



## **Masters Project**

## **M.Sc. Data Analytics**

# **Machine Learning-Based Loan Approval Automation**

**Enhancing Efficiency, Accuracy, and Fairness in Credit Decision-Making**

**Hellen Otieno: up2280493**  
**Project unit: M32616-2024/25**  
**Supervisor: Dr Mani Ghahremani**  
**September 2024**

1 Introduction	5
1.1 Context of Research	5
1.2 Project Aim	5
1.3 Project Objectives	5
2 Literature Review	7
2.1 Research Plan	7
2.2 Research Background and Definitions	8
2.3 Similar Works	9
2.3.1 The integration of the Artificial Bee Colony (ABC) method with Random Forest to further improve prediction accuracy	9
2.3.2 Utilising K-Nearest Neighbours, Support Vector Machine, and Decision Trees to predict loan approval for customers	10
2.3.3 Comparison of decision tree, random forest, support vector machine, K-nearest neighbour, and decision tree with Ada Boost machine learning methods on predicting the customer loan eligibility	10
2.3.4 Machine learning and deep learning techniques to assess loan eligibility	11
2.3.5 Logistic Regression-Based Loan Approval Model	11
2.3.6 Logistic Regression algorithm, Random forest, Decision trees, Linear Regression, Support Vector Machine (SVM), Naïve Bayes, K-means and K Nearest Neighbours (KNN) algorithms.	12
2.3.7 Evaluating the effects of feature selection methods, K-Best and RFE, on model performance in loan approval prediction	12
2.3.8 Logistic regression, Decision tree, Random Forest, support vector machine (SVM), Ada Boost and Neural Network in loan application approval.	13
2.3.9 Predictive finance decision support system using Naive Bayes and Decision Tree algorithms.	13
2.3.10 Enhancing transparency and understanding in AI, including model-agnostic approaches like LIME and SHAP, and intrinsically interpretable models such as decision trees and rule-based systems.	14
2.3.11 An explainable AI model for credit risk management	14
2.3.12 Ensemble Machine Learning Approaches for Bank Loan Approval Prediction	15
2.3.13 The integration of ANN and SVM in an ensemble model	15
2.3.14 AI in credit scoring	15
2.3.15 Comparative Analysis of Machine Learning Models for Loan Approval Prediction	16
2.4 Summary	16
3 Methodology	20
3.1 Machine Learning Methodology	20
3.1.1 Suitable Frameworks	20
3.1.1.1 Kanban	20
3.1.1.2 Scrum	20
3.1.1.3 Cross-Industry Standard Process for Data Mining (CRISP-DM)	21
3.1.2 Framework Used	22
3.2 Project Planning	22

3.2.1 Initial Plan	22
3.2.2 Actual Plan	24
3.3 Ethical Considerations	25
3.3.1 Fairness and Non-Discrimination	25
3.3.2 Transparency and Explainability	25
3.3.3 Data Privacy	25
4 Requirements	26
4.1 Requirement Gathering	26
4.2 Organisation of Requirements	26
4.3 Functional Requirements	27
4.4 Non-functional Requirements	28
5 Design	29
5.1 Machine Learning Pipeline	29
5.2 Pipeline Design	29
5.3 Machine Learning Models and Ensemble Techniques	32
5.4 Evaluation Metrics	34
6 Implementation	36
6.1 Iteration 1	36
6.1.1 Data Collection	36
6.1.2 Exploratory Data Analysis	36
6.1.3 Data Pre-processing	41
6.1.4 Random Forest Model Training	44
6.1.5 Random Forest Model Tuning	46
6.1.6 Model Comparison: Before and After Hyperparameter Tuning	47
6.1.7 Review	48
6.2 Iteration 2	49
6.2.1 Data Collection and Pre-processing	49
6.2.2 XGBoost Model Training	49
6.2.3 XGBoost Hyperparameter Tuning	50
6.2.4 Model Comparison: Before and After Hyperparameter Tuning	52
6.2.5 Review	52
6.3 Iteration 3	53
6.3.1 Data Collection and Pre-processing	53
6.3.2 Stacking Ensemble Model Training	53
6.3.3 Voting Ensemble Model Training	54
6.3.4 Model Comparison: Stacking Ensemble vs Voting Ensemble	55
6.3.5 Review	56

6.4 Final Model Comparison and Recommendation	57
6.5 Explainable AI Interpretation of the Random Forest Model using LIME	58
7 Evaluation	60
7.1 Functional Requirements	60
7.2 Non-functional Requirements	61
7.3 Limitations	61
8 Conclusion and Future Work	62
8.1 Summary	62
8.2 Lessons Learnt	62
8.3 Future Work	63
Bibliography	64
Appendix I	67

# 1 Introduction

## 1.1 Context of Research

This project explores the application of machine learning in automating loan approval processes, with a focus on optimising efficiency, accuracy, and fairness in credit decision-making. Traditionally, determining loan eligibility involves manually verifying an applicant's financial details after they submit a loan application. This process is often time-consuming and susceptible to human bias, resulting in inefficiencies and a lack of transparency. Implementing an automated system that instantly assesses loan eligibility based on an individual's details offers significant potential benefits.

When correctly designed and deployed, the automation of loan approvals can provide substantial advantages for financial institutions, including faster decision-making, reduced operational costs, and the ability to scale operations more effectively. However, this transition also introduces important ethical and regulatory considerations. Financial institutions must ensure that machine learning-based loan approval systems treat applicants equitably and do not inadvertently disadvantage particular demographic groups (Nazim et al., 2023).

This research aims to replace traditional, manual credit assessment methods with data-driven machine learning models that offer improved predictive performance. Machine learning techniques have demonstrated strong capabilities in analysing large-scale, complex datasets (Nancy Deborah et al., 2023). By integrating interpretable and fair machine learning approaches, this study contributes to the development of responsible, transparent, and equitable financial systems.

## 1.2 Project Aim

This project aims to develop a machine learning-based loan approval model designed to enhance decision-making efficiency and predictive accuracy while promoting fairness and transparency in credit assessments.

## 1.3 Project Objectives

The primary objectives of this project are as follows:

- 1. Literature Review:**

Conduct a critical review of existing research on machine learning applications in loan approval, focusing on predictive performance, fairness in decision-making, and model interpretability. This will help identify research gaps, best practices, and effective modelling strategies to inform subsequent stages of the project.

- 2. Data Preparation and Exploration:**

Utilise a publicly available loan dataset from Kaggle and perform data pre-processing, including handling missing values, encoding categorical features, and engineering new variables relevant to loan approval. Conduct Exploratory Data Analysis (EDA) to uncover key patterns, detect biases, and inform model design.

**3. Model Development and Implementation:**

Develop and implement supervised machine learning models to classify loan approval status. Apply hyperparameter tuning to optimise model performance and evaluate the use of ensemble techniques, including stacking and voting, for improved predictive accuracy.

**4. Fairness and Explainability Analysis:**

Employ explainable AI techniques (e.g., LIME) to interpret model predictions. Assess fairness metrics to evaluate model behaviour across demographic and financial subgroups, ensuring the absence of discriminatory bias.

**5. Evaluation and Validation:**

Evaluate model performance using metrics such as accuracy, precision, recall, F1-score, and Area Under the Curve (AUC). Apply cross-validation to ensure reliability and generalisability, while incorporating fairness indicators to quantify equity in predictions.

**6. Planning and Reporting:**

Maintain detailed documentation of the methodology, results, and insights throughout the project. The final dissertation will present key findings, highlight limitations, discuss ethical considerations, and provide recommendations for future work.

## 2 Literature Review

### 2.1 Research Plan

The literature review was conducted through a search of relevant academic sources using Google Scholar, EBSCOhost, and IEEE to gather recent peer-reviewed research on machine learning for loan approval automation. Initial broad search terms like “machine learning for loan approval automation” were refined to more specific phrases such as “AI in credit decision-making” to ensure relevance. To maintain currency and quality, filters were applied to restrict results to peer-reviewed publications dated between 2022 and 2025. This ensured the inclusion of the latest advances and trends in the rapidly evolving field of AI-driven credit risk assessment. Abstracts and methodologies were reviewed closely. Particular attention was paid to studies discussing model performance, fairness considerations, interpretability techniques, and identified areas for future research.

#### Summary of results:

##### Google Scholar

Search term	Number of results	Relevant results
Machine learning for loan approval automation	16800	2
AI in credit decision-making	34100	2
Loan approval machine learning models	16700	1

*Table 2.1 Google Scholar search results*

##### EBSCO

Search term	Number of results	Relevant results
Machine learning for loan approval automation	1136	0
AI in credit decision-making	22800	0
Loan approval machine learning models	4646	3

*Table 2.2 EBSCO search results*

## IEEE

Search term	Number of results	Relevant results
Machine learning for loan approval automation	10	5
AI in credit decision-making	111	0
Loan approval machine learning models	99	2

*Table 2.3 IEEE search results*

## 2.2 Research Background and Definitions

Machine Learning is the study of computer algorithms that improve automatically through experience (Mitchell, 1997). In the context of machine learning systems, interpretability is defined as the capacity to provide explanations or present information in a manner that is comprehensible to humans (Doshi-Velez & Kim, 2017).

The problem of loan approval automation has gained significant attention in recent years, particularly due to the rapid advancement of machine learning techniques and the growing need for more efficient, accurate, and fair credit decision-making processes. Traditional loan approval systems are frequently manual and rely on a limited set of parameters. While these parameters offer some insight into a borrower's creditworthiness, they sometimes fail to capture the full complexity of an individual's financial situation.

The credit risk management and evaluation constitute a significant challenge within the credit sector. The requirement to coordinate multiple internal departments and external stakeholders during the loan approval process often results in prolonged decision times, which may adversely impact a financial institution's ability to capitalise on business opportunities (Nancy Deborah et al., 2023).

Furthermore, these systems are prone to human error, bias, and inefficiency, resulting in delayed decisions, inconsistent loan approvals, and a lack of transparency in the decision-making process (Addy et al., 2024). Interpretability is essential for mitigating bias in AI models, particularly in credit scoring systems, where fairness and transparency are paramount for equitable decision-making (Addy et al., 2024). Lack of transparency in AI models can lead to significant issues regarding fairness and trust. Ensuring that AI models are interpretable allows stakeholders to understand the reasons behind credit decisions, which is crucial for regulatory compliance and consumer acceptance (Addy et al., 2024).

Automation systems use a wide range of data sources to make more informed predictions about the likelihood of loan approval, leveraging advanced algorithms like decision trees, random forests, and support vector machines (Nazim et al., 2023). However, despite the potential advantages, machine learning-based systems face significant challenges, particularly in terms of ensuring fairness, transparency, and interpretability. For instance, there is a risk that the models might extend existing biases, such as discrimination based on race, gender, or socioeconomic status, if the training data used contains inherent prejudices (Srivastava, 2024).



Explainable AI techniques are widely used to enhance transparency, allowing stakeholders to better understand how a model arrived at a particular decision (Srivastava, 2024). Despite these advancements, ensuring that automated loan approval systems are accurate and fair remains a complex issue, requiring careful attention to data selection, model design, and continuous monitoring for potential biases (Addy et al., 2024).

Several factors, including data quality and availability, constrain the solution to the problem. Furthermore, legal and ethical considerations, such as compliance with financial regulations on fairness, data privacy, and discrimination, must be addressed to ensure the model adheres to regulatory standards and is free from bias.

## 2.3 Similar Works

### 2.3.1 The integration of the Artificial Bee Colony (ABC) method with Random Forest to further improve prediction accuracy

Andhale et al. (2024) directly address the inefficiencies and biases that arise in the traditional manual loan approval process. Their work aimed to enhance the loan approval workflow by automating and optimising it through machine learning, with a particular focus on improving predictive accuracy by identifying the ideal set of hyperparameters for the Random Forest model. To achieve this, the authors integrated the Artificial Bee Colony (ABC) algorithm, a population-based optimisation technique inspired by honeybee foraging behaviour, with the Random Forest classifier.

The ABC algorithm simulates the foraging behaviour of honeybees to search for the optimal solution in a given space. It consists of employed bees, onlooker bees, and scout bees that work collaboratively to explore the search space, evaluate potential solutions, and converge on the most promising areas.

The ABC algorithm is particularly useful in optimising complex, high-dimensional problems like hyperparameter tuning in machine learning models. In this context, the authors used ABC to repeatedly search the hyperparameter space and identify the most effective configuration for the Random Forest model in predicting loan approval status.

Alongside this hybrid model, they also evaluated the performance of standalone Support Vector Machines (SVMs) and Random Forest classifiers, providing a comparison of their effectiveness in predicting loan approval outcomes.

The methodology involved pre-processing a dataset containing applicant attributes such as income, education, credit history, and property area, which were then used to train and test the machine learning models. ABC was employed to fine-tune parameters like the number of trees and depth in the Random Forest algorithm. The performance of the optimised model was assessed using standard metrics including accuracy, precision, and recall. Results showed that the ABC-enhanced Random Forest achieved an accuracy of 78%, a precision of 98.86%, and a recall of 77.02%, indicating a strong ability to predict loan approvals reliably.

While the study demonstrated notable improvements in accuracy and efficiency, some limitations are identifiable. The dataset's size and diversity were not fully discussed, which raises concerns about how well the model would generalise to other datasets or demographic groups. Additionally, Random Forest and ABC

are complex models, and the paper did not address how interpretable the models' decisions are to stakeholders like loan officers.

### **2.3.2 Utilising K-Nearest Neighbours, Support Vector Machine, and Decision Trees to predict loan approval for customers**

Nancy Deborah et al. (2023) experimented with various machine learning models, including K-Nearest Neighbours (KNN), Decision Trees, and most notably, a Support Vector Classifier (SVC).

They used a dataset containing 615 records, each with features like gender, marital status, education, applicant and co-applicant income, credit history, property area, and loan status. The models were evaluated based on accuracy, precision, recall, and F1 score. Among the models tested, the Support Vector Classifier achieved the highest performance, recording an accuracy, precision, and recall of 83%, with similarly strong scores across other evaluation metrics.

The study results show that the authors successfully met their aim of enhancing the loan approval workflow through automation and predictive modelling. However, while the study mentions accuracy and efficiency, it does not engage with fairness-related concerns such as bias across demographic groups, which is critical in the context of loan approvals.

The authors also acknowledge that their model's performance could be enhanced in future work by retraining and fine-tuning the SVM as more data becomes available and computational resources improve. They suggest incorporating additional features, such as economic indicators and alternative data sources, to better adapt the model to dynamic market conditions.

### **2.3.3 Comparison of decision tree, random forest, support vector machine, K-nearest neighbour, and decision tree with Ada Boost machine learning methods on predicting the customer loan eligibility**

Naveen Kumar et al. (2022) employed a structured methodology involving data collection from a public repository, pre-processing to handle missing values, feature selection using analysis of variance (ANOVA), and model training/testing. The dataset of 614 records was split into an 80:20 ratio for training and testing.

They implemented and compared several machine learning models, including Decision Tree, Random Forest, Support Vector Machine (SVM), K-Nearest Neighbour (KNN), and an ensemble model combining Decision Tree with Ada Boost.

While the ensemble model achieved a high training accuracy of 91% and a test accuracy of 84%, the authors did not report other critical evaluation metrics such as precision, recall, or F1 score. This makes it difficult to assess how well the model balances false positives and negatives, which are essential considerations in loan approval systems where misclassification can have significant financial and ethical consequences.

The authors propose the use of deep learning models in future work to further improve prediction reliability.

### 2.3.4 Machine learning and deep learning techniques to assess loan eligibility

Rao et al. (2024) implemented a hybrid approach using a range of ML algorithms, including Decision Tree (DT), Random Forest (RF), Support Vector Machine (SVM), and K-Nearest Neighbour (KNN), as well as DL models like Recurrent Neural Networks (RNN) and Long Short-Term Memory (LSTM) networks.

The system was trained on a dataset of loan applicants using standard pre-processing techniques such as null value handling and categorical encoding. The models were evaluated primarily using accuracy as the performance metric, with DT, SVM, RNN, KNN, and LSTM achieving accuracy scores of 82%, 80%, 79.03%, 84%, and 81.78%, respectively. The results indicate that the system effectively predicted loan eligibility, with KNN and DT performing slightly better than the others.

Overall, the results show that the project largely succeeded in meeting its aim of automating loan analysis and improving efficiency and predictive performance. However, the evaluation was based solely on accuracy, with no mention of other important metrics such as precision, recall, or F1 score, which are especially crucial in financial applications where the cost of false approvals or rejections can be high. Additionally, while the system claimed to enhance fairness, no empirical fairness analysis or bias mitigation strategy was conducted.

The authors also proposed several directions for future development of their system. They suggest that generative AI could be integrated to estimate loan options and make more advanced recommendations for users. Additionally, incorporating features such as interactive financial planning tools, instructional materials, and personalised loan advice could enhance user engagement and financial literacy.

Importantly, the authors also acknowledge the need for ongoing collaboration with industry stakeholders and regulators to ensure that the system remains aligned with evolving standards of fairness, transparency, and responsible lending in the financial sector.

### 2.3.5 Logistic Regression-Based Loan Approval Model

Lohani et al. (2022) used Logistic regression as the predictive algorithm due to its mathematical simplicity and suitability for binary classification tasks. The methodology involved standard data pre-processing steps, including handling missing values, normalisation, and feature selection through correlation analysis. The dataset was then split into training and testing sets to build and evaluate the model.

