LOAN APPROVAL PREDICTION USING

MACHINE LEARNING

***A Project Report submitted in the partial fulfillment of the Requirements for the award of the degree***

## BACHELOR OF TECHNOLOGY

**in**

## COMPUTER SCIENCE AND ENGINEERING

**Submitted by**

## B. Venkata Siva (20471A05K0)

**SK. Aariz Ahmed (21475A0516)**

**Under the esteemed guidance of Dr. Rama Krishna Eluri, M.Tech., Ph.D.,**

**Assistant Professor**



## DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

NARASARAOPETA ENGINEERING COLLEGE: NARASARAOPET

**(AUTONOMOUS)**

**Accredited by NAAC with A+ Grade and NBA under Tier -1 Approved by AICTE, New Delhi, Permanently Affiliated to JNTUK, Kakinada**

**KOTAPPAKONDA ROAD, YALAMANDA VILLAGE, NARASARAOPET- 522601**

**2023-2024**

## NARASARAOPETA ENGINEERING COLLEGE: NARASARAOPET

**(AUTONOMOUS)**

## DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING



**CERTIFICATE**

### This is to certify that the project that is entitled with the name “Loan Approval Prediction Using Machine Learning” is a bonafide work done by the team B. Venkata Siva (20471A05K0), Sk. Aariz Ahmed (21475A0516) in partial fulfillment of the requirements for the award of the degree of BACHELOR OF TECHNOLOGY in the Department of COMPUTER SCIENCE AND ENGINEERING during 2023-2024.

PROJECT GUIDE PROJECT CO-ORDINATOR

**Dr. Rama Krishna Eluri, M.Tech., Ph.D. Dr. M. Sireesha, M.Tech., Ph.D.**

### Assistant Professor Associate Professor

HEAD OFTHE DEPARTMENT EXTERNAL EXAMINER

**Dr. S. N. Tirumala Rao, M.Tech., Ph.D.,**

**Professor**

**DECLARATION**

We declare that this project work titled “LOAN APPROVAL PREDICTION USING MACHINE LEARNING” is composed by ourselves that the work contain here is our own except where explicitly stated otherwise in the text and that this work has been submitted for any other degree or professional qualification except as specified.

B. Venkata Siva (20471A05K0)

Sk. Aariz Ahmed (21475A0516)

# ACKNOWLEDGEMENT

We wish to express my thanks to various personalities who are responsible for the completion of the project. We are extremely thankful to our beloved chairman sri **M.V. Koteswara Rao, B.Sc.,** who took keen interest in us in every effort throughout this course. We owe out sincere gratitude to our beloved principal **Dr. M. Sreenivasa Kumar, M.Tech., Ph.D., MISTE., FIE(I).,** for showing his kind attention and valuable guidance throughout the course.

We express our deep felt gratitude towards **Dr. S. N. Tirumala Rao, M.Tech., Ph.D.,** HoD of CSE department and also to our guide **Dr. Rama Krishna Eluri, M.Tech., Ph.D.,** Assistant Professor of CSE department whose valuable guidance and unstinting encouragement enable us to accomplish our project successfully in time.

We extend our sincere thanks towards **Dr. M. Sireesha, M.Tech., Ph.D.** Associate professor& Project coordinator of the project for extending her encouragement. Their profound knowledge and willingness have been a constant source of inspiration for us throughout this project work.

We extend our sincere thanks to all other teaching and non-teaching staff to department for their cooperation and encouragement during our B. Tech degree.

We have no words to acknowledge the warm affection, constant inspiration and encouragement that we received from our parents.

We affectionately acknowledge the encouragement received from our friends and those who involved in giving valuable suggestions had clarifying out doubts which had really helped us in successfully completing our project.

### By

**B. Venkata Siva (20471A05K0)**

**SK. Aariz Ahmed (21475A0516)**

### 



**INSTITUTE VISION AND MISSION**

### INSTITUTION VISION

To emerge as a Centre of excellence in technical education with a blend of effective student centric teaching learning practices as well as research for the transformation of lives and community,

### INSTITUTION MISSION

M1: Provide the best class infra-structure to explore the field of engineering and research

M2: Build a passionate and a determined team of faculty with student centric teaching, imbibing experiential, innovative skills

M3: Imbibe lifelong learning skills, entrepreneurial skills and ethical values in students for addressing societal problems



# DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

### VISION OF THE DEPARTMENT

To become a center of excellence in nurturing the quality Computer Science & Engineering professionals embedded with software knowledge, aptitude for research and ethical values to cater to the needs of industry and society.

### MISSION OF THE DEPARTMENT

The department of Computer Science and Engineering is committed to

**M1:** Mould the students to become Software Professionals, Researchers and Entrepreneurs by providing advanced laboratories.

**M2:** Impart high quality professional training to get expertise in modern software tools and technologies to cater to the real time requirements of the Industry.

**M3:** Inculcate team work and lifelong learning among students with a sense of societal and ethical responsibilities.



# Program Specific Outcomes (PSO’s)

**PSO1:** Apply mathematical and scientific skills in numerous areas of Computer Science and Engineering to design and develop software-based systems.

**PSO2:** Acquaint module knowledge on emerging trends of the modern era in Computer Science and Engineering

**PSO3:** Promote novel applications that meet the needs of entrepreneur, environmental and social issues.



## Program Educational Objectives (PEO’s)

The graduates of the programme are able to:

**PEO1:** Apply the knowledge of Mathematics, Science and Engineering fundamentals to identify and solve Computer Science and Engineering problems.

**PEO2:** Use various software tools and technologies to solve problems related to academia, industry and society.

**PEO3:** Work with ethical and moral values in the multi-disciplinary teams and can communicate effectively among team members with continuous learning.

**PEO4:** Pursue higher studies and develop their career in software industry.



# Program Outcomes

1. **Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **Problem analysis:** Identify, formulate, research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
3. **Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
4. **Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
5. **Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
6. **The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
7. **Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. **Individual and team work**: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

**10. Communication**: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

**11. Project management and finance**: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one’s own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

**12. Life-long learning**: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.



**Project Course Outcomes (CO’S):**

**CO421.1:** Analyze the System of Examinations and identify the problem.

**CO421.2:** Identify and classify the requirements. **CO421.3:** Review the Related Literature **CO421.4:** Design and Modularize the project

**CO421.5:** Construct, Integrate, Test and Implement the Project.

**CO421.6:** Prepare the project Documentation and present the Report using appropriate method.

## Course Outcomes – Program Outcomes mapping

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **PO1** | **PO2** | **PO3** | **PO4** | **PO5** | **PO6** | **PO7** | **PO8** | **PO9** | **PO10** | **PO11** | **PO12** | **PSO1** | **PSO2** | **PSO3** |
| **C421.1** |  | ✓ |  |  |  |  |  |  |  |  |  |  | ✓ |  |  |
| **C421.2** | ✓ |  | ✓ |  | ✓ |  |  |  |  |  |  |  | ✓ |  |  |
| **C421.3** |  |  |  | ✓ |  | ✓ | ✓ | ✓ |  |  |  |  | ✓ |  |  |
| **C421.4** |  |  | ✓ |  |  | ✓ | ✓ | ✓ |  |  |  |  | ✓ | ✓ |  |
| **C421.5** |  |  |  |  | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| **C421.6** |  |  |  |  |  |  |  |  | ✓ | ✓ | ✓ |  | ✓ | ✓ |  |

**Course Outcomes – Program Outcome correlation**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **PO1** | **PO2** | **PO3** | **PO4** | **PO5** | **PO6** | **PO7** | **PO8** | **PO9** | **PO10** | **PO11** | **PO12** | **PSO1** | **PSO2** | **PSO3** |
| **C421.1** | 2 | 3 |  |  |  |  |  |  |  |  |  |  | 2 |  |  |
| **C421.2** |  |  | 2 |  | 3 |  |  |  |  |  |  |  | 2 |  |  |
| **C421.3** |  |  |  | 2 |  | 2 | 3 | 3 |  |  |  |  | 2 |  |  |
| **C421.4** |  |  | 2 |  |  | 1 | 1 | 2 |  |  |  |  | 3 | 2 |  |
| **C421.5** |  |  |  |  | 3 | 3 | 3 | 2 | 3 | 2 | 2 | 1 | 3 | 2 | 1 |
| **C421.6** |  |  |  |  |  |  |  |  | 3 | 2 | 1 |  | 2 | 3 |  |

## Note: The values in the above table represent the level of correlation between CO’s and PO’s:

* 1. **Low level**

## Medium level

* 1. **High level**

## Project mapping with various courses of Curriculum with Attained PO’s:

|  |  |  |
| --- | --- | --- |
| **Name of the course from which principles are applied in this project** | **Description of the device** | **Attained PO** |
| C2204.2, C22L3.2 | Gathering the requirements and defining the problem, plan to develop a Loan Approval Prediction System. | PO1, PO3 |
| CC421.1, C2204.3, C22L3.2 | Each and every requirement is critically analyzed, the process model is identified and divided into  4 modules – preprocessing, training, evaluation, Loan Approval Prediction. | PO2, PO3 |
| CC421.2, C2204.2, C22L3.3 | Logical design is done by using the unified modelling language which involves individual team work. | PO3, PO5, PO9 |
| CC421.3, C2204.3, C22L3.2 | Each and every module is tested, integrated, and evaluated in our project. | PO1, PO5 |
| CC421.5, C2204.2, C22L3.3 | Documentation is done by all our two members in the form of a group. | PO10 |
| CC421.5, C2204.2, C22L3.3 | Each and every phase of the work in group is presented periodically. | PO10, PO11 |
| C2202.2, C2203.3, C1206.3, C3204.3, C4110.2 | Implementation is done and the project will be handled by the transport department. | PO4, PO7 |
| C32SC4.3 | The physical design includes a website to predict the Loan Approval from the given data. | PO5, PO6 |

**ABSTRACT**

Loans are a crucial part of the modern world, and banks receive a significant portion of their profits from them. However, deciding whether to grant a loan to an applicant is a complex process that requires banks to consider many factors.

In this study, we suggest a machine learning-based method to streamline the loan acceptance prediction process. To determine whether or not a loan applicant's profile is relevant for approval, we employ effective machine learning algorithms. We base our predictions on important features. Additionally, we present a comparison study of various categorization methods to demonstrate how machine learning algorithms might enhance the loan approval procedure. Our results show that machine learning algorithms can significantly reduce the risk of loan defaults and improve the loan approval process. Moreover, to enhance prediction accuracy, we incorporate a voting ensemble technique into our methodology. This additional layer of analysis further refines our predictions, contributing to more reliable loan approval decisions.

Furthermore, among all the models, the top three models are Random Forest, Gradient Boost, and Extra Trees. We applied a voting ensemble with these three models and achieved an accuracy of 95.56, where the existing accuracy is 87.26%. This significant improvement underscores the effectiveness of ensemble learning techniques in enhancing prediction accuracy and reliability. By leveraging the collective insights of the top-performing models, our approach ensures more robust and accurate loan approval decisions, ultimately benefiting both financial institutions and loan applicants in the modern financial landscape.

# INDEX

|  |  |  |
| --- | --- | --- |
| **S.NO.** | **CONTENT** | **PAGE NO** |
| 1. | [**Introduction**](#INTRODUCTION) | 01 |
|  | 1.1 [Introduction](#INTRODUCTION) | 03 |
|  | 1.2 [Existing System](#EXISTING_SYSTEM) | 03 |
|  | 1.3 [Proposed System](#PROPOSED_SYSTEM) | 03 |
|  | 1.4 [System Requirements](#SYSTEM_REQUIREMENTS) | 04 |
|  | 1.4.1 [Hardware Requirements](#SYSTEM_REQUIREMENTS) | 04 |
|  | 1.4.2 [Software Requirements](#SOFTWARE_REQUIREMENTS) | 05 |
| 2. | [**Literature**](#LITERATURE_SURVEY) | 06 |
|  | 2.1 [Literature Survey](#LITERATURE_SURVEY) | 06 |
|  | 2.2 [Some Machine Learning Methods](#MACHINELEARNING_METHODS) | 08 |
|  | 2.3 [Applications of Machine Learning](#APPLICATIONS) | 09 |
|  | 2.4 [Machine Learning Models](#Models) | 10 |
| 3. | [**System Analysis**](#SYSTEM_ANALYSIS) | 13 |
|  | 3.1 [Importance of Machine Learning Using Python](#SYSTEM_ANALYSIS) | 13 |
|  | 3.2 [Implementation of Machine Learning Using Python](#USING_PYTHON) | 13 |
|  | 3.3 [Scope of the project](#SCOPE) | 16 |
|  | 3.4 [Data Set Analysis](#ANALYSIS) | 16 |
|  | 3.5 [Data Pre-Processing](#Preprocessing) | 17 |
|  | 3.6 [Classification](#Classification) | 21 |
| 4. | [**Implementation of code**](#IMPLEMENTATION) | 23 |
| 5. | [**Result Analysis**](#RESULT) | 49 |
| 6. | [**Output Screens**](#SCREENS) | 53 |
| 7. | [**Conclusion**](#CONCLUSION) | 57 |
| 8. | [**Future Scope**](#FUTURE) | 58 |
| 9. | [**References**](#REFERENCES) | 59 |

**LIST OF FIGURES**

**S.NO. LIST OF FIGURES PAGE NO**

1. [Fig 1.3.1 Flow chart](#flow) 04
2. [Fig 2.2.1 Types of Machine Learning](#fmeths) 08
3. [Fig 2.4.1 Ensemble Voting Model](#fvote) 12
4. [Fig 3.4.1 Dataset](#fdataset) 16
5. [Fig 3.4.2 Class Imbalance](#fimbalance) 17
6. [Fig 3.5.1 Frequency of Null Values in Dataset](#fbnull) 17
7. [Fig 3.5.2 Frequency of Null Values after Removal](#fanull) 18
8. [Fig 3.5.3 Columns After One-hot Encoding](#fhot) 18
9. [Fig 3.5.4 Visualizing Data Outliers](#fout) 19
10. [Fig 3.5.5 Correlation](#fcorr) 20
11. [Fig 3.5.6 Loan\_Status after SMOTE](#fsmote) 21
12. [Fig 5.1 Comparing Proposed and Existing Accuracies](#fcompare) 51
13. [Fig 5.2 Confusion Matrix of Voting Ensemble Model](#fconf) 52
14. [Fig 6.1 Home Page](#fhome) 53
15. [Fig 6.2 Form Page](#fform) 54
16. [Fig 6.3 Approval Status Page](#fyes) 55
17. [Fig 6.4 Disapproval Status Page](#fno) 56

**LIST OF TABLES**

**S.NO. LIST OF TABLES PAGE NO**

1. [Table 1 Metric Values for all Models](#table) 50

# INTRODUCTION

## INTRODUCTION

The lending function within the banking sector stands as a cornerstone in upholding a country's financial stability. Loans generate a significant portion of banks' interest revenue, thereby underscoring the criticality of a robust loan approval process. Yet, the prevalent manual methodologies often present challenges in terms of accuracy and efficiency. Individual bank managers assume the onus of assessing the risk associated with loan default and applicant eligibility. However, this approach may inadvertently trigger systemic disruptions, posing potential threats to the broader economy and subjecting banks to financial losses.

Historically, discerning creditworthy borrowers from a vast pool of applicants has proven to be a daunting task for banks [[1]](#REFERENCES). Effective prediction of loan defaults assumes paramount importance in contemporary banking systems, as inaccuracies in these projections could precipitate far-reaching financial crises.

Against this backdrop, this study endeavors to optimize the loan approval process through data-driven methodologies. Our approach commences with meticulous data preprocessing, entailing the elimination of null values, duplicates, and outliers, thus ensuring data cleanliness [[2]](#REFERENCES). Subsequently, we embark on a thorough exploration of inter-variable relationships, facilitating informed feature selection aimed at identifying the most salient attributes.

In the pursuit of optimizing the loan approval process through data-driven methodologies, our journey commences with a meticulous examination of data preprocessing techniques. Null values, notorious for their potential to skew analysis, are systematically addressed through strategic imputation or deletion, thereby ensuring the integrity of our dataset [[3]](#REFERENCES). Simultaneously, outlier removal, a crucial step in data cleaning, serves to mitigate the undue influence of extreme observations, thus fostering robust model performance.

With a pristine dataset at our disposal, our focus shifts towards uncovering intricate inter-variable relationships through correlation analysis [[4]](#r2). This enables us to discern patterns and dependencies among features, laying the foundation for informed feature selection. Furthermore, we leverage sqrt transformations to normalize skewed distributions [[5]](#r2), thereby enhancing the interpretability and predictive power of our models.

An inherent challenge in loan approval prediction tasks lies in the disparate class sizes between approved and rejected applications. To redress this imbalance, we employ the Synthetic Minority Over-sampling Technique (SMOTE) [[6]](#r2), a widely acclaimed approach that generates synthetic minority class instances, effectively rebalancing our dataset and mitigating the risk of biased model outcomes.

Armed with a balanced dataset and enriched with insights gleaned from preprocessing, we embark on the core phase of our study: leveraging a diverse repertoire of machine learning techniques [[7]](#r2). Gradient boosting, renowned for its ensemble learning prowess, offers nuanced insights into complex data patterns, while decision trees provide transparent, interpretable models ideal for capturing non-linear relationships [[8].](#r2) Complementing these, random forests excel in handling high-dimensional data and offer robust performance in the presence of noisy features.

