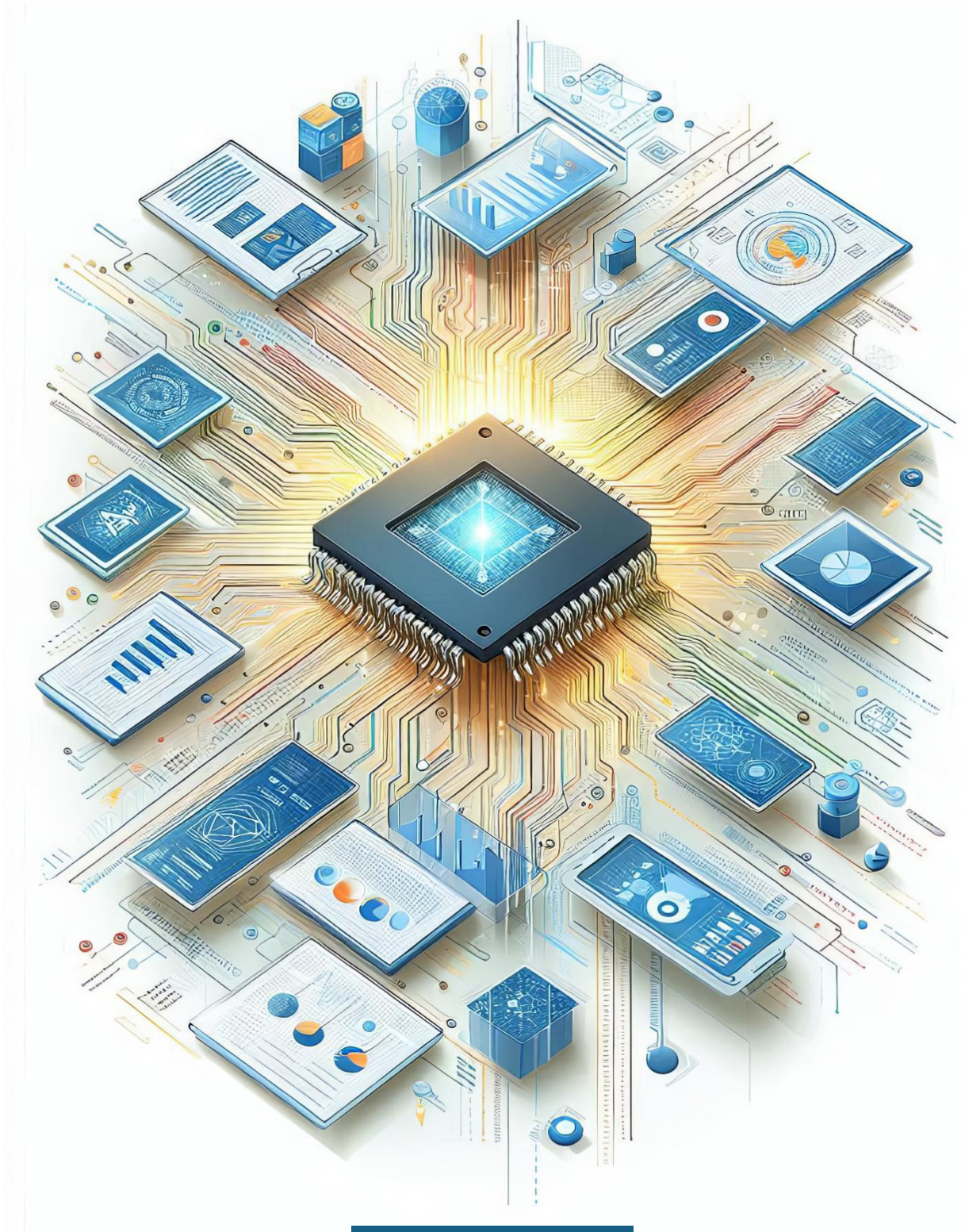
National Bank of Moldova Working Paper

Predicting Bank Defaults with AI

Improvements over Statistical and Machine Learning Methods



Alexandru Monahov

Predicting Bank Defaults with AI:

Improvements over Statistical and Machine Learning Methods

Alexandru Monahov
National Bank of Moldova

June 2025

Abstract

Accurately forecasting bankruptcies within the financial sector is an essential objective for prudential regulators tasked with maintaining financial stability. While Machine Learning techniques, in particular the more advanced ensemble methods, and neural networks have been proven to perform well in forecasting loan defaults, these methods have yet to be integrated into workflows for assessing risk and predicting the failure of financial institutions.

To identify the most effective approach to predicting the default of banks and NBFIs, this study investigates the performance of eight leading predictive modeling techniques of varying complexity – from traditional statistical models (Logistic and Linear Regression), to advanced Machine Learning methods in the field of classification (such as Random Forests, XGBoost and Support Vector Machines), against Large Language Models (LLMs), a rapidly growing area of Artificial Intelligence which has recently made notable improvements in its ability to process large quantities of unstructured textual and numeric data.

The paper develops a new workflow which uses LLMs to analyze the risk exposure of financial institutions and determine their probability of default. A new PD metric, that LLMs are capable of generating accurately, is created as the joint outcome of risk and profitability, whose impacts are separately estimated by the model. To further improve the analysis, the paper proposes a new financial performance indicator and adaptations for traditional ratios to enable their usage in both going concern and failure contexts.

The results of the study reveal that while traditional methods like regression models and Random Forests can provide very good predictive capabilities, the best performance is achieved with the Large Language Model, which significantly surpasses all other methods in the majority of evaluation metrics. The LLM's ability to capture complex patterns and contextual nuances within financial data results in superior predictive accuracy and robustness. This highlights the potential of incorporating advanced language-based modeling approaches into financial risk management systems, paving the way for more intelligent and adaptive frameworks that enhance decision-making and regulatory policy in the financial industry.

Keywords: default, bank, risk, financial institutions, AI, Large Language Models, regression, classification, Machine Learning, Random Forest, XGBoost, Neural Network.

JEL Classification: G21; G23; G33; C38; C45.

1. Introduction

Financial regulation and supervision are focused on monitoring and analyzing a wide spectrum of risks that banking institutions and NBFIs are confronted with. The Global Financial Crisis of 2008 revealed that financial institutions took excessive risks in pursuit of returns.[1] The crisis resulted in the default of major banking institutions and financial turmoil, which spread far beyond the United States, where it first emerged. In pursuing their mission of protecting the stability of the financial system, Central Banks focus on two interrelated, but distinct notions – risk and default, both of which are important in the prevention and management of crises. Whereas "risk" is an accumulating pressure that induces vulnerability, the "default" of an institution represents one of two final outcomes of "risk" when it materializes.

Key risk types that banks are confronted with during crisis events include:

- solvency risk (insufficient capital to absorb losses)
- liquidity risk (lack of funds, often as a result of runs on the bank)
- credit risk (counterparty failure to meet obligations)
- market risk (losses from trading portfolio movements)
- operational risk (failed internal processes, people, or systems).
- interest rate risk (resulting from variations in interest rates and potentially leading to a bank's inability to repay)
- country risk (political or economic instability affecting obligations).

These risks accumulate and, at a certain point may lead to "default", which in finance, signifies a failure to meet debt obligations.[2] This can be a debt service default (in the case of a missed payment) or a technical default (as a result of covenant violations). Default may occur due to one of two major causes: insolvency (inability to repay debts) and illiquidity (insufficient cash).[3]

Modeling bank risk and default is essential for quantifying uncertainties and making informed decisions across financial activities such as lending, underwriting, capital raising, portfolio management and stress testing. However, reliance on models to drive key policy choices introduces "model risk",[4] where inaccuracies can lead to incorrect actions resulting in significant financial losses. It is for this reason that the literature strives to improve and enhance the modeling toolkit by integrating novel approaches, capable of reducing error and improving model accuracy.

Traditional models can predict the default of financial institutions based on performance and risk metrics. The result of these models is a probability of default, which can be used to infer information about the health of an entity. Taken together with the underlying factors that cause the specific default or non-default situation, these models give an idea about the risk level that an institution is subject to. The worse the realization of both the dependent and independent variables, the higher the risk.

However, the limited data included in traditional regression and classification models, may be insufficient to explain the dynamics observed in the variables. This makes it difficult to understand the root causes of a financial entity experiencing heightened or, conversely, lower risk of default. More broadly, the context of the observed movements in the data matters, and this is where LLMs can provide a comprehensive and

structured view of the overall risk environment, through their ability to process large amounts of data, describe their thinking process and provide arguments for the choices that they make.

This paper proposes a new method, capable of responding to both of the fundamental needs that regulators have in the process of ensuring financial stability: monitoring risk, its origins and manifestations, as well as quantifying the probability of default of financial institutions. A default classification mechanism using an LLM is developed to produce an overview of the health of a financial institution, a probability of default score and an outline of the key risks affecting the entity. The method addresses the lack of context of traditional classification models by providing a clear understanding of the key drivers of risk, a comprehensive and detailed assessment of the overall risk environment and a combined scoring approach that dissociates between risk and profitability to give a more precise assessment of an institution's financial situation and resulting probability of failure. The approach allows for a better identification of early signs of financial distress and proves to be an excellent predictor of both bank and non-bank financial institution defaults.

2. Literature review

The quantitative prediction of financial distress has its roots in early statistical methodologies, which laid the groundwork for subsequent advancements in bank default classification such as Machine Learning techniques.

2.1. Statistical models for bank default prediction

Modern academic inquiry into the failure of financial institutions commenced with the pioneering work of Beaver, who in 1968 predicted the bankruptcy of non-financial firms by means of financial ratios.[5] This marked the first instance of default classification using accounting data. Later on, statistical methods were applied to banks by Meyer and Pifer (1970).[6] Since then, researchers have employed multivariate techniques to both explain past bank closures and forecast future failures.[7]

Discriminant Analysis functions as an established statistical procedure designed to classify banks into distinct groups, typically "defaulted" and "non-defaulted" categories. This classification is primarily based on the financial characteristics of the institutions, with the overarching objective of identifying financial weakness at the nascent stages of deterioration. Early works established the paradigm of utilizing financial ratios as predictive variables, a practice that remains integral to even the most sophisticated contemporary models. However, these initial statistical approaches also illuminated the inherent limitations of relying solely on restrictive statistical assumptions, thereby paving the way for the development of more flexible and robust methodologies.

Growing interest in bank failure prediction during the early 1970s coincided with the emergence and refinement of new classification techniques, notably logit and probit regression. These models gained favor over traditional discriminant analysis due to their less restrictive statistical assumptions, as they do not necessitate the normal distribution of independent variables or the homogeneity of variance-covariance matrices, and they offer improved empirical discrimination.

The Logit Model was first applied to the construction of early warnings of bank failure by Martin (1977).[8] In the business failure prediction strand of the literature, Ohlson (1980) was widely recognized as a pioneer for

his application of logit analysis within a multivariate conditional probability model which managed to identify approximately 88% of bankrupt companies about one year prior to their bankruptcy by examining ratios related to liquidity, profitability, leverage, and solvency.[9] Logistic regression operates by predicting the probability of a binary event (e.g., default or non-default, represented as 0 or 1) through a logit transformation, which allows the calculated probability to be directly interpreted as the logarithm of chances.[10]

The concept of Probit Analysis[11] was initially introduced by Bliss (1934)[12] and subsequently detailed by Finney (1947).[13] Zmijewski (1984) was the first to employ a probit model specifically for financial distress prediction, analyzing independent variables such as the ratio of net income to total assets, total liabilities to total assets, and current assets to current liabilities.[14] A key distinction between probit and logit models lies in their underlying assumptions: probit models assume a normal distribution for the independent variables within the model. Comparative studies on the out-of-sample performance of probit and logit models indicate that their predictive capabilities are not significantly divergent, with both demonstrating adequate performance in default prediction.[15]

The transition from Discriminant Analysis to Logit and Probit models represented a significant methodological advancement in quantitative finance. While DA provided a simple classification, Logit and Probit models offered a direct probability of default, which is inherently more valuable for nuanced risk management and capital allocation decisions. Ohlson's critique of DA's statistical assumptions underscored the academic drive for models that are not only predictive but also statistically sound, leading to more robust and interpretable outcomes. The continued effectiveness of these models, even with the emergence of more complex techniques, highlights their enduring foundational value. This evolution reflects a growing sophistication in quantitative finance, moving beyond simple categorization to precise probability estimation, which directly supports more accurate risk pricing and regulatory capital requirements.

Table 1: Evolution of Statistical Models for Bank Default Prediction

Model type	Seminal papers	Core mechanism, assumptions	Key variables	Advantages	Disadvantages
Discriminant Analysis	Beaver (1968), Meyer and Pifer (1970) (banks)	Classifies into groups; assumes multivariate normality and equal variance-covariance matrices	Financial ratios (CAMEL factors), Net income/total assets, Total liabilities/total assets	Early quantitative approach for classification	Restrictive assumptions; provides ordinal ranking, not probability of default
Logit Regression	Martin (1977), Ohlson (1980)	Predicts probability (0-1); uses logistic	Liquidity, profitability, leverage,	Provides probability of default; less	Can be sensitive to estimation

		function; requires less restrictive assumptions (no normality/homogeneity required)	solvency ratios; Net income/total assets, Gross charge-offs/net operating income, Company size	restrictive assumptions than Discriminant Analysis; better empirical discrimination	sample size and crisis definition[16]
Probit Regression	Bliss (1934), Finney (1947), Zmijewski (1984)	Predicts probability (0-1); uses cumulative normal distribution; assumes normal distribution of independent variables	Net income/total assets, Total liabilities/total assets, Current assets/current liabilities	Provides probability of default; statistically rigorous under its assumptions	Assumes normal distribution of independent variables; can be sensitive to estimation sample size and crisis definition[16]

This table summarizes the key contributions to statistical methods for default classification.

2.2. The Machine Learning revolution in default prediction

The advent of machine learning (ML) profoundly transformed the landscape of default prediction, offering models with enhanced accuracy and efficiency. The application of ML models became particularly prominent around 2010, driven by their superior out-of-sample performance.[17]

2.2.1. Core Machine Learning techniques

While Logistic Regression is a statistical model, its widespread use as a baseline in classification problems often places it within discussions of ML techniques. It predicts the probability of default based on various factors, offering simplicity and interpretability, though it may not capture highly complex relationships.

Decision Trees (DT) are popular for both classification and regression tasks. They are valued for their transparency and ease of interpretation, which facilitates understanding the factors influencing failure to repay in the analysis of events such as loan defaults. However, they are susceptible to overfitting, especially with complex datasets, and can exhibit instability where minor data changes lead to significant structural alterations.[18]

Support Vector Machines (SVMs) are widely utilized for classification and regression problems, having been widely applied to loan default prediction based on factors like credit score, income, and employment status, as well as firm defaults[19]. SVMs offer considerable flexibility in modeling complex data patterns, but their computational expense can be a significant drawback, and their "black-box" nature often renders them less interpretable than simpler models.

Neural Networks (NN) and Deep Learning (DL) represent a powerful class of ML models, excelling at identifying intricate and non-linear patterns within financial data. They can automatically extract and learn features directly from raw data, eliminating the need for manual feature engineering and contributing to high predictive accuracy. Deep learning models have been tested against traditional credit scoring models and consistently demonstrated superior performance. A contribution in the field of credit score prediction based on current scores, loan amount, size of the company and sector of activity by Dima and Vasilache (2016), shows an improvement in accuracy of nearly 10% when switching from a logistic regression to a multilayer perceptron-type neural network which achieved an accuracy score of 86.8%.[20] But NNs and DL models require large quantities of labeled data in order to produce significant improvements in performance. Furthermore, despite efforts to enhance the explainability of neural networks, which began in the late 90s[21] and continue to this day,[22] their even higher degree of "black-box" nature poses significant challenges for interpretability, which can be a concern for financial institutions or regulators needing to explain their decisions.

2.2.2. Ensemble methods

To mitigate the overfitting issues inherent in single decision trees and enhance overall predictive accuracy, ensemble methods combine multiple individual models. These methods have proven highly effective in predicting loan failures.

Random Forests (RF) operate by constructing multiple decision trees, each trained on a randomly selected subset of variables and observations. The final prediction is derived by averaging the predictions of all individual trees, a process that effectively reduces variance and overfitting, thereby improving model stability and performance. RF models are recognized for their robustness, ability to handle missing data, and their capacity to provide feature importance scores.

Gradient Boosting (GB) and its optimized variant, XGBoost, sequentially combine multiple "weak" predictive models, typically decision trees. Each successive tree is trained to correct the errors made by its predecessors, leading to a cumulative improvement in accuracy and predictive power. These techniques are highly powerful, capable of handling large datasets, and are less prone to overfitting than single models, effectively capturing complex relationships within the data. Studies, such as those by Brown and Mues (2011),[23] have consistently found that gradient boosting and random forests exhibit superior performance in loan default prediction. In particular, XGBoost has demonstrated high Recall and AUC scores in various applications.[24] In a study by Zhou (2022), the maximum accuracy score across several Machine Learning methods is 83.7%, attained by a Random Forest implementation.

While many of the more advanced Machine Learning methods have been extensively used in loan default prediction, so far, they have not yet been widely adopted in the analysis of bank risk and the prediction of failed financial institutions.

Table 2: Overview of Machine Learning Models in Bank Default Prediction

Model Type	Seminal papers	Core mechanism	Key advantages	Key disadvantages
Decision Trees (DT)	Brown and Mues (2012), Halabaku and Bytyçi (2024), Zhou (2022)	Tree-like model of decisions and consequences	Easy to interpret; handles categorical / numerical data; captures non-linear effects	Overfitting, instability; prone to underfitting
Random Forests (RF)	Brown and Mues (2012), Halabaku and Bytyçi (2024), Zhou (2022)	Ensemble of multiple DTs, averaging predictions	More accurate/robust than single DTs; handles missing values; less prone to overfitting; provides feature importance	Less interpretable than single DTs; more computational resources
Gradient Boosting (GB) / XGBoost	Brown and Mues (2012), Zhou (2022)	Sequential ensemble, correcting errors of previous models	High accuracy; handles large datasets; less prone to overfitting; captures complex relationships	Requires careful tuning to prevent overfitting
Support Vector Machines (SVM)	Brown and Mues (2012), Chen et al. (2011)	Finds optimal hyperplane for classification	Flexible, powerful for complex datasets; robust to outliers	Computationally expensive; less interpretable than simpler models
Neural Networks (NN) / Deep Learning (DL)	Brown and Mues (2012), Dima and Vasilache (2016)	Interconnected layers of nodes learning complex patterns	High accuracy; automatic feature extraction; captures complex non-linear patterns	High computational resources; requires large labeled datasets; "black-box" interpretability challenge; overfitting; local optima

This table summarizes the key contributions in the field of Machine Learning and Neural Networks for default classification.

2.3. Artificial Intelligence: the dawn of a new era in data analysis

Large Language Models (LLMs) signify a profound and transformative advancement within Artificial Intelligence, fundamentally reshaping the landscape of natural language processing (NLP) and extending their capabilities across a wide array of complex problem-solving domains. LLMs are based on the core principle of acquiring extensive knowledge about the world and languages through training on existing textual data. This approach facilitates the creation of highly versatile, "universal models" capable of addressing diverse challenges. The field has notably transitioned from a traditional paradigm of training specialized systems from scratch using vast amounts of labeled data to a new methodology centered on large-scale pre-training of foundational models, which are subsequently refined through fine-tuning, alignment, and sophisticated prompting techniques.

This shift from training specialized systems for narrow applications to developing generalized foundation models represents a significant evolution in AI development strategy. Historically, AI models were often bespoke solutions, each requiring labor-intensive, labeled datasets for specific tasks. The advent of LLMs, however, demonstrates that a single, extensively pre-trained model can serve as a robust base for numerous downstream applications. This implies a substantial reduction in the cost and time associated with developing AI for new applications, as the intensive process of general knowledge acquisition is performed once during pre-training. This progression also points towards the development of more adaptable and versatile AI systems, capable of broader utility beyond predefined tasks.

Seminal papers in the field include work by Vaswani et al. (2017)[25], Devlin et al. (2018)[26], Brown et al. (2020)[27], Raffel et al. (2020)[28] and Ouyang et al. (2022).[29]

Table 3: Seminal Papers in LLM Development

Paper Title	Key Authors	Year	Primary Contribution	Significance
Attention Is All You Need	Vaswani et al.	2017	Introduced the Transformer architecture with self-attention, eliminating recurrence.	Enabled parallel processing, dramatically accelerated training, and facilitated efficient scaling, becoming the backbone of modern LLMs and catalyzing the "AI boom".
BERT: Pre-training of Deep Bidirectional Transformers for	Devlin et al.	2018	Pioneered deep bidirectional representations using masked	Achieved state-of-the-art results across 11 NLP tasks with minimal

Language Understanding			language modeling and next sentence prediction.	fine-tuning, validating the power of general, contextualized pre-trained representations.
Language Models are Few-Shot Learners	Brown et al.	2020	Demonstrated dramatic improvements in task-agnostic, few-shot performance through extreme scaling (175 billion parameters).	Showcased emergent in-context learning and meta-learning abilities, reducing reliance on large task-specific datasets and highlighting the power of scale.
Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer	Raffel et al.	2020	Introduced a unified text-to-text framework, reframing all NLP tasks as text generation.	Simplified the NLP landscape by demonstrating the versatility of a single model and training objective across diverse language problems, achieving state-of-the-art results.
Training Language Models to Follow Instructions with Human Feedback	Ouyang et al.	2022	Introduced Reinforcement Learning from Human Feedback (RLHF) for aligning LLMs with human preferences.	Enabled smaller models (1.3B) to outperform larger ones (GPT-3 175B) in instruction following, leading to models like ChatGPT that are more helpful, truthful, and safe.

This table summarizes key contributions in the field of Large Language Modeling and presents the paper title, authors, year of publication, as well as the importance of the work in the development of current-generation LLMs.

3. Data and model specification

3.1. Global coverage of the dataset including developed and developing economies

The objective of bank default prediction models is to assess the likelihood of banks failing. Traditional models often focus on developed economies, potentially overlooking risk factors preponderantly present in developing markets, but which can also impact advanced markets by means of periodic occurrence or through contagion. For this reason, this paper collects a diversified dataset that includes banks from both economic spheres to create a more comprehensive and reliable prediction model.

Incorporating banks from both developing and developed economies into bank default prediction models has the potential to significantly enhance predictive accuracy by capturing a comprehensive spectrum of risk profiles. More specifically, banks operating within developed economies typically benefit from mature financial systems characterized by stringent regulations and advanced risk management practices. On the other hand, banks in smaller, developing economies might be susceptible to various external disruptions and may lack sufficient support mechanisms when crises occur.

A diverse dataset enables prediction models to generalize across a wide range of bank features and financial conditions, mitigating the risk of overfitting specific market characteristics. Training on data from both developed and developing economies enhances model adaptability, allowing for accurate predictions even in unfamiliar or extreme scenarios.

Furthermore, diversified data improves a model's ability to detect subtle patterns and interactions that may be obscured within homogenous datasets. This enhanced predictive power allows models to better differentiate between transient issues and systemic risks across different banking systems.

Given these considerations, the model includes 64 financial institutions – 59 banks and 5 NBFIs, to test the model's performance across different types of financial entities. Of the 59 banks, 29 are from developed economies and 30 from developing nations. Of the developing nations, 16 are transition economies from Central, Eastern and South-Eastern Europe. A bit less than half of the sample represents defaulted bank, such that 26 entities are failed banks and one failed NBFI was also included.

Major global organizations, trading blocks and groups of nations were accounted for, alongside specimens from each continent of the world. In particular, countries adhering to the following international bodies are covered in their entirety:

- G20
- European Union
- BRICS
- USMCA (former NAFTA countries)

On the European continent, banks operating in all countries were included in the sample, with the exception of very small nations with less than 100,000 inhabitants as of 2024.[30]

Table 4. Overview of financial institutions included in the sample

Country statistics	
Developed	**29**
Active	16
Defaulted	13
Developing	**30**
Active	17
Defaulted	13
Transition	16
Emerging	9
BRICS	5
Banks	**59**
NBFIs	**5**

This table shows the wide range of countries from both developed and developing economies included in the sample.

