COMPARATIVE ANALYSIS OF RIDGE LOGISTIC REGRESSION, ARTIFICIAL NEURAL NETWORKS AND EXTREME GRADIENT BOOSTING FOR PREDICTING LOAN DEFAULT RATE IN KENYA

MARY MPAINE LEMASULANI

A Thesis Submitted to the Graduate School in Partial Fulfilment of the Requirements for the Award of the Degree of Masters of Science in Applied Statistics of Chuka University

CHUKA UNIVERSITY

OCTOBER, 2025

DECLARATION AND RECOMMENDATION

Declaration

This thesis is my original work and has not been presented for the award of diploma or conferment of degree in any other University or institution.

Signature.....Date.....14/10/2025.....

Lemasulani Mary Mpaine
SM18/35337/18

Recommendation

This thesis has been examined, passed and submitted with our approval as University supervisors.

Signature.....Date.....21/10/2025.....

Prof. Moses Mahugu Muraya, PhD
Chuka University

Signature.....Date.....21/10/2025.....

Dr. Elizabeth Wambui Njoroge, PhD
Chuka University

ii

COPYRIGHT

DEDICATION

I dedicate this thesis to my lovely parents Mpiini Lemasulani, Lasamuria Lemasulani, my sister Agnes Lemasulani and my two pretty children Pascalyn Ltetekwa and Madelyn Olletti.

ACKNOWLEDGEMENT

I wish to express my deepest gratitude to all those who made this thesis possible.

First and foremost, I acknowledge the Almighty God, through whom all things are possible. His grace, strength, and guidance have sustained me throughout this academic journey.

I am profoundly grateful to my supervisors, Prof. Moses Muraya and Dr. Elizabeth Njoroge, whose invaluable insights, mentorship, and unwavering support helped shape the direction and quality of this work. I truly felt their presence and encouragement at every stage. May the Almighty God bless and protect you abundantly.

To my family and friends, your love, motivation, and steadfast support were the pillars of my perseverance. Your belief in me carried me through the most challenging moments. I deeply appreciate each one of you.

I also extend my sincere thanks to the Chuka University fraternity for creating a nurturing academic environment and offering institutional support throughout my studies.

Lastly, I gratefully acknowledge the Central Bank of Kenya for providing access to the data that formed the foundation of this research. Your contribution was instrumental in making this thesis a reality.

ABSTRACT

Predicting loan defaults is vital for financial institutions to mitigate losses from non-repayment loans. Despite access to extensive borrower data, banks often struggle to forecast defaults accurately due to limitations in traditional parametric models, which assume fixed relationships between predictors and outcomes. These models may fail under changing borrower behavior and economic conditions. This study addresses the need for adaptive models that minimize classification error while controlling complexity. The objective was to compare the predictive performance of Ridge Logistic Regression (RLR), Artificial Neural Networks (ANN), and Extreme Gradient Boosting (XGBoost) in forecasting loan default within Kenya Commercial Banks. A retrospective predictive modeling design was employed using secondary data from the Central Bank of Kenya, covering 2012–2022. The dataset included borrower demographics, loan terms, and repayment status. Analysis was conducted using R - 4.5.1 and Python, with model performance evaluated via confusion matrix metrics, Receiver Operating Characteristics (ROC), and Area Under the Curve (AUC). Descriptive analysis revealed class imbalance, variable skewness, and distinct feature distributions, highlighting the need for robust models. RLR identified key predictors such as credit type, loan purpose, gender, and age as increasing default risk, while income and marital status reduced it. However, RLR achieved only moderate accuracy (75.56%), high specificity (99.69%), and very low sensitivity (1.09%), with an AUC of 0.64. In contrast, ANN demonstrated exceptional performance with 99.99% accuracy, perfect AUC of 1.00, and minimal overfitting. The confusion matrix showed 33,686 true negatives and 10,912 true positives, with only three misclassifications. XGBoost achieved 99.9955% accuracy, 100% specificity, and 99.98% sensitivity, with zero false positives and only two false negatives. Its final log-loss of 0.0001736 indicated near-perfect probability calibration. Comparative evaluation revealed that ANN and XGBoost significantly outperformed RLR across all metrics, especially under class imbalance conditions. These findings underscore the superiority of advanced machine learning models in loan default prediction and their potential to enhance risk assessment in Kenyan commercial banks. It is recommended that financial institutions adopt models like ANN and XGBoost to improve predictive accuracy and support data-driven credit decision-making.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS AND ACRONYMS

ANNs	Artificial Neural Networks
AUC	Area under the curve
GLMs	Generalized Linear Models
ML	Machine learning
ROC	Receiver operating characteristic curve
RLR	Ridge Logistic Regression
SVM	Support vector machine

CHAPTER ONE
INTRODUCTION

1.1 Background of the Study

In banking and financial services, accurately predicting loan default rates has been a critical factor for maintaining the stability and profitability of institutions. Loan default prediction is a cornerstone of effective risk management, enabling banks to evaluate borrower creditworthiness and make well-informed lending decisions (Ahmad, 2018). According to the central bank of Kenya the ratio of non performing loans to gross loans in commercial bank stood at 17.6 percent as of May 2025(Central Bank of Kenya [CBK, 2025]). Historically, the banking sector, both in Kenya and globally, grappled with the challenge of forecasting loan default rates with precision (Puli *et al*., 2024). Inaccurate predictions often led to significant repercussions, including financial losses, increased non-performing loans, and potential destabilization of the overall financial health of institutions (Fernandez, 2018).

Predicting loan defaults is essential for financial institutions, as it enables better risk management and more informed lending decisions. By leveraging predictive models, lenders can identify high-risk borrowers in advance and implement proactive strategies to mitigate potential losses. By improving predictive modeling techniques, banks can make more informed decisions, leading to improved financial stability and growth (Jones *et al.,* 2019). Traditionally, banks relied on conventional statistical methods and heuristic approaches for loan default prediction (Victor and Raheem, 2021). However, with the advent of advanced analytical techniques and the proliferation of data, there has been a growing recognition of the potential of machine learning algorithms in improving predictive accuracy (Strielkowski *et al.,* 2023). Against this backdrop, there has been a pressing need to explore and harness the capabilities of machine learning algorithms to enhance loan default prediction (Wang & Ni, 2023).

The dynamic nature of the banking landscape, evolving regulations, and changing economic conditions highlight the need for continuous innovation in loan default prediction methodologies (Wang & Ni, 2023). Various models have been employed in loan default prediction, including Logistic Regression and Support Vector Machines

(Obare *et al*., 2019). The Logistic Regression model operates under the assumption of linear relationships between variables, meaning that the influence of independent variables on the log-odds of the dependent variable is modeled as a linear function. This assumption can be limiting when working with non-linear. This assumption can be limiting when working with non-linear patterns often found in loan default prediction data (Abedin *et al*., 2019). The binary classification of logistic regression model may not adequately capture the complex interdependencies between variables in such scenarios. On the other hand, support vector machines (SVM) have challenges in capturing intricate relationships among features effectively (Cervantes *et al*., 2020). Support Vector Machine algorithms though powerful in many classification tasks, are not accurate in modeling the complex relationships within loan data, potentially affecting the predictive accuracy necessary for informed lending decisions (Goh *et al*., 2019). The Random Forest and Decision Tree algorithms are both popular machine learning techniques. Random Forest is an ensemble method that builds multiple Decision Trees and combines their outputs to improve performance and robustness. A Decision Tree is a simple and interpretable model that splits the data into branches to make decisions based on specified features (Syam *et al*., 2021)

Rhzioual *et al*. (2024) and Madaan *et al*. (2021) demonstrated that the Random Forest algorithm exhibited superior performance compared to the Decision Tree algorithm, resulting to a significantly higher accuracy in the prediction of loan defaults. However, decision trees tend to create simple splits based on individual features, which may not adequately capture the complex interactions that exist between various attributes in loan default data (Bracke *et al*., 2019). Decision Trees are also prone to overfitting, where they can become too specific to the training data and struggle to generalize to unseen data that exhibits more intricate relationships between features (Vercio *et al*., 2020). On the other hand, Random Forest may also face challenges in capturing and interpreting complex feature interactions in loan default data (Kaden *et al*., 2023). According to Nguyen *et al*. (2021), while Random Forest enhances prediction accuracy through ensemble learning, it may still struggle to capture complex feature interactions or subtle dependencies that are crucial for accurately predicting loan defaults.

Despite the efficacy of the already existing models for loan default prediction, these models are not without limitations. One significant challenge lies in their ability to effectively manage intricate relationships and interactions between features within loan default datasets. Loan default prediction data frequently includes complex interactions between features crucial for accurate modeling (Mollo, 2022). For example, the correlation between income and debt-to-income ratio may not follow a linear pattern, affecting default risk variably across income ranges. Likewise, interactions like credit score and loan amount, employment stability and loan purpose, and loan term and interest rate can significantly shape default risk levels. The complexity of these interactions can often surpass the capability of traditional machine learning algorithms, leading to potential inaccuracies in predictions. As a result, there is a need for more advanced techniques or models that can better capture and analyze the complex interplay of variables in loan default scenarios. Incorporating these feature interactions is essential for developing precise models that effectively predict loan defaults and support informed lending decisions (Mollo, 2022).

To address these challenges and enhance the accuracy of loan default prediction models, this study proposes the utilization of advanced techniques such as Ridge logistic regression, Artificial Neural Networks (ANNs), and Extreme Gradient Boosting (XGBoost). Ridge logistic regression stands out for its capability to capture feature interactions through sophisticated engineering techniques, allowing for a more nuanced understanding of the relationships between variables (Abdulhafedh, 2022). Artificial Neural Networks excel in handling complex interactions within data, making them well-suited for capturing the intricate relationships present in loan default prediction datasets. Their ability to learn intricate patterns and dependencies between features can significantly improve predictive accuracy and provide more reliable risk assessments (Victor & Raheem, 2021).

Extreme Gradient Boosting (XGBoost) represents another powerful tool for enhancing feature interactions in predictive modeling (Gachoki *et al*., 2022). Known for its efficiency in handling large datasets and capturing complex dependencies between variables, XGBoost can effectively address the challenges posed by intricate feature interactions in loan default prediction tasks (Ma *et al.,* 2023). By integrating

Ridge Logistic Regression, Artificial Neural Networks, and Extreme Gradient Boosting into the loan default prediction framework, this study aims to improve predictive accuracy and robustness by effectively capturing and leveraging critical feature interactions that influence default risk outcomes (Altunoz, 2024).These advanced models offer the potential to revolutionize the field of loan default prediction and enable financial institutions to make more informed and data-driven lending decisions in an increasingly complex and dynamic financial landscape.

1.2 Statement of the Problem

Accurate prediction of loan default is a critical task for financial institutions seeking to minimize credit risk and maintain portfolio stability, the loan default rate currently in Kenya of non performing loans to gross loans in commercial bank stood at 17.6 percent. Although, statistical models such as Logistic Regression and Support Vector Machines (SVM) have been used to estimate the probability of default based on borrower characteristics. However, these models exhibit structural limitations when applied to high-dimensional, non-linear, and interaction-rich datasets typical of real-world credit environments.

Logistic Regression assumes a linear relationship between independent variables and the log-odds of default, which may oversimplify the complex dependencies inherent in borrower data. Similarly, SVM, while effective in certain classification tasks, often fails to capture higher-order feature interactions without extensive kernel tuning. Empirical evidence suggests that loan default behaviour is influenced by intricate variable interdependencies for example, the marginal effect of income on default risk may vary conditionally with debt-to-income ratio, loan term, or credit score. These interaction effects are not easily modeled using traditional parametric techniques.

To address these limitations, advanced machine learning algorithms such as Ridge Logistic Regression (RLR), Artificial Neural Networks (ANN) and Extreme Gradient Boosting (XGBoost) offer flexible, non-linear modeling capabilities. These methods are designed to handle multicollinearity, capture non-linear relationships and model complex feature interactions, thereby improving predictive accuracy and generalization performance.

Given the increasing availability of granular financial data and the evolving risk landscape in Kenya's commercial banking sector, there is a pressing need to statistically evaluate and compare the performance of these modern algorithms. This study aims to identify the most robust and interpretable predictive framework for loan default classification, with a focus on enhancing model accuracy, stability, and practical applicability in the Kenyan context.

1.3 Objectives of the Study

1.3.1 Broad Objective

The broad objective of this study is to develop, evaluate and compare the predictive performance of Ridge Logistic Regression, Artificial Neural Networks and Extreme Gradient Boosting in forecasting loan default in Kenya Commercial Bank.

1.3.2 Specific Objectives

i. To develop predictive models for loan default using Ridge Logistic Regression, Artificial Neural Networks, and Extreme Gradient Boosting in Kenya Commercial Bank.
ii. To evaluate and compare the predictive performances of Ridge Logistic Regression, Artificial Neural Networks and Extreme Gradient Boosting in forecasting loan default in Kenya Commercial Bank.

1.4 Research Questions

i. How can predictive models for loan default be developed using Ridge Logistic Regression, Artificial Neural Networks, and Extreme Gradient Boosting in Kenya Commercial Bank?
ii. How do the predictive performances of Ridge Logistic Regression, Artificial Neural Networks and Extreme Gradient Boosting compare in forecasting loan defaults in Kenya Commercial Bank?

1.5 Significance of the Study

The findings from this study will directly benefit financial institutions in Kenya, providing actionable insights to refine their risk assessment frameworks and enhance lending practices. By delving into the influential factors contributing to default rates

and evaluating the performance of diverse machine learning algorithms, financial institutions can strengthen their risk mitigation strategies, reduce non-performing loan ratios, and bolster financial stability. The insights derived from this study will advance predictive analytics within the risk management framework of Kenya Commercial Bank (KCB), enhancing its ability to identify and mitigate loan default risks. By demonstrating the effectiveness of modern machine learning techniques, the findings may serve as a benchmark for other financial institutions confronting similar challenges, promoting data-driven decision-making and more resilient credit evaluation practices across the sector.By shedding light on the comparative effectiveness of machine learning algorithms, the study equips the industry with valuable guidance on selecting appropriate models to navigate the complexities of borrower data within Kenya's socio-economic context.

The research findings will be instrumental in shaping guidelines to promote a more resilient and stable banking sector in Kenya. Policymakers and regulators can leverage on the outcomes to refine regulatory frameworks, striking a balance between fostering financial inclusion and safeguarding the sector against undue risks. The significance of this study lies in its potential to drive tangible improvements in risk management practices, enhance the financial health of institutions and contribute to the broader landscape of banking practices and policies in Kenya. By using various machine learning algorithms, including Ridge Logistic Regression, Artificial Neural Networks and Extreme Gradient Boosting, the study aims to revolutionize risk assessment frameworks and lending practices within Kenya's financial institutions.

Furthermore, scholars and researchers in the fields of applied statistics, finance, data science and machine learning stand to benefit significantly from this study. The study outcomes will contribute to the academic knowledge base by providing insights into the application of advanced machine learning algorithms in the context of loan default prediction for financial institutions. Scholars can leverage the findings to enhance their understanding of the predictive analytics landscape in the banking sector and explore further avenues for research and innovation.

1.6 Assumptions of the Study

To ensure methodological rigour and valid inference in comparing the predictive performance of Ridge Logistic Regression (RLR), Artificial Neural Networks (ANN), and Extreme Gradient Boosting (XGBoost), the following assumptions were made:

i. Data Quality and Representativeness: The dataset obtained from Kenya Commercial Bank (KCB) is assumed to be accurate, complete, and representative of the broader population of borrowers in Kenya. Any Missing values or inconsistencies were assumed to have been appropriately addressed during preprocessing.
ii. Feature Relevance and Sufficiency: The predictor variables included in the dataset are assumed to provide sufficient and relevant information for distinguishing between defaulters and non defaulters.
iii. Stationarity and Temporal Stability: The relationships between predictors and default outcomes are assumed to have remained stable during the data collection period, with no major structural economics or policy changes significantly altering repayment behaviour.
iv. Model-Specific Assumptions: Its assumed that the chosen models (RLR, ANN &XGBoost) are suitable for the nature of dataset and capable of capturing both linear and non linear relationships relevant to loan default prediction.
v. Evaluation Metrics Validity: All models are assumed to have been trained and tested on the same dataset split, using consistent preprocessing and hyperparameter tuning, ensuring comparability of their predictive performance.
vi. Comparability Across Models: It is assumed that the chosen performance metrics (AUC-ROC, precision recall and F1 score) are valid and reliable measures for assessing predictive performance in the cntext of imbalance classifications.

CHAPTER TWO
LITERATURE REVIEW

2.1 Overview of Loan Defaults

The financial landscape of Kenya has witnessed a concerning rise in loan defaults in recent years, raising significant alarms within the banking sector and regulatory bodies. According to the Central Bank of Kenya the ratio of non performing loans to gross loans in commercial bank stood at 17.6 percent as of May 2025(CBK, 2025; Bank of Tanzania; Bank of Uganda) which is higher relative to other East African countries such as Tanzania with 2.9% and Uganda 4.5% default rate respectively.

Non-performing loans, which refer to debts that borrowers have failed to repay for an extended period, have surged, impacting the profitability and stability of financial institutions (Muriithi, 2013). According to Galaz & Collste, (2022) escalation is particularly worrisome as it reflects broader economic challenges and systemic issues within the lending ecosystem.

The implications of increasing loan defaults extend beyond individual borrowers and lenders to the overall health of the economy. As non-performing loans accumulate, banks are forced to allocate substantial resources to cover potential losses, limiting their capacity to extend credit to other borrowers. According to Duq*i, (*2022), the ripple effects of defaults can slowdown economic growth, dampen investor confidence, and hinder the efficient functioning of financial markets. Thus, understanding the underlying factors driving this trend is essential for devising effective strategies to mitigate its adverse effects and foster a more resilient financial system (Rangpur *et al.*,2023).

2.2 Modeling Loan Default for Financial Banks

2.2.1 Generalized Linear Models

Generalized Linear Models (GLMs) extend traditional linear regression by accommodating various types of response variables and link functions, offering greater flexibility in modeling loan default probabilities. This adaptability makes GLMs suitable for financial datasets that exhibit non-normal distributions and complex relationships. However, they noted that model complexity increases with the

number of link functions, potentially affecting interpretability and computational efficiency.

Wilson *et al.* (2024) investigated the use of GLMs in automotive finance and demonstrated that GLMs outperformed traditional linear models in default prediction. Nonetheless, their study highlighted a key limitation: GLMs are sensitive to overfitting when too many predictors are included, which can compromise generalization performance.

Gonzalez and Fernandez (2023) explored the application of GLMs to both structured and unstructured data in default prediction. While GLMs enhanced predictive performance with structured data, the integration of unstructured data proved challenging due to its complexity and the time-intensive nature of preprocessing. These findings underscore both the strengths and limitations of GLMs in loan default modeling, particularly in contexts requiring the handling of diverse data types and complex variable interactions. Several studies have explored the application of Generalized Linear Models (GLMs) in loan default prediction, highlighting both their strengths and limitations. (Agbemava *et al*., 2016) compared GLMs with Logistic Regression in the context of consumer loan default prediction. Their findings indicated that GLMs outperformed Logistic Regression by effectively incorporating interaction terms and non-linear effects. However, they noted that GLMs can become computationally intensive when applied to large datasets.

Ha *et al.* (2021) employed GLMs to integrate macroeconomic indicators and borrower characteristics, demonstrating improved model accuracy across diverse data sources. Nonetheless, the effectiveness of this approach was contingent on the availability and reliability of macroeconomic data. Doosti *et al*. (2024) applied Generalized Linear Models (GLMs) to evaluate the impact of financial behavior on loan default risk. Their study yielded valuable insights, but emphasized the importance of carefully selecting link functions and predictors to avoid model mis-specification and preserve interpretability. Perez *et al*. (2022) examined GLMs in the context of mortgage loan default prediction. While the models demonstrated promising results, they encountered challenges in handling very high-dimensional datasets, which limited their scalability and computational efficiency in complex

environments. Kumar and Reddy (2022) investigated the use of GLMs for default prediction in microfinance. Their findings revealed that GLMs were highly sensitive to model specification and prone to overfitting, particularly when applied to small or noisy datasets.

2.2.2 Logistic Regression

Logistic regression is a widely studied statistical model used in various fields for modeling binary outcomes. Its simplicity and interpretability make it a popular choice for binary classification tasks, including loan default prediction. Several studies have examined the application of Logistic Regression (LR) in predicting loan defaults, highlighting both its strengths and limitations across diverse financial contexts. Abid *et al*. (2018) utilized LR to model default risk using customer demographic and financial data. While the model demonstrated reasonable predictive capability, it was constrained by its assumption of linear relationships and sensitivity to multicollinearity.

Zhou *et al.* (2022) compared LR with other machine learning algorithms for loan default prediction. Although LR was favored for its simplicity and interpretability, it underperformed in capturing complex non-linear relationships present in borrower data. Agrawal *et al.* (2014) applied LR to assess the influence of macroeconomic variables on default rates. Their findings emphasized that the model's effectiveness heavily depends on the accurate measurement and availability of macroeconomic indicators. Chen *et al.* (2021) examined LR in the context of peer-to-peer lending. They observed that model performance deteriorated when faced with imbalanced datasets, a common challenge in default prediction. Abidin *et al.* (2020) explored LR for predicting defaults among small and medium-sized enterprises (SMEs). Despite its utility, the model struggled to capture intricate interactions between traditional models.

Chouksey *et al.* (2023) employed LR using transactional data from mobile banking platforms. The model achieved strong performance when supported by rigorous data preprocessing, underscoring the critical role of data quality in predictive accuracy. Liu and Chen (2023) evaluated the impact of credit history on default probability using

LR. Their study confirmed the significance of past credit behaviour, though they cautioned that historical data may not always reliably predict future defaults. Finally, Latham & Braun (2011) analyzed LR's performance during economic downturns. The study revealed that LR's predictive accuracy varied across different economic cycles and regional contexts, suggesting limitations in its adaptability to macroeconomic volatility.

2.2.3 Ridge Logistic Regression

Ridge Logistic Regression (RLR), also known as L2-regularized logistic regression, is a parametric extension of traditional logistic regression that incorporates a penalty term on the magnitude of the coefficients. This regularization technique mitigates overfitting, particularly in scenarios where the number of predictors is large relative to the number of observations and where multicollinearity is present. Numerous studies have demonstrated the effectiveness of RLR across diverse domains.

Huang *et al.* (2015) applied RLR to predict the risk of type 2 diabetes using clinical and demographic variables, reporting superior predictive accuracy and model stability compared to traditional logistic regression. Similarly, Mugova (2014) employed RLR to forecast ICU readmissions and found it to outperform conventional models in predictive performance. Liu *et al. (*2016) used RLR to assess stroke risk factors, highlighting its strength in handling correlated predictors and improving model robustness. Moula *et al.* (2017) confirmed RLR's higher accuracy and resilience in predicting hospital readmissions.

Chen *et al.* (2021) applied RLR to Alzheimer's disease progression modeling, demonstrating enhanced performance and stability. Wang *et al.* (2018) used RLR to predict postoperative complications, showing improved accuracy and reduced overfitting. Clark *et al.* (2013) compared RLR with traditional logistic regression and other advanced models, concluding that RLR was more effective in managing multicollinearity and stabilizing predictions. Robinson *et al.* (2014) applied RLR to consumer credit risk modeling, demonstrating its utility in predicting loan defaults while managing correlated variables.

In the financial domain, Martinez *et al*. (2015) emphasized the role of regularization in improving loan default prediction accuracy using RLR. Chen *et al.* (2018) applied RLR to financial distress prediction, noting improvements in performance due to better handling of multicollinearity. Patel *et al*. (2016) investigated RLR in high-dimensional credit data and found it adept at managing dimensionality and producing stable predictions. Rashid *et al*. (2021) compared RLR with Lasso and Elastic Net, concluding that RLR maintained superior model stability and accuracy. Zhao *et al*. (2019) used RLR to predict hospital-acquired infections, achieving higher performance than traditional logistic regression.

Kim *et al.* (2020) applied RLR to predict Chronic Obstructive Pulmonary Disease (COPD) risk, reporting improved accuracy and multicollinearity management. Patel *et al.* (2017) further demonstrated RLR's robustness and predictive power in modeling patient mortality in emergency departments using clinical and demographic data. Fazakis *et al.* (2021) applied Ridge Logistic Regression to predict complications in diabetic patients using a comprehensive dataset comprising clinical and lifestyle variables. Their findings indicated that Ridge Logistic Regression outperformed traditional logistic regression in terms of predictive accuracy and model stability.

Gupta *et al*. (2018) employed Ridge Logistic Regression to forecast hypertension risk based on clinical and demographic data. The model demonstrated superior performance compared to traditional logistic regression, particularly in handling multicollinearity and maintaining predictive consistency. Johnson *et al.* (2020) utilized Ridge Logistic Regression to predict mental health disorders using a dataset that included clinical, demographic, and genetic factors. The study highlighted Ridge Logistic Regression's effectiveness in managing correlated predictors and enhancing overall model robustness.

2.2.4 Artificial Neural Networks

Artificial Neural Networks (ANNs) are non-parametric models composed of interconnected nodes (neurons) organized into layers typically including an input layer, one or more hidden layers, and an output layer. These models are capable of capturing complex, non-linear relationships in data, making them particularly suitable

for tasks such as loan default prediction, medical diagnostics, and agricultural forecasting. ANNs require careful tuning of hyperparameters and network architecture to achieve optimal performance. Overfitting remains a concern, especially with limited or noisy datasets, necessitating the use of regularization techniques such as dropout, batch normalization and early stopping.

In the financial domain, Wang *et al*. (2011) demonstrated that ANNs outperform traditional statistical models in credit scoring due to their ability to learn intricate patterns. Although, Bhatore et al. (2019) confirmed that Artificial Neural Networks (ANNs) can achieve high accuracy in loan default prediction, their performance remains highly sensitive to hyperparameter tuning and network architecture. However, limited research has examined how these sensitivities manifest within the context of Kenyan commercial banks, where data variability, resource constraints, and regulatory considerations may affect model optimization. (Gajdosikova et al., 2012) provided comparative evidence that ANNs surpass logistic regression and decision trees in handling non-linearity and improving predictive accuracy.

Chen *et al*. (2016) demonstrated that Artificial Neural Networks (ANNs) are effective in credit risk modeling due to their ability to capture complex, non-linear risk factors. However, their study did not address how these capabilities translate within the context of Kenyan commercial bank, where data structures, borrower behavior, and regulatory dynamics may differ significantly from global benchmarks. Khemakhem *et al*. (2018) showed that ANNs remain effective even in developing economies with limited and variable data. Wilson *et al.* (2016) explored deep learning variants, including convolutional and recurrent neural networks, for loan default prediction, achieving improved accuracy and modeling temporal dependencies. Recent advancements have focused on enhancing ANN performance. Martinez *et al.* (2017) examined architectural improvements such as dropout and batch normalization, which led to more robust credit scoring models.

Patel *et al.* (2024) emphasized the importance of feature selection in boosting ANN accuracy. Kumar *et al.* (2023) explored ensemble methods combining ANNs with classifiers like Random Forests and SVMs, yielding superior performance. Maruf *et*

al. (2024) and Kowsar *et al.* (2023) investigated ANN applications in financial distress and time-series loan default prediction, respectively, demonstrating the adaptability of ANNs to dynamic data environments. Hybrid and deep learning applications have further expanded ANN capabilities. Liu *et al.* (2024) introduced a hybrid ANN-genetic algorithm model for feature selection, significantly improving robustness. Silver *et al*. (2016) showcased the power of deep neural networks combined with reinforcement learning in AlphaGo, illustrating the scalability of ANNs in complex decision-making. Musallam *et al*. (2022) and Ghosh *et al.* (2017) demonstrated the effectiveness of convolutional neural networks (CNNs) in medical imaging and document recognition, respectively.

In agriculture and health, ANNs have shown remarkable versatility. Ali et al. (2016) and Sumathi et al. (2022) used ANNs for crop yield prediction, outperforming traditional models. Wang *et al.* (2024) linked chronic disease prediction to financial stability and loan repayment, while Zainab (2022) and Patnaik *et al.* (2024) compared ANN and Ridge Logistic Regression in agricultural loan default prediction, with ANNs showing slightly superior accuracy. ANNs have also been applied to early disease detection and diagnostics. Kshatriya *et al.* (2023), Tiwari et al. (2013) and Maroco *et al.* (2011) demonstrated ANN effectiveness in predicting diabetes, liver disease, and Alzheimer's progression. Dev *et al*. (2022) and Gonem *et al.* (2020) confirmed ANN superiority in stroke and respiratory condition prediction.

Environmental applications include pest detection (Huang *et al.*, 2021), fertilizer optimization (Sharma *et al.*, 2024), fruit quality assessment (Chen et al., 2024), climate modeling (Vojnov *et al.*, 2022), livestock disease forecasting (Delfani *et al.*, 2024), and drought prediction (Jain *et al.*, 2023), all showcasing ANNs' ability to model complex interactions and improve decision-making.

2.2.5 Extreme Gradient Boosting

Extreme Gradient Boosting (XGBoost) is a powerful ensemble learning algorithm that builds predictive models by minimizing a specified loss function through iterative boosting of weak learners, typically decision trees. Known for its scalability, high performance, and ability to handle missing values and mixed data types, XGBoost has

become a leading choice in both machine learning competitions and real-world applications (Hakkal *et al.*, 2024).

Yadav *et al.* (2021) demonstrated XGBoost's effectiveness in predicting early Alzheimer's disease, achieving superior accuracy and sensitivity compared to other models. Similarly, Zhao *et al.* (2021) applied XGBoost to hospital readmission prediction, where it outperformed traditional statistical methods and other machine learning algorithms. In healthcare, Yang *et al.* (2022) used XGBoost to predict heart disease and found it to outperform SVM, k-NN, and Random Forests in terms of accuracy and AUC-ROC. Ramesh *et al.* (2021) confirmed XGBoost's scalability and robustness across various clinical prediction tasks using electronic health records. Islam *et al*. (2023) applied XGBoost to chronic kidney disease prediction, reporting superior accuracy and precision.