Crucially, our approach extends beyond mere model implementation; rigorous performance evaluation forms the bedrock of our analysis [[9]](#r3). We subject each algorithm to a battery of diverse metrics, encompassing accuracy, precision, recall, F1 score, and area under the ROC curve (AUC-ROC), among others [[10]](#r3). This exhaustive evaluation framework affords a comprehensive understanding of each algorithm's efficacy, enabling us to make informed decisions regarding model selection and refinement [[11].](#r3)

Central to our study is the recognition of the imperative to enhance the precision and effectiveness of the loan approval procedure. Accordingly, we meticulously dissect the main shortcomings associated with conventional loan approval processes, thereby laying the groundwork for the implementation of data-driven enhancements.

## 1.2 EXISTING SYSTEM

* The existing system has embraced cutting-edge preprocessing techniques, ensuring that the data is cleansed of null values, duplicates, and outliers, laying a robust foundation for subsequent analysis.
* It represents a significant advancement from manual methods, enabling a more efficient and scalable approach to loan approval prediction.
* Implementation of a voting ensemble method enhances the predictive power of the system by aggregating predictions from multiple base models, thereby fostering more accurate outcomes.
* Despite these strengths, the system exhibits certain limitations, notably in terms of accuracy, where its performance may fall short of desired benchmarks.
* The reliance on a vast number of features for prediction introduces complexity and computational overhead, potentially impeding model interpretability and scalability.

## DISADVANTAGES

* Lack of outlier handling undermines data integrity.
* Utilizing all features increases complexity and computational burden.
* Low efficiency hampers system performance.
* Risk of false positives poses a significant concern.

## 1.3 PROPOSED SYSTEM

* The proposed system as depicted in [fig 1.3.1](#flow), entails comprehensive data preprocessing, including handling null values, outliers, and performing correlation analysis, as well as square root transformations and feature selection. Additionally, the Synthetic Minority Over-sampling Technique (SMOTE) is employed to address class imbalance.
* Multiple machine learning algorithms such as random forest, decision tree, K-nearest neighbors (KNN), gradient boosting, extra trees, and logistic regression are leveraged, with the implementation of a voting ensemble method to enhance predictive accuracy and robustness.

## Fig 1.3.1 Flow Chart

## ADVANTAGES

* + - Generates accurate and efficient results.
    - Reduced time and resources.
    - Using fewer features streamlines the analysis, reducing complexity and computational burden.
    - Cost effective.
    - Efficient for future usage.

## 1.4 SYSTEM REQUIREMENTS

### 1.4.1 HARDWARE REQUIREMENTS

* + - Processor: Intel Core i5
    - Cache Memory: 4MB
    - Hard Disk: 30GB or more
    - RAM: 1GB or more

### 1.4.2 SOFTWARE REQUIREMENTS

* Operating System: Windows 10
* Coding Language: Python
* Python Distribution: Anaconda, Flask
* Browser: Any Latest Browser Like Chrome

# 2 LITERATURE SURVEY

## LITERATURE SURVEY

Mahankali *et al.* [[12]](#r4) forecast loan approvals with an accuracy rate of 80.945% by using logistic regression. Their all-inclusive strategy comprises testing, model creation, and data pretreatment, offering a solid foundation for automated loan approval systems. This study provides useful information about the use of machine learning algorithms in banking settings and sets the standard for further research in the area.

Decision Trees and other machine learning models were used by Supriya *et al.* [[13]](#r4) to predict loan defaulters with an accuracy of 81.1%. Their research focused on data preparation methods, such as managing outliers and missing information, and resulted in a thorough examination of the characteristics that affect loan acceptance. The writers provided insightful information for improving credit risk assessment in the banking industry by highlighting the importance of variables like income level and credit history in loan sanctioning decisions.

Manjeet Kumar *et al.* [[14]](#r4) evaluated a number of classifiers, such as Light Gradient Boosting Machine (LGBM), Extra Trees, Random Forest, and Extreme Gradient Boosting (XGB) for the purpose of predicting bank loan default. Their study provides insightful information for financial institutions by highlighting the significance of debt income and work history in forecasting defaults. The comparative study of classifiers by Kumar et al. offers a thorough grasp of performance criteria including key metrics.

In the research, Mehul Madaan *et al*. [[15]](#r4) obtained 73% and 80% accuracy, respectively, in loan default prediction using Decision Trees and Random Forest algorithms. For financial organizations looking to enhance loan approval procedures and reduce credit risks, their study provides insightful information. Through the analysis of these algorithms' performance on a shared dataset, the research adds to the continuing investigation of machine learning applications in the banking industry.

When comparing machine learning algorithms for forecasting bank loan risks, Alsaleem *et al*. [[16]](#r4) discovered that Multilayer Perceptron has the best accuracy (80%). With an emphasis on useful applications and decision support systems, this study provides a baseline for comparable research in the field and offers insightful information on using neural networks for loan classification.

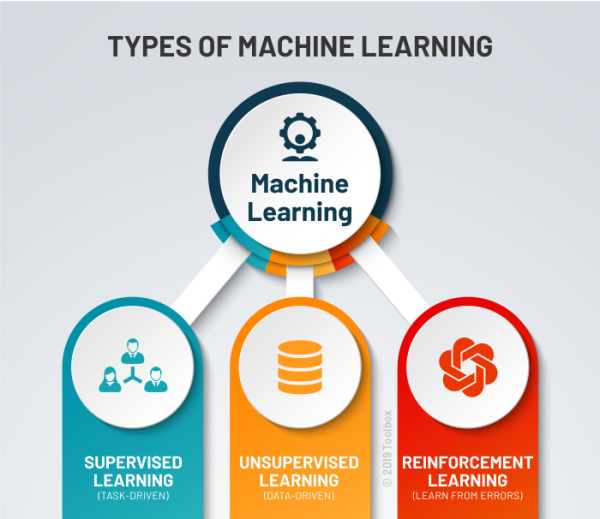
In order to predict loan acceptance, Ramachandra *et al*.'s [[17]](#r5) study used machine learning techniques like Random Forest, Decision Tree, Logistic Regression and it was 86% accurate. Their study provides insights into data pretreatment, algorithm selection, and result interpretation, and it shows that cloud-based platforms can be a viable solution for implementing loan prediction models.

A modernized loan approval system based on the XGBoost, Random Forest, and Decision Tree algorithms was created by Singh *et al*. [[18]](#r5) to achieve accurate loan prediction. By reliably determining loan eligibility and boosting lending volume, their study helps banks reduce losses. The architecture diagram of the system demonstrates how well it can forecast the results of loan approval by improving efficiency and risk management.

Karthikeyan *et al*. [[19]](#r5) study focuses on enhancing loan prediction models using machine learning techniques, particularly Random Forest and Boruta Algorithm. It addresses the challenges in credit risk assessment and feature selection for loan approval. The research emphasizes the importance of data preprocessing and feature engineering to improve prediction accuracy. Additionally, it explores the application of social network analysis in identifying key customers for loan approval. This study contributes to the advancement of loan prediction systems in the banking sector.

The study investigated by Amira *et al*. [[20]](#r5) focused on consumer loan default prediction using ensemble neural networks with different training algorithms, comparing their efficacy. Results indicate that the ensemble model outperforms individual models, with the Levenberg-Marquardt algorithm and a specific attribute filtering function showing promising results. This contributes to understanding credit risk assessment and model optimization in banking.

## SOME MACHINE LEARNING METHODS:

****

**Fig 2.2.1** Types of Machines Leaning

### Supervised machine learning algorithms:

Supervised machine learning algorithms can apply what has been learned in the past to new data using labeled examples to predict future events. Starting from the analysis of a known training dataset, the learning algorithm produces an inferred function to make predictions about the output values. The system is able to provide targets for any new input after sufficient training. The learning algorithm can also compare its output with the correct, intended output and find errors in order to modify the model accordingly.

### Unsupervised machine learning algorithms:

Unsupervised machine learning algorithms are used when the information used to train is neither classified nor labeled. Unsupervised learning studies how systems can infer a function to describe a hidden structure from unlabeled data. The system doesn’t figure out the right output, but it explores the data and can draw inferences from datasets to describe hidden structures from unlabeled data.

### Reinforcement machine learning algorithms:

Reinforcement machine learning algorithms is a learning method that interacts with its environment by producing actions and discovers errors or rewards. Trial and error search and delayed reward are the most relevant characteristics of reinforcement learning. This method allows machines and software agents to automatically determine the ideal behavior within a specific context in order to maximize its performance. Simple reward feedback is required for the agent to learn which action is best. This is known as the reinforcement signal.

## APPLICATIONS OF MACHINE LEARNING:

* + 1. Virtual Personal Assistants
    2. Predictions while Commuting
    3. Videos Surveillance
    4. Social Media Services
    5. Email Spam and Malware Filtering
    6. Online Customer Support
    7. Search Engine Result Refining
    8. Product Recommendations
    9. Online Fraud Detection

# 2.4 Machine Learning Models:

# Random Forest:

# Random Forest is a versatile ensemble learning method that constructs multiple decision trees during training. Each tree is trained on a random subset of features and makes independent predictions. The final prediction is determined by aggregating the predictions of individual trees, typically through a voting mechanism. This approach enhances model robustness and mitigates overfitting, making Random Forest particularly effective for handling complex datasets with high dimensionality. Additionally, Random Forest provides insights into feature importance, aiding in the interpretation of model predictions. Its ability to handle missing values and outliers further enhances its utility in real-world applications. Overall, Random Forest is a powerful algorithm suitable for a wide range of classification and regression tasks.

# Decision Tree:

# Decision Tree is a fundamental supervised learning algorithm that recursively partitions the feature space into regions based on the value of input features. These partitions form a tree-like structure where each internal node represents a decision based on a feature, and each leaf node corresponds to a class label or a numerical value. Decision Trees are intuitive and easy to interpret, making them valuable for exploratory analysis and decision-making. However, they are prone to overfitting, especially with complex datasets. Techniques like pruning and limiting tree depth help alleviate overfitting and improve generalization performance. Despite their limitations, Decision Trees serve as building blocks for more advanced ensemble methods like Random Forest and Gradient Boosting.

# Logistic Regression:

# Logistic Regression is a classic linear model used for binary classification tasks. It models the probability of a binary outcome as a function of input features using the logistic function. Despite its name, Logistic Regression is a classification algorithm rather than a regression algorithm. It is robust, computationally efficient, and offers interpretable coefficients that quantify the impact of each feature on the predicted probability. However, Logistic Regression assumes a linear relationship between features and the log-odds of the outcome, limiting its flexibility in capturing complex patterns. Regularization techniques like L1 and L2 regularization help prevent overfitting and improve generalization performance.

# K-Nearest Neighbors (KNN):

# K-Nearest Neighbors is a simple yet powerful instance-based learning algorithm used for classification and regression tasks. It classifies a new data point by assigning it the most common class label among its k nearest neighbors in the feature space. KNN is non-parametric and requires no training phase, making it easy to implement and suitable for datasets with complex decision boundaries. However, KNN's performance heavily depends on the choice of the distance metric and the value of k. Additionally, it suffers from the curse of dimensionality, where the effectiveness of nearest neighbor search decreases as the dimensionality of the feature space increases.

# Gradient Boosting:

# Gradient Boosting is a powerful ensemble learning technique that builds a predictive model in a stage-wise fashion by sequentially adding weak learners to minimize the loss function. Each weak learner is trained to correct the errors of its predecessor, resulting in a strong learner that combines the strengths of multiple models. Gradient Boosting is versatile and can be applied to various types of data and loss functions. It is particularly effective for regression and classification tasks, achieving state-of-the-art performance in many domains. However, Gradient Boosting is computationally intensive and sensitive to hyperparameters, requiring careful tuning to prevent overfitting. Regularization techniques like shrinkage and tree pruning help control model complexity and improve generalization performance.

# Parameterized Gradient Boosting:

# Parameterized Gradient Boosting is an extension of traditional Gradient Boosting that introduces additional parameters to control the learning process. These parameters include the learning rate, which determines the contribution of each weak learner to the final model, and the maximum depth of the trees, which controls the complexity of individual learners. Parameterized Gradient Boosting allows for more fine-grained control over the learning process and can improve model performance by mitigating overfitting. However, tuning these parameters requires careful experimentation and validation to achieve optimal results.

# Extra Trees:

# Extra Trees, short for Extremely Randomized Trees, is an ensemble learning method that builds multiple decision trees using random subsets of features and random splits at each node. Unlike Random Forest, which selects the best split among a subset of features, Extra Trees randomly selects splits, resulting in faster training times and reduced variance. Extra Trees is robust to noise and outliers and less prone to overfitting compared to traditional decision trees. However, the randomness introduced during training makes the interpretation of feature importance less straightforward. Nonetheless, Extra Trees is a powerful algorithm suitable for high-dimensional datasets and complex classification and regression tasks.

# Voting Ensemble:

# Voting Ensemble is a meta-algorithm that combines the predictions of multiple base models to make a final prediction. It operates by aggregating the individual predictions through a voting mechanism, where the most common prediction among the base models is chosen as the final output. Voting Ensemble can be implemented with various methods, including simple majority voting, weighted voting, and soft voting. It leverages the diversity of base models to improve predictive performance and generalization. Voting Ensemble is particularly effective when the base models have complementary strengths and weaknesses, leading to more robust and accurate predictions and also high robustness compared to other models as shown in [fig 2.4.1](#fvote).

**Fig 2.4.1** Ensemble Voting Model

# 3. SYSTEM ANALYSIS

## IMPORTANCE OF MACHINE LEARNING USING PYTHON:

The importance of machine learning in wine quality is increasing because of its ability to process huge datasets efficiently beyond the range of human capability, and then dependably convert analysis of that data into clinical insights that assist in planning and providing care, which ultimately leads to better outcomes, reduces the consumption. Using these types of advanced analytics, we can provide better information at the point of consumption.

## 3.2 IMPLEMENTATION OF MACHINE LEARNING USING PYTHON:

Python is a popular programming language. It was created in 1991 by Guido van Rossum. It is used for.

1. Web development (server-side),
2. Software development,
3. Mathematics,
4. System scripting.

The most recent major version of Python is Python 3. However, Python 2, although not being updated with anything other than security updates, is still quite popular. It is possible to write Python in an Integrated Development Environment, such as Thonny, PyCharm, NetBeans or Eclipse, Anaconda which are particularly useful when managing larger collections of Python files.

Python was designed for its readability. Python uses new lines to complete a command, as opposed to other programming languages which often use semicolons or parentheses. Python relies on indentation, using whitespace, to define scope; such as the scope of loops, functions and classes. Other programming languages often use curly-brackets for this purpose. In the

older days, people used to perform Machine Learning tasks manually by coding all the algorithms and mathematical and statistical formula.

This made the process time consuming, tedious and inefficient. But in the modern days, it is become very much easy and efficient compared to the olden days by various python libraries, f frameworks, and modules. Today, Python is one of the most popular programming languages for or this task and it has replaced many languages in the industry, one of the reasons is its vast co llection of libraries.

Python libraries that used in Machine Learning are:

1.Numpy, 2. Scipy, 3. Scikit-learn, 4. Pandas, 5. Matplotlib.

1. **NumPy:** It is a very popular python library for large multi-dimensional array and matrix processing, with the help of a large collection of high-level mathematical functions. It is very useful for fundamental scientific computations in Machine Learning. It is particularly useful for linear algebra, Fourier transform, and random number capabilities. High-end libraries like TensorFlow uses NumPy internally for manipulation of Tensors.
2. **SciPy**: is a very popular library among Machine Learning enthusiasts as it contains different modules for optimization, linear algebra, integration and statistics. There is a difference between the SciPy library and the SciPy stack. The SciPy is one of the core packages that make up the SciPy stack. SciPy is also very useful for image manipulation.
3. **Scikit-learn:** Scikit-learn is one of the most popular Machine Learning libraries for classical Machine Learning algorithms. It is built on top of two basic Python libraries, NumPy and SciPy. Scikit- learn supports most of the supervised and unsupervised learning algorithms. Scikit learn can also be used for datamining and data-analysis, which makes it a great tool who is starting out with Machine Learning.

## 4. Pandas: Pandas is a popular Python library for data analysis. It is not directly related to Machine Learning. As we know that the dataset must be prepared before training. In this case, Pandas comes handy as it was developed specifically for data extraction and preparation. It provides high-level data structures and wide variety tools for data analysis. It provides many inbuilt methods for groping, combining and filtering data.

## 5. Matplotlib:

Matplotlib is a very popular Python library for data visualization. Like Pandas, it is not directly related to Machine Learning. It particularly comes in handy when a programmer wants to visualize the patterns in the data. It is a 2D plotting library used for creating 2D graphs and plots. A module named pyplot makes it easy for programmers for plotting as it provides features to control line styles, font properties, formatting axes, etc. It provides various kinds of graphs and plots for data 13 visualization, histogram, error charts, bar chats, etc.

## 3.3 SCOPE OF THE PROJECT

The scope of this system is to maintain text messages datasets labeled as stress and no stress, train the model using the large quantity of data present in datasets and predict whether the person has stress or not on new data during testing.