The results showed that certain demographic factors, such as being married or having a graduate degree, correlated positively with loan approval, while self-employment status was found to have little predictive value. The logistic regression model was able to reduce the time and resources required by banks in the approval process without significantly compromising accuracy.

While the model met its stated objectives, the authors acknowledged limitations, including logistic regression's dependency on a large sample size and its assumption of linearity among independent variables. They also noted the presence of human bias reflected in model outputs, a common concern in credit decision automation. Nonetheless, the study concluded that predictive models like theirs could be integrated into banking systems, especially when future iterations use more diverse and realistic financial datasets.

### **2.3.6 Logistic Regression algorithm, Random forest, Decision trees, Linear Regression, Support Vector Machine (SVM), Naïve Bayes, K-means and K Nearest Neighbours (KNN) algorithms.**

Nureni & Adekola(2022) employed several machine learning algorithms, and the methodology involved training the models on a dataset obtained from Kaggle, which included various features such as borrower details and socioeconomic indicators. Performance metrics such as accuracy, precision, recall, and F1-score were used to evaluate the models' effectiveness.

The results revealed that Logistic Regression outperformed other models in terms of accuracy, with 83.24% and 78.13% accuracy across two datasets. However, the study found that different algorithms performed variably across the datasets, with Random Forest and Naïve Bayes showing competitive accuracy levels as well.

While the models demonstrated strong predictive performance, some limitations were noted. Specifically, K-Means and K-Nearest Neighbours were less effective in terms of sensitivity and recall, indicating their poor performance in classifying loan approvals correctly.

The authors also highlighted that the models' reliance on certain features, such as income and marital status, may overlook other significant factors, potentially introducing bias into the loan approval process. The authors suggested future improvements through ensemble learning to improve the robustness and accuracy of predictions.

### **2.3.7 Evaluating the effects of feature selection methods, K-Best and RFE, on model performance in loan approval prediction**

Sinap (2024) aimed to evaluate how the Recursive Feature Elimination (RFE) and K-Best feature selection methods influenced the effectiveness of machine learning models, specifically focusing on Random Forest (RF), Logistic Regression (LR), K-Nearest Neighbours (KNN), Support Vector Machine (SVM), and Decision Tree (DT). By using a loan approval dataset, the study sought to identify which features were most predictive of loan approval and how various feature selection techniques impacted the performance metrics such as accuracy, precision, recall, and F1-Score.

The methodology involved training multiple models with and without feature selection and comparing their performance using accuracy and other classification metrics. The models were evaluated using cross-validation and a traditional train-test-validation split. Notably, the study revealed that RF was the most successful algorithm, achieving an accuracy rate of 97.71%, significantly higher than other models tested.

The performance of the models improved notably when feature selection was applied, especially with the RFE method. The RFE-selected features outperformed both the models built with all features and those using the K-Best method, demonstrating that redundant features can negatively impact model performance.

Despite the strong results, the study acknowledged some limitations, including the relatively high complexity of the RF algorithm, which may limit interpretability, and the reliance on specific datasets that might not generalise to all loan approval contexts.

The study concluded that feature selection, especially using RFE, can significantly enhance model performance by removing irrelevant features, which is critical for improving the efficiency and accuracy of loan approval predictions in real-world banking applications.

### **2.3.8 Logistic regression, Decision tree, Random Forest, support vector machine (SVM), Ada Boost and Neural Network in loan application approval.**

Chunyu (2024) focused on six candidate models: Logistic Regression, Decision Tree, Random Forest, Support Vector Machine (SVM), Ada Boost, and Neural Networks. The methodology involved using a mortgage application dataset from Australia, which included applicant-specific features such as income, housing expenses, and loan details. Data pre-processing steps included oversampling to balance the dataset and removing irrelevant features, resulting in a dataset of 3,413 entries. The models were evaluated based on their accuracy on test data and their interpretability, using an 80:20 split for training and testing.

The results showed that Logistic Regression outperformed all other models in terms of accuracy (76%) and interpretability, making it the most suitable model for practical applications in loan approval. Random Forest and Neural Networks, although accurate, performed less well in terms of interpretability due to their “black-box” nature. The study also found that SVM provided robust predictions with a 73% accuracy rate but lacked the transparency of logistic regression.

The author concluded that while more complex models like Random Forest and Neural Networks could improve predictive performance, the interpretability of Logistic Regression made it the best choice for loan approval automation. However, the study noted that different datasets and feature sets might affect model performance, and future research should explore these variations. Additionally, the authors pointed out that more personalised features, such as the applicant's region or mental state, could significantly impact prediction accuracy but were difficult to include due to data limitations.

### **2.3.9 Predictive finance decision support system using Naive Bayes and Decision Tree algorithms.**

Malarvizhi et al. (2024) employed machine learning models such as Decision Tree and Naive Bayes, using a dataset from Kaggle, which included features like credit history, income, marital status, and loan details. The dataset was split into training and testing sets to evaluate the models' performance.

The results indicated that Naive Bayes outperformed Decision Tree, achieving an accuracy of 82.92%, compared to Decision Tree's 70.73%. The Naive Bayes model provided a more accurate and efficient method for predicting loan approval, making it a practical choice for the banking sector.

However, the study acknowledged that while Naive Bayes offered better predictive accuracy, the Decision Tree model was more interpretable, offering insights into the decision-making process. The study also noted that the dataset used may have limitations in terms of generalizability, and future work could explore incorporating additional features or using more complex models like neural networks to improve accuracy further.

### **2.3.10 Enhancing transparency and understanding in AI, including model-agnostic approaches like LIME and SHAP, and intrinsically interpretable models such as decision trees and rule-based systems.**

Pillai (2024) aimed to address the critical issue of transparency in AI decision-making processes, particularly in finance. The paper explored various methods to enhance the interpretability and understanding of AI models, focusing on both model-agnostic approaches, such as LIME and SHAP, and uniquely interpretable models like decision trees and rule-based systems.

The goal was to improve trust and accountability in AI systems, which are often criticised for their “black-box” nature, especially when used in critical applications like loan approvals. The paper proposed strategies to combine transparent models with high-performance models, offering hybrid solutions that maintain both accuracy and interpretability. Additionally, it emphasised the importance of user-centric design and regulatory frameworks to ensure that AI systems are ethically deployed.

The methodology involved an extensive review of existing techniques and their application in sectors such as healthcare and finance. The author demonstrated the practical application of LIME and SHAP in explaining complex AI predictions, helping users better understand and trust these models. The results highlighted that while complex models like deep neural networks offer high accuracy, they lack interpretability, which can be mitigated using simpler, interpretable models or hybrid approaches.

The study concluded that a balanced approach, combining interpretability with high performance, is crucial for the ethical deployment of AI. However, the paper also acknowledged the challenge of maintaining interpretability without sacrificing predictive accuracy, particularly in complex decision-making scenarios.

### **2.3.11 An explainable AI model for credit risk management**

Nallakaruppan et al. (2024) focused on creating an explainable AI (XAI)-driven system for binary credit classification, particularly suited for peer-to-peer lending environments. The authors employed Decision Tree and Random Forest classifiers, which were chosen for their compatibility with predictive performance and model transparency.

These models were further modified with Local Interpretable Model-Agnostic Explanations (LIME) and SHapley Additive exPlanations (SHAP) to enhance interpretability at both local and global levels. The methodology involved pre-processing a dataset of 1125 instances using imputation for missing values, then training and evaluating the classifiers. The study later validated the models on a larger dataset of over 32,000 records, applying random oversampling to address class imbalance.

The results demonstrated strong performance, with the Random Forest model achieving an accuracy of 93% and the Decision Tree model closely following at 90%. SHAP and LIME were successfully used to identify feature importance and offer individualised explanations for loan approval or rejection decisions. These explanations contributed to a more transparent decision-making process, helping stakeholders understand how different features influenced predictions.

However, the authors noted that while interpretability improved, the computational complexity introduced by SHAP, particularly in large-scale applications, remains a limitation. Moreover, the binary classification framework may oversimplify the delicate field of creditworthiness. The study recommends future research to

explore multi-class models, incorporate additional socioeconomic indicators, and balance computational efficiency with explainability.

### **2.3.12 Ensemble Machine Learning Approaches for Bank Loan Approval Prediction**

Nazim et al. (2023) experimented with a variety of machine learning and deep learning algorithms, including Logistic Regression, Decision Trees, Random Forest, Extra Trees, SVM, K-Nearest Neighbours, Naïve Bayes, Ada Boost, Gradient Boosting, Dense Neural Networks (DNN), LSTM, and RNN. The methodology involved extensive data pre-processing, including normalisation, encoding, and balancing via SMOTE. The team trained the models on a Kaggle dataset and evaluated performance using metrics such as accuracy, precision, recall, and F1-score.

The results demonstrated that the Extra Trees classifier was the best individual performer, with an accuracy of 86.64%. However, by employing a voting-based ensemble of the top three models (Random Forest, Extra Trees, and KNN), the authors achieved an even higher accuracy of 87.26%. This ensemble approach significantly outperformed both individual classifiers and several state-of-the-art models reported in prior literature. Additionally, the study introduced a desktop application with a user interface that allows bank staff or customers to input data and receive real-time predictions.

Despite these promising results, the authors acknowledged limitations such as the lack of hyperparameter tuning and the exclusive use of one ensemble technique. Moreover, deep learning models underperformed compared to machine learning classifiers due to the tabular nature of the dataset, highlighting the importance of model-data compatibility. The study concluded with suggestions for future work, including expanding real-time datasets, experimenting with other ensemble strategies, and exploring unsupervised learning methods to further refine the system's practical utility in real-world financial settings.

### **2.3.13 The integration of ANN and SVM in an ensemble model**

Gothai et al. (2024) proposed using a hybrid model that combined Artificial Neural Networks (ANN) and Support Vector Machines (SVM) to predict the likelihood of loan repayment. Their methodology involved training these models on a dataset sourced from Kaggle, which included various attributes related to loan applicants such as income, credit score, and previous loan history. The performance of the model was evaluated based on accuracy and recall, key metrics for assessing the quality of the predictions.

The results of the study showed that the ensemble model, which integrated ANN and SVM, significantly improved the accuracy of loan approval predictions, achieving an impressive 91% accuracy and 86% recall. This result suggests that the hybrid approach effectively leverages the strengths of both algorithms, providing a more robust solution for loan approval automation.

The ensemble model demonstrated that combining multiple machine learning techniques can enhance the reliability of predictions, particularly in the banking sector, where the risk of default is a major concern. However, the study also highlighted some limitations, such as the computational complexity of training the ensemble model and the potential for overfitting when using large datasets.

### **2.3.14 AI in credit scoring**

Addy et al. (2024) aimed to explore the diverse AI models used in credit scoring and predictive analytics, assessing their strengths, limitations, and real-world implications. The review focused on various machine

learning algorithms such as decision trees, random forests, support vector machines, and neural networks, alongside advanced predictive analytics methods. The authors also examined the integration of alternative data sources like social media activity and unconventional financial indicators to enhance creditworthiness assessments, particularly for individuals with limited credit histories.

The methodology involved a systematic review of existing studies on AI in credit scoring, analysing the application of different models and their impact on the accuracy, fairness, and transparency of credit evaluations. The results highlighted that while AI models offer superior predictive accuracy over traditional credit scoring methods, challenges remain in terms of model interpretability and fairness. The authors noted that while AI has the potential to streamline credit access, especially for underserved populations, it also raises concerns about the potential for algorithmic bias and discriminatory outcomes.

The study stressed the need for regulatory frameworks to mitigate the ethical concerns associated with AI models. The study called for more research into fairness-aware machine learning and methods to ensure the responsible deployment of AI in credit assessments. Although the review provided a thorough exploration of AI's benefits in credit scoring, it also highlighted the gaps in research regarding the integration of explainable models and the continuous monitoring of AI systems to prevent biases.

### **2.3.15 Comparative Analysis of Machine Learning Models for Loan Approval Prediction**

Srivastava (2024) used five different machine learning models: Logistic Regression, Support Vector Machine (SVM), Decision Tree Classifier, Random Forest Classifier, and an ensemble method, and compared their performance on a dataset of real bank credit applications. The dataset included attributes such as gender, marital status, education, income, loan amount, and credit history, which were pre-processed using techniques like encoding categorical data and normalising numerical values. The system was built using Python, with models trained on 70% of the dataset and evaluated using the remaining 30%.

The results revealed that the ensemble method performed the best, achieving an accuracy of 73.79%, followed by Random Forest at 63.23%. Logistic Regression showed the lowest performance, with an accuracy of 56.28%. However, the study also pointed out that while the ensemble method outperformed individual models, it introduced complexity in model interpretation. Furthermore, the models were trained on a limited dataset, which may not generalise well to all types of loan applications, particularly in different regions or financial institutions.

## **2.4 Summary**

The review of existing studies on machine learning-based loan approval systems has provided key insights that shape the requirements, design, and implementation of this project. It is clear that no single algorithm is universally superior; model effectiveness varies based on dataset characteristics and pre-processing strategies, emphasising the need to test multiple algorithms like Random Forest, Logistic Regression, SVM, and ensemble methods.

Hybrid models, as demonstrated by (Andhale et al., 2024) and (Gothai et al., 2024), can enhance performance, informing the project design to incorporate a flexible framework for combining models. The importance of interpretability and fairness, highlighted by (Pillai, 2024) and (Nallakaruppan et al., 2024), underscores the



need to integrate explainable AI tools such as LIME and SHAP to ensure transparency and meet ethical standards.

Additionally, the literature emphasises the significance of using multiple evaluation metrics, such as precision, recall, and F1-score, alongside accuracy, to minimise false positives and negatives, guiding the implementation strategy. Finally, the necessity for thorough data pre-processing, including feature selection and class balancing, as seen in the works of (Sinap, 2024) and (Srivastava, 2024), directly influences the project's technical design and data preparation workflow.

Algorithm name (and in-text citation)	Year	Statistics/Findings	Future work/Conclusion
The integration of the Artificial Bee Colony (ABC) method with Random Forest (Andhale et al., 2024).	2024	ABC with Random Forest: Accuracy=78% Precision=98.86% Recall=77.02%	— The ABC algorithm is advantageous for optimisation problems involving complex search spaces and high-dimensional data, especially when interpretability and efficiency are important.
K-Nearest Neighbours, Support Vector Machine, and Decision Trees (Nancy Deborah et al., 2023).	2023	SVC: Accuracy=83.33% Precision=83.17% Recall=83.33% F1 score=82.53%	— Retraining and fine-tuning the SVM as more data becomes available and computational resources improve. — Incorporating additional features, such as economic indicators and alternative data sources, to better adapt the model to dynamic market conditions.
Decision tree, Random Forest, Support Vector Machine, K-nearest Neighbour, and Decision tree with Ada Boost (Naveen Kumar et al., 2022)	2022	Decision tree with AdaBoost: Train Accuracy=91% Test Accuracy=84%	— Proposal of a deep learning model for Customer loan eligibility prediction.
Machine learning and Deep learning techniques (Rao et al., 2024).	2024	KNN Accuracy=84%	— AI integration to estimate loan options and make more advanced recommendations for users. — Incorporating features that could enhance user engagement and financial literacy.
Logistic Regression (Lohani et al., 2022)	2022	Not disclosed	— Using more diverse and realistic financial datasets.
Logistic Regression algorithm, Random forest, Decision trees, Linear	2022	Logistic regression: Dataset 1- Accuracy=83.2432%	— Ensemble models to enhance the reliability of the results

Regression, Support Vector Machine (SVM), Naïve Bayes, K-means and K Nearest Neighbours (KNN) algorithms (Nureni & Adekola, 2022)		Precision=82.3899% Recall=97.76119%  Dataset 2- Accuracy=78.125% Precision=78.9790% Recall=78.125%	
Evaluating the effects of feature selection methods, K-Best and RFE, on model performance (Sinap, 2024)	2024	Random Forest: Accuracy=97.68% Precision=92% Recall=95% F1 Score=93%	— The use of the feature selection methods (K-Best and RFE) significantly improved model performance.
Logistic regression, Decision tree, Random Forest, support vector machine (SVM), Ada Boost and Neural Network (Chunyu, 2024)	2024	Logistic regression accuracy=76%	— Different datasets and feature sets might affect model performance, and future research should explore these variations
Naive Bayes and Decision Tree algorithms (Malarvizhi et al., 2024)	2024	Naive Bayes accuracy=82.92%	— Incorporating additional features or using more complex models like neural networks to improve accuracy further
LIME and SHAP (Pillai, 2024)	2024	LIME, SHAP, hybrid models, user-focused design, and strong regulation together support building effective, interpretable, and responsible AI systems.	— Exploring a hybrid model approach, which combines complex, high-accuracy models with simpler, interpretable ones
Decision Tree, Random Forest with LIME and SHAP (Nallakaruppan et al., 2024)	2024	Random Forest accuracy=93%	— Explore multi-class models, incorporate additional socioeconomic indicators, and balance computational efficiency with explainability
Logistic Regression, Decision Trees, Random Forest, Extra Trees, SVM, K-Nearest Neighbours, Naïve Bayes, Ada Boost, Gradient Boosting, Dense Neural Networks (DNN), LSTM, and RNN (Nazim et al., 2023)	2023	Ensemble model accuracy=87.26%	— Expanding real-time datasets, experimenting with other ensemble strategies, and exploring unsupervised learning methods to further refine the system's practical utility in real-world financial settings
Artificial Neural Networks (ANN) and Support Vector	2024	Ensemble model accuracy=91%	— Not disclosed

Machines (SVM) (Gothai et al., 2024)			
AI in credit scoring (Addy et al., 2024)	2024	Interpretable AI, data privacy, security, and balancing accuracy with fairness remain key priorities.	— Research into fairness-aware machine learning and methods to ensure the responsible deployment of AI in credit assessments
Logistic Regression, Support Vector Machine (SVM), Decision Tree Classifier, Random Forest Classifier, and an ensemble method (Srivastava, 2024)	2024	Ensemble model=73.79%	— The integration of these models into the banking system to remain competitive, competent, and customer-centric

*Table 2.4: Literature Review Summary*

## 3 Methodology

### 3.1 Machine Learning Methodology

#### 3.1.1 Suitable Frameworks

##### 3.1.1.1 Kanban

Kanban is a method for coordinating workflows, primarily designed to improve process efficiency through visualisation and limiting work in progress (WIP). In a machine learning development context, Kanban emphasises visualising the workflow, limiting work-in-progress (WIP), and continuously improving processes incrementally. (Al-Baik & Miller, 2014).

