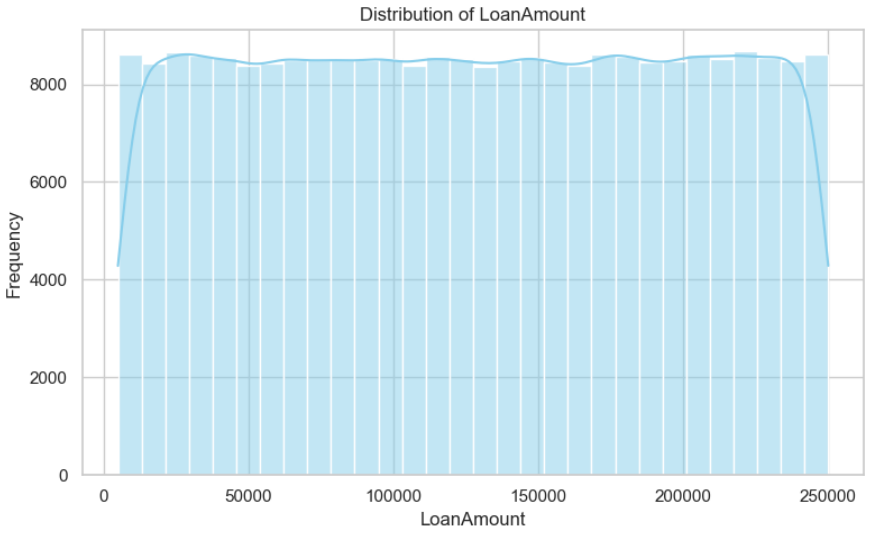

Figure 8.3: Loan amount distribution

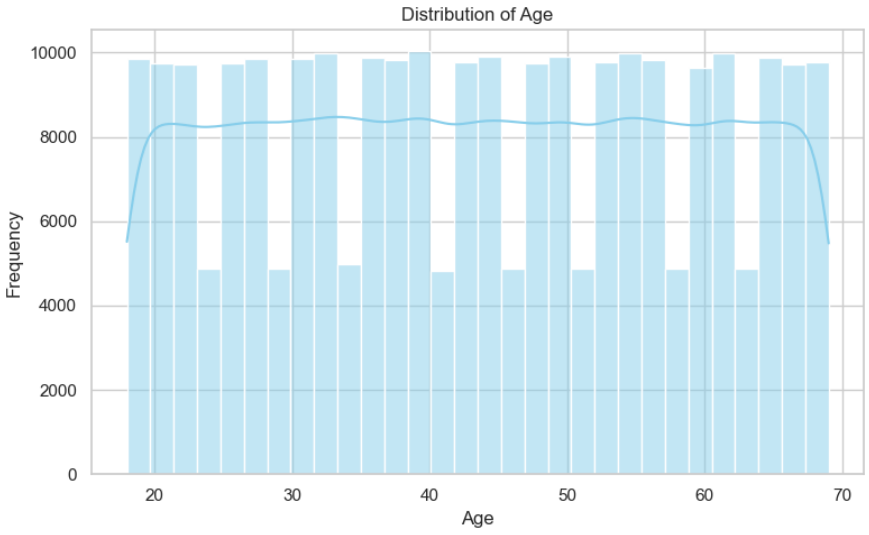

Figure 8.4: Age distribution

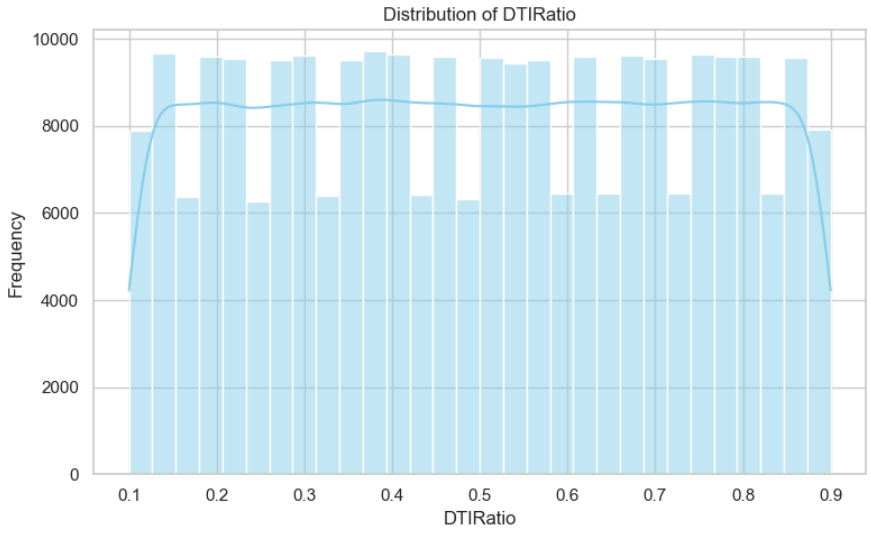

Figure 8.5: DTI distribution

- Controlled input ranges in data collection (e.g., standardized credit scoring systems)
- Feature engineering or preprocessing from upstream data pipelines
- The nature of synthetic or publicly available datasets designed to avoid biases

8.2.3 Default Distribution

As regards the target feature distribution : As seen, the dataset is significantly imbalanced:

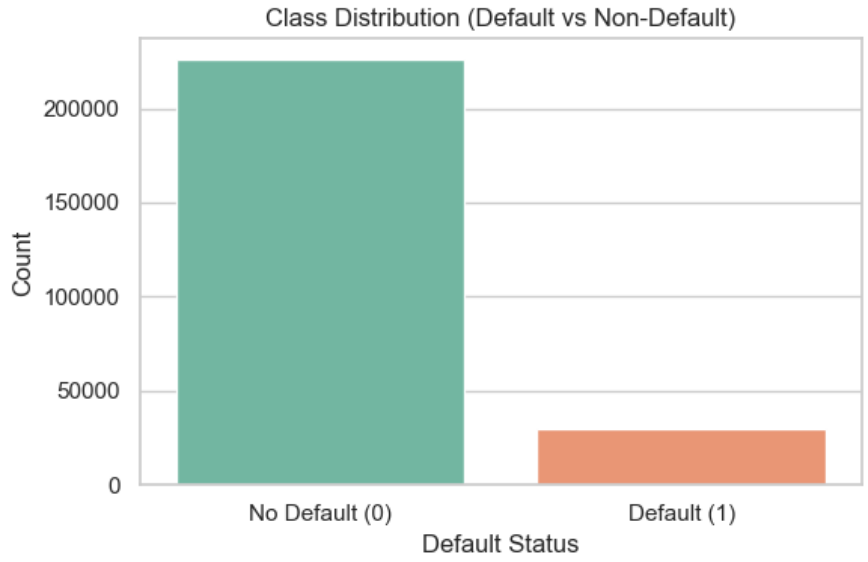

Figure 8.6: Default distribution

- **Non-default (0)**: 225,694 samples (88.39%)

- **Default (1)**: 29,653 samples (11.61%)

This imbalance can lead to biased predictions in favor of the majority class. //// To address the class imbalance in the target variable, we manually adjusted the decision threshold used to convert predicted probabilities into class labels. Instead of the conventional threshold of 0.5, we adopted a lower threshold (e.g., 0.2) to increase the model's sensitivity to default cases. This approach aims to reduce false negatives and improve recall for the minority class, which is critical in credit risk prediction. The threshold value was selected based on performance trade-offs evaluated through the precision-recall curve.

8.2.4 Missing Values

Upon evaluation, the dataset was found to contain **no missing values** across any of the features. This indicates a high level of data quality and consistency, potentially due to:

- Well-defined data entry protocols
- Controlled data sourcing from financial systems
- Pre-cleaned or engineered dataset prior to distribution

8.2.5 Correlation Matrix

The correlation matrix reveals that most numerical features in the dataset are largely uncorrelated with one another, showing near-zero Pearson coefficients. This suggests a low risk of multicollinearity, allowing us to retain all features for modeling.

Regarding the target variable `Default`, no feature shows a strong linear correlation. However, a few weak associations are worth noting:

- **Age** has a negative correlation of -0.17 with default risk, suggesting that younger applicants are slightly more likely to default.
- **InterestRate** shows a weak positive correlation of 0.13, indicating that higher-interest loans are slightly more associated with default.
- **Income** and **MonthsEmployed** both correlate negatively (-0.10), which aligns with the expectation that higher income and employment history may reduce risk.