## 3.4 DATA SET ANALYSIS

# We are collecting a dataset from Kaggle website to make predictions and we have collected dataset that is available [[21].](#r5) The dataset contains Following Columns as shown in [fig 3.4.1](#fdataset).

# Fig 3.4.1 Dataset

# 1. Illustrating the Class Imbalance

# In [Fig 3.4.2](#fimbalance), the depiction of the "Loan\_Status" column reveals a notable class imbalance, with approximately 68.7% of entries categorized as "Y" denoting loan approval, while the remaining 31.3% are labelled as "N" indicating loan denial. This imbalance presents a significant challenge in predictive modeling, particularly for algorithms sensitive to class distribution.

# To address this issue, various strategies can be employed to mitigate the impact of class imbalance. Resampling techniques, including oversampling of the minority class and under sampling of the majority class, aim to rebalance the dataset and alleviate the effects of class imbalance. Oversampling methods like Synthetic Minority Over-sampling Technique (SMOTE) generate synthetic instances of the minority class to increase its representation in the dataset, while under sampling methods randomly remove instances from the majority class.

# Fig 3.4.2 Class Imbalance

# 3.5 Data Pre-processing

# Preprocessing is a crucial stage following data visualization, preparing the data for model training. This process typically involves handling missing values, outliers, and scaling features to a common range. Additionally, categorical variables may be encoded for compatibility with machine learning algorithms. Finally, the dataset is split into training and testing sets to facilitate model evaluation and validation.

# 1. Dealing Null Values:

# Addressing null values is paramount for ensuring the reliability of machine learning models and the overall quality of the dataset. By effectively managing null values through methods such as imputation or elimination, models can achieve greater accuracy while preserving the integrity of the data. This proactive approach reduces potential biases, enabling more precise forecasts and insightful analysis. [Fig 3.5.1](#fbnull) and [Fig 3.5.2](#fanull) highlight the frequency of missing values across different columns, shedding light on deficiencies in critical characteristics like Gender, Dependents, Self\_Employed, and Credit\_History.

# Fig 3.5.1 Frequency of Null Values in Dataset

# It is imperative to tackle these null values to uphold data reliability and integrity in subsequent analyses. To mitigate this issue, we employed the fillna approach for imputing missing values, thereby minimizing the risk of biases and enhancing the overall quality of the dataset. This meticulous handling of null values ensures a robust foundation for further studies and model development.

# Fig 3.5.2 Frequency of Null values After removal

# 2. One-hot Encoding:

# One-hot encoding plays a pivotal role in enhancing the compatibility of categorical variables with numerical computations, thereby facilitating their utilization in machine learning algorithms. This transformation process converts categorical variables into a format suitable for algorithmic analysis, enabling algorithms to effectively interpret and learn from categorical features. By expressing categorical data as binary vectors, one-hot encoding expands the feature space while preserving the unique characteristics of each category. This augmentation of the feature space enriches the model's ability to capture nuances within categorical variables, ultimately enhancing its predictive performance. Following the removal of irrelevant columns, as depicted in [Fig 3.5.3](#fhot), the dataset undergoes a streamlined transformation, ensuring that only pertinent features are retained for subsequent analysis. This meticulous approach to feature engineering, including one-hot encoding, contributes to the creation of a robust and comprehensive dataset, primed for effective model training and predictive analytics.

# Fig 3.5.3 Columns After One-hot Encoding

# 3. Eliminating Data Outliers

**Fig 3.5.4** Visualizing data outliers

# Addressing outliers is paramount for ensuring the accuracy and reliability of both machine learning models and statistical analyses. Outliers possess the potential to significantly skew results, leading to misinterpretations and spurious findings. By acknowledging and effectively managing outliers, researchers can uphold the robustness and credibility of their data analysis, thereby enhancing the quality and validity of their research outcomes. [Fig.3.5.4](#fout) serves as a visual representation of the presence of outliers within the dataset, highlighting the need for intervention to mitigate their influence on investigative outcomes and model performance. Leveraging the quantile approach, we were able to tackle this challenge by establishing thresholds based on quantiles to identify and subsequently eliminate extreme values. Through this strategic approach, we successfully attenuated the impact of outliers on our research findings, safeguarding the integrity and precision of our study. This meticulous handling of outliers underscores our commitment to conducting rigorous and reliable data analysis, ultimately fortifying the trustworthiness and significance of our research endeavors.

# 4. Square Root Transformation

Square root transformation serves as a crucial tool for stabilizing variance, especially in datasets characterized by skewed distributions. By mitigating the influence of extreme values, this transformation fosters greater symmetry within the data and aligns it more closely with normality assumptions. This alignment is essential for ensuring reliable inference and bolstering the performance of statistical models reliant on such assumptions. Through square root transformation, the distribution of the data becomes more uniform, facilitating more accurate parameter estimation and hypothesis testing. This process not only enhances the interpretability of the data but also strengthens the validity and generalizability of the insights derived from subsequent analyses. In essence, square root transformation plays a pivotal role in optimizing the robustness and efficacy of statistical modeling approaches, ultimately contributing to more informed decision-making and a deeper understanding of complex phenomena.

## 5. Correlation

## Fig 3.5.5 Correlation

## The linear link between two variables is measured using correlation to determine its strength and direction. There in [fig 3.5.5](#fcorr), doesn't seem to be any discernible relationship between Applicant\_Income, Coapplicant\_Income, and Loan\_mount in the illustration. This implies that there is not always a linear relationship between changes in one variable and changes in the other variables.

# 6. Feature Selection

# Feature selection is crucial for enhancing model performance and interpretability by identifying the most relevant attributes that contribute to the predictive task. By reducing the dimensionality of the dataset, feature selection mitigates the risk of overfitting and improves computational efficiency. Additionally, it enhances model generalization by focusing on the most informative features, leading to more accurate predictions on unseen data. Moreover, feature selection aids in identifying redundant or irrelevant features, simplifying the model and facilitating better understanding of underlying relationships. Overall, effective feature selection is essential for building more robust, efficient, and interpretable machine learning models.

# Recursive Feature Elimination: This Process systematically removes less important features, iteratively refining the model's predictive power. By repeatedly training the model and discarding the least significant features, it enhances model efficiency and interpretability. This process helps mitigate the curse of dimensionality, improving model generalization and reducing overfitting. Additionally, recursive feature elimination streamlines the feature set, making the model more computationally tractable and enhancing its performance on unseen data. Overall, this iterative approach to feature selection is a powerful technique for optimizing model accuracy and efficiency.

# 7. SMOTE

# In situations where one class significantly outweighs the other, such as in the prediction of loan acceptance, addressing class imbalance becomes imperative, and SMOTE (Synthetic Minority Over-sampling Technique) emerges as a pivotal solution. This technique plays a crucial role in rebalancing datasets by generating synthetic instances for the minority class, thereby reducing bias towards the majority class and enhancing overall model performance. As depicted in [Fig 3.5.6](#fsmote), the application of SMOTE effectively mitigates the class imbalance bias, resulting in a more balanced and representative dataset suitable for training machine learning models. By leveraging SMOTE, the model's ability to generalize across both classes is significantly bolstered, fostering fairer and more accurate forecasts of loan acceptance. This approach not only ensures equitable treatment of both classes but also strengthens the model's predictive capabilities, ultimately leading to more reliable and trustworthy outcomes in the domain of loan approval prediction.

## Fig 3.5.6 Loan\_Status after SMOTE

# 3.6 Classification:

# In the development of our classification model, we employ a diverse array of machine learning algorithms to ensure robust performance across various scenarios. Random Forest harnesses the power of ensemble learning, constructing multiple decision trees to improve predictive accuracy and handle complex datasets effectively. Decision Tree offers transparency and simplicity, facilitating easy interpretation of decision-making processes. Logistic Regression provides a probabilistic framework for binary classification tasks, offering insights into the influence of individual features on the outcome. K-Nearest Neighbors (KNN) relies on similarity measures to classify data points, making it intuitive and versatile for classification tasks. Gradient Boosting iteratively improves model performance by sequentially fitting weak learners to correct errors, achieving high predictive accuracy. Parameterized Gradient Boosting introduces additional parameters to fine-tune the learning process, enhancing model flexibility and performance. Extra Trees leverages randomization to build decision trees, reducing variance and improving generalization. Finally, Voting Ensemble combines predictions from multiple base models to produce a robust and reliable final prediction, leveraging the diversity of individual algorithms for improved accuracy and stability. Collectively, these models offer a comprehensive toolkit for classification tasks, each with its unique strengths and capabilities to address diverse challenges in the domain.

# IMPLEMENTATION CODE

**Importing The Required Modules**

importimportimportimport numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

import warnings

import os

import scipy

from scipy import stats

from scipy.stats import pearsonr

from scipy.stats import ttest\_ind

from sklearn.metrics import make\_scorer, roc\_auc\_score, confusion\_matrix, accuracy\_score, roc\_curve,classification\_report

from sklearn.model\_selection import train\_test\_split

from sklearn.linear\_model import LogisticRegression

from sklearn.neighbors import KNeighborsClassifier

from sklearn.tree import DecisionTreeClassifier

from sklearn.ensemble import RandomForestClassifier

from sklearn.ensemble import GradientBoostingClassifier

from sklearn.ensemble import ExtraTreesClassifier

from sklearn.model\_selection import GridSearchCV, RandomizedSearchCV

from sklearn.preprocessing import LabelEncoder

from sklearn.model\_selection import cross\_val\_score, KFold

**Reading The Dataset**

df = pd.read\_csv('/content/drive/MyDrive/project/train\_u6lujuX\_CVtuZ9i.csv')

df.head()

**Dataset Analysis**

df.shape

df.columns

df.info()

df.describe()

**Dataset Visualization**

df.Gender.value\_counts(dropna=False)

countMale = len(df[df.Gender == 'Male'])

countFemale = len(df[df.Gender == 'Female'])

countNull = len(df[df.Gender.isnull()])

print("Percentage of Male applicant: {:.2f}%".format((countMale / (len(df.Gender))\*100)))

print("Percentage of Female applicant: {:.2f}%".format((countFemale / (len(df.Gender))\*100)))

print("Missing values percentage: {:.2f}%".format((countNull / (len(df.Gender))\*100)))

df.Married.value\_counts(dropna=False)

sns.countplot(x="Married", data=df, palette="mako")

plt.show()

countMarried = len(df[df.Married == 'Yes'])

countNotMarried = len(df[df.Married == 'No'])

countNull = len(df[df.Married.isnull()])

print("Percentage of married: {:.2f}%".format((countMarried / (len(df.Married))\*100)))

print("Percentage of Not married applicant: {:.2f}%".format((countNotMarried / (len(df.Married))\*100)))

print("Missing values percentage: {:.2f}%".format((countNull / (len(df.Married))\*100)))

df.Education.value\_counts(dropna=False)

sns.countplot(x="Education", data=df, palette="mako")

plt.show()

countGraduate = len(df[df.Education == 'Graduate'])

countNotGraduate = len(df[df.Education == 'Not Graduate'])

countNull = len(df[df.Education.isnull()])

print("Percentage of Graduates: {:.2f}%".format((countGraduate / (len(df.Education))\*100)))

print("Percentage of NotGraduates: {:.2f}%".format((countNotGraduate / (len(df.Education))\*100)))

print("Missing values percentage: {:.2f}%".format((countNull / (len(df.Education))\*100)))

df.Self\_Employed.value\_counts(dropna=False)

sns.countplot(x="Self\_Employed", data=df, palette="magma")

plt.show()

countNo = len(df[df.Self\_Employed == 'No'])

countYes = len(df[df.Self\_Employed == 'Yes'])

countNull = len(df[df.Self\_Employed.isnull()])

print("Percentage of Not self employed: {:.2f}%".format((countNo / (len(df.Self\_Employed))\*100)))

print("Percentage of self employed: {:.2f}%".format((countYes / (len(df.Self\_Employed))\*100)))

print("Missing values percentage: {:.2f}%".format((countNull / (len(df.Self\_Employed))\*100)))

df.Credit\_History.value\_counts(dropna=False)

sns.countplot(x="Credit\_History", data=df, palette="Spectral")

plt.show()

count1 = len(df[df.Credit\_History == 1])

count0 = len(df[df.Credit\_History == 0])

countNull = len(df[df.Credit\_History.isnull()])

print("Percentage of Good credit history: {:.2f}%".format((count1 / (len(df.Credit\_History))\*100)))

print("Percentage of Bad credit history: {:.2f}%".format((count0 / (len(df.Credit\_History))\*100)))

print("Missing values percentage: {:.2f}%".format((countNull / (len(df.Credit\_History))\*100)))

df.Property\_Area.value\_counts(dropna=False)

sns.countplot(x="Property\_Area", data=df, palette="mako")

plt.show()

countUrban = len(df[df.Property\_Area == 'Urban'])

countNotRural = len(df[df.Property\_Area == 'Rural'])

countSemiurban = len(df[df.Property\_Area == 'Semiurban'])

countNan = len(df[df.Property\_Area.isnull()])

print("Percentage of Urban Applicants: {:.2f}%".format((countUrban / (len(df.Property\_Area))\*100)))

print("Percentage of Rural applicant: {:.2f}%".format((countNotRural / (len(df.Property\_Area))\*100)))

print("Percentage of Semi urban applicant: {:.2f}%".format((countSemiurban / (len(df.Property\_Area))\*100)))

print("Missing values percentage: {:.2f}%".format((countNull / (len(df.Property\_Area))\*100)))

df.Loan\_Status.value\_counts(dropna=False)

plt.figure(figsize=(8, 8))

df['Loan\_Status'].value\_counts().plot.pie(autopct='%1.1f%%', colors=['green','red'])

plt.title('Loan Approval Status Distribution')

plt.show()

countY = len(df[df.Loan\_Status == 'Y'])

countN = len(df[df.Loan\_Status == 'N'])

countNull = len(df[df.Loan\_Status.isnull()])

print("Percentage of Approved: {:.2f}%".format((countY / (len(df.Loan\_Status))\*100)))

print("Percentage of Rejected: {:.2f}%".format((countN / (len(df.Loan\_Status))\*100)))

print("Missing values percentage: {:.2f}%".format((countNull / (len(df.Loan\_Status))\*100)))

**Dealing Null Values**

df.isnull().sum()

df['Gender'].fillna(df['Gender'].mode()[0],inplace=True)

df['Married'].fillna(df['Married'].mode()[0],inplace=True)

df['Dependents'].fillna(df['Dependents'].mode()[0],inplace=True)

df['Self\_Employed'].fillna(df['Self\_Employed'].mode()[0],inplace=True)

df['Credit\_History'].fillna(df['Credit\_History'].mode()[0],inplace=True)

df['Loan\_Amount\_Term'].fillna(df['Loan\_Amount\_Term'].mode()[0],inplace=True)

df['LoanAmount'].fillna(df['LoanAmount'].mean(),inplace=True)

df.isnull().sum()

**Dealing Duplicates**

df.duplicated().sum()

**Visualizing Loan Approval status vs individual features**

def bar\_chart(col):

Approved = df[df["Loan\_Status"]=="Y"][col].value\_counts()

Disapproved = df[df["Loan\_Status"]=="N"][col].value\_counts()

df1 = pd.DataFrame([Approved, Disapproved])

df1.index = ["Approved", "Disapproved"]

df1.plot(kind="bar")

plt.title(f'Loan Approval Status based on {col}')

plt.xlabel(col)

plt.ylabel('Count')

bar\_chart("Gender")

bar\_chart("Married")

bar\_chart("Education")

bar\_chart("Self\_Employed")

bar\_chart("Credit\_History")

bar\_chart("Property\_Area")

num=['ApplicantIncome', 'CoapplicantIncome', 'LoanAmount',

'Loan\_Amount\_Term']

for column in num:

plt.figure(figsize=(8,4))

plt.hist(df[column])

plt.title(column)

plt.xlabel('Values')

plt.ylabel('Count')

plt.show()

**Dropping Loan\_ID**

df = df.drop(['Loan\_ID'], axis = 1)

df.head()

**One-hot Encoding**

df = pd.get\_dummies(df)

df = df.drop(['Gender\_Female', 'Married\_No', 'Education\_Not Graduate',

'Self\_Employed\_No', 'Loan\_Status\_N'], axis = 1)

new = {'Gender\_Male': 'Gender', 'Married\_Yes': 'Married',

'Education\_Graduate': 'Education', 'Self\_Employed\_Yes': 'Self\_Employed',

'Loan\_Status\_Y': 'Loan\_Status'}

df.rename(columns=new, inplace=True)

print(df.head())

df.shape

**Handling Outliers**

plt.figure(figsize=(12,12))

for i,col in enumerate(['ApplicantIncome', 'CoapplicantIncome', 'LoanAmount',

'Loan\_Amount\_Term', 'Credit\_History', 'Gender', 'Married','Dependents\_0', 'Dependents\_1', 'Dependents\_2', 'Dependents\_3+','Education','Self\_Employed', 'Property\_Area\_Rural','Property\_Area\_Semiurban', 'Property\_Area\_Urban']):

plt.subplot(4,4,i+1)

sns.boxplot(x=col,data=df)

plt.show()

Q1 = df.quantile(0.25)

Q3 = df.quantile(0.75)