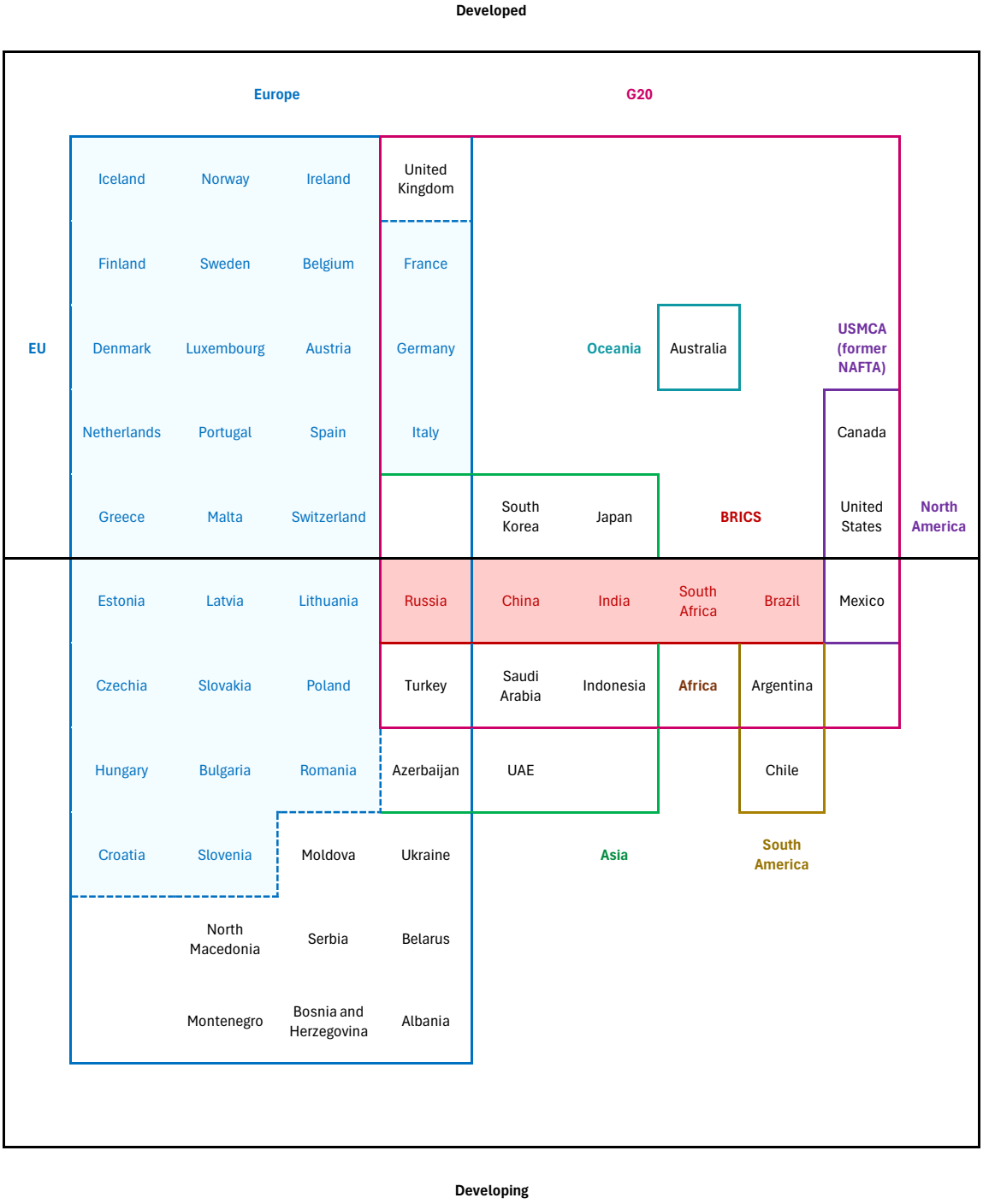
Given the global nature of the study, it is indeed easier to visualize the nations where the banks included in this sample operate through a chart. The visualization groups all of the nations by:

- Development level
- Geographic location
- Economic blocks

As such, countries are grouped in two major categories of development level that transcend all other categorizations – developed and developing economies. After being identified as "developed" or "developing", countries are positioned by geographic location and, finally, regrouped into major blocks such as the EU, BRICS, G20 or USMCA (former NAFTA). Individual countries can be part of multiple groups at the same time. Countries from each continent have been included in the study, however, due to data availability reasons, the level of representation differs. In the case of Europe, countries are depicted as members and non-members of the EU.

The visual representation highlights that this analysis primarily focuses on Europe, where the vast majority of countries are included in the sample. At the same time, it is important to note that the US is represented by a relatively larger set of banks, when compared to other countries. The greater presence of U.S. banks can be attributed to their significant role in precipitating global financial crises, such as during the 2008 Global Financial Crisis, as well as the comprehensive and detailed data available in the public domain on both defaulted and active entities.

Figure 1. Sample diversity by development level, international block and geography



The figure depicts countries included in the analysis and groups them according to various criteria: development level, international block and continent.

3.2. Sources, extraction and transformation of textual and numeric data

Predicting bank defaults involves collecting, synthesizing and processing various financial risk and performance metrics to form a comprehensive view of an institution's health.

The data used in this paper consists of quarterly or annual financial reports which represent textual bodies of data containing both description and numbers. The length of the reports varies from 3 pages to 1540 pages, as does the complexity of the analysis presented in the reports. While some limit themselves to presenting the balance sheet and income statement, others discuss in minute detail various aspects of the bank's operating environment, strategic choices, key risks, financial ratios, stress testing, predictive analytics and plans for future actions.

The date of the reports provided to the LLM differed as a function of the bank's status.

For going concerns – the latest available quarterly or annual report, on the date of collection.

The defaulted institutions – the latest available quarterly or annual report, typically before the bank defaulted.

The date of the financial institution's default is generally taken to be the day on which the Central Bank revoked the entity's license to operate. Some exceptions from this rule exist, for instance, in the case of institutions that decided to self-liquidate. In this instance the date of default is taken to be the day on which the decision to self-liquidate was taken. Similar exceptions exist for banks deemed unviable by supranational EU bodies. In such cases, the date of default is taken to be the day on which the regulatory entity announced the financial institution as unviable.

For some financial institutions, quarterly reporting was spread across multiple files. In such instances, the ensemble of files was provided to the LLM for processing.

Key financial performance indicators were extracted from the reports either manually or with assistance from AI, depending on the complexity of the documents. If AI assistance was used, a manual verification of each extracted value was performed to verify the outcome. The set of financial performance indicators reported by institutions differed on a case-by-case basis. Some missing indicators could be calculated based on balance sheet items, income statements or additional information that was reported. For the purpose of this analysis, only those variables were kept which were available for all of the entities in the sample, either as directly reported values or which could be calculated based on the reported data.

The retained indicators are used in the estimation of two model specifications – one containing only financial ratios (4 variables) and another including information about loan growth and profit variation (6 variables).

- Loan growth
- Profit variation
- Loan-to-Deposit ratio (LDR)
- Cost-to-Income ratio (CIR)
- Capital Adequacy Ratio (CAR)
- Non-Performing Loan (NPL) ratio

Financial Ratios Model
4-variable specification

Extended Model
6-variable specification

Both the loan growth and the profit variation are expressed as year-on-year growth rates.

3.3. Model specification

The extracted financial indicators were used as regressors in the statistical, Machine Learning and Neural Network models, while the entire unstructured report text was provided to the LLM for analysis and assessment.

In the non-LLM modeling workflow, two models were estimated for each of the methodologies adopted by this paper, and discussed in the next section:

1. Financial ratios model (4-variable specification)

This model tests if financial ratios, which are widely considered to be efficient in evaluating a bank's financial health are sufficient to produce accurate estimates of a bank's default.

$$PD = \beta_0 + \beta_1 ldr + \beta_2 cir + \beta_3 car + \beta_4 npl + \varepsilon \quad (1)$$

where:

ldr – the Loan-to-Deposit Ratio

cir – the Cost-to-Income Ratio

car – the Capital Adequacy Ratio

npl – the Non-Performing Loans ratio

ε – the stochastic error terms

β_0 – the intercept

$\beta_1 - \beta_4$ – the slope coefficients

2. Extended model (6-variable specification)

Given the limited set of data available, it is of interest to try and make use of all the available data to make predictions. Since, in addition to the widely reported financial ratios, all of the banks in the sample also reported information about loan growth and profit, these two additional indicators were added to the analysis.

The complete version includes all the variables in the dataset aiming to make maximal use of sparse data.

$$PD = \beta_0 + \beta_1 ldr + \beta_2 cir + \beta_3 car + \beta_4 npl + \beta_5 loansd + \beta_6 droa + \varepsilon \quad (2)$$

ldr – the Loan-to-Deposit Ratio

cir – the Cost-to-Income Ratio

car – the Capital Adequacy Ratio

npl – the Non-Performing Loans ratio

$loansd$ – loan growth

$droa$ – Dynamic Return On Assets

ε – the stochastic error terms

β_0 – the intercept

$\beta_1 - \beta_6$ – the slope coefficients

Because financial ratios are widely used by policy-makers and the markets as decision guides, it is interesting to see if they alone are able to provide sufficient accuracy for the classification of financial institutions. This specification includes the 4 available ratios (LDR, CIR, CAR and NPL).

Due to the sample including both bank and non-bank entities, as well as going concerns and defaulted institutions, from a broad range of economies with different levels of financial development and complexity, some of the financial performance and risk metrics had to be rethought or had to have constrains imposed on them, as the properties of the traditional indicators proved unsuitable for the analysis of all categories of institutions.

Profit growth is an important indicator which provides information about the dynamics of an entity's financial situation. To render the indicator comparable across institutions with vastly different levels of returns, it is customary for the indicator to be calculated in terms of percentage changes. However, when confronted with going concerns and defaulted institutions, with varying positive or negative profit levels, the financial indicator stops producing informative results.

To counteract this, a new indicator was created to measure profit growth, drawing inspiration from the ROA metric, which remains usable in both going concern and default scenarios, due to assets typically remaining positive. The newly introduced Dynamic Return on Assets (DROA) indicator measures the variation in profit relative to total assets.

$$DROA = \frac{Prof_t - Prof_{t-1}}{Assets_t} \tag{3}$$

Where:

$Prof_t$ – is the entity's profit in the current period

$Prof_{t-1}$ – is the profit in the previous period

$Assets_t$ – represents current period total assets

The metric combines the logic behind two important accounting metrics – profit growth and return on assets – into a single combined indicator, which can be interpreted as a dynamic ROA. The advantage of the indicator is that even when profits become increasingly negative or switch from positive to negative (and vice-versa), the indicator retains the proper direction of the dynamics due to assets remaining positive. This overcomes the impossibility of using profit growth to compare the performance evolution of going concerns and defaulted institutions.

Furthermore, to correct for outlier values without excluding observations altogether:

- Negative DROA values were capped at -2%.
- The LDR was capped at 200%.
- The CIR had to be both upper and lower bound, with an additional adjustment by which both values under the lower bound of 0, as well as surpassing the 100% upper bound were constrained to unitary value.
- Negative CARs were limited to -25%.

In the case of the 6-variable model, one observation had to be dropped because of presenting multiple outliers.

4. Methodology

Logistic regression, random forests, and XGBoost are among the most widely used methods for solving classification problems. Additionally, linear regression adapted to classification problems can provide an additional validation score aiding in the comparison of model performance.

Logistic regression is a straightforward and interpretable statistical method used in the estimation of probabilities of default, allowing for an easy understanding of how different variables influence the likelihood of a bank failing. Its transparency makes logistic regression a valuable tool for applications that require trackability of the risk factors affecting the model's predictions.

Random forests is a Machine Learning method that offers robustness against overfitting through its ensemble approach which aggregates predictions from numerous decision trees. This method effectively handles large datasets with complex interactions between features, providing high predictive accuracy without necessitating extensive feature engineering.

XGBoost, an optimized gradient boosting algorithm, further enhances prediction accuracy by focusing on the most difficult to classify instances through iterative learning. It is highly efficient and scalable, making it suitable for large datasets. XGBoost's regularization techniques help prevent overfitting while still capturing intricate patterns within the data.

In what follows, this section presents a more detailed description of each method.

4.1. Linear Models for Classification

Linear models are foundational in econometric analysis and serve as a starting point for understanding more complex algorithms. They establish a linear relationship between input features and an outcome, providing an easily interpretable view of the key drivers determining the dynamics of an estimated model.

4.1.1. Linear regression for classification

Linear regression is a straightforward method that models the relationship between a set of independent variables (features) X and a continuous dependent variable y. Although it's intended for regression, it can be repurposed for binary classification by establishing a threshold.

In this approach, the binary outcome variable y (coded as 0 and 1) is treated as if it were a continuous variable. The model fits a hyperplane to the data that best represents the linear relationship between the features and this numeric outcome.

The core of linear regression is the hypothesis function, $y_\theta(x)$, which predicts the outcome based on a linear combination of the input features x. For a given observation x_i with n features, the hypothesis is given by:

$$y_\theta(x_i) = \theta_0 + \theta_1 x_{i1} + \theta_2 x_{i2} + \cdots + \theta_n x_{in} \tag{4}$$

This can be expressed more concisely in vectorized form:

$$y_\theta(x) = \theta^T x \tag{5}$$

where:

$y_\theta(x)$ – is the predicted output.

θ – is the vector of model parameters (weights), including the bias term θ_0.

x – is the vector of input features, with x_0 conventionally set to 1.

The parameters θ are learned by minimizing a cost function, which measures the discrepancy between the predicted values and the actual values in the training data. The standard cost function for linear regression is the Mean Squared Error (MSE):

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(y_\theta(x^{(i)}) - y^{(i)})^2 \tag{6}$$

where:

m – is the number of training examples.

$y_\theta(x^{(i)})$ – is the prediction for the i-th training example.

$y^{(i)}$ – is the actual label for the i-th training example.

This function is minimized using an optimization algorithm like Gradient Descent, which iteratively adjusts the parameters θ to reduce the error.

To convert the continuous output of the linear regression model into a class prediction, a threshold is applied. A common choice for a binary (0/1) problem is 0.5:

$$\hat{y} = \begin{cases} 1 & \text{if } y_\theta(x) \geq 0.5 \\ 0 & \text{if } y_\theta(x) < 0.5 \end{cases}$$

Using linear regression for classification has certain limitations in that the model's output is not constrained to the $[0,1]$ interval, making it difficult to interpret as a probability. It can produce predictions like 1.7 or -0.3, which are not interpretable in a probabilistic context. Furthermore, there is the issue of sensitivity to outliers, given that the MSE cost function penalizes all errors quadratically, regardless of whether the prediction is on

the correct side of the decision boundary. This means that a data point far from the boundary can significantly skew the fit of the model and alter the classification threshold, leading to suboptimal decision boundaries. For this reason, usage of this method requires careful handling of outlier values and setting a pertinent classification threshold.

4.1.2. Logistic regression

Logistic Regression is the standard linear model for binary classification, explicitly designed to address the shortcomings of using linear regression for this task. It models the probability that an input x belongs to a particular class.

The key innovation in logistic regression is the use of the sigmoid function, $\sigma(z)$, which "squashes" any real-valued number into the range $(0,1)$.

$$\sigma(z) = \frac{1}{1 + e^{-z}} \tag{7}$$

This functional form allows the model's output to be interpreted as a probability.

The hypothesis function for logistic regression passes the linear combination of features through the sigmoid function:

$$h_\theta(x) = \sigma(\theta^T x) = \frac{1}{1 + e^{-\theta^T x}} \tag{8}$$

Here, $h_\theta(x)$ represents the estimated probability that $y = 1$ given the input x, i.e., $P(y = 1|x; \theta)$. The prediction is then made based on the probability threshold:

$$\hat{y} = \begin{cases} 1 & \text{if } h_\theta(x) \geq 0.5 \\ 0 & \text{if } h_\theta(x) < 0.5 \end{cases}$$

The quadratic cost function of linear regression is not suitable for logistic regression because it would result in a non-convex optimization problem with many local minima. Instead, logistic regression uses a cost function derived from the principle of maximum likelihood estimation, known as the Log Loss or Binary Cross-Entropy:

$$J(\theta) = -\frac{1}{m}\sum_{t=1}^{m} y^{(i)} log(h_\theta(x^{(i)})) + (1 - y^{(i)}) log(1 - h_\theta(x^{(i)})) \tag{9}$$

This cost function has the desirable properties of being convex and penalizing confident but incorrect predictions heavily. For an observation where the true class is $y = 1$, the cost approaches ∞ as the predicted probability $h_\theta(x)$ approaches 0. Conversely, if the true class is $y = 0$, the cost approaches ∞ as the prediction approaches 1. This ensures that the model is strongly incentivized to produce accurate probabilities.

4.2. Machine Learning (ML) methods

Classification problems require distinguishing between different categories based on observed characteristics. Traditionally solved with rule-based systems, these tasks are increasingly addressed through machine learning, a paradigm focused on algorithms that learn to categorize data rather than being explicitly programmed with decision rules. ML models achieve this by identifying patterns within labeled training datasets, where the correct category is already known, and building a predictive function to assign new, unseen data points to one of those categories. This learning process allows systems to adapt and improve their classification accuracy without constant manual adjustments. This is of particular value when designing supervisory or policy-oriented decision mechanisms employing high-dimensional or complex data. The following sub-sections detail some of the most prominent machine learning methods, which are all tested in this study.

4.2.1. Random Forest (RF)

Random Forest is an ensemble learning method that operates by constructing a multitude of decision trees at training time. For a classification task, the final output is the class selected by the most trees (majority vote). It is a form of bagging (Bootstrap Aggregating) with an added layer of randomness for feature selection.

The base learner in a Random Forest is a decision tree. A decision tree recursively partitions the data into subsets based on the values of the input features. The goal at each node is to create a split that makes the resulting child nodes as "pure" as possible – that is, containing observations from a single class. Two common criteria for measuring this purity are Gini Impurity and Entropy.

Gini Impurity measures the frequency at which any element from the set would be incorrectly labeled if it was randomly labeled according to the distribution of labels in the subset. For a given node with K classes, the Gini impurity is:

$$G = 1 - \sum_{k=1}^{K} p_k^2 \tag{10}$$

where:

p_k – is the proportion of samples belonging to class k at the node.

A Gini score of 0 indicates perfect purity.

Entropy measures the level of disorder or uncertainty in a set. For a given node, the entropy is:

$$H = -\sum_{k=1}^{K} p_k \log_2(p_k) \tag{11}$$

where:

p_k – is the proportion of samples for class k.

Entropy is 0 for a perfectly pure node.

The Random Forest algorithm builds a "forest" of B decision trees. Each tree is trained on a different bootstrap sample of the training data. Furthermore, at each split in a tree, only a random subset of features is considered. This process decorrelates the trees and reduces variance.

For a new input observation x', the final prediction of the Random Forest is obtained by aggregating the predictions of all the individual trees. For classification, this is done via a majority vote:

$$\hat{y}_{RF} = \text{majority\_vote}\{\hat{y}_b(x')\}_{b=1}^{B} \tag{12}$$

where:

$\hat{y}_b(x')$ – is the prediction of the b-th individual tree.

4.2.2. XGBoost

XGBoost (eXtreme Gradient Boosting) is an advanced and highly efficient implementation of the gradient boosting framework. Unlike Random Forest where trees are built independently, boosting involves building models sequentially, where each new model corrects the errors made by the previous ones.

The prediction of a gradient boosting model is the sum of the predictions of all the individual trees. If $\hat{y}_i$ is the prediction for observation i after K rounds (trees), it is given by:

$$\hat{y}_i = \sum_{k=1}^{K} f_k\,(x_i) \tag{13}$$

where:

f_k – is the prediction of the k-th tree.

This is an additive model that is built in stages. The prediction at stage t is:

$$\hat{y}_i^{(t)} = \hat{y}_i^{(t-1)} + f_t(x_i) \tag{14}$$

where:

$\hat{y}_i^{(t-1)}$ – is the prediction from the previous $t-1$ trees.

f_t – is the new tree being trained to minimize the remaining error.

The core of XGBoost is its objective function, which it seeks to minimize at each step t. This function includes both a loss term and a regularization term. The regularization term penalizes the complexity of the model, helping to avoid overfitting.

$$\text{Obj}^{(t)} = \sum_{i=1}^{n} l\left(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)\right) + \Omega(f_t) \tag{15}$$

where:

$l(y_i, \hat{y}_i)$ – is a differentiable loss function (i.e. Log Loss for classification).

n – is the number of observations.

$\Omega(f_t)$ – is the regularization term for the new tree f_t.

The regularization term is defined as:

$$\Omega(f_t) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2 \tag{16}$$

where:

T – is the number of leaves in the tree f_t.

w_j – is the score (or weight) of the j-th leaf.

γ and λ – are regularization parameters that control the penalty for the number of leaves and the magnitude of the leaf weights, respectively.

To optimize this objective efficiently, XGBoost uses a second-order Taylor expansion of the loss function. This transforms the objective into a function of the gradients and Hessians of the loss function, which can be optimized to find the optimal structure and leaf weights for the new tree f_t at each iteration. This sophisticated optimization is a key reason for XGBoost's high performance and speed.

4.2.3. Naïve Bayes (NB)

Naive Bayes classifiers represent a family of probabilistic machine learning algorithms used for classification tasks. Rooted in Bayes' Theorem, these models operate under the simplifying assumption of conditional independence between features given the class label – hence the "naive" designation. While this assumption is often unrealistic in real-world datasets, Naïve Bayes classifiers frequently achieve surprisingly high accuracy, particularly with high-dimensional data, and are computationally efficient, making them a valuable baseline model and suitable for large-scale applications.

Bayes' Theorem provides the foundation for calculating the posterior probability of a class *C* given observed features $F = (f_1, f_2, \ldots, f_n)$:

$$P(C|F) = \frac{P(F|C) * P(C)}{P(F)} \tag{17}$$

where:

$P(C \mid F)$ – is the posterior probability of class *C* given features *F*, the target to be predicted.

$P(F \mid C)$ – is the likelihood, representing the probability of observing features *F* given that the instance belongs to class *C*.

$P(C)$ – is the prior probability of class *C*. This reflects our initial belief about the prevalence of each class before considering any evidence.

$P(F)$ – is the evidence, representing the probability of observing features *F*. This acts as a normalizing constant and is often omitted in classification since it's the same for all classes when comparing probabilities.

The "naive" simplification comes into play when estimating $P(F \mid C)$. Instead of modeling the joint probability distribution of all features directly (which can be computationally intractable), Naive Bayes assumes that each feature f_i is conditionally independent of all other features given the class *C*. This allows us to decompose the likelihood as follows:

$$P(F|C) = P(f_1|C) * P(f_2|C) * \dots * P(f_n|C) \tag{18}$$

Given a training dataset, the Naive Bayes classifier learns the prior probabilities $P(C)$ and the conditional probabilities $P(f_i \mid C)$ from the data. During classification, it calculates the posterior probability for each class and assigns the instance to the class with the highest probability.

In this research, Gaussian Naïve Bayes is employed due to its effectiveness in handling the continuous feature representation of the financial ratios, as well as loan and profit growth rates included in the data sample.

4.2.4. k-Nearest Neighbors (k-NN)

The k-Nearest Neighbors algorithm is a non-parametric, instance-based learning method used for both classification and regression tasks. Its simplicity and effectiveness make it a widely applicable technique in economics and finance. Unlike parametric methods that assume a specific underlying data distribution, k-NN relies on storing the entire training dataset and classifying or predicting new instances based on the characteristics of their nearest neighbors.

Given a new, unlabeled instance x, the k-NN algorithm identifies the *k* closest instances from the training set according to a chosen distance metric. For classification, the class label assigned to x is determined by majority voting among its *k* nearest neighbors – the most frequent class amongst those neighbors is predicted as the class for x.

The selection of an appropriate distance metric is important for k-NN performance. This research utilizes the Euclidean distance to quantify the dissimilarity between instances. The Euclidean distance is calculated as:

$$d(x_\mathrm{i}, x_\mathrm{j}) = \sqrt{\Sigma(x_\mathrm{ik} - x_\mathrm{jk})^2} \tag{19}$$

where: x_i and x_j are two data points, while *k* represents the number of features.

Euclidian distance was chosen due to its suitability for continuous variables, interpretability and common use in similar studies.

A key parameter of the k-NN algorithm is the value of *k*, representing the number of neighbors considered. The optimal value of *k* significantly impacts model performance – a small value of *k* can induce sensitivity to

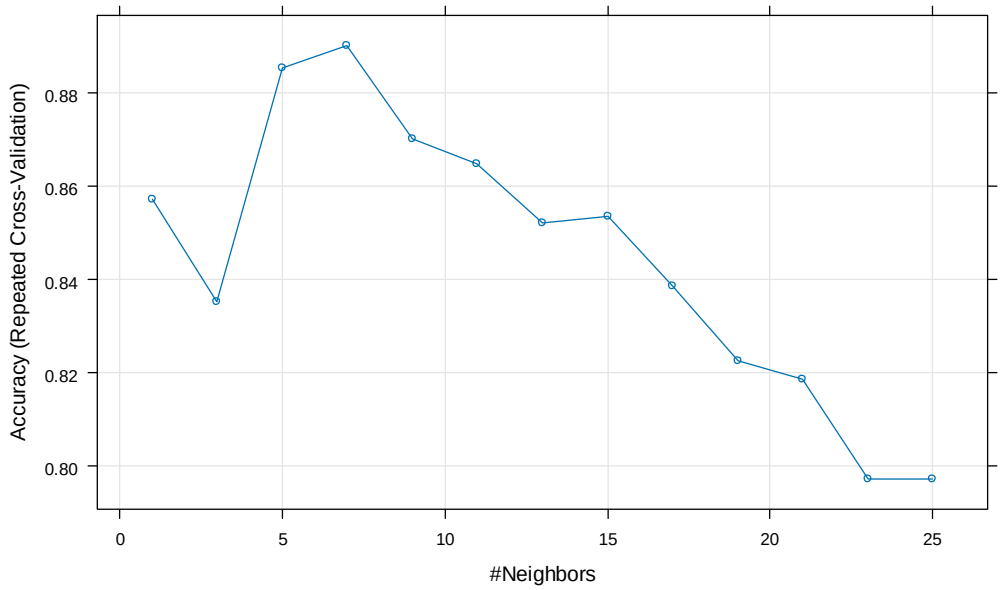
noise and overfitting, while a large value may smooth out decision boundaries and result in underfitting. In this study, *k* was selected using cross-validation with a range for values of k comprised between 1 and 25.