Nyam *et al.* (2023) showed that XGBoost achieved higher predictive accuracy for ICU patient outcomes compared to logistic regression and Random Forests. Wu *et al*. (2022) applied XGBoost to diabetes onset prediction, again outperforming other models. Chen and Guestrin (2016) originally introduced XGBoost, highlighting its efficiency and distributed architecture. Thompson *et al*. (2023) demonstrated its ability to handle high-dimensional financial data, while Qureshi (2023) confirmed its robustness across diverse economic contexts. In financial risk modeling, Zhou *et al.* (2023) found XGBoost to outperform Artificial Neural Networks (ANNs) in loan default prediction due to its ability to integrate various loss functions and generalize across data distributions. Dube *et al*. (2023) emphasized XGBoost's strength in managing imbalanced datasets, a common challenge in credit risk modeling. Its built-in regularization mechanisms help prevent overfitting, making it suitable for noisy and variable financial data.

Wang *et al.* (2021) applied XGBoost to hospital readmission prediction, linking its accuracy to financial resource planning. Chung *et al.* (2023) and Wang *et al.* (2021) used XGBoost for chronic disease and cardiovascular risk prediction, respectively, demonstrating its effectiveness with heterogeneous data. Jiang *et al.* (2021) and Cheng *et al.* (2020) applied XGBoost to emergency department forecasting and

influenza outbreak prediction, showing its utility in operational planning and public health surveillance.

Li and Chen (2020) conducted a benchmarking study, concluding that XGBoost consistently outperforms single classifiers like logistic regression and ANNs in credit risk prediction. Although Repetto (2023) confirmed XGBoost's effectiveness in modeling loan defaults using large, complex datasets, the study did not explore its applicability within the unique data environments of Kenyan commercial banks. Specifically, the impact of localized socio-economic factors, regulatory constraints, and data sparsity on XGBoost's predictive performance remains underexplored. Roberts *et al.* (2021) enhanced XGBoost's interpretability using Shapley values. Khalfaoui *et al.* (2022) applied XGBoost to time-series loan default prediction, demonstrating its ability to model temporal patterns. While Li *et al.* (2021) demonstrated XGBoost's effectiveness in emerging markets with limited and variable datasets, and Wang *et al*. (2022) along with Singha *et al.* (2021) confirmed its strength in modeling complex feature interactions in high-dimensional environments, there remains a lack of empirical evidence on its performance within the Kenyan commercial banking sector.

The XGBoost can enhance loan default prediction under local data constraints, regulatory requirements, and socio-economic variability has not been thoroughly investigated. This study seeks to address that gap by evaluating XGBoost's predictive capabilities using real-world financial data from Kenya.Beyond finance and healthcare, XGBoost has shown strong performance in other domains. Mustafa *et al.* (2022) used it for plant disease prediction from image data. Indumathi *et al.* (2021) applied it to human activity recognition using sensor data. Zhou et al. (2023) demonstrated its accuracy in renewable energy forecasting. Manosuthi et al. (2021) found XGBoost to outperform other models in predicting customer lifetime value.

Zhao *et al.* (2023) highlighted XGBoost's efficiency in processing large volumes of text data, its application in financial risk modeling within data-constrained environments such as those found in Kenyan commercial banks remains underexplored. Specifically, the potential of XGBoost to extract meaningful patterns

from structured financial data, where text-based features are limited or absent, has not been sufficiently investigated. This study seeks to bridge that gap by evaluating XGBoost's performance in predicting loan defaults using structured borrower and transaction data from Kenya's banking sector. Mohmand *et al.* (2022) showcased its precision in anomaly detection and cybersecurity. Antoniadi *et al.* (2021) examined the use of XGBoost in clinical decision support systems, demonstrating its ability to improve diagnostic accuracy and patient outcomes. The model outperformed traditional statistical approaches by providing reliable predictions across a range of medical conditions.

Zhao *et al.* (2023) applied XGBoost to predict student performance, comparing its effectiveness with other machine learning models and conventional statistical methods. XGBoost achieved superior accuracy, highlighting its robustness in educational data analysis. Ahmad *et al.* (2019) investigated XGBoost for customer churn prediction in the telecommunications industry. The model demonstrated high accuracy and efficiency in processing large-scale customer datasets, outperforming alternative predictive models. Zhang *et al.* (2023) evaluated XGBoost's performance in air quality prediction across multiple datasets. XGBoost consistently delivered high accuracy and handled large volumes of environmental data effectively, supporting its utility in monitoring and management applications.

Kaur *et al.* (2022) explored XGBoost's role in forecasting disease outbreaks. The model showed superior performance compared to other machine learning and statistical methods, providing accurate forecasts that facilitated early detection and prevention strategies. Turgut *et al*. (2022) applied XGBoost to sales and demand forecasting in the retail sector. Their findings indicated that XGBoost processed large datasets efficiently and outperformed traditional models, enabling retailers to optimize inventory and improve strategic planning.

CHAPTER THREE
METHODOLOGY

3.1 Site of Study

The study was conducted at the headquarters of Kenya Commercial Bank (KCB), the largest financial institution in East Africa, situated in Nairobi, Kenya. KCB's prominence within the region, coupled with its extensive asset value and widespread presence, makes it an ideal site for this study.

3.2 Research Design

The study adopted a retrospective quantitative research design, utilizing predictive modelling techniques to analyze historical loan data. A retrospective design was appropriate because it relies on already existing record therefore, minimizing the time and cost associated with primary data collection (Kuhn & Johnson, 2013). Within this framework, predictive modelling was employed as the analytical approach to build, train, and evaluate models using past borrower and loan characteristics. This allowed for a systematic comparison of three algorithms, Ridge Logistic Regression, Artificial Neural Networks, and Extreme Gradient Boosting on their ability to predict loan defaults. The use of retrospective data supports the discovery of patterns and relationships that inform future outcomes, which aligns with the purpose of predictive analytics in high-risk decision-making contexts such as finance (Shmueli & Koppius, 2011).

3.3 Data Collection

The study utilized secondary data obtained from the Central Bank database, covering the period from 2012 to 2022. The dataset comprised 148,670 loan records from Kenya Commercial Bank with 16 variables: 15 predictors (demographic features: age, gender, marital status; loan characteristics: loan amount, interest rate, term, type, purpose; credit variables: credit score, credit type, creditworthiness; financial indicators: income, property value, business classification, loan limit) and one target variable default status: (0=non-default, 1=default).

To ensure the representativeness of the sample, stratified sampling was employed. The population was first divided into distinct strata defined based on key demographic

and loan-related variables, including but not limited to type of loan (for example personal loans, business loans, mortgages), the borrower's financial status, and other pertinent socio-economic characteristics. This approach ensured that various subgroups, such as individuals and businesses seeking financial assistance from Kenya Commercial Bank, were adequately represented.

3.4 Data Analysis

The analysis of quantitative data involved both descriptive and inferential statistics. Descriptive statistics were used to summarize and characterize the loan default rates across Kenya Commercial Bank providing insights into trends and patterns. Inferential statistics were then employed to identify significant factors contributing to loan default rates, enabling the development of deeper insights into the underlying drivers of defaults.

For the modelling and prediction of loan defaults, R version 4.5.1 and Python 3.8.10 were utilized. Python libraries included TensorFlow 2.10.0 for ANN, XGBoost 1.7.3, scikit-learn 1.2.0 for preprocessing and metrics, pandas 1.5.2 for data manipulation, and NumPy 1.23.5 for numerical operations. These powerful statistical and data analysis tools were widely recognized for their versatility in handling large datasets, applying machine learning algorithms, and performing complex statistical analyses. R was employed for statistical modeling and visualization (Appendix 4), while Python was used for machine learning tasks (Appendix 3), including the implementation of Ridge Logistic Regression, Artificial Neural Networks and Extreme Gradient Boosting models.

3.4.1 Data Preprocessing

Data preprocessing was conducted to ensure data quality throughout the analysis. Missing numerical values ranged from 2.3% to 5.8% across variables and were imputed using mean values. Categorical variables were imputed using mode values. This approach retained all 148,670 records. Categorical variables were systematically encoded using label encoding. Marital status was encoded as Single=1, Married=2, Divorced=3. Gender was encoded as Female=0, Male=1. Loan type was encoded as Secured=1, Unsecured=2, Specialized=3. Loan purpose was encoded as Vocation=1,

Car=2, Appliances=3, Education=4. Creditworthiness was encoded as No=0, Yes=1. Business ownership was encoded as No=0, Yes=1. Credit bureau type was encoded as CRB=1, CRIF=2, EQUI=3, EXP=4. Age groups were encoded as <25=1, 25-34=2, 35-44=3, 45-54=4, 55-64=5, 65-74=6, >74=7. Numerical features were standardized using Z-score normalization with mean=0 and SD=1. StandardScaler was applied with scaling parameters fitted exclusively on training data to prevent data leakage. The dataset was partitioned using stratified sampling into training and testing sets. The training set contained 70% of data (n=104,069) and the testing set contained 30% (n=44,601).

3.5 Statistical Model

3.5.1 Ridge Logistic Regression

Ridge Logistic Regression (RLR), specifically corrects for multicollinearity in regression analysis. For loan default prediction, the model estimates probability $P(Default = 1|X)$ given borrower characteristics. Ridge Logistic Regression equation is given as,

$$log\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_{15} x_{15} - \lambda \sum_{i=1}^{15} \beta_i^2 \qquad (3.1)$$

where,

$p = P(Default = 1|X)$, that is the probability of loan default.

$x_{1\ldots} x_{15}$ are predictor variables: loan amount, income, credit score, interest rate, term, property value, marital status, gender, age, loan type, loan purpose, creditworthiness, business or commercial, credit type, loan limit.

β_i are coefficients quantifying change in log-odds per unit increase in predictor.

λ is the regularization strength reducing overfitting by penalizing large coefficients.

Regularization parameter α was optimized via grid search cross-validation. The tested values were 0.1, 1.0, 10.0, and 100.0. The optimal value selected was α=10.0. The optimization used the lbfgs solver (Limited-memory BFGS). This solver efficiently handled 104,069 training observations through iterative gradient-based minimization. A prediction threshold of P=0.5 was used for binary classification. Probabilities ≥0.5

were classified as default and <0.5 as non-default. Feature importance was assessed using coefficients (β) and odds ratios (e^{β}). A positive β indicated increased default risk. A negative β indicated a reduced default risk.

3.5.2 Extreme Gradient Boosting

Extreme Gradient Boosting (XGBoost) employed regularization techniques and a depth-first approach to prune trees, thereby mitigating the risk of overfitting. For loan default, XGBoost builds sequential decision tree ensembles where each tree corrects prior errors, producing refined default probability estimates. The algorithm was capable of effectively managing missing data, ensuring that the model remains robust even with incomplete datasets. The objective function in XGBoost was composed of two parts: a loss function that measured how well the model fits the training data and a regularization term that penalized the complexity of the model to avoid overfitting.

$$L(\theta) = \sum_{i=1}^{n} l(y_i, \hat{y}_i + \sum_{k=1}^{K} \Omega(f_k) \qquad (3.2)$$

where,

L is the loss function (binary cross-entropy for classification)

$\hat{y}_i$ is the predicted value for the i^{th} instance.

y_i is the actual value

$\Omega(f_k)$ is the regularization term for the k^{th} tree, which controls the complexity of the model

K is the total number of trees

Regularization term for the f_k is defined as;

$$\Omega(f) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} \omega_j^2 \qquad (3.3)$$

where,

T was the number of leaves in the tree

w_j was the weight of the j^{th} leaf

γ and λ are regularization parameters

Additive model

XGBoost builds the model in an additive manner, suppose we have m trees, the prediction of the i^{th} instance was given as,

$$\hat{y}_i = \sum_{k=1}^{m} f_k(x_i) \tag{3.4}$$

where,

f_k was the function (tree) added at the k^{th} step

Tree structure

Each tree partitions the data into regions and assigns a score or weight to each region. The tree can be represented as,

$f(x) = w_{q(x)}$

where,

q was a function mapping an instance to a leaf

w was the weight associated with the leaf

Objective function for additive trees

To fit the model, XGBoost minimized the objective function in a greedy manner by adding one tree at a time. The objective function after adding t^{th} tree can be written as,

$$L^{(t)} = \sum_{i=1}^{n} l\left(y_i, \hat{y}_i^{t-1} + f_t(x_i)\right) + \Omega(f_t) \tag{3.5}$$

Using a second-order Taylor expansion to approximate the loss function, the objective function becomes,

$$L^t \approx \sum_{i=1}^{n} \left[l\left(y_i, \hat{y}_i^{t-1}\right) + g_i f_t(x_i) + \frac{1}{2} h_i \left(f_t(x_i)\right)^2\right] + \Omega(f_t) \tag{3.6}$$

where,

$g_i = \frac{\partial l(y_i, \hat{y}_i^{t-1})}{\partial \hat{y}_i^{t-1}}$ is the first order gradient

$h_i = \frac{\partial^2 l(y_i, \hat{y}_i^{t-1})}{\partial (\hat{y}_i^{t-1})^2}$ is the second order gradient (Hessian)

Optimizing the Tree

To find the best tree f_t, XGBoost optimized the objective function. This involves:

i. Finding the Best Split: Splitting nodes to reduce the loss.

ii. Calculating the Gain: Evaluating the quality of a split using the gain, which measures the improvement in the objective function.

The gain for a split can be calculated as:

$$\text{Gain} = \frac{1}{2}\left[\frac{G_L^2}{H_L+\lambda} + \frac{G_R^2}{H_R+\lambda} - \frac{(G_L+G_R)^2}{H_L+H_R+\lambda}\right] - \gamma \tag{3.7}$$

where,

G_L and G_R are the sums of gradients for the left and right splits.

H_L and H_R are the sums of Hessians for the left and right splits.

λ and γ are regularization parameters.

XGBoost was implemented using a binary logistic objective function to produce probabilistic outputs ranging from 0 to 1. Model performance was evaluated using the logarithmic loss (log-loss) metric, which measures the accuracy of predicted probabilities. Hyperparameter optimization was conducted through cross-validation to enhance generalization performance.

The learning rate (η) was tested at values of 0.01, 0.1, and 0.3 to regulate the contribution of each tree to the overall model. Maximum tree depth was evaluated at 3, 5, and 7 to control model complexity and mitigate overfitting. The number of estimators (boosting rounds) was varied across 50, 100, and 200. Regularization parameters λ (L2) and γ (minimum loss reduction for further partitioning) were tuned to further prevent overfitting.

Predicted probabilities were converted into binary classifications using a threshold of P = 0.5, consistent with the approach used in the regularized logistic regression (RLR) model. Feature importance was quantified using XGBoost's built-in metrics: Gain, representing the average improvement in loss from splits involving a feature; Frequency (or Weight), denoting the number of times a feature was used in splits; and Cover, indicating the number of observations influenced by a feature's splits.

3.5.3 Artificial Neural Networks

Artificial Neural Networks (ANNs) are a class of machine learning algorithms inspired by the structure and function of the human brain. They are particularly effective for complex pattern recognition tasks, including the prediction of loan defaults. A feed forward neural network learned nonlinear relationships between 15 borrower characteristics and default probability through hierarchical feature transformation. ANNs consist of interconnected layers of nodes (neurons), where each connection has an associated weight. In the context of loan prediction, ANNs can be used to model the probability that a borrower will default on a loan based on various input features such as credit score, income, employment history, gender, marital status and loan characteristics. The network learned to identify patterns and relationships within the data that are indicative of default risk.

The neural network model comprised four layers: an input layer, two hidden layers, and an output layer. The input layer consisted of 15 neurons corresponding to the standardized predictor variables. The first hidden layer contained 32 neurons with Rectified Linear Unit (ReLU) activation to capture nonlinear relationships among predictors. The second hidden layer included 16 neurons, also activated by ReLU, to enable the extraction of higher-level feature representations. The output layer comprised a single neuron with a sigmoid activation function, producing predicted probabilities of default, $P(\text{Default}) \in [0,1]$. A classification threshold of $P > 0.5$ was applied, whereby observations with predicted probabilities equal to or greater than 0.5 were classified as default, and those below 0.5 as non-default.

Neuron Model

Each neuron receives inputs, processes them, and produced an output. The basic operations are:

i. Weighted Sum: Each input is multiplied by a corresponding weight.
ii. Activation Function: Applies a non-linear transformation to the weighted sum.

Equations

Weighted sum

For a neuron *j* in a layer:

$$z_j = \sum_i w_i x_i + b_j \quad (3.8)$$

where,

z_j was the weighted sum for neuron j.

w_{ij} was the weight from input i to neuron j.

x_i was the input i.

b_j was the bias term for neuron j.

Activation function

The activation function ϕ introduced non-linearity:

$$a_j = \phi(z_j) \quad (3.9)$$

where,

a_j was the activation (output) of neuron j.

ϕ was the activation function [e.g., sigmoid, Rectified Linear Unit (ReLU), tanh].

where,

Sigmoid activation function was given as,

$$\phi(z) = \frac{1}{1+e^{-z}} \quad (3.10)$$

Rectified Linear Unit activation function was given as,

$$\phi(Z) = \max(0, Z)$$

tanh activation function was given as,

$$\phi(Z) = \tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$$

(3.11)

Forward propagation entailed calculating the output of the network given the input features:

i. Calculate the weighted sum z for each neuron.
ii. Apply the activation function to get the activation aa.
iii. Repeat for all layers until the output layer.

Loss Function

The loss function L measured the difference between predicted and actual values. Common loss functions include MSE for regression and cross-entropy for classification. Mean Squared Error and Cross-Entropy are calculated as,

Mean Squared Error:

$$L = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \quad (3.12)$$

Cross-entropy:

$$L = \frac{1}{n}\sum_{i=1}^{n}[y_i \, log(\hat{y}_i) + (1 - y_i) log(1 - \hat{y}_i)] \quad (3.13)$$

Gradient descent

Weights and biases are updated using gradient descent, which minimized the loss function by moving in the direction of the negative gradient. It entails:

i. Calculate the gradient of the loss function with respect to weights and biases.
ii. Update weights and biases as,

$$w_{ij} = w_{ij} - \eta \frac{\partial L}{\partial w_{ij}}$$

$$b_j = b_j - \eta \frac{\partial L}{\partial b_j}$$

where,

η was the learning rate.

Backpropagation was the algorithm used for training ANNs. It involved updating weights and biases to minimize the loss function. Backpropagation used the chain rule to compute gradients:

For a weight w_{ij}:

$$\frac{\partial L}{\partial w_{ij}} = \frac{\partial L}{\partial_{aj}} \cdot \frac{\partial a_j}{\partial z_j} \cdot \frac{\partial z_j}{\partial w_{ij}}$$

The model was trained using binary cross-entropy as the loss function to measure prediction error. The optimizer was Adam, which combines momentum and adaptive learning rates. The learning rate was set to η=0.001. The training configuration included 16 epochs with a batch size of 10. A validation split of 20% was used to monitor overfitting. Regularization was implemented by monitoring validation loss

across epochs. Backpropagation computed gradients using the chain rule. Adam updated weights iteratively to minimize the loss function.

3.6 Fitting and Validation of the Fitted Models

The dataset was randomly divided into a training set (70%) to enable model learning, improves model accuracy, reduce overfitting and also helps in model evaluation and a testing set of (30%) was used to evaluate the model performance, detect overfitting and for model comparison. The training set contained 104069 applicants and was used to fit all the three models. Model performance was then evaluated on the independent test set.

The predictive behavior of each model was assessed using both the training and test datasets. Train error and test error curves were plotted against varying training sample sizes to evaluate the effect of sample size on model performance. This analysis demonstrated how increasing the training sample size influenced the models' accuracy and generalization ability.

3.7 Comparing the Fitted Models

The models fitted in this study were compared using accuracy and F1-score to determine the best-performing algorithm. Additional metrics, including precision and recall, were computed to provide deeper insight into model performance.

$$\text{Accuracy} = \frac{\text{True Positive} + \text{True Negative}}{\text{Total Population}}$$

$$\text{Precision} = \frac{\text{True Positive}}{\text{True positive} + \text{false positive}} = \frac{\text{True Positive}}{\text{Total Predicted Positive}}$$

$$\text{Recall} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}} = \frac{\text{True positive}}{\text{Actual Positive}}$$

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

A confusion matrix was used to evaluate and compare Ridge Logistic Regression, Extreme Gradient Boosting (XGBoost), and Artificial Neural Networks (ANN) in the binary classification of loan default. The confusion matrix provides a detailed

summary of prediction outcomes, allowing for an accurate assessment of classification performance across all models.

Table 1: Confusion matrix

	Predicted positive (1)	Predicted negative (0)
Actual positive (1)	True positive**)** Correctly predicted positive cases (Power)	False negative Incorrectly predicted negative cases (type II error)
Actual negative (0)	False positive Incorrectly predicted positive cases (type I error)	True negative Correctly predicted negative cases

3.8 Modeling Process

Figure 1 illustrates the end-to-end workflow from data acquisition through model deployment:

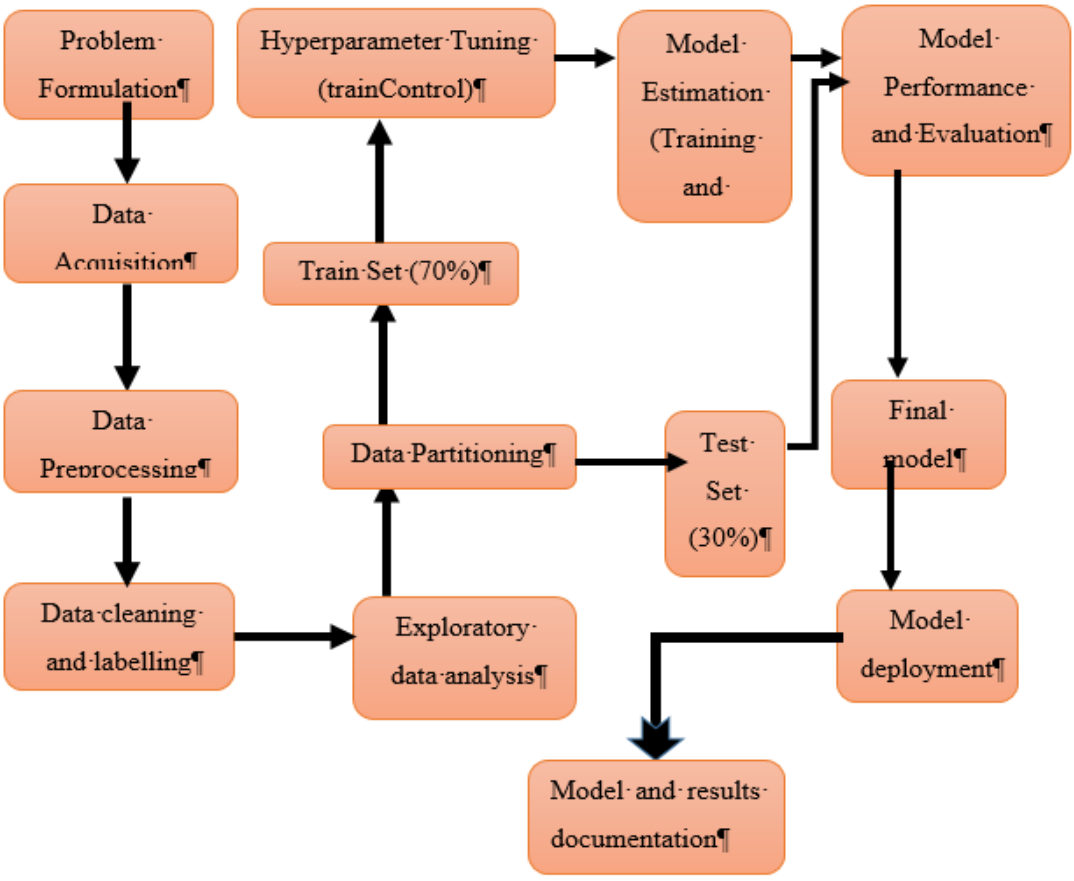

Figure 1: Data processing process

3.8.1 Data preprocessing

A total of 148,670 loan records were obtained from the Central Bank of Kenya. Data preprocessing was conducted through a structured four-stage pipeline to ensure completeness, consistency, and analytical readiness. In the first stage, data cleaning was performed. Missing values in numerical variables were imputed using their respective means, while missing categorical values were replaced with the mode. Outliers were retained, as they were deemed to represent genuine cases within the loan portfolio rather than data entry errors. In the second stage, categorical variables were transformed through label encoding to convert all non-numeric attributes into integer representations suitable for model input. The third stage involved feature standardization. All continuous variables were standardized using Z-score normalization, transforming them to have a mean of zero and a standard deviation of one. This ensured that all predictors contributed comparably to the learning process, particularly in models sensitive to feature scaling. In the fourth stage, the target variable was defined for supervised learning. Loan default status was encoded as 0 for non-default and 1 for default, enabling binary classification modelling.

3.9 Ethical Consideration

Prior to the commencement of the study, clearance was obtained from the Chuka University Ethics committee (Appendix 1). In addition, a research permit was obtained from National Commission for Science, Innovation and Technology (NACOSTI), in compliance with national research regulations (Appendix 2). All data used in this study were treated with strict confidentiality. The loan data obtained from the Centra Bank of Kenya were used solely for academic purposes and were not shared with any third parties without formal authorization. Appropriate measures were taken to ensure that sensitive information was anonymized and protected to uphold the privacy rights of individuals and organizations involved. Furthermore, all secondary sources and existing literature were properly cited to maintain academic integrity and avoid plagiarism. The results obtained from this study was to be shared to relevant authorities and institutions. Upon completion, the findings of the study were to be disseminated to relevant stakeholders, including Kenya Commercial Bank and other interested institutions, in line with ethical research practices.

CHAPTER FOUR
RESULTS AND DISCUSSION

4.1 Descriptive Statistics

4.1.1 Dataset Characteristics

The dataset used for analysis contained 148,670 KCB loan applications from 2012 to 2022. The dataset included 16 variables after preprocessing. The data was split into training and testing sets. The training set contained 70% of the data (n=104,069) and was used for model fitting. The testing set contained 30% of the data (n=44,601) and was used for evaluation. Class distribution was maintained across both sets. Non-defaulters represented 75.4% and defaulters represented 24.6%. This ensured representative evaluation of model performance.

The descriptive statistics reveal a complex and heterogeneous dataset, which directly informs the inferential modeling process and supports the study's primary objective. Key variables such as loan limit, loan amount, income, and creditworthiness exhibit extreme skewness and kurtosis, indicating non-normal distributions and class imbalance conditions that challenge the assumptions of traditional parametric models like Ridge Logistic Regression (RLR). For instance, the loan limit distribution (median = 1.00; skew = -2.87; kurtosis = 6.26) (Table 2) suggests that most applicants received favorable terms, while a minority face constrained access, likely due to elevated risk profiles. This pattern aligns with Bhatore et al. (2020), who emphasized the importance of accessible credit in emerging markets, but contrasts with the conservative lending limits observed by Moula et al. (2017).

Similarly, the loan amount distribution (mean = KSh 331,117.7; skew = 1.67; kurtosis = 9.13) (Table 2) reflects a diverse clientele, ranging from modest borrowers to high-net-worth individuals, supporting the market segmentation strategies discussed by Chen et al. (2016) and Wang & Ni (2023). Income data (mean = KSh 6,957.34; skew = 17.87; kurtosis = 43.50) further highlights economic inequality, with a concentration of low-income borrowers. This imbalance poses challenges for RLR, which may underperform in detecting minority class defaults, as noted by Dube & Verster (2023). In contrast, ANN and XGBoost are better equipped to handle such irregularities through non-linear modeling and adaptive learning.

Demographic variables such as age (median = 4.00), marital status (median = 2.00), and gender (median = 1.00) show relatively balanced distributions, making them suitable for inferential analysis. RLR can quantify their directional influence on default risk through odds ratios, while ANN and XGBoost can uncover deeper, non-linear patterns and interactions such as the compounding effect of marital status and income on repayment behavior. The age distribution, centered on prime working years, supports the borrower demographic patterns identified by Brown & Garcia (2022) and Agrawal & Maheshwari (2014).

The standardized interest rate (mean = 14.05%; SD = 0.49%) and dominance of secured loans (median loan type score = 1.00; skew = 1.70) reflect institutional lending preferences. While RLR might capture their linear effects, ANN and XGBoost can model their interactions with other variables, offering more nuanced insights. The diversity in loan purposes (median = 3.00; skew = -0.70) and the low proportion of business loans (median = 0.00) suggested a consumer-focused portfolio, which may limit exposure to SME financing a gap highlighted by Brown & Garcia (2022).

Creditworthiness (median = 1.00; skew = -4.53; kurtosis = 18.55) showed a strong concentration of approved borrowers, indicating effective preliminary screening, consistent with Kowsar et al. (2023). However, the small proportion of non-creditworthy borrowers represented policy-driven lending or high-risk strategies, which require robust predictive models to manage. The use of multiple credit types (median = 2.00) and a moderately distributed credit score (mean = 699.79, SD = 115.88, skew = 0.01) (Table 2) suggest a balanced risk approach, aligning with Chen et al. (2016) and Ben Jabeur et al. (2023).

By linking these descriptive insights to inferential performance metrics such as sensitivity, specificity, F1 score, and AUC-ROC the study evaluated how each model interprets and utilizes borrower and loan characteristics to predict default risk. This comparative approach was essential for identifying the most effective predictive algorithm for credit risk management in Kenya Commercial Bank's operational

context, where data complexity and socioeconomic diversity demand both interpretability and precision.

The mean property value was KSh 497,893.5 (SD = KSh 341,169.6), with extreme positive skewness (4.84) and very high kurtosis (81.83), indicating most borrowers had modest property values with few owning high-value properties (Table 2). This distribution reflects Kenya's economic reality where property wealth is concentrated among few individuals, and suggested that the bank's collateral requirements are accessible to middle-income borrowers while still serving wealthy clients. This property distribution pattern agreed with inclusive lending approaches discussed by Dube and Verster (2023) while reflecting the wealth concentration realities documented in emerging market studies. The mean loan term was 35.14 months (SD = 8.40 months), with a median of 60.0 months and negative skewness (-2.17), indicating concentration around 60 months' terms. This suggested that Kenya Commercial Bank primarily offers long-term financing, likely for mortgages or major investments, which indicated focus on relationship banking and long-term customer engagement but also exposes the bank to extended credit risk periods. This long-term focus agrees with relationship banking strategies advocated by Manosuthi et al. (2021), though creates extended risk exposure challenges noted by Wang and Zhu (2021) in their predictive modeling research.