IQR = Q3 - Q1

aaf=df[~((df < (Q1 - 1.5 \* IQR)) |(df > (Q3 + 1.5 \* IQR))).any(axis=1)]

print(aaf.head())

print(aaf.shape)

df=aaf

ddf=df['Loan\_Status']

plt.figure(figsize=(12,12))

for i,col in enumerate(['ApplicantIncome', 'CoapplicantIncome', 'LoanAmount',

'Loan\_Amount\_Term', 'Credit\_History', 'Gender', 'Married','Dependents\_0', 'Dependents\_1', 'Dependents\_2', 'Dependents\_3+','Education','Self\_Employed', 'Property\_Area\_Rural','Property\_Area\_Semiurban', 'Property\_Area\_Urban']):

plt.subplot(4,4,i+1)

sns.boxplot(x=col,data=df)

plt.show()

**Square root Transformation**

column\_to\_transform = 'ApplicantIncome '

fig, axes = plt.subplots(1, 2, figsize=(12, 5))

sns.histplot(df['ApplicantIncome'], kde=True, ax=axes[0])

axes[0].set\_title('Original Data')

axes[0].set\_xlabel(column\_to\_transform)

axes[0].set\_ylabel('Frequency')

df.ApplicantIncome = np.sqrt(df.ApplicantIncome)

df.columns

sns.histplot(df['ApplicantIncome'], kde=True, ax=axes[1])

axes[1].set\_title('Square Root Transformed Data')

axes[1].set\_xlabel('Square Root of ' + column\_to\_transform)

axes[1].set\_ylabel('Frequency')

plt.tight\_layout()

plt.show()

column\_to\_transform = 'CoapplicantIncome '

fig, axes = plt.subplots(1, 2, figsize=(12, 5))

sns.histplot(df['CoapplicantIncome'], kde=True, ax=axes[0])

axes[0].set\_title('Original Data')

axes[0].set\_xlabel(column\_to\_transform)

axes[0].set\_ylabel('Frequency')

df.CoapplicantIncome = np.sqrt(df.CoapplicantIncome)

df.columns

sns.histplot(df['CoapplicantIncome'], kde=True, ax=axes[1])

axes[1].set\_title('Square Root Transformed Data')

axes[1].set\_xlabel('Square Root of ' + column\_to\_transform)

axes[1].set\_ylabel('Frequency')

plt.tight\_layout()

plt.show()

column\_to\_transform = 'LoanAmount'

fig, axes = plt.subplots(1, 2, figsize=(12, 5))

sns.histplot(df['LoanAmount'], kde=True, ax=axes[0])

axes[0].set\_title('Original Data')

axes[0].set\_xlabel(column\_to\_transform)

axes[0].set\_ylabel('Frequency')

df.LoanAmount = np.sqrt(df.LoanAmount)

df.columns

sns.histplot(df['LoanAmount'], kde=True, ax=axes[1])

axes[1].set\_title('Square Root Transformed Data')

axes[1].set\_xlabel('Square Root of ' + column\_to\_transform)

axes[1].set\_ylabel('Frequency')

plt.tight\_layout()

plt.show()

**Correlation**

plt.figure(figsize=(7,7))

plt.title('Correlation of variables')

corr = df[['ApplicantIncome','CoapplicantIncome','LoanAmount']]

sns.heatmap(corr.astype(float).corr(),vmax=1.0, annot=True, cmap='Greens', cbar=True, square= True, fmt='.1f', annot\_kws={'size':15})

**Feature selection using Recursive Feature Elimination**

import pandas as pd

from sklearn.feature\_selection import RFE

from sklearn.linear\_model import LogisticRegression

from sklearn.model\_selection import train\_test\_split

X = df.drop(["Loan\_Status"], axis=1)

y = df["Loan\_Status"]

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

logreg\_model = LogisticRegression()

rfe = RFE(estimator=logreg\_model, n\_features\_to\_select=6)

rfe.fit(X\_train, y\_train)

selected\_features = X\_train.columns[rfe.support\_]

X\_train\_selected = rfe.transform(X\_train)

X\_test\_selected = rfe.transform(X\_test)

print("Selected Features:", selected\_features)

**Applying SMOTE**

X = df

y = ddf

from imblearn import under\_sampling, over\_sampling

from imblearn.over\_sampling import SMOTE

X, y = SMOTE().fit\_resample(X,y)

plt.figure(figsize=(8, 8))

y.value\_counts().plot.pie(autopct='%1.1f%%', colors=['green','red'])

plt.title('Class distribution after SMOTE')

plt.show()

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size = 0.2, random\_state = 4)

print ('Train set:', X\_train.shape, y\_train.shape)

print ('Test set:', X\_test.shape, y\_test.shape)

**Model Development**

Training\_Accuracy\_L=[0]\*6

Test\_Accuracy\_L=[0]\*6

Sensitivity\_L=[0]\*6

Specificity\_L=[0]\*6

F1Score\_L=[0]\*6

Precision\_L=[0]\*6

Negative\_Predictive\_Value\_L=[0]\*6

False\_Negative\_Rate\_L=[0]\*6

False\_Positive\_Rate\_L=[0]\*6

False\_Discovery\_Rate\_L=[0]\*6

False\_Omission\_Rate\_L=[0]\*6

cv\_accuracy\_L=[0]\*6

import math

def rounder(n):

try:

return math.ceil(n \* 1000) / 1000

except:

return n

def fun(model,name,num\_folds,index):

test\_pred = model.predict(X\_test)

train\_pred = model.predict(X\_train)

train\_acc=rounder(accuracy\_score(y\_train,train\_pred)\*100)

test\_acc=rounder(accuracy\_score(y\_test,test\_pred)\*100)

Training\_Accuracy\_L[index]=train\_acc

Test\_Accuracy\_L[index]=test\_acc

print("\nTraining Accuracy:", train\_acc)

print("\nTesting Accuracy:",test\_acc)

print(classification\_report(y\_test,test\_pred))

test\_conf\_matrix = confusion\_matrix(y\_test,test\_pred)

plt.figure(figsize=(4, 4))

sns.heatmap(test\_conf\_matrix, annot=True, fmt='g', cmap='Greens', cbar=False)

t=name+' Confusion Matrix - Test Set'

plt.title(t)

plt.xlabel('Predicted')

plt.ylabel('Actual')

plt.show()

tn, fp,fn,tp = test\_conf\_matrix.ravel()

Sensitivity=rounder((tp) / (tp + fn))

Sensitivity\_L[index]=Sensitivity

Specificity=rounder((tn) / (tn + fp))

Specificity\_L[index]=Specificity

F1Score=rounder( (2 \* tp) / (2 \* tp+ fp + fn))

F1Score\_L[index]=F1Score

Precision=rounder((tp) / (tp +fp))

Precision\_L[index]=Precision

Negative\_Predictive\_Value= rounder((tn) / (tn + fn))

Negative\_Predictive\_Value\_L[index]=Negative\_Predictive\_Value

False\_Negative\_Rate=rounder((fn) / (fn + tp))

False\_Negative\_Rate\_L[index]=False\_Negative\_Rate

False\_Positive\_Rate=rounder((fp) / (fp + tn))

False\_Positive\_Rate\_L[index]=False\_Positive\_Rate

False\_Discovery\_Rate=rounder((fp) / (fp + tp))

False\_Discovery\_Rate\_L[index]=False\_Discovery\_Rate

False\_Omission\_Rate=rounder((fn) / (fn+ tn))

False\_Omission\_Rate\_L[index]=False\_Omission\_Rate

print('Sensitivity:', Sensitivity)

print('Specificity:', Specificity)

print('F1 Score:', F1Score)

print('Precision:',Precision)

print('Negative Predictive Value:', Negative\_Predictive\_Value)

print('False Negative Rate:',False\_Negative\_Rate)

print('False Positive Rate:',False\_Positive\_Rate)

print('False Discovery Rate:',False\_Discovery\_Rate)

print('False Omission Rate:', False\_Omission\_Rate)

test\_probabilities =model.predict\_proba(X\_test)[:, 1]

auc\_score = roc\_auc\_score(y\_test,test\_probabilities)

fpr, tpr, thresholds = roc\_curve(y\_test, test\_probabilities)

plt.figure(figsize=(8, 6))

plt.plot(fpr, tpr, label=''+name+'(AUC = {:.2f})'.format(auc\_score))

plt.plot([0, 1], [0, 1], linestyle='--', color='gray', label='Random Guess')

plt.xlabel('False Positive Rate')

plt.ylabel('True Positive Rate')

plt.title('ROC Curve - '+name)

plt.legend()

plt.show()

kf = KFold(n\_splits=num\_folds, shuffle=True,random\_state=42)

cv\_scores = cross\_val\_score(model, X\_train, y\_train, cv=kf, scoring='accuracy')

print(f"\n{num\_folds}-Fold Cross-Validation Scores:")

print(cv\_scores)

print(f"\nCross-Validation Accuracy Score: {max(cv\_scores) \* 100:.2f}%")

cv\_accuracy\_L[index]=max(cv\_scores)\*100

**Random Forest**

scoreListRF = []

for i in range(2,25):

RFclassifier = RandomForestClassifier(n\_estimators = 1000, random\_state = 1, max\_leaf\_nodes=i)

RFclassifier.fit(X\_train, y\_train)

scoreListRF.append(RFclassifier.score(X\_test, y\_test))

plt.plot(range(2,25), scoreListRF)

plt.xticks(np.arange(2,25,1))

plt.xlabel("RF Value")

plt.ylabel("Score")

plt.show()

RFAcc = max(scoreListRF)

print("Random Forest Accuracy: {:.2f}%".format(RFAcc\*100))

fun(RFclassifier,'Random Forest',14,0)

**Extra tree clssifier**

from sklearn.ensemble import ExtraTreesClassifier

scoreListET=[]

for i in range(2, 25):

ETclassifier = ExtraTreesClassifier(n\_estimators=1000, random\_state=1, max\_leaf\_nodes=i)

ETclassifier.fit(X\_train, y\_train)

scoreListET.append(ETclassifier.score(X\_test, y\_test))

plt.plot(range(2, 25), scoreListET)

plt.xticks(np.arange(2, 25, 1))

plt.xlabel("Number of Leaf Nodes")

plt.ylabel("Accuracy Score")

plt.title("Extra Trees Classifier - Accuracy vs Number of Leaf Nodes")

plt.show()

ETAcc = max(scoreListET)

print("Extra Trees Classifier Accuracy: {:.2f}%".format(ETAcc \* 100))

fun(ETclassifier,'Extra Trees Classifier',6,1)

**LR Clssifier**

LRclassifier = LogisticRegression(solver='saga', max\_iter=500, random\_state=1)

LRclassifier.fit(X\_train, y\_train)

y\_pred = LRclassifier.predict(X\_test)

LRAcc = accuracy\_score(y\_pred,y\_test)

print('LR accuracy: {:.2f}%\n'.format(LRAcc\*100))

fun(LRclassifier,'LR Classifier',5,2)

**KNN Classifier**

scoreListknn = []

for i in range(1,21):

KNclassifier = KNeighborsClassifier(n\_neighbors = i)

KNclassifier.fit(X\_train, y\_train)

scoreListknn.append(KNclassifier.score(X\_test, y\_test))

plt.plot(range(1,21), scoreListknn)

plt.xticks(np.arange(1,21,1))

plt.xlabel("K value")

plt.ylabel("Score")

plt.show()

KNAcc = max(scoreListknn)

print("KNN best accuracy: {:.2f}%".format(KNAcc\*100))

fun(KNclassifier,'KNN Classifier',8,3)

**Decision Tree Classifier**

scoreListDT = []

for i in range(2,21):

DTclassifier = DecisionTreeClassifier(max\_leaf\_nodes=i)

DTclassifier.fit(X\_train, y\_train)

scoreListDT.append(DTclassifier.score(X\_test, y\_test))

plt.plot(range(2,21), scoreListDT)

plt.xticks(np.arange(2,21,1))

plt.xlabel("Leaf")

plt.ylabel("Score")

plt.show()

DTAcc = max(scoreListDT)

print("Decision Tree Accuracy: {:.2f}%".format(DTAcc\*100))

fun(DTclassifier,'DecisionTree Classifier',6,4)

**ParamsGB Classifier**

paramsGB={'n\_estimators':[100,200,300,400,500],

'max\_depth':[1,2,3,4,5],

'subsample':[0.5,1],

'max\_leaf\_nodes':[2,5,10,20,30,40,50]}

GB = RandomizedSearchCV(GradientBoostingClassifier(), paramsGB, cv=20)

GB.fit(X\_train, y\_train)

GBclassifier = GradientBoostingClassifier(subsample=0.5, n\_estimators=400, max\_depth=4, max\_leaf\_nodes=10)

GBclassifier.fit(X\_train, y\_train)

y\_pred = GBclassifier.predict(X\_test)

GBAcc = accuracy\_score(y\_pred,y\_test)

print('Gradient Boosting accuracy: {:.2f}%'.format(GBAcc\*100))

fun(GBclassifier,'GradientBoosting Classifier', 12,5)

**Plotting Metric Values for all the Models**

compare = pd.DataFrame({'Model': ['Random Forest','Extra Trees Classifier','Logistic Regression', 'K Neighbors', 'Decision Tree',

'Gradient Boost'],

'Accuracy': [RFAcc\*100,ETAcc\*100,LRAcc\*100, KNAcc\*100,

DTAcc\*100, GBAcc\*100],

'Training Accuracy':Training\_Accuracy\_L,

'Test Accuracy':Test\_Accuracy\_L,

'Sensitivity':Sensitivity\_L,

'Specificity':Specificity\_L,

'F1 Score':F1Score\_L,

'Precision':Precision\_L,

'Negative Predictive Value':Negative\_Predictive\_Value\_L,

'False Negative Rate':False\_Negative\_Rate\_L,

'False Positive Rate':False\_Positive\_Rate\_L,

'False Discovery Rate':False\_Discovery\_Rate\_L,

'False Omission Rate':False\_Omission\_Rate\_L,

'cv-accuracy score':cv\_accuracy\_L,

})

compare.sort\_values(by='Accuracy', ascending=False)

**Visualized Comparison**

colors = ['aqua','skyblue', 'orange', 'green', 'red', 'purple', 'pink']

plt.figure(figsize=(10, 6))

plt.barh(compare['Model'], compare['Accuracy'], color=colors)

plt.xlabel('Accuracy (%)')

plt.title('Model Comparison - Accuracy')

plt.xlim(0, 100)

for index, value in enumerate(compare['Accuracy']):

plt.text(value, index, f'{value:.2f}', va='center', fontsize=10)

plt.show()

**Building Voting Ensemble Method**

from sklearn.ensemble import VotingClassifier

from sklearn.metrics import accuracy\_score

voting\_clf = VotingClassifier(estimators=[('randf', RFclassifier), ('extra', ETclassifier), ('pgb', GBclassifier)], voting='hard')

voting\_clf.fit(X\_train, y\_train)

ensemble\_prediction = voting\_clf.predict(X\_test)

ensemble\_accuracy = accuracy\_score(y\_test, ensemble\_prediction)

print("Accuracy - Voting Ensemble:", ensemble\_accuracy\*100)

num\_folds=13

kf = KFold(n\_splits=num\_folds, shuffle=True,random\_state=42)

cv\_scores = cross\_val\_score(voting\_clf, X\_train, y\_train, cv=kf, scoring='accuracy')

print(f"\n{num\_folds}-Fold Cross-Validation Scores:")

print(cv\_scores)

print(f"\nCross-Validation Accuracy Score: {max(cv\_scores) \* 100:.2f}%")

cv\_accuracy\_L.append(max(cv\_scores)\*100)

**Model Saving**

import joblib

joblib.dump(voting\_clf, "mymodel\_ensemble.pkl")

# Templates:

a.index.html b.form.html

c.result.html

## a.index.html

#Home Page

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Loan Approval Prediction System</title>

<link rel="stylesheet" href='static/index.css'>

<link rel="stylesheet" href='static/navbar.css'>

</head>

<body>

<div class='navbar'>

<h2>Loan Approval Prediction System</h2>

<div class='list'>

<ul>

<li><a href='../'>Home</a></li>

<li><a href='/form'>Check Eligibility</a></li>

</ul>

</div>

</div>

<div class='body'>

<div class='data'>

<h1>Welcome to <br>Loan Approval Prediction System 💰</h1><br>

<h3 style='margin-top:-8px'>Your Path to Financial Confidence🔑</h3>

<br>

<p style='width:500px'>Unlock the door to your dreams with our innovative Loan Approval Prediction System. We understand the importance of financial stability and are here to guide you on your journey. Rest assured, our system boasts a high accuracy rate, providing you with reliable insights to empower your financial decisions.</p>

<br>

<button><a href='/form'>Check Your Eligibility 🚀</a></button>

</div>

<div class='pic'>

<img src='/static/fincore.gif'/>

</div>

</div>

</body>

</html>

**b.form.html**

#Form Page

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Loan Approval Prediction System</title>