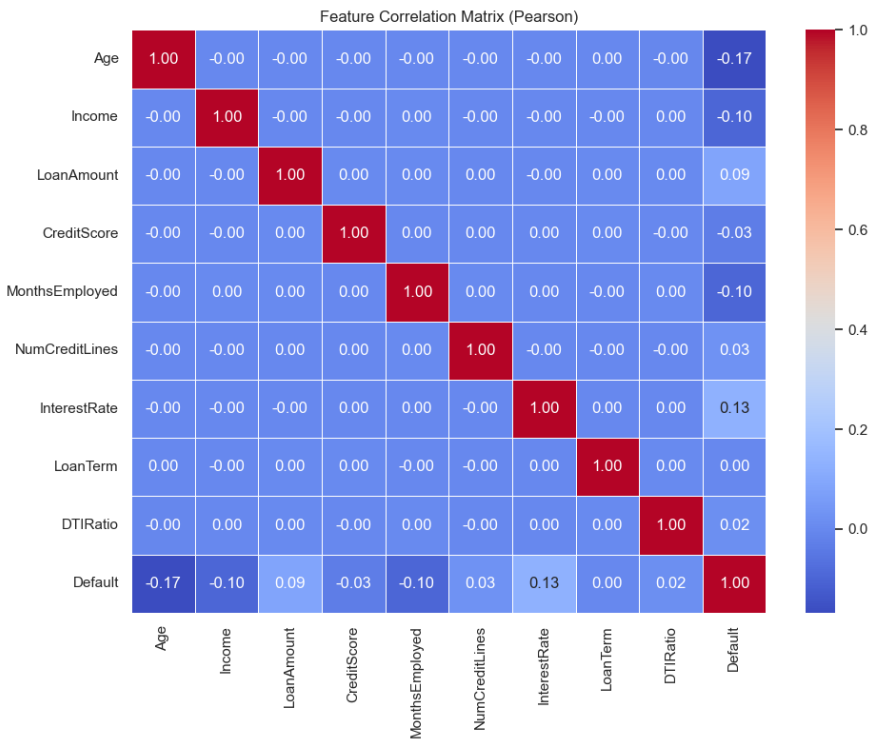

Figure 8.7: Feature correlation

All remaining features show negligible correlation with `Default`, supporting the idea that relationships in the data are likely non-linear and multi-dimensional.

8.3 Feature Preproccesing

Before jumping into modeling, we took several steps to prepare our dataset and ensure that the input features were in the right shape and format for training.

8.3.1 Credit Score Mapping

In our dataset, the credit score follows the normal FICO score ranges meaning the values range from 300 to 850 respectively. Credit score is used in this dataset as a metric of a user's creditworthiness, although the metric in this Thesis as it was previously introduced is the purchase-power. So the first step of feature processing is to map the credit score column in the dataset to the range of the values of the purchase-power using the min-max mapping method. Min-max mapping is a linear transformation technique used to rescale numerical features from their original range into a specified target range. This method preserves the relative relationships between values while ensuring that the transformed values fit within a

desired scale, which is especially useful for models sensitive to feature magnitude.

The general formula for min-max scaling is:

$$x' = \left(\frac{x - x_{\text{min}}}{x_{\text{max}} - x_{\text{min}}} \right) \cdot (r_{\text{max}} - r_{\text{min}}) + r_{\text{min}} \tag{8.1}$$

Where:

- x is the original feature value,
- x_{min} and x_{max} are the minimum and maximum values of the feature in the dataset,
- r_{min} and r_{max} define the new desired range (e.g., $[0, 1]$ or $[200, 1000]$),
- x' is the rescaled value.

This transformation ensures that the minimum value in the original feature maps to r_{min}, the maximum value maps to r_{max}, and all other values are linearly interpolated between them.

To further apply this algorithm in Credit score column we must calculate the ranges of the values that we aim to have. To implement this we will do SQL queries to find the min and max value of purchase power in the app. The following SQL query retrieves the minimum and maximum value from the `low_limit_values` column in the `country_macros` table:

```
SELECT MIN(low_limit_values),MAX(medium_limit_values)
FROM country_macros;
```

The query results are that the minimum value purchase power takes is 207.8 and maximum value 750. We will keep the minimum value in the range as it is but the maximum value needs more exploration.As the purchase power is changing through client exemplary behavior the values maybe easily pass 1000. We will assume that the maximum mapping value to dataset is 1200. this static decision is mandatory for the first development of model although, dynamic change of the upper limit must be introduces in production development.

8.3.2 Feature Scaling and Encoding

As the implementation we used in our first machine learning model for initial credit check, we will use Standard-scaler for feature scaling and Label-encoding for encoding the categorical values

Table 8.3: Example of Min-Max Mapping of Credit Scores

Original CreditScore	Mapped Value
520	520
458	389
451	375
743	991
633	759
720	943
429	328
531	543
827	1170
480	436

```
from sklearn.preprocessing import LabelEncoder
encoder=LabelEncoder()
X_cat['Education'] = encoder.fit_transform(X_cat['Education']) #highschool=0,bachelor=1,master=2,phd=3
X_cat['MaritalStatus'] = encoder.fit_transform(X_cat['MaritalStatus']) #single=1,married=0
X_cat['HasDependents']= encoder.fit_transform(X_cat['HasDependents']) #no=0,yes=1
X_cat['HasCoSigner']=encoder.fit_transform(X_cat['HasCoSigner']) #no=0,yes=1
X_cat['HasMortgage']=encoder.fit_transform(X_cat['HasMortgage']) #no=0,yes=1
X_cat['EmploymentType']=encoder.fit_transform(X_cat['EmploymentType']) #full time=0,part_time=1,self_employed=2,unemployed=3
```