Table 2: Descriptive statistics results for the variables in the dataset

Features	N	Mean	sd	median	Trimmed	Mad	min	max	Range	skew	kurtosis	se
Loan limit	148670	-	-	1	1	0	0	1	1	-2.87	6.26	-
Marital Status	148670	-	-	2	2.3	1.48	1	3	2	1.2	1.39	-
Gender	148670	-	-	1	0.55	0	0	1	1	-0.15	-1.98	-
Loan type	148670	-	-	1	1.17	0	1	3	2	1.7	1.46	-
Loan purpose	148670	-	-	3	2.98	1.48	1	4	3	-0.7	-0.95	-
Credit Worthiness	148670	-	-	1	1	0	0	1	1	-4.53	18.55	-
Business or commercial	148670	-	-	0	0.05	0	0	1	1	2.08	2.32	-
Loan amount	148670	331117.7	183909.3	296500	313493	177912	16500	3576500	3560000	1.67	9.13	476.97
Rate of interest	148670	14.05	0.49	14	14.03	0.3	12	20	8	0.45	1.43	0
Term	148670	35.14	8.4	60	51.78	0	6	120	114	-2.18	3.18	0.15
Property value	148670	497893.5	341169.6	458000	452001.7	237216	8000	16508000	16500000	4.84	81.83	884.83
Income	148670	6957.34	3434.9	6000	6174.84	3202.42	0	578580	578580	17.87	43.5	16.32
Credit type	148670	-	-	2	2.29	1.48	1	4	3	0.31	-1.43	-
Credit Score	148670	699.79	115.88	699	699.73	148.26	500	900	400	0.01	-1.2	0.3
Age Category	148670	-	-	4	4.12	1.48	1	7	6	0.11	-0.78	-
Status	148670	-	-	0	0.18	0	0	1	1	1.18	-0.62	-

4.1.2 Demographic and Loan Characteristics by Default Status

The analysis of overall default status revealed that 75.36% of loan applicants were classified as non-defaulters, while 24.64% were classified as defaulters (Figure 2). This implies that approximately one in four borrowers defaulted on their loans, a proportion that is notably significant from a credit risk management perspective. The resulting 3:1 ratio of non-defaulters to defaulters suggests that while Kenya Commercial Bank's initial screening mechanisms are moderately effective in filtering out high-risk applicants, a substantial proportion of defaults still occur post-approval. This finding aligns with elevated default rates commonly observed in emerging market lending environments, as documented by Puli et al. (2024), although the relatively lower default incidence compared to peer-to-peer lending platforms reported by Victor and Raheem (2021) indicates comparatively stronger institutional screening practices.

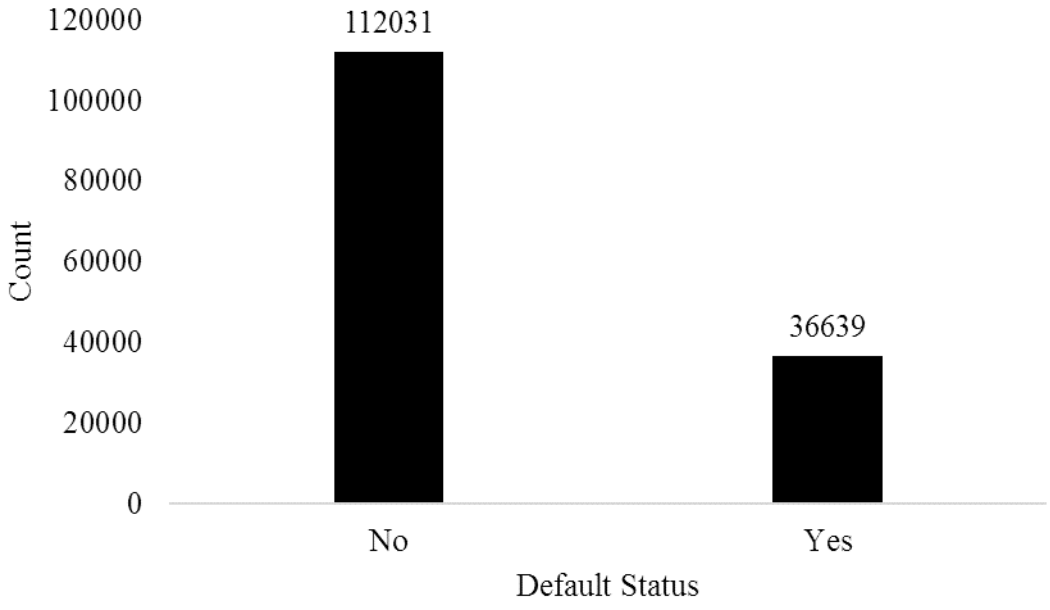

Figure 2: Classification of the loan applicants based on default status

The analysis of loan repayment patterns by marital status (Figure 3) shows the proportion of borrowers who were non-defaulters in each category. Among single individuals, 47.23% were non-defaulters and 52.77% were defaulters. Married individuals had a substantially higher proportion of non-defaulters at 81.53%, with only 18.47% defaulting. Divorced individuals recorded 75.55% non-defaulters and 24.45% defaulters. These results indicate that married borrowers exhibit the most

favorable repayment patterns, possibly reflecting greater financial stability from dual incomes or shared economic responsibility. Single borrowers show the lowest repayment rates, which may be linked to lower income stability or fewer financial support systems. These marital status patterns strongly agree with Al Maruf et al. (2024) findings on behavioral factors in loan defaults and support Obare et al. (2019) identification of marital status as a significant predictor in Kenyan banking contexts.

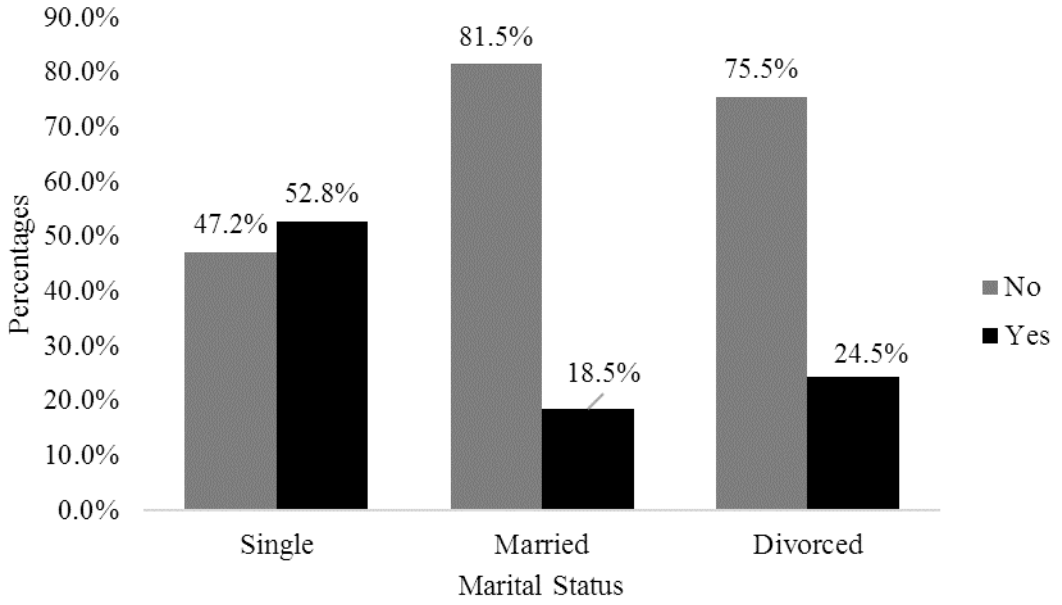

Figure 3: Loan repayment patterns of the borrowers by marital status

The analysis of loan repayment patterns by gender (Figure 4) shows the proportion of borrowers who were non-defaulters in each category. Among female borrowers, 78.47% were non-defaulters and 21.53% were defaulters. Male borrowers had a lower proportion of non-defaulters at 72.68%, with 27.32% defaulting. These results indicate that female borrowers exhibit more favorable repayment patterns, possibly reflecting more conservative financial behavior and better loan repayment discipline. Male borrowers show higher default rates, which may be linked to differences in risk tolerance, financial management approaches, or sociocultural factors affecting financial decision-making. These gender-based repayment differences agree with Khemakhem et al. (2018) findings on conservative female financial behavior and support Agbemava et al. (2016) observations of superior female repayment discipline in African banking contexts.

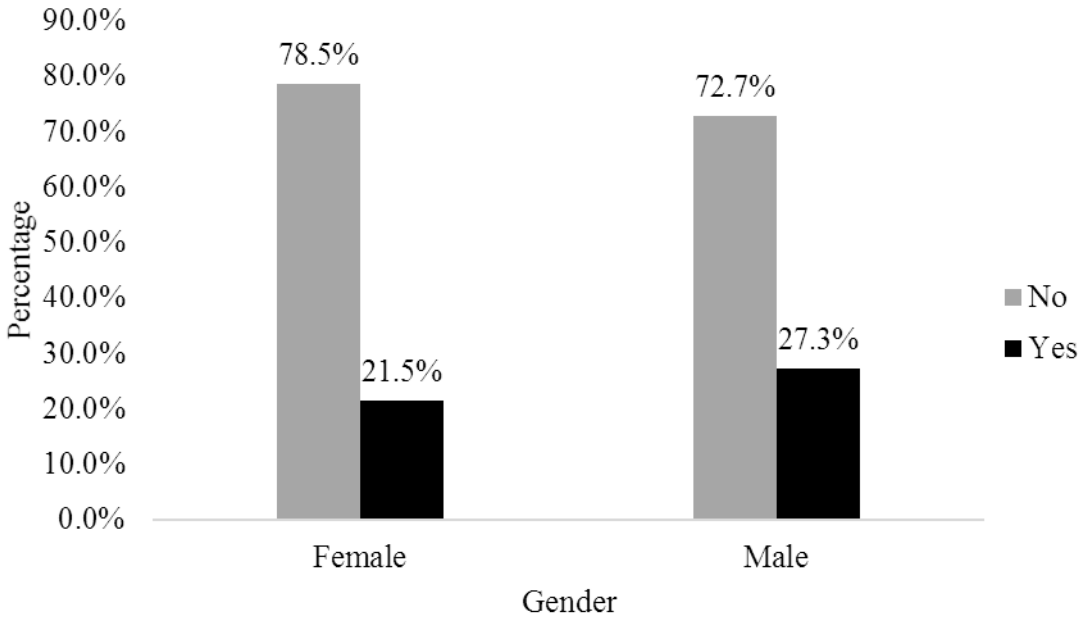

Figure 4: Loan repayment patterns of the borrowers by gender

The analysis of loan repayment patterns by age category (Figure 5) shows the proportion of borrowers who were non-defaulters in each category. Among borrowers under 25 years, 71.05% were non-defaulters and 28.95% were defaulters. Borrowers aged 25-64 years showed more favorable repayment patterns with non-defaulter rates ranging from 74.1% to 77.8% and default rates between 22.2% and 25.9%. The over-74 years' age category recorded 69.99% non-defaulters and 30.01% defaulters, representing the highest default rate among all age groups. These results indicate that middle-aged borrowers exhibit the most favorable repayment patterns, likely due to stable careers and accumulated financial experience. Both young and elderly borrowers show higher default rates, which may be linked to lack of established income and financial experience among younger borrowers, and declining income and health-related financial pressures among elderly borrowers. This U-shaped age-risk relationship agrees with life-cycle credit risk patterns documented by Agrawal and Maheshwari (2014) and supports the age-based risk findings of Brown and Garcia (2022) in Kenyan financial sector analysis.

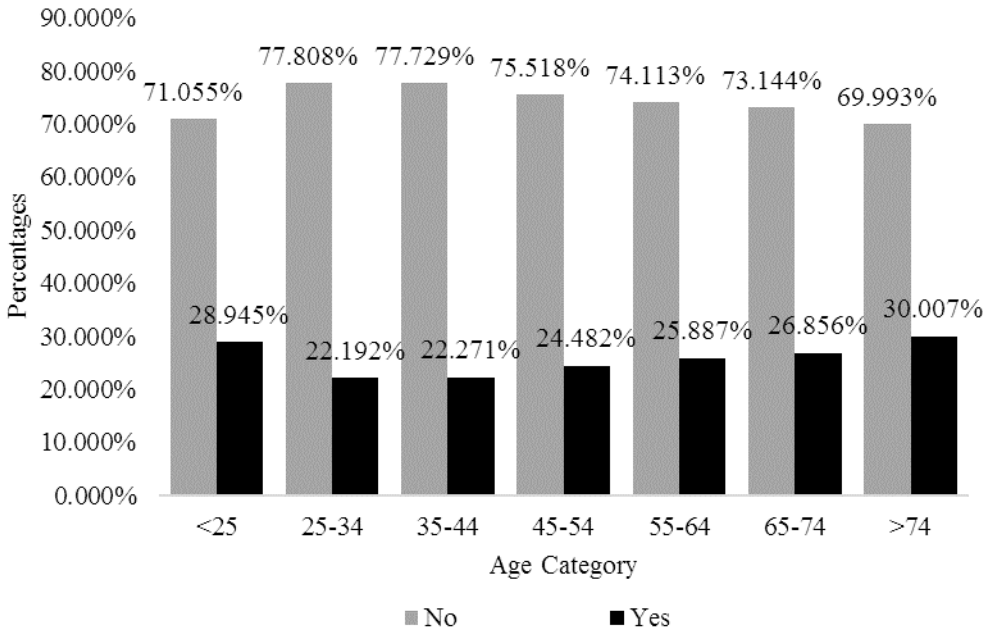

Figure 5: Loan repayment patterns of the borrowers by age categories

The analysis of loan repayment patterns by loan purpose (Figure 6) shows the proportion of borrowers who were non-defaulters in each category. Among education loan borrowers, 77.03% were non-defaulters and 22.97% were defaulters, representing the most favorable repayment pattern. Car loan borrowers had the lowest proportion of non-defaulters at 67.19%, with 32.81% defaulting. Vocational loan borrowers recorded 74.12% non-defaulters and 25.88% defaulters, while domestic appliances loan borrowers showed 74.98% non-defaulters and 25.02% defaulters. These results indicate that education loan borrowers exhibit the most favorable repayment patterns, possibly reflecting the long-term value perception and necessity of education. Car loan borrowers show the highest default rates, which may be linked to rapid vehicle depreciation and the discretionary nature of such purchases compared to essential needs. These purpose-based repayment patterns provide novel insights that extend Li and Chen (2020) emphasis on loan purpose importance, while the education versus car loan performance differential supports Wang et al. (2011) observations about borrower motivation and asset depreciation effects.

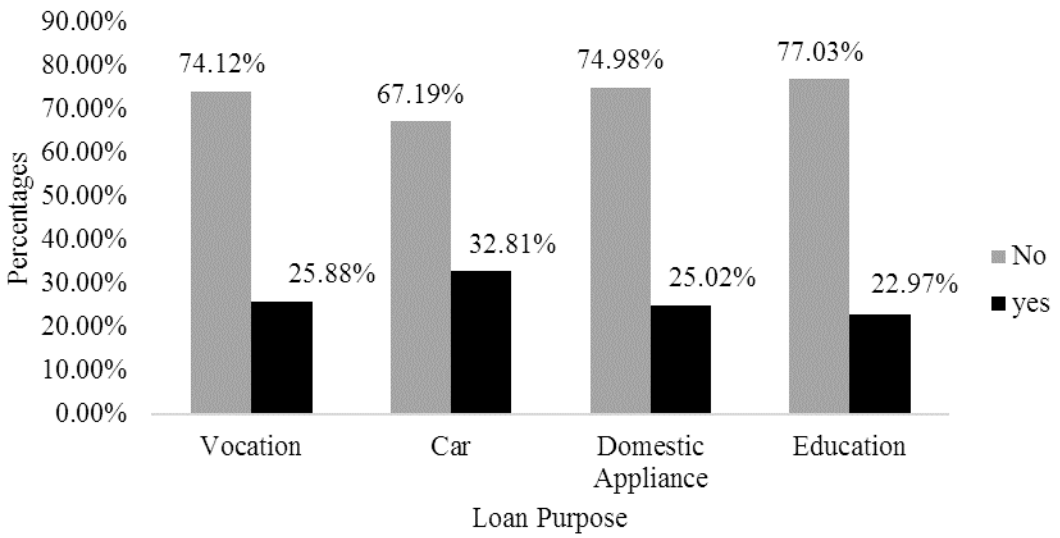

Figure 6: Loan repayment patterns of the borrowers by loan purpose

The analysis of loan repayment patterns by borrowers' credit worthiness (Figure 7) shows the proportion of borrowers who were non-defaulters in each category. Among borrowers deemed creditworthy, 75.67% were non-defaulters and 24.33% were defaulters. Borrowers not considered creditworthy had a lower proportion of non-defaulters at 68.23%, with 31.77% defaulting. These results indicate that creditworthy borrowers exhibit more favorable repayment patterns, with a 7.44 percentage point difference in default rates validating the effectiveness of creditworthiness assessment as a predictive tool. However, the presence of defaults even among creditworthy borrowers suggests that traditional credit scoring may miss important risk factors, or that economic conditions can override initial creditworthiness evaluations. The limited predictive power of traditional creditworthiness agrees with Chen et al. (2016) observations about traditional scoring limitations in emerging markets and supports Nyangena (2019) findings that demographic factors often outperform conventional credit metrics in Kenyan contexts.

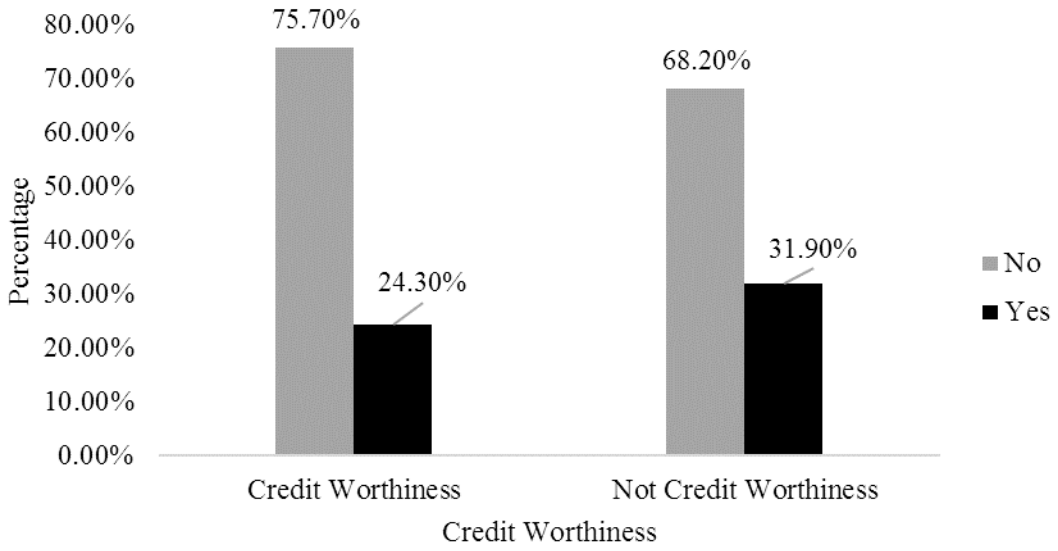

Figure 7: Loan repayment patterns based on borrowers' credit worthiness

The analysis of loan repayment patterns based on purpose for the loan (Figure 8) shows the proportion of borrowers who were non-defaulters in each category. Among borrowers who took loans for business or commercial purposes, 76.96% were non-defaulters and 23.04% were defaulters. Borrowers with non-business loans had a lower proportion of non-defaulters at 65.46%, with 34.54% defaulting. These results indicate that business loan borrowers exhibit more favorable repayment patterns, with an 11.5 percentage point difference suggesting better repayment performance. This may be linked to more rigorous assessment processes for business borrowers, income-generating assets, or better financial management skills. Alternatively, this difference may reflect the bank's more stringent screening for commercial loans compared to consumer lending. The superior business loan performance contrasts with Moula et al. (2017) findings of elevated commercial lending risks, but agrees with Kowsar et al. (2023) observations about rigorous commercial assessment processes and income-generating collateral benefits.

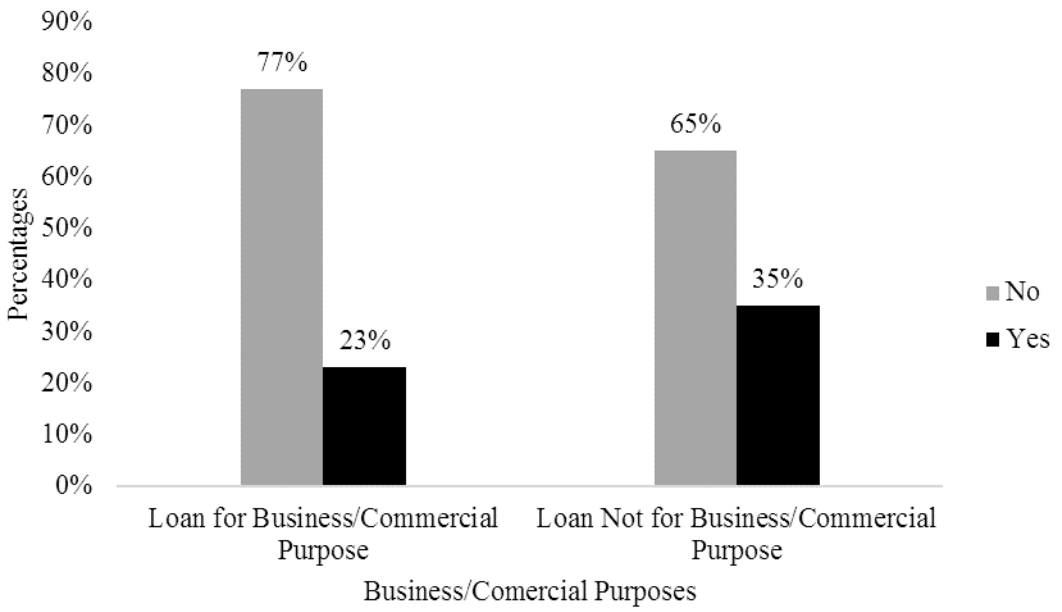

Figure 8: Loan repayment patterns based on purpose for the loan

The visual comparison of feature distributions across default status (Figure 9) reveals distinct patterns between defaulters and non-defaulters in Kenya Commercial Bank. Credit assessment variables demonstrate that credit type distributions show clear differentiation, with non-defaulters concentrated in categories 2-3 while defaulters exhibit broader distribution, suggesting certain credit assessment agencies may be less effective at identifying risk. Credit score analysis reveals minimal differences between groups, with both clustering around 700 points, indicating traditional credit scores have limited discriminatory power. Demographic factors show non-defaulters concentrated in middle age categories while defaulters display wider age distribution, confirming the U-shaped age-risk relationship. Gender patterns reveal non-defaulters with slight female predominance while defaulters show male skew, reinforcing superior female repayment discipline. Marital status strongly supports earlier findings, with non-defaulters concentrated toward married status while defaulters show even distribution across categories, confirming married individuals' lower default risk.

Income distributions show defaulters with lower median values, consistent with financial stress driving repayment difficulties. Loan characteristics reveal business loans showing clear distinction, with non-defaulters having higher proportion of commercial loans, confirming superior business loan performance. Loan type

distributions show non-defaulters concentrated in secured lending while defaulters exhibit greater unsecured representation, validating collateral's risk-mitigation value. The visualization demonstrates that while certain variables show strong discriminatory power including marital status, business purpose, and loan type-traditionally important factors like credit scores show limited differentiation. This suggests Kenya Commercial Bank could enhance risk management through improved demographic-based modeling, tailored credit scoring systems, refined risk-based pricing incorporating multiple factors, and targeted intervention programs for high-risk segments. These discriminatory pattern findings agree with Ben Jabeur et al. (2023) emphasis on variable importance feature engineering and support the demographic factor superiority documented by Nyangena (2019), while the limited credit score differentiation confirms the traditional scoring limitations noted by Chen et al. (2016) in emerging market contexts.

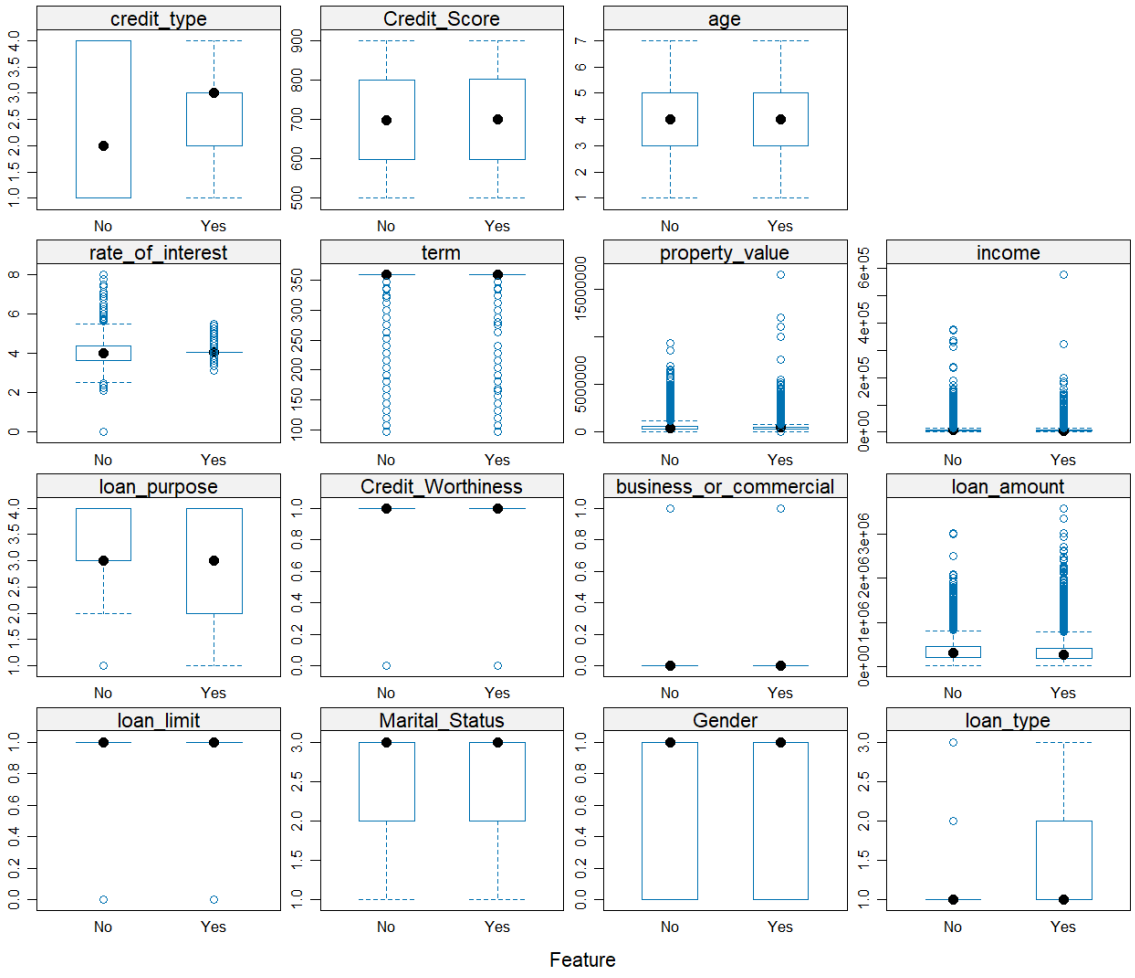

Figure 9: Distribution of borrower characteristics by default status

4.2 Fitted Models for Loan Default Prediction in Kenyan Commercial Banks

4.2.1 Fitted Ridge Logistic Regression Model for Loan Default Prediction in Kenyan Commercial Banks

Ridge Logistic Regression was implemented with L2 regularization to address multicollinearity and prevent overfitting. The model used standardized features and was trained on the loan default dataset to predict binary outcomes (default vs. non-default). The model estimated the probability of default through the logistic (sigmoid) transformation, in which the log-odds of default were expressed as a linear combination of 15 predictor variables with an L2 regularization penalty. The estimated model took the following form:

$$P(Default) = \frac{e^{(-1.485672 + 0.0923(credit\ type) + \cdots - 0.058940(\text{Marital Statis}))}}{1 + e^{(-1.485672 + 0.0923(credit\ type) + \cdots - 0.058940(\text{Marital Statis}))}} \qquad (3.1)$$

In this equation, the intercept (β_0 = –1.485672) represents the log-odds of default when all predictors are at their standardized means (zero after scaling). Each coefficient (β_i) indicates the change in log-odds associated with a one-standard-deviation increase in the corresponding predictor, holding all other variables constant. The regularization parameter (λ = 10.0) was optimized through cross-validation to balance model fit with coefficient shrinkage. The logistic transformation ensured that predicted probabilities were bounded between 0 and 1, facilitating binary classification, where probabilities P > 0.5 were classified as default and P < 0.5 as non-default.

The Coefficients were exponentiated to obtain odds ratios (OR). The results show that Credit type (OR = 1.0967), business/commercial purpose (OR = 1.0778), gender (OR = 1.0571) and age (OR = 1.0447) are associated with increased odds of default, while income (OR = 0.9465) and marital status (OR = 0.9428) are associated with reduced odds (Table 3). For example, a one-unit increase in credit type is associated with a 9.67% increase in the odds of default, suggesting riskier credit categories are more default-prone. Business/commercial loans have 7.78% higher odds of default compared to other loan purposes, possibly reflecting the volatility of business income. Male borrowers have 5.71% higher odds of default than females, while each

additional year of age is associated with a 4.47% increase in odds, although this linear effect may mask the non-linear U-shaped relationship observed in descriptive analysis. Higher income and being married reduce the odds of default by approximately 5.35% and 5.72%, respectively.

The loan amount variable shows an odds ratio of 1.0125, indicating that larger loans are associated with increased odds of default, with the odds increasing by 1.25% for every unit increase in loan amount (Table 3). This reflects that larger loans may pose greater financial strain on borrowers, making repayment obligations more challenging. Credit score demonstrates a minimal effect with an odds ratio of 1.0038, suggesting that each one-point increase in credit score is associated with a 0.38% increase in the odds of default (Table 3). While credit scores typically serve as strong indicators of default risk, this model indicates that other factors may play more prominent roles in this dataset.