<link rel="stylesheet" href='static/navbar.css'>

<link rel="stylesheet" href='static/form.css'>

</head>

<body>

<div class='navbar'>

<h2>Loan Approval Prediction System</h2>

<div class='list'>

<ul>

<li><a href='../'>Home</a></li>

<li><a href='/form'>Check Eligibility</a></li>

</ul>

</div>

</div>

<div class='body'>

<img src='/static/i4.png'>

<div class="form">

<h3>Applicant Details</h3>

<form method="post" action="{{url\_for('form')}}">

<div class='i'>

<span class='ii'><p>Loan ID<span id='red'>\*</span></p></span>

<input type='text' value='' name='id'required/>

</div>

<div class='i'>

<span class='ii' ><p>Applicant Name<span id='red'>\*</span></p></span>

<input type='text' value='' name='name' required/>

</div>

<div class='i'>

<span class='ii'><p>Gender<span id='red'>\*</span></p></span>

<select name='gender' required>

<option value=''>select</option>

<option value='1'>Male</option>

<option value='0'>Female</option>

<option value='0'>Others</option>

</select>

</div>

<div class='i'>

<span class='ii'><p>Marrital Status<span id='red'>\*</span></p></span>

<select name='married' required>

<option value=''>select</option>

<option value='1'>Married</option>

<option value='0'>UnMarried</option>

</select>

</div>

<div class='i'>

<span class='ii'><p>Dependents<span id='red'>\*</span></p></span>

<input type='number' value='' name='dependents' required />

</div>

<div class='i'>

<span class='ii'><p>Education<span id='red'>\*</span></p></span>

<select name='education' required>

<option value=''>select</option>

<option value='1'>Graduate</option>

<option value='0'>Not Graduate</option>

</select>

</div>

<div class='i'>

<span class='ii'><p>Self Employed<span id='red'>\*</span></p></span>

<select name='semployed' required>

<option value=''>select</option>

<option value='1'>Yes</option>

<option value='0'>No</option>

</select>

</div>

<div class='i'>

<span class='ii'><p>Applicant Income<span id='red'>\*</span></p></span>

<input type='number' value='' name='aincome' required/>

</div>

<div class='i'>

<span class='ii'><p>Co-Applicant Income<span id='red'>\*</span></p></span>

<input type='number' value='' name='coincome' required/>

</div>

<div class='i'>

<span class='ii'><p>Loan Amount<span id='red'>\*</span></p></span>

<input type='number' value='' name='lamount' required/>

</div>

<div class='i'>

<span class='ii'><p>Loan Amount Term<span id='red'>\*</span></p></span>

<input type='number' value='' name='lterm' required/>

</div>

<div class='i'>

<span class='ii'><p>Credit History<span id='red'>\*</span></p></span>

<select name='chistory' required>

<option value=''>select</option>

<option value='1'>Yes</option>

<option value='0'>No</option>

</select>

</div>

<div class='i'>

<span class='ii'><p>Property Area<span id='red'>\*</span></p></span>

<select name='parea' required>

<option value=''>select</option>

<option value='rural'>Rural</option>

<option value='urban'>Urban</option>

<option value='semiurban'>Semi Urban</option>

</select>

</div>

<div class='i' id='buts'>

<input type='submit' value='Check Eligibility'/>

<input type='reset' value='Reset'/>

</div>

</form>

</div>

<img src='/static/i3.avif' id='i3'>

</div>

</body>

</html>

**c.result.html**

#Result Page

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Loan Approval Prediction System</title>

<link rel="stylesheet" href='static/result.css'>

<link rel="stylesheet" href='static/navbar.css'>

</head>

<body>

<div class='navbar'>

<h2>Loan Approval Prediction System</h2>

<div class='list'>

<ul>

<li><a href='../'>Home</a></li>

<li><a href='/form'>Check Eligibility</a></li>

</ul>

</div>

</div>

<div class='body'>

{% if x==1 %}

<div class='yes'>

<img src='/static/success.gif' id='i1'>

<div class='data'>

<h2 style='margin-left:40px;margin-top:-50px;line-height:40px'>Congratualtions 🎉🎉🎉<br> Dear {{name}}!! <br> You are Eligible For The Loan 💰 </h2>

<button><a href='../'>Go to Home🏡</a></button>

</div>

<img src='/static/success.gif' id='i2'>

</div>

{% else %}

<div class='no'>

<img src='/static/fail.gif' id='i1'>

<div class='data' id='fail'>

<h2 style='margin-left:40px;margin-top:-50px;line-height:40px'>Oops!! 😞😞😞<br> Dear {{name}}!! <br> You are Not Eligible For The Loan 💰 </h2>

<button><a href='../'>Go to Home🏡</a></button>

</div>

<img src='/static/fail.gif' id='i2'>

</div>

{% endif %}

</div>

</body>

</html>

# Static Files:

a.navbar.cssb.index.css

c.form.css

c.result.css

## a.navbar.css

\*{

padding: 0px;

margin: 0px;

box-sizing: border-box;

}

body{

width: 100%;

height: 100vh;

}

.navbar{

width: 100%;

height: 80px;

background-color:rgb(8, 106, 205);

box-shadow: 1px 2px 4px black;

display: flex;

justify-content: center;

align-items: center;

}

.navbar h2{

color: #fff;

}

.navbar .list{

position: absolute;

float: right;

margin-left: 73%;

margin-top: 5%;

width: fit-content;

}

.navbar .list ul{

display: flex;

flex-direction: row;

gap: 2px;

list-style-type: none;

}

.navbar .list ul li{

margin: 3px;

color: #fff;

}

.navbar .list ul li a{

color: #fff;

text-decoration: none;

padding: 2px 6px;

}

.navbar .list ul li a:hover{

background-color: #fff;

border-radius: 8px;

color: rgb(8, 106, 205);

}

.body{

margin-top: 3px;

width: 100%;

height: 84%;

display: flex;

flex-direction: row;

}

::-webkit-scrollbar {

width: 0px;

height: 10px;

}

/\* Track \*/

::-webkit-scrollbar-track {

background: #f1f1f1;

}

/\* Handle \*/

::-webkit-scrollbar-thumb {

background: #888;

}

/\* Handle on hover \*/

::-webkit-scrollbar-thumb:hover {

background: #555;

}

## b.index.css

## .body .data{

## width: fit-content;

## height: 100%;

## overflow-y:scroll;

## padding: 30px;

## padding-top: 50px;

## }

## .body .data button{

## background-color: rgba(8, 107, 205, 0.812);

## padding:5px 15px;

## border-radius: 10px;

## border: none;

## box-shadow: 2px 1px 1px black;

## }

## .body .data button a{

## color: #fff;

## text-decoration: none;

## }

## .body .data button:hover{

## background-color: rgb(8, 106, 205);;

## box-shadow: 2px 3px 3px black;

## }

## .body .pic{

## width: 50%;

## height: 100%;

## overflow: hidden;

## z-index: -1;

## position: fixed;

## margin-left: 50%;

## }

## .body .pic img{

## margin-left:0px;

## margin-top: 6%;

## }

## h1{

## color: rgb(8, 106, 205);;

## }

## ::-webkit-scrollbar {

## width: 0px;

## height: 10px;

## }

## 

## /\* Track \*/

## ::-webkit-scrollbar-track {

## background: #f1f1f1;

## }

## 

## /\* Handle \*/

## ::-webkit-scrollbar-thumb {

## background: #888;

## }

## 

## /\* Handle on hover \*/

## ::-webkit-scrollbar-thumb:hover {

## background: #555;

## }

## c.form.css

## .body{

## 

## display: flex;

## justify-content: center;

## padding-top: 20px;

## height: 84%;

## overflow-y: hidden;

## }

## .body img:first-child{

## margin-top: 220px;

## margin-left: -15px;

## margin-right: 5px;

## width: 230px;

## height: 200px;

## 

## }

## #i3{

## margin-top: 190px;

## margin-left: 10px;

## width: 270px;

## height: 250px;

## z-index: -1;

## }

## .form{

## width: 60%;

## height:100%;

## border-radius: 15px;

## border-bottom-left-radius: 0px;

## border-bottom-right-radius: 0px;

## box-shadow: 0 0 10px rgba(0, 0, 0, 0.497);

## padding: 20px;

## padding-bottom: 0px;

## display: flex;

## flex-direction: column;

## }

## .form h3{

## background-color: rgb(8, 106, 205);;

## color: #fff;

## border-radius: 10px;

## width: 100%;

## text-align: center;

## padding: 5px 10px;

## margin-bottom: 15px;

## }

## .form form{

## display: flex;

## flex-direction: column;

## align-items: center;

## height: 98%;

## overflow-y: scroll;

## padding-left: 20px;

## padding-bottom: 15px;

## font-size: small;

## }

## .form form .i{

## margin: 5px;

## padding: 3px;

## width: 95%;

## height: 35px;

## display: flex;

## flex-direction: row;

## align-items: center;

## 

## }

## .form form .i .ii{

## width:150px;

## background-color:rgb(8, 106, 205);;

## color: #fff;

## margin: 3px;

## padding: 3px;

## display: flex;

## align-items: center;

## justify-content: center;

## border-radius: 10px;

## font-size: small;

## 

## }

## input,select{

## font-size: smaller;

## cursor: pointer;

## }

## input:not([type="radio"]),select {

## background-color: #f5f5f5;

## border: none;

## border-radius: 5px;

## padding: 4px;

## box-shadow: inset 0 2px 4px rgba(0, 0, 0, 0.1), 0 4px 5px rgba(0, 0, 0, 0.1);

## outline: none;

## width: 280px;

## 

## }

## 

## input:not([type="radio"]):focus,select:focus{

## box-shadow: inset 0 2px 4px rgba(0, 0, 255, 0.2), 0 4px 5px rgba(0, 0, 255, 0.1);

## }

## #buts{

## justify-content: space-around;

## }

## #buts input{

## width: 150px;

## border-radius: 8px;

## padding: 8px 10px;

## font-size: small;

## color: #fff;

## margin: -5px;

## margin-top: 10px;

## 

## }

## #buts input:hover{

## box-shadow: 2px 3px 3px black;

## }

## #buts input:first-child{

## background-color: green;

## }

## #buts input:last-child{

## background-color: red;

## }

## #red{

## color: red;

## }

## 

## c.result.css

## .yes,.no{

## width: 100%;

## height: 100%;

## display: flex;

## flex-direction: row;

## align-items: center;

## justify-content: center;

## overflow: hidden;

## background-color: rgb(248, 246, 246);

## background-color: rgb(247, 247, 246);

## }

## .no{

## background-color: #fff;

## color: red;

## }

## .yes img{

## width: 350px;

## height: 300px;

## position: relative;

## background-color: red;

## }

## #i1{

## margin-left: -40px;

## margin-right: 5px;

## }

## #i2{

## margin-left: 5px;

## margin-right: -40px;

## }

## .data{

## width: 500px;

## height: 300px;

## 

## display: flex;

## flex-direction: column;

## justify-content: center;

## align-items: center;

## color: green;

## }

## #fail{

## color: red;

## }

## button{

## font-size: smaller;

## cursor: pointer;

## width: 150px;

## margin-top: 30px;

## padding: 6px 8px;

## border: none;

## border-radius: 10px;

## background-color:rgba(8, 107, 205, 0.812);

## color: #fff;

## outline: none;

## box-shadow:2px 3px 3px black;

## }

## button a{

## text-decoration: none;

## color: #fff;

## 

## }

## button a:hover{

## text-decoration: underline;

## 

## }

## button:hover {

## 

## border: none;

## padding: 6px 10px;

## border-radius: 11px;

## background-color:rgba(8, 107, 205, 0.945);

## 

## }

# app.py

from flask import Flask, render\_template, request

import numpy as np

import joblib

import pandas as pd

app = Flask(\_\_name\_\_)

@app.route('/')

def hello():

return render\_template('/index.html')

def fun(x):

l=[]

l.append(float(np.sqrt(float(x[0]))))

l.append(float(np.sqrt(float(x[1]))))

l.append(x[2])

l+=x[3:]

d = pd.DataFrame([l], columns=['ApplicantIncome', 'LoanAmount', 'Married', 'Property\_Area\_Rural', 'Property\_Area\_Semiurban', 'Property\_Area\_Urban'])

return m.predict(d)[0]

@app.route('/form', methods=['GET','POST'])

def form():

if request.method == 'POST':

id = request.form['id'].strip()

name=request.form['name'].strip()

gender=request.form['gender'].strip()

married=request.form['married'].strip()

dependents=request.form['dependents'].strip()

education=request.form['education'].strip()

semployed=request.form['semployed'].strip()

aincome=request.form['aincome'].strip()

coincome=request.form['coincome'].strip()

lamount=request.form['lamount'].strip()

lterm=request.form['lterm'].strip()

chistory=request.form['chistory'].strip()

parea=request.form['parea'].strip()

print(id,name,gender,married,dependents,education,semployed,aincome,coincome,lamount,lterm,chistory,parea)

if id=='':

return '<script>alert("Please Enter a Valid Loan ID"); window.history.back();</script>'

elif name=='':

return '<script>alert("Please Enter a Valid Name"); window.history.back();</script>'

elif gender=='':

return '<script>alert("Please Choose Your Gender"); window.history.back();</script>'

elif married=='':

return '<script>alert("Please Choose Your Marrital Status"); window.history.back();</script>'

elif dependents=='':

return '<script>alert("Please Enter Valid Dependents"); window.history.back();</script>'

elif education=='':

return '<script>alert("Please Choose Your Education Status"); window.history.back();</script>'

elif semployed=='':

return '<script>alert("Please Choose Your Employment Status"); window.history.back();</script>'

elif aincome=='':

return '<script>alert("Please Enter valid Applicant Income"); window.history.back();</script>'

elif coincome=='':

return '<script>alert("Please Enter Valid CoApplicant Income"); window.history.back();</script>'

elif lamount=='':

return '<script>alert("Please Enter a Valid Loan Amount"); window.history.back();</script>'

elif lterm=='':

return '<script>alert("Please Enter a Valid Loan Term"); window.history.back();</script>'

elif chistory=='':

return '<script>alert("Please Choose Your Credit History"); window.history.back();</script>'

elif parea=='':

return '<script>alert("Please Choose Your Property Area"); window.history.back();</script>'

else:

married=int(married)

r=int(parea=='rural')

s=int(parea=='semiurban')

u=int(parea=='urban')

l=[aincome,lamount,married,r,s,u]

x=fun(l)

return result(x,id,name)

else:

return render\_template('/form.html')

@app.route('/result')

def result(x,id,name):

return render\_template('result.html',x=x,id=id,name=name.capitalize())

if \_\_name\_\_ == '\_\_main\_\_':

m=joblib.load('mymodel.pkl')

app.run(debug=True)

# RESULT ANALYSIS

* **Confusion matrix:**

A confusion matrix is a table that is often used to describe the performance of a classification model (or "classifier") on a set of test data for which the true values are known as true positive (tp) is a result where the model predicts the positive class correctly. Similarly, a true negative (tn) is an outcome where the model correctly predicts the negative class. A false positive (fp) is an outcome where the model incorrectly predicts the positive class. And a false negative (fn) is an outcome where the model incorrectly predicts the negative class.

## Sensitivity or recall or hit rate or true positive rate (TPR):

## It is the proportion of individuals who actually have the disease were identified as having the disease. TPR = tp / (tp + fn)

## Specificity, selectivity or true negative rate (TNR):

## It is the proportion of individuals who actually do not have the disease were identified as not having the disease.

## TNR = tn / (tn + fp) =1-FPR

## Precision or positive predictive value (PPV):

## If the test result is positive what is the probability that the patient actually has the disease. PPV = tp / (tp + fp)

## Accuracy:

## The accuracy reflects the total proportion of individuals that are correctly classified [14] ACC = (tp + tn) / (tp + tn + fp + fn)

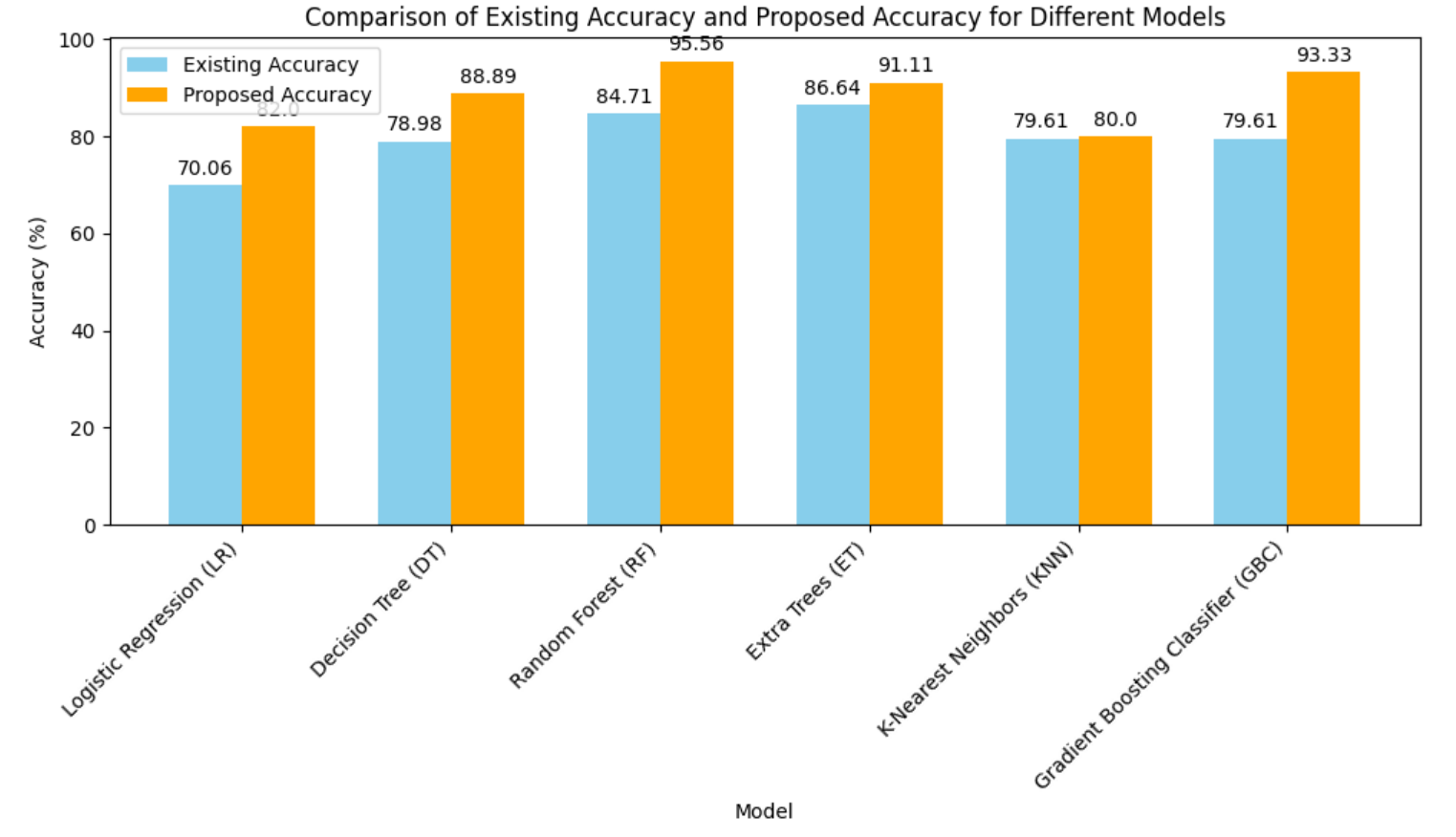
## F1 score:

## It is the harmonic mean of precision and sensitivity F1 = 2tp / (2tp+ fp + fn)

Among the diverse set of models utilized for predicting loan acceptance, Random Forest emerged as the standout performer, boasting an impressive accuracy rate of 95.56%. This model exhibited remarkable resilience across both the training and testing phases, achieving accuracies of 98.325% and 95.556%, respectively. Following closely behind, the Extra Trees Classifier showcased robust predictive capabilities, achieving an accuracy of 91.11%. Gradient Boosting also demonstrated its effectiveness with a commendable accuracy of 93.33%. However, other models such as Decision Tree, K-Nearest Neighbors, and Logistic Regression yielded lower accuracies of 88.89%, 80.00%, and 82.22%, respectively. These findings, depicted in [Fig 5.1](#fcompare), underscore the superiority of Random Forest in accurately forecasting loan acceptance, highlighting its resilience and reliability in the context of predictive modeling. Using The notable performance disparities among the various models emphasized in [table 1](#table), the importance of selecting appropriate algorithms tailored to specific predictive tasks, ultimately optimizing model accuracy and efficacy in real-world applications.

**Table-1** Metric Values for all Models

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| S. No | Model Name | Sensitivity | Specificity | F1 Score | Precision | Accuracy  (%) |
| 1 | Random Forest | 0.96 | 0.95 | 0.96 | 0.96 | 95.56 |
| 2 | GradientBoost | 0.92 | 0.95 | 0.93 | 0.96 | 93.33 |
| 3 | Extra Trees | 0.96 | 0.85 | 0.92 | 0.89 | 91.11 |
| 4 | Decision Tree | 0.92 | 0.85 | 0.92 | 0.89 | 88.89 |
| 5 | Logistic Regression | 0.76 | 0.90 | 0.83 | 0.90 | 82.20 |
| 6 | KNN Classifier | 0.40 | 0.85 | 0.53 | 0.77 | 80.0 |

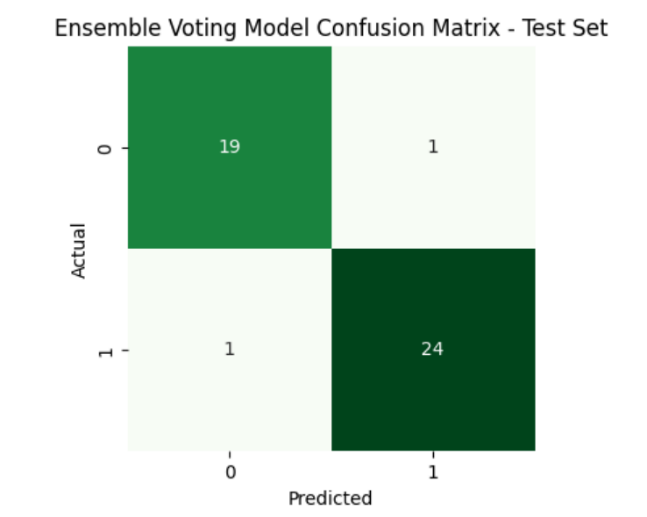
**Fig 5.1** Comparing Existing and Proposed Accuracies****

**Voting Ensemble Model**

In order to increase the overall accuracy, we will integrate the predictions of several machine learning models in our next stage, which is the implementation of a voting ensemble approach.

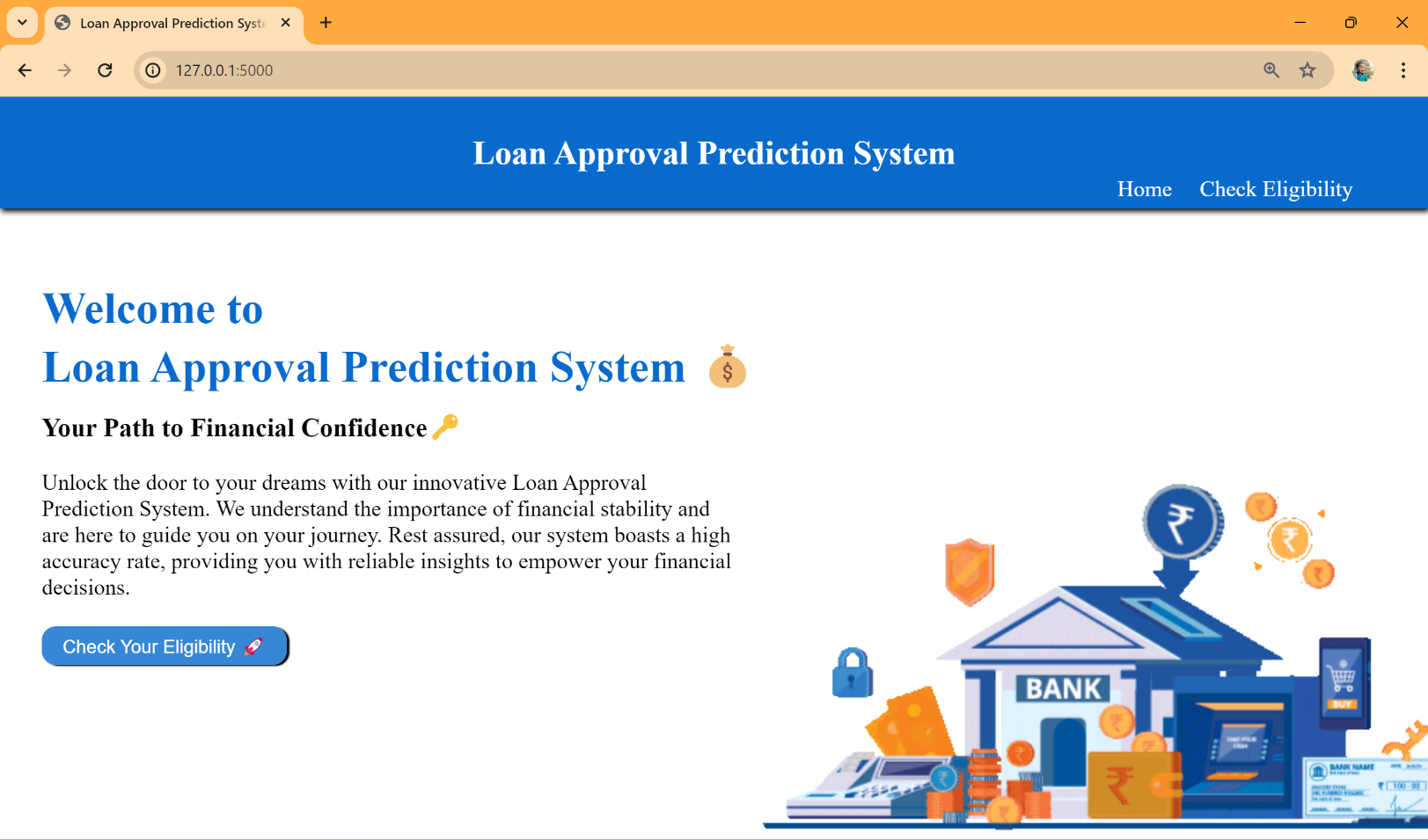
After careful examination, Random Forest was clearly the best performer, with an astounding accuracy of 95.56%. In training and testing, this model demonstrated exceptional robustness, with accuracy values of 98.325% and 95.556%, respectively. With an accuracy of 91.11%, the Extra Trees Classifier demonstrated noteworthy predictive capability, trailing closely behind. Likewise, Gradient Boosting demonstrated its efficacy with a 93.33% accuracy rate.

By combining Random Forest, Extra Trees Classifier, and Gradient Boosting, we will use ensemble learning to take advantage of the advantages of these high-performing models. In order to improve the accuracy of loan approval forecasts, this strategy seeks to use the combined predictive potential of models. Using the best-performing models and the ensemble approach, we were able to obtain an astounding accuracy of 95.55%. This suggests a notable improvement in the accuracy of loan approval forecasts.

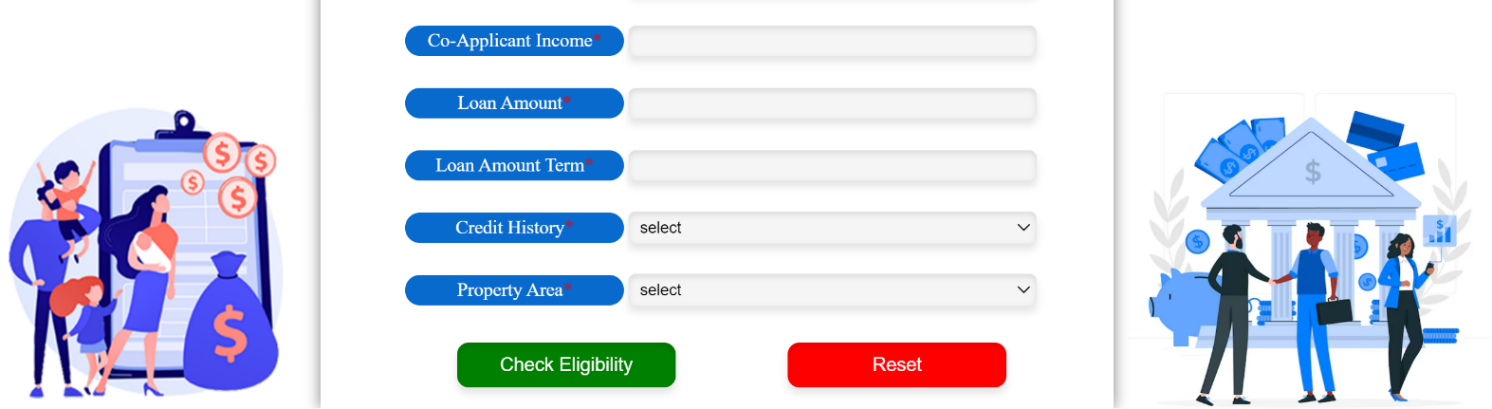
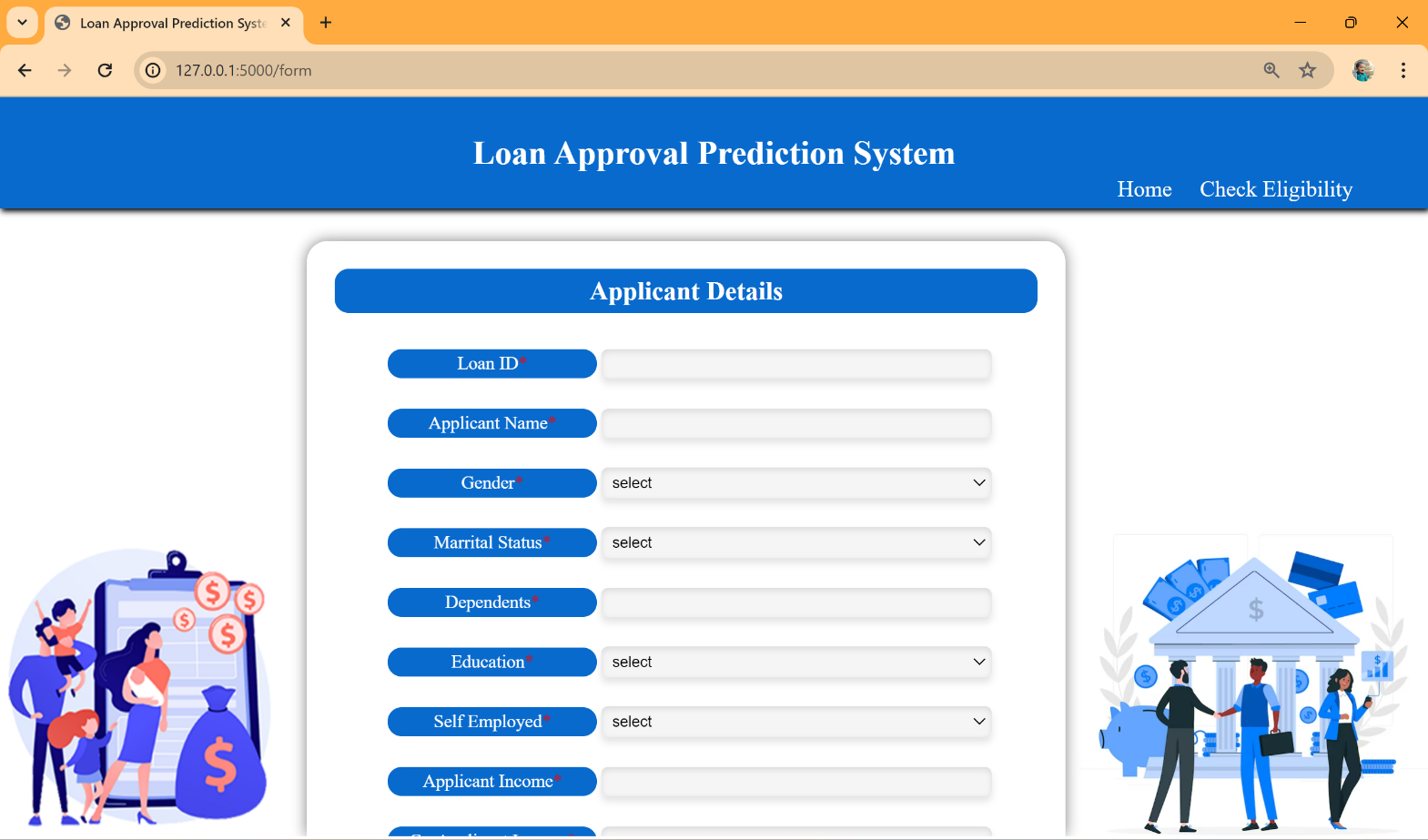
The confusion matrix's [fig 5.2](#fconf), highlighted values suggest that the model is performing satisfactorily in classifying loan applications. The accuracy is 94% with 24 true positives and 19 genuine negatives. With a precision of 95% for loans that have been granted, there is a high percentage of accurate forecasts among loans with this approval status. Furthermore, for loans that are authorized, recall is 100% perfect, providing there are no false negatives. Four false positives, on the other hand, show that some loans were incorrectly identified as approved despite being refused.