Kanban is well-suited for projects where tasks arrive unpredictably or evolve continuously, such as data cleaning, iterative model tuning, and tasks related to explainability in a machine learning pipeline. Its focus on visualising workflow and limiting work in progress makes it practical for ML teams that need to handle changing data, maintain models, and deliver incremental updates without fixed-length sprints.

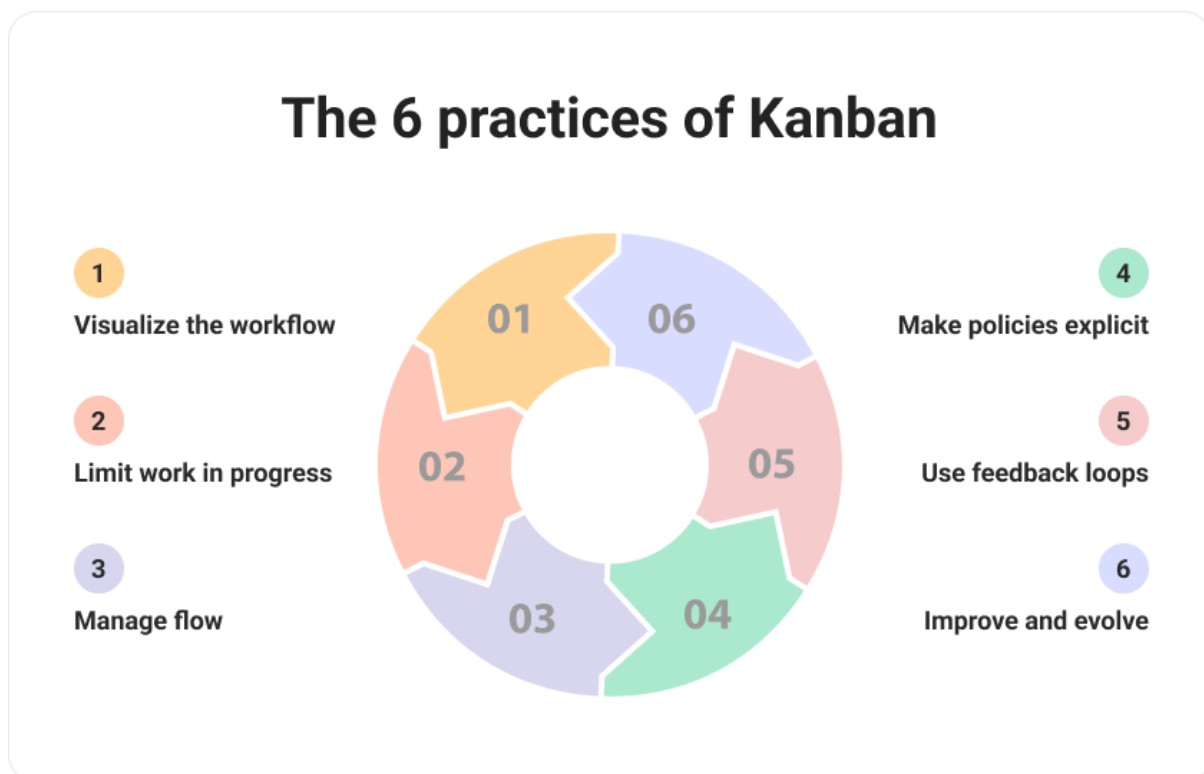


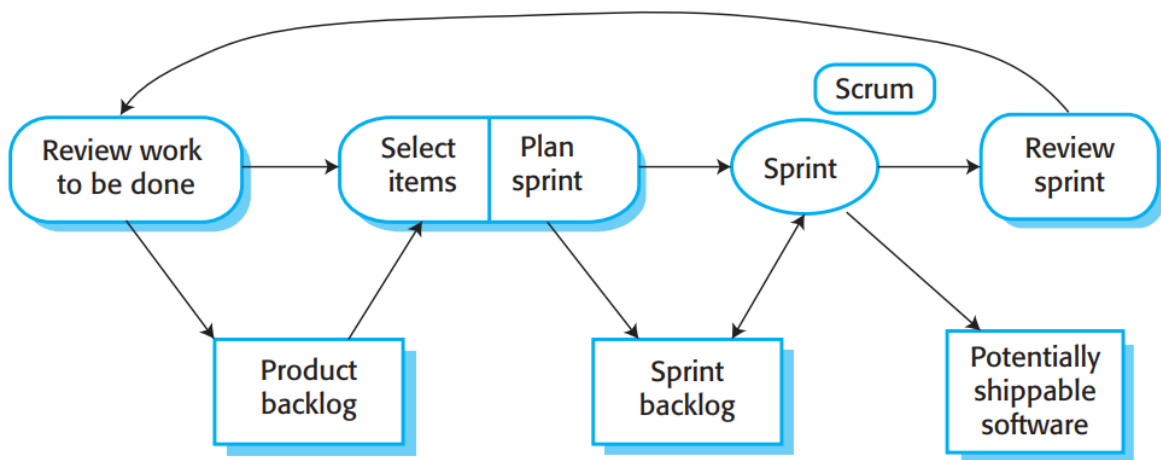
Figure 3.1: The six practices of Kanban (Bordio, 2025)

##### 3.1.1.2 Scrum

Scrum is an agile framework that manages repetitive and continuous development through time-boxed sprints, daily stand-ups, sprint reviews, and retrospectives (Sommerville, 2015).

Scrum is best suited for projects where deliverables can be developed incrementally with clear sprint goals.

For example, in creating a loan approval system, Scrum can be used to break down tasks into sprints: one sprint for data pre-processing, another for model training, another for explainability integration, and another for user testing and validation. The method's emphasis on stakeholder feedback and regular review helps keep the project aligned with fairness and regulatory requirements.



*Figure 3.2: The Scrum Sprint Cycle (Sommerville, 2015)*

### 3.1.1.3 Cross-Industry Standard Process for Data Mining (CRISP-DM)

CRISP-DM is a widely adopted methodology that provides a structured approach to data mining projects. It encompasses six phases: Business Understanding, Data Understanding, Data Preparation, Modelling, Evaluation, and Deployment. This iterative and flexible framework guides practitioners through the entire data mining process, from defining objectives to deploying models and evaluating their performance (IBM Corporation, 2021).

CRISP-DM is particularly well-suited for projects that involve complex data analysis, such as predictive modelling, classification, and clustering. Its adaptability makes it applicable across various industries, including finance, healthcare, marketing, and manufacturing. By emphasising a clear understanding of business goals and data, CRISP-DM facilitates the development of models that are both effective and aligned with organisational objectives.

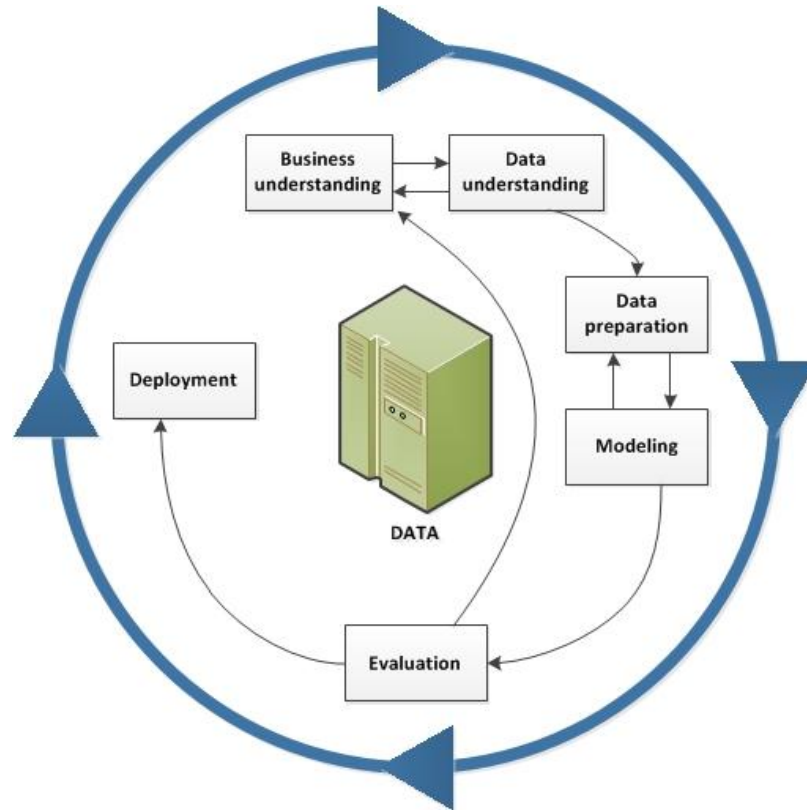


Figure 3.3: The CRISP\_DM Data mining life cycle (IBM Corporation, 2021)

### 3.1.2 Framework Used

For this project, we have chosen to adopt the CRISP-DM (Cross-Industry Standard Process for Data Mining) methodology, as it offers a structured and well-established framework for data-driven projects. CRISP-DM's six phases: Business Understanding, Data Understanding, Data Preparation, Modelling, and Evaluation, align closely with the steps needed for developing and evaluating machine learning models for loan approval automation.

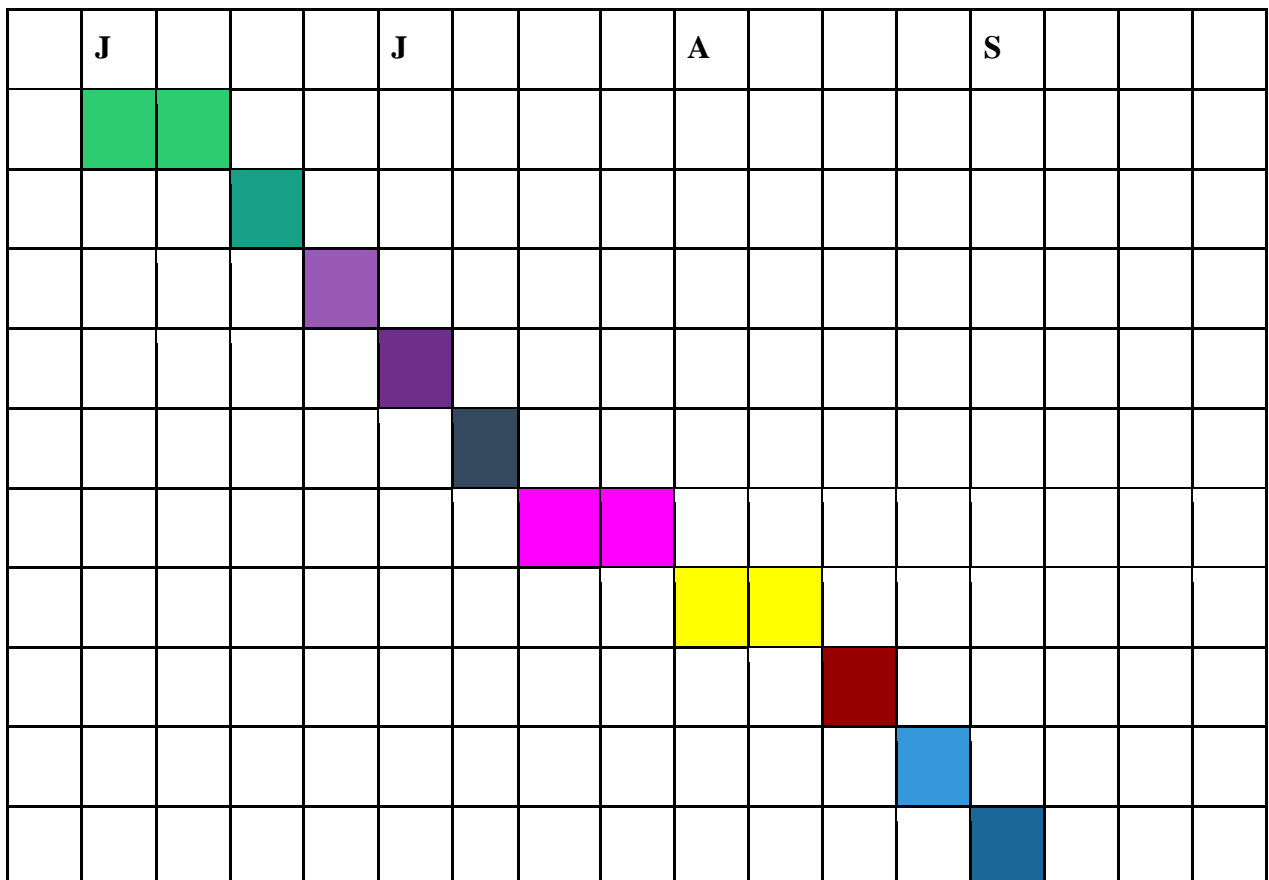
This methodology provides a clear, repeatable process while allowing for iteration as new data or insights emerge.

## 3.2 Project Planning

### 3.2.1 Initial Plan

	Start	Duration	Task
	01/06	2 weeks	Literature Review
	16/06	1 week	Methodology
	23/06	1 week	Requirement gathering
	01/07	1 week	Requirement organisation

*Table 3.4 Initial Project Planning*



*Table 3.5 Actual Project Plan*

24



[illegible]

In the initial plan, the Literature Review was scheduled to finish by 01/07/2025, but in reality, it extended to 10/07/2025 due to additional time required for sourcing and reviewing more recent research papers relevant to machine learning fairness in loan approvals.

This delay slightly shifted subsequent tasks, but the main reason for overlapping the Implementation and Evaluation phases was a deliberate choice not to follow a rigid waterfall method. Instead, an iterative approach was adopted, where evaluation was conducted in parallel with implementation to allow early identification of issues and rapid refinements.

This led to the introduction of Implementation: Part 3 and Evaluation: Part 3, accommodating additional features and improvements identified during testing. This flexible scheduling ensured the project remained on track for its final deadline while improving overall quality.

### 3.3 Ethical Considerations

### 3.3.1 Fairness and Non-Discrimination

It is essential to ensure that the loan approval system does not worsen existing biases, particularly those related to race, gender, or socioeconomic status. Implementing fairness-aware algorithms will help ensure that the model provides equitable access to credit for all applicants, as mentioned in [3.3.2](#).

### 3.3.2 Transparency and Explainability

Transparency in automated decisions is crucial for fostering trust in the system. Explainable AI tools will be integrated to provide clear, interpretable reasons for loan approvals or rejections. This will ensure that applicants and financial institutions understand the rationale behind each decision.

### 3.3.3 Data Privacy

The project must comply with data protection regulations, such as the General Data Protection Regulation (GDPR), to protect applicants' personal information. The data used in this project is anonymised and used solely for loan approval, ensuring applicants' privacy and control over their data.

## 4 Requirements

### 4.1 Requirement Gathering

This project builds directly on the study by [\(Nazim et al., 2023\)](#), using the same dataset structure, target label, and feature set for consistency and comparability. The work will use the same input features to predict whether a loan should be approved or not. The same binary classification target and the same core performance metrics, accuracy, precision, recall, and F1-score, will be applied to assess the predictive models.

The project will also build on the study by [\(Nallakaruppan et al., 2024\)](#), which focuses on interpretability using explainable AI. Interpretability in this study is defined as the ability of the model to make its decisions understandable through tools such as LIME and SHAP, which explain how individual features contribute to each prediction at both local and global levels.

This project will retain that focus on interpretability but aims to improve on it by exploring whether clearer visualisations, or refinements in feature importance analysis, can further enhance transparency and trust. In doing so, the project seeks to strengthen the practical value of explainable credit risk assessments while maintaining or improving predictive performance.

### 4.2 Organisation of Requirements

The project requirements have been categorised into functional and non-functional requirements. Functional requirements specify the key behaviours and features that the system must deliver, while non-functional requirements define the operational characteristics that the system should exhibit.

To prioritise the requirements of this project, we will use the MoSCoW method, which is a requirements prioritisation technique used in agile project management (Stray et al., 2022).