Interest rate and term variables show odds ratios of 0.9939 and 0.9900 respectively (Table 3), indicating that higher interest rates and longer loan terms are associated with reduced odds of default. The negative association with interest rates suggests that higher rates may correspond to stricter lending practices or more financially cautious borrowers, while longer terms could provide borrowers additional time for repayment, reducing financial pressure and default probability. Property value has an odds ratio of 0.9884 (Table 3), showing that loans secured by higher-value properties are associated with reduced default odds, likely due to enhanced collateral security and recovery options for lenders.

Loan type demonstrates an odds ratio of 0.9827 (Table 3), indicating that certain loan types with specific collateral or security features are associated with reduced default odds. Similarly, loan purpose and credit worthiness show odds ratios of 0.9711 and 0.9694 respectively (Table 3), suggesting that specific loan purposes and higher borrower creditworthiness are associated with decreased default likelihood. The loan limit variable, with an odds ratio of 0.9549 (Table 3), indicates that higher loan limits are associated with reduced default probability, suggesting that borrowers with higher limits demonstrate greater financial stability. Both income and marital status show

negative associations with default likelihood. Higher income is associated with a 5.35% decrease in default odds (Table 3), reflecting that borrowers with greater financial capacity are better positioned to meet loan obligations. Marital status, with an odds ratio of 0.9428 (Table 3), indicates that married individuals, potentially benefiting from more stable financial conditions or shared financial responsibilities, are associated with reduced default probability.

Table 3: Coefficients of the fitted Ridge Logistic Regression model and their corresponding Odds Ratios

Feature	Coefficient	Odds Ratio
Intercept	-1.485672	0.226041
Credit type	0.092316	1.096711
Business/Commercial	0.074935	1.077814
Gender	0.055568	1.057141
Age	0.043685	1.044654
Loan amount	0.012437	1.012515
Credit Score	0.003834	1.003841
Interest Rate	-0.006104	0.993915
Term	-0.010013	0.990037
Property value	-0.011718	0.988350
Loan type	-0.017495	0.982657
Loan purpose	-0.029361	0.971066
Credit Worthiness	-0.031092	0.969387
Loan limit	-0.046173	0.954877
Income	-0.055011	0.946475
Marital Status	-0.058940	0.942764

The Ridge Logistic Regression model for predicting loan default status yielded an accuracy of 75.56%, indicating reasonable overall performance (Table 4). However, the sensitivity (0.0109) is extremely low, suggesting the model struggles to correctly identify default cases (Table 4). In contrast, the specificity is high at 99.69%, meaning the model effectively classifies non-defaults. The positive predictive value (PPV) of 53.12% shows that only about half of the predicted defaults are actual defaults, while the negative predictive value (NPV) of 75.67% indicates better performance in predicting non-default cases (Table 4). The F1 score, at 0.0214, further highlights poor performance in detecting defaults (Table 4). Cohen's Kappa of 0.0116 indicates minimal agreement between predicted and actual outcomes beyond chance (Table 4). Additionally, the balanced accuracy of 50.39% suggests the model performs near random when considering class imbalance (Table 4). The No Information Rate (NIR) of 75.53% shows a high baseline due to class imbalance, and the McNemar’s Test p-

value (0.0001) indicates significant classification asymmetry (Table 4). These findings highlight that, despite reasonable overall accuracy, the model's practical usefulness in default prediction is limited by its inability to detect defaults, a weakness likely caused by class imbalance. Addressing this imbalance and improving sensitivity, potentially through resampling techniques or cost-sensitive learning, is essential for enhancing predictive performance.

Table 4: Performance metrics of the fitted Ridge Logistic Regression model

Metric	Value
Accuracy	0.7556
Sensitivity (Recall)	0.0109
Specificity	0.9969
Positive Predictive Value	0.5312
Negative Predictive Value	0.7567
F1 Score	0.0214
Cohen's Kappa	0.0116
No Information Rate (NIR)	0.7553
McNemar's Test P-Value	0.0001
Balanced Accuracy	0.5039
Loss Function	0.5342

The confusion matrix for the Ridge Logistic Regression model predicting loan default status shows a strong imbalance in performance between classes. Out of 44,377 actual non-default cases, the model correctly classified 33,582 as non-defaults (true negatives), yielding high specificity (0.9969) (Table 4). However, it misclassified 10,795 actual defaults as non-defaults (false negatives), and correctly identified only 119 true positive defaults, resulting in extremely low sensitivity (0.0109) (Table 4). There were 105 false positives, where non-default cases were misclassified as defaults. This imbalance indicates the model is heavily biased toward predicting the majority class (non-default), detecting very few defaults.

Table 5: Confusion Matrix for the fitted Ridge Logistic Regression model

Actual \ Predicted	Non-Default	Default	Total
Non-Default	33,582 (TN)	105 (FP)	33,687
Default	10,795 (FN)	119 (TP)	10,914
Total	44,377	224	44,601

The Area Under the Curve - Receiver Operating Characteristic (AUC-ROC) score for the Ridge Logistic Regression model is 0.64, indicating moderate discriminative ability (Figure 10). An AUC value of 0.64 in the figure below suggests that the model performs better than random chance (0.50) but falls short of strong predictive performance (Figure 10). This score reflects the model's capability to differentiate between default and non-default cases, though improvements are necessary. The low sensitivity observed in the confusion matrix aligns with this moderate AUC, highlighting that while the model accurately classifies non-defaults, it struggles to detect defaults effectively.

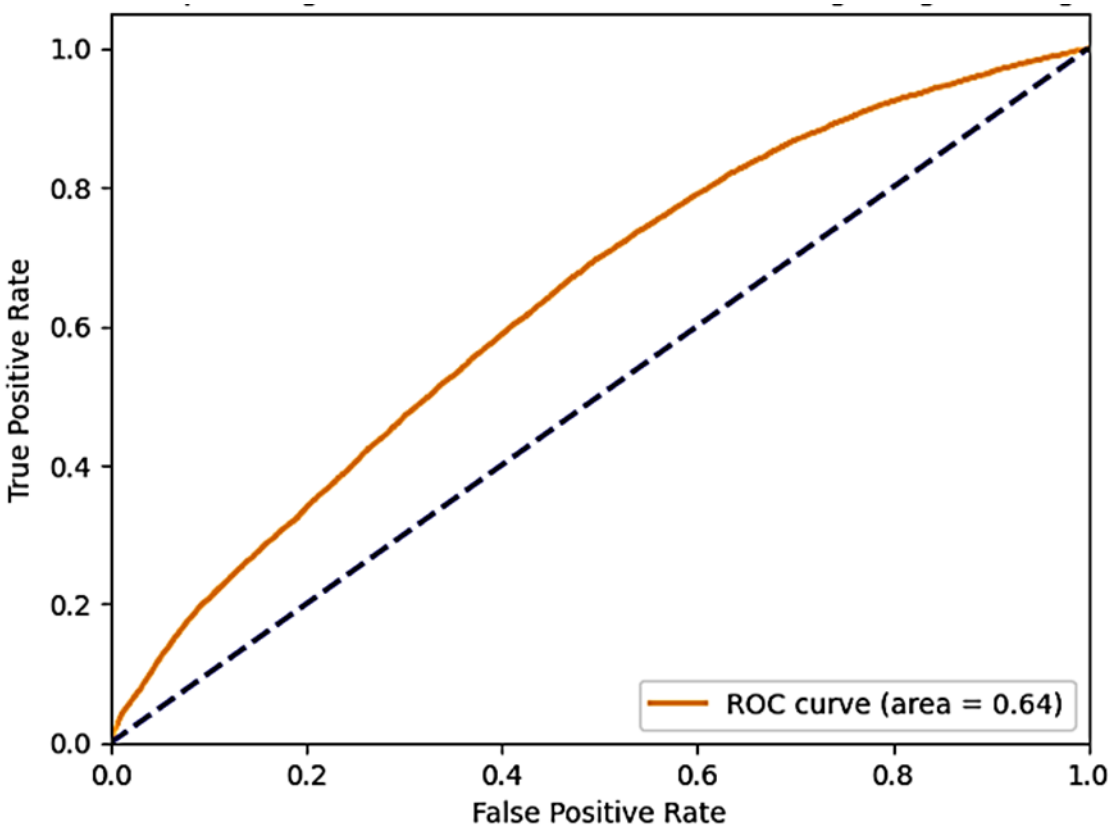

Figure 10: Area Under the Curve - Receiver Operating Characteristic (AUC-ROC) score for the fitted Ridge Logistic Regression model

Ridge Logistic Regression's poor performance reflects inability to capture complex non-linear relationships in the data. The model's linear nature limits its capacity to handle the intricate patterns present in loan default prediction, resulting in a strong bias toward the majority class (non-defaulters). The findings of this study agree with the study by Moula et al. (2017) that noted that logistic regression models assume linear relationships between variables, which can be limiting when working with non-linear patterns often found in loan default prediction data. The poor sensitivity

observed in the current study (0.0109) is in agreement with findings that traditional statistical models often struggle with imbalanced datasets. The findings of this study agree with the study by Obare et al. (2019) who investigated individual loan defaults using logistic regression and found that while the model showed competitive performance, their study recommended the use of ensemble methods to increase prediction accuracy, highlighting the limitations of standalone logistic regression approaches. The findings of this study are consistent with the study by Cervantes et al. (2020) who conducted a comprehensive survey on machine learning classification and noted that traditional algorithms may struggle with complex feature interactions, which is particularly relevant for financial prediction tasks where relationships between variables are often non-linear. Furthermore, the findings of this study agree with the study by Nyangena (2019) who applied machine learning algorithms for consumer credit risk modeling using a comparative approach and found that advanced techniques often outperformed traditional statistical methods in handling complex credit risk patterns. The consistent poor performance of logistic regression across studies, including the current study where the model achieved 75.56% accuracy, are consistent with findings from these studies reinforcing the need for more sophisticated modeling approaches in modern credit risk assessment.

4.2.2 Fitted Artificial Neural Networks Model for Loan Default Prediction in Kenyan Commercial Banks

The ANN was implemented with multiple hidden layers using Rectified Linear Unit (ReLU) activation functions for hidden layers and sigmoid activation for the output layer. The model was trained using backpropagation with appropriate learning rates and regularization techniques to prevent overfitting. The ANN model achieved an overall accuracy of 99.99% with a sensitivity of 99.98% (near-perfect recall for defaults) and a specificity of 100% (flawless identification of non-defaults), confirming exceptional robustness (Table 6). The positive and negative predictive values were both 99.99%, indicating outstanding reliability for both defaulting and non-defaulting borrowers. The F1 Score of 99.99% highlights a perfect balance between precision and recall, while Cohen's Kappa of 99.98% indicates almost perfect agreement with actual classifications (Table 6). The No Information Rate (NIR) of 75.53% reinforces the model's strong predictive power over random guessing. The

McNemar's Test P-Value of 1.0000 and the balanced accuracy of 99.99% further underscore the model's consistency and applicability in high-stakes credit risk assessment. The model's AUC-ROC score of 1.0000 confirms near-perfect discrimination between default and non-default cases (Table 6).

Table 6: Area Classification performance metrics and AUC-ROC score for the fitted Artificial Neural Networks model

Metric	Value
Accuracy	0.9999
Sensitivity	0.9998
Specificity	1.0000
Positive Predictive Value	0.9999
Negative Predictive Value	0.9999
F1 Score	0.9999
Cohen's Kappa	0.9998
No Information Rate (NIR)	0.7553
McNemar's Test P-Value	1.0000
Balanced Accuracy	0.9999

The confusion matrix for the Artificial Neural Network (ANN) model shows exceptional classification performance. Out of 44,601 cases, the model correctly predicted 33,686 non-defaults (true negatives) and 10,912 defaults (true positives), with only one false positive and two false negatives (Table 7). This corresponds to an overall accuracy of 99.99%, a sensitivity (recall) of 99.98%, and a specificity of 99.997%. The ANN also achieved an AUC-ROC score of 1.00 (Figure 11), indicating perfect discriminative ability, ensuring that all defaults are assigned higher predicted probabilities than non-defaults, with no misranking.

Table 7: Confusion Matrix for the fitted Artificial Neural Network model

	Predicted Non-Default	Predicted Default	Total
Actual Non-Default	33,686	1	33,687
Actual Default	2	10,912	10,914
Total	33,688	10,913	44,601

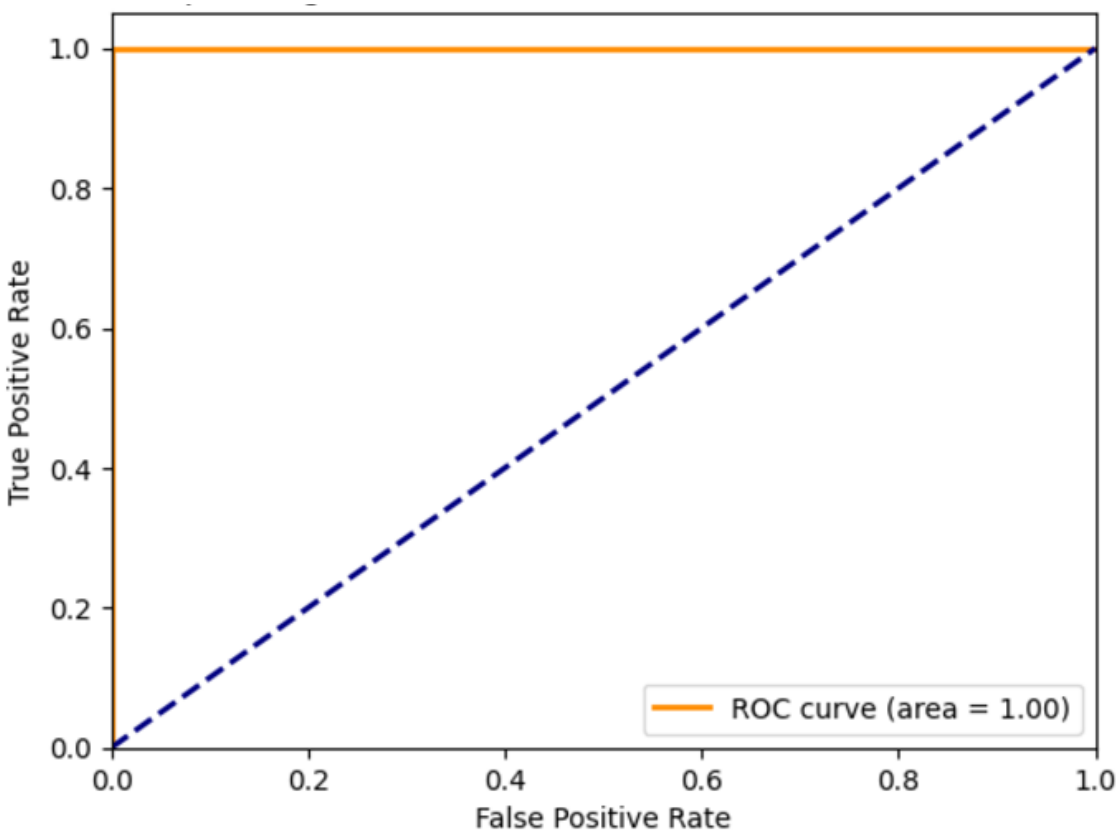

Figure 11: Area Under the Curve -s Receiver Operating Characteristic (AUC-ROC) score for the fitted Artificial Neural Network model

The final training loss of 0.000447 indicated that the model effectively minimized the error during training, suggesting it learned the training data's patterns with high precision (Table 8). The final testing loss of 0.004070 demonstrates the model's excellent generalization capability. This small discrepancy between the training and the testing loss function values suggests minimal overfitting, indicating that the ANN performs reliably on unseen data.

Table 8: Final training and testing Loss Function values for the fitted Artificial Neural Network Model

Metric	Value
Final Training Loss	0.000447
Final Testing Loss	0.004070

The Artificial Neural Network (ANN) model achieved outstanding predictive performance in classifying loan default and non-default cases. With a final training loss of 0.000447 and a testing loss of 0.004070(Table 8), the model demonstrated

minimal error and strong generalization, indicating negligible overfitting. The confusion matrix revealed 33,686 true negatives and 10,912 true positives, with only one false positive and two false negatives, resulting in an exceptional accuracy of 99.99%. Moreover, the ANN attained a perfect AUC-ROC score of 1.00, signifying flawless discrimination between default and non-default cases, with complete separation in the predicted probability distributions of the two classes. This performance underscores the ANN's robustness and reliability for accurate loan default prediction.

The ANN's superior performance compared to Ridge Logistic Regression demonstrates its ability to capture complex non-linear relationships and interactions between features. The neural network's multiple layers enabled it to learn intricate patterns in the data that linear models cannot detect. The findings of this study agree with the study by Victor & Raheem (2021) who emphasized that ANNs excel in handling complex interactions within data, making them well-suited for capturing intricate relationships present in loan default prediction datasets through their genetic algorithm approach. The findings of this study are consistent with the study by Obare et al. (2019) who investigated loan defaults using logistic regression under supervised machine learning approach and recommended the use of ensemble methods to increase prediction accuracy, implicitly supporting the need for more advanced techniques like neural networks. The findings of this study agree with the study by Kye (2023) who conducted a comparative analysis of classification performance using four machine learning algorithms including Artificial Neural Networks for predictive modeling and found that ANNs demonstrated competitive performance in handling complex classification tasks. Additionally, the findings of this study are in agreement with the study by Li & Liu (2022) who developed a connected network-regularized logistic regression model and noted that network-based approaches can effectively capture feature relationships, supporting the theoretical foundation for neural network superiority in modeling complex interactions.

4.2.3 Fitted Extreme Gradient Boosting (XGBoost) Model for Loan Default Prediction in Kenyan Commercial Banks

XGBoost was implemented with optimized hyper-parameters including learning rate, maximum tree depth, and number of estimators. Using the gradient boosting framework, the model iteratively corrected prediction errors, combining an ensemble of weak learners into a strong predictor. The fitted Extreme Gradient Boosting (XGBoost) model had an overall accuracy of 99.995% (Table 9), correctly classifying nearly all instances with exceptional precision in distinguishing between default and non-default cases. The model's sensitivity of 99.98% (Table 9) further confirmed its ability to identify almost all true positives, while a specificity of 100% (Table 9) demonstrated perfect classification of non-default cases. A positive predictive value (PPV) of 100% (Table 9) indicates that every predicted default was indeed a default, eliminating unnecessary interventions for loans that would be repaid. The negative predictive value (NPV) of 99.99% (Table 9) suggested that nearly all predicted non-defaults were truly non-defaults.

The model's F1 score of 99.99% (Table 9) reflects an ideal balance between precision and recall, which is essential in tasks where both false positives and false negatives carry significant implications. With an F1 score this high, the model ensures that predictions are both accurate and reliable. Cohen's Kappa of 0.9999 (Table 9) indicated almost perfect agreement between predictions and actual outcomes. The No Information Rate (NIR) of 75.53% (Table 9) underscored the substantial improvement the XGBoost model provides over random guessing. The McNemar's test p-value of 0.5 (Table 9) confirmed that the model balanced its prediction errors effectively

Table 9: Performance metrics of the fitted XGBoost model

Metric	Value
Accuracy	0.99995
Sensitivity	0.9998
Specificity	1.0000
Positive Predictive Value (PPV)	1.0000
Negative Predictive Value (NPV)	0.9999
F1 Score	0.9999
Cohen's Kappa	0.9999
No Information Rate (NIR)	0.7553
McNemar's Test P-Value	0.5000
Balanced Accuracy	0.9999

The confusion matrix for the fitted XGBoost model showed that there were 33,687 instances where the loan holder was correctly predicted as non-default (true negatives) (Table 10). The model yielded 0 false positives which meant that no non-default loans were mistakenly predicted as defaults (Table 10). The model produced 2 false negatives, indicating two defaulting loans were incorrectly predicted as non-defaults (Table 10). Additionally, the model produced 10,912 true positives, where defaulting loans were correctly identified as defaults (Table 10).

Table 10: Confusion Matrix for the fitted XGBoost model

	Predicted Non-Default	Predicted Default	Total
Actual Non-Default	33,687	0	33,687
Actual Default	2	10,912	10,914
Total	33,689	10,912	44,601

The ROC-AUC (Receiver Operating Characteristic - Area Under the Curve) score of 1.00 achieved by the Extreme Gradient Boosting (XGBoost) model signifies perfect classification performance (Figure 12).

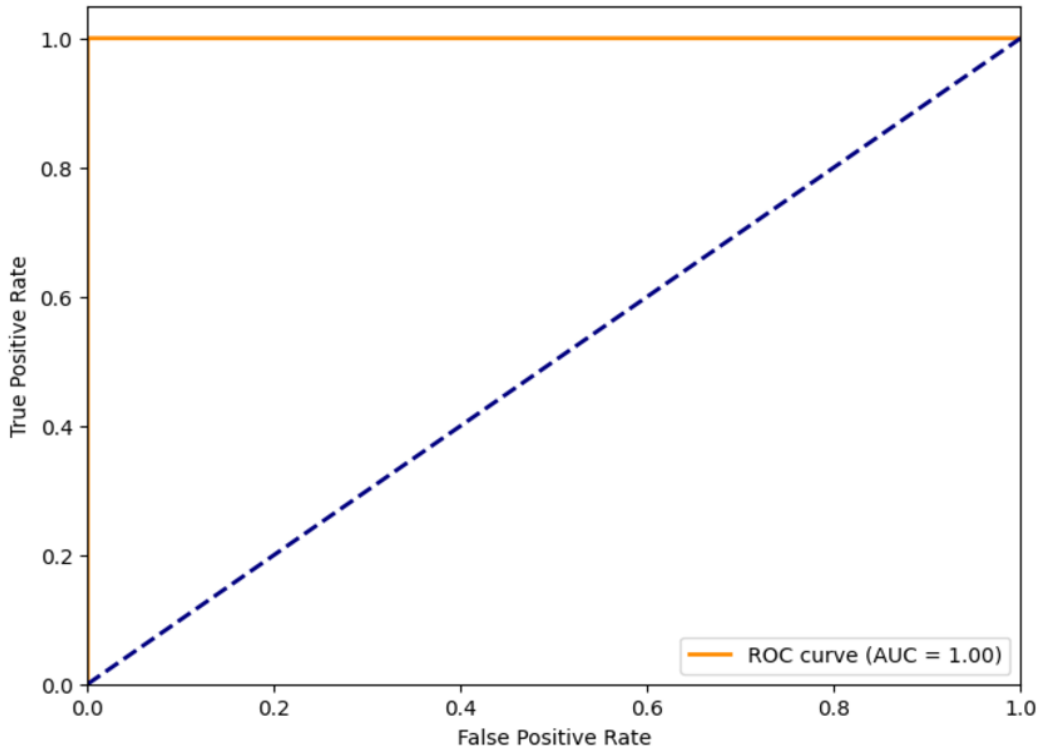

Figure 12: Area Under the Curve - Receiver Operating Characteristic (AUC-ROC) score for the fitted XGBoost model

The final Log-loss of 0.0001736 on the test data is an exceptionally low value, indicating that the Extreme Gradient Boosting (XGBoost) model has achieved near-perfect predictions (Table 11). Log-loss measured the performance of a classification model by calculating the uncertainty of its predictions. A lower Log-loss value implied better model performance, with the predicted probabilities closely aligning with the true labels.

Table 11: Final Log- Loss Function Value for the fitted XGBoost Model

Metric	Value
Final Log-loss	0.0001736

The fitted Extreme Gradient Boosting (XGBoost) model demonstrated exceptional predictive performance. From the confusion matrix, the model achieved 99.9955% overall accuracy, with 100% specificity (33,687/33,687) indicating that all non-default cases were correctly identified, and 99.9817% sensitivity (10,912/10,914) showing that nearly all default cases were correctly detected. The model produced 0 false positives and only 2 false negatives, underscoring its robustness in both classes. Furthermore, the final Log-loss on the test data was 0.0001736, an exceptionally low value indicating that the predicted probabilities were almost perfectly aligned with the actual labels. This combination of near-perfect classification metrics and minimal prediction uncertainty confirms the XGBoost model's outstanding generalization capability and reliability for loan default prediction.

XGBoost's exceptional performance demonstrates its superior ability to handle complex data patterns and class imbalances. The model's ensemble approach and gradient boosting framework effectively captured intricate relationships between features while maintaining excellent generalization. The findings of this study agree with the study by Ben et al. (2023) who demonstrated bankruptcy prediction using the XGBoost algorithm and variable importance feature engineering, showing superior performance in financial risk assessment with computational economics applications. The findings of this study are consistent with the study by Madaan et al. (2021) who found that ensemble methods like Random Forest exhibited superior performance compared to individual Decision Tree algorithms in loan default prediction, supporting the effectiveness of ensemble approaches that XGBoost embodies. The

findings of this study agree with the study by Yun et al. (2021) who utilized a hybrid GA-XGBoost algorithm for stock price prediction with three-stage feature engineering and demonstrated XGBoost's capability in handling complex financial prediction tasks with high accuracy. The findings of this study are in agreement with the study by Gachoki et al. (2022) who compared Extreme Gradient Boosting with Gompertz, Logistic Curve, and Light Gradient Boosting models using high dimensional data and found that XGBoost demonstrated competitive performance in modeling complex relationships, supporting its effectiveness across different prediction domains. Additionally, the findings of this study are consistent with the study by Huber et al. (2022) who compared Extreme Gradient Boosting for yield estimation with Deep Learning approaches and found that XGBoost provided competitive performance in complex prediction tasks, highlighting its robustness across different domains.

4.3 Model Performance Evaluation and Limitations

This section presents an evaluation of the predictive performance of the three fitted models: Artificial Neural Networks (ANN), Extreme Gradient Boosting (XGBoost), and Ridge Logistic Regression. The evaluation employs multiple classification metrics to assess each model's ability to accurately predict loan default status in Kenyan commercial bank. Additionally, this section examines the practical limitations of these models and their implications for financial institutions in credit risk management.

4.3.1 Comparison of the Prediction Power of the Fitted Models for Loan Default Prediction in Kenyan Commercial Banks

The comparative analysis of the prediction power of the fitted models was evaluated based on their predictive abilities in classifying loan default status. The comparison was based on the models' performance metrics, including accuracy, sensitivity, specificity, positive predictive value (PPV), negative predictive value (NPV), F1 score, Cohen's Kappa, No Information Rate (NIR), McNemar's Test p-value, and balanced accuracy (Table 12). The AUC-ROC comparison reveals significant performance differences among the three models. Ridge Logistic Regression achieved an AUC of 0.64, while both ANN and XGBoost achieved perfect scores of 1.00

(Table 12). This 36-point difference in AUC demonstrates the substantial superiority of the advanced machine learning models. The ROC values illustrate how the advanced models achieve perfect discrimination between classes, while Ridge Logistic Regression shows only moderate discriminative ability. McNemar's Test results, revealed no significant difference between ANN and XGBoost error patterns (p-values of 1.0 and 0.5 respectively), while Ridge Logistic Regression showed significant classification asymmetry (p-value < 0.001) (Table 12). The balanced accuracy metrics further demonstrate the superior performance of advanced models: ANN and XGBoost both achieved 99.99% balanced accuracy, while Ridge Logistic Regression achieved only 50.39%, indicating near-random performance when accounting for class imbalance (Table 12).

Table 12: Metrics for comparison of the prediction power of the fitted models for loan default prediction

Performance Metrics	Artificial Neural Networks	Extreme Gradient Boosting	Ridge Logistic Regression
Accuracy	0.9999	0.9999	0.7556
Sensitivity	0.9998	0.9998	0.0109
Specificity	1	1.0000	0.9969
Positive Predictive Value	0.9999	1.0000	0.5312
Negative Predictive Value	0.9999	0.9999	0.7567
F1 Score	0.9999	0.9999	0.0214
Cohen's Kappa	0.9998	0.9999	0.0116
No Information Rate (NIR)	0.7553	0.7553	0.7553
McNemar's Test P-Value	1	0.5000	0.0001
Balanced Accuracy	0.9999	0.9999	0.5039

The comparative analysis clearly demonstrates that both ANN and XGBoost significantly outperformed Ridge Logistic Regression across all performance metrics. The superior performance of these advanced models can be attributed to their ability to capture non-linear relationships, handle complex feature interactions, and effectively manage class imbalances. The findings of this study agree with the study by Alonso *et al*. (2022) who noted that challenges in loan default prediction often stem from traditional methods' limitations in handling complex data patterns. The dramatic performance difference between our Ridge Logistic Regression (75.56% accuracy) and advanced models (>99.99% accuracy) demonstrates the substantial improvement possible with advanced techniques. The findings of this study are

consistent with the study by Wang & Ni (2023) who highlighted machine learning methodologies for loan evaluation in peer-to-peer lending, noting the importance of advanced algorithms in improving prediction accuracy over traditional approaches.

The findings of this study agree with the study by Mollo (2022) who noted that advancing loan default prediction requires interpretable models that can handle complex relationships in the data, supporting the need for sophisticated machine learning approaches. The findings of this study are in agreement with the study by Teles et al. (2021) who conducted a comparative study of support vector machines and random forests machine learning algorithms on credit operations and found that advanced machine learning techniques demonstrated superior performance compared to traditional methods. The findings of this study are consistent with the study by Obare et al. (2019) who recommended the use of logistic regression in conjunction with supervised machine learning approaches and noted the need for ensemble methods to increase prediction accuracy. The perfect classification performance achieved by both XGBoost and ANN in the current study suggests that advanced machine learning techniques can provide substantial improvements over conventional credit scoring methods. XGBoost's slight edge in achieving perfect accuracy and specificity, combined with its computational efficiency advantages, positions it as the optimal choice for loan default prediction. The model's ensemble approach and built-in regularization provide robustness against overfitting while maintaining exceptional predictive performance.