**Fig 5.2** Confusion Matrix of Voting Ensemble Model

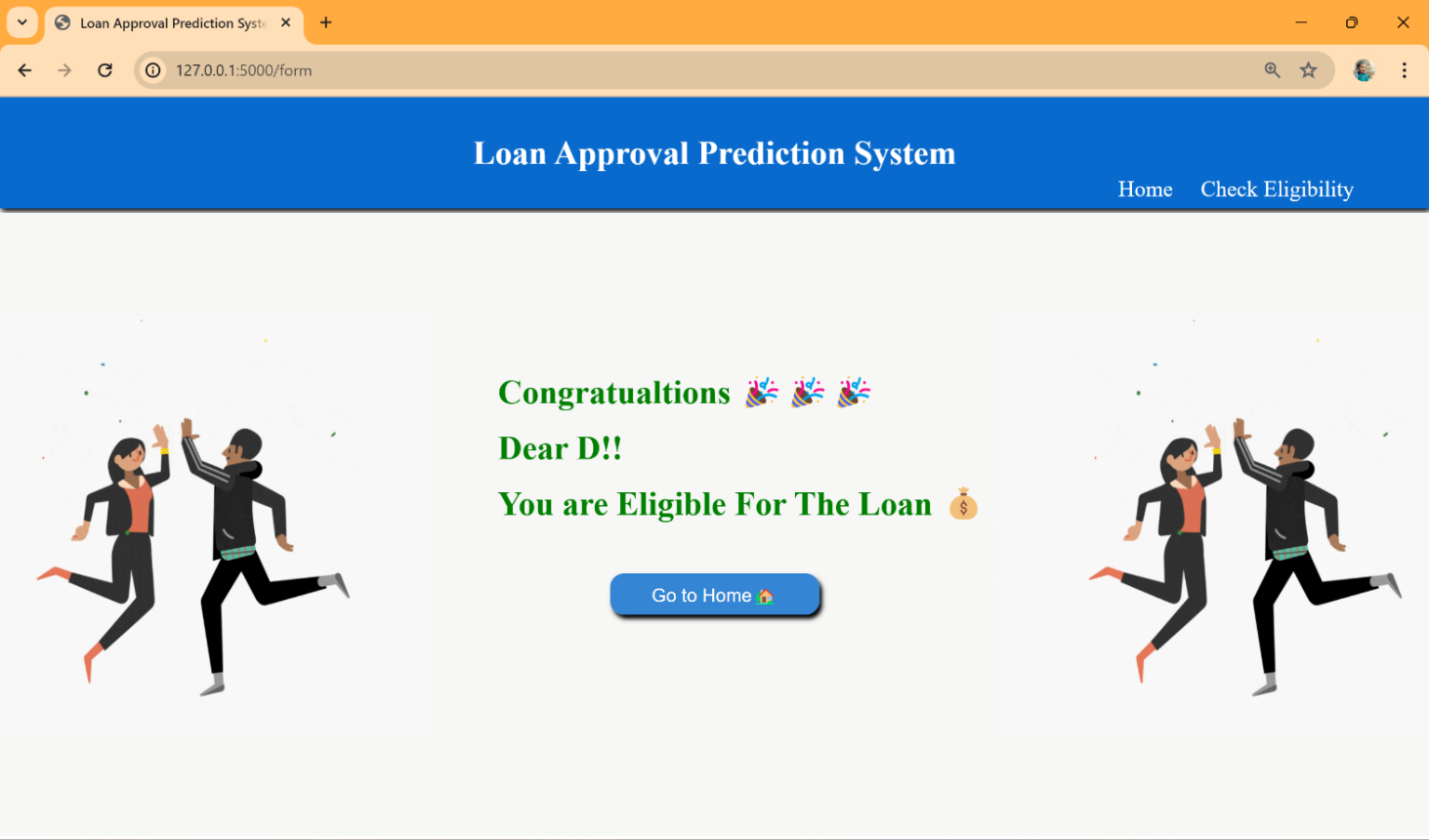
# OUTPUT SCREENS



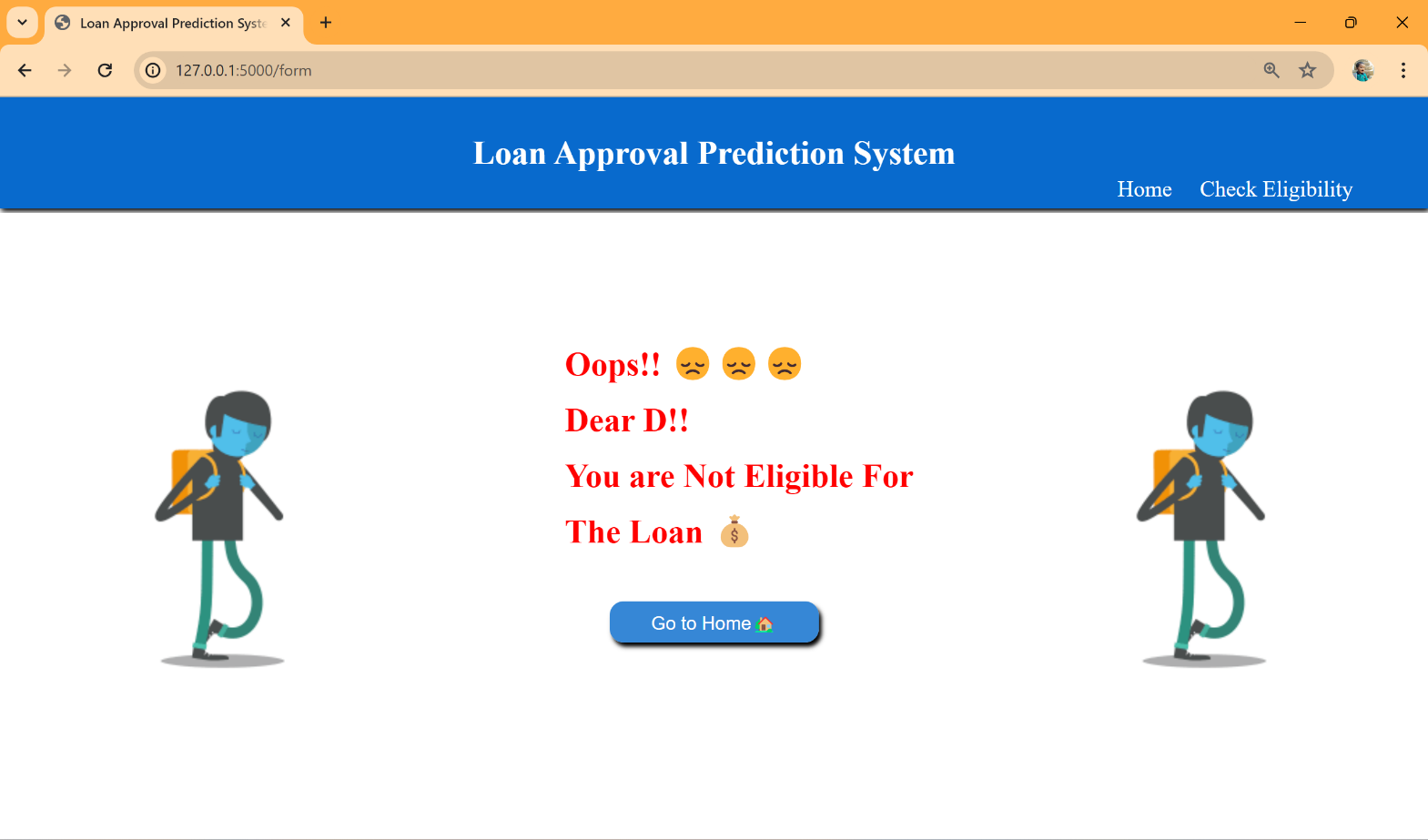
**Fig 6.1** Home Page



**Fig 6.2** Form Page



**Fig 6.3** Approval Status Page



**Fig 6.4** Disapproval Status Page

**7.** **CONCLUSION**

In summary, our endeavor has illuminated the remarkable impact of machine learning (ML) methodologies on revolutionizing the loan approval prediction process. Through meticulous examination of our dataset and the application of sophisticated preprocessing techniques—ranging from null value management to outlier handling, square root transformation, correlation analysis, feature selection, and SMOTE—we have laid a robust foundation for our predictive models.

Moving forward, our model has achieved an extraordinary accuracy of 95.56%, a testament to the unparalleled precision afforded by ML techniques. This exceptional performance marks a significant advancement beyond the baseline accuracy of 87.26% documented in previous studies, underscoring the substantial progress facilitated by our rigorous methodology.

By leveraging the power of ML, we have not only enhanced the accuracy of loan approval predictions but also demonstrated the transformative potential of data-driven approaches in the financial domain. Our findings underscore the importance of adopting advanced analytical techniques to optimize decision-making processes, ultimately paving the way for more efficient informed and effective lending practices.

# FUTURE SCOPE

Our research opens up a world of opportunities for improving loan approval prediction models in the future. Subsequent investigations may explore further into the field of deep learning, utilizing neural networks capacity to address complex patterns in loan data. In addition, to guarantee flexibility in response to changing patterns and dynamics in loan application and repayment behavior, ongoing model monitoring systems will be necessary. We are steadfast in our research to advance ML innovation and bring about revolutionary change in the field of financial decision-making processes as we move forward.

**Implementation of Personal Loan Prediction Models:** Future research can focus on developing specialized prediction models tailored to specific loan types, such as personal loans, home loans, or automotive loans. By customizing models to individual loan categories, financial institutions can enhance accuracy and tailor decision-making processes to meet the unique needs of borrowers in various contexts.

**Integration of Deep Learning Techniques:** Expanding upon traditional machine learning approaches, future investigations may explore the integration of deep learning techniques to analyze complex patterns and relationships within loan data. Neural networks offer the potential to uncover hidden insights and improve prediction accuracy by capturing intricate dependencies among diverse features.

**Incorporation of Alternative Data Sources:** Future research endeavors may explore the incorporation of alternative data sources, such as social media activity, transactional data, or geospatial information, to enrich existing loan approval prediction models. By leveraging diverse datasets, researchers can gain deeper insights into borrower behavior and improve the accuracy of credit risk assessments, ultimately enhancing the decision-making process for lenders.

These future scopes hold the potential to revolutionize the field of loan approval prediction, enabling financial institutions to make more informed and data-driven decisions while providing borrowers with personalized and equitable lending opportunities.

# REFERENCES

* + 1. Nitesh Pandey et al. (2022). "Loan Approval Prediction using Machine Learning Algorithms Approach.", IRJMETS, this paper achieved an accuracy of 78.3% using Logistic Regression, Decision Trees, and KNN. It discusses limitations in feature engineering and imbalanced datasets.
    2. J. Tejaswini (2022). "Accurate Loan Approval Prediction Based on Machine Learning Approach.", IRJMETS, this paper achieved an accuracy of 88.1% using XGBoost. It emphasizes the importance of careful data pre-processing and feature engineering for optimal performance.
    3. Anant Shinde et al. (2022). "Loan Prediction System Using Machine Learning." Published in the field of Finance by IEEE, this paper compares Logistic Regression, Decision Tree, and Random Forest models. It concludes that Decision Tree and Random Forest outperform Logistic Regression due to their non-linearity handling capabilities.
    4. Dharavath Sai Kiran. Avula et al. (2023). "Loan Approval Prediction using Adversarial Training and Data Science." Within the domain of Finance, this paper discusses a model that produced low accuracy values and could only handle minimum-sized data.
    5. Zhu L, Qiu D, Ergu D, Ying C, Liu K (2019) A study on predicting loan default based on the random forest algorithm. Procedia Comput Sci 162:503–513.
    6. Nigmonov A, Shams S, Alam K (2022) Research Macroeconomic determinants of loan defaults: evidence from the U.S. peer-to-peer lending market. Res Int Bus Finan 59:101516.
    7. Lee JW, Lee WK, Sohn SY (2021) Graph convolutional network-based credit default prediction utilizing three types of virtual distances among borrowers. Expert Syst Appl 168:114411.
    8. Lim S-J, Thiel C, Sehm B, Deserno L, Lepsien J, Obleser J (2022) Distributed networks for auditory memory differentially contribute to recall precision. NeuroImage 256:119227.
    9. Fontem B, Smith J (2019) Analysis of a chance-constrained new product risk model with multiple customer classes. Eur J Oper Res 272(3):999–1016 23. Bianco S, Mazzini D, Napoletano P, Schettin R (2019) Multitask painting categorization by deep multibranch neural network. Expert Syst Appl 135:90–101.
    10. Wang L, Chen Y, Jiang H, Yao J (2020) Imbalanced credit risk evaluation based on multiple sampling, multiple kernel fuzzy self-organizing map and local accuracy ensemble. Appl Soft Comput 91:106262
    11. Vuttipittayamongkol P, Elyan E, Petrovski A (2021) On the class overlap problem in imbalanced data classification. Knowl-Based Syst 212:106631 26. Papouskova M, Hajek P (2019) Two stage consumer credit risk modelling using heterogeneous ensemble learning. Decis Support Syst 118:33–45.
    12. Gopinath, Mahankali, K. Srinivas Shankar Maheep, and R. Sethuraman. 2021. “Customer Loan Approval Prediction Using Logistic Regression.” Advances in Parallel Computing.
    13. Pidikiti Supriya, Myneedi Pavani, Nagarapu Saisushma,Namburi Vimala Kumari, k Vikash,“Loan Prediction by using Machine Learning Models”, International Journal of Engineering and Techniques.Volume 5 Issue 2, Mar-Apr 2019.
    14. A. Uzair, T. Aziz, H. Ilyas, S. Asim, B. N. Kadhar, "An Empirical Study on Loan Default Prediction Models" Journal of Computational and Theoretical Nanoscience, Volume 16, Number 8, August 2019, pp. 3483-3488(6).
    15. M. Madaan et al. "Loan default prediction using decision trees and random forest: A comparative study" IOP Conf. Ser.: Mater. Sci. Eng. 2014.
    16. Alsaleem, M. Y., & Hasoon, S. O. (2020). Predicting bank loan risks using machine learning algorithms. AL-Rafidain J. Comput. Sci. Math., 14(1), 149–158.
    17. Ramachandra, H. V., G. Balaraju, R. Divyashree, and Harish Patil. 2021. “Design and Simulation of Loan Approval Prediction Model Using AWS Platform.” 2021 International Conference on Emerging Smart Computing and Informatics (ESCI).
    18. Singh, Vishal, Ayushman Yadav, Rajat Awasthi, and Guide N. Partheeban. 2021. “Prediction ofModernized Loan Approval System Based on MachineLearning Approach.” 2021 International Conference on IntelligentTechnologies(CONIT).
    19. S. M. Karthikeyan, and Pushpa Ravikumar. 2021. “A Comparative Analysis of Feature Selection for Loan Prediction Model.” InternationalJournalofComputerApplications.
    20. Hassan, Amira Kamil Ibrahim and Ajith Abraham. "Modeling consumer loan default prediction using ensemble neural networks", 2013 International Conference On Computing, Electrical And Electronic Engineering (ICCEEE). IEEE, 2013.
    21. N. Uddin, M.K. Uddin Ahamed, M.A. Uddin, Md. Manwarul Islam, Md. Alamin Talukder, and Sunil Aryal, "An ensemble machine learning based bank loan approval predictions system with a smart application," International Journal of Cognitive Computing in Engineering, vol. 4, pp. 327-339, 2023.

Loan Approval Prediction using Machine Learning

|  |  |  |
| --- | --- | --- |
| Dr. Rama Krishna Eluri  Assistant Professor  Dept. of Computer Science and Engineering,  Narasaraopeta Engineering College  Narasaraopet, Andhra Pradesh,  India  ramakrishnaephd7@gmail.com | Battu Venkata Siva  Student  Dept. of Computer Science and Engineering  Narasaraopeta Engineering College  Narasaraopet, Andhra Pradesh,  India  [venkatasivabattu2002@gmail.com](mailto:venkatasivabattu2002@gmail.com) | Shaik Aariz Ahmed  Student  Dept. of Computer Science and Engineering  Narasaraopeta Engineering College  Narasaraopet, Andhra Pradesh, India  [aarizahmed2002@gmail.com](mailto:aarizahmed2002@gmail.com) |

*ABSTRACT-Loans are a crucial part of the modern world, and banks receive a significant portion of their profits from them. However, deciding whether to grant a loan to an applicant is a complex process that requires banks to consider many factors.*

*In this study, we suggest a machine learning-based method to streamline the loan acceptance prediction process. To determine whether or not a loan applicant's profile is relevant for approval, we employ effective machine learning algorithms. We base our predictions on important features. Additionally, we present a comparison study of various categorization methods to demonstrate how machine learning algorithms might enhance the loan approval procedure. Our results show that machine learning algorithms can significantly reduce the risk of loan defaults and improve the loan approval process. Moreover, to enhance prediction accuracy, we incorporate a voting ensemble technique into our methodology. This additional layer of analysis further refines our predictions, contributing to more reliable loan approval decisions.*

*KEYWORDS--Loan approval prediction, Loan default risk, Predictive modeling, Feature selection, SMOTE, Decision Tree,, Gradient Boost, Random Forest, Extra Tree, Comparative analysis, Voting Ensemble.*

# INTRODUCTION

One of the main functions of the banking sector is lending, which is crucial to maintaining the financial stability of a country. Banks receive a large amount of their interest revenuefrom loans, thus the loan approval procedure is quite important. Individual bank managers evaluate the risk of loan default and application eligibility, which might cause systemic disruptions that could affect the economy as a whole as well as possible financial losses for banks.

In this research, we use data-driven strategies to streamline the loan approval process. We start with the removal of null values [4], duplicates, and outliers from the data [2], cleaning it up, and examining any relationships between the variables. The purpose of feature selection is to find the most pertinent qualities. To address the disparity in class sizes, we employ the SMOTE [22]. We then use machine learning methods to forecast the results of loan approval, such as gradient boost, decision tree, random forest, and other techniques. Each algorithm's performance is assessed using various metrics. We also examine how each algorithm contributes to enhancing the loan approval procedure. Since the primary objective of this study is to employ data-driven methodologies to enhance the precision and effectiveness of the loan approval procedure, We do this with the implementation voting ensemble method.