The acronym stands for:

- **Must Have:** Essential features that are critical for the success of the project.
- **Should Have:** Important features that are not essential but should be included if possible.
- **Could Have:** Desirable features that can enhance the project but are not critical.
- **Won't Have:** Features that are out of scope for the current iteration or project phase.

The MoSCoW method has been chosen for this project due to its clear and structured approach to prioritising requirements. It allows for flexibility and adaptability, which is crucial in a machine learning-based loan approval system where requirements may evolve.

## 4.3 Functional Requirements

I D	Description	Arrived from	Priority	Justification
1	Obtain the dataset from ( <a href="#">Nazim et al., 2023</a> )	Nazim et al., 2023	Must Have	This project will be built upon the findings of this <a href="#">paper</a> , and using the same dataset allows for direct comparison of results.
2	Data exploration and pre-processing	Nazim et al., 2023	Must Have	Data pre-processing is essential to ensure the dataset is ready for model training and produces reliable predictions.
3	Build the model using Random Forest, XGBoost and Ensemble methods.	Nazim et al., 2023	Must Have	These models are critical for testing and comparison with other algorithms for loan approval predictions.
4	Hyperparameter tuning	Nazim et al., 2023	Must Have	Hyperparameter tuning is crucial for optimising a model's performance by finding the best combination of settings that enhance accuracy, generalisation, and efficiency.
5	Evaluate the models using accuracy, precision, recall, and F1-score	Nazim et al., 2023	Must Have	Evaluation metrics are necessary to assess the effectiveness and fairness of the machine learning models in predicting loan approval outcomes.
6	Model comparison	Nazim et al., 2023	Must Have	Model comparison helps identify the most effective and reliable algorithm for the given dataset.
7	Implement Explainable AI tools	Nallakaruppan et al., 2024	Should Have	Incorporating explainability will ensure transparency in decision-making, meeting ethical and regulatory standards.
8	Model Deployment	(Srivastava, 2024)	Won't Have	Model deployment is essential for real-world applications, but is beyond the current scope of this project. The

				focus is on model development and evaluation.
9	User App	(Srivastava, 2024)	Won't Have	A user app would allow stakeholders to interact with the system, but it is outside the current project scope.

## 4.4 Non-functional Requirements

ID	Description	Arrived from	Priority	Justification
1	Anonymising data as part of research (for GDPR reasons)	Data regulations	Must Have	To ensure compliance with GDPR, identifiable personal data will not be stored or used in the research, ensuring ethical data handling.

## 5 Design

### 5.1 Machine Learning Pipeline

The design of this project follows a structured machine learning (ML) pipeline, which provides a systematic workflow for transforming raw data into actionable predictive insights. A machine learning pipeline is a conceptual framework that combines data pre-processing, model training, evaluation, and deployment stages, enabling reproducibility and scalability in ML tasks (Armehi et al., 2019).

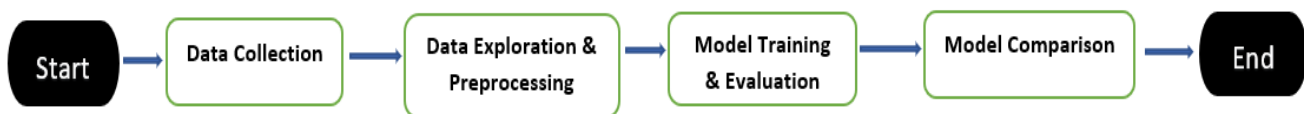
### 5.2 Pipeline Design

The machine learning pipeline for this project is divided into different segments that align with the CRISP-DM methodology. The pipeline begins with obtaining the dataset used in this [paper](#). The next step is data exploration through visualisations and examining the correlation between the different features. This is followed by data pre-processing, which involves handling missing values, encoding categorical features, and normalising numerical values. Following this, the model training phase includes Random Forest (RF), XGBoost, and a custom ensemble model that integrates predictions from multiple classifiers to improve overall performance.

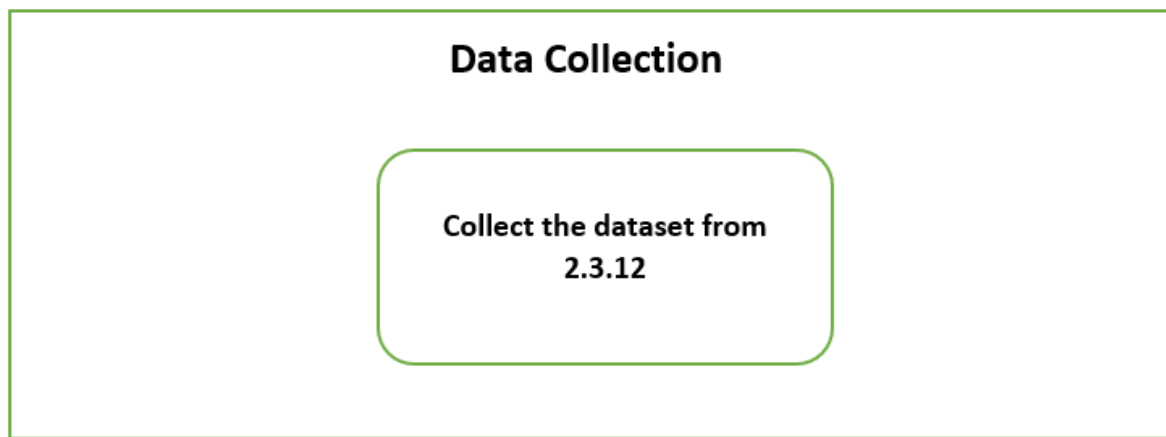
Random Forest is a robust ensemble method that combines multiple decision trees to reduce overfitting and improve generalisation. XGBoost, a gradient boosting algorithm, is known for its efficiency and accuracy in structured data problems. The ensemble model in this context will aggregate predictions from both RF and XGBoost, leveraging their strengths while mitigating individual weaknesses.

The evaluation phase compares these models using multiple metrics such as accuracy, precision, recall, and F1-score to ensure balanced performance. Beyond these metrics, interpretability tools like LIME and SHAP are applied to understand feature influence on individual predictions, ensuring transparency and fairness.

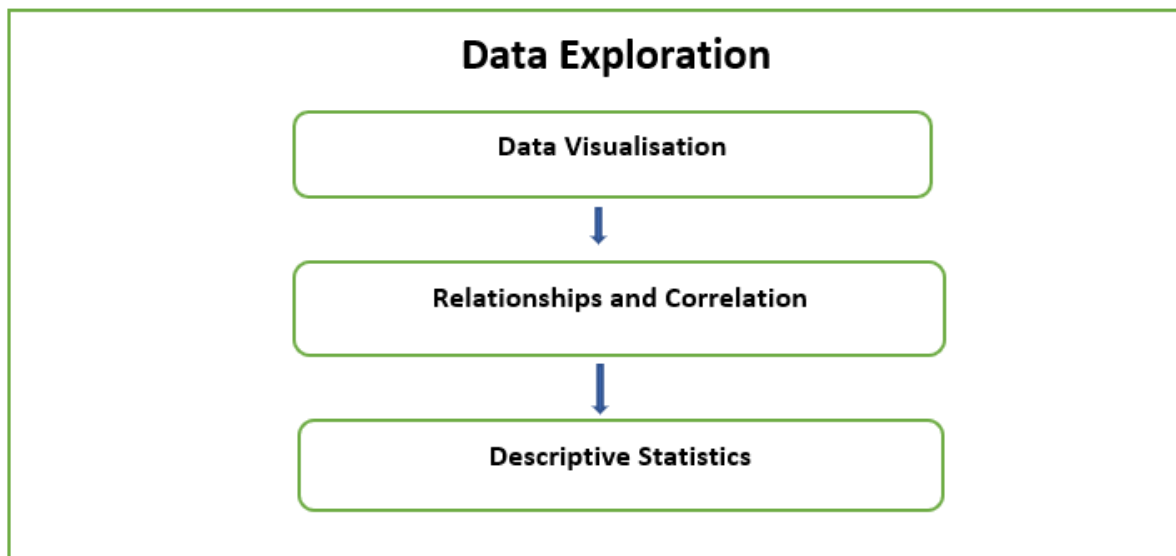
Finally, model comparison is based on both performance metrics and interpretability insights to determine the most effective model. The following diagram illustrates the full pipeline:



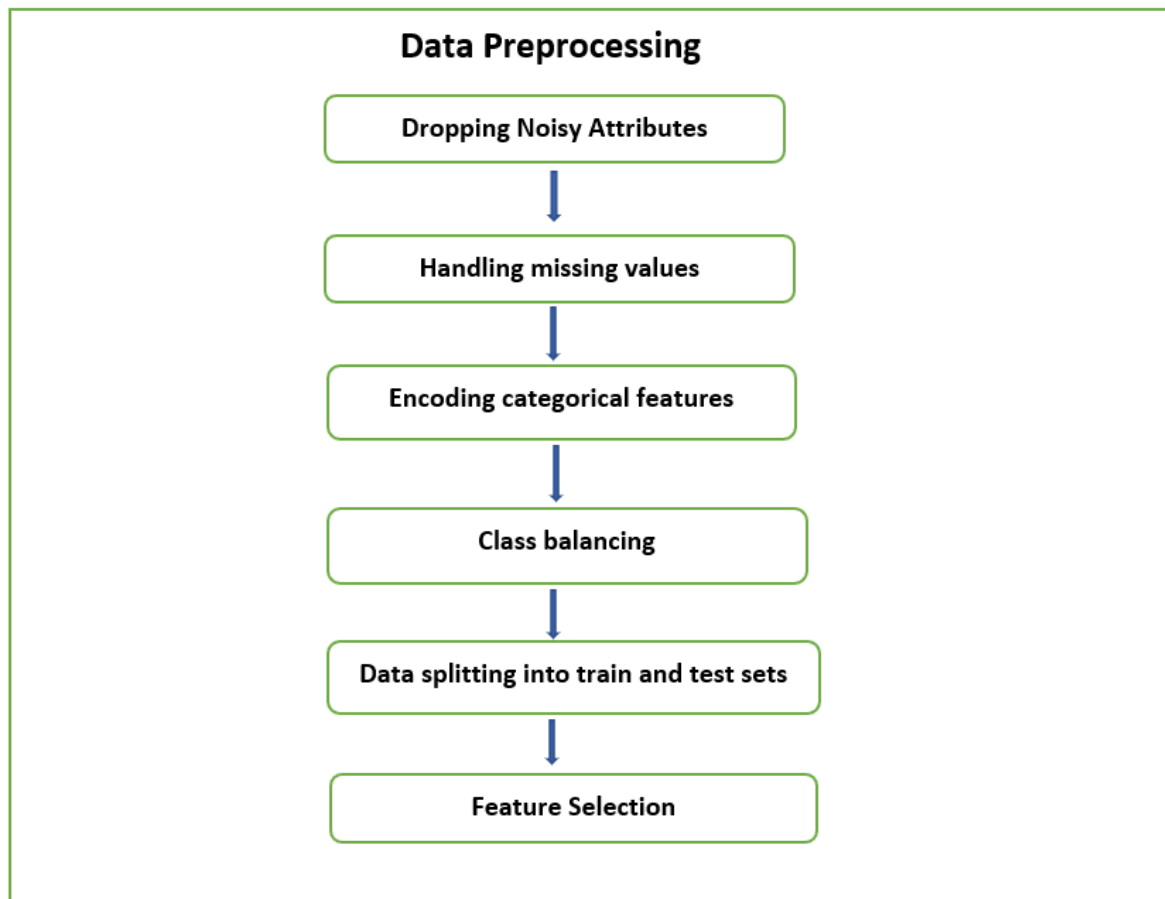
*Figure 5.1: Proposed Machine Learning Pipeline*



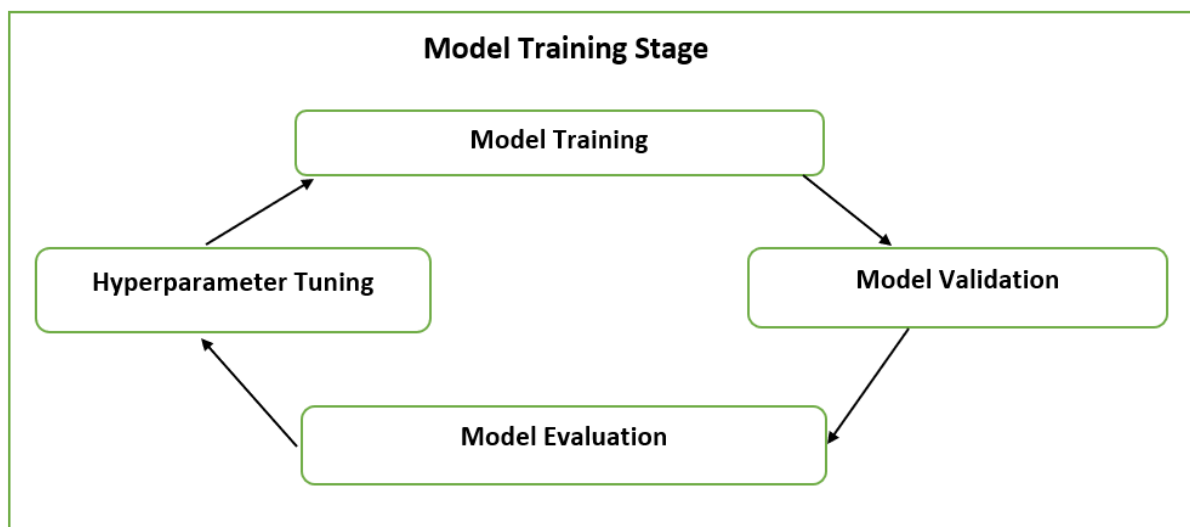
*Figure 5.2: Data Collection Stage*



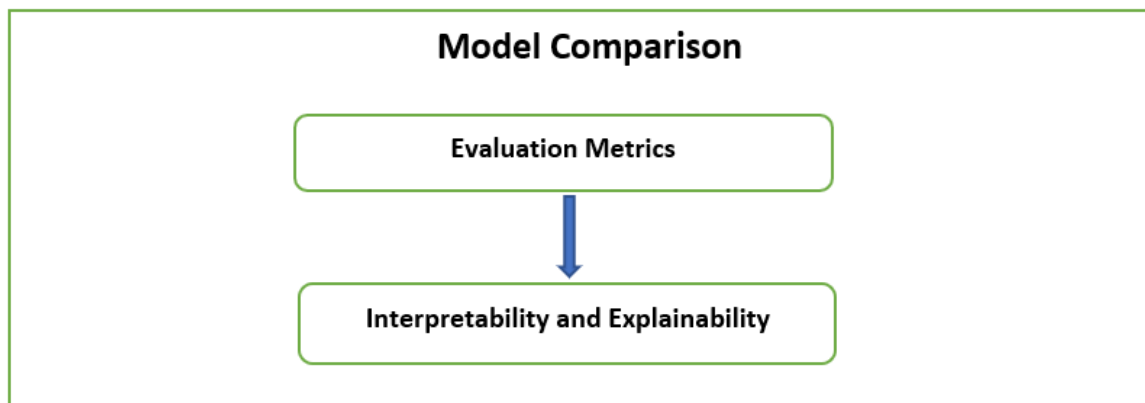
*Figure 5.3: Data Exploration Stage*



*Figure 5.4: Data Pre-processing Stage*



*Figure 5.5: Model Training Stage*



*Figure 5.6: Model Comparison Stage*

## 5.3 Machine Learning Models and Ensemble Techniques

### XGBoost Model

Extreme Gradient Boosting (XGBoost) is an optimised implementation of gradient boosting designed for speed and performance. It builds an ensemble of weak learners, typically decision trees, in a sequential manner where each new tree corrects the errors of the previous ones. XGBoost is particularly effective due to its use of regularisation, shrinkage, and parallel processing, which help reduce overfitting while maintaining high predictive accuracy.

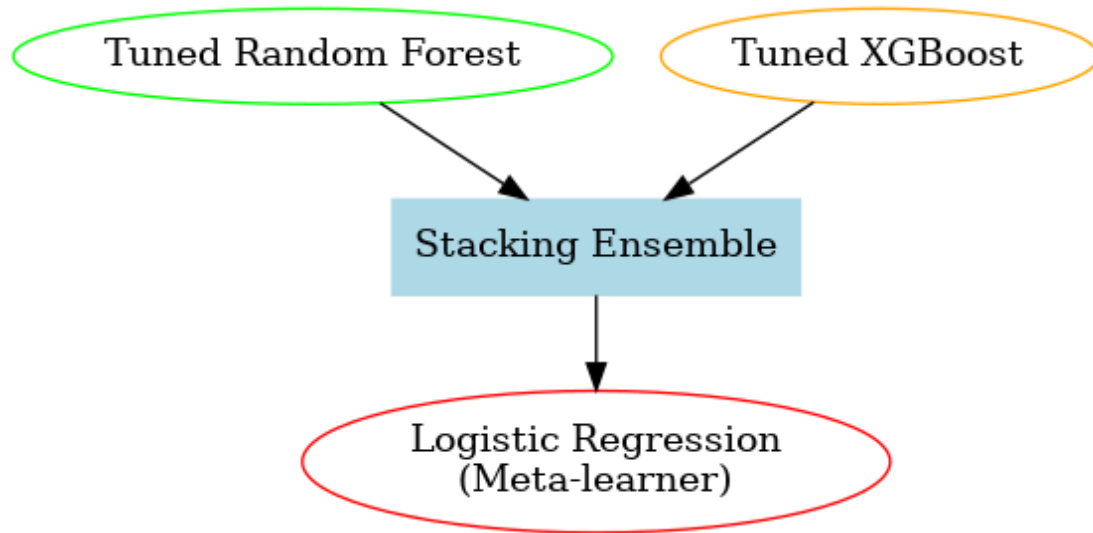
### Random Forest Model

Random Forest is an ensemble learning method that constructs a collection of decision trees during training and outputs either the majority class (for classification) or the average prediction (for regression) as the final result. Each tree is trained on a random subset of the data and features, which increases diversity among the trees and reduces overfitting. This makes Random Forest highly robust and capable of handling large, complex datasets.