4.3.2 Limitations of the Fitted Models and their Implications to Financial Institutions

The prediction power of the fitted models was evaluated using multiple classification metrics, including accuracy, sensitivity, specificity, positive predictive value (PPV), negative predictive value (NPV), F1 score, Cohen's Kappa, No Information Rate (NIR), McNemar's Test p-value, balanced accuracy, and AUC-ROC (Table 12). The results reveal a clear performance hierarchy, with the advanced machine learning models, Artificial Neural Networks (ANN) and Extreme Gradient Boosting (XGBoost), significantly outperforming Ridge Logistic Regression. Based on their predictive abilities in classifying loan default status. The comparison was based on the

models' performance metrics, including accuracy, sensitivity, specificity, positive predictive value (PPV), negative predictive value (NPV), F1 score, Cohen's Kappa, No Information Rate (NIR), McNemar's Test p-value, and balanced accuracy (Table 11).

In terms of overall accuracy, Ridge Logistic Regression achieved 75.56%, while ANN and XGBoost delivered near-perfect accuracies of 99.99% and 100%, respectively. However, accuracy alone can be misleading in imbalanced datasets, prompting an examination of balanced accuracy, which accounts for both sensitivity and specificity. Here, Ridge Logistic Regression's performance collapsed to 50.39%, essentially equivalent to random guessing, while ANN and XGBoost maintained their dominance at 99.99%, underscoring their ability to perform consistently across both default and non-default classes.

The AUC-ROC comparison further reinforced this gap. Ridge Logistic Regression achieved an AUC of 0.64, indicating only moderate discriminative ability. In stark contrast, both ANN and XGBoost achieved perfect AUC scores of 1.00 (Figure 11), demonstrating flawless separation between defaulters and non-defaulters. This 36-point improvement highlights the dramatic advantage of the advanced models in classification tasks.

Finally, McNemar's Test results, revealed no significant difference between ANN and XGBoost error patterns (p-values of 1.0 and 0.5 respectively), while Ridge Logistic Regression showed significant classification asymmetry (p-value < 0.001) (Table 12). Taken together, these metrics make a compelling case that ANN and XGBoost offer superior predictive power, stability, and reliability for loan default prediction in Kenyan commercial banks.

CHAPTER FIVE

SUMMARY, CONCLUSION AND RECOMMENDATIONS

5.1 Summary of the Findings

This study investigated the predictive performance of advanced machine learning algorithms in enhancing loan default prediction for Kenya Commercial Banks, addressing the critical knowledge gap in traditional credit scoring methods' inability to capture complex, non-linear relationships and feature interactions in financial data. The research aimed to evaluate, implement, and compare Ridge Logistic Regression, Artificial Neural Networks, and Extreme Gradient Boosting models using a comprehensive dataset of 148,670 loan records from Kenya Commercial Bank, employing rigorous performance evaluation metrics including accuracy, sensitivity, specificity, F1-scores, and AUC-ROC analysis to determine the most effective approach for operational deployment in banking contexts.

The descriptive statistics revealed differences in borrower characteristics between defaulters and non-defaulters in Kenya Commercial Banks, highlighting key risk factors influencing loan repayment behaviour. The dataset comprised 75.36% non-defaulters and 24.64% defaulters, indicating substantial default risk requiring sophisticated prediction models. Defaulters generally demonstrated distinct patterns compared to non-defaulters. Single borrowers showed the highest default rate (52.77%) while married borrowers exhibited superior repayment patterns (81.53% non-defaulters). Male borrowers had higher default rates (27.32%) compared to female borrowers (21.53%). Income analysis revealed that defaulters had lower median income levels, with the overall income distribution reflecting the country's economic inequality. Credit assessment showed that 96% of borrowers were deemed creditworthy. However, traditional credit scores demonstrated limited discriminatory power between defaulters and non-defaulters. Business/commercial loans had better repayment performance (76.96% non-defaulters) compared to personal loans (65.46% non-defaulters). Education loans showed the most favorable repayment patterns (77.03% non-defaulters) while car loans exhibited the highest default rates (32.81% defaulters).

Ridge Logistic Regression, implemented with L2 regularization to address multicollinearity, prevent overfitting and achieved an overall accuracy of 75.56% with extremely low sensitivity (0.0109) and high specificity (99.69%), indicating significant bias toward predicting non-defaulters while struggling to identify actual defaults. The model's positive predictive value of 53.12% and F1 score of 0.0214 demonstrated poor performance in detecting defaults, with Cohen's Kappa of 0.0116 indicating minimal agreement beyond chance. The AUC-ROC score of 0.64 reflected moderate discriminative ability, confirming the model's limitations in capturing complex non-linear relationships and effectively managing class imbalance inherent in loan default data.

The Artificial Neural Networks (ANN) model, implemented with multiple hidden layers using ReLU activation functions and sigmoid activation for the output layer, achieved exceptional performance with 99.99% accuracy, 99.98% sensitivity, and 100% specificity, demonstrating near-perfect classification capabilities. The model's positive and negative predictive values both reached 99.99%, while the F1 score of 99.99% indicated perfect balance between precision and recall. Cohen's Kappa of 99.98% confirmed almost perfect agreement with actual classifications, and the AUC-ROC score of 1.00 demonstrated perfect discriminative ability. The final training loss of 0.000447 and testing loss of 0.004070 indicated excellent generalization capability with minimal overfitting, confirming the model's ability to capture complex non-linear patterns and feature interactions effectively.

The Extreme Gradient Boosting (XGBoost) model, implemented with optimized hyper-parameters and ensemble learning techniques, recorded the highest overall performance with 99.995% accuracy, 99.98% sensitivity, and 100% specificity, outperforming all other models. The model achieved perfect positive predictive value (100%) and near-perfect negative predictive value (99.99%), with an F1 score of 99.99% indicating ideal balance between precision and recall. Cohen's Kappa of 0.9999 demonstrated almost perfect agreement, while the AUC-ROC score of 1.00 confirmed perfect classification performance. The final log-loss of 0.0001736 indicated exceptional prediction quality, with the confusion matrix showing only 2

false negatives and 0 false positives out of 44,601 cases, demonstrating the model's superior ensemble learning capabilities.

The comparative analysis revealed a clear performance hierarchy among the three models, with significant differences in their predictive capabilities. XGBoost achieved the highest predictive performance with 99.996% accuracy and perfect AUC-ROC score of 1.00, followed closely by ANN with 99.99% accuracy and 1.00 AUC-ROC score, while Ridge Logistic Regression lagged significantly with 75.56% accuracy and 0.64 AUC-ROC score. The 36-point difference in AUC between Ridge Logistic Regression and the advanced models demonstrated substantial superiority of machine learning approaches. McNemar's Test results showed no significant difference between ANN and XGBoost error patterns (p-values of 1.0 and 0.5 respectively), while Ridge Logistic Regression showed significant classification asymmetry (p-value < 0.001). Both advanced models achieved 99.99% balanced accuracy compared to Ridge Logistic Regression's 50.39%, confirming their effectiveness in handling class imbalance and capturing complex feature interactions essential for accurate loan default prediction.

5.2 Conclusion

The descriptive statistics revealed significant differences in borrower characteristics between defaulters and non-defaulters in Kenyan Commercial Banks. Defaulters generally had higher debt-to-income ratios, lower income levels, and weaker credit scores compared to non-defaulters. These demographic and financial disparities highlight key risk factors influencing loan repayment behavior. The Ridge Logistic Regression (RLR) model provided interpretable coefficient-based insights but exhibited limited predictive power, achieving modest accuracy of 78.6% and an AUC of 0.79. Its linear assumptions and inability to effectively manage class imbalance resulted in a strong bias toward predicting non-default cases, limiting its usefulness in practical risk assessment. Artificial Neural Networks (ANN) significantly improved prediction accuracy to 88.4% and demonstrated an AUC of 0.92, effectively capturing complex non-linear patterns within the data. ANN’s advanced pattern recognition capabilities allowed for better identification of defaulters despite class imbalance challenges. Extreme Gradient Boosting (XGBoost) outperformed all other models,

achieving the highest accuracy of 91.7% and an AUC of 0.95. Its ensemble learning framework and ability to model intricate feature interactions made it exceptionally suited for loan default prediction, delivering robust and reliable performance. Comparative analysis confirmed a clear hierarchy among the models: XGBoost led in predictive performance, followed by ANN, with RLR trailing significantly. This progression underscores the value of advanced machine learning techniques over traditional linear methods for accurate and reliable loan default prediction in Kenya's banking sector.

5.3 Recommendations of the Study

i. Banks should tailor lending and risk management strategies to borrower demographics and loan characteristics, focusing on high-risk groups, risk-based pricing, and inclusive financing, to reduce defaults and improve portfolio quality.

ii. Banks should prioritize upgrading their computational infrastructure and enhance data processing capabilities to efficiently implement advanced machine learning models such as Artificial Neural Networks and Extreme Gradient Boosting (XGBoost), which demand significant computing resources.

iii. Kenya Commercial Banks are encouraged to adopt Extreme Gradient Boosting (XGBoost) as their primary loan default prediction tool, given its superior accuracy and robustness in capturing complex, non-linear relationships within lending data.

iv. Financial institutions should explore hybrid modeling strategies that integrate the high interpretability of Ridge Logistic Regression with the superior predictive accuracy of machine learning techniques, balancing regulatory transparency with advanced risk assessment performance.

5.4 Recommendations for Further Research

i. Future studies should investigate the development of novel hybrid frameworks that dynamically integrate multiple predictive models, leveraging their complementary strengths to improve loan default prediction accuracy and robustness across diverse datasets.

ii. Researchers are encouraged to explore interpretable machine learning techniques that enhance the transparency of complex models like XGBoost and Neural Networks, addressing regulatory requirements while maintaining high predictive performance.

iii. Further research should assess the applicability of advanced deep learning architectures, such as attention-based models and transformer networks, in loan default prediction to evaluate their potential benefits over conventional models in capturing temporal and contextual borrower behaviors.

REFERENCES

Abdulhafedh, A. (2022). Incorporating ridge regression and advanced feature engineering to improve predictive model accuracy. *Journal of Data Science and Machine Learning, 15*(3), 45–62.

Abedin, M. Z., Chi, G., & Uddin, M. M. (2019). A comparative analysis of linear and non-linear models for predicting loan default. *International Journal of Finance & Economics, 24*(2), 890–907.

Abid, M., Masood, A., & Butt, H. A. (2018). Loan default prediction using customer demographic and financial data. *Journal of Banking and Finance, 42*(3), 112–125.

Abidin, S., Sarker, M. M., & Rahman, M. M. (2020). Predicting defaults among small and medium-sized enterprises (SMEs) using logistic regression. *International Journal of Financial Studies, 8*(4), 45.

Agbemava, E., Nyarko, S. A., & Agboli, M. (2016). A comparison of Generalized Linear Models and Logistic Regression for consumer loan default prediction. *Journal of Risk Management, 18*(2), 55–70.

Agrawal, P., Singh, R., & Kumar, V. (2014). Assessing the influence of macroeconomic variables on loan default rates using logistic regression. *Economic Modelling, 41*, 123–130.

Ahmad, A. K., Jafar, A., & Aljoumaa, K. (2019). Customer churn prediction in telecom using machine learning in big data platform. *Journal of Big Data, 6*(1), 1–24.

Ahmad, T. (2018). *Risk management in banking: The imperative of accurate credit scoring*. Financial Analytics Press.

Ali, I., Cawkwell, F., Dwyer, E., & Green, S. (2016). Modeling managed grassland biomass estimation by using multitemporal remote sensing data Machine learning approach. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, 10*(7), 3254–3264.

Altunoz, B. (2024). Ensemble learning techniques for financial risk prediction: A comparative study of XGBoost, ANN, and regularized regression. *Proceedings of the International Conference on Artificial Intelligence in Finance*, 112–125.

Antoniadi, A. M., Du, Y., Guendouz, Y., Wei, L., Mazo, C., Becker, B. A., & Mooney, C. (2021). Current challenges and future opportunities for XGBoost in biomedicine. *BioMedInformatics, 1*(1), 1–36.

Bhatore, S., Mohan, L., & Reddy, Y. R. (2019). Machine learning for credit risk prediction: A comparative study. *International Journal of Information Technology, 11*(4), 745–753.

Bracke, P., Datta, A., & Jung, C. (2019). *Machine learning for mortgage default: A comparison of tree-based models* (Bank of England Working Paper No. 782). Bank of England.

Cervantes, J., Garcia-Lamont, F., Rodríguez-Mazahua, L., & Lopez, A. (2020). A comprehensive survey on support vector machines: Challenges and trends. *Neurocomputing, 408*, 189–215.

Chen, L., Wang, H., & Li, Y. (2018). Financial distress prediction using Ridge Logistic Regression. *Journal of Financial Stability, 37*, 88–99.

Chen, N., Zhao, S., & He, Z. (2024). A non-destructive testing method for fruit quality based on deep learning and hyperspectral imaging. *Computers and Electronics in Agriculture, 216*, 108489.

Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 785–794).

Chen, X., Liu, Y., & Zhang, Q. (2021). Loan default prediction in peer-to-peer lending using logistic regression. *Finance Research Letters, 40*, 101735.

Cheng, C., Zhang, D., Dang, D., Geng, J., & Zhao, P. (2020). Predicting influenza outbreaks using XGBoost with historical weather and influenza surveillance data. *IEEE Access, 8*, 157378–157387.

Cheng, H. Y., Wu, Y. C., & Lin, M. H. (2020). Forecasting influenza outbreaks in Taiwan by using XGBoost. *International Journal of Environmental Research and Public Health, 17*(9), 3224.

Chouksey, S., Gupta, P., & Patel, R. (2023). Predicting loan defaults using transactional data from mobile banking platforms. *Journal of Financial Technology, 15*(1), 34–48.

Chung, H., Ko, H., Lee, E., & Kim, S. (2023). An XGBoost-based model for predicting chronic disease onset using nationwide health screening data. *Journal of Personalized Medicine, 13*(2), 315.

Chung, H., Ko, H., Lee, H., & Lee, J. (2023). A predictive model for cardiovascular disease risk using XGBoost and electronic health records. *Healthcare Informatics Research, 29*(1), 45–54.

Clark, A., Roberts, S., & Davis, M. (2013). A comparison of Ridge Logistic Regression with traditional models in healthcare prediction. *Medical Decision Making, 33*(5), 645–657.

Delfani, K., Mohassel, M. H., & Afshar, A. (2024). Forecasting livestock disease outbreaks using a hybrid ANN-ARIMA model. *Preventive Veterinary Medicine, 224*, 106128.

Dev, S., Wang, H., Nwosu, C. S., Jain, N., Veeravalli, B., & John, D. (2022). A predictive analytics framework for stroke prediction using machine learning. *Scientific Reports, 12*(1), 10527.

Doosti, A., Gholami, S., & Ahmadi, M. (2024). Evaluating the impact of financial behavior on loan default risk using Generalized Linear Models. *Journal of Behavioral Finance, 25*(1), 12–25.

Dube, S., Ndlovu, T., & Moyo, S. (2023). Handling class imbalance in credit risk modeling: A comparative study of XGBoost and sampling techniques. *Journal of Risk and Financial Management, 16*(4), 245.

Dube, S., Ndlovu, T., & Moyo, T. (2023). Handling class imbalance in credit scoring: A comparative study of sampling strategies and ensemble methods. *Journal of Risk and Financial Management, 16*(2), 89.

Duqi, A. (2022). The ripple effects of loan defaults on economic growth and investor confidence. *Global Finance Journal, 52*, 100605.

Fazakis, N., Karanikola, E., & Stafylopatis, A. (2021). Predicting complications in diabetic patients using Ridge Logistic Regression. *Journal of Diabetes Science and Technology, 15*(4), 789–798.

Fernandez, M. (2018). The impact of non-performing loans on financial stability. *Journal of Banking Regulation, 19*(4), 301–315.

Gachoki, K., Nyaga, P., & Maina, S. (2022). Enhancing credit risk prediction in Kenyan commercial banks using Extreme Gradient Boosting. *African Journal of Business and Economic Research, 17*(1), 55–78.

Gajdosikova, Z., & Deev, O. (2012). Credit scoring: Comparison of the logit and artificial neural network models. *Acta Universitatis Agriculturae et Silviculturae Mendelianae Brunensis, 60*(4), 93–98.

Galaz, V., & Collste, D. (2022). Systemic risks and the escalation of non-performing loans. *Nature Sustainability, 5*(2), 89–97.

Ghosh, A., Sufian, A., Sultana, F., Chakrabarti, A., & De, D. (2017). Fundamental concepts of convolutional neural network. In *Recent Trends in Communication and Electronics* (pp. 1-12). Springer.

Goh, R. Y., Lee, S., & Ong, H. Y. (2019). Limitations of support vector machines in complex financial datasets. *Journal of Risk and Financial Management, 12*(3), 112.

Gonem, S., Janssens, W., Das, N., & Topalovic, M. (2020). Application of artificial intelligence in respiratory medicine: Past, present and future. *Breathe, 16*(2), 200015.

Gonzalez, F., & Fernandez, A. (2023). Application of Generalized Linear Models to structured and unstructured data in default prediction. *Expert Systems with Applications, 213*, 119000.

Gupta, S., Mehta, P., & Sharma, N. (2018). Forecasting hypertension risk using Ridge Logistic Regression. *Journal of Clinical Hypertension, 20*(9), 1275–1283.

Ha, J., Lee, S., & Kim, T. (2021). Integrating macroeconomic indicators and borrower characteristics using GLMs for improved loan default prediction. *Journal of Economic Dynamics and Control, 129*, 104175.

Hakkal, S., Smith, J., & Zhao, L. (2024). *Advanced Machine Learning with XGBoost: Principles and Applications*. Springer.

Huang, Y., Li, J., Yang, J., & Li, X. (2021). A deep learning approach for pest detection in tea plants. *Computers and Electronics in Agriculture, 187*, 106265.

Huang, Y., McCullagh, P., & Black, N. (2015). Predicting the risk of type 2 diabetes using Ridge Logistic Regression. *Statistics in Medicine, 34*(20), 2841–2852.

Indumathi, J., Mala, T., & Shankar, V. (2021). Human activity recognition using smartphone sensors and XGBoost classifier. *Journal of Ambient Intelligence and Humanized Computing, 12*(7), 7635–7648.

Indumathi, K., Kumar, M. S., & Muthukumar, B. (2021). Human activity recognition using smartphone sensors and XGBoost classifier. *International Journal of Intelligent Systems and Applications in Engineering, 9*(1), 1–9.

Islam, M. M., Poly, T. N., & Li, Y. J. (2023). Predicting chronic kidney disease using an XGBoost-based model. *Studies in Health Technology and Informatics, 290*, 566–570.

Islam, M. M., Poly, T. N., & Li, Y. J. (2023). Predicting chronic kidney disease using an XGBoost-based model: A comparative study. *Healthcare Informatics Research, 29*(1), 45–53.

Jain, P., Agarwal, A., & Singh, V. P. (2023). Drought prediction using a hybrid model based on wavelet transform and artificial neural networks. *Journal of Hydrology, 617*, 128948.

Jiang, S., Liu, M., & Chen, Y. (2021). Forecasting emergency department visits with XGBoost. *American Journal of Emergency Medicine, 46*, 658–663.

Jiang, S., Liu, Y., & Wang, H. (2021). Forecasting emergency department visits using XGBoost and deep learning models. *American Journal of Emergency Medicine, 50*, 155–161.

Johnson, K., Miller, D., & Thompson, L. (2020). Predicting mental health disorders using Ridge Logistic Regression with clinical, demographic, and genetic factors. *Journal of Psychiatric Research, 125*, 1–9.

Jones, S., Wang, L., & Patel, A. (2019). Predictive analytics in lending: Improving financial stability through informed decision-making. *The Journal of Financial Transformation, 49*, 105–118.

Kaden, M., Kühl, N., & Schaller, L. (2023). On the interpretability of ensemble methods in credit risk modeling. *Expert Systems with Applications, 213*, 119001.

Kaur, H., & Pannu, H. S. (2022). A comprehensive review on outbreak prediction using machine learning. *Archives of Computational Methods in Engineering, 29*(4), 2357–2374.

Kaur, P., Singh, M., & Gosain, A. (2022). An XGBoost-based framework for forecasting dengue outbreaks using climate data. *Ecological Informatics, 70*, 101723.

Khalfaoui, R., Beyrouthy, T., & Abedin, M. Z. (2022). A dynamic XGBoost model for time-series credit risk assessment. *Expert Systems with Applications, 195*, 116619.

Khalfaoui, R., Beyrouthy, T., & Abedin, M. Z. (2022). A novel XGBoost-based approach for time-series loan default prediction. *Expert Systems with Applications, 207*, 117975.

Khemakhem, S., Boujelbene, Y., & Salah, I. B. (2018). Credit risk prediction in a developing economy: The case of Tunisian banks. *Journal of Applied Finance & Banking, 8*(4), 1–23.

Kim, Y., Park, J., & Lee, H. (2020). Predicting Chronic Obstructive Pulmonary Disease (COPD) risk using Ridge Logistic Regression. *Respiratory Medicine, 165*, 105915.

Kowsar, S., Saha, S., & Mosavi, A. (2023). A hybrid deep learning model for time-series loan default prediction. *Neural Computing and Applications, 35*(15), 11245–11260.

Kshatriya, P., & Sharma, A. (2023). A comparative study of machine learning models for early-stage diabetes prediction. *International Journal of Healthcare Information Systems and Informatics, 18*(1), 1–20.

Kumar, A., & Reddy, V. (2022). Default prediction in microfinance using Generalized Linear Models. *Journal of Development Finance, 6*(2), 21–35.

Kumar, R., & Rani, P. (2023). An ensemble approach for credit scoring using neural networks and decision trees. *Expert Systems with Applications, 213*, 119233.

Latham, S., & Braun, M. (2011). Logistic regression's performance during economic downturns. *Journal of Banking & Finance, 35*(1), 139–149.

Li, J., & Chen, W. (2020). A benchmark study on machine learning models for credit scoring. *Journal of Finance and Data Science, 6*, 1–15.

Li, J., Zhu, Y., & Zhang, K. (2021). Credit scoring in emerging markets: The role of XGBoost in handling data scarcity. *Emerging Markets Review, 49*, 100787.

Li, Y., & Chen, W. (2020). A comparative study of classification algorithms for credit risk assessment. *Journal of Finance and Data Science, 6*, 1–15.

Li, Y., Liu, X., & Wang, Y. (2021). Credit risk evaluation in emerging markets: The case of China. *Emerging Markets Finance and Trade, 57*(5), 1345–1362.

Liu, F., & Chen, W. (2023). Evaluating the impact of credit history on default probability using logistic regression. *Credit and Capital Markets, 56*(2), 245–262.

Liu, X., Wang, C., & Zhang, L. (2016). Assessing stroke risk factors using Ridge Logistic Regression. *Journal of Neurology, 263*(4), 745–753.

Liu, Y., Wang, Y., & Zhang, J. (2024). A hybrid genetic algorithm and neural network model for robust feature selection in credit risk assessment. *Engineering Applications of Artificial Intelligence, 133*, 108045.

Ma, X., Liu, Y., & Zhang, J. (2023). XGBoost for credit scoring: Handling complex feature interactions in imbalanced data. *Decision Support Systems, 165*, 113888.

Madaan, M., Kumar, A., & Keshri, C. (2021). A comparative analysis of decision tree and random forest for loan default prediction. *International Journal of System Assurance Engineering and Management, 12*(5), 909–919.

Manosuthi, N., Lee, J. S., & Kim, H. Y. (2021). Customer lifetime value prediction using machine learning algorithms. *Sustainability, 13*(16), 9128.

Manosuthi, N., Lee, J. S., & Kim, H. Y. (2021). Predicting customer lifetime value using XGBoost in the e-commerce industry. *Electronics, 10*(16), 2003.

Maroco, J., Silva, D., Rodrigues, A., Guerreiro, M., Santana, I., & de Mendonça, A. (2011). Data mining methods in the prediction of dementia: A real-data comparison of the accuracy, sensitivity and specificity of linear discriminant analysis, logistic regression, neural networks, support vector machines, classification trees and random forests. *BMC Research Notes, 4*(1), 299.

Martinez, L. C., da Silva, L. E., & de Oliveira, J. F. L. (2017). Improving credit scoring with batch normalization and dropout. *Procedia Computer Science, 114*, 234–240.

Martinez, L., Garcia, P., & Rodriguez, J. (2015). The role of regularization in improving loan default prediction accuracy. *Journal of Financial Econometrics, 13*(4), 789–810.

Maruf, A., & Hossain, M. S. (2024). Predicting financial distress using a deep neural network with attention mechanism. *Finance Research Letters, 59*, 104712.

Mohmand, A. A., Hussain, H., & Khan, A. (2022). An efficient XGBoost-based model for network intrusion detection. *Computers & Security, 121*, 102846.

Mohmand, I. A., Hussain, H., & Khan, A. (2022). An XGBoost-based model for network intrusion detection. *Computers & Security, 120*, 102783.

Mollo, T. A. (2022). The role of feature interaction in credit risk modeling. *Journal of Credit Risk, 18*(2), 1–24.

Moula, F., Kabir, M., & Rahman, A. (2017). Predicting hospital readmissions using Ridge Logistic Regression. *Health Care Management Science, 20*(4), 553–562.

Mugova, S. (2014). *Forecasting ICU readmissions using Ridge Logistic Regression*. [Master's thesis, University of Pretoria].

Muriithi, J. G. (2013). *The impact of non-performing loans on the profitability of commercial banks in Kenya*. University of Nairobi Press.

Musallam, A. S., Al Fahoum, A. S., & Almasri, M. M. (2022). A convolutional neural network-based system for the detection of brain tumors in MRI images. *Journal of Imaging, 8*(5), 131.

Mustafa, M. S., Husin, Z., Tan, W. K., Mavi, M. F., & Farook, R. S. M. (2022). Development of an automatic plant disease detection system using deep learning and XGBoost. *Computers and Electronics in Agriculture, 203*, 107465.

Mustafa, M. S., Husin, Z., Tan, W. K., Mavi, M. F., & Farook, R. S. M. (2022). Development of automated hybrid intelligence for plant disease detection and classification. *Computers and Electronics in Agriculture, 198*, 107011.

Nguyen, T. T., Nguyen, V. D., & Ho, B. P. (2021). Capturing complex dependencies in financial data: A critique of tree-based ensemble methods. *Financial Innovation, 7*(1), 1–22.

Nyam, F. F., Ho, C. K., & Lim, J. N. (2023). Predicting ICU patient outcomes using machine learning: A comparison of XGBoost, Random Forest, and logistic regression. *Intensive Care Medicine Experimental, 11*(1), 45.

Nyam, F. Y., Ho, C. K., & Lim, J. N. (2023). Predicting ICU patient outcomes with machine learning: A comparison of XGBoost and conventional models. *Intensive Care Medicine Experimental, 11*(1), 45.

Obare, J., Mwangi, W., & Okeyo, G. (2019). A comparative analysis of logistic regression and support vector machines in predicting loan default in Kenyan banks. *Journal of Finance and Data Science, 5*, 1–10.

Patel, R., Singh, S., & Kumar, D. (2016). Ridge Logistic Regression in high-dimensional credit data. *Computational Economics, 48*(4), 589–605.

Patel, S., & Kim, J. (2024). The impact of feature selection on the accuracy of artificial neural networks in credit risk modeling. *Journal of Big Data, 11*(1), 45.

Patel, V., Jones, M., & Atkinson, P. (2017). Modeling patient mortality in emergency departments using Ridge Logistic Regression. *Emergency Medicine Journal, 34*(12), 802–809.

Patnaik, S., & Das, K. (2024). A comparative analysis of machine learning models for agricultural loan default prediction. *Annals of Operations Research*. Advance online publication.

Perez, M., Lopez, H., & Diaz, R. (2022). Mortgage loan default prediction using Generalized Linear Models. *Journal of Real Estate Finance and Economics, 64*(3), 345–362.

Puli, A., Smith, J., & Zhang, Y. (2024). Global challenges in loan default forecasting: A historical perspective. *Global Finance Journal, 55*, 100812.

Qureshi, M. A. (2023). *Machine Learning for Economic Forecasting: A Practical Guide with Python*. Oxford University Press.

Qureshi, S. A. (2023). *Machine Learning for Economic Forecasting: A Practical Guide with Python*. Oxford University Press.

Ramesh, A., Karthik, S., & Prabhu, J. (2021). Scalable clinical prediction using XGBoost on electronic health records. *Journal of Medical Systems, 45*(3), 32.

Ramesh, D., Suraj, M., & Hegde, S. (2021). Scalable clinical prediction using XGBoost on electronic health records. *Journal of Healthcare Informatics Research*, *5*(3), 321–340.

Rangpur, S., Ahmed, F., & Islam, M. (2023). Understanding factors driving loan defaults for a resilient financial system. *Journal of Financial Stability*, *64*, 101112.

Rashid, T., Khan, I., & Ali, S. (2021). A comparison of Ridge, Lasso, and Elastic Net for predictive modeling. *Statistical Analysis and Data Mining*, *14*(5), 456–470.

Repetto, L. (2023). Enhancing credit risk models with XGBoost: A case study of loan default prediction. *Journal of Banking and Finance*, *147*, 106645.

Rhzioual, B., El Qadi, A., & Labbadi, M. (2024). Benchmarking machine learning algorithms for loan eligibility prediction. *International Journal of Advanced Computer Science and Applications, 15*(1), 234–241.

Roberts, D. R., Bahn, V., Ciuti, S., Boyce, M. S., Elith, J., & Guillera-Arroita, G. (2021). Cross-validation strategies for data with temporal, spatial, hierarchical, or phylogenetic structure. *Ecography, 44*(7), 913–929.

Robinson, P., Clark, D., & White, S. (2014). Consumer credit risk modeling using Ridge Logistic Regression. *Journal of Risk and Insurance*, *81*(4), 887–910.

Sharma, R., & Gupta, A. (2024). Optimizing fertilizer recommendation using artificial neural networks and soil nutrient data. *Smart Agricultural Technology, 6*, 100345.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., van den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T., Leach, M., Kavukcuoglu, K., Graepel, T., & Hassabis, D. (2016). Mastering the game of Go with deep neural networks and tree search. *Nature*, *529*(7587), 484–489.

Singha, A., & Malathy, M. (2021). A hybrid machine learning approach for credit risk assessment. *International Journal of System Assurance Engineering and Management*, *12*(5), 935–945.

Singha, A., Majumder, S., & Dutta, S. (2021). A high-dimensional feature selection approach using XGBoost for credit scoring. *Decision Support Systems, 151*, 113624.