Figure 8.8: Label encoding

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
```

Figure 8.9: Feature scaling

8.4 Model Evaluation

To evaluate the effectiveness of different machine learning algorithms for predicting credit risk, we implemented and tested several classification models. The goal was not only to assess their classification accuracy, but also their ability to provide meaningful and well-calibrated probability scores for default risk.

The models used include:

- **Logistic Regression**.
- **Random Forest**.
- **Gradient Boosting**.
- **XGBoost**.
- **MLP Classifier**.

The performance of each model was evaluated on the test set using the ROC AUC score as well as precision, recall, and F1-score for the minority class (defaults). These results are summarized in Table 8.4.

Table 8.4: Model Performance Comparison on Test Set

Model	ROC AUC	Precision (1)	Recall (1)	F1-Score (1)
Gradient Boosting	0.7566	0.65	0.05	0.10
Logistic Regression	0.7513	0.22	**0.70**	0.33
MLP Classifier	0.7413	0.53	0.09	0.16
XGBoost	0.7405	0.23	0.62	**0.34**
Random Forest	0.7340	**0.69**	0.02	0.05

While ROC AUC scores were relatively close across models, Logistic Regression and XGBoost stood out for achieving a good balance between recall and F1-score for the minority class. Notably, tree-based models like Random Forest and Gradient Boosting achieved high precision but extremely low recall, indicating they were not effective in identifying default cases.

8.4.1 Focus on Probability Estimation:

In this work, our primary objective is not to produce a binary classification outcome (i.e., default or not), but rather to estimate the **probability of classified in default class**. This probability output is more useful in risk-based decision making, as it allows the system to rank applicants by risk level.

- **Balanced Performance:** Logistic Regression and MLP provided the most balanced trade-off between sensitivity (recall for defaults) and specificity. In contrast, ensemble models such as Random Forest and Gradient Boosting exhibited high overall accuracy but failed to identify a meaningful number of defaults (very low recall).
- **Class Imbalance Handling:** Logistic Regression allowed for easy integration of class weights and threshold tuning, which significantly improved recall on the minority class without severely sacrificing precision. Similarly, MLP was flexible in capturing non-linear relationships, showing better recall for defaults compared to tree-based models.
- **Interpretability and Explainability:** Logistic Regression offers a clear mathematical interpretation of feature importance through its coefficients, making it suitable for explainable AI (XAI) applications in regulated financial environments.
- **Model Flexibility:** The MLP model, although less interpretable, benefits from its capacity to capture complex feature interactions and nonlinearities, which is useful given that most features in our dataset are uniformly distributed and lack strong linear correlation with the target.
- **Robustness Against Overfitting:** Both selected models showed consistent ROC AUC values on the test set (Logistic Regression: 0.75, MLP: 0.74), indicating good generalization and robustness without excessive complexity.

Ultimately, the combination of Logistic Regression (for audit-ability and baseline confidence) and MLP (for enhanced pattern recognition) enables both transparency and performance in our model development.

8.5 Logistic Regression Model

The first thing is to check for the feature coefficients.

Table 8.5: Feature Importance Coefficients

Feature	Coefficient	Absolute Coefficient
Age	-0.584881	0.584881
InterestRate	0.450589	0.450589
MonthsEmployed	-0.332468	0.332468
Income	-0.317988	0.317988
LoanAmount	0.282576	0.282576
EmploymentType	0.146707	0.146707
HasCoSigner	-0.137696	0.137696
CreditScore	-0.120297	0.120297
HasDependents	-0.119874	0.119874
NumCreditLines	0.099093	0.099093
MaritalStatus	0.087267	0.087267
HasMortgage	-0.078211	0.078211
Education	-0.074511	0.074511
DTIRatio	0.061444	0.061444
LoanTerm	-0.000441	0.000441

After investigating the coefficients table we will drop the below features:

- DTI ratio
- LoanTerm
- Education

This dropout cost 4% decline in the ROC-AUC score.

Probability Evaluation

In probability-based classification tasks such as default prediction, it is not enough for a model to simply classify correctly. It must also assign meaningful probabilities that reflect real-world likelihoods. To evaluate and improve the quality of probability estimates, we used the following two tools:

- **Brier Score**: A proper scoring rule that measures the mean squared difference between predicted probabilities and the actual binary outcomes. This score evaluates both the calibration and the sharpness of predictions.
- **Calibration**: A post-processing technique that adjusts raw predicted probabilities to better align with true event frequencies. We applied *isotonic regression*, a non-parametric method that fits a non-decreasing function to the model's outputs.

Brier Score Formula: Given N predictions, the Brier Score is computed as:

$$\text{Brier Score} = \frac{1}{N}\sum_{i=1}^{N}(\hat{p}_i - y_i)^2 \tag{8.2}$$

Where:

- $\hat{p}_i$ is the predicted probability of the positive class,
- y_i is the actual outcome (1 for default, 0 otherwise),
- Lower values indicate better probability estimates (perfectly calibrated predictions would yield a score of 0).

Calibration Overview: Let $f(x)$ be the raw score output from the model (e.g., MLP). Calibration learns a function $g(f)$ such that the adjusted probability $\hat{p} = g(f)$ better represents the observed likelihood of the event:

$$\hat{p} = g(f(x)) \quad \text{where } g \text{ is learned via isotonic regression} \tag{8.3}$$

Isotonic regression ensures g is monotonic and non-parametric, which allows flexible calibration of poorly distributed probability outputs.

Example of Calibration: To make calibration more understandable , consider a weather forecast scenario:

Suppose a model predicts the probability of rain for 5 consecutive days as follows:

Day	1	2	3	4	5
Predicted Probability of Rain	0.9	0.7	0.6	0.4	0.2
Observed Outcome (Rain = 1)	1	1	0	0	0