10-fold cross-validation, repeated three times, was performed for both 4 and 6-variable specifications and optimal values of *k* were determined separately for each model:

- k = 7 (for the 4-variable model)
- k = 5 (for the 6-variable model)

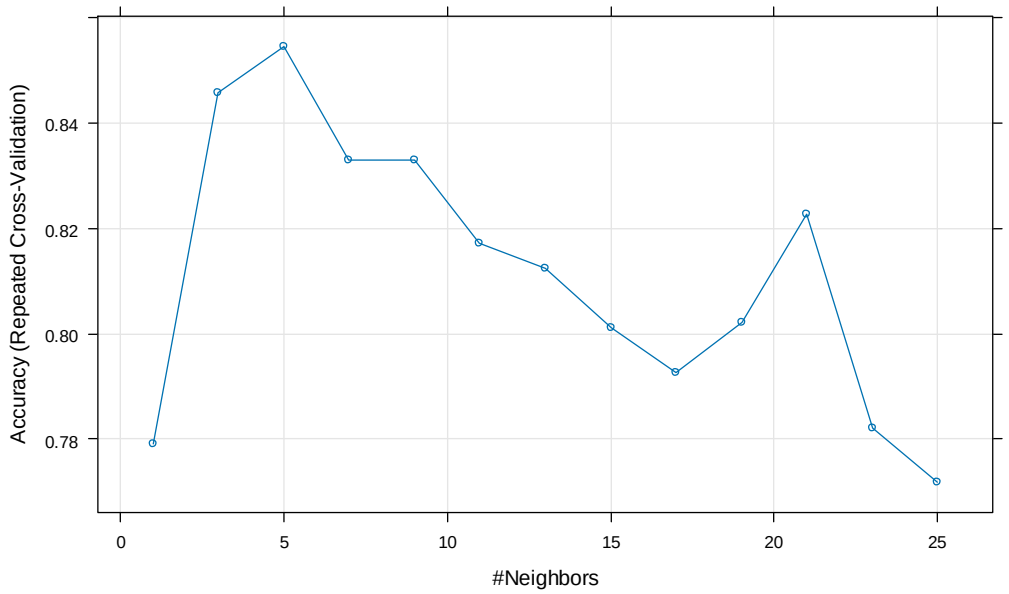
The results of the search can be visualized as charts depicting the level of accuracy attainable for different numbers of neighbors considered by the algorithm. The k-parameter associated with the highest accuracy score is retained for further modeling.

Figure 2. k-parameter search for the 4-variable model



This figure presents the results of the search process for the 4-variable model. The figure plots accuracy on the y-axis and the k-parameter value on the x-axis. The k-parameter value associated with the highest accuracy score is retained for further modeling. In the case of the 4-variable model specification, k=7.

Figure 3. k-parameter search for the 6-variable model



This figure presents the results of the search process for the 6-variable model. The figure plots accuracy on the y-axis and the k-parameter value on the x-axis. The k-parameter value associated with the highest accuracy score is retained for further modeling. In the case of the 6-variable model specification, k=5.

4.2.5. Support Vector Machines (SVM)

Support Vector Machines are a powerful set of supervised learning algorithms widely used for classification and regression tasks. The SVM method is well-suited for small datasets and is able to model non-linear relationships through the use of kernel functions. Unlike methods that aim to minimize overall error, SVM focuses on finding an optimal hyperplane that maximizes the margin between different classes, leading to improved generalization performance.

At its core, an SVM aims to find a hyperplane in *n*-dimensional space (where *n* is the number of features) that best separates data points belonging to different classes. "Best" is defined as maximizing the distance between the hyperplane and the nearest data point from each class – these nearest points are known as support vectors.

The optimization problem to be solved is:

$$\min_{w,b}\left(\frac{1}{2}||w||^2\right) \tag{20}$$

$$subject\ to\colon\ y_i(w^T x_i + b) \geq 1\ for\ all\ i = 1,\ldots,n$$

where:

w – weight vector defining the hyperplane's orientation.

b – bias term determining the hyperplane's position.

x_i – the *i*-th data point (feature vector).

y_i – the label of the *i*-th data point (+1 or -1 for binary classification).

$w^T x_i + b$ – the decision function, determines which side of the hyperplane a point falls on.

$||w||^2$ – the squared magnitude of the weight vector (used to minimize, effectively minimizing the norm of w and maximizing margin).

The margin is the distance between the hyperplane and the support vectors, and a larger margin generally indicates better classification accuracy on unseen data.

$$margin = \frac{2}{||w||} \tag{21}$$

For linearly separable datasets, this process is straightforward. However, many real-world problems involve non-linearly separable data. SVM addresses this challenge through the use of kernel functions. These functions map the original input features into a higher-dimensional space where linear separation becomes possible. The specification adopted for this paper uses a Radial Basis Function (RBF) Kernel, which is particularly effective in high-dimensional spaces and where there are potential non-linear relationships within the dataset. It uses a gamma parameter to control the influence of individual training examples.

The kernel function computes the dot product of two data points in the transformed feature space:

$$K(x_i, x_j) = \Phi(x_i)^T \Phi(x_j) \tag{22}$$

where:

$\Phi(x)$ – The mapping function that transforms the input features into a higher-dimensional space. While we need not know Φ explicitly, we require K.

The Radial Basis Function (RBF) or Gaussian kernel is defined as:

$$K(x_i, x_j) = e^{-\gamma ||x_i - x_j||^2} \tag{23}$$

where, $\gamma > 0$, is a kernel parameter.

In this case, the modified optimization problem now uses the kernel function instead of the explicit dot product:

$$\min_{w,b,\xi} \left(\frac{1}{2} ||w||^2 + c \sum_{i=1}^{n} \xi_i \right) \tag{24}$$

$$subject\ to: y_i(w^T \Phi(x_i) + b) \geq 1 - \xi_i \ \ for\ all\ i = 1, \dots, n,$$

$$\xi_i \geq 0 \ \ for\ all\ i = 1, \dots, n.$$

The gamma parameter (γ) defines how far the influence of a single training example reaches. Small values mean a larger radius of influence, thus smoother decision boundaries. Larger values create more localized borders that fit training data well, but can overfit.

Additionally, the cost parameter (c) controls the trade-off between low training error and having a large margin. Therefore, it penalizes misclassifications. A higher cost means more emphasis on correctly classifying training data, potentially leading to overfitting, while a small value encourages a larger margin but may allow for more misclassifications (soft margin).

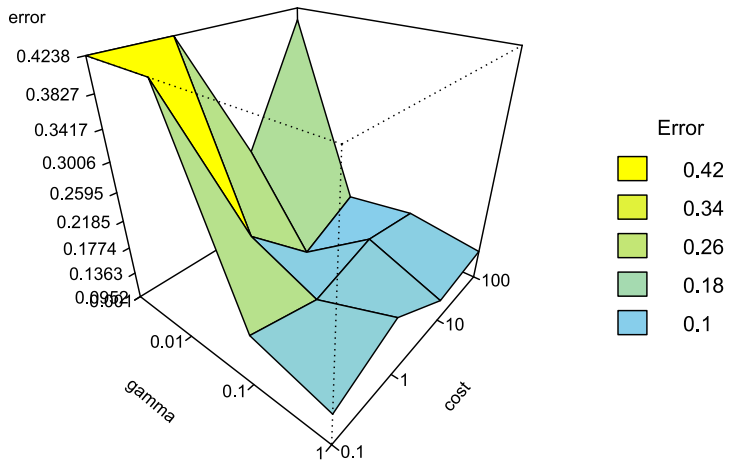
Selection of the two parameters, is achieved through 10-fold cross-validation using a search grid on the intervals of: 0.1 – 100 for the cost parameter and 0.001 – 1 for gamma.

The optimal values for the parameters are found to be:

- γ = 0.1 and c = 1 (for the 4-variable model)
- γ = 0.1 and c = 10 (for the 6-variable model)

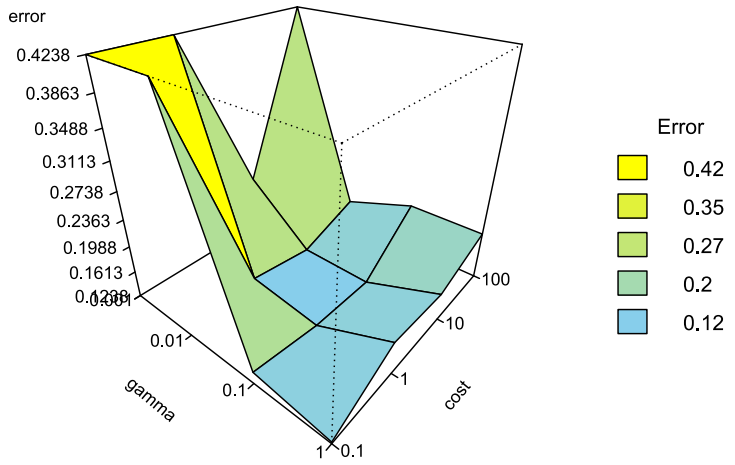
The complete results of the tuning process can be visualized as areas with regions of high and low errors, depending on the combination of cost and gamma parameters that are tested. The grid search finds the point with the lowest error and returns the optimal combination of the gamma and cost parameters that achieve the minimal error.

Figure 4. SVM parameter tuning results for the 4-variable model



This figure shows the results of the tuning process for the 4-variable model. The figure depicts lower error with darker shades of blue. The bottom region of the figure, corresponding to the lowest errors, can be found where the cost parameter equals 1 and the gamma parameter equals 0.1.

Figure 5. SVM parameter tuning results for the 6-variable model



This figure shows the results of the tuning process for the 6-variable model. The figure depicts lower error with darker shades of blue. The bottom region of the figure, corresponding to the lowest errors, can be found where the cost parameter equals 10 and the gamma parameter equals 0.1.

4.3. Deep Learning (DL) methods

Deep learning uses artificial neural networks with multiple layers to analyze data and learn complex patterns. These models excel at tasks like natural language processing. Recent advancements include large-scale Large Language Models (LLMs), such as GPT, capable of generating human-quality text and summarizing documents.

4.3.1. Neural Networks (NN)

Artificial Neural Networks are computational models inspired by the structure and function of biological neural networks found in animal brains. They are a core component of modern machine learning, particularly effective at identifying complex non-linear relationships within data. Unlike traditional algorithms that require explicit programming for specific tasks, Neural Networks learn from data by adjusting internal parameters to minimize prediction error. This ability makes them well-suited for problems where defining precise rules is difficult or impossible, such as classification in the presence of non-linear dynamics in the data.

For this research, I employ a fully connected neural network (FCNN), also known as a Multi-Layer Perceptron (MLP). This specific configuration is a fundamental type of neural network, characterized by layers of interconnected nodes (or neurons). Each neuron in one layer is connected to every neuron in the subsequent layer, hence the term "fully connected." Information flows unidirectionally from the input layer, through one or more hidden layers, to the output layer.

The strength of each connection is represented by a weight, and each neuron applies an activation function to its weighted sum of inputs, introducing non-linearity into the model. Given the binary classification task,

this paper uses the logistic function as an activation function. This combination of weighted sums and non-linear activations allows FCNNs to approximate any continuous function, given sufficient network capacity.

The process of adjusting the weights within the network is known as training. supervised learning is utilized, where the network learns from labeled data – pairs of input features and corresponding target values. The network's performance is evaluated using a loss function that quantifies the difference between predicted outputs and true targets. Cross-Entropy is used for classification tasks.

To minimize the loss function, the backpropagation algorithm coupled with Stochastic Gradient Descent (SGD) is employed. Backpropagation efficiently calculates the gradient of the loss function with respect to each weight in the network, allowing for iterative updates that reduce error.

A fully connected neural network operates through a series of linear transformations and non-linear activations. Its functioning is represented by the succession of several key operations for a single layer *l*, where $l = 1,2,\dots,L$, with *L* representing the total number of layers in the network.

The input to layer *l* is denoted as $x^{(l-1)}$, which is the output from the previous layer (or the input data for the first layer, $l = 1$). Each neuron in layer *l* receives a weighted sum of its inputs. This can be expressed as:

$$z^{(l)} = W^{(l)}x^{(l-1)} + b^{(l)} \tag{25}$$

where:

$z^{(l)}$ – is the vector of weighted sums for layer *l*.

$W^{(l)}$ – is the weight matrix connecting layer $(l-1)$ to layer *l*. Its dimensions are $(n_l \times n_{l-1})$, where n_l is the number of neurons in layer *l* and n_{l-1} is the number of neurons in layer *(l-1)*.

$x^{(l-1)}$ – is the vector of activations from the previous layer $(l-1)$.

$b^{(l)}$ – is the bias vector for layer *l*, with dimensions $(n_l \times 1)$.

The weighted sum $z(l)$ is then passed through an activation function, denoted as σ, to introduce non-linearity:

$$a^{(l)} = \sigma\left(z^{(l)}\right) \tag{26}$$

where:

$a^{(l)}$ – is the vector of activations for layer *l*.

σ – represents the activation function. The specific form of σ depends on the chosen activation function

The final layer (*L*) produces the network's output, denoted as $\hat{y}$. This is simply the activation of the last layer:

$$\hat{y} = a^{(L)} = \sigma\left(z^{(L)}\right)$$

The process of calculating the output from input data by sequentially applying these operations through all layers is called forward propagation. Essentially, one iterates:

$l = 1\ to\ L$:

- $z^{(l)} = W^{(l)}x^{(l-1)} + b^{(l)}$
- $a^{(l)} = \sigma(z^{(l)})$

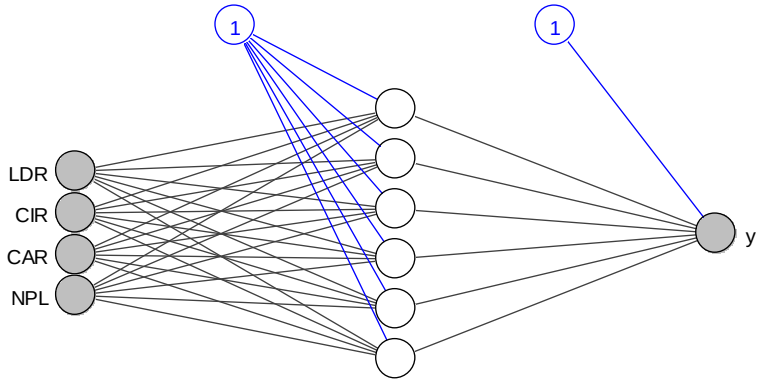
During training, a loss function *J* quantifies the difference between the network's predictions ($\hat{y}$) and the true target values (y). The goal of training is to minimize this loss function using optimization algorithms like gradient descent.

For the purpose of this paper, fully connected neural networks utilizing resilient backpropagation with weight backtracking are configured. The error calculation function is taken to be the sum of squared errors. The learning rate is set to 1.2 and the threshold for the partial derivatives of the error function as stopping criteria is 0.01, with the maximum steps for the training of the neural network set to 100000 (although in the majority of cases, the neural network completes its training in less than 1000 steps). The logistic activation function was retained given the binary classification problem, coupled with a shallow network configuration which is sufficient to capture the moderate complexity of interactions between the variables included in the study.

The optimal structure of the neural network is found by iterating through combinations of hidden layers and units ranging from one to ten hidden units on up to three hidden layers. The relatively compact specifications are sufficient to handle the small size of the dataset and moderate number of features. In total 2220 neural networks are trained across the two model specifications and their results are compared to find the best performance.

The resulting optimal structure for the 4-variable model specification is found to be:

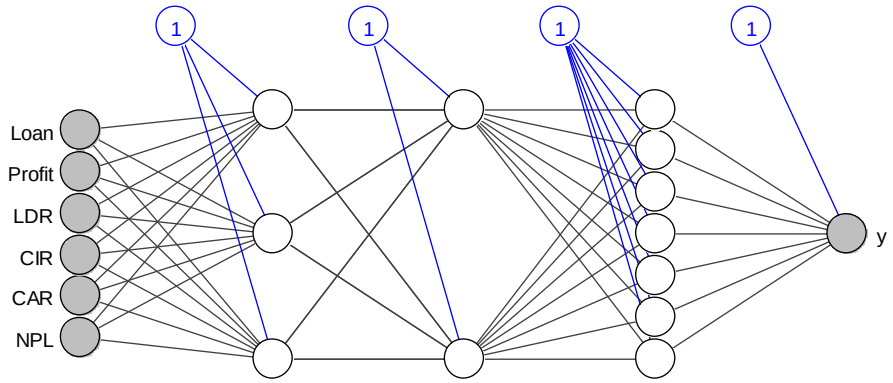
Figure 6. Optimal neural network structure for the 4-variable model



The optimal structure for the 4-variable model is found to be a network with one hidden layer and six hidden units. This shallow network is sufficient to capture the dynamics of the compact model specification.

As for the 6-variable specification, its optimal structure is a three-layer network.

Figure 7. Optimal neural network structure for the 6-variable model



The optimal structure for the 6-variable model is found to be a network with three hidden layers with 3, 2 and 7 hidden units on each of the three layers, respectively. This deeper network captures the dynamics of the extended model specification.

4.3.2. Large Language Models (LLMs)

The field of artificial intelligence has witnessed a significant development with the advent of Large Language Models. These models represent a substantial advancement in the domain of natural language processing, demonstrating a remarkable capacity to comprehend and generate human-like text. An LLM is a sophisticated neural network model, distinguished by its extensive number of parameters, often numbering in the billions. This scale, coupled with training on vast and diverse datasets, enables these models to acquire a nuanced understanding of language, encompassing grammar, semantics, and even elements of reasoning. The development of LLMs has been propelled by advancements in computational power and the availability of massive textual corpora, allowing them to perform a wide array of language-based tasks without task-specific training.

The functional underpinning of modern LLMs is the transformer architecture, a novel neural network design introduced in 2017. The transformer architecture departed from the sequential processing of recurrent neural networks (RNNs), which were previously the standard for language tasks. Instead, it processes entire sequences of text simultaneously, a design choice that facilitates parallel computation and, consequently, training on much larger datasets. The architecture is composed of a stack of encoders and decoders. The encoder's role is to process the input text and build a contextual representation of it. This representation captures the relationships between all words in the input sequence. The decoder then utilizes this contextual information to generate the output text, one word at a time, in a coherent and contextually appropriate manner.

A foundational component of the transformer architecture is the self-attention mechanism. This mechanism allows the model to weigh the importance of different words in the input text when processing a particular word. For instance, when encountering the word "it" in a sentence, the self-attention mechanism helps the model determine which preceding noun "it" refers to by attending to the most relevant words in

the context. This is achieved by creating three vector representations for each word in the input sequence: a Query vector, a Key vector, and a Value vector. The model calculates a score between the Query vector of the current word and the Key vectors of all other words in the sequence. These scores are then used to create a weighted sum of the Value vectors, effectively focusing the model's 'attention' on the most pertinent parts of the input to inform its understanding and subsequent output. This ability to dynamically assess word relationships across long distances in a text is a significant contributor to the coherence and accuracy of LLM-generated content.

The training of a Large Language Model is a multi-stage process, beginning with what is known as pre-training. During this phase, the model is exposed to a massive and diverse corpus of text data from the internet and digitized books. This pre-training is typically unsupervised, meaning the model learns from the raw text without explicit labels. The objective is for the model to learn the statistical patterns and structures of language. A common pre-training task is to predict a masked or missing word in a sentence, forcing the model to develop a deep understanding of context and word associations. This initial phase is computationally intensive and is what endows the model with its general linguistic capabilities.

Following pre-training, a model undergoes a process of fine-tuning to enhance its performance on specific downstream tasks. Fine-tuning involves training the pre-trained model on a smaller, task-specific dataset. This dataset is labeled, providing the model with explicit examples of the desired input-output behavior. For example, a model can be fine-tuned for summarization by training it on a dataset of articles paired with their summaries. This process adapts the general linguistic knowledge acquired during pre-training to the specific nuances of the target task. The fine-tuning process is substantially less resource-intensive than pre-training and allows for the specialization of a general-purpose LLM for a wide range of applications, from translation and question answering to code generation and creative writing. Through this two-phased training regimen, LLMs achieve a versatile and powerful command of language, enabling them to function as highly capable tools in a multitude of domains.

Indeed, LLMs exhibit exceptional proficiency across a broad spectrum of general natural language processing tasks. Their core ability to understand, interpret, and generate human-like text forms the basis for these applications. LLMs can efficiently condense large volumes of text into concise and coherent summaries, extracting key information while maintaining the original context. They can interpret natural language questions and generate precise, contextually relevant answers by drawing upon the vast knowledge acquired during training.

The versatility of LLMs allows for their adaptation to highly specialized domains, where they offer significant value by processing and generating domain-specific language with high accuracy. In finance, LLMs are utilized in the financial services industry to translate complex financial data into human-understandable language, streamline data processing, and improve decision-making. Key applications include AI risk management and fraud detection. Concrete examples of specialized models for finance include models such as BloombergGPT[31] and FinGPT[32], which are trained on extensive financial datasets to address industry-specific NLP tasks.

The continuous refinement of these models and their training methodologies suggests a future of even more sophisticated and integrated linguistic AI.

4.4. Cross-validation

In statistical modeling and machine learning, it is important to assess performance on test samples to understand how well the model generalizes to unseen data. Cross-validation emerges as a robust technique that addresses this need by providing a more accurate estimate of a model's predictive power compared to traditional train-test splits. Particularly when dealing with small sample sizes, cross-validation is beneficial due to its capacity to maximize the use of available data and reduce variance in performance estimates.

Cross-validation operates on the principle of systematically partitioning a dataset into complementary subsets, training the model on some partitions (the training set) while evaluating it on others (the validation or test set).

For small sample sizes, where the luxury of large datasets is absent, random cross-validation methods, such as repeated random sub-sampling or Monte Carlo cross-validation, offer significant benefits. These methods involve repeatedly partitioning the data into training and testing subsets in a randomized manner and averaging the performance metrics over multiple iterations. This process ensures that every observation has an equal probability of being included in each subset, effectively utilizing all available data.

One of the primary advantages of random cross-validation is its flexibility and adaptability to varying dataset sizes. Unlike k-fold cross-validation, which fixes the number of partitions, random sub-sampling allows for arbitrary sizes of training and testing sets. This can be particularly useful when dealing with small datasets where retaining more data in each training fold (as in k-folds) might not leave enough data for a reliable validation set.

Moreover, repeated iterations in random cross-validation help stabilize performance estimates by reducing the variance associated with any single partition. Each iteration provides a unique train-test split, and averaging results across multiple splits offers a more comprehensive view of model performance under different conditions. This is important when datasets are limited, as it reduces the likelihood of outlier partitions disproportionately influencing the evaluation metrics.

As the collected data sample is relatively small, in order to produce more accurate model performance metrics, multiple rounds of random cross-validation are performed. In each round, the set of banks used to train the model is different, as is the test set. The training set is constrained to include 80% of the observations, with the remaining 20% being left for the test set. This allows for a better understanding of the model's performance on different configurations of banks.

For each of the eight retained statistical, Machine Learning and Neural Network methodologies, one hundred rounds of model runs are performed. Given that two model configurations are estimated, a four-variable and a six-variable specification, this requires the estimation of 1600 models.

The LLM model does not undergo prior training on the existing data and relies only on the knowledge bank that it has acquired during general-purpose training. As such, there is no scope for cross validation for the LLM model. Instead, its performance will be measured on the entirety of the dataset, which serves as a test set. This makes the results comparable to those produced by the cross-validation mechanism adopted for this paper, in which the test set differs by cross-validation iteration. Unlike the other methods which generate

predictions for all banks at once, the LLM is set to generate predictions separately for each bank, so that information leakage between financial institutions is excluded from the model's context. This increases the number of estimations required to generate workable output to 64, but eliminates the model runs associated with cross-validation. Despite the lower number of overall runs associated with the LLM method, the computation time is greater due to the complexity of the selected model and the size of the context.

Taken together with the LLM model runs performed for each of the banks included in the sample, the total number of estimations performed increases to 1664.

5. Model specification

The interplay between financial performance and risk metrics reveals much about a bank's profile. For instance, a combination of high NPLs with a low CAR indicate concurrent credit stress and insufficient capital reserves to cushion against losses, suggesting heightened default risk. Similarly, declining ROA, increased operating costs, potentially coupled with liquidity challenges can signal operational inefficiencies alongside potential short-term funding issues.