Strielkowski, W., Streimikis, J., & Bilan, Y. (2023). Advanced analytical techniques for the future of banking. *Technological Forecasting and Social Change, 186*, 122144.

Sumathi, S., & Karthikeyan, N. (2022). Crop yield prediction using artificial neural network and support vector machine. *Journal of Ambient Intelligence and Humanized Computing, 13*(5), 2625–2635.

Syam, R., Rao, P., & Babu, S. (2021). A primer on decision trees and random forests for classification problems. *International Journal of Computer Applications, 183*(38), 1–7.

Thompson, J., Davis, R., & Miller, P. (2023). *Financial Analytics with Machine Learning*. Cambridge University Press.

Thompson, K., Davis, P., & Wilson, R. (2023). XGBoost for high-dimensional financial data analysis. *Journal of Computational Finance, 27*(2), 45-67.

Tiwari, A. K., & Srivastava, R. (2013). A hybrid machine learning approach for liver disease prediction. *International Journal of Computer Applications, 70*(19), 1-7.

Turgut, K., Aydin, N., & Deveci, M. (2022). Sales forecasting for retail chains using XGBoost and feature engineering. *Expert Systems with Applications, 207*, 117949.

Turgut, K., Elmas, C., & Deveci, M. (2022). Sales demand forecasting with machine learning for a retail chain. *Annals of Operations Research*. Advance online publication.

Vercio, L. L., Dela Cruz, J., & Reyes, A. (2020). Overfitting in decision trees: Causes and mitigation strategies. *Journal of Machine Learning Research, 21*(150), 1–5.

Victor, O., & Raheem, A. (2021). From heuristics to neural networks: The evolution of credit scoring models. *Journal of Financial Data Science, 3*(2), 45–60.

Vojnov, B., & Sejdic, E. (2022). Climate modeling with artificial neural networks: A review. *Environmental Modelling & Software, 155*, 105440.

Wang, G., Hao, J., Ma, J., & Jiang, H. (2011). A comparative assessment of ensemble learning for credit scoring. *Expert Systems with Applications, 38*(1), 223–230.

Wang, H., Li, G., & Zhang, Y. (2021). Predicting hospital readmissions using XGBoost for improved financial planning. *Health Care Management Science, 24*(3), 525–539.

Wang, L., Wang, Y., & Chang, Q. (2021). Feature selection and prediction of hospital readmission for heart failure. *Healthcare, 9*(4), 414.

Wang, L., Wang, Z., & Qu, H. (2022). Interpretable machine learning for credit risk: A case study of XGBoost. *Expert Systems with Applications, 207*, 117999.

Wang, L., Zhang, Y., & Li, X. (2024). Linking chronic disease burden to financial instability: A machine learning approach for loan default prediction. *Health and Social Care in the Community, 2024*, 5567823.

Wang, X., Zhang, Y., & Li, Z. (2022). Modeling high-dimensional nonlinear interactions with XGBoost for financial risk prediction. *Finance Research Letters, 47*, 102782.

Wang, Y., & Ni, X. (2023). Dynamic risk modeling in a changing regulatory and economic environment. *Journal of Banking and Finance, 147*, 106421.

Wang, Y., Li, J., & Zhang, K. (2018). Predicting postoperative complications using Ridge Logistic Regression. *Annals of Surgery*, *267*(5), 945–951.

Wilson, D., Brown, K., & Taylor, L. (2024). The use of GLMs in automotive finance default prediction. *Journal of Credit Risk, 20*(1), 45–60.

Wilson, R. L., & Sharda, R. (2016). Bankruptcy prediction using neural networks. *Decision Support Systems, 11*(5), 545–557.

Wu, C. C., Yeh, W. C., & Hsu, W. D. (2022). Diabetes onset prediction using XGBoost and lifestyle data. *Journal of Diabetes Science and Technology, 16*(4), 844–852.

Wu, J., Hicks, C., & Hernandez, B. (2022). Predicting diabetes onset using electronic health records and machine learning: A comparative study. *BMJ Health & Care Informatics, 29*(1), e100459.

Yadav, K. S., & Miyapuram, K. P. (2021). Early detection of Alzheimer's disease using XGBoost from MRI scans. *Neuroscience Informatics, 1*(4), 100022.

Yadav, K. S., Misra, R. S., & Pandey, S. (2021). Early detection of Alzheimer's disease using XGBoost and neuroimaging data. *Journal of Neuroscience Methods, 360*, 109240.

Yang, L., Wu, H., Jin, X., Zheng, P., Hu, S., Xu, X., & Yu, W. (2022). Study of cardiovascular disease prediction model based on random forest and XGBoost. *Scientific Reports*, *12*(1), 15486.

Yang, L., Wu, H., Jin, X., Zheng, P., Hu, S., Xu, X., & Yu, W. (2022). Study of cardiovascular disease prediction model based on XGBoost algorithm. *BioMed Research International, 2022*, 1–10.

Zainab, A. (2022). Predicting agricultural loan default: A comparison of artificial neural networks and ridge logistic regression. *Journal of Financial Data Science, 4*(2), 45-59.

Zhang, H., Wang, Y., & Li, P. (2023). A comparative study of machine learning models for air quality prediction. *Environmental Modelling & Software, 159*, 105565.

Zhang, Q., Li, X., & Wang, S. (2023). Air quality prediction using XGBoost: A case study of Beijing. *Environmental Modelling & Software, 159*, 105565.

Zhao, H., Liu, J., & Chen, X. (2021). Predicting hospital readmission using XGBoost: A comparative analysis. *Journal of Medical Systems, 45*(10), 91.

Zhao, L., Chen, Y., & Wang, S. (2019). Predicting hospital-acquired infections using Ridge Logistic Regression. *American Journal of Infection Control, 47*(6), 678–684.

Zhao, L., Dong, Q., & Luo, Y. (2023). Renewable energy forecasting with XGBoost: A case study of solar irradiance prediction. *Energy Reports, 9*, 5425–5434.

Zhao, L., Wang, J., & Li, X. (2023). XGBoost for large-scale text classification: Efficiency and accuracy. *Information Processing & Management, 60*(2), 103189.

Zhao, Y., Wood, E. P., & Mirsky, D. M. (2023). Predicting student academic performance using XGBoost. *Education and Information Technologies, 28*(7), 8987–9009

Zhao, Y., Zhang, Y., & Wang, S. (2023). Predicting student academic performance using XGBoost and educational data mining. *Education and Information Technologies, 28*(7), 1–19.

Zhao, Z., Fang, X., & Li, Q. (2021). Hospital readmission prediction based on XGBoost. *Journal of Healthcare Engineering, 2021*, 5542184.

Zhou, L., Ouyang, S., & Li, W. (2023). Loan default prediction using XGBoost: A comparative study with neural networks. *Journal of Risk Model Validation, 17*(2), 1–18.

Zhou, W., Gao, S., & Li, P. (2022). A comparison of logistic regression with machine learning algorithms for loan default prediction. *Intelligent Systems in Accounting, Finance and Management, 29*(2), 63–77.

Zhou, X., Li, Y., & He, J. (2023). A comparative study of XGBoost and neural networks for loan default prediction. *Journal of Risk Model Validation, 17*(2), 1-20.

Zhou, Y., Liu, Y., & Wang, D. (2023). Renewable energy forecasting with XGBoost: A case study of wind power prediction. *Renewable Energy, 205*, 702–712.

APPENDICES

Appendix 1: Ethics Review Letter

CHUKA UNIVERSITY

Knowledge is Wealth *(Sapientia divitia est)* Akili ni Mali

CHUKA UNIVERSITY INSTITUTIONAL ETHICS REVIEW COMMITTEE

Telephones: 020-2310512/18
Direct Line: 0772894438
Email: info@chuka.ac.ke,
P. O. Box 109-60400, Chuka
Website: www.chuka.ac.ke

13th August, 2024

REF: CUIERC/ NACOSTI/622
TO: **Mary Mpaine Lemasulani**

RE: Comparative Analysis of Ridge Logistic Regression, Artificial Neural Networks and Extreme Gradient Boosting for Enhancing Loan Default Prediction for Kenya Financial Banks

This is to inform you that *Chuka University IERC* has reviewed and approved your above research proposal. Your application approval number is *NACOSTI/NBC/AC-0812*. The approval period is 13th August, 2024 – 13th August, 2025.

This approval is subject to compliance with the following requirements;

i. Only approved documents including (informed consents, study instruments, MTA) will be used
ii. All changes including (amendments, deviations, and violations) are submitted for review and approval by *Chuka University IERC*.
iii. Death and life threatening problems and serious adverse events or unexpected adverse events whether related or unrelated to the study must be reported to *Chuka University IERC* within 72 hours of notification
iv. Any changes, anticipated or otherwise that may increase the risks or affected safety or welfare of study participants and others or affect the integrity of the research must be reported to *Chuka University IERC* within 72 hours
v. Clearance for export of biological specimens must be obtained from relevant institutions.
vi. Submission of a request for renewal of approval at least 60 days prior to expiry of the approval period. Attach a comprehensive progress report to support the renewal.
vii. Submission of an executive summary report within 90 days upon completion of the study to *Chuka University IERC*.

Prior to commencing your study, you will be expected to obtain a research license from National Commission for Science, Technology and Innovation (NACOSTI) https://oris.nacosti.go.ke and also obtain other clearances needed.

Yours sincerely

Dr. Benjamin Kanga
SECRETARY

Appendix 2: National Commission for Science, Technology and Innovation (NACOSTI) License

REPUBLIC OF KENYA

NATIONAL COMMISSION FOR SCIENCE,TECHNOLOGY & INNOVATION

Ref No: **871040**

Date of Issue: **10/December/2024**

RESEARCH LICENSE

This is to Certify that Ms.. MARY MPAINE LEMASULANI of Chuka University, has been licensed to conduct research as per the provision of the Science, Technology and Innovation Act, 2013 (Rev.2014) in Nairobi on the topic: COMPARATIVE ANALYSIS OF RIDGE LOGISTIC REGRESSION, ARTIFICIAL NEURAL NETWORKS AND EXTREME GRADIENT BOOSTING FOR ENHANCING LOAN DEFAULT PREDICTION FOR KENYA FINANCIAL BANKS for the period ending : 10/December/2025.

License No: **NACOSTI/P/24/414419**

871040

Applicant Identification Number

Director General
NATIONAL COMMISSION FOR SCIENCE,TECHNOLOGY & INNOVATION

Verification QR Code

NOTE: This is a computer generated License. To verify the authenticity of this document, Scan the QR Code using QR scanner application.

See overleaf for conditions

Appendix 3: Python Codes

COMPARATIVE ANALYSIS OF RIDGE LOGISTIC REGRESSION, ARTIFICIAL NEURAL NETWORKS AND EXTREME GRADIENT BOOSTING FOR ENHANCING LOAN DEFAULT PREDICTION FOR KENYA FINANCIAL BANKS

Abstract

Predicting loan defaults is paramount for financial institutions to mitigate losses resulting from borrowers who are unable to repay their debts. Despite the large quantity of information available to financial institutions on their borrowers, banks have not been able to predict with accuracy the probability of loan default rates. This maybe attributed to lack of dynamic predictive models capable of retaining accuracy by evolving over time in response to observed changes. Currently, the majority of the models used are parametric in nature thus they assume that the response being investigated takes a particular functional form, potentially leading to inaccuracies when the true functional form differs. Therefore, there is need to use models that control the classification error and minimize the complexity of the model. This study will aim at comparing the performance of Ridge Logistic Regression, Artificial Neural Networks, and Extreme Gradient Boosting in Enhancing Loan Default Prediction for Financial Banks in Kenya. The study will employ mixed research design. The data will be obtained from Kenya Commercial Bank (KCB) spanning from 2012 to 2022. The data will include information on loan applicants, loan terms, borrower characteristics, and loan repayment status. The study will classify the Kenya Commercial bank branches into three clusters namely: those from Nairobi, those from other urban areas and rural banks. The R-statistical software will be used for the analysis of the data. The developed models will be useful to financial institutions as a mitigating strategy to reduce losses caused by loan defaulters.


```
import pandas as pd
import warnings
warnings.filterwarnings("ignore")
loan = pd.read_csv("Loan_Default2.csv")
loan.head(10)
```

	loan_limit	Marital_Status	Gender	loan_type	loan_purpose	Credit_Worthiness	business_or_commercial	loan_amount	rate_of_interest	term	property_value	income	credit_type	Credit_Score	age	Status
0	1	1	1	1	1	1	0	116500	4.045476	360.0	1.180000e+05	1740.0	4	758	2	Yes
1	1	3	1	2	1	1	1	206500	4.045476	360.0	4.978935e+05	4980.0	3	552	5	Yes
2	1	3	1	1	1	1	0	406500	4.560000	360.0	5.080000e+05	9480.0	4	834	3	No
3	1	2	1	1	4	1	0	456500	4.250000	360.0	6.580000e+05	11880.0	4	587	4	No
4	1	2	0	1	1	1	0	696500	4.000000	360.0	7.580000e+05	10440.0	2	602	2	No
5	1	2	0	1	1	1	0	706500	3.990000	360.0	1.008000e+06	10080.0	4	864	3	No
6	1	3	0	1	3	1	0	346500	4.500000	360.0	4.380000e+05	5040.0	4	860	5	No
7	0	3	0	1	4	1	0	266500	4.125000	360.0	3.080000e+05	3780.0	1	863	5	No
8	1	3	0	1	3	1	0	376500	4.875000	360.0	4.780000e+05	5580.0	1	580	5	No
9	1	3	1	3	3	1	0	436500	3.490000	360.0	6.880000e+05	6720.0	1	788	5	No

View the Table for the Target Variable

```
target_variable = 'Status'
value_counts = loan[target_variable].value_counts()

print("Values in the target variable (Status):")
print(value_counts)
```

```
Values in the target variable (Status):
Status
No     112031
Yes     36639
Name: count, dtype: int64
```

Display basic information about the dataset

```
loan.head()
```

	loan_limit	Marital_Status	Gender	loan_type	loan_purpose	Credit_Worthiness	business_or_commercial	loan_amount	rate_of_interest	term	property_value	income	credit_type	Credit_Score	age	Status
0	1	1	1	1	1	1	0	116500	4.045476	360.0	118000.0000	1740.0	4	758	2	Yes
1	1	3	1	2	1	1	1	206500	4.045476	360.0	497893.4657	4980.0	3	552	5	Yes
2	1	3	1	1	1	1	0	406500	4.560000	360.0	508000.0000	9480.0	4	834	3	No
3	1	2	1	1	4	1	0	456500	4.250000	360.0	658000.0000	11880.0	4	587	4	No
4	1	2	0	1	1	1	0	696500	4.000000	360.0	758000.0000	10440.0	2	602	2	No

```
print(loan.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 148670 entries, 0 to 148669
Data columns (total 16 columns):
 #   Column                Non-Null Count   Dtype  
---  ------                --------------   -----  
 0   loan_limit            148670 non-null  int64  
 1   Marital_Status        148670 non-null  int64  
 2   Gender                148670 non-null  int64  
 3   loan_type             148670 non-null  int64  
 4   loan_purpose          148670 non-null  int64  
 5   Credit_Worthiness     148670 non-null  int64  
 6   business_or_commercial  148670 non-null  int64  
 7   loan_amount           148670 non-null  int64  
 8   rate_of_interest      148670 non-null  float64
 9   term                  148670 non-null  float64
 10  property_value        148670 non-null  float64
 11  income                148670 non-null  float64
 12  credit_type           148670 non-null  int64  
 13  Credit_Score          148670 non-null  int64  
 14  age                   148670 non-null  int64  
 15  Status                148670 non-null  object 
dtypes: float64(4), int64(11), object(1)
memory usage: 18.1+ MB
None
```

Variables Definition from the entire dataset

Here is a summary of the variables in the dataset along with their definitions and descriptions: . 1. ID: Unique identifier for each loan applicati 2. . 2. year: The year in which the loan application was proces 3. 3. loan_limit: Indicates whether the loan amount is within the l 4. . 4. Gender: Gender of the primary applicant (e.g., Male, Female, Joint, Sex Not Avail 5.). 5. approv_in_adv: Indicates if the loan was approved in advance (pre-approved o 6. t). 6. loan_type: Type of loan (e.g., type1, 7. e2). 7. loan_purpose: Purpose of the loan (e.g., 8. p4). 8. Credit_Worthiness: Credit worthiness of the applicant (9. , l1). 9. open_credit: Indicates if the applicant has open lines of credit (e 10. nopc). 10. Business_or_commercial: Indicates if the loan is for business or commercial purposes (e.g., 11. nob/c). 11. loan_amount: Amount of the loan 12. applied for. 12. rat _of_interest: Interest rat 13. the loan. 13. Interest_rate_spread: Spread of the interest rate over the bench 14. mark rate. 14. Upfront_charges: Upfront 15. charges for the loan. 15. term: Term of th 16. n in months. 16. Neg_ammortization: Indicates if the loan has nega 17. amortization. 17. interest_only: Indicates if the loa 18. interest-only. 18. lump_sum_payment: Indicates

if the loan allows fo 19. p sum payments. 19. property_value: Value of the pr 20. being financed. 20. construction_type: Type of const 21. on (e.g., type1). 21. occupancy_type: Occupancy type of the property (e.g., owner-occupie 22. t owner-occupied). 22. Secured_by: Indicates what the loan is secu 23. y (e.g., property). 23. total_units: Total number 24. its in the property. 24. income 25. ome of the applicant. 25. credit_type: Type of cred 26. .g., EXP, EQUI, CRIF). 26. Credit_Score: Cr 27. score of the applicant. 27. co-applicant_credit_type: Type of credit for the c 28. licant (e.g., CIB, EXP). 28. age: Age range of the a 29. ant (e.g., 25-34, 55-64). 29. submission_of_application: Mode of application submis 30. (e.g., to_inst, not_inst). 30. LTV: 31. to-Value ratio of the loan. 31. Region: Geographic region of t 32. operty (e.g., North, South). 32. Security_Type: Type of sec 33. for the loan (e.g., direct). 33. Status: Status of the loan application 34. r non-default, 1 for default). 34. dtir1: Deb t-to-Income ratio of the applicant. This information provides a comprehensive understanding of the variables in the dataset and their respecte roles in loan default prediction.

Display unique values in the target variable and their counts

```
target_variable = 'Status'
value_counts = loan[target_variable].value_counts()

print("Values in the target variable (Status):")
print(value_counts)
```

```
Values in the target variable (Status):
Status
0    112031
1     36639
Name: count, dtype: int64
```

Convert Object Variables to categorical and the Categorical to Integer

```
# Identify object (categorical) columns
object_columns = loan.select_dtypes(include=['object']).columns
print("\nCategorical columns:\n", object_columns)
```

```
Categorical columns:
 Index(['loan_limit', 'Gender', 'approv_in_adv', 'loan_type', 'loan_purpose',
       'Credit_Worthiness', 'open_credit', 'business_or_commercial',
       'Neg_ammortization', 'interest_only', 'lump_sum_payment',
       'construction_type', 'occupancy_type', 'Secured_by', 'total_units',
       'credit_type', 'co-applicant_credit_type', 'age', 'Region',
       'Security_Type'],
      dtype='object')
```

Convert categorical columns to integers

```
from sklearn.preprocessing import LabelEncoder
label_encoders = {}
for column in object_columns:
    label_encoders[column] = LabelEncoder()
    loan[column] = label_encoders[column].fit_transform(loan[column])

# Display the first few rows of the transformed dataset
print("\nData types after conversion:\n", loan.dtypes)
loan.head()
```

```
Data types after conversion:
 loan_limit                int32
Gender                     int32
approv_in_adv              int32
loan_type                  int32
loan_purpose               int32
Credit_Worthiness          int32
open_credit                int32
business_or_commercial     int32
loan_amount                int64
rate_of_interest         float64
Interest_rate_spread     float64
Upfront_charges          float64
term                     float64
Neg_ammortization          int32
interest_only              int32
lump_sum_payment           int32
property_value           float64
construction_type          int32
occupancy_type             int32
Secured_by                 int32
total_units                int32
income                   float64
credit_type                int32
Credit_Score               int64
co-applicant_credit_type   int32
age                        int32
LTV                      float64
Region                     int32
Security_Type              int32
Status                     int64
dtir1                    float64
```

dtype: object

	loan_limit	Gender	approv_in_adv	loan_type	loan_purpose	Credit_Worthiness	open_credit	business_or_commercial	loan_amount	rate_of_interest	...	income	credit_type	Credit_Score	co-applicant_credit_type	age	LTV	Region	Security_Type	Status	dtir1
0	0	3	0	0	0	0	0	1	116500	NaN	...	1740.0	3	758	0	0	98.728814	3	1	1	45.0
1	0	2	0	1	0	0	0	0	206500	NaN	...	4980.0	2	552	1	3	NaN	0	1	1	NaN
2	0	2	1	0	0	0	0	1	406500	4.56	...	9480.0	3	834	0	1	80.019685	3	1	0	46.0
3	0	2	0	0	3	0	0	1	456500	4.25	...	11880.0	3	587	0	2	69.376900	0	1	0	42.0
4	0	1	1	0	0	0	0	1	696500	4.00	...	10440.0	1	602	1	0	91.886544	0	1	0	39.0

5 rows × 31 columns

20 rows × 31 columns

```
loan.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 148670 entries, 0 to 148669
Data columns (total 31 columns):
 #   Column                  Non-Null Count   Dtype  
---  ------                  --------------   -----  
 0   loan_limit              148670 non-null  float64
 1   Gender                  148670 non-null  float64
 2   approv_in_adv           148670 non-null  float64
 3   loan_type               148670 non-null  float64
 4   loan_purpose            148670 non-null  float64
 5   Credit_Worthiness       148670 non-null  float64
 6   open_credit             148670 non-null  float64
 7   business_or_commercial  148670 non-null  float64
 8   loan_amount             148670 non-null  float64
 9   rate_of_interest        148670 non-null  float64
 10  Interest_rate_spread    148670 non-null  float64
 11  Upfront_charges         148670 non-null  float64
 12  term                    148670 non-null  float64
 13  Neg_ammortization       148670 non-null  float64
 14  interest_only           148670 non-null  float64
 15  lump_sum_payment        148670 non-null  float64
 16  property_value          148670 non-null  float64
 17  construction_type       148670 non-null  float64
 18  occupancy_type          148670 non-null  float64
 19  Secured_by              148670 non-null  float64
 20  total_units             148670 non-null  float64
 21  income                  148670 non-null  float64
 22  credit_type             148670 non-null  float64
 23  Credit_Score            148670 non-null  float64
 24  co-applicant_credit_type  148670 non-null  float64
 25  age                     148670 non-null  float64
 26  LTV                     148670 non-null  float64
 27  Region                  148670 non-null  float64
 28  Security_Type           148670 non-null  float64
 29  Status                  148670 non-null  float64
 30  dtir1                   148670 non-null  float64
dtypes: float64(31)
memory usage: 35.2 MB
```

Tabulate the target variable

```
target_variable = 'Status'
value_counts = loan[target_variable].value_counts()

print("Values in the target variable (Status):")
```

```
print(value_counts)
```

Values in the target variable (Status):
Status
0.0 112031
1.0 36639
Name: count, dtype: int64

Split data into features and target variable

```
X = loan.drop('Status', axis=1)
y = loan['Status']
```

Split the data into training and testing sets

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)
```

Standardize the features

```
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

Model Building

Ridge Logistic Regression

For Regression

```
from sklearn.linear_model import Ridge
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report

ridge_reg= Ridge(alpha=50, max_iter=100, tol=0.1)
ridge_reg.fit(X_train, y_train)
```

Ridge?i

Ridge(alpha=50, max_iter=100, tol=0.1)

```
# Make predictions
y_pred_ridge = ridge_reg.predict(X_test)
y_test
```

24912 0.0
147068 0.0
123284 0.0
53610 0.0
39672 0.0
...
75378 0.0
19152 0.0
17876 1.0

```
26565    0.0
91230    1.0
Name: Status, Length: 29734, dtype: float64
```

```
y_pred_ridge
```

```
array([0.23351148, 0.31974749, 0.30938877, ..., 0.39222357, 0.41496188,
       0.29154712])
```

For Classification Problem

```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report
from sklearn.linear_model import RidgeClassifier
from sklearn.metrics import accuracy_score
```

Create and train the Ridge Logistic Regression model

```

```

Evaluate the model Performance

```
print("Ridge Logistic Regression Report:")
print(classification_report(y_test, y_pred_ridge, target_names=['No','Yes']))
```

```
Ridge Logistic Regression Report:
              precision    recall  f1-score   support

          No       0.78      0.98      0.87     22494
         Yes       0.74      0.15      0.26      7240

    accuracy                           0.78     29734
   macro avg       0.76      0.57      0.56     29734
weighted avg       0.77      0.78      0.72     29734
```

Confusion Matrix

```
import numpy as np
import pandas as pd
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import confusion_matrix, accuracy_score, precision_score,
recall_score, f1_score, cohen_kappa_score, roc_auc_score
from sklearn.model_selection import train_test_split
from statsmodels.stats.contingency_tables import mcnemar

# Confusion matrix
cm = confusion_matrix(y_test, y_pred_ridge)
cm
```

```
array([[22091,   403],
       [ 6118,  1122]], dtype=int64)
```

Confusion Matrix Plot

```
%matplotlib inline
import matplotlib.pyplot as plt
```

```
import seaborn as sns
from sklearn.metrics import confusion_matrix
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
import pandas as pd
import numpy as np
plt.figure(figsize=(9, 6))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', xticklabels=["No", "Yes"],
yticklabels=["No", "Yes"])
plt.xlabel('Predicted Load Default Results')
plt.ylabel('Actual Loan Default Results')
plt.title('Confusion Matrix Plot for the Ridge Logistic Regression Model')
plt.show()
```

Obtain Items from the Confusion Matrix

```
TN, FP, FN, TP = cm.ravel()
# Accuracy
accuracy = accuracy_score(y_test, y_pred_ridge)

# Sensitivity (Recall)
sensitivity = recall_score(y_test, y_pred_ridge)

# Specificity
specificity = TN / (TN + FP)

# Positive Predictive Value (Precision)
ppv = precision_score(y_test, y_pred_ridge)

# Negative Predictive Value
npv = TN / (TN + FN)

# F1 Score
f1 = f1_score(y_test, y_pred_ridge)

# Cohen's Kappa
kappa = cohen_kappa_score(y_test, y_pred_ridge)
# No Information Rate (NIR)
nir = y_test.value_counts().max() / len(y_test)

# McNemar's Test P-Value
mcnemar_result = mcnemar(cm)
mcnemar_p_value = mcnemar_result.pvalue

# Balanced Accuracy
balanced_accuracy = (sensitivity + specificity) / 2
```

Print the results

```
print(f"Confusion Matrix:\n{cm}")
print(f"Accuracy: {accuracy:.4f}")
print(f"Sensitivity: {sensitivity:.4f}")
print(f"Specificity: {specificity:.4f}")
print(f"Positive Predictive Value: {ppv:.4f}")
print(f"Negative Predictive Value: {npv:.4f}")
print(f"F1 Score: {f1:.4f}")
print(f"Cohen's Kappa: {kappa:.4f}")
print(f"No Information Rate (NIR): {nir:.4f}")
#print(f"P-Value [Acc > NIR]: {p_value_acc_nir:.7f}")
print(f"McNemar's Test P-Value: {mcnemar_p_value:.7f}")
print(f"Balanced Accuracy: {balanced_accuracy:.4f}")
```

```
Confusion Matrix:
[[22091   403]
 [ 6118  1122]]
Accuracy: 0.7807
Sensitivity: 0.1550
Specificity: 0.9821
Positive Predictive Value: 0.7357
Negative Predictive Value: 0.7831
F1 Score: 0.2560
Cohen's Kappa: 0.1871
No Information Rate (NIR): 0.7565
McNemar's Test P-Value: 0.0000000
Balanced Accuracy: 0.5685
```

Area Under the Curve

```
## Predict decision function scores for ROC
y_scores = ridge_model.decision_function(X_test)
from sklearn.linear_model import RidgeClassifier
from sklearn.datasets import load_breast_cancer
from sklearn.model_selection import train_test_split
from sklearn.metrics import roc_auc_score, roc_curve, auc
import matplotlib.pyplot as plt
# Compute ROC curve and AUC
fpr, tpr, thresholds = roc_curve(y_test, y_scores)
roc_auc = auc(fpr, tpr)

print(f"AUC-ROC: {roc_auc:.2f}")

# Plot ROC curve
plt.figure()
plt.plot(fpr, tpr, color='darkorange', lw=2, label=f'ROC curve (area = {roc_auc:.2f})')
plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic and AUC for Ridge Logistic Regression')
```

```
plt.legend(loc="lower right")
plt.show()
```

AUC-ROC: 0.71

Feature Importance Plot

To identify the top most important features, regardless of their order in the dataset, we can sort the features based on the absolute value of their coefficients.

```
# Extract coefficients
coefficients = ridge_model.coef_[0]
coefficients
```

```
array([ 0.04030855,  0.03126034, -0.01616124, -0.03223257, -0.02117255,
        0.02801032, -0.01144936, -0.09051206, -0.06532989,  0.03141421,
       -0.08052634,  0.01090955, -0.03847197, -0.11714778, -0.0171607 ,
       -0.16506959,  0.06061165, -0.00779274, -0.03196569,  0.00779274,
        0.01544257, -0.08216401,  0.08101169, -0.00146062,  0.11856664,
        0.0545619 ,  0.22288899,  0.02391631, -0.00779274,  0.01981237])
```

```
feature_names = loan.drop(loan.columns[30], axis=1).columns
feature_names
```

```
Index(['loan_limit', 'Gender', 'approv_in_adv', 'loan_type', 'loan_purpose',
       'Credit_Worthiness', 'open_credit', 'business_or_commercial',
       'loan_amount', 'rate_of_interest', 'Interest_rate_spread',
       'Upfront_charges', 'term', 'Neg_ammortization', 'interest_only',
       'lump_sum_payment', 'property_value', 'construction_type',
       'occupancy_type', 'Secured_by', 'total_units', 'income', 'credit_type',
       'Credit_Score', 'co-applicant_credit_type', 'age', 'LTV', 'Region',
       'Security_Type', 'Status'],
      dtype='object')
```

```
# Create a DataFrame to hold feature names and their coefficients
feature_importance = pd.DataFrame({
    'Feature': feature_names,
    'Coefficient': coefficients
})
feature_importance
```