Although the predictions are mostly correct in terms of ranking (higher probability when it rains), they may not be **calibrated**. A well-calibrated model should predict probabilities that match actual frequencies. For example:

- If the model predicts 0.7 probability of rain on 10 different days, it should rain on approximately 7 of them.

- In the table above, the model predicted rain with 60% confidence on Day 3, but it did not rain — if this pattern repeats, the model is **overconfident**.

Calibration adjusts these predicted probabilities so that they better reflect reality. For instance, the model might learn that when it predicts 0.6, rain actually happens only 40% of the time — and thus recalibrates future 0.6 predictions to 0.4.

This concept directly applies to credit risk: if a model says a customer has a 20% chance of default, we expect that in a group of 100 such customers, about 20 will default. If 40 do, the model is poorly calibrated.

Evaluation Results Non-Calibrated Model:

- ROC AUC: 0.7198
- Brier Score: 0.2141
- Precision (class 1): 0.13, Recall: 0.98, F1-score: 0.22

Calibrated Model

- ROC AUC: 0.7197
- Brier Score: **0.0946** (significantly improved)
- Precision (class 1): 0.28, Recall: 0.36, F1-score: 0.31

Conclusion: Although calibration did not improve the ROC AUC score, it led to a substantial reduction in the Brier Score. This indicates that the calibrated MLP model outputs probability estimates that are far more reliable and better aligned with true default risk.

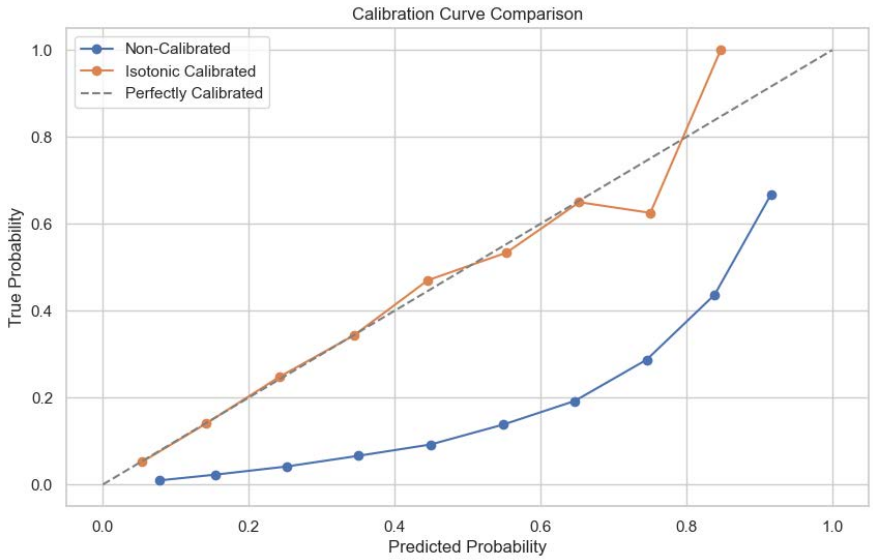

Figure 8.10: Calibration Curve

Calibration curve: We can observe from the figure 8.10 that the pre-calibration model predicted higher probabilities than were actually observed. Until 0.6 calibrated curved near perfectly fits the diagonal line . Although the observation outlines some issues.

- After 0.6 model becomes overconfident again but with not dramatic changes.

- After a certain point, model becomes under confident.

In a traditional banking loan credit system this would oppose a big problem. Although in the nature of our thesis, where this model will be applied to low amount, short-term credit there is near 0% probability that the predicted and actual probabilities would exceed the 40% scope.

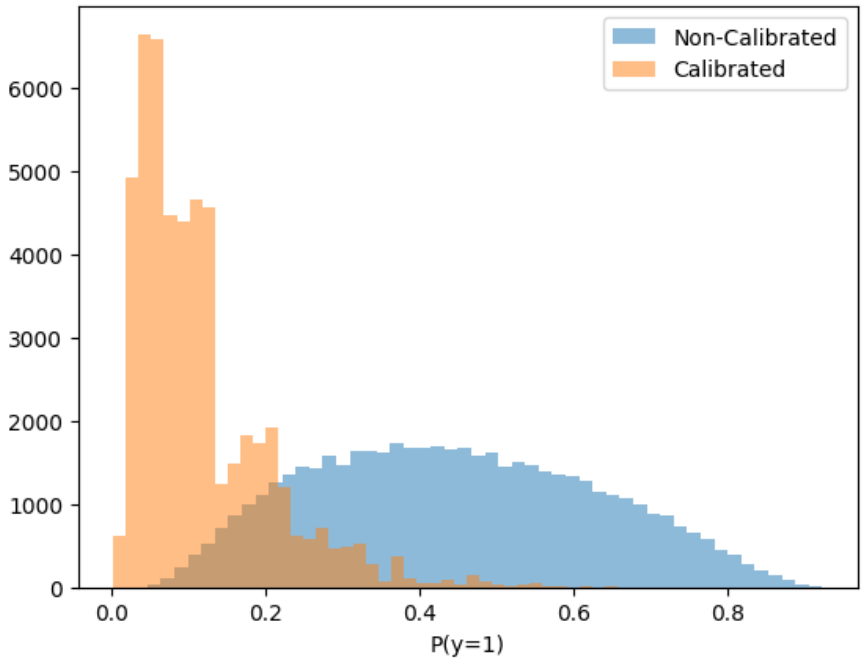

Figure 8.11: Probability distribution

8.5.1 Validation of Probability Distribution

To validate the probability distribution in our isotonic calibrated model we will use Lending clubs dataset.

Table 8.6: Distribution of Predicted Default Probabilities in LendingClub Dataset

PD Range	Number of Loans	Percentage
0.00–0.05	120,000	60%
0.05–0.10	40,000	20%
0.10–0.20	25,000	12.5%
0.20–0.30	10,000	5%
0.30–1.00	5,000	2.5%

As we can see 92.5% of loans are in the range of 0 to 0,2 probability of default. In our case we will check for the percentage of loan that are in the range 0-0.2. This results validate that

Table 8.7: Percentage of Loans with PD $\leq$ 0.2

Model	**Loans with PD $\leq$ 0.2**
Non-Calibrated	14.28%
Calibrated (Isotonic)	84.00%

the probability distribution in which the model is trained is in bibliographically very good range.

8.6 Multilayer Perceptron (MLP)

As our application is based on short-term ,low amount loans we will introduce MLP as the model that will be feed with new data from the application we developed and we will approximately catch non linear relationships that customer behavior oppose. The **Multilayer Perceptron (MLP)** is a class of feedforward artificial neural network that excels at modeling complex nonlinear relationships in data. Unlike linear models, MLPs can approximate virtually any continuous function given sufficient network capacity [18].

Architecture

An MLP consists of:

- An **input layer** (features)
- One or more **hidden layers** with nonlinear activation functions
- An **output layer** (prediction)

Why MLP for Nonlinearity?

MLPs capture nonlinear relationships through:

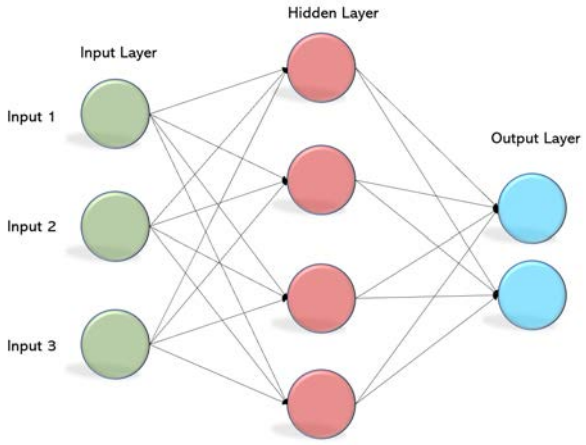

Figure 8.12: Typical MLP architecture with one hidden layer

- **Activation functions** (ReLU, sigmoid, tanh) that introduce nonlinear transformations:

$$\sigma(z) = \frac{1}{1 + e^{-z}} \quad \text{(Sigmoid)}$$

- **Hidden layer composition** enabling function approximation:

$$f(\mathbf{x}) = W_2\sigma(W_1\mathbf{x} + \mathbf{b}_1) + \mathbf{b}_2$$

  where W are weights and $\mathbf{b}$ are biases

- **Universal approximation** capability [19]

Advantages for Financial Risk Modeling

- Handles complex interactions between features
- Adapts to non-monotonic relationships (e.g., age vs default risk)
- Outperforms linear models when decision boundaries are nonlinear

8.7 MLP Implementation

We started from the already preprocessed and scaled dataset that had been used in our Logistic Regression model. This ensured consistency across models and eliminated the need for additional feature scaling.

Encoder and Dimensionality Reduction

As was previously discussed our features show no linear relationships with each other and low correlation to the target feature. First step of catching non linear relationship in the dataset is by training and autoencoder. The encoder is a neural network that learns a compressed embedding of the original data by minimizing the reconstruction error.

Mathematically, the encoder maps input $\mathbf{x} \in \mathbb{R}^n$ to a latent space $\mathbf{z} \in \mathbb{R}^d$, where $d < n$:

$$\mathbf{z} = f_{\text{encoder}}(\mathbf{x}) = \sigma(W\mathbf{x} + \mathbf{b})$$

where W and $\mathbf{b}$ are trainable weights and biases, and σ is a non-linear activation function.

We tested various encoding sizes (4, 8, 16, 32) and evaluated their impact on downstream classification by feeding the compressed outputs into a logistic regression model. The performance was measured using the ROC AUC metric. This approach allowed us to identify the optimal encoding dimension that balances information preservation with model simplicity.

The results were visualized in a plot of AUC scores against encoding dimensions to guide our selection.

Table 8.8: Autoencoder Classification Performance by Latent Dimension Size

Latent Dimension	AUC Score
4	0.5847
8	0.6697
16	0.7511
32	0.7513

Note: Performance measured by Area Under Curve (AUC) metric on test set.