### Stacking Ensemble Model

Stacking is an ensemble technique that combines multiple base models (level-0 models) to improve predictive performance. Unlike simple averaging, stacking involves training a meta-model (level-1 model) that learns how to best combine the predictions of the base models. This layered approach leverages the strengths of different algorithms, resulting in a more generalised and accurate final model.

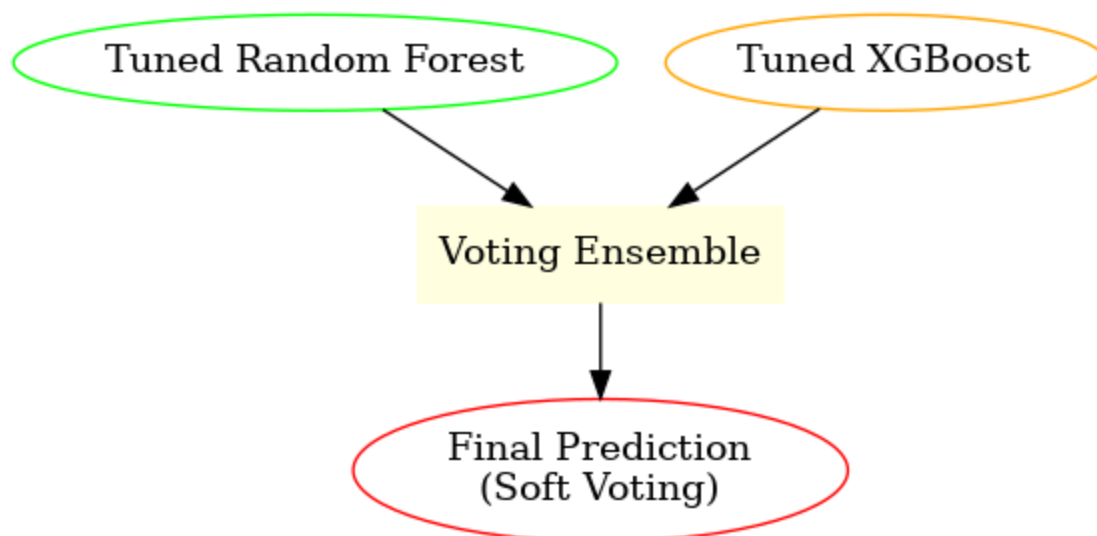




*Figure 5.7 Stacking Ensemble Architecture*

### Voting Ensemble Model

Voting is an ensemble technique that aggregates the predictions of multiple base learners to make a final decision. In classification, this is typically done through majority voting (hard voting), where the class predicted by the majority of models is chosen, or through averaging predicted probabilities (soft voting). Voting ensembles are straightforward yet effective, as they reduce variance and improve stability by combining multiple models.



*Figure 5.8 Voting Ensemble Architecture*

## 5.4 Evaluation Metrics

This project makes use of a range of metrics, including precision, recall, F1-score, accuracy, and the Area Under the ROC Curve (AUC). These metrics collectively capture different aspects of model performance, allowing for a more balanced and comprehensive evaluation of predictive accuracy, error trade-offs, and the model's ability to generalise to unseen data.

**Precision** measures the proportion of correct positive predictions. In other words, it shows how many applicants who were predicted as eligible are actually eligible. High precision indicates a low false positive rate.

$$Precision = \frac{TruePositives(TP)}{TruePositives(TP) + FalsePositives(FP)}$$

Where TP is the number of true positive guesses and FP is the number of false positive predictions.

**Recall** (also called sensitivity or true positive rate) measures the proportion of actual positives that are correctly identified. It shows how many eligible applicants the model correctly identifies. High recall indicates a low false negative rate.

$$Recall = \frac{TruePositives(TP)}{TruePositives(TP) + FalseNegatives(FN)}$$

Where FN is the number of false negative predictions.

**F1-score** is the harmonic mean of precision and recall, providing a single metric that balances both. It is beneficial when the dataset is imbalanced, since it penalises extreme differences between precision and recall.

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

**Accuracy** measures the proportion of all predictions (both positive and negative) that are correct. While easy to interpret, accuracy can be misleading in imbalanced datasets, as a model could achieve high accuracy by simply predicting the majority class.

$$Accuracy = \frac{TruePositives(TP) + TrueNegatives(TN)}{TP + TN + FP + FN}$$

Where TN is the number of true negative predictions.

**AUC (Area Under the Curve)** represents the area under the Receiver Operating Characteristic (ROC) curve, which plots the true positive rate (recall) against the false positive rate. The AUC score reflects the model's ability to distinguish between classes across all decision thresholds, with a value closer to 1 indicating better discriminative power.

$$AUC = \int_0^1 \text{TPR}(\text{FPR}) d(\text{FPR})$$

where

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

$$\text{FPR} = \frac{FP}{FP + TN}$$

Where TPR is the proportion of actual positives correctly identified, and FPR is the proportion of actual negatives incorrectly classified as positives

## 6 Implementation

### 6.1 Iteration 1

In this iteration, we focused on requirements 1, 2, 3, 4, and 5 as listed in this [subsection](#).

#### 6.1.1 Data Collection

The dataset used in this project is sourced from a publicly available Kaggle dataset (Chatterjee, 2021) and referenced in the study by Nazim et al. (2023). Therefore, no additional data collection will take place for this project.

#### 6.1.2 Exploratory Data Analysis

##### Development environment

The implementation of this project was carried out using Google Colab, which provides a cloud-based environment with pre-configured support for Python and popular machine learning libraries.

The primary libraries used for data exploration and pre-processing in the project included:

```
import matplotlib.pyplot as plt      #For plotting and visualisations
import seaborn as sns                #For statistical visualisations
import pandas as pd                  #For data manipulation & analysis
import numpy as np                   #For numerical operations
```

##### Feature Description

The dataset consists of 614 instances and includes 13 features. The target variable is **Loan\_Status**, which indicates whether a loan application was approved (Y) or rejected (N), forming a binary classification task. Table 6.1 shows the features and their respective descriptions.

Feature	Description
Loan_ID	Unique identifier for each loan application
Gender	Applicant's gender (Male/Female)
Married	Marital status of the applicant
Dependents	The number of dependents the applicant has
Education	Education level of the applicant (Graduate/Not Graduate)
Self_Employed	Whether the applicant is self-employed (Yes/No)
ApplicantIncome	The monthly income of the applicant
CoapplicantIncome	Monthly income of the co-applicant

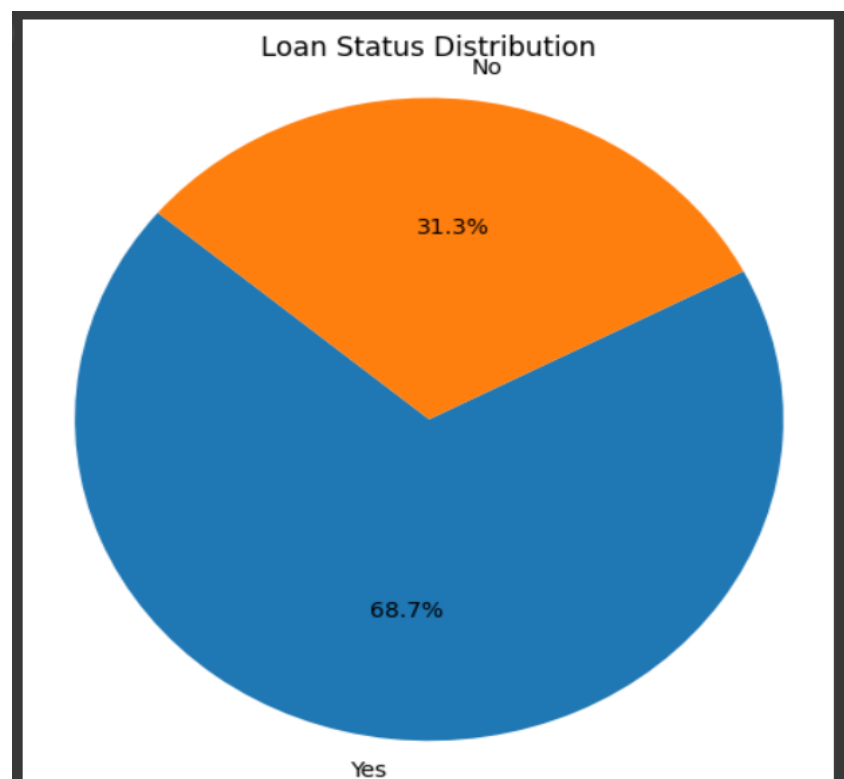
LoanAmount	Loan amount applied for (in thousands)
Loan_Amount_Term	Term of the loan in months
Credit_History	Credit history meets guidelines (1.0 = meets, 0.0 = does not meet)
Property_Area	Type of property area (Urban/Semiurban/Rural)
Loan_Status	Loan approval status (Y = Approved, N = Not Approved)

*Table 6.1 Feature Description*

### Visualisation of the **Loan\_Status** feature

This pie chart portrays the proportion of loan approvals (Yes) versus rejections (No) within the dataset. Approximately 68.7% of the applications were approved, while 31.3% were denied.

This class imbalance is important to highlight, as it can significantly influence the performance of classification models. Without proper handling, the model may become biased toward predicting the majority class (“Yes”), thereby failing to accurately identify high-risk (rejected) applicants. This distribution justifies the need for data balancing techniques in the pre-processing phase to ensure fair and reliable predictions across both classes.



*Figure 6.1: Loan Status Distribution*

## Correlation Analysis

The Pearson correlation coefficient measures the strength and direction of the linear relationship between pairs of continuous variables. It ranges from -1 to 1, where a value of 1 indicates a perfect positive linear relationship, -1 indicates a perfect negative linear relationship, and 0 implies no linear correlation between the variables.

Figure 6.2 displays the pairwise Pearson correlation coefficients between key numerical features in the dataset, including **ApplicantIncome**, **CoapplicantIncome**, **LoanAmount**, **Loan\_Amount\_Term**, and **Dependents**.

The most notable correlation is observed between **ApplicantIncome** and **LoanAmount**, with a coefficient of 0.57, suggesting a moderate positive relationship. This indicates that applicants with higher incomes tend to request or qualify for larger loans. **CoapplicantIncome** also shows a weak positive correlation with **LoanAmount** ( $r = 0.19$ ), indicating its limited but still relevant influence.

Other relationships, such as those involving **Dependents** and **Loan\_Amount\_Term**, demonstrate very low or near-zero correlation values, indicating minimal linear dependency with different features.

This correlation analysis helps in identifying multicollinearity issues and supports feature selection decisions during model development. In this case, no strong multicollinearity is observed, allowing all features to be retained for further analysis.

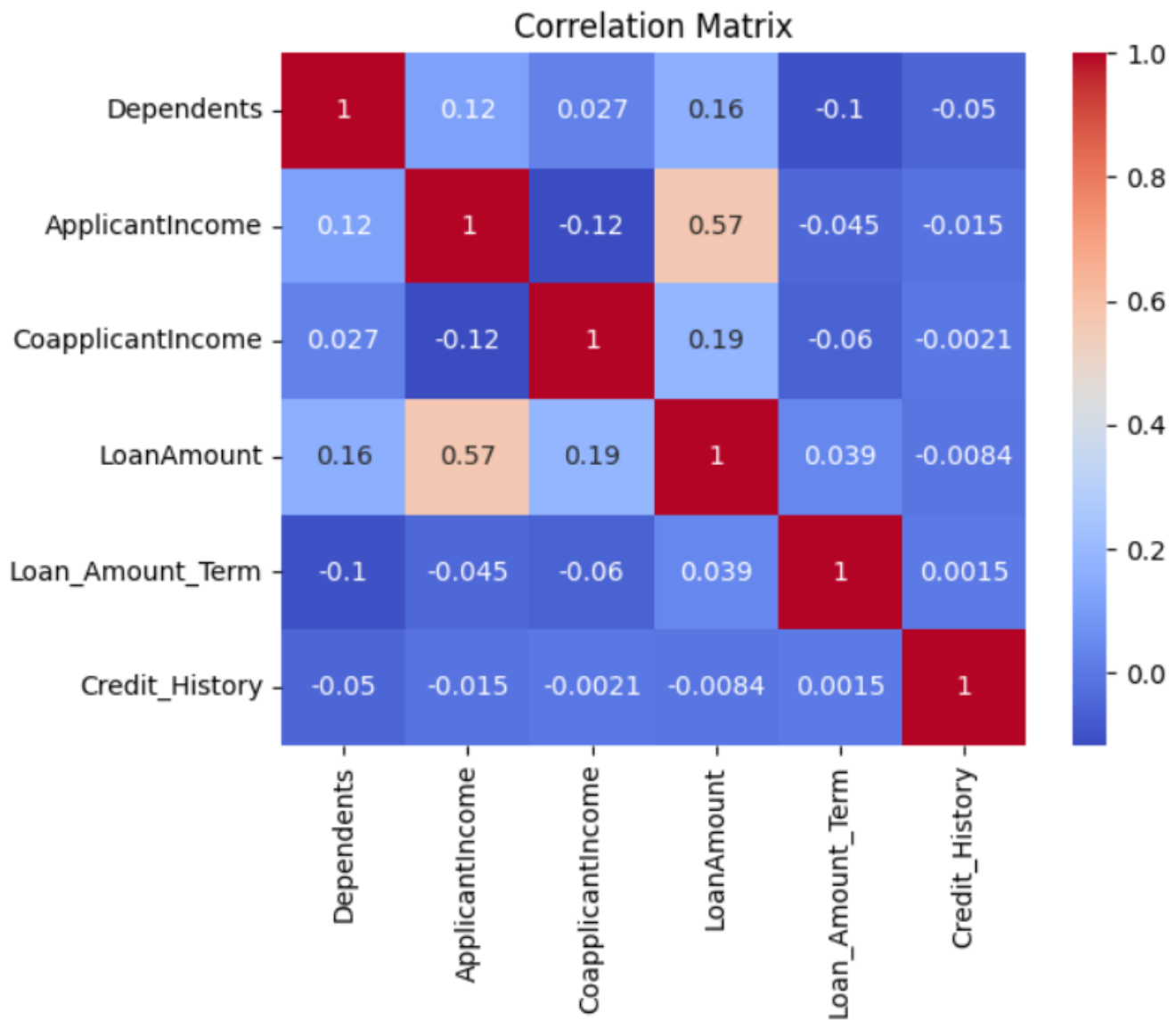


Figure 6.2: Correlation Matrix

## Statistical Analysis

Figure 6.3 summarises the descriptive statistics of the five key numerical features in the dataset. The **ApplicantIncome** feature has a wide range, with values ranging from 150 to 81,000, and a high standard deviation (6109), indicating the presence of income outliers or significant income disparity among applicants. Similarly, **CoapplicantIncome** also varies substantially, with a maximum value of 41,667 and a mean of 1621.25. Both income features have a median lower than their mean, suggesting a right-skewed distribution.

The **LoanAmount** has a mean of approximately 146 and a median of 128, with most loans falling between 100 and 168 (interquartile range). The maximum loan amount recorded is 700, again indicating potential outliers. The **Loan\_Amount\_Term** is fairly consistent across applicants, with most terms concentrated at 360 months (30 years), as reflected in the 25th, 50th, and 75th percentiles.

Lastly, the **Dependents** feature has a mean of 0.76, with the majority of applicants reporting 0 dependents. The max number of dependents is 3, and the interquartile range spans from 0 to 2.

These statistics are critical for identifying data skewness, outliers, and potential pre-processing requirements such as normalisation or imputation, which influence model performance and fairness.

	Dependents	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term
count	599.000000	614.000000	614.000000	592.000000	600.000000
mean	0.762938	5403.459283	1621.245798	146.412162	342.000000
std	1.015216	6109.041673	2926.248369	85.587325	65.12041
min	0.000000	150.000000	0.000000	9.000000	12.000000
25%	0.000000	2877.500000	0.000000	100.000000	360.000000
50%	0.000000	3812.500000	1188.500000	128.000000	360.000000
75%	2.000000	5795.000000	2297.250000	168.000000	360.000000
max	3.000000	81000.000000	41667.000000	700.000000	480.000000

*Figure 6.3: Descriptive Statistics of Numerical Features*

Figure 6.4 summarises the categorical features in the dataset, including the number of entries, unique values, the most common category, and its frequency. Each loan is uniquely identified (**Loan\_ID**), and most applicants are male, married, and graduates. The majority are not self-employed and have a good credit history. “Semi-urban” is the most common property area, and most loans (422 out of 614) were approved.