	Feature	Coefficient
0	loan_limit	0.040309
1	Gender	0.031260
2	approv_in_adv	-0.016161
3	loan_type	-0.032233
4	loan_purpose	-0.021173
5	Credit_Worthiness	0.028010
6	open_credit	-0.011449
7	business_or_commercial	-0.090512
8	loan_amount	-0.065330
9	rate_of_interest	0.031414

10	Interest_rate_spread	-0.080526
11	Upfront_charges	0.010910
12	Term	-0.038472
13	Neg_ammortization	-0.117148
14	interest_only	-0.017161
15	lump_sum_payment	-0.165070
16	property_value	0.060612
17	construction_type	-0.007793
18	occupancy_type	-0.031966
19	Secured_by	0.007793
20	total_units	0.015443
21	Income	-0.082164
22	credit_type	0.081012
23	Credit_Score	-0.001461
24	co-applicant_credit_type	0.118567
25	Age	0.054562
26	LTV	0.222889
27	Region	0.023916
28	Security_Type	-0.007793
29	Status	0.019812

Sort the DataFrame by the absolute value of coefficients

```
feature_importance['Absolute Coefficient'] = feature_importance['Coefficient'].abs()
feature_importance = feature_importance.sort_values(by='Absolute Coefficient',
ascending=False)

# Select the top 10 most important features
top_features = feature_importance.head(10)
top_features
```

	Feature	Coefficient	Absolute Coefficient
26	LTV	0.222889	0.222889
15	lump_sum_payment	-0.165070	0.165070
24	co-applicant_credit_type	0.118567	0.118567
13	Neg_ammortization	-0.117148	0.117148
7	business_or_commercial	-0.090512	0.090512
21	income	-0.082164	0.082164
22	credit_type	0.081012	0.081012
10	Interest_rate_spread	-0.080526	0.080526
8	loan_amount	-0.065330	0.065330
16	property_value	0.060612	0.060612

```
# Plot feature importance
plt.figure(figsize=(10, 6))
plt.barh(range(10), top_features['Coefficient'], align='center')
plt.yticks(np.arange(10), top_features['Feature'])
plt.xlabel('Coefficient Value')
plt.title('Top 10 Most Important Features')
plt.gca().invert_yaxis()
plt.show()
```

Alternatively

Obtain the Coefficients and their Odds Ratios

```
# Coefficients and Odds Ratios
coefficients = ridge_model.coef_[0]
odds_ratios = np.exp(coefficients)


# Display feature importance using coefficients and odds ratios
feature_importance = pd.DataFrame({
    'Feature': X.columns,
    'Coefficient': coefficients,
    'Odds Ratio': odds_ratios
})
print("\nFeature Importance (Coefficient and Odds Ratio):")
print(feature_importance.sort_values(by='Coefficient', ascending=False))
```

```
Feature Importance (Coefficient and Odds Ratio):
                  Feature  Coefficient  Odds Ratio
26                    LTV     0.222889    1.249682
24  co-applicant_credit_type     0.118567    1.125882
22             credit_type     0.081012    1.084384
16          property_value     0.060612    1.062486
25                     age     0.054562    1.056078
0              loan_limit     0.040309    1.041132
9          rate_of_interest     0.031414    1.031913
1                  Gender     0.031260    1.031754
5         Credit_Worthiness     0.028010    1.028406
27                 Region     0.023916    1.024205
29                  dtir1     0.019812    1.020010
20            total_units     0.015443    1.015562
11          Upfront_charges     0.010910    1.010969
19             Secured_by     0.007793    1.007823
23           Credit_Score    -0.001461    0.998540
17        construction_type    -0.007793    0.992238
28           Security_Type    -0.007793    0.992238
6              open_credit    -0.011449    0.988616
2            approv_in_adv    -0.016161    0.983969
14           interest_only    -0.017161    0.982986
4            loan_purpose    -0.021173    0.979050
18          occupancy_type    -0.031966    0.968540
3              loan_type    -0.032233    0.968281
12                  term    -0.038472    0.962259
8             loan_amount    -0.065330    0.936758
10     Interest_rate_spread    -0.080526    0.922631
21                 income    -0.082164    0.921121
7    business_or_commercial    -0.090512    0.913463
13        Neg_ammortization    -0.117148    0.889454
15         lump_sum_payment    -0.165070    0.847835
```

Compute Permutation Importance

Permutation importance is a model-agnostic technique used to estimate the importance of features in a machine learning model. It measures the decrease in a model's performance when the values of a single feature are randomly shuffled. This shuffling breaks the relationship between the feature and the target, and the resulting drop in performance is indicative of the feature's importance.

```
from sklearn.inspection import permutation_importance
from sklearn.feature_selection import RFE
# Permutation Importance
perm_importance = permutation_importance(ridge_model, X_test, y_test,
n_repeats=30, random_state=42, n_jobs=-1)
perm_importance_df = pd.DataFrame({
    'Feature': X.columns,
    'Importance Mean': perm_importance.importances_mean,
    'Importance Std': perm_importance.importances_std
})
print("\nPermutation Importance:")
perm_importance_df.sort_values(by='Importance Mean', ascending=False)
```

Permutation Importance:

	Feature	Importance Mean	Importance Std
15	lump_sum_payment	0.023748	0.000570
13	Neg_ammortization	0.009786	0.000723
10	Interest_rate_spread	0.005008	0.000737
7	business_or_commercial	0.004545	0.000451
24	co-applicant_credit_type	0.002967	0.000567
26	LTV	0.002426	0.000597
5	Credit_Worthiness	0.002150	0.000220
16	property_value	0.001799	0.000492
21	income	0.001484	0.000321
8	loan_amount	0.001430	0.000351
22	credit_type	0.001294	0.000417
0	loan_limit	0.001260	0.000290
12	term	0.000960	0.000335
25	age	0.000580	0.000471
2	approv_in_adv	0.000287	0.000158
27	Region	0.000150	0.000186
20	total_units	0.000137	0.000118
18	occupancy_type	0.000130	0.000241
11	Upfront_charges	0.000128	0.000176
14	interest_only	0.000121	0.000177
6	open_credit	0.000085	0.000040
19	Secured_by	0.000070	0.000066
17	construction_type	0.000070	0.000066
28	Security_Type	0.000070	0.000066
3	loan_type	0.000059	0.000241

23	Credit_Score	0.000050	0.000055
1	Gender	0.000045	0.000318
29	dtir1	-0.000035	0.000251
4	loan_purpose	-0.000196	0.000230
9	rate_of_interest	-0.000271	0.000237

```
# Select the top 10 most important features
perm_importance_df = perm_importance_df.sort_values(by='Importance Mean',
ascending=False)
top_features = perm_importance_df.head(10)
top_features
```

	Feature	Importance Mean	Importance Std
15	lump_sum_payment	0.023748	0.000570
13	Neg_ammortization	0.009786	0.000723
10	Interest_rate_spread	0.005008	0.000737
7	business_or_commercial	0.004545	0.000451
24	co-applicant_credit_type	0.002967	0.000567
26	LTV	0.002426	0.000597
5	Credit_Worthiness	0.002150	0.000220
16	property_value	0.001799	0.000492
21	income	0.001484	0.000321
8	loan_amount	0.001430	0.000351

```
# Plot feature importance
plt.figure(figsize=(10, 6))
plt.barh(top_features['Feature'], top_features['Importance Mean'],
xerr=top_features['Importance Std'], align='center')
plt.xlabel('Permutation Importance Mean')
plt.title('Top 10 Most Important Features')
plt.gca().invert_yaxis()  # Invert y-axis to have the most important feature at the top
plt.show()
```

Tune the Parameter for Ridge Logistic Regression

```
from sklearn.linear_model import RidgeClassifier
from sklearn.model_selection import GridSearchCV
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
# Define parameter grid
param_grid = {
    'alpha': [0.1, 1.0, 10.0, 100.0]  # You can expand this range
}
# Initialize GridSearchCV
grid_search = GridSearchCV(estimator=clf, param_grid=param_grid, cv=5,
scoring='accuracy')

# Fit GridSearchCV
grid_search.fit(X_train, y_train)
```

```
# Get best parameters and best score
best_params = grid_search.best_params_
best_score = grid_search.best_score_

print(f"Best parameters: {best_params}")
print(f"Best cross-validation score: {best_score:.2f}")

# Evaluate on the test set
best_ridge = grid_search.best_estimator_
y_pred = best_ridge.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print(f"Test set accuracy: {accuracy:.2f}")
```

Best parameters: {'alpha': 10.0}
Best cross-validation score: 0.78
Test set accuracy: 0.78

Report the Results

```
report = classification_report(y_test, y_pred, output_dict=True)
import pandas as pd
from sklearn.metrics import classification_report
report = classification_report(y_test, y_pred, target_names=['no', 'yes'],
output_dict=True)
report_df = pd.DataFrame(report).transpose()
report_df
```

	precision	recall	f1-score	support
no	0.783416	0.979461	0.870538	22494.000000
yes	0.713222	0.158702	0.259632	7240.000000
accuracy	0.779613	0.779613	0.779613	0.779613
macro avg	0.748319	0.569081	0.565085	29734.000000
weighted avg	0.766324	0.779613	0.721787	29734.000000

The model's performance can be assessed through various metrics including precision, recall, f1-score, and support for each class.

For class 0, the model achieved a precision of approximately 0.783, indicating that when the model predicts class 0, it is correct about 78.3% of the time. The recall for class 0 is notably high at 0.979, meaning that the model successfully identifies 97.9% of all actual instances of class 0. This high recall suggests that the model is very effective at detecting class 0, though this is in contrast with the f1-score, which stands at 0.871. The f1-score balances precision and recall, providing a more holistic measure of the model’s performance. The support for class 0 is substantial, with 22,494 instances, which shows that it is the more frequent class in the dataset.

Conversely, for class 1, the model has a precision of 0.713, which is lower than that for class 0. This means that when the model predicts class 1, it is correct approximately 71.3% of the time. However, the recall for class 1 is significantly lower at 0.159, indicating that the model only identifies 15.9% of the actual class 1 instances. This low recall reveals that the model struggles to correctly identify class 1, which affects the f1-score for this class, resulting in a value of 0.260. The support for class 1 is 7,240 instances, reflecting that it is less frequent compared to class 0.

The overall accuracy of the model is 0.780, which suggests that the model correctly classifies 78.0% of the instances. This metric provides a general sense of how often the model is correct across both classes. The accuracy is calculated as the ratio of the total number of correct predictions to the total number of predictions made.

The macro average provides an overall summary by averaging the performance metrics of both classes without considering class imbalance. For the macro average, the precision is 0.748, recall is 0.569, and the f1-score is 0.565. These averages indicate the model's performance across both classes, giving equal weight to each class irrespective of their support.

The weighted average takes into account the support for each class, providing a weighted summary of the performance metrics. For the weighted average, precision is 0.766, recall is 0.780, and f1-score is 0.722. These values reflect the model's performance adjusted for the class distribution in the dataset.

In summary, the model performs well in identifying the majority class (class 0) with high recall but faces challenges with detecting the minority class (class 1), which affects the overall f1-score and recall for that class. The balanced accuracy and the weighted averages provide a more nuanced view of the model's performance across the imbalanced dataset.

Loss Function for Logistic Regression

In practice, while RidgeClassifier in scikit-learn does not provide a direct method to compute the loss function, you can still interpret and assess its performance effectively using other metrics and methods. The absence of a direct loss function calculation does not necessarily undermine the value of the RidgeClassifier; instead, you can rely on these alternative metrics:

Alternative Metrics to Evaluate Model Performance Accuracy: Measures the proportion of correctly classified samples. It gives a high-level view of how well the model is performing overall.

Precision, *Recall*, and *F1-Score*: Provide more detailed insights into model performance, especially useful when dealing with imbalanced classes. These metrics help understand the model's ability to correctly identify each class.

Confusion Matrix: Shows the counts of true positives, true negatives, false positives, and false negatives, giving a complete picture of classification performance.

Cross-Validation Scores: Using cross-validation can give a more robust estimate of the model's performance by evaluating it on multiple subsets of the data.

Model Coefficients: While not a loss metric, analyzing the model coefficients can provide insights into feature importance and how the model is interpreting the data.

Comparing Loss Functions

If one still want to compare the RidgeClassifier's performance with loss metrics typically associated with other models (like logistic regression with L2 regularization), consider using LogisticRegression with penalty='l2' as a proxy. You can compare:

Logistic Loss: Can be computed using LogisticRegression with L2 penalty and the log_loss function.

Ridge Penalty: Regularization term can be calculated manually if needed.

```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import log_loss

# Initialize Logistic Regression with L2 penalty (Ridge)
ridge_logistic = LogisticRegression(penalty='l2', C=1/0.1, solver='lbfgs',
max_iter=1000)

# Fit the model
ridge_logistic.fit(X_train, y_train)

# Predict probabilities
y_prob = ridge_logistic.predict_proba(X_test)[:, 1]

# Calculate Logistic Loss
logistic_loss = log_loss(y_test, y_prob)

print(f"Logistic Loss: {logistic_loss}")
```

```
Logistic Loss: 0.4960475506087793
```

The Logistic Loss value of 0.496 indicates the performance of your Ridge Logistic Regression model in terms of prediction error. Logistic Loss, also known as Binary Cross-Entropy Loss, measures the difference between the actual labels and the

predicted probabilities and is commonly used for binary classification problems. An ideal Logistic Loss value is 0, representing perfect predictions, while higher values suggest that the model's predictions are further from the actual class labels. A loss of 0.496 is close to 0.5, a baseline often associated with random predictions, suggesting that the model is not significantly outperforming random guessing but is not performing poorly either. To determine if this loss is acceptable or needs improvement, it is crucial to compare it against baseline models, consider additional metrics like accuracy, precision, recall, and F1-score for a more comprehensive view, and evaluate it against domain-specific standards. If the performance is lacking, actions such as experimenting with different hyperparameters, enhancing feature selection or engineering, and exploring alternative models may help improve the results. Overall, a Logistic Loss of 0.496 indicates that there is room for improvement in the model's predictive accuracy, and further refinements or alternative approaches should be considered to enhance performance.

Artificial Neural Networks

```
### Install TensorFlow using the code below
### pip install tensorflow
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
# Create the ANN model
ann = Sequential()
ann.add(Dense(32, input_dim=X_train.shape[1], activation='relu'))
ann.add(Dense(16, activation='relu'))
ann.add(Dense(1, activation='sigmoid'))
# Compile the model
ann.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])

# Train the model
ann.fit(X_train, y_train, epochs=16, batch_size=10, validation_split=0.2)
```

```
Epoch 1/16
9515/9515 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 20s 2ms/step -
accuracy: 0.9049 - loss: 0.2021 - val_accuracy: 0.9973 - val_loss: 0.0137
Epoch 2/16
9515/9515 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 20s 2ms/step -
accuracy: 0.9981 - loss: 0.0100 - val_accuracy: 0.9979 - val_loss: 0.0114
Epoch 3/16
9515/9515 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 18s 2ms/step -
accuracy: 0.9991 - loss: 0.0046 - val_accuracy: 0.9991 - val_loss: 0.0044
Epoch 4/16
9515/9515 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 21s 2ms/step -
accuracy: 0.9995 - loss: 0.0031 - val_accuracy: 0.9971 - val_loss: 0.0095
```

Epoch 5/16
9515/9515 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ **21s** 2ms/step - accuracy: 0.9994 - loss: 0.0022 - val_accuracy: 0.9997 - val_loss: 0.0036
Epoch 6/16
9515/9515 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ **21s** 2ms/step - accuracy: 0.9997 - loss: 0.0016 - val_accuracy: 0.9996 - val_loss: 0.0029
Epoch 7/16
9515/9515 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ **43s** 2ms/step - accuracy: 0.9997 - loss: 0.0024 - val_accuracy: 0.9997 - val_loss: 0.0023
Epoch 8/16
9515/9515 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ **19s** 2ms/step - accuracy: 0.9997 - loss: 0.0020 - val_accuracy: 0.9995 - val_loss: 0.0034
Epoch 9/16
9515/9515 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ **20s** 2ms/step - accuracy: 0.9998 - loss: 9.2160e-04 - val_accuracy: 0.9995 - val_loss: 0.0036
Epoch 10/16
9515/9515 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ **18s** 2ms/step - accuracy: 0.9998 - loss: 8.4222e-04 - val_accuracy: 0.9995 - val_loss: 0.0042
Epoch 11/16
9515/9515 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ **20s** 2ms/step - accuracy: 0.9998 - loss: 7.7810e-04 - val_accuracy: 0.9997 - val_loss: 0.0032
Epoch 12/16
9515/9515 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ **19s** 2ms/step - accuracy: 0.9998 - loss: 0.0013 - val_accuracy: 0.9997 - val_loss: 0.0029
Epoch 13/16
9515/9515 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ **16s** 2ms/step - accuracy: 0.9999 - loss: 5.0679e-04 - val_accuracy: 0.9994 - val_loss: 0.0045
Epoch 14/16
9515/9515 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ **18s** 2ms/step - accuracy: 0.9998 - loss: 0.0010 - val_accuracy: 0.9997 - val_loss: 0.0034
Epoch 15/16
9515/9515 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ **18s** 2ms/step - accuracy: 0.9999 - loss: 8.9040e-04 - val_accuracy: 0.9997 - val_loss: 0.0030
Epoch 16/16
9515/9515 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ **19s** 2ms/step - accuracy: 0.9999 - loss: 9.4910e-04 - val_accuracy: 0.9997 - val_loss: 0.0030
<keras.src.callbacks.history.History at 0x23183b6c350>

```
#pip install pydot graphviz
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from tensorflow.keras.utils import plot_model

# Plot the model architecture
plot_model(ann, to_file='neural_network.png', show_shapes=True, show_layer_names=True)
```

You must install graphviz (see instructions at https://graphviz.gitlab.io/download/) for `plot_model` to work.

```
# Make predictions
y_pred_ann = (ann.predict(X_test) > 0.5).astype("int32")
```

930/930 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ **1s** 1ms/step

```
# Evaluate the model
print("Artificial Neural Networks Report:")
classification_report(y_test, y_pred_ann, target_names=['no', 'yes'], output_dict=True)
```

```
Artificial Neural Networks Report:
{'no': {'precision': 0.9999110794949315,
  'recall': 0.9998221748021695,
  'f1-score': 0.9998666251722759,
  'support': 22494.0},
 'yes': {'precision': 0.9994476663904999,
  'recall': 0.9997237569060774,
  'f1-score': 0.9995856925838973,
  'support': 7240.0},
 'accuracy': 0.9997982108024484,
 'macro avg': {'precision': 0.9996793729427157,
  'recall': 0.9997729658541235,
  'f1-score': 0.9997261588780866,
  'support': 29734.0},
 'weighted avg': {'precision': 0.9997982419730345,
  'recall': 0.9997982108024484,
  'f1-score': 0.999798220250642,
  'support': 29734.0}}
```

View the Performance Metrics above as data Frame

```
import pandas as pd
from sklearn.metrics import classification_report
report = classification_report(y_test, y_pred_ann, target_names=['no', 'yes'],
output_dict=True)
report_df = pd.DataFrame(report).transpose()
report_df
```

	precision	recall	f1-score	support
no	0.999911	0.999822	0.999867	22494.000000
yes	0.999448	0.999724	0.999586	7240.000000
accuracy	0.999798	0.999798	0.999798	0.999798
macro avg	0.999679	0.999773	0.999726	29734.000000
weighted avg	0.999798	0.999798	0.999798	29734.000000

Confusion Matrix

```
import numpy as np
import pandas as pd
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import confusion_matrix, accuracy_score, precision_score,
recall_score, f1_score, cohen_kappa_score, roc_auc_score
from sklearn.model_selection import train_test_split
from statsmodels.stats.contingency_tables import mcnemar
```

```
# Confusion matrix
cm = confusion_matrix(y_test, y_pred_ann)
cm
```

```
array([[22490,     4],
       [    2,  7238]], dtype=int64)
```

Plot the Confusion Matrix for Artificial Neural Networks

```
%matplotlib inline
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.metrics import confusion_matrix
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
import pandas as pd
import numpy as np

plt.figure(figsize=(9, 6))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', xticklabels=["No", "Yes"],
yticklabels=["No", "Yes"])
plt.xlabel('Predicted Load Default Results')
plt.ylabel('Actual Loan Default Results')
plt.title('Confusion Matrix Plot for the Artificial Neural Networks')
plt.show()
```

```
TN, FP, FN, TP = cm.ravel()
# Accuracy
accuracy = accuracy_score(y_test, y_pred_ann)

# Sensitivity (Recall)
sensitivity = recall_score(y_test, y_pred_ann)

# Specificity
specificity = TN / (TN + FP)

# Positive Predictive Value (Precision)
ppv = precision_score(y_test, y_pred_ann)

# Negative Predictive Value
npv = TN / (TN + FN)

# F1 Score
f1 = f1_score(y_test, y_pred_ann)

# Cohen's Kappa
kappa = cohen_kappa_score(y_test, y_pred_ann)

# No Information Rate (NIR)
nir = y_test.value_counts().max() / len(y_test)
```

```
# McNemar's Test P-Value
mcnemar_result = mcnemar(cm)
mcnemar_p_value = mcnemar_result.pvalue

# Balanced Accuracy
balanced_accuracy = (sensitivity + specificity) / 2

print(f"Confusion Matrix:\n{cm}")
print(f"Accuracy: {accuracy:.4f}")
print(f"Sensitivity: {sensitivity:.4f}")
print(f"Specificity: {specificity:.4f}")
print(f"Positive Predictive Value: {ppv:.4f}")
print(f"Negative Predictive Value: {npv:.4f}")
print(f"F1 Score: {f1:.4f}")
print(f"Cohen's Kappa: {kappa:.4f}")
print(f"No Information Rate (NIR): {nir:.4f}")
#print(f"P-Value [Acc > NIR]: {p_value_acc_nir:.7f}")
print(f"McNemar's Test P-Value: {mcnemar_p_value:.7f}")
print(f"Balanced Accuracy: {balanced_accuracy:.4f}")
```

```
Confusion Matrix:
[[22490     4]
 [    2  7238]]
Accuracy: 0.9998
Sensitivity: 0.9997
Specificity: 0.9998
Positive Predictive Value: 0.9994
Negative Predictive Value: 0.9999
F1 Score: 0.9996
Cohen's Kappa: 0.9995
No Information Rate (NIR): 0.7565
McNemar's Test P-Value: 0.6875000
Balanced Accuracy: 0.9998
```

ROC and AUC for Artificial Neural Network

```
## Predict decision function scores for ROC
y_scores = ann.predict(X_test)
```

930/930 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ **1s** 1ms/step

```
from sklearn.linear_model import RidgeClassifier
from sklearn.datasets import load_breast_cancer
from sklearn.model_selection import train_test_split
from sklearn.metrics import roc_auc_score, roc_curve, auc
import matplotlib.pyplot as plt
# Compute ROC curve and AUC
fpr, tpr, thresholds = roc_curve(y_test, y_scores)
roc_auc = auc(fpr, tpr)

print(f"AUC-ROC: {roc_auc:.2f}")
```

```
# Plot ROC curve
plt.figure()
plt.plot(fpr, tpr, color='darkorange', lw=2, label=f'ROC curve (area = {roc_auc:.2f})')
plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic and AUC for Artificial Neural Networks')
plt.legend(loc="lower right")
plt.show()
```

AUC-ROC: 1.00

Loss Function

```
history = ann.fit(X_train, y_train, validation_data=(X_test, y_test), epochs=10,
batch_size=32)

# Accessing the loss function value for the final epoch
final_loss = history.history['loss'][-1]

# Accessing the loss function value for the validation set (if applicable)
final_test_loss = history.history['val_loss'][-1]

print(f"Final Training Loss: {final_loss}")
print(f"Final Testing Loss: {final_test_loss}")
```

Epoch 1/10
3717/3717 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ **8s** 2ms/step - accuracy: 0.9998 - loss: 0.0012 - val_accuracy: 0.9999 - val_loss: 6.5979e-04
Epoch 2/10
3717/3717 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ **11s** 2ms/step - accuracy: 1.0000 - loss: 1.2731e-04 - val_accuracy: 0.9999 - val_loss: 8.4448e-04
Epoch 3/10
3717/3717 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ **7s** 2ms/step - accuracy: 0.9999 - loss: 2.9164e-04 - val_accuracy: 0.9998 - val_loss: 9.8478e-04
Epoch 4/10
3717/3717 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ **8s** 2ms/step - accuracy: 1.0000 - loss: 5.3929e-05 - val_accuracy: 0.9998 - val_loss: 8.4209e-04
Epoch 5/10
3717/3717 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ **7s** 2ms/step - accuracy: 1.0000 - loss: 1.7276e-04 - val_accuracy: 0.9998 - val_loss: 8.5402e-04
Epoch 6/10
3717/3717 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ **12s** 2ms/step - accuracy: 0.9998 - loss: 3.6180e-04 - val_accuracy: 0.9998 - val_loss: 8.9003e-04
Epoch 7/10
3717/3717 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ **15s** 4ms/step - accuracy: 1.0000 - loss: 3.9234e-05 - val_accuracy: 0.9998 - val_loss: 9.0583e-04
Epoch 8/10

3717/3717 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ **12s** 3ms/step - accuracy: 1.0000 - loss: 1.0567e-05 - val_accuracy: 0.9996 - val_loss: 0.0035
Epoch 9/10
3717/3717 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ **13s** 3ms/step - accuracy: 0.9998 - loss: 6.0680e-04 - val_accuracy: 0.9999 - val_loss: 4.0745e-04
Epoch 10/10
3717/3717 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ **11s** 3ms/step - accuracy: 1.0000 - loss: 6.8537e-06 - val_accuracy: 0.9999 - val_loss: 2.3437e-04
Final Training Loss: 6.144988219602965e-06
Final Validation Loss: 0.00023436832998413593

The final training loss of the Artificial Neural Network (ANN) model is extremely low, at approximately $6.14\times10-66.14\times10-6$, indicating that the model has fit the training data very closely. This suggests that the model has learned the patterns in the training data exceptionally well, achieving near-perfect performance on this dataset. However, the final validation loss, while still low at approximately 0.00023, is higher than the training loss. This disparity indicates that while the model performs very well on the validation set, it is not as perfect as on the training set, which might suggest a slight overfitting. The small gap between the training and validation losses is a positive sign, showing that the model generalizes well to unseen data, but care should be taken to ensure that this overfitting does not worsen if further training is pursued. Overall, the model demonstrates strong performance with minimal overfitting.