The experimental results indicate that an encoding dimension of 16 yields optimal tradeoff withing performance and overfitting for the autoencoder model.

8.7.1 MLP Architecture

The MLP model was implemented using Keras as a `Sequential` model. It consisted of:

- One dense layer with 64 neurons (ReLU activation),

- Dropout layer (30%) to prevent overfitting,
- Another dense layer with 32 neurons (ReLU),
- Final dropout (20%),
- An output layer with sigmoid activation for probability output.

Training Setup The model was compiled with:

- Loss function: `binary_crossentropy`,
- Optimizer: `Adam` (learning rate = 0.001),
- Metrics: `accuracy` and `AUC`.

We trained the model for up to 50 **epochs**. An epoch refers to one full pass over the training dataset. To avoid overfitting, we used **early stopping** with a patience of 5 epochs — meaning training would stop early if the validation loss didn't improve for 5 rounds.

8.8 Evaluation

Table 8.9: MLP Classification Report on Test Set

Class	Precision	Recall	F1-score	Support
0 (Non-Default)	0.93	0.83	0.88	45170
1 (Default)	0.29	0.51	0.37	5900
Accuracy		0.80		
Macro Avg	0.61	0.67	0.62	51070
Weighted Avg	0.85	0.80	0.82	51070

The MLP model achieved an overall accuracy of 0.80 on the test set. For class 0 (Non-Default), it showed strong performance with 0.93 precision and 0.83 recall, while for class 1 (Default) the metrics were lower at 0.29 precision and 0.51 recall. The weighted average F1-score was 0.82, indicating better performance on the majority class, with the macro average F1-score of 0.62 reflecting the challenge of balanced classification across both classes.

The model demonstrated significantly better performance in identifying non-default cases compared to default cases.

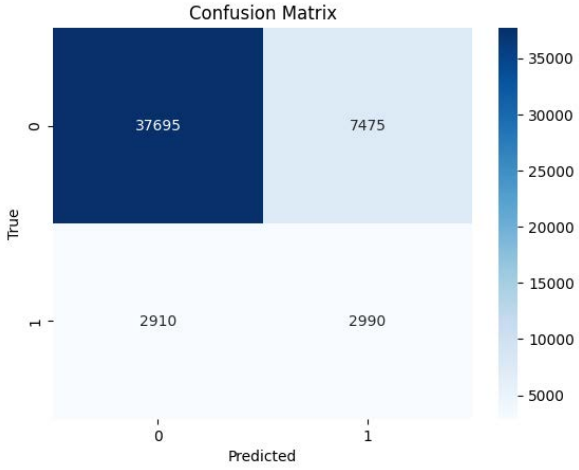

Figure 8.13: Confusion matrix

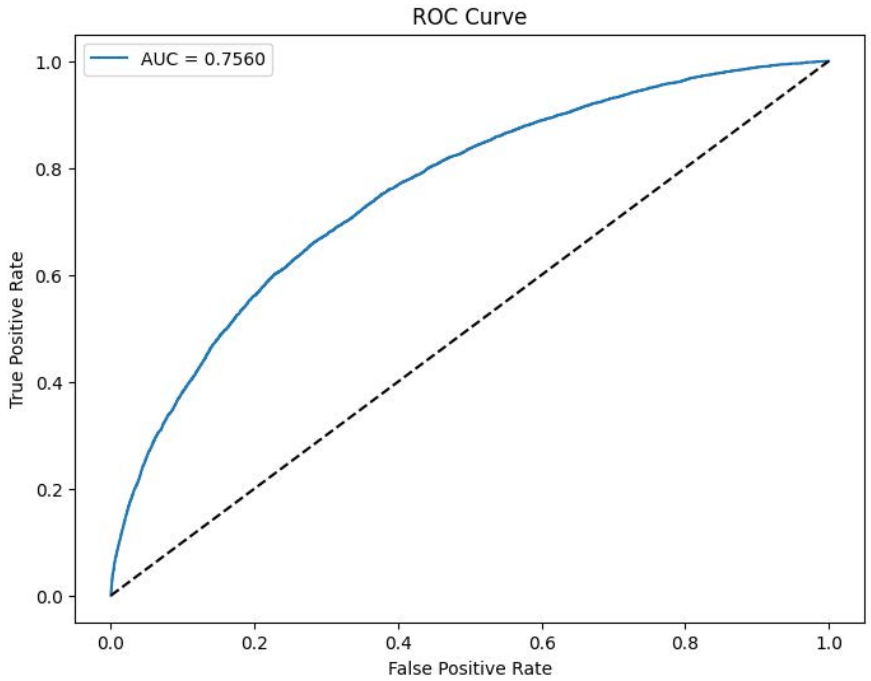

Figure 8.14: ROC-AUC Curve

The ROC curve shows the MLP model's performance with an AUC (Area Under the Curve) score of 0.7560, indicating a moderately strong discriminatory ability in classifying the two classes. While not exceptional (AUC > 0.9), this suggests the model performs better than random chance (AUC = 0.5) and has reasonable predictive power

8.8.1 PD distribution

After evaluating some of the performance metrics of the model we should also in this model find the probability of default distribution. The plot 40 shows that the predicted probability distribution seems very similar to our logistic regression model that has already been analyzed and cross validated with real world application data.

8.8.2 Probability Calibration

In our logistic regression model we introduced calibration to enhance its accuracy and performance . We will start examining the need for calibration in our MLP architecture . As we can see it well fits the perfect calibrated line. That is one of the reasons that calibration is not optimal here. A second and more important reason is that as long as the MLP model will be retrained with new app-born data, calibration will introduce extreme bias to the predictions.

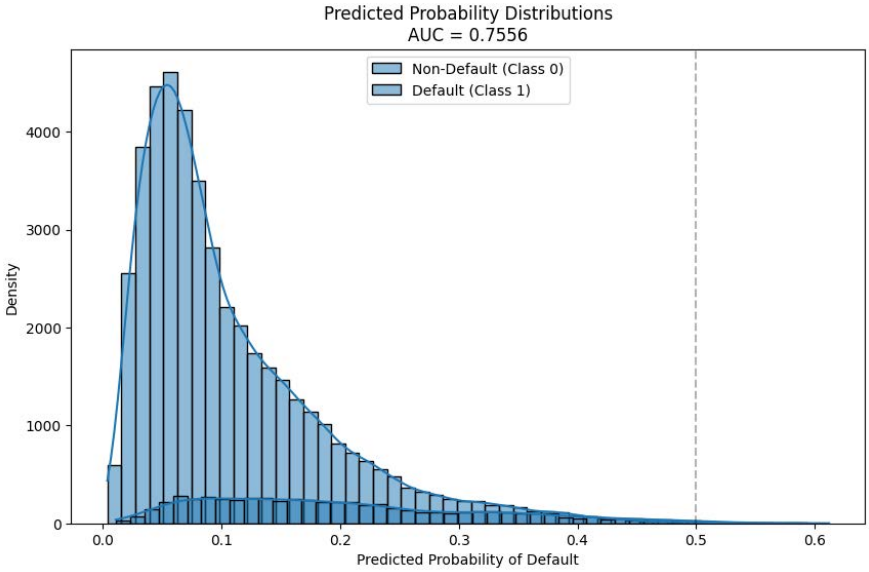

Figure 8.15: PD distribution MLP

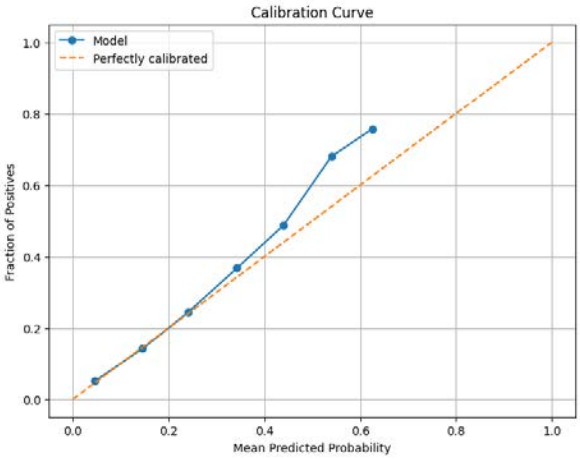

Figure 8.16: MLP Calibration Curve

This is because calibration works by finding the median actual probability of similar instances and validates the predicted one. In our case where similar instances (same clients) will be entered into the dataset calibration will not efficiently work.

Chapter 9

Model retraining

As we previously said the dataset that the models were trained is about traditional banking requests like car and mortgage loans. The huge gaps in the loan amount between traditional loans and BNPL loans may propose prediction difficulties. Small amount differences for the original dataset may be huge as regards our BNPL system. Let's say we are trying to predict the probability of default on a specific loan where :

1. Loan amount = 500
2. Credit score = 400
3. User income = 20000

This loan gives a probability of default of 10% in our model. If we again try to compute the probability of default but this time with a loan amount of 300, the current model won't be triggered by this difference although it is a 40% decrease. This issue makes it essential to retrain the model with application-born data.

9.1 Prediction Logs

As part of our retraining policy, we developed the Prediction Logs table that was discussed in the database section. The meaning of developing this table is to store the predictions of the models until the real outcome is observed. As the tables imply we are storing the:

1. **Userid**. Userid is stored because when the actual outcome of this specific order is observed this prediction log should be inserted into the model. The model is trained

Table 9.1: User Default Prediction Results

User	Order ID	ID	Purchase Power	LR Prob	MLP Prob	Actual	Amount	Timestamp
1	1	1	373	0.0738	0.0712	0	133.33	2025-05-07 12:08:14

LR Prob: Logistic Regression Probability of Default

MLP Prob: Multi-Layer Perceptron Probability of Default

with many different demographic and economic features. So userid is the foreign key that will give us access to this user data.

2. **Orderid**. Orderid is also stored for the same reason as Userid.

3. **Logid**.Logid is the table's primary key and its usage will be highlighted in the next sections.

4. **Purchase power** . Although purchase power can be obtained through the user's foreign key, we must store the specific purchase power the user had when he made this order. Changes in the purchase-power in the meantime will feed the model with no meaningful data,

5. **LR prob** . This is the predicted probability of the Logistic regression model

6. **MLP prob** . The prediction of the MLP model

7. **Actual** . It is the observed outcome of the loan (0 or 1)

8. **Amount** . The amount feature characterizes the installment amount

9. **Timestamp** . The time the prediction was made.

Whenever the outcome of the installment is observed the actual column changes and it is ready to be inserted into the retraining process

9.2 Log to Excel

In this part of the retraining process, we will implement the code that extracts the logs from the database, processes them, and feed them in a CSV file ('retraining_data.csv)

Processed ids