Missing values in features such as **Gender**, **Married**, **Credit History**, and **Property Area** highlight the need for imputation. The class imbalance in **Loan\_Status** was also addressed during pre-processing using **SMOTE** to improve model performance and fairness.

	Loan_ID	Gender	Married	Education	Self_Employed	Credit_History	Property_Area	Loan_Status
count	614	601	611	614	582	564	614	614
unique	614	2	2	2	2	2	3	2
top	LP002990	Male	Yes	Graduate	No	Good Credit History	Semiurban	Yes
freq	1	489	398	480	500	475	233	422

*Figure 6.4: Summary of Categorical Variables*



## 6.1.3 Data Pre-processing

### Dropping Noisy Attributes

The `Loan_ID` column was removed, since it serves as a unique identifier and carries no predictive value for the loan approval outcome.

```
# Drop 'Loan_ID' as it is a unique identifier and not useful for prediction
df = df.drop('Loan_ID', axis=1)
```

Additionally, the `Dependents` column originally included a categorical value of '3+', which was replaced with the numeric value 3 to facilitate numerical processing. The entire column was then converted to a numeric format to ensure consistency in subsequent model training steps. After these adjustments, the dataset was verified using the `head()` function to inspect the first few rows and confirm successful pre-processing.

```
# In the Dependents column, replace '3+' with 3 and convert to numeric
df['Dependents'] = df['Dependents'].replace('3+', 3)
df['Dependents'] = pd.to_numeric(df['Dependents'], errors='coerce')
```

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Property_Area	Loan_Status
0	LP001002	Male	No	0	Graduate	No	5849	0.0	NaN	360.0	1.0	Urban	Y
1	LP001003	Male	Yes	1	Graduate	No	4583	1508.0	128.0	360.0	1.0	Rural	N
2	LP001005	Male	Yes	0	Graduate	Yes	3000	0.0	66.0	360.0	1.0	Urban	Y
3	LP001006	Male	Yes	0	Not Graduate	No	2583	2358.0	120.0	360.0	1.0	Urban	Y
4	LP001008	Male	No	0	Graduate	No	6000	0.0	141.0	360.0	1.0	Urban	Y

Figure 6.6: Before dropping the `Loan_ID` feature

	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Property_Area	Loan_Status
0	Male	No	0.0	Graduate	No	5849	0.0	NaN	360.0	1.0	Urban	Y
1	Male	Yes	1.0	Graduate	No	4583	1508.0	128.0	360.0	1.0	Rural	N
2	Male	Yes	0.0	Graduate	Yes	3000	0.0	66.0	360.0	1.0	Urban	Y
3	Male	Yes	0.0	Not Graduate	No	2583	2358.0	120.0	360.0	1.0	Urban	Y
4	Male	No	0.0	Graduate	No	6000	0.0	141.0	360.0	1.0	Urban	Y

Figure 6.7: After dropping the `Loan_ID` feature

### Handling Missing Values

Missing values in the dataset were addressed using imputation techniques tailored to the type of variable. For categorical features such as `Gender`, `Married`, `Dependents`, `Self_Employed`, and `Credit_History`, the most frequent value (mode) was used for imputation. This approach preserves the most likely category without introducing bias toward a specific class.

```
# Using mode imputation for categorical features
df['Gender'] = df['Gender'].fillna(df['Gender'].mode()[0])
df['Married'] = df['Married'].fillna(df['Married'].mode()[0])
df['Dependents'] = df['Dependents'].fillna(df['Dependents'].mode()[0])
df['Self_Employed'] = df['Self_Employed'].fillna(df['Self_Employed'].mode()[0])
df['Credit_History'] = df['Credit_History'].fillna(df['Credit_History'].mode()[0])
```

For numerical features like `LoanAmount` and `Loan_Amount_Term`, the median was used to fill in missing values. Median imputation is less sensitive to outliers and is appropriate for skewed distributions. These strategies helped ensure data completeness without distorting the underlying patterns essential for model training.

```
# Using median imputation for numeric features
df['LoanAmount'] = df['LoanAmount'].fillna(df['LoanAmount'].median())
df['Loan_Amount_Term'] =
df['Loan_Amount_Term'].fillna(df['Loan_Amount_Term'].median())
```

	0
Gender	13
Married	3
Dependents	15
Education	0
Self_Employed	32
ApplicantIncome	0
CoapplicantIncome	0
LoanAmount	22
Loan_Amount_Term	14
Credit_History	50
Property_Area	0
Loan_Status	0

	0
Gender	0
Married	0
Dependents	0
Education	0
Self_Employed	0
ApplicantIncome	0
CoapplicantIncome	0
LoanAmount	0
Loan_Amount_Term	0
Credit_History	0
Property_Area	0
Loan_Status	0

Figure 6.8: Before and after handling missing values

## Encoding Categorical Features

To prepare categorical data for machine learning algorithms, label encoding was applied. Label encoding is a technique that converts categorical text values into numeric form by assigning a unique integer to each category. This is necessary because most machine learning models cannot work directly with string labels.

In this project, the `LabelEncoder` from `scikit-learn` was used to encode the categorical features as well as the target variable `Loan_Status`. This transformation enabled the model to process these variables numerically while preserving their categorical meaning. Label encoding was suitable in this case because the features contained a limited number of discrete, non-ordinal categories.

Figure 6.9 shows what the dataset looks like after label encoding.

	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Property_Area	Loan_Status
0	1	0	0.0	0	0	5849	0.0	128.0	360.0	1.0	2	1
1	1	1	1.0	0	0	4583	1508.0	128.0	360.0	1.0	0	0
2	1	1	0.0	0	1	3000	0.0	66.0	360.0	1.0	2	1
3	1	1	0.0	1	0	2583	2358.0	120.0	360.0	1.0	2	1
4	1	0	0.0	0	0	6000	0.0	141.0	360.0	1.0	2	1

*Figure 6.9: Features after label encoding*

## Handling Class Imbalance and Data Splitting

To prepare the data for model training, the dataset was split into predictor attributes and the target variable. The variable `X` was assigned all columns except the last one using `df.iloc[:, :-1].values`, representing the input features. The variable `y` was assigned the last column using `df.iloc[:, -1].values`, which contains the target variable `Loan_Status`. This separation is a necessary step before performing train-test splitting and model fitting.

```
X = df.iloc[:, :-1].values    # predictor attributes
y = df.iloc[:, -1].values    # target attribute
```

The target variable, `Loan_Status`, was imbalanced, with 422 approved loans (1) and 192 rejected loans (0). This imbalance can lead machine learning models to become biased toward the majority class, reducing their ability to identify high-risk applicants accurately.

To address this issue, the dataset was split into training and testing sets using an 80:20 ratio. The Synthetic Minority Over-Sampling Technique (SMOTE) was then applied to the training data to balance the class distribution. SMOTE works by generating synthetic samples for the minority class (0) rather than simply duplicating existing ones.

It achieves this by identifying the `k` nearest neighbours of a minority instance and interpolating new points along the line segments connecting the original point and its neighbours. This helps the model learn more generalisable decision boundaries.

up2280493

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)
oversample = SMOTE()
X_train_sm, y_train_sm = oversample.fit_resample(X_train, y_train) # using smote
```

As a result of applying SMOTE, the training set became balanced, with 342 samples for each class, ensuring that the model is not biased toward the majority class during training and improving fairness in predictions.

## 6.1.4 Random Forest Model Training

### Feature Selection

A Random Forest Classifier was initially trained on the original (imbalanced) training data to evaluate baseline performance and to extract feature importance values. This helped establish a benchmark and provided insight into which features were most influential in predicting loan approval outcomes.

```
model = RandomForestClassifier()
model.fit(X_train, y_train)
```

As shown in Figure 6.10, the most influential feature was **Credit\_History**, followed by **ApplicantIncome**, **LoanAmount**, and **CoapplicantIncome**. These features significantly impact loan approval predictions, suggesting that applicants with strong credit history and higher income are more likely to be approved.

In contrast, features such as **Self\_Employed**, **Education**, and **Gender** contributed relatively little to the model's decision-making. This insight is valuable for feature selection, helping to identify and eliminate variables with minimal predictive value in later model optimisation phases.

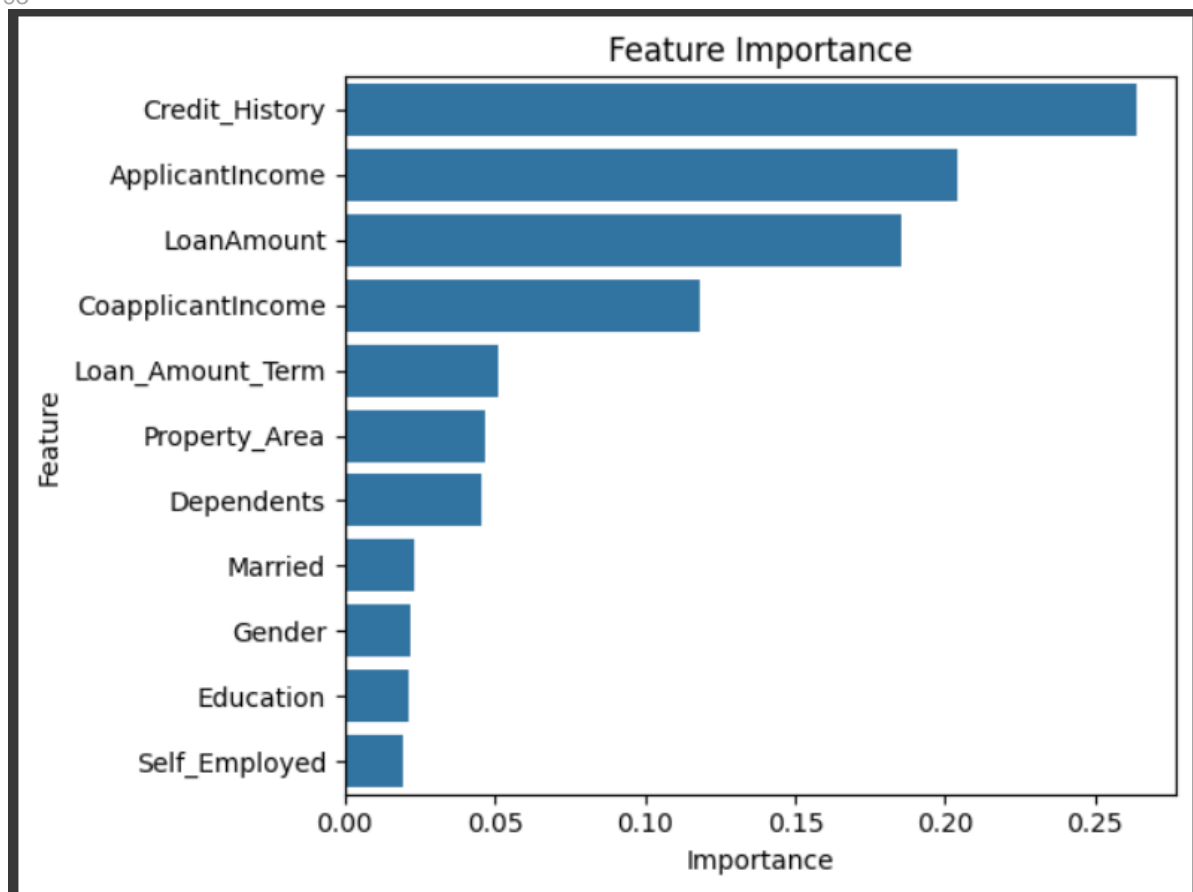


Figure 6.10: Feature Importance Analysis from the Random Forest Model

### Fitting Random Forest to the Balanced Data: Full Features vs Selected Features

In this section, we compare the performance of the Random Forest (RF) model when trained on the balanced dataset with all features versus the balanced dataset with only selected features. Initially, the RF model was trained on the entire dataset, which includes all available features.

After performing feature selection based on importance, a second model was trained using only the most important features to assess if reducing the feature set impacts performance. Both models were evaluated using various performance metrics, as shown in the table, and the results were compared to determine which approach yields better performance.

Metric	Model with All Features	Model with Selected Features
Accuracy	86.40%	85.81%
Precision	82.85%	81.81%
Recall	91.81%	92.10%
F1 Score	87.10%	86.66%
AUC Score	91.21%	90.05%

Table 6.2: Random Forest performance with all the features vs with selected features

Given the better F1 score and the ability to capture more nuanced patterns in the data, the model with all features is the preferred choice for this project. It strikes a good balance between precision and recall, offering a robust model for predicting loan approvals while minimising errors.

### 6.1.5 Random Forest Model Tuning

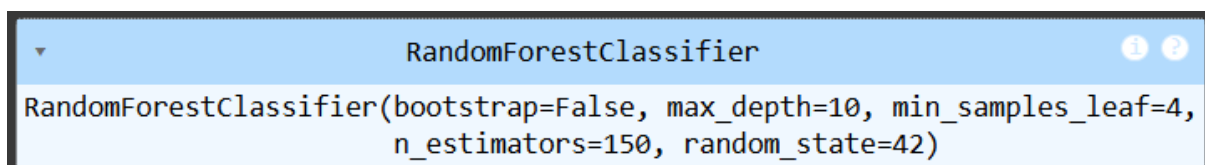
In this phase, `GridSearchCV` was used to fine-tune the hyperparameters of the Random Forest Classifier to optimise its performance. The hyperparameters tested included `max_depth`, which controls the depth of the trees, with values ranging from 5 to 20, and an option for no limit (`None`).

The `n_estimators` parameter, which determines the number of trees in the forest, was tested with values 50, 100, and 150. The `min_samples_split` parameter, which specifies the minimum number of samples required to split a node, was tested with values 2, 10, and 20. Additionally, `min_samples_leaf`, the minimum number of samples required at a leaf node, was tested with values 1, 2, and 4.

The `max_features` parameter, which dictates the maximum number of features considered for splitting a node, was tested with the values `'sqrt'` and `'log2'`. Lastly, the `bootstrap` parameter, indicating whether bootstrap samples are used when building trees, was tested with values `True` and `False`.

A 5-fold cross-validation was used to evaluate the performance of each set of hyperparameters, with the F1 score being the chosen metric for assessing the model's performance. The best-performing configuration of hyperparameters was selected based on the highest F1 score, ensuring that the model achieved optimal performance for the given dataset.

After completing the grid search, the best-performing combination of hyperparameters was identified, and the corresponding F1 score was recorded. The model was fitted with the optimal parameters, as illustrated in Figure 6.11.



```
RandomForestClassifier(bootstrap=False, max_depth=10, min_samples_leaf=4,
n_estimators=150, random_state=42)
```

*Figure 6.11: Random Forest Model with Tuned Hyperparameters*

```

Confusion Matrix:
[[262  80]
 [ 16 326]]
Accuracy: 0.8596491228070176
Precision: 0.8029556650246306
Recall: 0.9532163742690059
F1 Score: 0.8716577540106952
AUC Score: 0.9031069388871791

Classification Report:
              precision    recall  f1-score   support

     No         0.94         0.77         0.85         342
     Yes         0.80         0.95         0.87         342

 accuracy              0.86         684
  macro avg           0.87         0.86         0.86         684
 weighted avg           0.87         0.86         0.86         684

```

*Figure 6.12: Performance metrics for the tuned Random Forest model*

### 6.1.6 Model Comparison: Before and After Hyperparameter Tuning

The Random Forest model showed slight differences in performance after hyperparameter tuning. Before tuning, the model achieved an accuracy of 86.40%, which dropped to 85.96% after tuning.

Despite this small decrease in accuracy, the tuned model showed significant improvements in recall, increasing from 91.81% to 95.32%. This increase in recall means the model became better at identifying true loan approvals, which is a crucial improvement in the context of predicting loan approvals.

However, the precision for the 'Yes' class (loan approval) decreased from 82.85% to 80.30%, indicating that the model became more lenient in predicting approvals, leading to an increase in false positives.

The F1 score showed a minor improvement, increasing from 87.10% to 87.17%, which reflects the balance between precision and recall. The AUC score, however, slightly decreased from 0.9122 to 0.9031, indicating that the model's ability to distinguish between the classes (approved vs. denied loans) decreased slightly after tuning.

Given the context of the loan approval system, where the goal is to minimise false negatives (missed approvals), the tuned model would be considered more suitable, despite the slight reduction in accuracy and AUC. The improved recall outweighs the minor decrease in precision, making the tuned model more effective at identifying potential loan approvals.

Metric	Before Tuning	After Tuning
Accuracy	86.40%	85.96%
Precision	82.85%	80.30%
Recall	91.81%	95.32%
F1 Score	87.10%	87.17%
AUC Score	0.9122	0.9031

*Table 6.3: Random Forest performance before and after hyperparameter tuning*

### 6.1.7 Review

Throughout the process of building and optimising the Random Forest model, several important aspects stood out. Firstly, the importance of proper data pre-processing cannot be overstated. Handling missing values, encoding categorical features, and addressing class imbalance were critical steps in ensuring that the model performed effectively.

The application of SMOTE for balancing the dataset was particularly valuable in preventing the model from being biased toward the majority class, especially in a real-world application like loan approval prediction, where false negatives could have significant financial consequences.

One of the key takeaways from this iteration was understanding the trade-offs involved in hyperparameter tuning. While tuning improved recall, there was a slight reduction in accuracy and AUC. This was expected, as tuning often shifts the model's focus toward minimising false negatives, which is crucial in financial decision-making. The balancing act between precision and recall is particularly significant in this context, as the cost of missed approvals (false negatives) is much higher than the cost of incorrect rejections (false positives). The next step would be to train a different model to see whether the performance metrics can be improved.



## 6.2 Iteration 2

### 6.2.1 Data Collection and Pre-processing

In this iteration, the dataset and features remain the same as in the first iteration. No further pre-processing is required to build the XGBoost model.

### 6.2.2 XGBoost Model Training

In this section, we created and trained an XGBoost model for binary classification. The model was initialised with the `eval_metric='logloss'` parameter, which is the recommended evaluation metric for binary classification tasks. The random state was set to 42 to ensure reproducibility of results. After initialising the model, it was fitted to the training data (`X_train_sm` and `y_train_sm`), which had been pre-processed and balanced using SMOTE.