Beyond traditional metrics, integrating advanced analytical techniques such as machine learning enhances predictive capabilities. These methods can process datasets to uncover complex relationships between variables that might elude conventional analysis. Further improving the quality of the analysis are LLMs, capable of considering the broader economic context, financial institutions' strategic choices, and key risks identified by the entities themselves or derived from a larger range of data that they provide in both structured and unstructured formats.

Ultimately, a holistic multi-modal approach allows regulators and the institutions themselves to anticipate potential defaults more accurately by capturing a nuanced picture of both internal financial health and external conditions, thereby facilitating timely interventions aimed at mitigating risk.

5.1. Variables used in constructing the classification and regression models

Eight models, two of which statistical, five representing the leading Machine Learning algorithms for classification, and one of the most commonly used Deep Learning methods are estimated on the collected data.

- Logistic regression
- Linear regression
- Random Forest
- XGBoost
- Naïve Bayes
- k-Nearest Neighbors
- Support Vector Machines
- Neural Network

Two specifications of each model are run, a four and a six-variable configuration. The six variables that the full model includes are the: loan growth, change in profit, loan-to-deposit ratio, cost-to-income ratio, Capital

Adequacy Ratio and NPL ratio. The abridged version omits loan growth and the change in profit, focusing instead only on the performance ratios.

The complete version includes all the variables in the dataset aiming to make maximal use of sparse data.

$$PD = \beta_0 + \beta_1 loansd + \beta_2 droa + \beta_3 ldr + \beta_4 cir + \beta_5 car + \beta_6 npl + \varepsilon \quad (1)$$

The abridged model tests if financial ratios, which are widely considered to be efficient in evaluating a bank's financial health are sufficient to produce accurate estimates of a bank's default.

$$PD = \beta_0 + \beta_3 ldr + \beta_4 cir + \beta_5 car + \beta_6 npl + \varepsilon \quad (2)$$

Each method-specification pair is run over 100 iterations for cross-validation, which leads to an estimation of a total of 1600 models.

Anticipating the direction of the explanatory variables' impact on the dependent variable helps in specifying the correct form of the regression model. Ensuring that the anticipated signs are consistent with economic theory helps validate the model's foundations. This consistency strengthens the pertinence of the results and ensures they align with established financial principles and empirical evidence. While sometimes the relationship is clear-cut, there are also instances when the sign of the impact may vary as a function of certain entity characteristics, system dynamics or temporal factors. For this reason, thinking about the effects that variables produce ex-ante can elucidate specific transmission mechanisms or underlying characteristics of the system.

Loan growth (LOANSD)

The relationship between loan growth and a bank's PD is intricate, often demonstrating a non-linear or U-shaped trend. Moderate and sustainable loan growth reflects a thriving economic environment and growing business operations, enhancing interest income and bolstering the bank's financial standing. However, excessively rapid loan expansion generally heightens default risk. Aggressive lending during such periods can lead to relaxed credit standards, increased exposure to high-risk borrowers, and entry into unfamiliar markets, which raises future occurrences of non-performing loans (NPLs) and loan losses. This increases vulnerability to default. On the other hand, consistently low or negative loan growth might indicate economic constraints or limited profitable lending opportunities, affecting long-term revenue and financial strength.

Profit variation (DROA)

The change in profit is a direct indicator of operational efficiency and financial health. A positive change is typically associated with a reduction in the default probability. Consistent profitability boosts retained earnings, strengthening the bank's capital reserves and increasing its buffer against unforeseen losses from credit defaults or market volatility. Strong profits reflect effective risk management, sound operations, and a sustainable business model, all contributing to stability. Conversely, a decline in profit suggests financial deterioration, depleting capital reserves and weakening resilience to adverse conditions, thus increasing the bank's susceptibility to default.

Loan-to-Deposit ratio (LDR)

The Loan-to-Deposit ratio (LDR) reflects a bank's liquidity risk and funding structure. A higher LDR often correlates with increased default probability, while a lower ratio generally reduces it. A high LDR shows that much of the deposit base is loaned out, boosting interest income but reducing liquid assets, increasing vulnerability to liquidity shocks like sudden deposit withdrawals or unexpected funding needs. It may also indicate reliance on volatile and costly wholesale funding sources, heightening funding risk. Conversely, a lower LDR suggests conservative liquidity management with more deposits held in liquid form, providing a buffer against liquidity demands and lowering default probability.

Cost-to-Income ratio (CIR)

The cost-to-income ratio (CIR) measures operational efficiency. A higher CIR is linked to increased default probability, while a lower ratio denotes decreased risk of default. Excessive operating expenses relative to income, reduce net profitability and limit capital growth potential, thereby decreasing resilience to economic downturns or unexpected losses. Elevated costs can also suggest management inefficiencies, contributing further to risk. In contrast, a low CIR signifies effective cost control and operational efficiency, leading to greater profitability, strengthening the capital base, and enhancing shock absorption capacity, thus reducing the probability of default.

Capital Adequacy Ratio (CAR)

The capital adequacy ratio (CAR) is a mainstream indicator used to assess financial strength and loss-absorption capability. A higher CAR correlates with decreased default probability, while a lower CAR is reflective of heightened risk of default. The CAR measures the bank's capital against risk-weighted assets, serving as the main buffer against unexpected losses from credit, market, or operational risks. Strong capital reserves allow a bank to absorb asset write-downs and financial shocks without insolvency, protecting depositors and creditors. Regulatory standards like Basel Accords enforce minimum CAR levels for banking stability. Banks with inadequate capital are more vulnerable to adverse events and are therefore expected to have a higher default probability.

Non-Performing Loan (NPL) ratio

The non-performing loan (NPL) ratio is perhaps the most direct sign of a financial institution's risk of default. The indicator is used to assess asset quality and credit risk management effectiveness. A higher NPL ratio significantly increases the likelihood of default, while a lower ratio decreases it. The NPL ratio indicates the share of the loan portfolio with delinquent payments over a specific period. High NPL ratios directly impair a bank's profitability by means of reduced interest income and increased provisioning for loan losses, which directly erodes its capital base. Higher NPLs can be a signal of weaknesses in loan underwriting standards, poor portfolio monitoring and management or economic challenges affecting borrowers. Thus, a high NPL ratio is a strong indicator of potential financial distress and a key driver of bank default probability.

Analyzing these variables allows for an initial overview of a bank's vulnerability to financial distress However, the values of the variables and the estimated relationships alone provide little information about the overall financial situation of a bank (outside the select indicators included in the regression) and events surrounding

the reported data. Therefore, coefficient estimates can only be interpreted as the observed effect on a typical bank (included in the regression sample), but not on the specific bank under analysis. Large Language Models, with their ability to process the context surrounding the data are able to tailor the analysis to individual institutions, thereby providing a more relevant interpretation of the results. This allows for more data to be considered, deeper studies to be produced and, consequently, improved decision-making.

5.2. LLM-powered risk monitoring and default prediction model

The LLM workflow is built utilizing Google's Gemini 2.5 Pro model. The model presents a number of key advantages over other LLMs which prove to be particularly useful in analyzing financial reports.

The Gemini 2.5 Pro model is a significant advancement in the field of LLMs and is characterized by its robust capabilities in complex data processing and extensive textual analysis, making it highly suitable for academic and research applications involving voluminous datasets. An important feature in this regard is its expansive one-million token context window, which facilitates the ingestion and analysis of vast quantities of textual data within a single input. This capacity renders the model suitable for tasks necessitating a profound understanding of detailed financial reports, documentation, technical notes or any other large corpora of unstructured text, mitigating the challenges associated with context fragmentation in extended documents. Beyond its proficiency with textual inputs, the model is engineered to process diverse data modalities, including images, thereby broadening its applicability across various potential tasks.

The latest iteration exhibits advanced optical character recognition functionalities, enabling improved extraction of textual content from both image-based and scanned documents. This capability encompasses the identification and interpretation of structured data embedded within tables and the transcription of handwritten elements allowing for the retrieval of information from scanned financial records. The model supports operations such as document summarization, targeted question answering, and structured information extraction directly from complex layouts, consequently optimizing research workflows that traditionally rely on labor-intensive manual data discovery and entry.

The paper adopted the most recent LLM model version available at the time it was written and the hyperparameters used follow the values recommended by the model developers.

Model specifications	
Model name	Gemini 2.5 Pro
Version	2025-06-05
Max. input tokens	1,048,576
Max. output tokens	65,535
Type	Multimodal
Hyperparameters	
Temperature	1 (range 0 – 2)
topP	0.95 (range 0 – 1)
topK	64 (fixed)
candidateCount	1 (range 1 – 8)
Input type	
Query	Text
Context	PDF, TXT

The default hyperparameters were used in the model configuration to ensure optimal results and conformity with standard Gemini 2.5 Pro outputs.

The model seed was not user-configurable at the time when the analysis was conducted, hence a special validation stage was introduced to cover for potential variability resulting from generative randomness.

Although the model input is limited to 500MB, with an additional limit of 50MB per upload and 1000 pages per textual file – one report consisting of just over 600 pages failed to process giving an error. It was shortened to fit.

5.2.1. The LLM risk assessment model

The aim of the model is to provide a forward-looking risk assessment for financial institutions with a three-pronged objective: assessing the current situation of the institution, identifying the key risks that it faces and evaluating its probability of default.

A language model (LLM) can effectively analyze a bank or NBFI's financial situation using a sequential reasoning process which can be broken down into four steps.

Step 1. Describing the current conjuncture

Initially, the LLM describes the current conjuncture of the bank. Its objective is to provide an overview of the institution's current financial and operational status. It gathers relevant data from the supplied financial statement(s) deriving information about the bank's key indicators, market conditions, regulatory changes, and major events.

The LLM analyzes this data to describe:

- Key indicators – Capital adequacy, liquidity indicators, asset quality, etc.
- Market position – Market share, competitive landscape, customer base.
- Economic context – Impact of macroeconomic factors like interest rates, inflation, and economic growth on the bank.
- Major events – Major losses or gains, important shifts in strategy, restructurings, share emissions or any other significant events capable of incluencing the bank's financial situation.

The output of this phase is a comprehensive summary that captures the bank's operational environment, highlighting strengths and weaknesses, as well as key indicators highlighted in the report.

Step 2. Identifying the main drivers of risk

The goal of this phase is to Identify key risk factors affecting the bank. Attention is given to the impact of different risk categories such as credit risk, market risk, operational risk, liquidity risk, and compliance risk. Aside from the solvency and liquidity indicators that were collected in the first phase, the LLM examines both internal and external factors that are relevant for the analysis. Internal factors include governance structures, management quality and internal controls, while external factors focus on economic volatility, regulatory changes and geopolitical events. The output of this part of the workflow is a detailed list of primary risks with explanations on how they could impact the bank.

Step 3. The information-driven risk score

In order to quantify the level of risk faced by the bank, the model utilizes not only data in the form of indicators, but also the context in which the data resides. Therefore, the generated metric will be not only

"data-driven", but it will surpass the abilities of traditional data science and analysis, becoming an "information-driven" score.

The generated risk score is constrained to a range between 1 to 5 and is used to evaluate the severity of the risk to which the bank is exposed. It is notable that an increase in risk can originate from any of the previously-identified categories and the model explains the source of the risk.

The model uses established frameworks like Basel III or other industry standards to evaluate risk levels. It incorporates all the information from previous analyses, which includes data such as financial metrics, as well as identified risks and situational context.

In producing the score, the model takes into consideration both the:

- Probability of adverse events occurring across the different key risks
- Potential impact on the bank's operations and finances.

The output of this stage is a numerical risk score indicating the overall risk level of the bank.

Step 4. The information-driven profitability score

The final step is for the model to give an assessment of the institution's potential for profitability. This is separate from the entity's risk profile. When processing the documents to produce the score, the LLM considers

- Profitability indicators – such as Return on Assets (ROA), Return on Equity (ROE), net interest margin, and non-interest income.
- Trend analysis – evaluates historical performance trends from fata points and references provided in the provided documents and projects future profitability based on current strategies and market conditions.
- Competitive benchmarking – compares the bank's profitability metrics against industry standards to gauge relative performance.

The output is similar to that of the risk metric, whereby a profitability score or rating is produced, which reflects the bank's ability to generate profit under current and projected conditions.

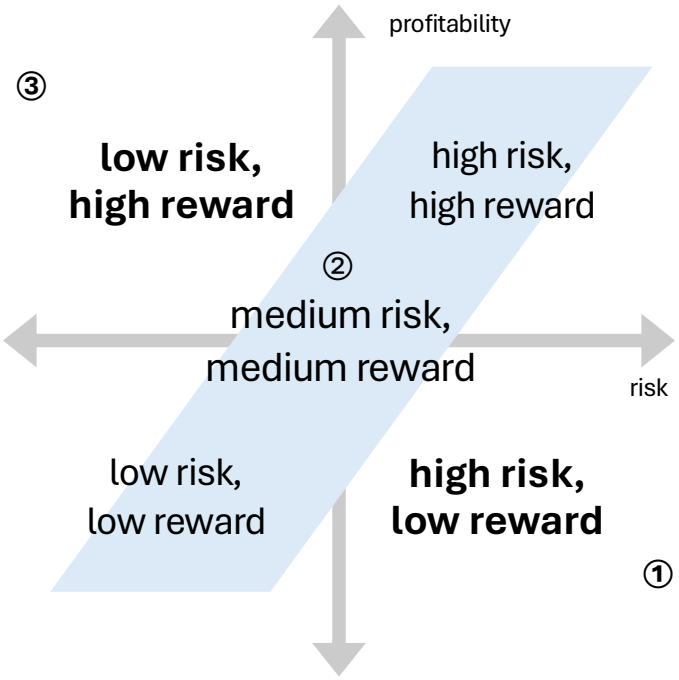
These content processing steps allow the LLM to provide a structured analysis of a bank's situation, identifying key risks and assessing both risk exposure, as well as the potential for profitability. This approach helps regulators to clearly identify the sources of risk, its intensity, the extent of the institution's vulnerabilities, potential mitigants, as well as an overview of the environment in which the bank operates.

5.2.2. The probability of default model

The probability of default metric is constructed by drawing from the classical notion of "risk-reward" adapted for financial institution performance and vulnerability.

The "risk-reward" concept is a fundamental principle in finance that reflects the relationship between the potential return on an investment and the amount of risk taken to achieve those returns. It is based on the premise that higher potential rewards are usually associated with greater risks. The principle suggests that to achieve higher rewards, investors must be willing to accept a higher level of risk. Conversely, lower-risk investments typically offer lower potential rewards.

Figure 8. Dimensions of risk and reward for LLM classification



The figure displays the dimensions of risk and reward considered by the LLM. With increasingly contrasting scores between risk and reward, comes increased confidence in the default or non-default classification.

In the implementation adopted in this paper, the premise is that banks are exposed to risk, which is a factor that increases their probability of default. However, highly profitable institutions, despite being risky, can utilize their high financial gains to compensate for short-term losses. As such, based on the overall level of risk to which a financial institution is exposed and its potential for profitability, a bank, or an NBFI, can fall into one of three circumstances:

① Higher risk than profitability – this is the category which has an increased probability of default
② Risk roughly equal to profitability – this is the "knife's edge" category whose equilibrium could fall into the direction of default under aggravating external or internal evolutions of events
③ Higher profitability that risk – this is the ensemble of institutions which are healthy going concerns that pose no particular threat to the financial stability of the system.

Visualizing the concept in matrix form underscores the challenging nature of the classification task which has two quadrants where predictions are unambiguous, surrounded by other regions, where deciding on the financial situation of the bank is significantly more difficult.

Box 1: Understanding the choice of grading options given to the LLM

The number of grades that the model can assign is purposefully kept low in order to avoid the mathematical processing issues that are associated with LLMs. In particular, the following constraints limit what LLMs can do in terms of mathematical calculations and logic:

- **Pattern recognition and data sparsity**
  LLMs excel at identifying patterns in data. With only 5 categories, there is more underlying information per category relative to the overall context. This allows the model to learn stronger associations between each category and the relevant chunks of information. A large number of possible scores makes it harder for the LLM to reliably distinguish between similar categories.
- **Reduced ambiguity and overlap**
  Fewer categories inherently mean less ambiguity and overlap between them. The boundaries between 5 categories are likely more distinct than those between 100, making classification easier. With a larger number of categories, there's a higher chance that an input could plausibly belong to multiple classes, leading to errors.
- **Reduced Bias**
  Large scales trigger "integer bias" (favoring round numbers like 70, 80, 90) or "central tendency bias" (where the model stays in the middle of the score distribution).
- **Computational complexity**
  While not likely a major bottleneck in the configuration adopted by this paper, increasing the number of output categories increases the computational complexity of the LLM's final layer (often a softmax layer). This can lead to slower processing and potentially reduced accuracy, especially with limited resources.

Zhuang et al. (2024)[33] explore how to improve scoring using large language models (LLMs) by incorporating fine-grained relevance labels into the prompts. Experiments across eight datasets demonstrate significant performance gains with this approach. However, their analysis reveals diminishing returns; increasing the granularity of the score beyond 8 levels decreases performance, suggesting LLMs struggle to effectively process excessively fine-grained scores.

Stureborg et al. (2024)[34] reveal that when using LLMs like GPT-3.5-Turbo and GPT-4 to rate summaries on a large continuous scale (i.e. 1-100), the models exhibit strong biases towards specific scores and round numbers, and don't fully utilize the entire range. In contrast, they find that a five-category score construct leads in a number of performance metrics and that good performance can be achieved with up to 10 score categories.

To understand the root cause of the difficulties LLMs face in fine-grained scoring, we can turn to standard machine learning theory: a model trained on limited data is prone to overfitting when the number of classes is high, leading to poor performance on unseen examples. Techniques like regularization and data augmentation can help, but they aren't always sufficient to overcome severe data sparsity.

In this set-up, higher levels of risk are associated with increased confidence in the probability of default, whereas profitability acts as a mitigant to the probability of default and lowers it. The value taken by the profitability indicator, similarly to that of the risk metric, signify the degree of confidence in the existence of mitigating circumstances attenuating the risk of default.

By configuring both the risk and profitability scores to take maximum intensity values at the number 5 on a symmetric scale of 1 to 5, and computing the difference between risk and profitability, it is possible to construct a metric which, if positive, signifies a heightened probability of default, whereas negative values denote decreased PD levels.

$$PD = Risk - Profitability \tag{27}$$

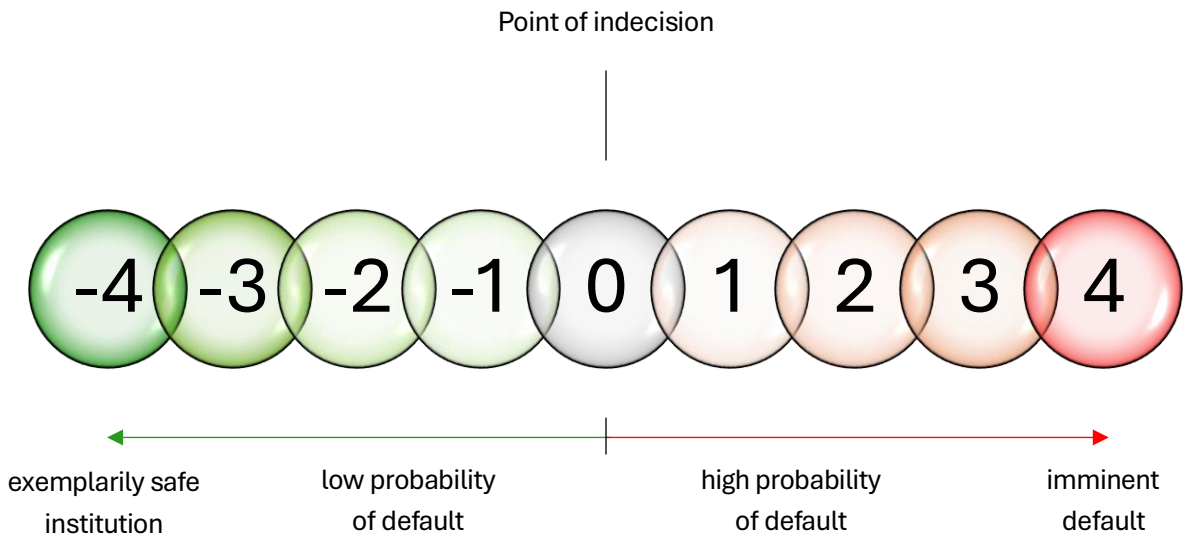
Where:

Risk – is the risk score given by the LLM which ranges on a scale from 1 to 5

Profitability – is the profitability score issued by the model, equally ranging on a scale from 1 to 5

In both cases, higher values, indicate a more confident assessment of the respective factor's intensity by the large language model.

Figure 9. Default score range and variation



The figure displays the range of values that the probability of default score can take, highlighting areas of low risk where the indicator is negative and high-risk zones, where the indicator is positive. At zero, there is doubt about the future evolution of the financial institution.

The higher the distance between risk and profitability, the higher the model's conviction is regarding the outcome of the entity – default or non-default. With a score of 4, the model is perfectly confident that the

bank is in imminent danger of defaulting and a score of -4 represents full conviction that the report describes an ideal bank operating in a low-risk environment and generating exceptionally high revenue. More generally:

- Values above zero represent varying degrees of conviction in the riskiness of the institution, but they all signify that the institution defaults
- Values below zero, conversely, represent varying degrees of conviction in the bank's soundness, but all indicate that the bank does not default
- A score of zero represents insufficient contextual information for the model to make a classification decision and therefore represents a lack of evidence to suggest that the institution defaults on its loans.

This mechanism ensures that the LLM's assessment is not biased towards a single viewpoint, that the LLM cross-validates its work and that the end-user obtains a metric of the model's confidence in the output (with values closer to zero indicating the presence of mitigating factors or insufficient evidence to make a clear classification).

5.2.3. Prompt Engineering

LLMs are good at understanding context even under imperfect conditions and the Google Gemini 2.5 Pro model excels at associating the user's prompt with the intended results even when sparse information is provided by the user.

Box 2: Getting the most out of an LLM doesn't take that much

A prompt consisting in directing the LLM to "*produce a score for the bank's risk level and a score for its profitability on a scale from 1 to 5*" is sufficient to produce results comparable with those reported in this paper. However, more nuance can improve the results for certain institutions where the data is ambiguous.

Techniques used to improve and refine outcome precision

Technique	**Additional prompt**
Complexification	Produce a highly detailed analysis of the bank's performance and profitability and assign a score [...]
Specification	Produce a score of the bank's risk level encompassing its solvency, liquidity and [...]
Personification	You are an experienced financial analyst working for a ratings agency. Your task is to produce scores for a bank's risk level and its profitability [...]

Techniques which fundamentally alter the outcome

Technique	**Additional prompt**	
Bias	**Negative**	Score this dangerous bank with a history of poor practices [...]
	Positive	Score this safe, reputable bank, well known for its impeccable [...]
Specification	**Negative**	[...] assume the bank is providing false or incomplete information [...]
	Positive	[...] ignore any information about poor management practices [...]
Focusing	**Temporal**	Current conditions matter more than historical performance.
	Sybject	Solvency issues are more stringent than liquidity issues.
Fixing	Score 1 means: [...]; Score 2 means: [...]; Score 3 means: [...]; [...]	

This paper adopts a conservative and succinct formulation for the query prompt and maintains the same formulation across all processed reports. No output-altering techniques are used to ensure that the results are unbiased and fair towards all institutions, even though the modeler is aware that some institutions were affected by prior negative events such as reputational issues, near-default events or deviations from regulation and norms. In reality, the modeler would have used this information in the prompt to change the depth and modality of the AI review process. However, for the purposes of this exercise, such contextual additions were not applied.

5.3. Model performance metrics retained for the analysis

In the area of prudential supervision, particularly concerning the classification of bank defaults, a number of performance metrics stand out as important in evaluating the predictive performance of the estimated models. These metrics provide insights into the model's ability to accurately identify default cases versus non-default cases, helping regulators and market participants make informed decisions based on the model's reliability and effectiveness.

True Positives (TP) and False Negatives (FN)

True Positives (TP) refer to instances where the model correctly predicts that a bank will default. These are critical in evaluating how well the model can identify actual default events. False Negatives (FN), conversely, occur when the model incorrectly predicts no default for banks that do indeed default. Minimizing FN is important as failing to detect defaults could result in significant financial losses and inadequate risk mitigation strategies.

False Positives (FP) and True Negatives (TN)

False Positives (FP) are cases where the model incorrectly predicts a default when it does not occur. These errors can lead to unnecessary actions, such as unwarranted financial interventions or heightened scrutiny of healthy banks, which may strain resources. True Negatives (TN) represent correctly identified non-defaults, indicating instances where the model accurately identifies banks that remain solvent.

Accuracy

Accuracy is defined as the proportion of total predictions (both TP and TN) that are correct out of all cases evaluated by the model. It is calculated as:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{28}$$

While accuracy provides a general sense of the model's overall performance, it can be misleading in imbalanced datasets where default events are rare compared to non-defaults.