```
def get_processed_ids(self):
    if not os.path.exists(PROCESSED_IDS_FILE):
        return set()
    with open(PROCESSED_IDS_FILE, 'r') as f:
        return set(line.strip() for line in f if line.strip())
```

This code checks for the already processed ids. This is mandatory as we do not want duplicates in the retraining dataset.

Save processed ids

```
def save_processed_ids(self, processed_ids):
        with open(PROCESSED_IDS_FILE, 'a') as f:
            for log_id in processed_ids:
                f.write(f"{log_id}\n")
```

This codes stores the recently processed ids in the txt file.

Extract new data

```
def extract_new_data(self, batch_size):

        processed_ids = self.get_processed_ids()
        new_processed_ids = set()

        with self.app.app_context():
            query = Prediction_Logs.query.filter(
                Prediction_Logs.True_label != None,
                ~Prediction_Logs.log_id.in_(processed_ids)
            ).order_by(Prediction_Logs.timestamp).limit(batch_size)

            new_logs = query.all()

            if not new_logs:
                return None, set()

                data = []
            for log in new_logs:
```

```
19                  user = Users.query.filter_by(user_id=log.user_id).
                        first()
20                  if user.marital_status == 'Married':
21                      marital_status = 1
22                  else:
23                      marital_status = 0
24                  if not user:
25                      continue
26                  #...........
27                  ........... # df making
28                  ...........
29                  return pd.DataFrame(data, columns=columns),
                        new_processed_ids
```

This is the main method of the class, which takes the logs from the database and transforms them into a data frame that now can be efficiently inserted into the CSV file. The other methods are going to be documented and not presented as a code

- **Append to csv** . Takes the data frame with the new processed logs and append it in the CSV.
- **Process batch** . Takes the logs and processes them as a batch. In our project batch size has been initialized to 10. Meaning each time the code works, it processes 10 prediction logs.
- **Create scheduler**. Organizes the total implementation and makes the scheduling of the process. In this project, these codes run each minute.
- **Run scheduler**. A function that as the name implies starts the scheduler.

9.3 Csv to Model

In this part of the retraining process we implemented code that takes the data from the csv and feed them in the MLP model. For this part the file schedule retraining was constructed This file consists of the following functions:

- **Initial training**. This function checks if the initial model already exist in the app environment , if it does it stops , if the model does not exist it starts the initial training that is in the initial_model_train file.

- **Run retraining**. The run retraining function manages the model retraining process when new data becomes available. It first checks for the existence of a retraining_data.csv file and verifies it contains sufficient data (at least 1KB, roughly 100 samples). When these conditions are met, it initiates the retraining by launching a subprocess to execute retrain.py