```
#Creating and fitting the model
xgb_model = XGBClassifier(
    eval_metric='logloss',    # Recommended for binary classification
    random_state=42
)

xgb_model.fit(X_train_sm, y_train_sm)
```

Results:

```
Confusion Matrix:
[[274  68]
 [ 49 293]]
Accuracy: 0.8289473684210527
Precision: 0.8116343490304709
Recall: 0.8567251461988304
F1 Score: 0.833570412517781
AUC Score: 0.8929414178721657

Classification Report:
              precision    recall  f1-score   support

      No         0.85        0.80        0.82         342
      Yes         0.81        0.86        0.83         342

   accuracy                0.83         684
  macro avg         0.83        0.83        0.83         684
 weighted avg         0.83        0.83        0.83         684
```

*Figure 6.13: XGBoost performance metrics*

### 6.2.3 XGBoost Hyperparameter Tuning

In this part, we performed hyperparameter tuning for the XGBoost model using GridSearchCV. The parameter grid defined the hyperparameters we wanted to tune, as shown in the table below:

Hyperparameter	Description
<code>n_estimators</code>	The number of boosting rounds (trees) to use.
<code>max_depth</code>	The maximum depth of each tree.
<code>learning_rate</code>	The rate at which the model adjusts based on the loss function.
<code>subsample</code>	The fraction of the data to be used for each boosting round.
<code>colsample_bytree</code>	The fraction of features to sample for each tree.

*Table 6.4: XGBoost model hyperparameter description*

We then set up the XGBoost classifier (XGBClassifier) with the evaluation metric set to `'logloss'`, which is suitable for binary classification tasks. Using GridSearchCV, we performed a 5-fold cross-validation to test different combinations of the hyperparameters defined in the grid. The scoring method was set to F1 to optimise the balance between precision and recall.

```
xgb = XGBClassifier(random_state=42, eval_metric='logloss')

grid_search = GridSearchCV(
    estimator=xgb,
    param_grid=param_grid,
    scoring='f1',
    cv=5,
    n_jobs=-1,
    verbose=1
)
```

Finally, the model was trained on the balanced dataset (`X_train_sm` and `y_train_sm`), and the best hyperparameters were selected based on the highest F1 score.

```
Best Parameters: {'colsample_bytree': 0.8, 'learning_rate': 0.01, 'max_depth': 10, 'n_estimators': 200, 'subsample': 0.8}
Best F1 Score: 0.8751709351190264
```

*Figure 6.14: Best hyperparameters for XGBoost Model*

```

Confusion Matrix:
[[266  76]
 [ 22 320]]
Accuracy: 0.8567251461988304
Precision: 0.8080808080808081
Recall: 0.935672514619883
F1 Score: 0.8672086720867209
AUC Score: 0.9015936527478541

Classification Report:
              precision    recall  f1-score   support

      No         0.92         0.78         0.84         342
      Yes         0.81         0.94         0.87         342

   accuracy                   0.86         684
  macro avg         0.87         0.86         0.86         684
 weighted avg         0.87         0.86         0.86         684

```

*Figure 6.15: Performance metrics for tuned XGBoost model*

## 6.2.4 Model Comparison: Before and After Hyperparameter Tuning

The performance of the XGBoost model was evaluated both before and after performing hyperparameter tuning. The results are shown in the table below.

Metric	XGBoost Before Tuning	XGBoost After Tuning
Accuracy	0.83	0.86
Precision	0.81	0.81
Recall	0.86	0.94
F1 Score	0.83	0.87
AUC Score	0.89	0.90

*Table 6.5: XGBoost performance before and after hyperparameter tuning*

The hyperparameter tuning has undoubtedly had a positive impact on the XGBoost model's performance. While the accuracy saw a modest increase, the major improvements were observed in recall, F1 score, and AUC score. The significant increase in recall is particularly important in the context of loan approval, as it means the model is now better at identifying loan approvals (the positive class). However, precision remained the same, which is a good sign since it means the model did not start predicting more false positives when it became more sensitive to approvals.

## 6.2.5 Review

In this iteration, we learned how hyperparameter tuning can significantly improve model performance, especially in terms of recall, which is critical for identifying loan approvals. We realised that accuracy alone is not sufficient, and precision, recall, and AUC are better indicators of the model's effectiveness in imbalanced datasets. Tuning helped balance these metrics, improving the model's ability to detect positive cases without compromising precision too much.

The main challenge was selecting the right evaluation metrics and understanding how recall can be improved at the cost of precision. Moving forward, experimenting with ensemble methods or refining the hyperparameter grid could provide further improvements.

## 6.3 Iteration 3

### 6.3.1 Data Collection and Pre-processing

The dataset used in this iteration remains the same, and no further pre-processing was done.

### 6.3.2 Stacking Ensemble Model Training

In this phase, we developed a stacking ensemble model to combine the predictive power of both the Random Forest and XGBoost classifiers, which had already been individually tuned. Stacking is an advanced ensemble technique where multiple base learners are trained, and their predictions are combined by a meta-learner to improve overall model performance.

```
# Base learners
estimators = [
    ('rf', final_model), # already tuned
    ('xgb', best_xgb_model) # already tuned
]

# Meta-learner (final estimator)
meta_learner = LogisticRegression()

# Stacking classifier
stacking_model = StackingClassifier(
    estimators=estimators,
    final_estimator=meta_learner,
    cv=5,
    n_jobs=-1,
    passthrough=False
)
```

For this approach, Random Forest and XGBoost served as the base learners, while a Logistic Regression model acted as the meta-learner, responsible for learning how best to combine the outputs of the base models. The stacking classifier was configured with 5-fold cross-validation, ensuring robust performance estimation and minimising overfitting.

This method allowed us to leverage the strengths of both algorithms: Random Forest's ability to handle complex feature interactions and XGBoost's high predictive power. The logistic regression meta-learner provided a linear combination to enhance generalisation, and the stacking ensemble aimed to improve classification metrics compared to using the individual models alone.

```

Confusion Matrix:
[[266  76]
 [ 28 314]]
Accuracy: 0.847953216374269
Precision: 0.8051282051282052
Recall: 0.9181286549707602
F1 Score: 0.8579234972677595
AUC Score: 0.8964553195855135

Classification Report:

```

	precision	recall	f1-score	support
No	0.90	0.78	0.84	342
Yes	0.81	0.92	0.86	342
accuracy			0.85	684
macro avg	0.85	0.85	0.85	684
weighted avg	0.85	0.85	0.85	684

Figure 6.16: Performance metrics for Stacking Ensemble Model

### 6.3.3 Voting Ensemble Model Training

In this phase, we implemented a Voting Ensemble Model that combines the predictions of the tuned Random Forest and XGBoost classifiers. A soft voting strategy was used, meaning the ensemble model averages the predicted probabilities from both models and selects the class with the highest combined probability.

The ensemble was built using the `VotingClassifier` from `Scikit-learn`, incorporating both base models that were individually optimised through hyperparameter tuning. By leveraging the strengths of Random Forest's bagging approach and XGBoost's boosting mechanism, the voting classifier aimed to improve overall predictive accuracy and robustness compared to using each model independently.

```

from sklearn.ensemble import VotingClassifier

# Build the voting ensemble
ensemble_model = VotingClassifier(
    estimators=[('rf', final_model), ('xgb', xgb_model)],
    voting='soft'
)

# Fit the ensemble model
ensemble_model.fit(X_train_sm, y_train_sm)

```

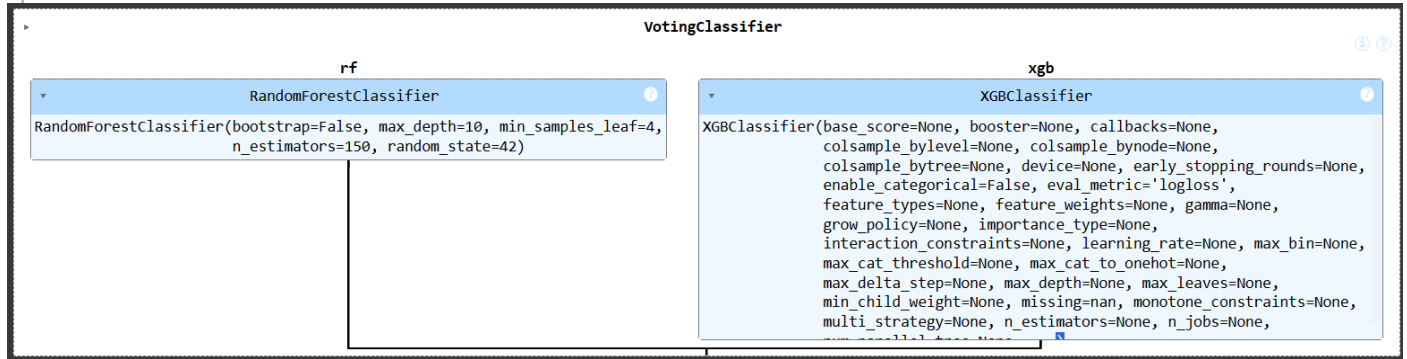


Figure 6.17: Voting Classifier

This approach enabled us to capture the different patterns identified by each algorithm, thereby enhancing generalisation and potentially reducing model bias and variance.

```
Confusion Matrix:
[[270  72]
 [ 30 312]]
Accuracy: 0.8508771929824561
Precision: 0.8125
Recall: 0.9122807017543859
F1 Score: 0.859504132231405
AUC Score: 0.9013115146540815

Classification Report:
              precision    recall  f1-score   support

     No         0.90         0.79         0.84         342
     Yes         0.81         0.91         0.86         342

 accuracy          0.85         0.85         0.85         684
 macro avg         0.86         0.85         0.85         684
 weighted avg      0.86         0.85         0.85         684
```

Figure 6.18: Performance metrics for Voting Ensemble Model

### 6.3.4 Model Comparison: Stacking Ensemble vs Voting Ensemble

The stacking ensemble achieved an accuracy of 84.8%, while the voting ensemble slightly outperformed it with an accuracy of 85.1%. Both models produced very similar results, with the voting ensemble having a marginally higher F1 score (0.860) compared to the stacking model (0.858). The AUC score also favoured the voting ensemble (0.901) over stacking (0.896), indicating a slightly better ability to discriminate between approved and rejected loans.

Metric	Stacking Ensemble	Voting Ensemble
Accuracy	0.848	0.851
Precision	0.805	0.813
Recall	0.918	0.912
F1 Score	0.858	0.860
AUC Score	0.896	0.901

Table 6.6: Model comparison between Stacking and Voting Ensemble

The differences in performance stem from their fundamental approaches. Stacking combines model predictions by training a meta-learner (in this case, Logistic Regression) on the outputs of base learners. While this can capture complex patterns, it introduces an additional layer of modelling that can sometimes lead to slight overfitting or less generalisation, especially when the base learners are already strong and complementary.

On the other hand, the voting ensemble uses soft voting to average probabilities from both tuned base models (Random Forest and XGBoost). This simpler approach avoids overfitting and benefits from the combined strengths of both algorithms, leading to slightly higher accuracy and AUC. In scenarios where base learners already perform well individually, soft voting often provides a more stable and reliable improvement over stacking.

While both ensembles delivered competitive results, the voting ensemble model showed a slight edge in overall performance metrics, likely due to its simpler probability averaging approach that efficiently balances predictions from strong base learners without introducing additional modelling complexity.

### **6.3.5 Review**

While stacking introduced an additional learning layer, it did not outperform the simpler voting ensemble, suggesting that the tuned Random Forest and XGBoost models already captured most of the predictive patterns in the data. This highlighted the importance of model simplicity and interpretability over unnecessary complexity.



## 6.4 Final Model Comparison and Recommendation

Based on the final model comparison, it is evident that the Random Forest model delivers the strongest overall performance across most evaluation metrics.

Model	Accuracy	Precision	Recall	F1 Score	AUC Score
Random Forest	0.8596	0.8030	0.9532	0.8717	0.9031
XGBoost	0.8567	0.8081	0.9357	0.8672	0.9016
Stacking Ensemble	0.8479	0.8051	0.9181	0.8579	0.8965
Voting Ensemble	0.8509	0.8125	0.9123	0.8595	0.9013

*Table 6.7: Final Model Comparison*

With an accuracy of 85.96%, an F1 score of 87.17%, and the highest recall (95.32%) and AUC score (90.31%), it outperforms XGBoost, stacking, and voting ensembles.

While XGBoost and the ensemble models achieved competitive results, their marginally lower F1 scores and recall values indicate that they are less effective at correctly identifying approved loans, which is critical in minimising false negatives for loan approval automation.

From the ensemble modelling phase, we learned that combining multiple strong classifiers does not always guarantee significant performance gains. Furthermore, the Random Forest model balances predictive power with interpretability, making it more practical for real-world deployment. Therefore, the Random Forest model is recommended as the final model for this project, as it best meets the objectives of enhancing efficiency, accuracy, and fairness in credit decision-making.

## 6.5 Explainable AI Interpretation of the Random Forest Model using LIME

This code snippet demonstrates how LIME (Local Interpretable Model-agnostic Explanations) was applied to the trained Random Forest model to enhance interpretability.

```
# Get feature names after dropping 'Loan_Status'
feature_names = df.drop('Loan_Status', axis=1).columns.tolist()

explainer = LimeTabularExplainer(
    training_data=X_train_sm,
    feature_names=feature_names,
    class_names=['No', 'Yes'],
    mode='classification',
    discretize_continuous=True
)
```

First, feature names were retrieved by dropping the target variable (**Loan\_Status**) to ensure that only predictor variables were included in the explanation. The **LimeTabularExplainer** was then initialised using the balanced training dataset (**X\_train\_sm**), feature names, class names (“No” and “Yes”), and the classification mode. The parameter **discretize\_continuous=True** allows LIME to discretise continuous variables into intervals, improving interpretability for tabular data.

```
#Choose an instance to explain
i = 4 # for example
instance = X_train_sm[i]
```

An individual instance from the training set (index 4 in this example) was selected for explanation. LIME generated a local explanation by creating perturbations of this instance and observing how the Random Forest model predicted outcomes. The **predict\_proba** function was used so that LIME could work with class probabilities. Finally, the explanation was displayed in the notebook, showing the top 10 features influencing the model's decision for that specific prediction.