Balanced Accuracy

Balanced Accuracy addresses the limitation of conventional accuracy by taking into account the balance between classes. It is calculated as the average of sensitivity (recall for the positive class) and specificity

(the true negative rate for the negative class). This metric provides a more nuanced view, particularly in datasets where there is an imbalance between default and non-default instances.

$$Precision = \frac{Sensitivity + Specificity}{2} = \frac{\frac{TP}{TP\ +\ FN} + \frac{TN}{TN\ +\ FP}}{2} \tag{29}$$

Precision

Precision, also known as Positive Predictive Value, measures the proportion of predicted defaults that are actually correct. It is calculated as:

$$Precision = \frac{TP}{TP\ +\ FP} \tag{30}$$

High precision indicates that when the model predicts a default, it is likely to be accurate, thus reducing the risk of unnecessary interventions based on false alarms.

Recall

Recall, or Sensitivity, quantifies the proportion of actual defaults that are correctly identified by the model. It is given by:

$$Recall = \frac{TP}{TP\ +\ FN} \tag{31}$$

High recall indicates that the model effectively captures most default events, which is essential for minimizing missed opportunities to address potential financial risks.

Specificity

Specificity, or True Negative Rate, measures the proportion of non-defaults that are correctly identified by the model. Calculated as:

$$Specificity = \frac{TN}{TN\ +\ FP} \tag{32}$$

Specificity reflects the model's ability to recognize banks that will not default, ensuring that resources are allocated appropriately and unnecessary actions are avoided.

F1 Score

The F1 Score is the harmonic mean of precision and recall, providing a single metric that balances both concerns. It is calculated as:

$$\mathrm{F1} = 2 * \frac{Precision\ *\ Recall}{Precision\ +\ Recall} = \frac{2 * TP}{2 * TP + FP + FN} \tag{33}$$

The F1 score is particularly useful when there is an uneven class distribution or when it is important to balance false positives and false negatives.

These metrics collectively offer a comprehensive framework for assessing the performance of predictive models in bank default classification. By examining TP, FN, FP, TN, accuracy, balanced accuracy, precision, recall, specificity, and the F1 score, stakeholders can gain a detailed understanding of a given model's strengths and weaknesses. This enables more informed decision-making processes concerning risk management strategies, the allocation of financial resources and regulatory intervention.

6. Results

The performance of the estimated statistical and Machine Learning models can be evaluated through an ensemble of indicators, specific to classification problems. Important metrics include the components of the confusion matrix, which reflect the True Positives (TP), False Negatives (FN), False Positives (FP) and True Negatives (TN). Of equal importance are the overall accuracy metrics, together with the balanced accuracy and F1 score. Also noteworthy are the results for precision and recall. These indicators, as well as a number of other important measures are discussed below.

In the case of failed institutions, by analyzing the financial reports that they published between one and four quarters before the default episode occurred, it is possible to test the early-warning properties of the model. From a regulatory perspective, a highly performing model is one that is able to accurately classify both defaulting and sound institutions, with extra importance assigned to avoiding misclassification of defaulting institutions as sound.

6.1. Performance of statistical, Machine Learning and Neural Network methods

The performance of all variable-driven methods (that is to say, all methods apart from the LLM) is compared on the test set of the model runs, separately for each of the two specifications.

In the four-variable specification, the SVM excels in all key performance metrics. The model's accuracy on the test set is 88.5% (and 87.8% in terms of Balanced Accuracy), producing the second lowest number of false negatives. The second-best performing model is the Naïve Bayes implementation, achieving an accuracy score of 87.8% (and 86.5% in terms of Balanced Accuracy). It is also the leading model in maximizing specificity and precision and minimizing false positives. On the other hand, the Random Forest model leads in terms of minimizing false negatives, as well as maximizing sensitivity and negative predictive value, positioning itself as the third best in terms of accuracy scores.

When running the six-variable specification, the Naïve Bayes model takes the lead with an accuracy score of 87.2% (and 86.3% for the Balanced Accuracy metric). It continues to dominate in terms of specificity and minimizing false positives, yet also gains superiority in terms of maximizing negative predictive value. In all the other performance metrics, it comes in as the third best. The second-best performing model in terms of accuracy becomes the neural network implementation. The underwhelming performance in the four-variable specification, and the significant improvement in the six-variable version, highlight neural networks' need for a larger feature set in order to produce meaningful results. With an even larger set of explanatory variables, the neural network could surpass the completion. As things currently stand, it is also the second-best in terms of minimizing false positives and maximizing specificity. In terms of minimizing false negatives,

the SVM is the model to beat, with the XGBoost coming in second and Naïve Bayes representing the third best option.

Among the statistical (linear and logistic regression) and ensemble machine learning (Random Forest and XGBoost) methods, which are more commonly used in the field of economic analysis, the Random Forest model proves to be capable of producing good results across the board in terms of performance metrics and consistently so, being the highest performer across different specifications.

Given the close performance of both model specifications, one can conclude that financial ratios, if properly selected for the task at hand, are sufficient to produce highly accurate estimates of the financial health of banks and NBFIs. While certain methods, like neural networks in particular, benefit from the inclusion of additional explanatory variables, most methodologies perform well even with more limited data.

Table 5. Performance of Statistical, Machine Learning and Neural Network models – test set

Spec.	Metric	Log	Linear	RF	XGB	NB	k-NN	SVM	NN
4-variable	TP	3.960	3.800	4.260	4.040	3.980	3.380	4.140	3.850
	FN	1.260	1.420	0.960	1.180	1.240	1.840	1.080	1.370
	FP	0.480	0.350	0.690	0.760	0.220	0.530	0.300	0.770
	TN	6.300	6.430	6.090	6.020	6.560	6.250	6.480	6.010
	Accuracy	0.855	0.853	0.863	0.838	0.878	0.803	0.885	0.822
	Balanced Accuracy	0.847	0.841	0.858	0.831	0.865	0.787	0.878	0.816
	F1	0.812	0.801	0.823	0.795	0.837	0.733	0.847	0.769
	Sensitivity	0.762	0.732	0.813	0.772	0.762	0.650	0.797	0.742
	Specificity	0.932	0.950	0.904	0.891	0.968	0.924	0.958	0.890
	Pos Pred Value	0.915	0.927	0.873	0.855	0.951	0.878	0.940	0.849
	Neg Pred Value	0.836	0.820	0.871	0.844	0.846	0.776	0.860	0.824
	Precision	0.915	0.927	0.873	0.855	0.951	0.878	0.940	0.849
	Recall	0.762	0.732	0.813	0.772	0.762	0.650	0.797	0.742
6-variable	TP	3.880	3.940	3.990	4.100	4.050	3.430	4.140	3.990
	FN	1.340	1.280	1.230	1.120	1.170	1.790	1.080	1.230
	FP	0.560	0.400	0.450	0.890	0.370	0.490	0.990	0.380
	TN	6.220	6.380	6.330	5.890	6.410	6.290	5.790	6.400
	Accuracy	0.842	0.860	0.860	0.833	0.872	0.810	0.828	0.866
	Balanced Accuracy	0.834	0.852	0.850	0.830	0.863	0.795	0.828	0.853
	F1	0.795	0.814	0.814	0.789	0.828	0.737	0.790	0.818
	Sensitivity	0.748	0.761	0.764	0.784	0.779	0.660	0.797	0.762
	Specificity	0.919	0.942	0.936	0.876	0.947	0.931	0.860	0.944
	Pos Pred Value	0.893	0.923	0.908	0.829	0.922	0.892	0.818	0.927
	Neg Pred Value	0.825	0.837	0.840	0.846	0.852	0.781	0.850	0.843
	Precision	0.893	0.923	0.908	0.829	0.922	0.892	0.818	0.927
	Recall	0.748	0.761	0.764	0.784	0.779	0.660	0.797	0.762

This table shows the performance of the Statistical, Machine Learning and Neural Network methods in the test set of the model runs.

6.2. Identifying key drivers of risk and default

Traditional statistical methods, such as linear regression and logistic regression, remain highly valuable tools in identifying key drivers of risk and default due to their inherent simplicity and interpretability. Linear regression provides a straightforward approach to modeling relationships between a continuous dependent variable and one or more independent variables, making it easy to quantify the impact of each predictor on the outcome. This clarity allows analysts to directly assess how changes in input features influence predicted values, facilitating an intuitive understanding of risk factors. Logistic regression extends this utility to binary outcomes, such as "defaulted" or "not defaulted" scenarios, by modeling the probability of occurrence for each class. It offers interpretable coefficients that can be transformed into odds ratios, providing clear insights into the relative significance and direction of different predictors. The transparency of these models not only aids in communicating findings to stakeholders, but also supports regulatory compliance and enhances trust in financial decision-making processes.

As each of the two statistical methods (logistic regression, linear regression) is run one hundred times in order for cross-validation to be performed, alongside the average coefficient estimates, an important summarizing metric is the explanatory variable's significance ratio, which highlights the proportion of model runs in which the individual regressor was significant. The results are, therefore, representative of a total of 400 models that were estimated across the two foundational methodologies and two model specifications.

Estimation results show that the anticipated directions of relationships between the regressors and dependent variable of the model were largely shown to hold. In particular, for both specifications, NPLs positively impact the probability of default, as do the CIR and the LDR. Similarly, in the specification including loan growth, the indicator produces a positive impact on the PD. The dynamic ROE ratio, as expected, produces a negative impact on the probability of default, higher returns indicating lower risk of failure. Finally, while the CAR is expected to similarly have a negative impact on the PD, in both the linear and logistic model specifications, it turns out to be insignificant. Indeed, for both methodologies, the feature importance is concentrated around one or two dominant factors, with the other variables having limited impact on the estimation outcome.

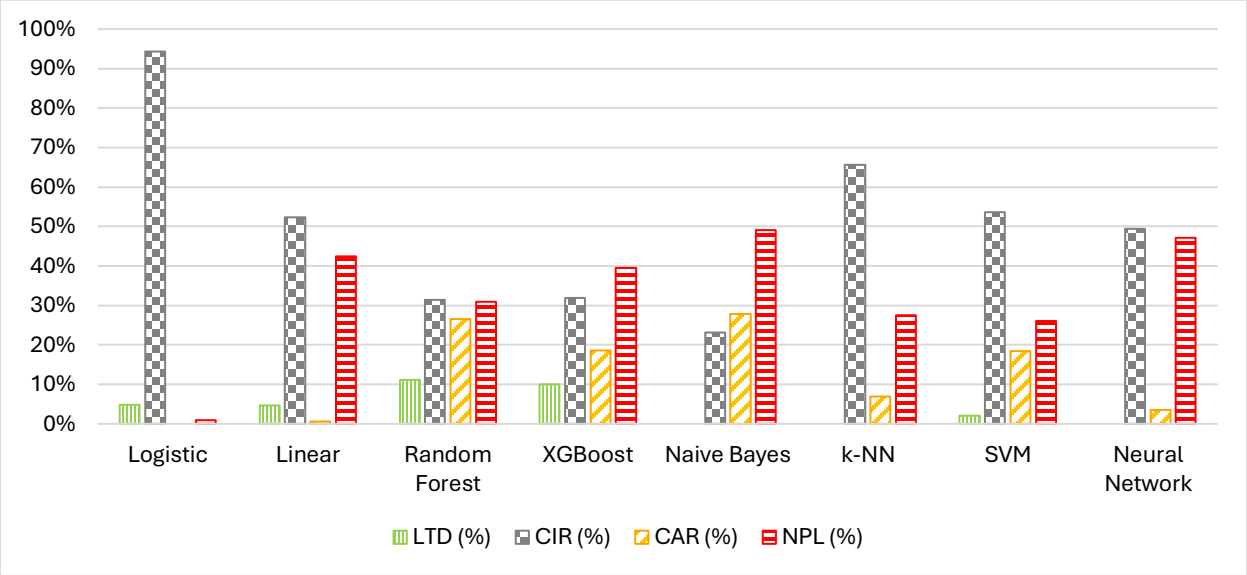
When moving away from statistical methods, automated feature importance calculation varies significantly by algorithm. Random Forests provide built-in functionality to calculate mean decrease in impurity (Gini or entropy) or mean decrease in accuracy. XGBoost offers similar capabilities to assess how much each feature contributes to reducing loss. Both methods provide relatively straightforward interpretations – higher values generally indicate greater importance. For the other methods utilized in this paper, assessing feature importance is less direct and for the purposes of this analysis is derived using permutation importance.

Feature importance scores produced by ensemble methods tend to distribute importance across all variables more evenly. This is particularly the case for the Random Forest model. Indeed, because these methods are capable of capturing non-linearities in the data, impact from factors which were linearly uncorrelated with the dependent variable can now be captured and integrated into the estimated models.

Naïve Bayes and SVM find a dominant factor and several auxiliary contributors, focusing more on the key contributors and filtering out variables with very low contributions. K-NN manifests similar tendencies, but produces a larger interval between features of decreasing importance. Finally, while Neural Networks also manifest selection-type behavior, the emphasis that they place on the key contributing features is more egalitarian than Naïve Bayes or SVM, resembling more closely the behavior of the Random Forest method.

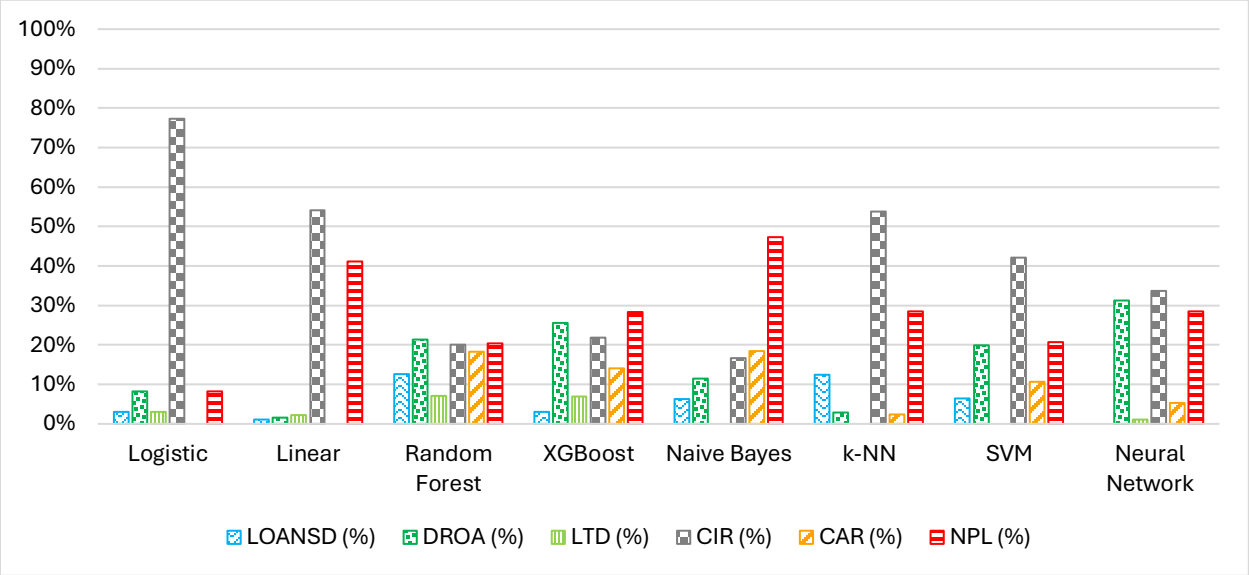
Just as in the case of the statistical methods, NPLs and the CIR are key contributing factors. However, in the Machine Learning models, the CAR also turns out as a key driver of bank defaults, in some cases even surpassing the importance of the CIR. Six-variable implementations of ensemble methods also find the DROA to be a significant predictor. A similar finding can also be observed with the SVM and Neural Network.

Figure 10. Impact of risk factors on the probability of default in the 4-variable specification



This figure summarizes the feature importance scores for the models estimated using the 4-variable specification. It highlights the importance of NPLs, the CIR and the CAR in explaining the probability of default of financial institutions.

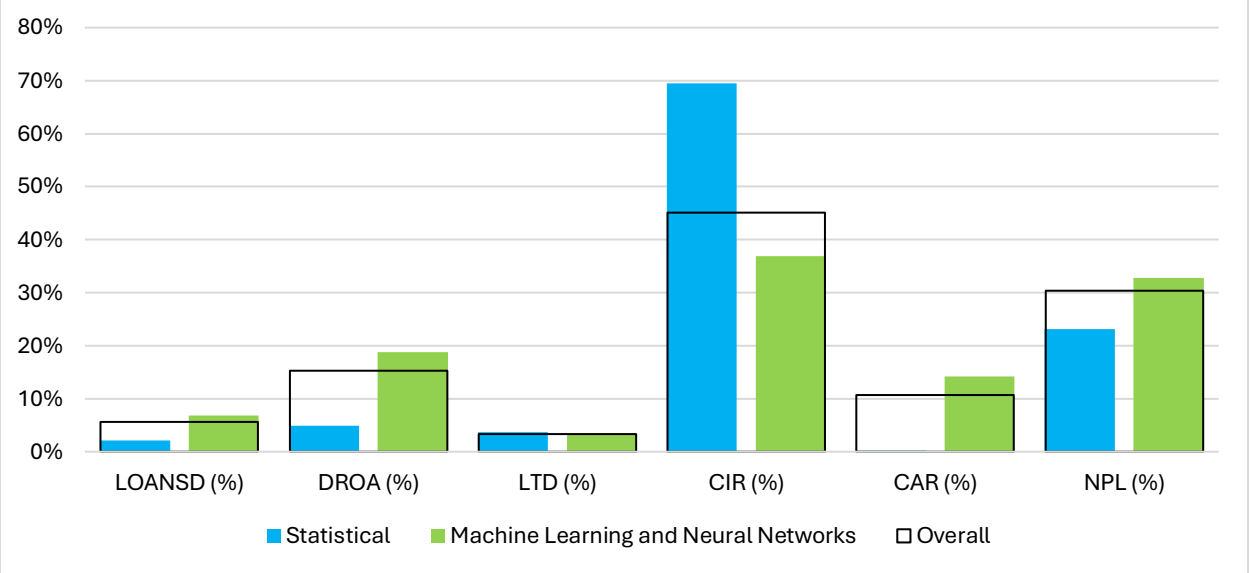
Figure 11. Impact of risk factors on the probability of default in the 6-variable specification



This figure summarizes the feature importance scores for the models estimated using the 6-variable specification. It highlights the importance of NPLs, the CIR, the CAR and the DROA in explaining the probability of default of financial institutions.

The results of the feature importance scores for individual methods can be summarized into average results for statistical methods, Machine Learning and Neural Network implementations, and compared with the overall importance across all methodologies and specifications.

Figure 12. Feature importance comparison, by type of method



This figure summarizes the feature importance scores for the models across all specifications, grouped by tupe of model: "Statistical" or "Machine Learning and Neural Networks". It underscores the high degree of

overall importance attributed to the CIR and NPL ratios, the medium level of importance attributed to the DROA and CAR, as well as the low overall importance of LOANSD and LTD.

Summarizing the results, we observe that the cost-to-income ratio is the leading factor capable of explaining the default of financial institutions in every model specification. NPLs also play a determining role with very high significance ratios in both statistical and ML methods. With a lower, but still noteworthy significance ratio, the Dynamic ROA proves to be important in the estimation of ML models in particular, but also in statistical methods to a certain degree. The Capital Adequacy Ratio is found to be important by ML methods, but is completely missed by the statistical counterparts. Finally, on the lower end of the importance spectrum, loan growth appears to have slight importance in the ML methods, but is less relevant for the linear and logistic regressions, while the LTD ratio achieves equally low importance across the board.

6.3. The benefits of using LLMs in default classification

While Large Language Models are fundamentally built upon complex Neural Network architectures, they offer certain advantages in terms of interpretability due to their capacity for generating explanatory text alongside their outputs. Unlike many traditional Machine Learning models, which might provide predictions without context, LLMs can be prompted to articulate the reasoning behind their conclusions in human-readable form. This ability to generate explanations allows users to understand not just what an LLM predicts or suggests but also how it arrived at that decision based on its training data and internal processes. By leveraging natural language generation capabilities, LLMs can break down complex patterns into understandable narratives, offering insights into the factors influencing their outputs. Moreover, this transparency in reasoning supports user trust and facilitates more informed decision-making, especially in applications such as finance, where understanding the rationale behind predictions is required for accountability and compliance. While challenges remain in fully interpreting neural network decisions, the ability of LLMs to contextualize and rationalize results represents a significant step forward in bridging the gap between advanced AI capabilities and human interpretability.

In practice, the Large Language Model proved that it could outperform all other models in the vast majority of performance metrics. In terms of accuracy, it achieved a score of 93.8%, which was 5.3 percentage points higher than the next best model, the 4-variable SVM. Even more impressive gains were obtained in the F1 score, which improved by 7.9 percentage points versus the same second best contender and constituted 92.6%, as well as the model's recall score which was 11.3 percentage points higher than the 4-variable Random Forest model, which was next in line. Finally, some of the highest gains were achieved in the model's False Negative ratio which was 60.9% lower than the next best 4-variable Random Forest model.

Since all of the financial institutions were run on the LLM without provision of any information about their default or non-default status, the results of the LLM metrics were compared with the test set of other models.

Table 6. LLM Performance compared to other models

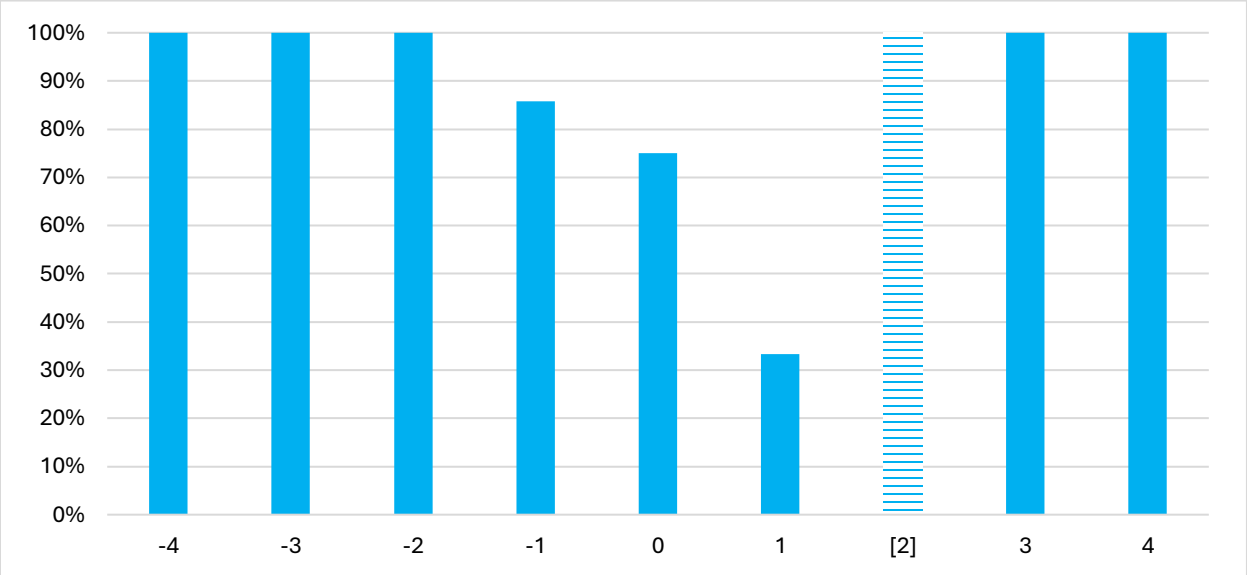
Set	Metric	LLM	Best model
Test	TP	25.000	LLM
	FN	2.000	LLM

FP	2.000	Naïve Bayes (4 var)
TN	35.000	LLM
Accuracy	0.938	LLM
Balanced Accuracy	0.936	LLM
F1	0.926	LLM
Sensitivity	0.926	LLM
Specificity	0.946	Naïve Bayes (4 var)
Pos Pred Value	0.926	Naïve Bayes (4 var)
Neg Pred Value	0.946	LLM
Precision	0.926	Naïve Bayes (4 var)
Recall	0.926	LLM

This table highlights the leading performance that the LLM was able to achieve across a wide range of performance metrics. In particular, its high accuracy, F1 and recall scores stood out among the competition.

To get a better understanding of the efficiency of the scoring mechanism for bank default classification, it is useful to view the accuracy metric broken down by individual score values. This gives us an idea of the degree of correlation between the model's confidence in the produced result and the actual realizations that occurred.

Figure 13. Correct classifications, by score



This figure shows the share of correctly classified observations for each score value. Scores -1 to 1 have a lower accuracy, whereas other score values manifest complete accuracy. No observations obtained a score of 2.