```
file_size = os.path.getsize('retraining_data.csv')
        if file_size > 1024:  # At least 1KB of data (~100
           samples)
            log("New data detected, starting retraining...")
            try:

                process = subprocess.Popen(
                    ['python', 'retrain.py'],
                    stdout=subprocess.PIPE,
                    stderr=subprocess.PIPE,
                    universal_newlines=True
```

9.4 Retraining

The `retrain()` function executes a batch update of the credit risk prediction model. It requires:

- Input: `retraining_data.csv` (minimum 10 samples)

- Pretrained components: Scaler (`nn_scaler.joblib`), encoder (`encoder.h5`), and classifier (`mlp.h5`)

The procedure follows:

1. **Data Preparation**: Loads the dataset, separates features (X) and labels (y), and removes timestamps

2. **Feature Processing**:
    - Updates the scaler with new data statistics
    - Applies standardization: $X_{\text{scaled}} = \frac{X-\mu}{\sigma}$
    - Encodes features: $X_{\text{encoded}} = \text{Encoder}(X_{\text{scaled}})$
3. **Model Update**: Retrains the MLP classifier for 10 epochs (batch size=64) on X_{encoded}
4. **Persistence**: Saves the updated classifier and removes processed data

Key technical specifications:

- Input dimensionality: 15 features + 1 label
- Training regime: Adam optimizer, binary cross-entropy loss
- Validation: Automatic via 10% holdout (Keras default)

Further evaluation of the impact of the model retraining on the predictability and efficiency of the model will be presented in the testing section.

Chapter 10

Risk management

In the context of credit risk assessment and loan portfolio management, three fundamental risk metrics are implemented and analyzed: **Expected Credit Loss (ECL)**, **Stress Testing**, and **Risk-Adjusted Return (RAR)**. These measures are very critical for financial institutions in decision-making, capital allocation, and compliance with regulatory standards.

10.1 Expected Credit Loss (ECL)

ECL. We introduced this metric in the theoretical background framework in this thesis. Its importance stems from several key factors:

- Regulatory Compliance (IFRS 9 & Basel III). For example, if a bank fails to calculate ECL properly, it may face penalties from regulators or insufficient capital gains during economic crisis or downturns.

- Proactive Risk Management. A lender using ECL can detect rising default risks in a portfolio and adjust lending policies before losses occur.

- Accurate Financial Reporting. If a fintech company underestimates or overestimates ECL, it may report false profits, misleading investors.

- Stress Testing & Scenario Analysis

$$ECL = PD \times EAD \times LGD$$

where:

- **PD (Probability of Default)**: In our cases, the mean value of the predicted probabilities of the logistic regression and MLP model is used.
- **EAD (Exposure at Default)**: The total value that is at risk at the time of default. In our case, is the loan amount for every order.
- **LGD (Loss Given Default)**: The proportion of the exposure that is expected to be lost after recovery efforts. When a loan is in a defaulted state the company that issued this loan is hiring agencies to recover some of the remaining amount of loan value. This agencies are called debt collectors. Generally, the idea of this is that the debt collectors get all this amount and then earn 30-60% as a commission. The LGD is the result of

  $$LGD = 1 - RR$$

  .where:

  – RR is the percentage of the exposure recovered after default ($0 \leq RR \leq 1$)

  So if the company gets the full amount and pays 45% to collectors the recovery rate is 55%, so the LGD is the same as the debt collectors commission. In this implementation, LGD is fixed at 45%.

10.2 Stress Testing

Stress Testing is a simulation technique used to assess the performance and stability of a credit portfolio under adverse economic scenarios. It provides insight into how risk metrics would behave during financial downturns or crises. The stressed ECL is computed as:

$$ECL_{stress} = ECL_{base} \times \alpha_{PD} \times \alpha_{LGD}$$

Where α_{PD} and α_{LGD} are stress multipliers applied to reflect the worsening of credit conditions.

A typical stress testing table might include scenarios such as:

Scenario	PD Multiplier (α_{PD})	LGD Multiplier (α_{LGD})
Baseline	1.0	1.0
Recession	1.7	1.3
Severe	2.1	1.6

This method is essential for regulatory reporting (e.g., Basel III) and for identifying vulnerabilities in the loan book under hypothetical but plausible economic shocks. In this thesis we are using 2 scenarios

- Recession(1.5, 1.2)
- Severe(2,1.5)

10.3 Risk-Adjusted Return (RAR)

RAR evaluates the profitability of a loan portfolio while accounting for the expected credit losses. It is a performance metric that allows institutions to compare the returns of different credit products.:

$$RAR = \frac{\sum(1+r) \times EAD - \sum ECL}{\sum EAD} \times 100\%$$

where $r = 3.5\%$ is the assumed base interest rate applied to the exposure.

RAR is a valuable metric for credit pricing, capital allocation, and strategic planning, as it balances the trade-off between profitability and risk.

10.4 DPD Distribution

DPD or days past due distribution is a method of categorizing active loans into delinquency stages based on payment delays, providing critical insights into portfolio management. In this thesis, we implemented the dpd distribution function with 4 bins.

- 0 to 4 days of delay
- 5 to 10
- 11 to 30
- >30

All these metrics are shown in the statistics route in our application

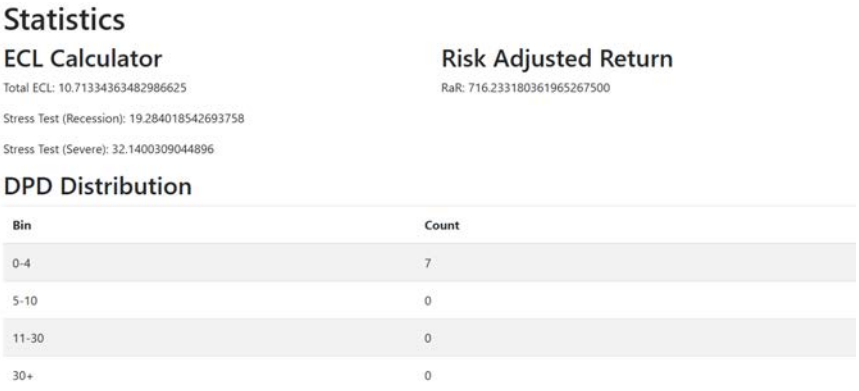

Figure 10.1: Statistics screenshot

Chapter 11

Testing and Results

We will start examining our workflow pipelines and then increment testing to the whole system.

11.1 Prediction System

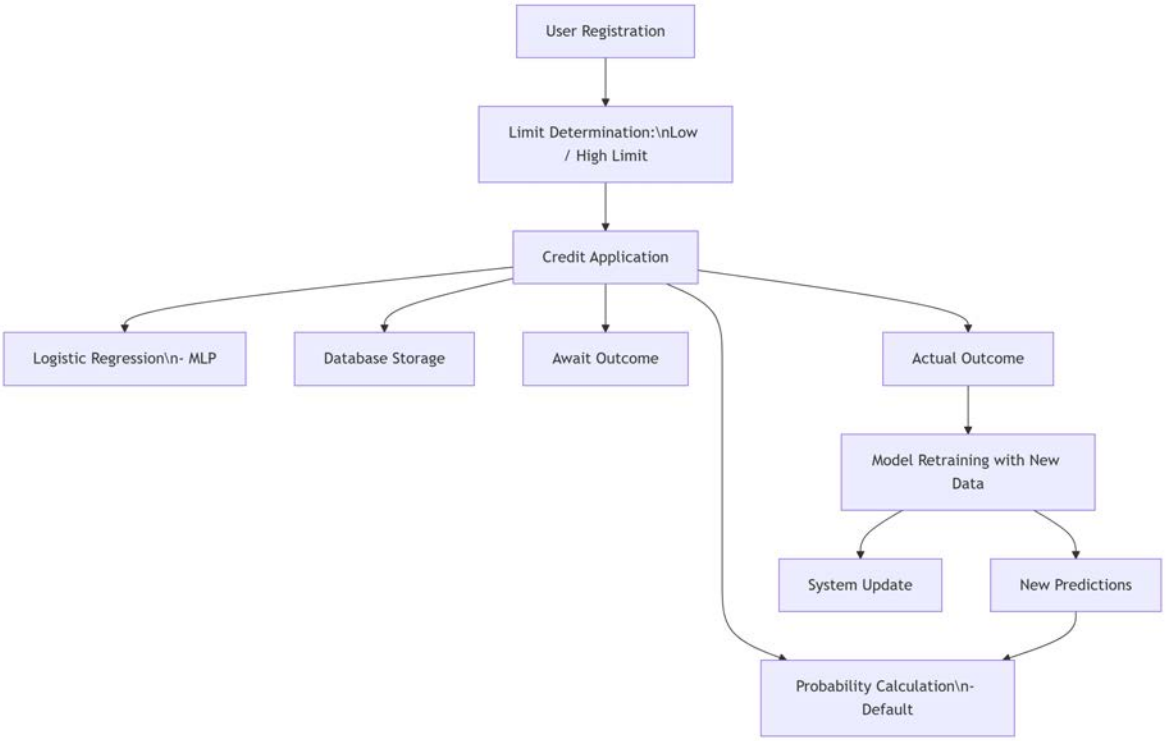

Figure 11.1: Prediction Pipeline

We will apply to this pipeline 3 different individuals and 3 different loans one of which will be used to test the retraining efficiency.

Table 11.1: User Profile Characteristics

Attribute	**User 1**	**User 2**	**User 3**
Age	24	32	29
Income (€)	45000	20000	55000
Ethnicity	Caucasian	Asian	African American
Nationality	USA	Greece	Cyprus
Education Years	16	18	14
Cards	2	1	3
Marital Status	Single	Married	Married
Is Student	No	Yes	No
Gender	Male	Female	Male
Months Employed	48	6	120
Mortgage	No	Yes	No
Number of Dependents	0	1	2
Employment Type	part-time	full time	self employed
Debt	3500	22000	0

Some of these characteristics will be used for the initial credit check. The results of the purchase power assignments are shown in the table below :

Table 11.2: User Limit Category and Purchase Power

User	Limit Category	Purchase Power (€)
User 1	Medium	577.98
User 2	Low	287.61
User 3	Medium	557.34

For User 3 we will try different credit requests and then the predicted probabilities of each one will be displayed.

Table 11.3: Installment Predictions and Outcomes for User ID 12

User	Purchase Power	LR PD	MLP PD	Outcome	Amount (€)	Timestamp
User3	557.00	0.0304	0.1023	0	100.00	2025-05-27 18:27:07
User3	707.00	0.0257	0.099	0	100.00	2025-05-27 18:28:30
User3	857.00	0.0242	0.097	-	100.00	2025-05-27 18:29:01

From table 11.3 we can observe the correlation between the purchase power change (through responsible lending) and the predicted probabilities. Another comment we can make is that in all differents credits logistic regression predicts lower probabilities than the MLP , although model retrain will compensate this gap .

In this stage, we will check the predicted probabilities of the model between the same loans from different users From this table we can observe that even though User1 has biggest

Table 11.4: Installment Probability Predictions for different users

User	Purchase Power	LR PD	MLP PD	Outcome	Amount (€)	Timestamp
User1	557.98	0.0738	0.21	-	100.00	2025-05-27 18:35:21
User2	287.61	0.0691	0.18	-	100.00	2025-05-27 18:38:30
User3	557.34	0.0304	0.1023	-	100.00	2025-05-27 18:40:01

credit score (purchase power) than User2, user's 2 age (32) play crucial role in the predicted probabilities. Our model's showed 45% feature importance for the age feature in the class prediction.

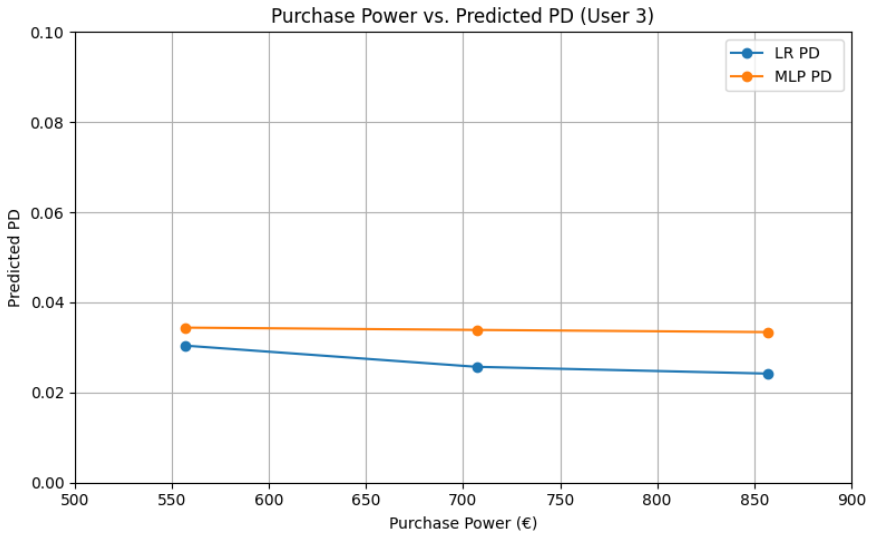

Figure 11.2: PD to Purchase power

After Retraining We tried to trigger the model retraining process to see the differences in the predicted probabilities.

For this approach, we will observe the predicted probabilities of the same amount of loan (100) after 4 consecutive on-time payments from user 1 and 6 on-time payments from different users. For this case, we kept his credit score as it was first initialized.

Table 11.5: Installment Probability Predictions for different users