```
explanation = explainer.explain_instance(
    data_row=instance,
    predict_fn=final_model.predict_proba,
    num_features=10
)
explanation.show_in_notebook()
```

This LIME output explains why the Random Forest model predicted “No” for this specific loan application.

On the left, the prediction probabilities indicate a 96% likelihood of the loan being rejected (“No”) and a 4% chance of approval (“Yes”). The middle section shows the contributions of individual features to this decision.

Features pushing towards rejection (“No”) are shown in blue. The most influential negative factors include a low credit history ( $\leq 0.72$ ), zero dependents, low co-applicant income, and a low loan amount ( $\leq 103$ ). These features strongly decrease the probability of loan approval.

Features slightly supporting approval (“Yes”) are shown in orange. Factors such as being married, a longer loan term (360 months), an applicant income of 4,166, and a property area value of \$ 1 provided some positive contributions toward approval, but were not strong enough to outweigh the negative factors.

The right side shows the exact feature values for this applicant. Overall, the model’s decision is primarily driven by the applicant’s poor credit history and low financial indicators, which significantly reduce the approval likelihood.

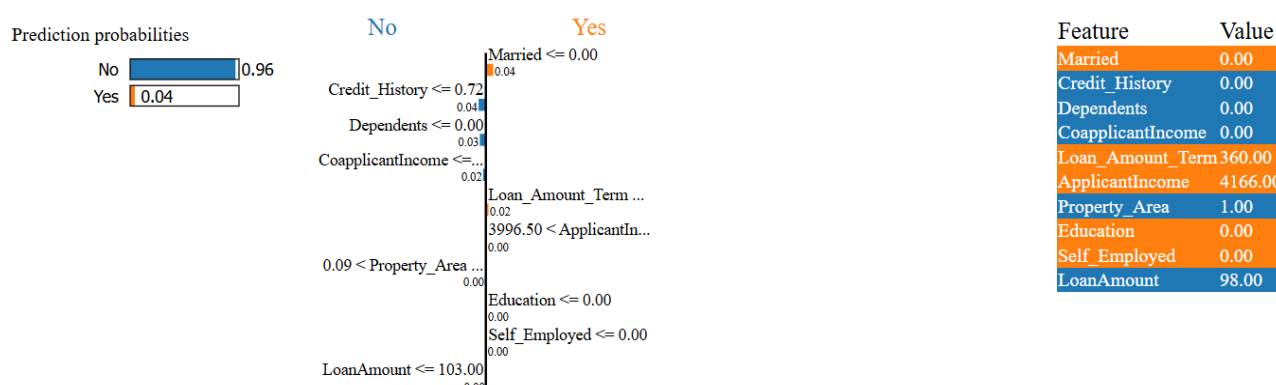


Figure 6.19: LIME interpretation of a “No” prediction from the Random Forest Model

In Figure 6.20, the model predicted loan approval (“Yes”) with 90% probability, compared to only 10% for “No.” Features highlighted in orange pushed the decision towards approval, while those in blue pushed it towards rejection. In this case, having three dependents and a positive credit history were the strongest factors favouring approval. On the other hand, features such as being married, having a loan amount of 130, and being self-employed introduced opposing influence, leaning slightly towards rejection. However, the positive contributions outweighed the negatives, leading to a strong overall prediction of loan approval.

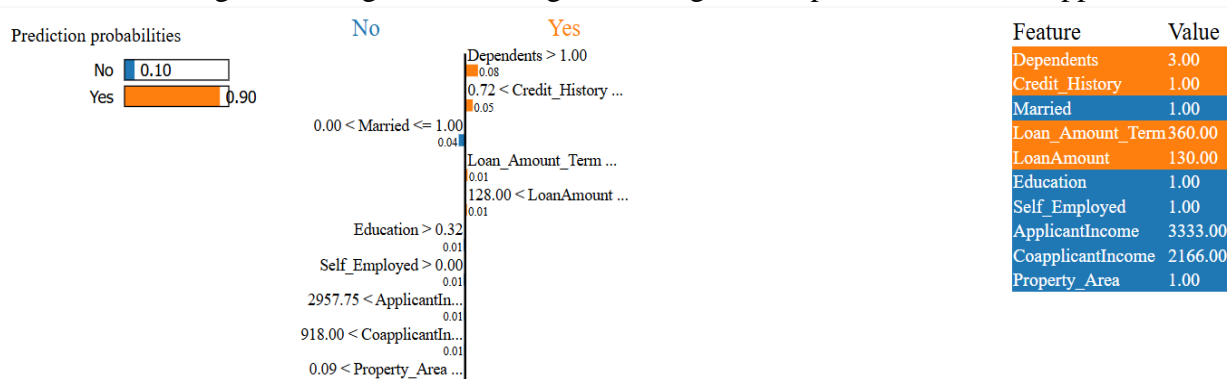


Figure 6.20: LIME interpretation of a “Yes” prediction from the Random Forest Model

# 7 Evaluation

## 7.1 Functional Requirements

ID	Description	Priority	Satisfaction	Explanation
1	Obtain the dataset from <a href="#">(Nazim et al., 2023)</a>	Must Have	Satisfied	We were able to obtain the dataset used in this paper.
2	Data exploration and pre-processing	Must Have	Satisfied	The dataset was pre-processed using different tools and approaches. This was done to ensure the dataset is ready for modelling.
3	Build the model using Random Forest, XGBoost and Ensemble methods.	Must Have	Satisfied	The models were initiated and trained on the balanced dataset.
4	Hyperparameter tuning	Must Have	Satisfied	Hyperparameter tuning was applied to these models using GridSearchCV.
5	Evaluate the models using accuracy, precision, recall, and F1-score	Must Have	Satisfied	Each of the models was evaluated based on these performance metrics.
6	Model comparison	Must Have	Satisfied	The models were compared based on different metrics, and the Random Forest model outperformed the XGBoost and Ensemble models.
7	Implement Explainable AI tools	Should Have	Satisfied	LIME was used to enhance the interpretability of the best-performing model.
8	Model Deployment	Won't Have	Unsatisfied	Due to time limitations, this feature was omitted.

9	User App	Won't Have	Unsatisfied	This feature is for future work.
---	----------	------------	-------------	----------------------------------

## 7.2 Non-functional Requirements

ID	Description	Priority	Satisfaction	Explanation
1	Anonymising data as part of research (for GDPR reasons)	Must Have	Satisfied	The dataset used in this project was already anonymised.

## 7.3 Limitations

### Data Limitations

The dataset used had missing values, and the target variable had imbalanced classes, which were handled using imputation and SMOTE, respectively. However, these techniques may introduce bias or synthetic patterns that do not fully reflect real-world distributions.

### Limited Feature Scope

The dataset only included a limited number of features. Key variables such as credit score, employment length, or detailed financial history were not available, which could improve model performance and fairness.

### Limited Hyperparameter Tuning

Hyperparameter tuning was conducted using grid search, which is computationally expensive and restricted to predefined ranges. More advanced methods like randomised search or Bayesian optimisation could yield better results, but were not explored due to time and resource constraints.

## 8 Conclusion and Future Work

### 8.1 Summary

The approval of loan applications is a critical process in the financial industry, where accurate, fair, and explainable decision-making is essential. This project aimed to build a reliable machine learning-based solution for predicting loan approval outcomes, incorporating model evaluation and interpretability techniques to enhance transparency and performance. The project addressed common challenges in credit decision-making, such as class imbalance, feature selection, and the need for interpretability in automated systems.

The objectives were achieved through a structured workflow that included data pre-processing, exploratory analysis, class balancing with SMOTE, feature engineering, model development, and evaluation. Several machine learning models were trained and compared, including Random Forest, XGBoost, and ensemble methods like stacking and voting classifiers. Additionally, hyperparameter tuning using GridSearchCV was conducted to optimise model performance. To further enhance transparency, LIME was used to explain individual predictions made by the models.

Overall, the Random Forest model with all features and proper tuning achieved the best balance of precision, recall, and F1-score, making it the recommended model for deployment. The project also demonstrated the importance of explainable AI in sensitive domains like finance, where understanding the factors driving a model's decisions is crucial for trust and accountability.

### 8.2 Lessons Learnt

Throughout this project, we gained valuable technical and personal skills. We significantly improved our understanding of the end-to-end machine learning pipeline, from data collection and cleaning to feature engineering, model training, and evaluation.

We became confident working with ensemble methods, such as Random Forest and XGBoost, and we deepened our knowledge of model performance metrics and hyperparameter tuning using GridSearchCV.

One of the most important skills we developed was learning how to apply explainable AI techniques, particularly using LIME (Local Interpretable Model-Agnostic Explanations). This allowed us to interpret model predictions on an individual level and understand the features driving the decision, which is crucial in high-stakes applications such as loan approvals.

Beyond the technical aspects, we also learnt the importance of time management and iterative experimentation. Managing the various stages of the project while meeting academic deadlines helped us become more organised and methodical in our workflow. We also encountered and learned to overcome challenges, such as class imbalance and feature selection trade-offs, which strengthened our problem-solving skills.

## 8.3 Future Work

For future work, it would be beneficial to explore advanced deep learning techniques, such as neural networks or transformer-based models, to potentially improve classification performance. If the project timeline were extended, more time could be allocated to fine-tuning hyperparameters for each model and comparing performance across additional ensemble approaches.

Experimenting with more diverse base learners or advanced stacking methods, such as gradient blending or neural meta-learners, could potentially enhance performance further by leveraging the strengths of different algorithms more effectively. Integrating more detailed financial features, such as credit history, banking transactions, or macroeconomic indicators, could also strengthen the predictive power of the models.

Additionally, explainability could be expanded by incorporating SHAP in addition to LIME, allowing for both local and global interpretation of model decisions. Finally, deploying the model in a real-time environment, such as a web-based loan screening tool or decision support dashboard, would demonstrate its practical application and support informed credit decisions.

## Bibliography

- Addy, W. A., Ajayi-Nifise, A. O., Bello, B. G., Tula, S. T., Odeyemi, O., & Falaiye, T. (2024). AI in credit scoring: A comprehensive review of models and predictive analytics. *Global Journal of Engineering and Technology Advances*, 18(2), 118-129. 10.30574/gjeta.2024.18.2.0029
- Al-Baik, O., & Miller, J. (2014, November 11). The kanban approach, between agility and leanness: a systematic review. *Empirical Software Engineering*, 1861–1897. 10.1007/s10664-014-9340-x
- Andhale, A., Bansode, O., Chavan, R., & Bagade, J. (2024). Bank Loan Prediction System By Using Machine Learning. *2024 International Conference on Intelligent Systems and Advanced Applications (ICISAA)*, 1-5. 10.1109/ICISAA62385.2024.10829100}
- Armeshi, S., Begel, A., Bird, C., DeLine, R., Gall, H., Kamar, E., Nagappan, N., Nushi, B., & Zimmerman, T. (2019). Software Engineering for Machine Learning: A Case Study. *2019 IEEE/ACM 41st International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)*, 291-300. 10.1109/ICSE-SEIP.2019.00042
- Bordio. (2025, January 1). *Kanban Methodology. What Is Kanban?* Bordio. Retrieved June 27, 2025, from <https://bordio.com/blog/kanban-methodology/>
- Chatterjee, D. (2021). *Loan Prediction Problem Dataset*. Kaggle. Retrieved June 1, 2025, from <https://www.kaggle.com/datasets/altruistdelhite04/loan-prediction-problem-dataset>
- Chunyu, Y. (2024). Research on loan approval and credit risk based on the comparison of Machine learning models. *SHS Web of Conferences*, 181. 10.1051/shsconf/202418102003
- Doshi-Velez, F., & Kim, B. (2017, March 2). Towards A Rigorous Science of Interpretable Machine Learning.
- Gothai, E., Rajalaxmi, R.R., Thamilselvan, R., & K, S. (2024). Enhanced Loan Approval Prediction System using Ensemble Machine Learning Techniques. *2024 3rd International Conference on Automation, Computing and Renewable Systems (ICACRS)*, 1182-1187. 10.1109/ICACRS62842.2024.10841604}
- IBM Corporation. (2021, August 17). *CRISP-DM Help Overview*. IBM SPSS Modeler. [https://www.ibm.com/docs/en/spss-modeler/saas?topic=dm-crisp-help-overview&mhsrc=ibmsearch\\_a&mhq=crisp%20dm](https://www.ibm.com/docs/en/spss-modeler/saas?topic=dm-crisp-help-overview&mhsrc=ibmsearch_a&mhq=crisp%20dm)



- Lohani, B. P., Trivedi, M., Singh, R. J., Bibhu, V., Ranjan, S., & Kushwaha, P. K. (2022). Machine Learning Based Model for Prediction of Loan Approval. *2022 3rd International Conference on Intelligent Engineering and Management (ICIEM)*, 465-470. 10.1109/ICIEM54221.2022.9853160}
- Malarvizhi, N., Reddy, Y. S., Yaraswi, Y. M., & Harshitha, K. (2024, June 15). Predictive Finance Decision Support System using Machine Learning. *Grenze International Journal of Engineering & Technology (GIJET)*, 10(2, Part 4), 5382–5386. <https://research.ebsco.com/linkprocessor/plink?id=3ffadbc7-9d25-36ec-a181-aed0841e31c7>
- Mitchell, T. M. (1997). *Machine Learning*. McGraw-Hill Education.
- Nallakaruppan, M. K., Chaturvedi, H., Grover, V., Balusamy, B., Jaraut, P., Bahadur, J., Meena, V. P., & Hameed, I. A. (2024). Credit Risk Assessment and Financial Decision Support Using Explainable Artificial Intelligence. *Risks*, 12(10). 10.3390/risks12100164
- Nancy Deborah, R., Alwyn Rajiv, S., Vinora, A., Manjula Devi, C., Mohammed Arif, S., & Mohammed Arif, G. S. (2023). An Efficient Loan Approval Status Prediction Using Machine Learning. *2023 International Conference on Advanced Computing Technologies and Applications (ICACTA)*, 1-6. 10.1109/ICACTA58201.2023.10392691
- Naveen Kumar, C., Keerthana, D., Kavitha, M., & Kalyani, M. (2022). Customer Loan Eligibility Prediction using Machine Learning Algorithms in Banking Sector. *2022 7th International Conference on Communication and Electronics Systems (ICCES)*, 1007-1012. 10.1109/ICCES54183.2022.9835725
- Nazim, U., Uddin, A., Ashraf, U., Manwarul, I., Alamin, T., & Sunil, A. (2023). An ensemble machine learning based bank loan approval predictions system with a smart application. *International Journal of Cognitive Computing in Engineering*, 4(2666-3074), 327-339. <https://doi.org/10.1016/j.ijcce.2023.09.001>
- Nureni, A. A., & Adekola, O. E. (2022). Loan approval prediction based on machine learning approach. *FUDMA Journal of Sciences*, 6(3). <https://doi.org/10.33003/fjs-2022-0603-830>
- Pillai, V. (2024, July). Enhancing Transparency and Understanding in AI Decision-Making Processes. *IRE Journals*, 8(1), 6. [https://www.researchgate.net/profile/Vinayak\\_Pillai/publication/384884277\\_Enhancing\\_Transparency\\_an](https://www.researchgate.net/profile/Vinayak_Pillai/publication/384884277_Enhancing_Transparency_an)

d\_Understanding\_in\_AI\_Decision-Making\_Processes/links/670c146bdc91726ad38ff9eb/Enhancing-  
Transparency-and-Understanding-in-AI-Decision-Making-Processes.pdf

Rao, M. G., Bhat, V., Bhat, V., Bhardawaj, Y., & Kumar, V. (2024). Loan Insight Analyzer: Transforming Loan Analysis with Machine Learning and Deep Learning. *2024 5th IEEE Global Conference for Advancement in Technology (GCAT)*, 1-6. 10.1109/GCAT62922.2024.10924038

Sinap, V. (2024). A Comparative Study of Loan Approval Prediction Using Machine Learning Methods. *Gazi University Journal of Science*, 12(2), 644-663. 10.29109/gujsc.1455978

Sommerville, I. (2015). *Software Engineering*. Pearson.

Srivastava, S. (2024). Automated Loan Approval System using Machine Learning. *2024 4th International Conference on Ubiquitous Computing and Intelligent Information Systems (ICUIS)*, 557-562.

<https://ieeexplore.ieee.org/abstract/document/10866059>

Stray, V., Stol, K.-J., Paasivaara, M., & Kruchten, P. (Eds.). (2022). *Agile Processes in Software Engineering and Extreme Programming: 23rd International Conference on Agile Software Development, XP 2022, Copenhagen, Denmark, June 13–17, 2022, Proceedings*. Springer International Publishing.  
file:///C:/Users/user/Downloads/978-3-031-08169-9.pdf

## Appendix I

# Certificate of Ethics Review

**Project title:**

Machine Learning-Based Loan Approval Automation: Enhancing Efficiency, Accuracy, and Fairness  
in Credit Decision-Making

<b>Name</b> :	Hellen Otieno	<b>User ID:</b>	up22804 93	<b>Application date:</b>	29/05/2 025 10:03:5 0	<b>ER Number:</b>	TETHIC-2025- 111110
------------------	---------------	-----------------	---------------	------------------------------	--------------------------------	-----------------------	------------------------

You must download your referral certificate, print a copy and keep it as a record of this review.

The FEC representative(s) for the **School of Computing** is/are **Elisavet Andrikopoulou, Kirsten Smith**

It is your responsibility to follow the University Code of Practice on Ethical Standards and any Department/School or professional guidelines in the conduct of your study including relevant guidelines regarding health and safety of researchers including the following:

- [University Policy](#)
- [Safety on Geological Fieldwork](#)

It is also your responsibility to follow University guidance on Data Protection Policy:

- [General guidance for all data protection issues](#)
- [University Data Protection Policy](#)

Which school/department do you belong to?: **School of Computing**

What is your primary role at the University?: **Postgraduate Student**

What is the name of the member of staff who is responsible for supervising your project?: **Dr. Mani**

**Ghahremani** Is the study likely to involve human subjects (observation) or participants?: No

Will financial inducements (other than reasonable expenses and compensation for time) be offered to participants?: No

Are there risks of significant damage to physical and/or ecological environmental features?: No

Are there risks of significant damage to features of historical or cultural heritage (e.g. impacts of study techniques, taking of samples)?: No

Does the project involve animals in any way?: No

Could the research outputs potentially be harmful to third parties?: No

Could your research/artefact be adapted and be misused?: No

Will your project or project deliverables be relevant to defence, the military, police or other security organisations and/or in addition, could it be used by others to threaten UK security?: No

Please read and confirm that you agree with the following statements: I confirm that I have considered the implications for data collection and use, taking into consideration legal requirements (UK GDPR, Data Protection Act 2018 etc.), I confirm that I have considered the impact of this work and and taken any reasonable action to mitigate potential misuse of the project outputs, I confirm that I will act ethically and honestly throughout this project

## Supervisor Review

As supervisor, I will ensure that this work will be conducted in an ethical manner in line with the University Ethics Policy. Supervisor comments:

Supervisor's Digital Signature: **mani.ghahremani@port.ac.uk** Date: **02/06/2025**