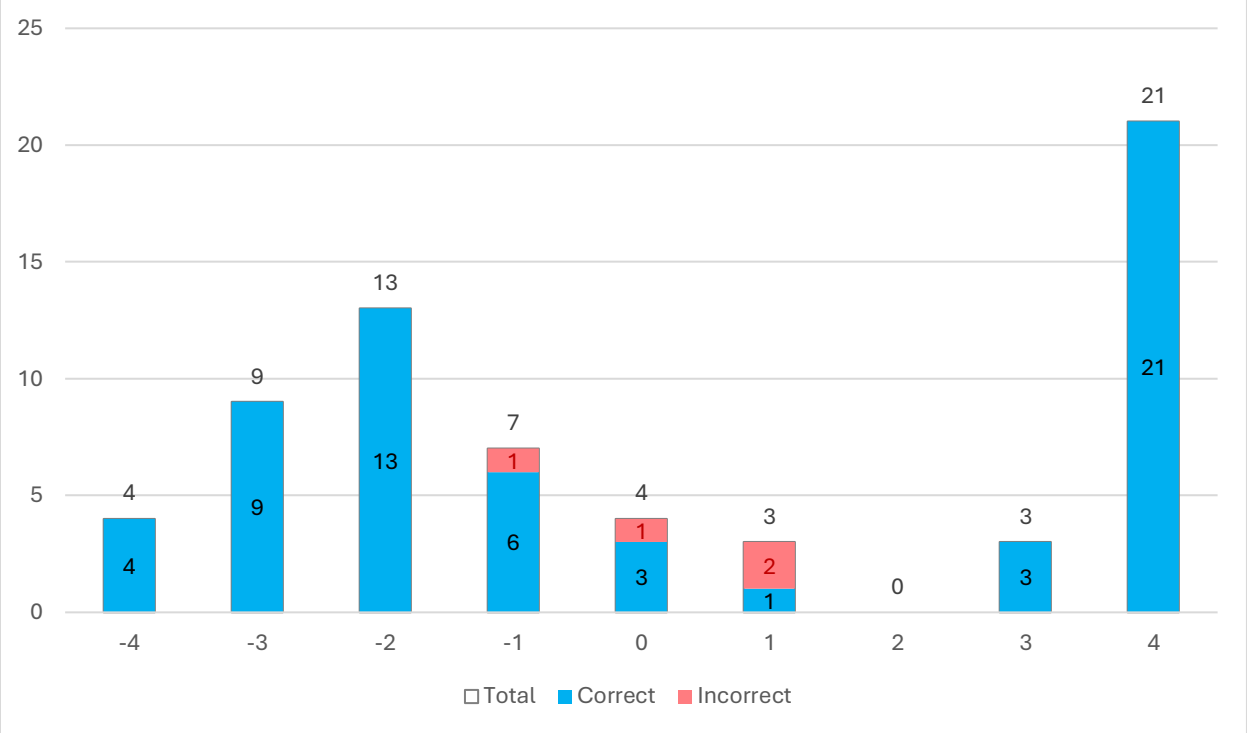
Higher scores are linked to more precise realizations, such that observations with scores of -4, -3, -2, 2 and 4 are always correctly classified. Some institutions which obtained frontier scores ranging between -1 and

1 were classified incorrectly. The incorrect classifications from the -1 to 1 score range reflect the weaker confidence of the LLM in its analysis.

It is also worthwhile to contextualize the disaggregated accuracy metrics with the number of observations that were assigned each score value. Looking at the number of observations for each score level, frontier values ranging between -1 and 1 represent a smaller share of the total number of observations. For the going concerns, the vast majority of institutions were classified with full conviction due to their good health and abundant mitigating factors. On the defaulted side, the confidence is less pronounced. This is because of the wide variety of different conjectures that were at the root of the default event and the varying degree of mitigating factors that were present. It is indeed the case that only some entities experienced failures within perfect storm conditions.

Overall, the results point towards the efficiency of the confidence scoring mechanism that was developed, as the model is able to produce well-informed results with little bias or hallucination and a correct assessment of the confidence level in the generated scores.

Figure 14. Correctly and incorrectly classified observations, by score



This figure displays the correctly and incorrectly classified observations, by score. The few observations that were classified incorrectly reside within the frontier score interval of -1 to 1.

7. Robustness checks

7.1. Assessing the sensitivity of traditional models to changes in configuration

The reliability of the findings presented in this paper was assessed through a series of robustness checks along three key dimensions. First, sensitivity to the train-test split was evaluated by comparing results obtained with 80/20, 75/25, and 85/15 splits. This determined whether performance varied significantly depending on the proportion of data allocated to training versus testing.

Second, each method's random train-test split was repeated 100 times throughout the analysis to account for inherent randomness in data partitioning. To further validate result stability, this number of repetitions was increased to 1000 and resulting distributions of performance metrics were compared. This allows assessment of whether observed differences between methods are consistent with a substantially larger sample of random splits.

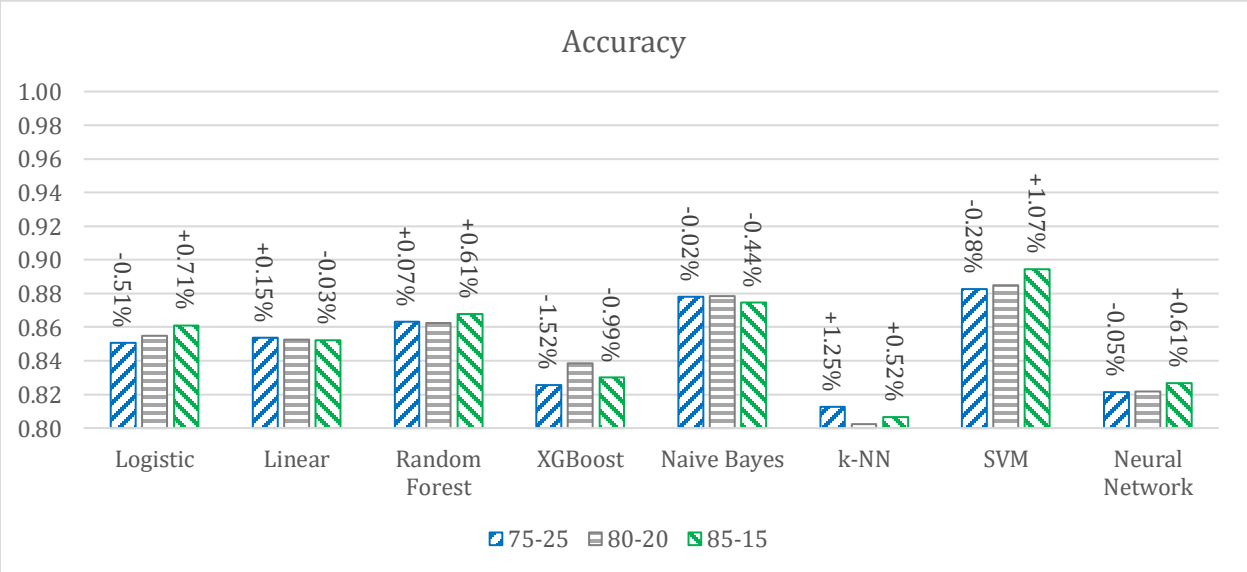
Finally, outlier handling involved capping certain variables in the primary analysis. The robustness of these decisions was examined by performing additional analyses. Specifically, the existing cap value was doubled and, separately, the capping threshold was removed entirely, allowing for unrestricted variable values. Comparing results under these alternative scenarios determines if conclusions are sensitive to the specific outlier handling employed.

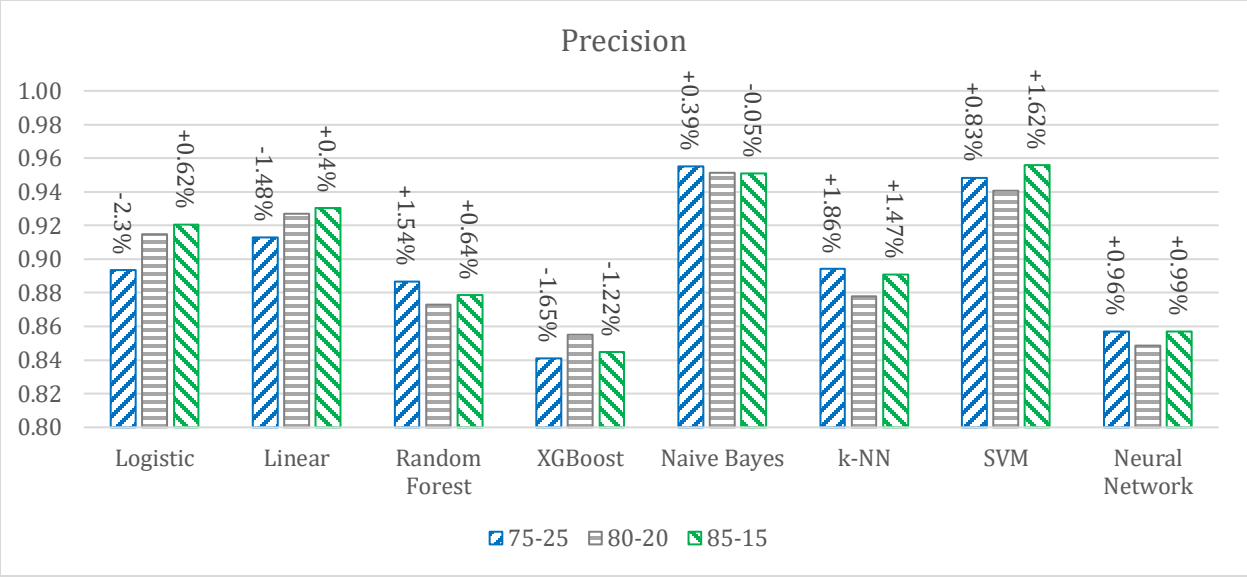
The sensitivity of the model to changes in specification is assessed by means of comparing the accuracy, precision and recall metrics of the model across method-specification-configuration pairs.

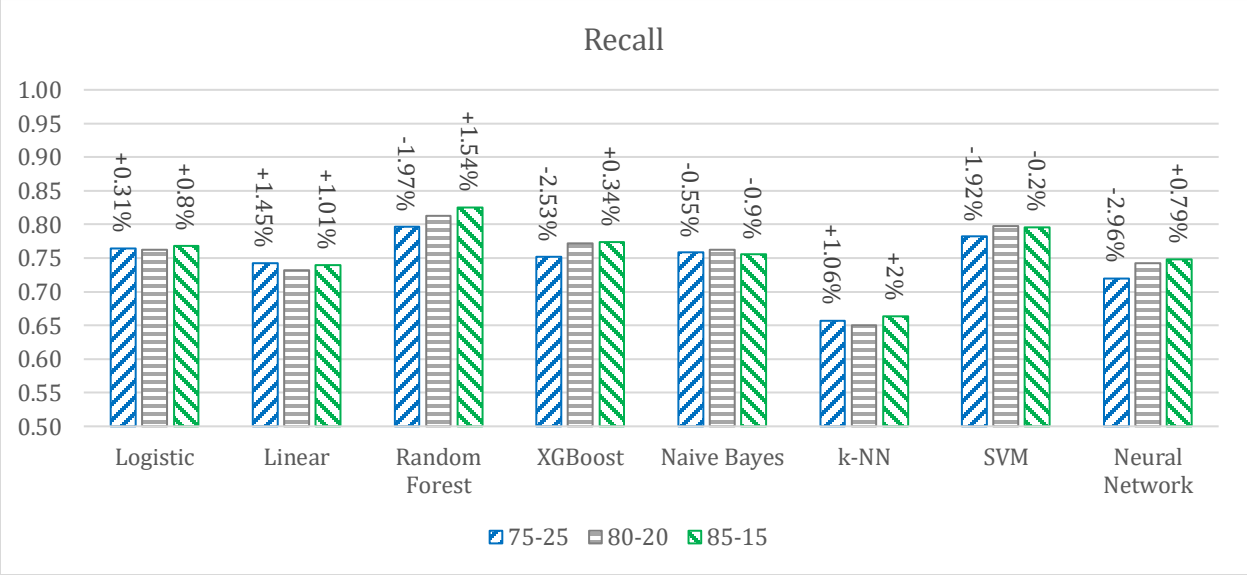
7.1.1. Train-test split

The train-test split associated with each model run was changed to 75/25 and, separately, 85/15 from the original 80/20 used by default in the paper. This exercise was performed for both model specifications – 4-variable and 6-variable.

Figure 15. Sensitivity of results to train-test split proportion modifications, 4-variable specification







The figure depicts performance metrics for various train-test splits in the 4-variable model specifications.

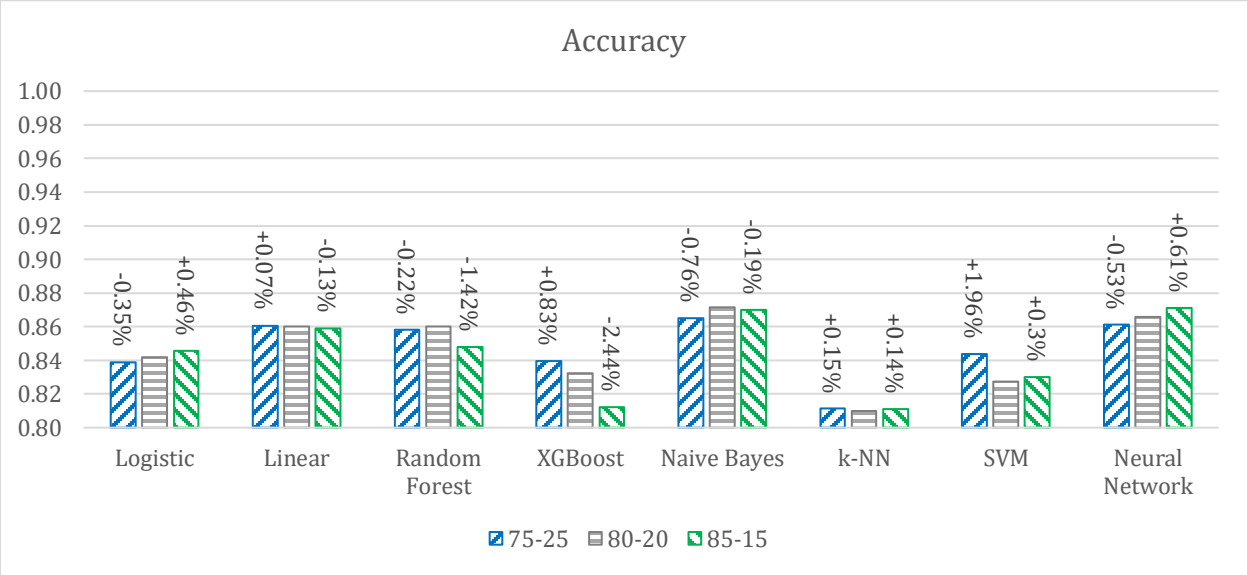
The 4-variable specification is robust to changes in the train-test split. Increasing the training set to 85%, which consequently reduces the test set to 15% of the sample, yields an increase in accuracy for 5 of the methods. Conversely, decreasing the training set to 75% of the sample and therefore increasing the test set to 25%, decreases the accuracy of 5 models. XGBoost and the Naïve Bayes method produce their highest score in the default 80-20 configuration, whereas the Random Forest and k-NN methods stand to gain in terms of accuracy either from increasing or decreasing the training set. The highest attained increase in accuracy is 1.25% and the largest decrease – 1.52%. On average, changing the train-test split configuration produces an increase in accuracy of 0.07%.

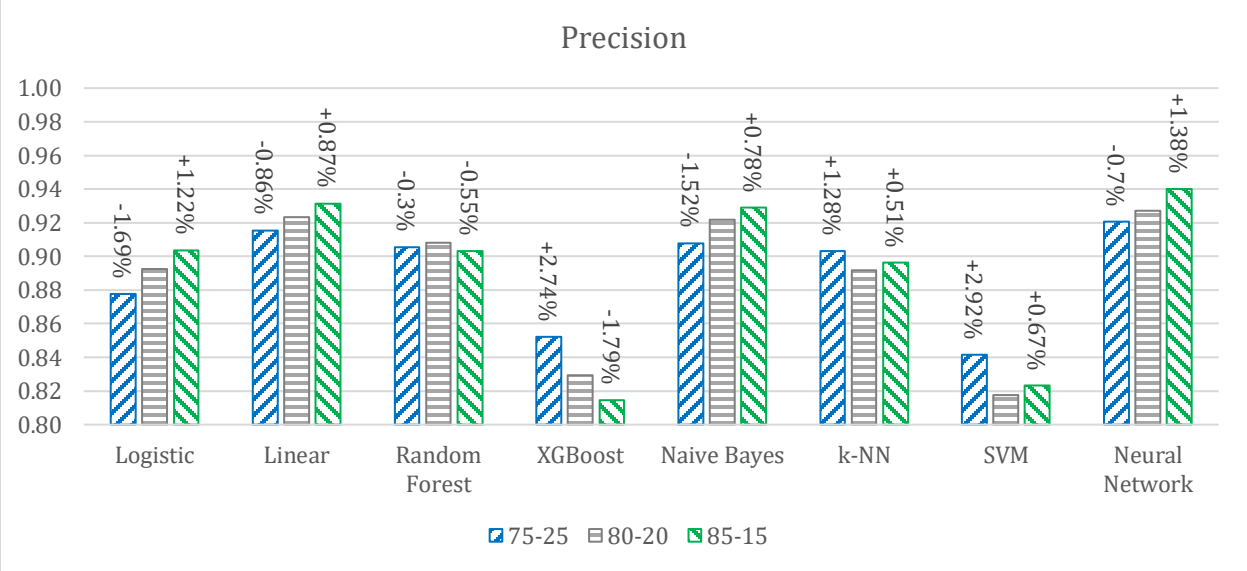
Six models improve their precision when the training set is increased to 85% of the sample. Three methods experience a decrease in precision when the training set is reduced to 75%. The XGBoost model performs

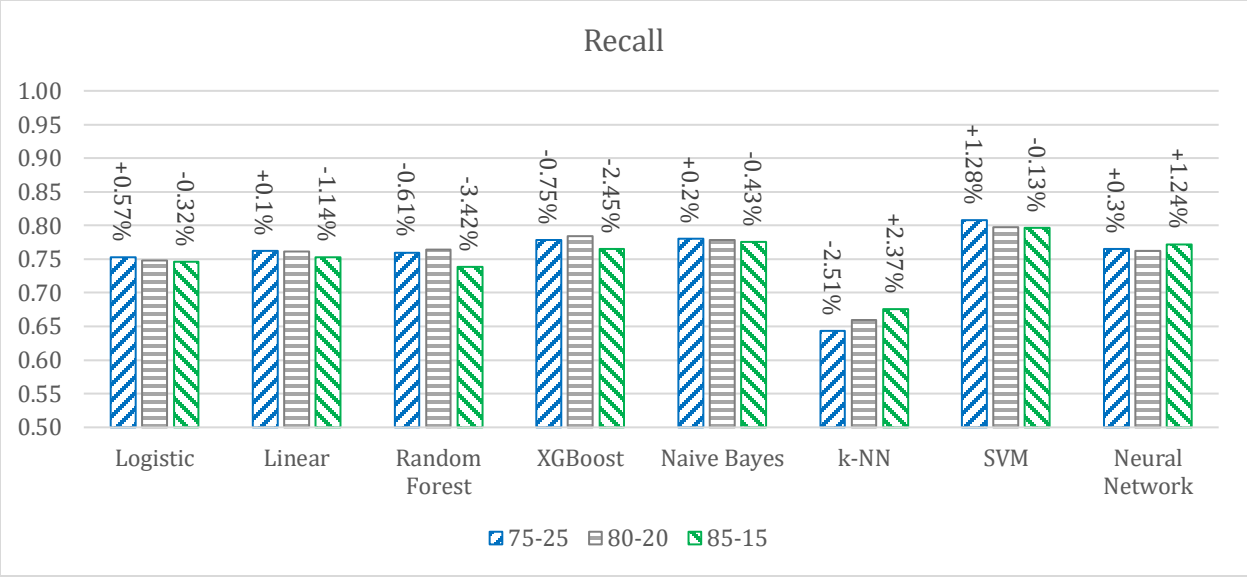
best on the 80-20 configuration, whereas four other methods gain from switching to alternative configurations. Changes in train-test split size can increase precision by as much as 1.86%, but the performance indicator can also drop by up to 2.3%. On average, performance increases by +0.29%.

As for the recall, its performance improves for six of the methods when the training set is increased to 85%. Five of the methods have a lower recall score when the training set is decreased to 75%. Under the default configuration, peak performance is achieved for Naïve Bayes and SVM, but three other models underperform: linear, logistic and k-NN. Recall improves by a maximum of 2%, but its largest drop constitutes 2.96%. On average, train-test split proportion variations produce a change in recall of –0.11%.

Figure 16. Sensitivity of results to train-test split proportion modifications, 6-variable specification







The figure depicts performance metrics for various train-test splits in the 6-variable model specifications.

With the 6-variable specification, in the case of four methods, increasing the training set to 85% of the sample yields an increase in accuracy. In the case of four of the methods, lowering the training set to 75% of the sample leads to a decrease in performance. For the SVM and k-NN methods, both increasing and decreasing the training set produces improvements in accuracy. The largest increase in accuracy is of 1.96% and the sharpest decrease – 2.44%. On average, the change from switching train-test split configurations constitutes -0.09%.

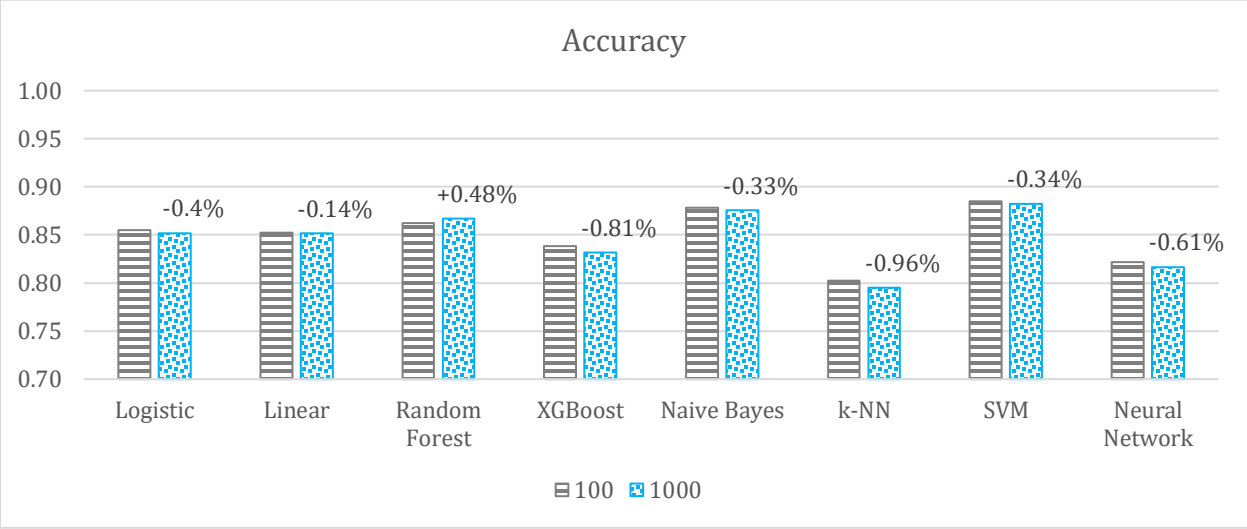
The precision of six methods improves when increasing the training set to cover 85% of the sample. In the case of five methods, the indicator drops, when the training set is reduced. Peak performance with the default configuration Is achieved for the Random Forest model and randomness caused a disadvantage for the k-NN and SVM methods under the baseline configuration. When altering the train-test split size, performance increases at most by 2.92% and decreases by 1.79%. On average, performance changes by +0.31%.

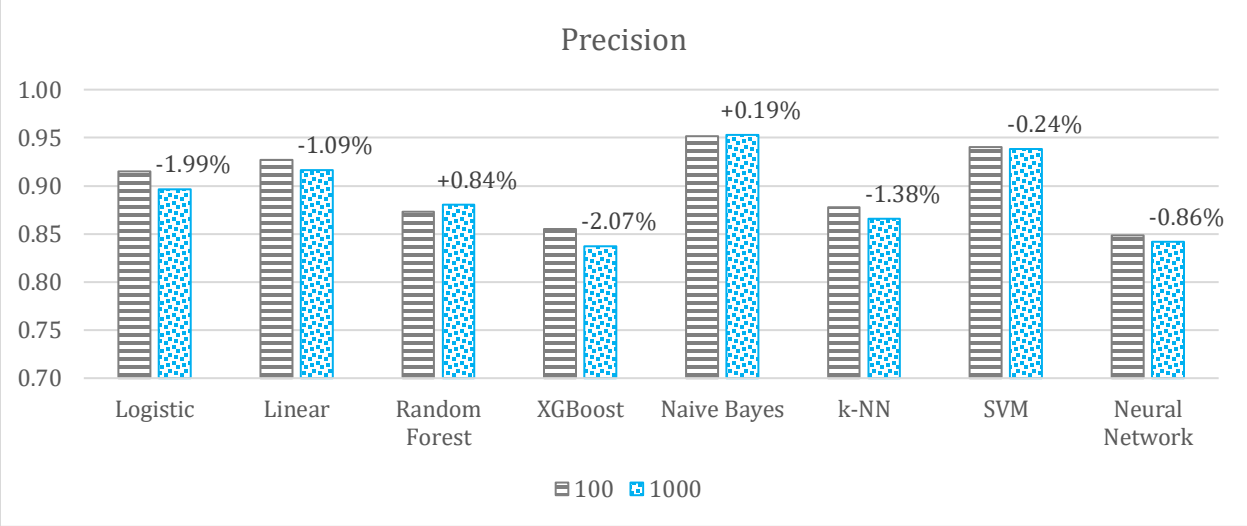
When expanding the training set, the recall of two models improves. Conversely, contracting the training set lowers the recall for three models. Under the default configuration, peak performance is achieved for the Random Forest and XGBoost models, but the Neural Network underperforms. The maximum increase in recall is 2.37% and the largest decrease is 3.42%. On average, a modification in train-test split configurations produces a change in recall of -0.36%.