User	Purchase Power	LR PD	MLP PD	Outcome	Amount (€)	Timestamp
User1	557.98	0.0738	0.0364	-	100.00	2025-05-27 18:52:21

We can easily observe that the MLP prediction dramatically changed after the retraining project for User 1, although the predicted probabilities for User 2 and User 3 change approximately from 18% to 17% and User 3 from 10% to 9.3%. This means that the model becomes biased to specific users whenever they have high interaction with the application. Although this probably would not oppose bigger problems as

- Even if the MLP pd becomes 0, we can also rely on the logistic regression prediction for decision-making.
- Model becomes biased, but it can considered an efficient prediction as the user paid on-time all of his obligations.

As our initial dataset had 220000 instances of loans , retrain process will oppose instability

issues to the model in the early development stages , although if the version of the models exceed a certain point we will observe an equilibrium where no extreme probabilities are predicted. In the scope of this thesis reaching this stabilized stage is not implementable so we are just assuming that after a lot of client involvement, the model will be able to predict sufficient probabilities.

11.2 Static Acceptance System

As regards the acceptance of credit requests we introduced a static procedure where the remaining amount should be bigger than the requested for the credit to be accepted.

11.3 Purchase Power Change

In this section, we will check how the purchase-power is been calculated through consecutive payments, some of them paid in time and the rest with a latency. This payments

Table 11.6: Purchase Power Evolution with Historical Tracking

Inst.	Amount	Days Late	Latency Rate	Duration Factor	Purchase Power
1	120	0	0.00	0.00	620.00
2	80	0	0.00	0.00	700.00
3	150	0	0.00	0.00	850.00
4	100	5	0.25	0.17	790.00
5	90	3	0.40	0.22	748.00
6	110	8	0.50	0.38	657.00
7	75	0	0.57	0.33	644.75
8	130	10	0.63	0.58	492.75
9	60	4	0.67	0.53	442.20
10	95	0	0.60	0.48	537.20
11	105	1	0.55	0.41	631.70
12	85	0	0.50	0.34	716.70
13	115	2	0.54	0.32	769.55
14	70	0	0.50	0.28	839.55

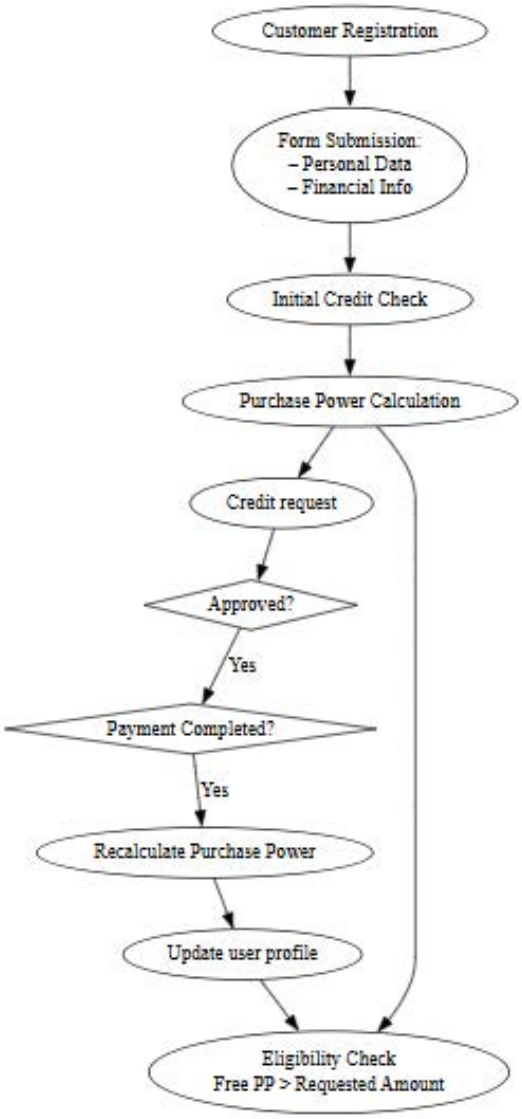

Figure 11.3: Acceptance Flowchart

formulate the purchase power graph in the figure 11.4.

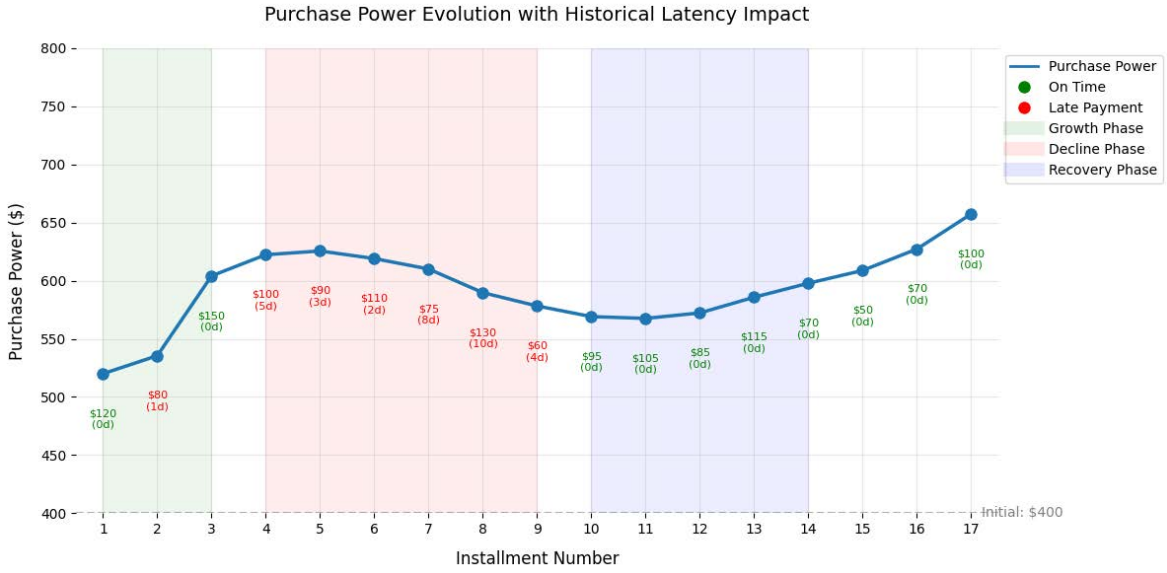

Figure 11.4: PP graph

From this figure is shown how latency in days , and the number of late-payments affect the user's PP.

Chapter 12

Conclusions and Future Work

This thesis presented the design and development of a dynamic, end-to-end creditworthiness and risk management system tailored for Buy Now, Pay Later (BNPL) platforms, utilizing machine learning techniques and financial modeling. The work spans from the theoretical foundation of credit risk to the practical deployment of a fully functional web-based application.

Initially, a thorough theoretical analysis was conducted on banking system risks, with a focus on credit, market, and operational risk in the context of BNPL. Following this, a soft credit check system was developed using classification models (Logistic Regression and Multilayer Perceptron), based solely on demographic and financial data. This initial evaluation was further enhanced using macroeconomic indicators to adjust user purchasing power.

The system then incorporated a dynamic probability of default (PD) estimation engine, along with the calculation of key risk metrics such as Expected Credit Loss (ECL) and Risk-Adjusted Return (RAR). These models were integrated into a real-time decision-making pipeline that accepts or rejects credit requests based on data-driven predictions. In addition, the platform supports behavioral analysis, automatic credit limit adjustment, and model retraining. All of these components were integrated into a web application built using Flask, complete with a backend database and API interfaces. This work establishes a robust foundation for applying advanced modeling techniques to BNPL risk management, although several areas of potential improvements and extensions have been identified:

- **Broader Dataset Integration**: The current model is trained on traditional banking datasets. Integrating more granular BNPL-specific transaction data (e.g., repayment behavior, merchant category) would improve domain alignment and model accuracy.

- **Explainable AI (XAI)**: Implementing SHAP or LIME could offer more transparency and explainability, especially in the MLP model.
- **Model Monitoring & Drift Detection**: As financial behavior and economic conditions evolve, the model's performance may degrade. Incorporating visualization dashboards and data drift detection systems would help maintain model performance over time.
- **User Behavior Profiling**: Future versions can incorporate behavioral analytics (e.g., session time, page navigation) to evaluate creditworthiness with digital behavior insights.Although creditworthiness can also be accessed via social media segmentation.
- **Automated Threshold Optimization**: The current acceptance system through purchase power is static and the models that have been developed for the PD prediction do not have an active role in decision-making. Future implementation, where the PD models are also a part of the credit request management process, will be a really powerful tool.
- **Risk Pooling and Portfolio Analysis**: Integrating more standards of the modern accounting framework, would make this project a real application (IFRS 9).
- **Mobile App or API-Driven Frontend**: The current Flask frontend could be complemented with a React-based dashboard or a mobile-friendly interface for end-users and admin review.
- **Fraud Detection Module**: A parallel model could be developed to detect suspicious behavior or fraud patterns during registration or payment activity, improving the system's robustness. Rapid change in spending amount or rapid change in purchase frequency would be really good fraud-detecting indicators.

These enhancements would transform the current proof-of-concept into a production-grade credit risk engine tailored for digital BNPL platforms.

Bibliography

[1] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002.

[2] Edward I. Altman. Financial ratios, discriminant analysis and the prediction of corporate bankruptcy. *The Journal of Finance*, 23(4):589–609, 1968.

[3] Allen N. Berger, Haoyang Liu, Lu Qi, and Steven Ongena. Fintech lending and cash flow-based screening. *Journal of Financial Intermediation*, 48:100923, 2021.

[4] Tony Bellotti and Jonathan Crook. Forecasting and explaining aggregate consumer credit demand: A machine learning approach. *International Journal of Forecasting*, 29(4):569–578, 2013.

[5] Wendy M. Edelberg. Risk-based pricing of interest rates for consumer loans. *Finance and Economics Discussion Series*, (1999-28), 1999.

[6] Sumit Agarwal, Souphala Chomsisengphet, Neale Mahoney, and Johannes Stroebel. Do banks pass through credit expansions to consumers who want to borrow? *The Quarterly Journal of Economics*, 133(1):129–190, 2018.

[7] LendingClub. Loan data. `https://www.lendingclub.com/info/download-data.action`, 2019. Accessed: 2025-05-22.

[8] Inc. Upstart Holdings. Upstart announces fourth quarter and full year 2024 results. `https://upstartnetworkinc.gcs-web.com/news-releases/news-release-details/upstart-announces-fourth-quarter-and-full-year-2024-results`, 2025. Accessed: 2025-05-22.

[9] Taeyoung Doh. Changing credit profile of consumers: Aging versus the business cycle. *Federal Reserve Bank of Kansas City Economic Bulletin*, 2017.

[10] D. Chandler. Age and credit risk: An empirical study. `https://www.example.com/chandler2020`, 2020. Unpublished manuscript.

[11] ZestFinance. How zestfinance employs ai to enhance credit scoring. `https://redresscompliance.com/how-zestfinance-employs-ai-to-enhance-credit-scoring/`, 2025. Accessed on May 22, 2025.

[12] World Bank. Gdp per capita, ppp (current international $). `https://data.worldbank.org/indicator/NY.GDP.PCAP.PP.CD`, 2023. Accessed May 2025.

[13] Paul Krugman and Robin Wells. *Macroeconomics*. Worth Publishers, 5th edition, 2018.

[14] U.S. Bureau of Labor Statistics. Consumer price index (cpi). `https://www.bls.gov/cpi/`, 2024. Accessed May 2025.

[15] Milton Friedman. *The Quantity Theory of Money: A Restatement*. University of Chicago Press, 1956.

[16] N. Gregory Mankiw. *Principles of Economics*. Cengage Learning, 4th edition, 2006.

[17] Arthur Okun. Potential gnp: Its measurement and significance. *American Statistical Association, Proceedings of the Business and Economic Statistics Section*, pages 98–103, 1962.

[18] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural networks*, 1989.

[19] George Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, 1989.