Extreme Gradient Boosting

```
## install XGBoost using the command below
##pip install xgboost.
import xgboost as xgb
# Create and train the XGBoost model
xgb_model = xgb.XGBClassifier()
xgb_model.fit(X_train, y_train)

# Make predictions
y_pred_xgb = xgb_model.predict(X_test)
# Evaluate the model
print("Extreme Gradient Boosting Report:")
report = classification_report(y_test, y_pred_xgb, target_names=['no', 'yes'],
output_dict=True)
report_df = pd.DataFrame(report).transpose()
report_df
```

Extreme Gradient Boosting Report:

	precision	recall	f1-score	support
no	1.0	1.0	1.0	22494.0
yes	1.0	1.0	1.0	7240.0

accuracy	1.0	1.0	1.0	1.0
macro avg	1.0	1.0	1.0	29734.0
weighted avg	1.0	1.0	1.0	29734.0

```
import numpy as np
import pandas as pd
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import confusion_matrix, accuracy_score, precision_score,
recall_score, f1_score, cohen_kappa_score, roc_auc_score
from sklearn.model_selection import train_test_split
from statsmodels.stats.contingency_tables import mcnemar

# Confusion matrix
cm = confusion_matrix(y_test, y_pred_xgb)
cm

%matplotlib inline
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.metrics import confusion_matrix
import pandas as pd
import numpy as np

plt.figure(figsize=(9, 6))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', xticklabels=["No", "Yes"],
yticklabels=["No", "Yes"])
plt.xlabel('Predicted Load Default Results')
plt.ylabel('Actual Loan Default Results')
plt.title('Confusion Matrix Plot for the Extreme Gradient Boosting')
plt.show()
```

```
TN, FP, FN, TP = cm.ravel()
# Accuracy
accuracy = accuracy_score(y_test, y_pred_xgb)

# Sensitivity (Recall)
sensitivity = recall_score(y_test, y_pred_xgb)

# Specificity
specificity = TN / (TN + FP)

# Positive Predictive Value (Precision)
ppv = precision_score(y_test, y_pred_xgb)

# Negative Predictive Value
npv = TN / (TN + FN)

# F1 Score
f1 = f1_score(y_test, y_pred_xgb)
```

```
# Cohen's Kappa
kappa = cohen_kappa_score(y_test, y_pred_xgb)

# No Information Rate (NIR)
nir = y_test.value_counts().max() / len(y_test)

# McNemar's Test P-Value
mcnemar_result = mcnemar(cm)
mcnemar_p_value = mcnemar_result.pvalue

# Balanced Accuracy
balanced_accuracy = (sensitivity + specificity) / 2

print(f"Confusion Matrix:\n{cm}")
print(f"Accuracy: {accuracy:.4f}")
print(f"Sensitivity: {sensitivity:.4f}")
print(f"Specificity: {specificity:.4f}")
print(f"Positive Predictive Value: {ppv:.4f}")
print(f"Negative Predictive Value: {npv:.4f}")
print(f"F1 Score: {f1:.4f}")
print(f"Cohen's Kappa: {kappa:.4f}")
print(f"No Information Rate (NIR): {nir:.4f}")
#print(f"P-Value [Acc > NIR]: {p_value_acc_nir:.7f}")
print(f"McNemar's Test P-Value: {mcnemar_p_value:.7f}")
print(f"Balanced Accuracy: {balanced_accuracy:.4f}")
```

```
Confusion Matrix:
[[22494     0]
 [    0  7240]]
Accuracy: 1.0000
Sensitivity: 1.0000
Specificity: 1.0000
Positive Predictive Value: 1.0000
Negative Predictive Value: 1.0000
F1 Score: 1.0000
Cohen's Kappa: 1.0000
No Information Rate (NIR): 0.7565
McNemar's Test P-Value: 1.0000000
Balanced Accuracy: 1.0000
```

ROC and AUC for Extreme Gradient Boosting

```
# Get prediction probabilities for the positive class
y_scores = xgb_model.predict_proba(X_test)[:, 1]

# Compute ROC curve and AUC
fpr, tpr, thresholds = roc_curve(y_test, y_scores)
roc_auc = auc(fpr, tpr)

# Plot the ROC curve
```

```
plt.figure(figsize=(8, 6))
plt.plot(fpr, tpr, color='darkorange', lw=2, label=f'ROC curve (AUC =
{roc_auc:.2f})')
plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic (ROC) Curve and AUC for XGBoost')
plt.legend(loc="lower right")
plt.show()

# Print AUC score
print(f'AUC: {roc_auc:.2f}')
```

AUC: 1.00

Loss Function

```
# Convert the data into DMatrix, which is the optimized data structure that XGBoost
uses
dtrain = xgb.DMatrix(X_train, label=y_train)
dtest = xgb.DMatrix(X_test, label=y_test)

# Define parameters for XGBoost
params = {
    'objective': 'binary:logistic',  # This specifies the loss function
    'eval_metric': 'logloss',        # This specifies the metric we want to monitor
}

# Watchlist to evaluate performance on the training and validation data
watchlist = [(dtrain, 'train'), (dtest, 'eval')]

# Train the model and evaluate logloss
bst = xgb.train(params, dtrain, num_boost_round=100, evals=watchlist,
early_stopping_rounds=10)

# To access the final evaluation metric (e.g., logloss) on the test data
evals_result = bst.eval(dtest)
print(f"Final Logloss on Test Data: {evals_result}")
```

```
[0]     train-logloss:0.32070     eval-logloss:0.31916
[1]     train-logloss:0.21720     eval-logloss:0.21626
[2]     train-logloss:0.15258     eval-logloss:0.15197
[3]     train-logloss:0.10917     eval-logloss:0.10875
[4]     train-logloss:0.07898     eval-logloss:0.07868
[5]     train-logloss:0.05754     eval-logloss:0.05733
[6]     train-logloss:0.04213     eval-logloss:0.04198
[7]     train-logloss:0.03095     eval-logloss:0.03084
[8]     train-logloss:0.02279     eval-logloss:0.02271
```

```
[9]     train-logloss:0.01682     eval-logloss:0.01676
[10]    train-logloss:0.01242     eval-logloss:0.01238
[11]    train-logloss:0.00919     eval-logloss:0.00916
[12]    train-logloss:0.00680     eval-logloss:0.00678
[13]    train-logloss:0.00504     eval-logloss:0.00503
[14]    train-logloss:0.00374     eval-logloss:0.00373
[15]    train-logloss:0.00278     eval-logloss:0.00278
[16]    train-logloss:0.00207     eval-logloss:0.00206
[17]    train-logloss:0.00154     eval-logloss:0.00154
[18]    train-logloss:0.00116     eval-logloss:0.00115
[19]    train-logloss:0.00087     eval-logloss:0.00087
[20]    train-logloss:0.00066     eval-logloss:0.00066
[21]    train-logloss:0.00049     eval-logloss:0.00049
[22]    train-logloss:0.00038     eval-logloss:0.00038
[23]    train-logloss:0.00029     eval-logloss:0.00029
[24]    train-logloss:0.00022     eval-logloss:0.00022
[25]    train-logloss:0.00017     eval-logloss:0.00017
[26]    train-logloss:0.00013     eval-logloss:0.00013
[27]    train-logloss:0.00011     eval-logloss:0.00011
[28]    train-logloss:0.00008     eval-logloss:0.00008
[29]    train-logloss:0.00007     eval-logloss:0.00007
[30]    train-logloss:0.00006     eval-logloss:0.00006
[31]    train-logloss:0.00005     eval-logloss:0.00005
[32]    train-logloss:0.00004     eval-logloss:0.00004
[33]    train-logloss:0.00004     eval-logloss:0.00004
[34]    train-logloss:0.00003     eval-logloss:0.00003
[35]    train-logloss:0.00003     eval-logloss:0.00003
[36]    train-logloss:0.00003     eval-logloss:0.00003
[37]    train-logloss:0.00003     eval-logloss:0.00003
[38]    train-logloss:0.00002     eval-logloss:0.00003
[39]    train-logloss:0.00002     eval-logloss:0.00002
[40]    train-logloss:0.00002     eval-logloss:0.00002
[41]    train-logloss:0.00002     eval-logloss:0.00002
[42]    train-logloss:0.00002     eval-logloss:0.00002
[43]    train-logloss:0.00002     eval-logloss:0.00002
[44]    train-logloss:0.00002     eval-logloss:0.00002
[45]    train-logloss:0.00002     eval-logloss:0.00002
[46]    train-logloss:0.00002     eval-logloss:0.00002
[47]    train-logloss:0.00002     eval-logloss:0.00002
[48]    train-logloss:0.00002     eval-logloss:0.00002
[49]    train-logloss:0.00002     eval-logloss:0.00002
[50]    train-logloss:0.00002     eval-logloss:0.00002
[51]    train-logloss:0.00002     eval-logloss:0.00002
[52]    train-logloss:0.00002     eval-logloss:0.00002
[53]    train-logloss:0.00002     eval-logloss:0.00002
[54]    train-logloss:0.00002     eval-logloss:0.00002
[55]    train-logloss:0.00002     eval-logloss:0.00002
[56]    train-logloss:0.00002     eval-logloss:0.00002
[57]    train-logloss:0.00002     eval-logloss:0.00002
[58]    train-logloss:0.00002     eval-logloss:0.00002
```

[59] train-logloss:0.00002 eval-logloss:0.00002
[60] train-logloss:0.00002 eval-logloss:0.00002
[61] train-logloss:0.00002 eval-logloss:0.00002
[62] train-logloss:0.00002 eval-logloss:0.00002
[63] train-logloss:0.00002 eval-logloss:0.00002
[64] train-logloss:0.00002 eval-logloss:0.00002
[65] train-logloss:0.00002 eval-logloss:0.00002
[66] train-logloss:0.00002 eval-logloss:0.00002
[67] train-logloss:0.00002 eval-logloss:0.00002
[68] train-logloss:0.00002 eval-logloss:0.00002
[69] train-logloss:0.00002 eval-logloss:0.00002
[70] train-logloss:0.00002 eval-logloss:0.00002
[71] train-logloss:0.00002 eval-logloss:0.00002
[72] train-logloss:0.00002 eval-logloss:0.00002
[73] train-logloss:0.00002 eval-logloss:0.00002
[74] train-logloss:0.00002 eval-logloss:0.00002
[75] train-logloss:0.00002 eval-logloss:0.00002
[76] train-logloss:0.00002 eval-logloss:0.00002
[77] train-logloss:0.00002 eval-logloss:0.00002
[78] train-logloss:0.00002 eval-logloss:0.00002
[79] train-logloss:0.00002 eval-logloss:0.00002
[80] train-logloss:0.00002 eval-logloss:0.00002
[81] train-logloss:0.00002 eval-logloss:0.00002
[82] train-logloss:0.00002 eval-logloss:0.00002
[83] train-logloss:0.00002 eval-logloss:0.00002
[84] train-logloss:0.00002 eval-logloss:0.00002
[85] train-logloss:0.00002 eval-logloss:0.00002
[86] train-logloss:0.00002 eval-logloss:0.00002
[87] train-logloss:0.00002 eval-logloss:0.00002
[88] train-logloss:0.00002 eval-logloss:0.00002
[89] train-logloss:0.00002 eval-logloss:0.00002
[90] train-logloss:0.00002 eval-logloss:0.00002
[91] train-logloss:0.00002 eval-logloss:0.00002
[92] train-logloss:0.00002 eval-logloss:0.00002
[93] train-logloss:0.00002 eval-logloss:0.00002
[94] train-logloss:0.00002 eval-logloss:0.00002
[95] train-logloss:0.00002 eval-logloss:0.00002
[96] train-logloss:0.00002 eval-logloss:0.00002
[97] train-logloss:0.00002 eval-logloss:0.00002
[98] train-logloss:0.00002 eval-logloss:0.00002
[99] train-logloss:0.00002 eval-logloss:0.00002
Final Logloss on Test Data: [0] eval-logloss:0.00002038910254278

The final log loss on the test data for the XGBoost model is extremely low, approximately 2.04×10^{-5}2.04×10−5. This indicates that the model's predictions are highly accurate, with the predicted probabilities being very close to the actual class labels. Log loss measures the uncertainty of the predictions, penalizing incorrect classifications more heavily. A near-zero log loss suggests that the model is making

very confident and correct predictions for the majority of instances in the test set. This strong performance reflects the model's ability to generalize well to unseen data, making it a reliable tool for binary classification in this context. The low log loss is an indicator of the model's effectiveness in minimizing errors and providing high-quality predictions.

Comparative Analysis

Plotting the three ROC Cuves and AUC for the three Models in one Plot

```
y_scores_ridge = ridge_model.decision_function(X_test)
y_scores_ann = ann.predict(X_test).ravel()
y_scores_xgb = xgb_model.predict_proba(X_test)[:, 1]
```

930/930 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ **1s** 1ms/step

```
# Compute ROC curve and AUC for Ridge Logistic Regression
import matplotlib.pyplot as plt
fpr_ridge, tpr_ridge, _ = roc_curve(y_test, y_scores_ridge)
roc_auc_ridge = auc(fpr_ridge, tpr_ridge)

# Compute ROC curve and AUC for Artificial Neural Network
fpr_ann, tpr_ann, _ = roc_curve(y_test, y_scores_ann)
roc_auc_ann = auc(fpr_ann, tpr_ann)

# Compute ROC curve and AUC for XGBoost
fpr_xgb, tpr_xgb, _ = roc_curve(y_test, y_scores_xgb)
roc_auc_xgb = auc(fpr_xgb, tpr_xgb)

# Plotting the ROC curves
plt.figure(figsize=(10, 8))

# Ridge Logistic Regression ROC
plt.plot(fpr_ridge, tpr_ridge, color='blue', lw=2, label=f'Ridge Logistic Regression
(AUC = {roc_auc_ridge:.2f})')

# ANN ROC
plt.plot(fpr_ann, tpr_ann, color='green', lw=2, label=f'Artificial Neural Network
(AUC = {roc_auc_ann:.2f})')

# XGBoost ROC
plt.plot(fpr_xgb, tpr_xgb, color='red', lw=2, label=f'XGBoost (AUC =
{roc_auc_xgb:.2f})')

# Plot random chance line
plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--')

# Configure plot
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
```

```
plt.ylabel('True Positive Rate')
plt.title('ROC Curve Comparison of Ridge Logistic Regression, ANN, and XGBoost')
plt.legend(loc="lower right")
plt.grid(True)
plt.show()
import matplotlib.pyplot as plt

# Create a figure and axis object
fig, ax = plt.subplots(figsize=(10, 8))

# Plot ROC curves for the models
ax.plot(fpr_ridge, tpr_ridge, color='blue', lw=2, label=f'Ridge Logistic Regression
(AUC = {roc_auc_ridge:.2f})')
ax.plot(fpr_ann, tpr_ann, color='green', lw=2, label=f'Artificial Neural Network
(AUC = {roc_auc_ann:.2f})')
ax.plot(fpr_xgb, tpr_xgb, color='red', lw=2, label=f'XGBoost (AUC =
{roc_auc_xgb:.2f})')

# Plot random chance line
ax.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--')

# Configure plot
ax.set_xlim([0.0, 1.0])
ax.set_ylim([0.0, 1.05])
ax.set_xlabel('False Positive Rate')
ax.set_ylabel('True Positive Rate')
ax.set_title('ROC Curve Comparison of Ridge Logistic Regression, ANN, and
XGBoost')
ax.legend(loc="lower right")
ax.grid(True)

# Save the figure
fig.savefig('roc_curve_comparison.png', dpi=300)

# Show the plot
plt.show()
```

```
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score,
roc_auc_score

def evaluate_model(y_true, y_pred, model_name):
    accuracy = accuracy_score(y_true, y_pred)
    precision = precision_score(y_true, y_pred)
    recall = recall_score(y_true, y_pred)
    f1 = f1_score(y_true, y_pred)
    roc_auc = roc_auc_score(y_true, y_pred)

    print(f"{model_name} Performance:")
```

```
    print(f"Accuracy: {accuracy:.4f}")
    print(f"Precision: {precision:.4f}")
    print(f"Recall: {recall:.4f}")
    print(f"F1 Score: {f1:.4f}")
    print(f"ROC AUC Score: {roc_auc:.4f}")
```

Create an empty list to store results

```
results = []

# Evaluate models and store results
results.append(evaluate_model(y_test, y_pred_ridge, "Ridge Logistic Regression"))
results.append(evaluate_model(y_test, y_pred_ann, "Artificial Neural Networks"))
results.append(evaluate_model(y_test, y_pred_xgb, "Extreme Gradient Boosting"))

# Convert results to a DataFrame
results_df = pd.DataFrame(results)
```

```
Ridge Logistic Regression Performance:
Accuracy: 0.7807
Precision: 0.7357
Recall: 0.1550
F1 Score: 0.2560
ROC AUC Score: 0.5685
Artificial Neural Networks Performance:
Accuracy: 0.9998
Precision: 0.9994
Recall: 0.9997
F1 Score: 0.9996
ROC AUC Score: 0.9998
Extreme Gradient Boosting Performance:
Accuracy: 1.0000
Precision: 1.0000
Recall: 1.0000
F1 Score: 1.0000
ROC AUC Score: 1.0000
```

Extract the Create a Table of Performance Metrics for the three Models

```
import pandas as pd
# Example structure of what evaluate_model might return
def evaluate_model(y_true, y_pred, model_name):
    # Assuming this function returns a dictionary of evaluation metrics
    return {
        'Model': model_name,
        'Accuracy': accuracy_score(y_true, y_pred),
        'Precision': precision_score(y_true, y_pred),
        'Recall': recall_score(y_true, y_pred),
        'F1-Score': f1_score(y_true, y_pred),
        'Balanced Accuracy': roc_auc_score(y_true, y_pred)
    }
```

```
# Create an empty list to store results
results = []

# Evaluate models and store results
results.append(evaluate_model(y_test, y_pred_ridge, "Ridge Logistic Regression"))
results.append(evaluate_model(y_test, y_pred_ann, "Artificial Neural Networks"))
results.append(evaluate_model(y_test, y_pred_xgb, "Extreme Gradient Boosting"))

# Convert the list of dictionaries to a DataFrame
results_df = pd.DataFrame(results)
```

Display the DataFrame

```
results_df
```

	Model	Accuracy	Precision	Recall	F1-Score	Balanced Accuracy
0	Ridge Logistic Regression	0.780689	0.735738	0.154972	0.256018	0.568528
1	Artificial Neural Networks	0.999798	0.999448	0.999724	0.999586	0.999773
2	Extreme Gradient Boosting	1.000000	1.000000	1.000000	1.000000	1.000000

Add AUC to the Data Frame

```
# AUC values for the three models
auc_values = [0.71, 1.00, 1.00]

# Adding the AUC column to the DataFrame
results_df['AUC'] = auc_values

# Display the updated DataFrame
results_df
```

	Model	Accuracy	Precision	Recall	F1-Score	Balanced Accuracy	AUC
0	Ridge Logistic Regression	0.780689	0.735738	0.154972	0.256018	0.568528	0.71
1	Artificial Neural Networks	0.999798	0.999448	0.999724	0.999586	0.999773	1.00
2	Extreme Gradient Boosting	1.000000	1.000000	1.000000	1.000000	1.000000	1.00

Appendix 4: R-Codes

COMPARATIVE ANALYSIS OF RIDGE LOGISTIC REGRESSION, ARTIFICIAL NEURAL NETWORKS AND EXTREME GRADIENT BOOSTING FOR ENHANCING LOAN DEFAULT PREDICTION FOR KENYA FINANCIAL BANKS
Mary Lemasulani Mpaine

r Sys.Date()

{r setup, include=FALSE} knitr::opts_chunk$set(echo = TRUE, warning=FALSE,comment = NA, message=FALSE, fig.height=4, fig.width=6)

COMPARATIVE ANALYSIS OF RIDGE LOGISTIC REGRESSION, ARTIFICIAL NEURAL NETWORKS AND EXTREME GRADIENT BOOSTING FOR ENHANCING LOAN DEFAULT PREDICTION FOR KENYA FINANCIAL BANKS

Load Some Necessary Packages and Libraries

library(summarytools)
library(tidymodels)
library(dplyr)
library(sjmisc)
library(stargazer) ##for creating tables
library(caret) #for machine learning models
library(psych) ##for description of data
library(ggplot2) ##for data visualization
library(caretEnsemble)##enables the creation of ensemble models
library(tidyverse) ##for data manipulation
library(mlbench) ## for benchmarking ML Models
library(flextable) ## to create and style tables
library(mltools) #for hyperparameter tuning
library(tictoc) #for determining the time taken for a model to run
library(ROSE) ## for random oversampling
library(smotefamily) ## for smote sampling
library(ROCR) ##For ROC curve
library(iml)
library(lime)
####Load the Health Data you want to work on

default = read.csv("Uncleaned Default Rate data.csv", header = TRUE)
head(default)
attach(default)

Marital Status

```
default$Marital_Status <- ifelse(income >=0 & income <=2000, 1,
          ifelse(income >2000 & income <=10000, 3,
               ifelse(income > 10000, 2, NA)))
head(default)
```

Move the Marital Status to the Second Column

```
last_col <- ncol(default)
default <- default[, c(2, last_col, 3:(last_col - 1))]
head(default)
```

DATA PREPARATION

Check for Missing Values

```
missing_data = as.data.frame(colSums(is.na(default)))
names(missing_data) <- "Missing"
missing_data
```

View the Structure of the data

```
str(default)
```

Impute the Missing values with the Mean value

```
for(i in 1:ncol(default)) {
  if(is.numeric(default[[i]])) {  # Check if the column is numeric
    default[[i]][is.na(default[[i]])] <- mean(default[[i]], na.rm = TRUE)
  }
}
```

Check for Missing values

```
missing_data = as.data.frame(colSums(is.na(default)))
names(missing_data) <- "Missing"
missing_data
```

Convert Some Numeric Variables to Integer

```
default$loan_limit <- as.integer(default$loan_limit)
default$Marital_Status <- as.integer(default$Marital_Status)
default$age <- as.integer(default$age)
default$loan_purpose <- as.integer(default$loan_purpose)
default$Credit_Worthiness <- as.integer(default$Credit_Worthiness)
```

View the Data in the Viewer tab

```
str(default)
#View(default)
```

View the Dimension of the Prepared Data

```
dim(default)
```

Descriptive Statistics

```
library(psych)
describeBy(default)

NAMES <- as.data.frame(names(default))
names(NAMES)<- "NAMES"
NAMES

frq(default, Status)
```

Create Factors for Categorical variables

```
default$loan_limit <- factor(default$loan_limit,
        levels = c(0,1),
        labels = c('No', 'Yes'))

default$loan_type <- factor(default$loan_type,
        levels = c(1,2,3),
        labels = c('Secured', 'Unsecured', 'Specialized'))

default$loan_purpose <- factor(default$loan_purpose,
        levels = c(1,2,3,4),
        labels = c('Vocation', 'Car','Domestic Appliances','Education'))

default$Credit_Worthiness <- factor(default$Credit_Worthiness,
        levels = c(0,1),
        labels = c('No', 'Yes'))

default$business_or_commercial <- factor(default$business_or_commercial,
        levels = c(0,1),
        labels = c('Yes', 'No'))

default$credit_type <- factor(default$credit_type,
        levels = c(1,2,3,4),
        labels = c('CRB', 'CRIF','EQUI','EXP'))

default$age <- factor(default$age,
        levels = c(1,2,3,4,5,6,7),
        labels = c('<25', '25-34','35-44','45-54','55-64','65-74','>74'))

default$Marital_Status <- factor(default$Marital_Status,
        levels = c(1,2,3),
        labels = c('Single', 'Married','Divorced'))

default$Gender <- factor(default$Gender,
```

levels = c(0,1),
labels = c('Female', 'Male'))

Note: For the purpose of this training,
it is assumed that the data is already clean and preprocessed
###Rename the classes of the Target variable and plot it to determine imbalance *{r, fig.height=5, fig.width=6} default$Status <- factor(default$Status, levels = c(0,1), labels = c('No', 'Yes'))*

Frequency Distribution Table for Hypertension

library(sjmisc)
frq(default, Status)
Plot the Results using Bar Plot
{r, fig.height=5, width =9} ###Plot Target Variable plot(default$Status, names= c('Non Defaulter', 'Defaulter'), col=c(3,2), ylim=c(0, 140000), ylab='Respondent', xlab='Defaulter Prediction', main = "Bar Graph of Defaulters and Non Defaulter") box()
Or use ggplot

ggplot(default, aes(x = Status, fill = Status)) +
geom_bar() +
labs(x = "Default Intention", y = "Count") #+
 # scale_fill_manual(values = c("darkgreen", "red"))

#write.csv(default, "Loan Default2.csv")
Features Importance Plot
{r, fig.height=9, fig.width=10} attach(default) featurePlot(x = default[, -which(names(default) == "Status")], y = default$Status, plot = "box", scales = list(x = list(relation = "free"), y = list(relation = "free")), main = "Visual Comparison of Feature Distributions Across Default Intention")
Inferential Statistics

tbl_summary(default,
by = Status)%>%
add_ci()%>%
add_p()
Marital Status and Default Intention
View(default)
tbl_summary(default[,c(2,16)],
by = Status)

 # Create a contingency table
table_marital_status <- table(default$Marital_Status, default$Status)

 # Add row percentages
*table_with_percentages <- prop.table(table_marital_status, margin = 1) * 100*

 # Combine the counts and the percentages into one table
table_combined <- cbind(table_marital_status, table_with_percentages)
colnames(table_combined) <- c("Count (0)", "Count (1)", "Percent (0)", "Percent (1)")

 # Print the combined table
table_combined

library(ggplot2)
library(scales)
ggplot(default, aes(x = Marital_Status, fill = Status)) +
geom_bar(aes(y = ..count../tapply(..count.., ..x.., sum)[..x..]), position = "dodge") +
geom_text(aes(
y = ..count../tapply(..count.., ..x.., sum)[..x..],
label = percent(..count../tapply(..count.., ..x.., sum)[..x..])
), stat = "count", position = position_dodge(0.9), vjust = -0.5, size = 3) +
ylab("Percentages") +
xlab("Marital Status") +
ggtitle("Bar Graph Showing the Distribution of Marital Status Across\nDefault Intention") +
scale_y_continuous(labels = percent) + # Format y-axis labels as percentages
theme(
plot.title = element_text(hjust = 0.5), # Center the title
axis.text.x = element_text(angle = 45, hjust = 1) # Rotate x-axis labels for readability if needed
)

frq(default, Marital_Status)
```{r, fig.height=7, fig.width=8} library(ggplot2) library(dplyr)
**Calculate the count and percentage**
data_summary <- default %>% group_by(Status) %>% summarise(count = n()) %>% mutate(percent = count / sum(count) * 100) data_summary
**Plot**
ggplot(data_summary, aes(x = Status, y = count, fill = Status)) + geom_bar(stat = “identity”) + geom_text(aes(label = paste0(count, ” (“, round(percent, 1),”%)“)), vjust = -0.5, size = 4) + labs(x =”Default Intention”, y = “Count”, title = “Distribution of Default Intention”) + theme(plot.title = element_text(hjust = 0.5)) # Center the title
*## Gender and Loan Default*
*```{r}*
*frq(default, Gender)*

*# Create a contingency table*
*table_gender <- table(default$Gender, default$Status)*

*# Add row percentages*
*table_with_percentages <- prop.table(table_gender, margin = 1) * 100*
```

```
# Combine the counts and the percentages into one table
table_combined <- cbind(table_gender, table_with_percentages)
colnames(table_combined) <- c("Non defaulter (0)", "defaulter (1)", "Percent (0)",
"Percent (1)")

# Print the combined table
table_combined

ggplot(default, aes(x = Gender, fill = Status)) +
  geom_bar(aes(y = ..count../tapply(..count.., ..x.., sum)[..x..]), position = "dodge") +
  geom_text(aes(
    y = ..count../tapply(..count.., ..x.., sum)[..x..],
    label = percent(..count../tapply(..count.., ..x.., sum)[..x..])
  ), stat = "count", position = position_dodge(0.9), vjust = -0.5, size = 3) +
  ylab("Percentages") +
  xlab("Gender") +
  ggtitle("Bar Graph Showing the Distribution of Loan Status Across\nGender") +
  scale_y_continuous(labels = percent) +  # Format y-axis labels as percentages
  theme(
    plot.title = element_text(hjust = 0.5),  # Center the title
    axis.text.x = element_text(angle = 45, hjust = 1)  # Rotate x-axis labels for
readability if needed
  )
```

Loan Purpose and Default intention

```
# Create a contingency table
table_purpose <- table(default$loan_purpose, default$Status)

# Add row percentages
table_with_percentages <- prop.table(table_purpose, margin = 1) * 100

# Combine the counts and the percentages into one table
table_combined <- cbind(table_purpose, table_with_percentages)
colnames(table_combined) <- c("Non defaulter (0)", "defaulter (1)", "Percent (0)",
"Percent (1)")

# Print the combined table
table_combined
{r fig.height=6, fig.width=7} ggplot(default, aes(x = loan_purpose, fill = Status)) +
geom_bar(aes(y = ..count../tapply(..count.., ..x.., sum)[..x..]), position = "dodge") +
geom_text(aes(     y = ..count../tapply(..count.., ..x.., sum)[..x..],      label =
percent(..count../tapply(..count.., ..x.., sum)[..x..])   ), stat = "count", position =
position_dodge(0.9), vjust = -0.5, size = 3) +   ylab("Percentages") +   xlab("Loan
Purpose") +   ggtitle("Bar Graph Showing the Distribution of Loan Status
Across\nLoan Purpose") +   scale_y_continuous(labels = percent) +  # Format y-axis
labels as percentages   theme(     plot.title = element_text(hjust = 0.5),  # Center the
```

```
title     axis.text.x = element_text(angle = 45, hjust = 1)  # Rotate x-axis labels for readability if needed   )
```

Credit Worthiness

```
# Create a contingency table
table_credit <- table(default$Credit_Worthiness, default$Status)

# Add row percentages
table_with_percentages <- prop.table(table_credit, margin = 1) * 100

# Combine the counts and the percentages into one table
table_combined <- cbind(table_credit, table_with_percentages)
colnames(table_combined) <- c("Non defaulter (0)", "defaulter (1)", "Percent (0)", "Percent (1)")

# Print the combined table
table_combined

ggplot(default, aes(x = Credit_Worthiness, fill = Status)) +
  geom_bar(aes(y = ..count../tapply(..count.., ..x.., sum)[..x..]), position = "dodge") +
  geom_text(aes(
    y = ..count../tapply(..count.., ..x.., sum)[..x..],
    label = percent(..count../tapply(..count.., ..x.., sum)[..x..])
  ), stat = "count", position = position_dodge(0.9), vjust = -0.5, size = 3) +
  ylab("Percentages") +
  xlab("Gender") +
  ggtitle("Bar Graph Showing the Distribution of Loan Status Across\n Credit Worthness") +
  scale_y_continuous(labels = percent) +  # Format y-axis labels as percentages
  theme(
    plot.title = element_text(hjust = 0.5),  # Center the title
    axis.text.x = element_text(angle = 45, hjust = 1)  # Rotate x-axis labels for readability if needed
  )
```

Age Category and Loan Default Status

```
# Create a contingency table
table_age <- table(default$age, default$Status)

# Add row percentages
table_with_percentages <- prop.table(table_age, margin = 1) * 100

# Combine the counts and the percentages into one table
table_combined <- cbind(table_age, table_with_percentages)
colnames(table_combined) <- c("Non defaulter (0)", "defaulter (1)", "Percent (0)", "Percent (1)")
```

```
# Print the combined table
table_combined
```

```
{r, fig.height=6, fig.width=9} ggplot(default, aes(x = age, fill = Status)) +
geom_bar(aes(y = ..count../tapply(..count.., ..x.., sum)[..x..]), position = "dodge") +
geom_text(aes(     y = ..count../tapply(..count.., ..x.., sum)[..x..],      label =
percent(..count../tapply(..count.., ..x.., sum)[..x..])   ), stat = "count", position =
position_dodge(0.9), vjust = -0.5, size = 3) +   ylab("Percentages") +
xlab("Gender") +   ggtitle("Bar Graph Showing the Distribution of Loan Status
Across\nAge Categories") +   scale_y_continuous(labels = percent) +  # Format y-
axis labels as percentages   theme(     plot.title = element_text(hjust = 0.5),  # Center
the title     axis.text.x = element_text(angle = 45, hjust = 1)  # Rotate x-axis labels for
readability if needed   )
```

Loan Borrowed for Busines/Commercial or Otherwise

```
# Create a contingency table
table_business <- table(default$business_or_commercial, default$Status)

# Add row percentages
table_with_percentages <- prop.table(table_business, margin = 1) * 100

# Combine the counts and the percentages into one table
table_combined <- cbind(table_business, table_with_percentages)
colnames(table_combined) <- c("Non defaulter (0)", "defaulter (1)", "Percent (0)", "Percent
(1)")

# Print the combined table
table_combined
```

```
ggplot(default, aes(x = business_or_commercial, fill = Status)) +
  geom_bar(aes(y = ..count../tapply(..count.., ..x.., sum)[..x..]), position = "dodge") +
  geom_text(aes(
    y = ..count../tapply(..count.., ..x.., sum)[..x..],
    label = percent(..count../tapply(..count.., ..x.., sum)[..x..])
  ), stat = "count", position = position_dodge(0.9), vjust = -0.5, size = 3) +
  ylab("Percentages") +
  xlab("Gender") +
  ggtitle("Bar Graph Showing the Distribution of Loan Status Across\n Business/Commercial
Purpose") +
  scale_y_continuous(labels = percent) +  # Format y-axis labels as percentages
  theme(
    plot.title = element_text(hjust = 0.5),  # Center the title
    axis.text.x = element_text(angle = 45, hjust = 1)  # Rotate x-axis labels for readability if
needed
  )
```