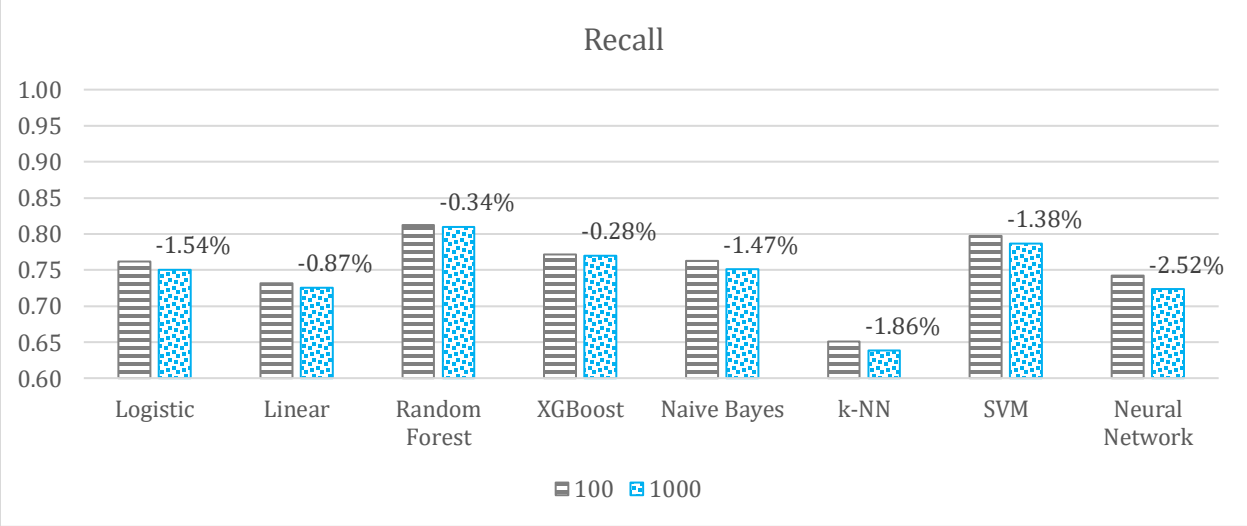
7.1.2. Number of repetitions

For both 4 and 6-variable specifications, the number of random train-test splits was increased by a factor of 10, from 100 to 1000 repetitions. This allows for an investigation of the sensitivity of the results to train-test split configurations. The higher number of repetitions provides a more robust estimate of performance, reducing the impact of any combination of, potentially favorable or unfavorable, splits.

Figure 17. Sensitivity of results to changes in sampling, 4-variable specification







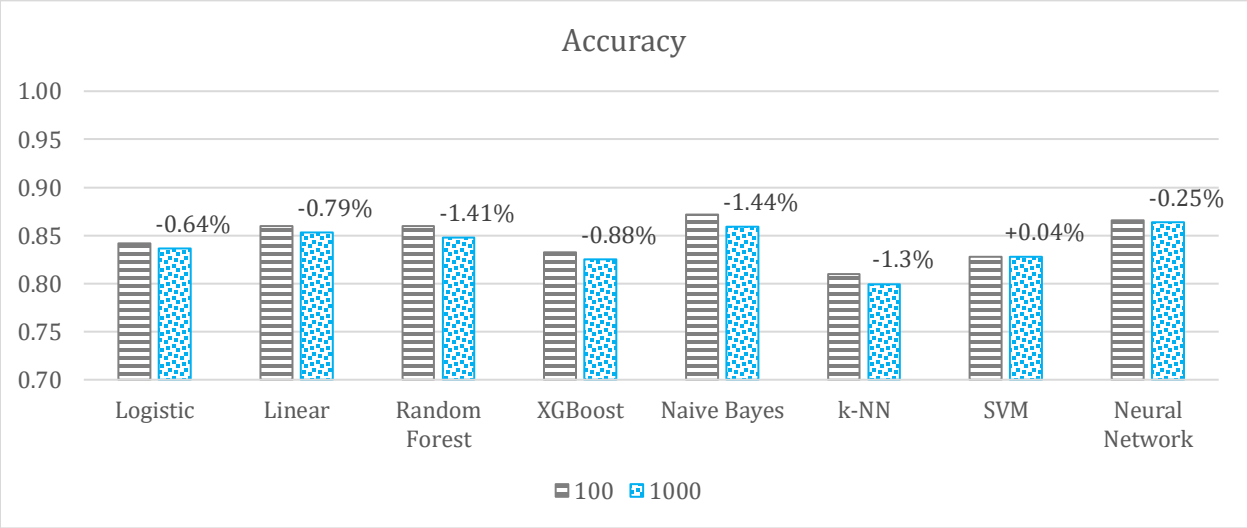
The figure depicts performance metrics for 100 vs. 1000 repetitions in the 4-variable model specifications.

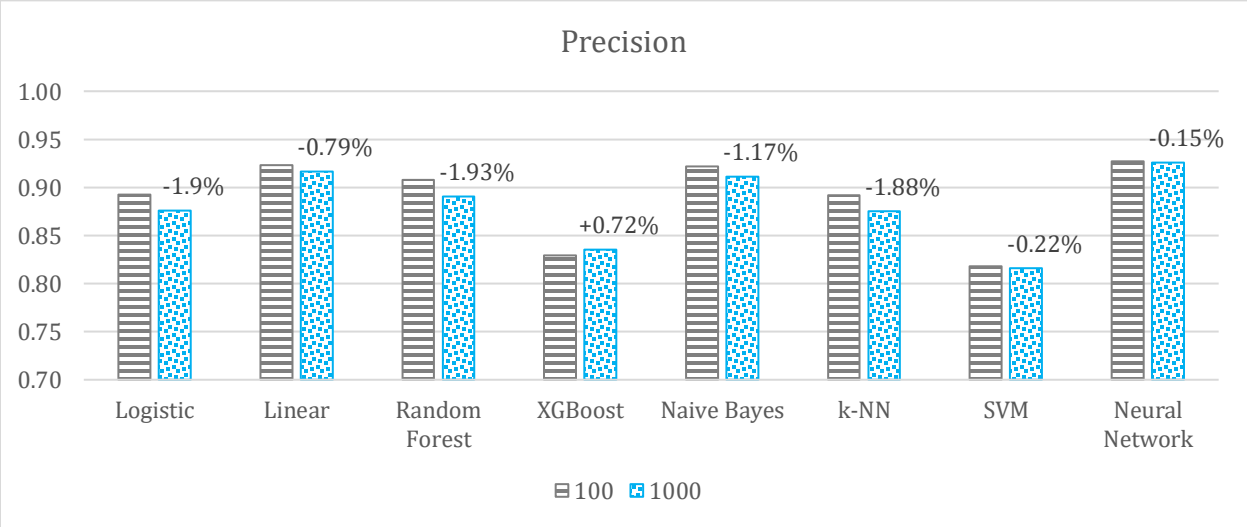
The impact of increasing the number of repetitions is limited and does not significantly impact the results of the analysis. In particular, switching to 1000 repetitions produces an average decrease in accuracy of 0.39%. The greatest loss in accuracy is of 0.96% in the case of the k-NN model. On the other hand, the Random Forest model stands to gain from increased sampling as its accuracy improves by 0.48%.

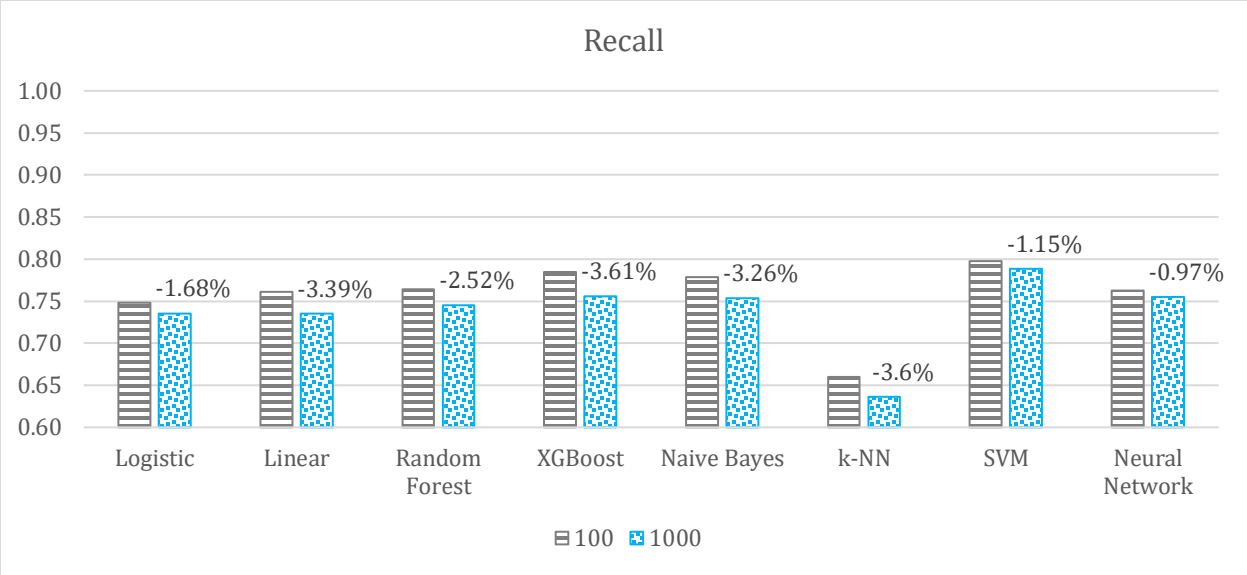
Precision drops by 0.82% on average with increased sampling, but its effect is more disparate. XGBoost and the logistic regression model suffer the most with a decline of around 2%, while the Random Forest and Naïve Bayes approaches improve their accuracy by up to 0.84%.

Recall suffers the most with an average decline of 1.28%. The Neural Network and the k-NN models are most impacted with drops of up to 2.52%, while the XGBoost and Random Forest models experience only minor decreases of around 0.3%.

Figure 18. Sensitivity of results to changes in sampling, 6-variable specification







The figure depicts performance metrics for 100 vs. 1000 repetitions in the 6-variable model specifications.

The 6-variable specifications are affected more than the 4-variable counterparts, however the impact remains limited. In terms of accuracy, the average impact is of -0.83%, with the Naïve Bayes method being the most affected with a decline of 1.44%. On the other hand, the SVM is unaffected by the number of repetitions that are performed.

The average decrease in precision constitutes 0.92%, which is on par with the level observed for the 4-variable specifications. The Random Forest model experiences the highest decline of 1.93%. On the other side of the spectrum, the XGBoost model improves its precision by 0.72%.

Finally, the recall is again the most impacted by expanded sampling, with an average decline of 2.52%. The XGBoost and k-NN models are among the most impacted with declines of around 3.6%. Here, the Neural Network suffers the lowest penalty, with a drop of just under one percent.

While sampling is found to produce an impact on the results, it is not consistent across the board, with some methods gaining performance as a result of increasing the number of repetitions. This is true for both the 4 and 6-variable specifications.

7.1.3. Treatment of outliers

Handling outliers is an essential step in data analysis, as they can disproportionately influence statistical results and model performance. Simply removing them can introduce bias and lead to the loss of potentially valuable information when outliers are a genuine, albeit rare, events that must be accounted for. Therefore, it is important to explore different approaches which can reduce their impact without completely discarding the information that they convey to the model. Testing various configurations allows for assessment of the sensitivity of the paper's findings to extreme values and has the potential to improve the balance between the models' key performance metrics such as accuracy, precision and recall. Failure to successfully incorporate outliers can lead to incorrect conclusions or the estimation of models that perform poorly in real-world scenarios where similar extreme values might occur.

The approach to handling outliers adopted in this paper is to cap extreme values at a threshold consistent with maximum levels that one could expect to encounter under typical circumstances. However, to test the model's stability to this approach, as well as the overall necessity of outlier treatment, two checks are performed:

1. Doubling of the threshold past which outlier values are capped
2. No outlier capping

The following rules were applied to model variables throughout the separate runs outlined above:

Table 7. Outlier treatment variations

	Outlier treatment		
Variable	Baseline	Double Threshold	No Threshold
DROA	$DROA \geq -2\%$	$DROA \geq -4\%$	$DROA \in (-\infty;\ +\infty)$
LDR	$LDR \leq 200\%$	$LDR \leq 400\%$	$LDR \in (-\infty;\ +\infty)$
CIR	$CIR = \begin{cases} 1, if CIR_u < 0 \\ CIR_u, if CIR_u \in [0;1] \\ 1, if CIR_u > 1 \end{cases}$	$CIR = \begin{cases} 2, if CIR_u < 0 \\ CIR_u, if CIR_u \in [0;2] \\ 2, if CIR_u > 2 \end{cases}$	$CIR \in (-\infty;\ +\infty)$
CAR	$CAR \geq -25\%$	$CAR \geq -50\%$	$CAR \in (-\infty;\ +\infty)$

This table explains the alternative outlier treatments applied to verify the robustness of the results to outlier values: thresholds are doubled, and subsequently, removed altogether.

Caps on negative values of the DROA and CAR and caps on positive values of the LDR are doubled. The interval of variation, as well as default value for observations residing outside the interval of variation is doubled in the case of the CIR.

Applying the alternative threshold configurations produces changes in the number of observations that are affected by the threshold, as well as the magnitude of the observations newly or originally processed by the threshold. The table below summarizes the changes produced in the dataset.

Table 8. Impact of outlier treatment under alternative threshold configurations

	Outlier treatment (compared to baseline)			
Variable	Double Threshold		No Threshold	
	Δ observations impacted	Times more extreme than baseline (avg.)	Δ observations impacted	Times more extreme than baseline (avg.)
DROA	11	**1.78470**	11	**21.64727**
LDR	5	**1.81140**	5	**5.30849**
CIR	15	**1.69031**	15	**4.43926**
CAR	3	**1.66780**	3	**7.24648**

This table shows the change in the number of observations which received outlier treatment as a result of switching treatment approaches, as well as, on average, how many times more extreme the additionally treated observations were compared to the baseline threshold value set for the respective variable.

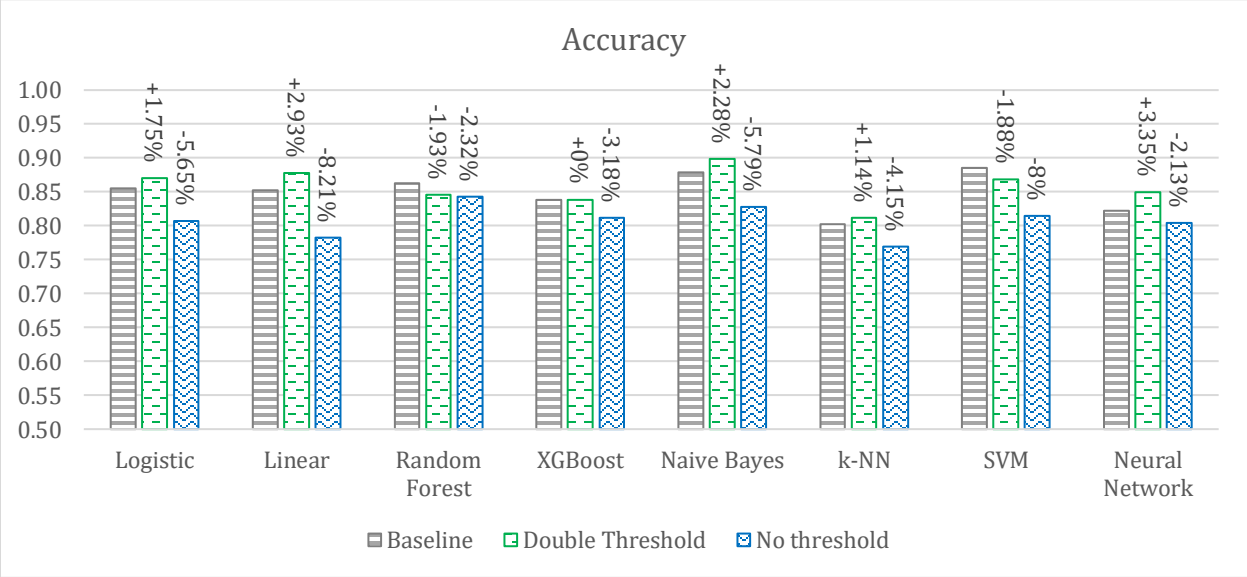
For each variable subject to outlier treatment, changing to any of the two alternative treatment methods produces an equivalent increase in the number of impacted observations. This indicates that doubling the baseline threshold is sufficient to cover all possible combinations of changes in impacted variables and that no additional testing is required in terms of further relaxing the thresholds.

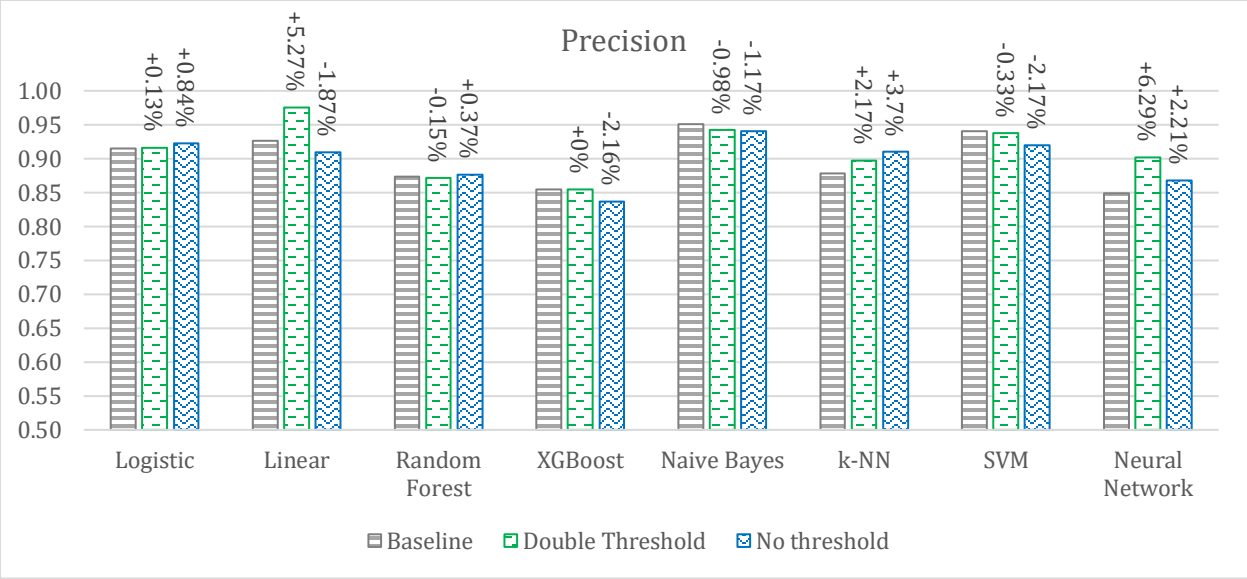
The DROA is most impacted when the threshold is completely released, as unconstrained observations become, on average, almost 22 times more extreme than the baseline treatment threshold. When relaxing the threshold twofold, observations under the relaxed regime are, on average, almost 1.8 times more extreme than the baseline threshold.

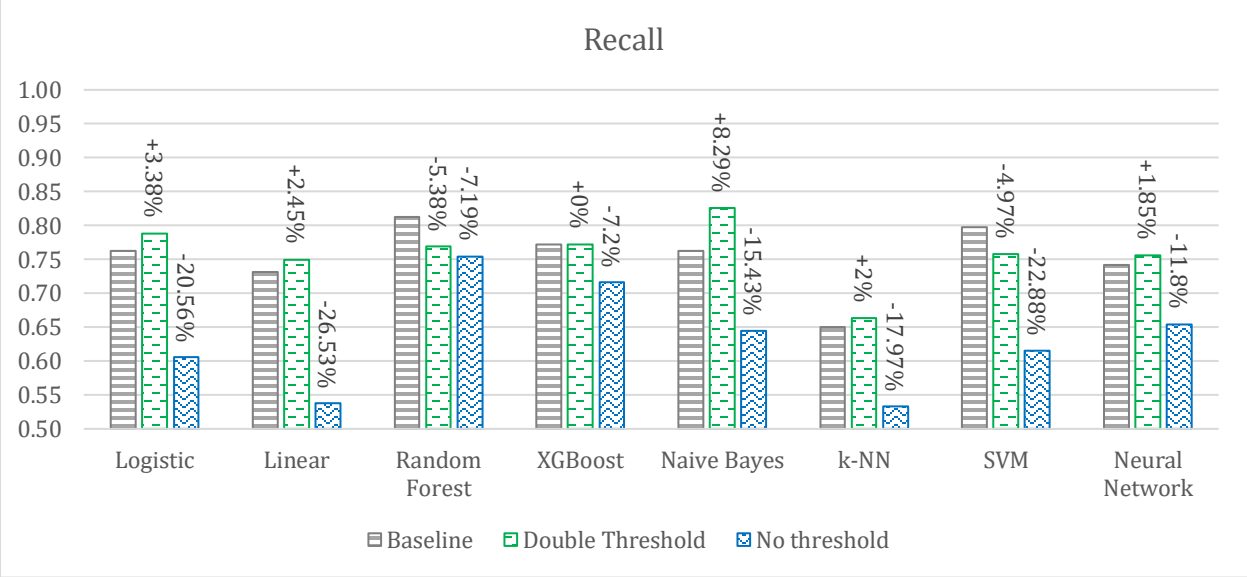
The CAR of outlier observations is also impacted to a high degree when the threshold is removed altogether. In the case of this variable, outlier observations are 7.2 times more extreme than the baseline threshold. Doubling the threshold produces a more moderate effect, with observations under the relaxed threshold being, on average, approximatively 1.7 times more extreme than under the baseline threshold.

The CIR and LDR also experience increases in the magnitude of outlier observations, between 1.7 and 1.8 times when doubling the threshold and between 4.4 and 5.3 times more extreme when completely removing the threshold.

Figure 19. Sensitivity of results to changes in outlier treatment, 4-variable specification







The figure shows performance metrics for outlier treatment configurations in the 4-variable specifications.

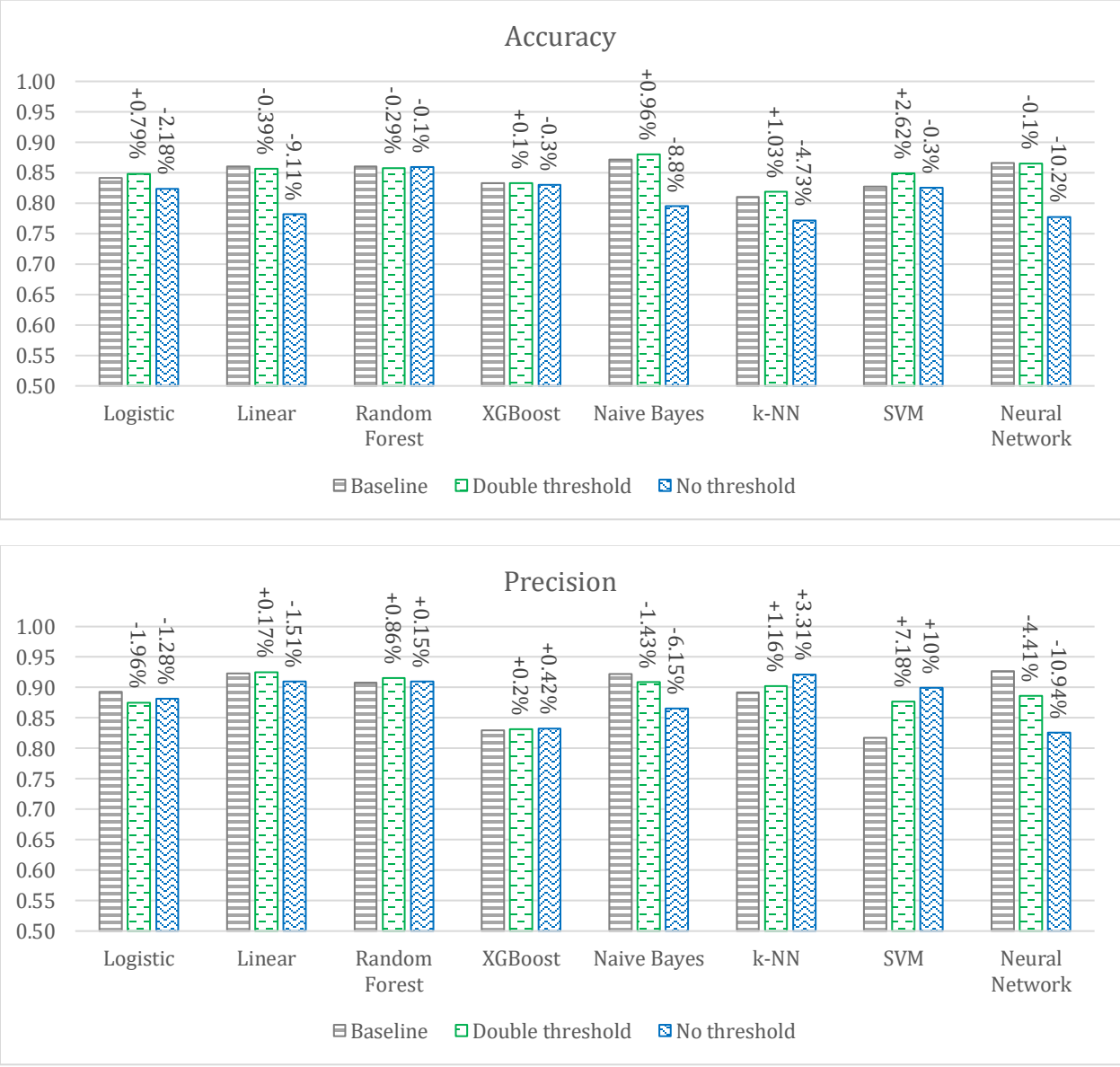
In the 4-variable specifications, doubling the outlier threshold produces, on average, an increase in accuracy of 0.95%. The weak effect is, however, not uniform across all methods. The Random Forest and SVM models, in particular, experience a worsening in performance when the outlier threshold is doubled. Removing the threshold altogether decreases accuracy by almost 5% and the negative impact is consistent across the board, with the Linear model being the hardest hit, experiencing a decline in accuracy of over 8%, while the Neural Network is the least affected with a drop of just over 2%.

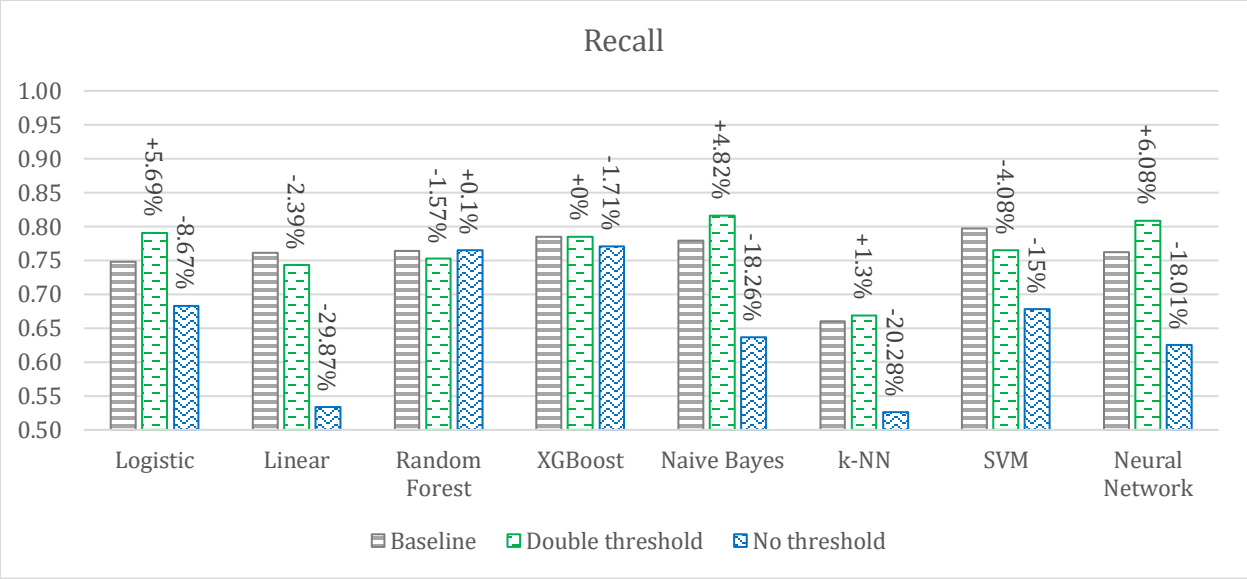
Precision is largely unaffected by changes in outlier thresholds. Doubling the threshold increases precision, on average, by around 1.5%, while removing the threshold produces an impact of -0.03%, which is very close to null. The Random Forest, Naïve Bayes and SVM models all lose performance when the threshold is doubled, while the XGBoost method is completely unaffected under this outlier treatment configuration.

Removing the threshold causes half of the methods to improve their precision scores and the other half to experience declining performance.

Recall is most affected by the removal of the threshold, as it lowers recall, on average, by 16.19%, with the Linear Regression, Logistic Regression and SVM models experiencing declines of over 20% in recall. None of the methods , however, achieve improvements in recall under this conjuncture. The Random Forest and XGBoost methods are the least affected with drops of only 7%. When the threshold is simply doubled, the impact on recall is actually positive and constitutes, on average, 0.95%. The Naïve Bayes method stands to gain the most with an increase in recall of 8%, while the Random Forest turns out to be at a disadvantage with a decrease of over 5%.

Figure 20. Sensitivity of results to changes in outlier treatment, 6-variable specification





The figure shows performance metrics for outlier treatment configurations in the 4-variable specifications.

The overall impact of outlier treatment modification is weaker on the 6-variable specification, compared to its 4-variable counterpart.

Accuracy declines on average by 4.46% when no outlier treatment is performed, compared to the baseline outlier treatment. Relaxing the threshold twofold, on the other hand, causes a slight increase of 0.59%, on average. The Linear Regression, Random Forest and Neural Network experience declines in accuracy regardless of which alternative outlier treatment scheme is adopted. All methods lose accuracy when the threshold is removed completely. On the other side of the spectrum, the SVM model experiences the highest gains in accuracy when doubling the threshold.

Precision remains unaffected under the 6-variable specification, as halting the treatment of outliers leads to a decline of 0.75%, while a doubling of the threshold produces an improvement of 0.22% in the indicator. The outright removal of the threshold produces a mixed effect on the retained methods. While the Neural Network experiences a decline of almost 11% in precision, the SVM model gains 10%. The Random Forest and XGBoost methods are least affected with gains of under 0.5%. Doubling the threshold similarly produces a dissimilar effect. Again, the SVM and Neural Network are the most sensitive to changes in outlier treatment method, with the former gaining over 7% and the latter losing around 4.5%. In this configuration, the XGBoost and Random Forest models remain mostly unaffected and are joined by the Linear Regression model, which is the least impacted method.

Recall is again the most impacted performance indicator out of the three. Removing the threshold decreases recall by just under 14%, while a two-fold relaxation increases it by 1.23%, on average. When foregoing outlier treatment, the linear model is the most impacted, with a decline in recall of almost 30%. The Random Forest and XGBoost methods remain resilient, with a decline of 1.7% in the case of XGBoost and a very slight increase in recall of 0.1% for the Random Forest method. The highest impact from doubling the threshold is observed with the Neural Network model, which gains upward of 6% in terms of recall, while the results for the XGBoost method remain unchanged.

Overall, with the exception of the removal of outlier treatment (which one would not expect to be performed under normal circumstances), the largest variations in the retained performance metrics are limited in size and the average variation is negligible. In some cases, performance decreases while in others it can increase. This indicates that while the results are robust to changes in configurations, there is still some scope towards improving the performance of the models by fine-tuning each individual method to its best performing configuration. However, for the purpose of equitably testing each method retained for this paper, under similar conditions, the baseline configurations are appropriate.

7.2. Robustness checks for the LLM

LLM model validation was performed for six banks included in the dataset, of which three were defaulted and another three are active. One of the active banks was taken from the list of incorrectly classified institutions by the base Gemini 2.5 Pro model. The alternative LLM models retained for the validation process, GPT 4.1 and Claude 4.0 Sonnet, were chosen for their prominence at the time of this paper's writing.

In the validation process, the same reports were given to the three different models. Aside from the change in model, all other conditions were kept constant, no external sources were allowed, no agent-based tools were used and no additional prompting was given.

The results of the validation show that despite minor differences in scoring, the default or non-default classification of the banks remains intact, with the exception of one institution whose potential default was discussed in recent times, but which remains active to this date, despite having known deficiencies.

In particular, the results for three banks, two defaulted and one active, were identical across al three models. In the case of one defaulted bank, the two validation models assigned less clear-but scores than the baseline model, but the degree of uncertainty increased by one 1 score point and did not change the overall conclusion about classifying the bank as defaulted. In the case of one active bank, Claude 4.0 Sonnet provided a profitability score two points below the baseline model, while GPT 4.1 agreed with the baseline model's profitability score. On the other hand GPT 4.1 assigned the bank a risk score 1 point below the baseline mode, whereas Claude 4.0 Sonnet agreed with Gemini 2.5 Pro on the risk score. Finally, in the case of the active, but high risk bank, both the baseline model and Claude 4.0 Sonnet assign risk levels of 3 and profitability of 2, leading us to consider the bank as nearing default, while GPT 4.1 inverts the risk and profitability score, correctly classifying the bank as active.

Table 9. Validation of LLM results

Bank	Model	Risk	Profitability
Defaulted bank 1	Gemini 2.5 Pro	5	1
	Claude 4.0 Sonnet	5	1
	GPT 4.1	5	1
Defaulted bank 2	Gemini 2.5 Pro	5	1
	Claude 4.0 Sonnet	5	1
	GPT 4.1	5	1
Defaulted bank 3	Gemini 2.5 Pro	5	1

	Claude 4.0 Sonnet	4	2
	GPT 4.1	4	2
Active bank 1	Gemini 2.5 Pro	2	5
	Claude 4.0 Sonnet	2	3
	GPT 4.1	1	5
Active bank 2	Gemini 2.5 Pro	2	5
	Claude 4.0 Sonnet	2	5
	GPT 4.1	2	5
Active bank 3 *Incorrectly classified by base model, but its risk of default was brought up over the years.*	Gemini 2.5 Pro	3	2
	Claude 4.0 Sonnet	3	2
	GPT 4.1	2	3

This table shows the results of the validation process for the LLM model. Although some changes in scoring exist for 50% of the sample, they are minor and do not change the overall conclusion regarding the classification of the bank as defaulted or non-defaulted. Special attention should be given to institutions which are close to the point of being classified as "in default", as model opinions might diverge.

Based on the validation sample discussed here, one can expect the difference in scores due to changes in the selected LLM to constitute a controlled amount, which does not materially influence the outcome of the analysis, but can help in identifying problem banks. In particular, where there is divergence between the defaulted or non-defaulted opinion across models, increased regulatory attention is warranted.

Table 10. Magnitude of the impact on score results produced by switching LLMs

Type of score change	Magnitude of change	Number of banks
Risk score	– 3 points or more	0%
	– 2 points	0%
	– 1 point	50%
	+ 1 point	0%
	+ 2 points	0%
	+ 3 points or more	0%
Profitability score	– 3 points or more	0%
	– 2 points	16.6%
	– 1 point	0%
	+ 1 point	33.3%
	+ 2 points	0%
	+ 3 points or more	0%
No change in score		50%

This table shows the impact of changing the LLM model on the generated scores. It highlights the small magnitude of change, typically concentrated around +/- 1 point. For half of the banks in the sample, there was no change in score at all, as the model agreed on the outputted assessments. In the table, the scores

do not add up to 100%, because there are multiple validation models considered, and each one of them can produce a different effect on the bank's score, when compared to the baseline.

Another validation step involved re-running the same model on the same document, using the very same configuration, to test the effects of randomness on the generated outcome. This was performed on a set of 10 banks. In the case of two banks the risk or profitability score changed on the second model run, as shown in the table below.

Table 11. Impact of randomness on score results

Type of score change	Magnitude of change	Number of banks
Risk score	– 3 points or more	0%
	– 2 points	0%
	– 1 point	0%
	+ 1 point	10%
	+ 2 points	0%
	+ 3 points or more	0%
Profitability score	– 3 points or more	0%
	– 2 points	0%
	– 1 point	0%
	+ 1 point	10%
	+ 2 points	0%
	+ 3 points or more	0%
No change in score		80%

This table shows the impact of randomness on the generated scores. It highlights the small magnitude of change, concentrated around +/- 1 point. For 80% of the banks in the sample, there was no change in score.

Finally, as a distinct validation exercise, for six banks, in separate runs, the model was also allowed to obtain sources from the internet, after reading the document and before making a score decision. The number of retrieved sources varied between 8 and 15 and constituted:

- bank financial information
- press-releases by the bank
- newspaper articles about the bank's performance
- general purpose information about risk measurement and calculation of risk scores, including:
    - ↳ educational materials and course content
    - ↳ tutorials produced by reputable authors
    - ↳ concept or notion explanations from the websites of financial institutions
- research into modeling risk and monitoring bank performance, including:
    - ↳ research papers
    - ↳ books and other publications
    - ↳ analytics
    - ↳ reports (technical, policy, etc.)

In none of the runs where additional source collection was allowed did the model produce a score different than the one issued by the restricted analysis.

8. Potential limitations

While this study demonstrates the superior performance of a Large Language Model (LLM) – specifically Google's Gemini 2.5 Pro – in predicting bank default compared to traditional statistical and Machine Learning methods, several limitations should be considered when interpreting these findings and applying them in practice.

First, despite achieving leading results with Gemini 2.5 Pro, it is important to acknowledge that the reliance on a specific LLM introduces inherent risks associated with model-specific vulnerabilities and potential obsolescence. The field of Large Language Models is characterized by rapid innovation such that newer models are continually being developed with potentially improved architectures, training data, and capabilities. While results were validated against other prominent LLMs like Claude and ChatGPT – confirming the general trend of superior performance for language-based approaches – these validations do not eliminate the possibility that future iterations or entirely new models could surpass Gemini 2.5 Pro's current capabilities. A continuous benchmarking process across a wider range of contemporary LLMs is essential in maintaining confidence in the chosen methodology and ensuring adaptability to evolving technological advancements.

Second, despite deliberate efforts to construct a diversified dataset encompassing 64 financial institutions – 59 banks and 5 Non-Bank Financial Institutions (NBFIs) – from both developed and developing economies, the sample size remains relatively constrained. This limited data availability poses a challenge for robust model validation and generalization. The complexities of global financial systems necessitate larger datasets to comprehensively capture the full spectrum of risk factors and economic conditions that contribute to bank failure. A more extensive dataset would allow for more rigorous statistical testing and reduce the potential for overfitting to specific market characteristics present within the current sample. Furthermore, while representation from developing economies is included, regional imbalances may exist within this subset, potentially limiting the model's ability to accurately predict defaults in underrepresented regions.

The scope of this study is specifically focused on banks and NBFIs. While these institutions represent a key component of the financial system, they are not exhaustive of all entities susceptible to financial distress. Applying this LLM-based workflow to other types of financial actors – such as insurance companies, investment funds, fintech firms, or specialized lending institutions – would likely require adaptation or retraining. These alternative entities often operate under different regulatory frameworks, possess unique risk profiles, and generate distinct types of financial data. The current workflow, focused exclusively on bank and NBFI reports, may not be directly transferable to these contexts without significant modifications to the input features, model architecture, or training process. Investigating the transferability of this approach to other financial entities represents a central area for future research. This includes exploring whether fine-tuning the LLM with data specific to these alternative institutions would yield satisfactory results, or if entirely new models need to be developed.

A practical limitation encountered during data processing was the input file size restriction imposed by Gemini 2.5 Pro – 500MB total, with sub-limits on upload and page count. While a single report exceeding these limits was addressed through truncation, this process inevitably resulted in the exclusion of potentially valuable information. The impact of such data loss on predictive accuracy remains uncertain and warrants further investigation. Future research should explore techniques for handling large financial reports within LLM constraints, such as intelligent summarization or selective feature extraction prior to input.

Furthermore, while efforts were made to mitigate generative randomness through a dedicated validation stage due to the lack of user-configurable model seeds at the time of analysis, inherent stochasticity within LLM outputs remains a concern. This variability could lead to inconsistencies in predictions across different runs and necessitates careful monitoring and potentially repeated analyses to ensure reliable results.

The study utilized default hyperparameters recommended by the model developers, prioritizing conformity with standard Gemini 2.5 Pro outputs. While this approach ensures reproducibility and comparability, it does not preclude the possibility that alternative hyperparameter configurations could further optimize performance. A systematic exploration of the hyperparameter space through techniques like grid search or Bayesian optimization could potentially yield improved predictive accuracy.

Finally, the current workflow relies heavily on textual data extracted from financial reports. While Gemini 2.5 Pro demonstrates strong capabilities in processing unstructured text, it may not fully capture all relevant information contained within numerical datasets or structured tables. Integrating quantitative data more effectively into the LLM's analysis, perhaps through hybrid modeling approaches that combine LLMs with traditional statistical techniques, could further enhance predictive power and provide a more holistic assessment of financial risk. The reliance on textual reports also assumes consistent reporting standards across institutions, which may not always be the case in practice, particularly when comparing entities from diverse regulatory environments.

9. Conclusion

In this comparative analysis, the performance of some of the most frequently used statistical, Machine Learning and Deep Learning models was evaluated against a Large Language Model across datasets of varying complexity: a 4-variable specification, 6-variable construct, and complete report text. The evaluation was conducted on both training and testing phases to understand the generalization capabilities of each model, with the exception of the LLM where the entire exercise was conducted on the full data sample, taken as a test set.

The performance of the models was compared in terms of accuracy, balanced accuracy, F1 score, recall (sensitivity), specificity, and precision metrics during the test phase. Of particular interest is how the LLM compared with the other models when applied to report text data, representing a more complex and unstructured use case.

The LLM achieved an impressive accuracy of 93.8% on the test set. This performance surpasses that of other models across all considered datasets. For instance, the Random Forest model, known for its robustness in structured data settings, recorded lower accuracy scores of 86.3% and 86% for the 4-variable and 6-variable datasets respectively during testing. The figures suggest a moderate decline in performance of the machine learning model when generalizing to unseen data.

The balanced accuracy score, which provides insight into model performance across potentially imbalanced classes, further underscores the LLM's strength. The LLM reported a balanced accuracy of 93.6% on the unstructured data, significantly higher than the next best performance recorded by the SVM (87.8%) and Naïve Bayes (86.3%) models in structured data settings. In the information-constrained 4-variable setup, the linear regression model showed a balanced accuracy score of 84.1%, positioning it in the mid-range of the performance spectrum and demonstrating its ability to compete with leading Machine Learning methods.

The F1 score, which balances precision and recall, was another area where the LLM excelled. It achieved an F1 score of 92.6% on the report textual data. This reflects a balanced handling of false positives and negatives, compared to the lower metrics attained by other tested models.

Recall, or sensitivity, is particularly important in default classification tasks as it measures the model's ability to identify positive cases correctly. The LLM posted a recall of 92.6%, indicating its capacity to capture most defaults accurately without missing them. The score is significantly higher than the Random Forest which scored recall values of 81.3% and 76.4% for the 4-variable and 6-variable datasets, respectively.

Specificity measures the model's success in correctly identifying negative cases. The LLM recorded a specificity of 94.6% on report text test data, where the linear model achieved slightly higher results of 95% on the 4-variable configuration and Naïve Bayes recorded the highest score of 96.8%.

Precision, which focuses on the proportion of positive identifications that were actually correct, was also high for the LLM which scored 92.6%. This aligns with its high recall to suggest a well-balanced model. Here, the linear regression model was again able to achieve a very slightly higher score of 92.7% on the 4-variable specification, just 0.01 percentage points higher, while Naïve Bayes led the scoreboard with 95.1%.

In summary, across most tested metrics, the LLM demonstrates superior performance relative to other models. Its high accuracy, balanced accuracy, F1 score and recall, highlight its ability to effectively handle unstructured data – a task where traditional statistical and Machine Learning algorithms may falter without extensive preprocessing or feature engineering. While Random Forest and linear regression models are able to achieve great results even when provided with very little data, the LLM's adaptability shines with richer datasets, making it a compelling choice for applications involving complex, textual financial information, given the comprehensiveness of the analysis that it can provide and its greater accuracy.

10. References

[1] Nier, E., & Merrouche, O. (2010). What caused the global financial crisis: evidence on the drivers of financial imbalances 1999: 2007. International Monetary Fund.

[2] Merriam-Webster Dictionary (2025). Online Edition.

[3] Vuillemey, G. (2014). Solvency vs. liquidity. A decomposition of European banks' credit risk over the business cycle. *International Economics*, 137, 32-51.

[4] Coudert, V., & Idier, J. (2018). Reducing model risk in early warning systems for banking crises in the euro area. *International economics*, 156, 98-116.

[5] Beaver, W. H. (1968). Alternative accounting measures as predictors of failure. *The accounting review*, 43(1), 113-122.

[6] Meyer, P. A., & Pifer, H. W. (1970). Prediction of bank failures. *The journal of finance*, 25(4), 853-868.

[7] Barr, R. S., & Siems, T. F. (1997). Bank failure prediction using DEA to measure management quality. In *Interfaces in computer science and operations research: Advances in metaheuristics, optimization, and stochastic modeling technologies* (pp. 341-365). Boston, MA: Springer US.

[8] Martin, D. (1977). Early warning of bank failure: A logit regression approach. *Journal of banking & finance*, 1(3), 249-276.

[9] Ohlson, J. A. (1980). Financial ratios and the probabilistic prediction of bankruptcy. *Journal of accounting research*, 109-131.

[10] Klieštik, T., Kočišová, K., & Mišanková, M. (2015). Logit and probit model used for prediction of financial health of company. *Procedia economics and finance*, 23, 850-855.

[11] Postelnicu, T. (2011). Probit analysis. In *International encyclopedia of statistical science* (pp. 1128-1131). Springer, Berlin, Heidelberg.

[12] Bliss, C. I. (1934). The method of probits. *Science*, 79(2037), 38-39.

[13] Finney, D. J. (1947). Probit analysis; a statistical treatment of the sigmoid response curve.

[14] Zmijewski, M. E. (1984). Methodological issues related to the estimation of financial distress prediction models. *Journal of Accounting research*, 59-82.

[15] van der Ploeg, S. (2010). Bank Default Prediction Models. Erasmus University.

[16] Comelli, F. (2016). Comparing the performance of logit and probit early warning systems for currency crises in emerging market economies. *Journal of Banking and Financial Economics*, 6(2), 5-22.

[17] Hastie, T., Tibshirani, R., & Friedman, J. (2009). The elements of statistical learning: data mining, inference and prediction, 2e. Springer Series in Statistics.

[18] Halabaku, E., & Bytyçi, E. (2024). Overfitting in Machine Learning: A Comparative Analysis of Decision Trees and Random Forests. *Intelligent Automation & Soft Computing*, 39(6).

[19] Chen, S., Härdle, W. K., & Moro, R. A. (2011). Modeling default risk with support vector machines. *Quantitative Finance*, 11(1), 135-154.

[20] Dima, A. M., & Vasilache, S. (2016). Companies default prediction using neural networks. *Romanian Journal of Economic Forecasting*, 19(3), 127.

[21] Benítez, J. M., Castro, J. L., & Requena, I. (1997). Are artificial neural networks black boxes?. *IEEE Transactions on neural networks*, 8(5), 1156-1164.

[22] Oh, S. J., Schiele, B., & Fritz, M. (2019). Towards reverse-engineering black-box neural networks. *Explainable AI: interpreting, explaining and visualizing deep learning*, 121-144.

[23] Brown, I., & Mues, C. (2012). An experimental comparison of classification algorithms for imbalanced credit scoring data sets. *Expert systems with applications*, 39(3), 3446-3453.

[24] Zhou, Y. (2022). Loan default prediction based on machine learning methods. In Proceedings of the 3rd International Conference on Big Data Economy and Information Management, BDEIM (pp. 2-3).

[25] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, 30.

[26] Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019, June). Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologi*es, volume 1 (long and short papers) (pp. 4171-4186).

[27] Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J.D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., & Agarwal, S. (2020). Language models are few-shot learners. *Advances in neural information processing systems*, 33, pp.1877-1901.

[28] Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., & Liu, P.J. (2020). Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140), pp.1-67.

[29] Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., & Schulman, J. (2022). Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35, pp.27730-27744.

[30] United Nations, Department of Economic and Social Affairs, Population Division (2024). World Population Prospects 2024, Online Edition.

[31] Wu, S., Irsoy, O., Lu, S., Dabravolski, V., Dredze, M., Gehrmann, S., Kambadur, P., Rosenberg, D., & Mann, G. (2023). BloombergGPT: A large language model for finance. *arXiv*. arXiv:2303.17564.

[32] Liu, X. Y., Wang, G., Yang, H., & Zha, D. (2023). FinGPT: Democratizing internet-scale data for financial large language models. *arXiv*. arXiv:2307.10485.

[33] Zhuang, H., Qin, Z., Hui, K., Wu, J., Yan, L., Wang, X., & Bendersky, M. (2024). Beyond Yes and No: Improving Zero-Shot LLM Rankers via Scoring Fine-Grained Relevance Labels, *arXiv*. arXiv:2310.14122v3.

[34] Stureborg, R., Alikaniotis, D., & Suhara, Y. (2024). Large Language Models are Inconsistent and Biased Evaluators, *arXiv*. arXiv:2405.01724v1.