# 2. LITERATURE SURVEY

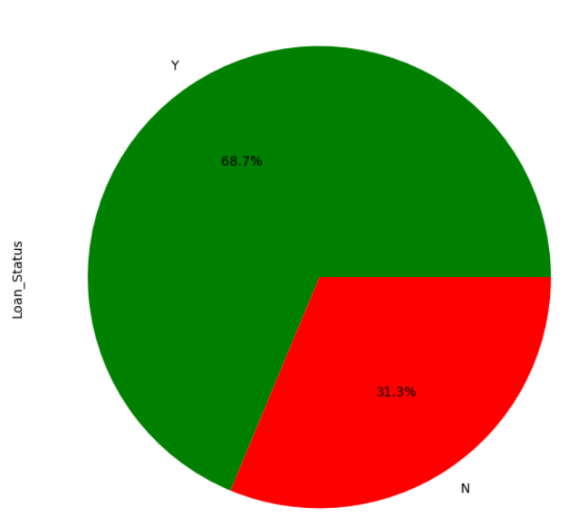
Manjeet Kumar et al. [3] evaluated a number of classifiers, such as Light Gradient Boosting Machine (LGBM), Extra Trees, Random Forest, and Extreme Gradient Boosting (XGB) for the purpose of predicting bank loan default. Their study provides insightful information for financial institutions by highlighting the significance of debt income and work history in forecasting defaults

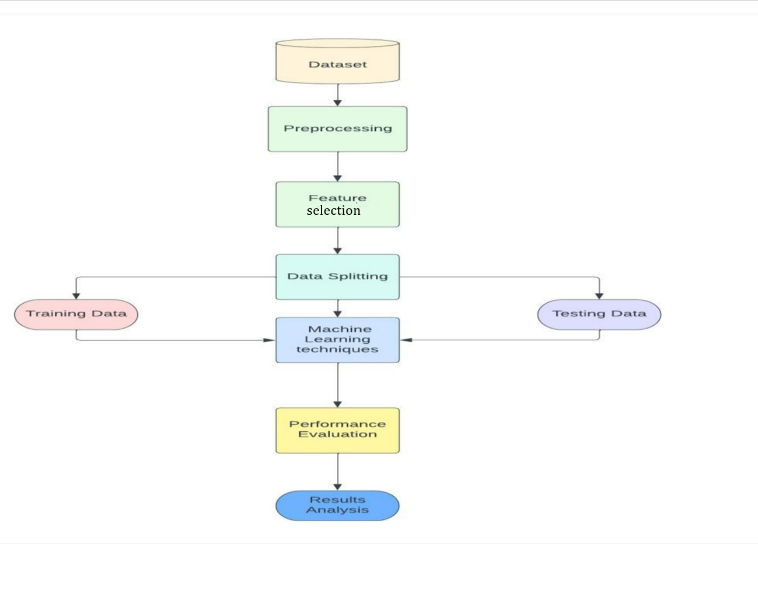
In the research, Mehul Madaan et al. [4] obtained 73% and 80% accuracy, respectively, in loan default prediction using Decision Trees and Random Forest algorithms. For financial organisations looking to enhance loan approval procedures and reduce credit risks, their study provides insightful information. Through the analysis of these algorithms' performance on a shared dataset, the research adds to the continuing investigation of machine learning applications in the banking industry.

Decision Trees and other machine learning models were used by Supriya et al. [2] to predict loan defaulters with an accuracy of 81.1%. Their research focused on data preparation methods, such as managing outliers and missing information, and resulted in a thorough examination of the characteristics that affect loan acceptance. The writers provided insightful information for improving credit risk assessment in the banking industry by highlighting the importance of variables like income level and credit history in loan sanctioning decisions.

Mahankali et al. [1] forecast loan approvals with an accuracy rate of 80.945% by using logistic regression. Their all-inclusive strategy comprises testing, model creation, and data pretreatment, offering a solid foundation for automated loan approval systems. This study provides useful information about the use of machine learning algorithms in banking settings and sets the standard for further research in the area. When comparing machine learning algorithms for forecasting bank loan risks, Alsaleem et al. [5] discovered that Multilayer Perceptron has the best accuracy (80%).

# 3. THE PROPOSED SYSTEM FOR LOAN APPROVAL PREDICTION

****Our proposed model for loan approval prediction is structured around distinct phases and steps as shown in Fig.1 , each tailored to maximize accuracy, efficiency, interpretability, robustness, and fairness. The criteria as follows:

* Analysis of Dataset
* Visualization of Data
* Preprocessing Techniques
* Model Development

# Fig.1 Flow Chart

# *A. Analysis of Dataset*

Fig.2: Data Set

# We used the well-known Kaggle website, which hosts a variety of datasets, to find the datasets we needed for our predictive study. The dataset that we used is accessible [6] For our research to be successful, having access to these datasets shown in Fig.2 is essential for performing in-depth analysis and forecasts. The dataset consists of 13 columns and 614 entries.

# *B. Visualization of Data*

# As the target class to be predicted, the "Loan\_Status" column shows a class imbalance in the Fig.3, with roughly 68.7% of the entries labelled as "Y" (meaning loan approval) and the remaining 31.3% labelled as "N" (showing loan denial). The performance of several models, especially those that are sensitive to class distribution, may be impacted by this imbalance.

Fig3. Loan\_Status

# *C.* *Preprocessing Techniques*

The next stage after data visualisation is preprocessing, which gets the data ready for model training. The following are the steps.

## *Dealing Null Values:*

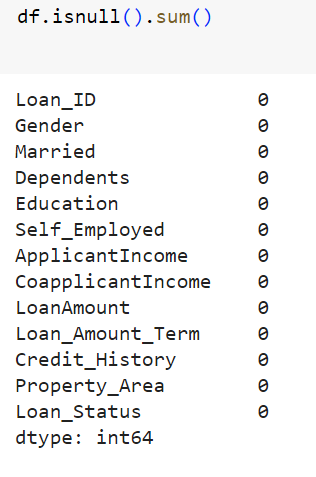
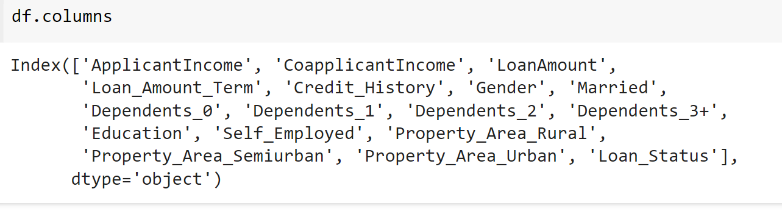
Reliability of the model and data quality depend on addressing null values [4]. By managing null values using methods like imputation or elimination, machine learning models become more accurate and the dataset's integrity is preserved. Potential biases are reduced by efficient null value management, allowing for more precise forecasts and perceptive analysis. The fig.4 illustrates how frequently data

Fig.4 Representing frequency of Null Values

are missing in various columns both before and after removal. It exposes deficiencies in important characteristics such as Gender, Dependents, Self\_Employed, and Credit\_History, for example.

* *One-hot Encoding:*

To ensure compatibility with numerical computations, one-hot encoding is essential for transforming categorical variables into a format that is appropriate for machine learning methods. Algorithms can successfully read and learn from categorical features by expressing categorical data as binary vectors. By doing this, the feature space is increased, maintaining the distinctive qualities of every category and improving.



## Fig.5 Columns after One-hot encoding

## *Eliminating Data Outliers*

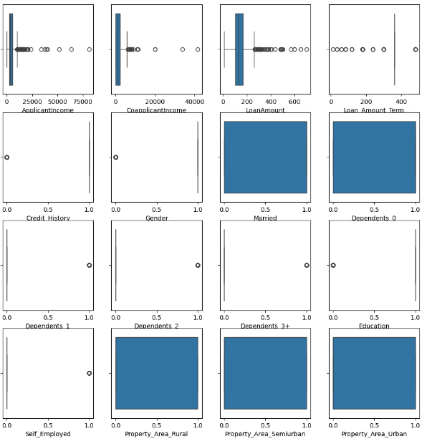
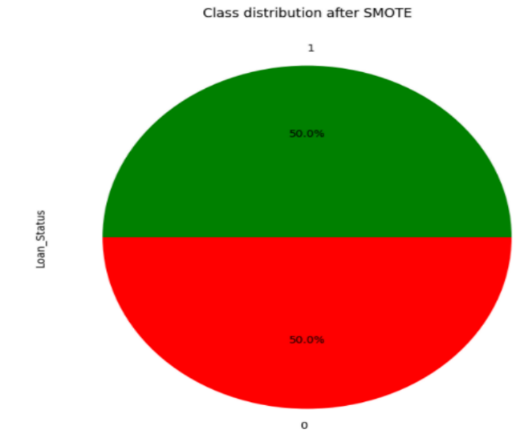


Fig 6: Visualizing data outliers

Handling outliers is essential to preserving the accuracy and integrity of machine learning models and statistical analysis [2]. Results can be severely skewed by outliers, which can also influence interpretations and provide false findings. Fig.6 Illustrating the presence of Outliers in the data. Since, the dataset had outliers that could have distorted the results of our investigation and the performance of the model. We utilised the quantile approach to solve this problem, identifying and eliminating extreme values by establishing thresholds based on quantiles.

* ***Square Root Transformation:***

## Because it lessens the impact of extreme values, square root transformation is essential for stabilising variance, particularly in datasets with skewed distributions. It works by improving the symmetry and conformance of the data to normalcy assumptions, which can guarantee reliable inference and enhance the performance of specific statistical models.

## *Feature Selection:*

Feature selection is an essential process that helps machine learning models operate at their best by locating the most significant predictors for the target variable. We employ Recursive Feature Elimination (RFE) in our method, which is an iterative process of selecting features according to their importance ratings. The impact of each feature to the loan approval prediction job is carefully evaluated by integrating RFE with a logistic regression estimator. We are able to determine a subset of features shown in Fig 8, that minimises model complexity and maximises predicted accuracy through this iterative method.

Fig 7:selected features after RFE

## *SMOTE:*

When one class predominates over another in a situation like predicting loan acceptance, SMOTE (Synthetic Minority Over-sampling Technique) is an essential technique for resolving class imbalance in datasets[22]. As in Fig 9, the class imbalance bias is successfully reduced by using SMOTE, resulting in a more representative and trustworthy dataset that we can use to train our machine learning models. By using this method, the model's capacity to generalise across both groups is improved, which eventually leads to forecasts of loan acceptance that are more fair and accurate.

## Fig 8:Loan\_Status after SMOTE

# *D. Model Development*

To ensure their resilience and usefulness in a range of situations, machine learning models are constructed and evaluated using training datasets. The research employs the following methods: Random Forest, Decision Tree, Gradient Boosting, K-Nearest Neighbours (KNN), Extra Trees, and Logistic Regression. We use the implementation voting ensemble approach to improve the accuracy and efficiency of the loan approval process.

* Voting Ensemble Method:

A machine learning method called voting ensemble combines several models to provide predictions. A majority vote or an average is used to decide the final forecast, which is based on the weighted predictions of each model. When compared to separate models, it frequently results in increased accuracy and resilience.

y^​=argmaxj​i=1∑n​1(y^​i​=j) (1)

# 4. RESULTS & DISCUSSION

We found that the models that were used are Random Forest, Extra Trees, Decision Trees, Gradient Boosting, Logistic Regression, and K-Nearest Neighbors performed differently. Every algorithm performed differently, as shown in Fig 9 . In order to highlight, With an accuracy of 95.56%, Random Forest stood out among the group of models employed to forecast loan acceptance. It demonstrated resilience during both the training (98.325%) and testing (95.556%) stages of the process. Closely behind, Extra Trees Classifier showed great predictive strength with an accuracy of 91.11%, and Gradient Boosting, though doing effective, showed its effectiveness with an accuracy of 93.33%. On the other hand, the accuracies of the Decision Tree, K-Nearest Neighbours, and Logistic Regression models were much lower at 88.89%, 80.00%, and 82.22%, respectively. Also illustrated the comparitive representation of accurcies of models in the existing research [22] and current research.

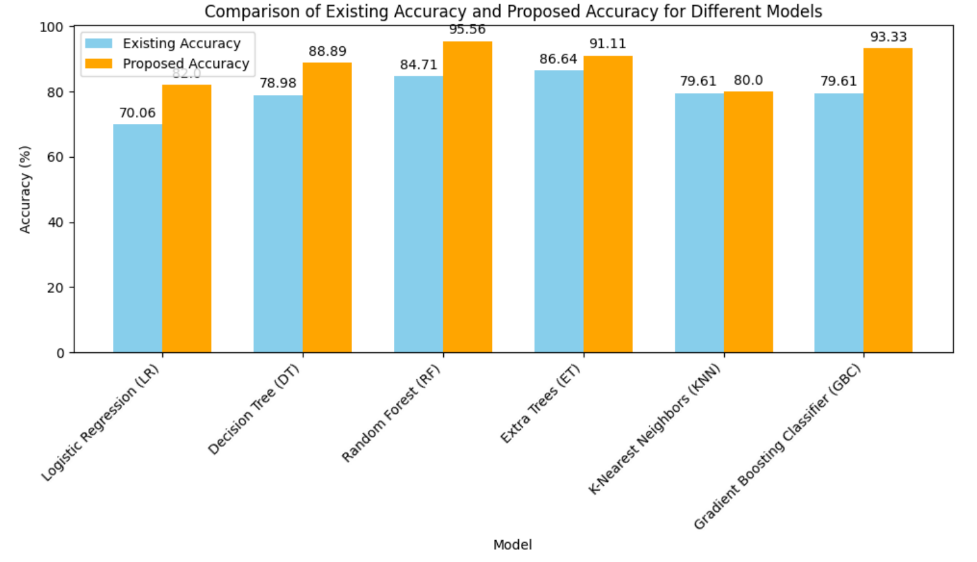


Fig.9 Comparing Accuracies

* *Voting Ensemble Method:*

In order to increase the overall accuracy, we will integrate the predictions of several machine learning models in our next stage, which is the implementation of a voting ensemble approach.

After careful examination, Random Forest was clearly the best performer, with an astounding accuracy of 95.56%. In training and testing, this model demonstrated exceptional robustness, with accuracy values of 98.325% and 95.556%, respectively. With an accuracy of 91.11%, the Extra Trees Classifier demonstrated noteworthy predictive capability, trailing closely behind. Likewise, Gradient Boosting demonstrated its efficacy with a 93.33% accuracy rate.

By combining Random Forest, Extra Trees Classifier, and Gradient Boosting, we will use ensemble learning to take advantage of the advantages of these high-performing models. In order to improve the accuracy of loan approval forecasts, this strategy seeks to use the combined predictive potential of models. Using the best-performing models and the ensemble approach, we were able to obtain an astounding accuracy of 95.55%. This suggests a notable improvement in the accuracy of loan approval forecasts.

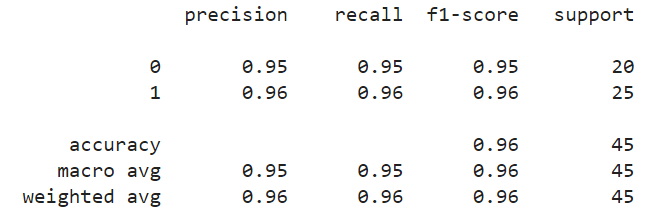


Fig.10 (a) Metrics for Voting Ensemble Model

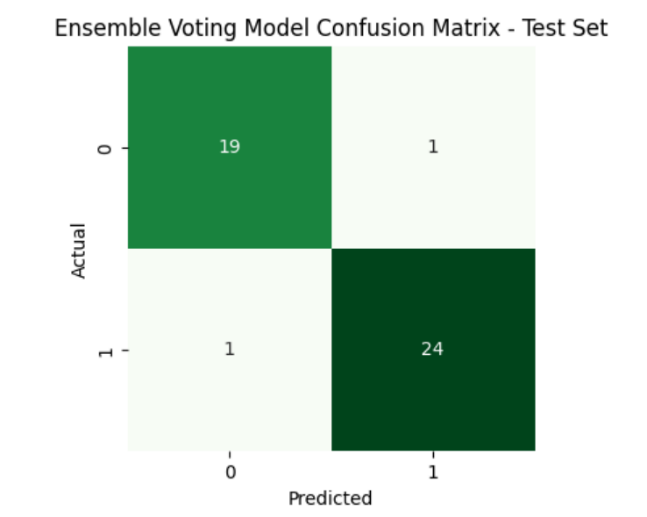
The confusion matrix's highlighted values suggest that the model is performing satisfactorily in classifying loan applications. The accuracy acquired with 24 true positives and 19 genuine negatives. With a precision of 95% for loans that have been granted, there is a high percentage of accurate forecasts among loans with this approval status. Furthermore, for loans that are authorised, recall is perfect and providing there are very few false negatives and false positives.

Fig.10 (b) Confusion Matrix of Voting Ensemble Model

# 5. CONCLUSION & FUTURE SCOPE

In summary, our initiative has shed light on the significant contribution that machine learning (ML) approaches have made to transforming the process of predicting loan approval. We have established a solid basis for our predictive models by carefully examining our dataset and utilising sophisticated preprocessing methods, such as managing null values, outliers, square root transformation, correlation analysis, feauture selection and SMOTE.

Going forward, our model achieved an amazing accuracy of 95.56%—serve as a testament to the ability of ML to provide incredibly precise predictions. This represents a notable improvement above the Existing accuracy of 87.26% from the previous base study and highlights the enormous progress made possible by our rigorous methodology.

Table1: Comparing Final Accuracies

|  |  |
| --- | --- |
|  | Accuracy(%) |
| Existing Accuracy | 87.26 |
| Proposed Accuracy | 95.56 |

Our research opens up a world of opportunities for improving loan approval prediction models in the future. Subsequent investigations may explore further into the field of deep learning, utilising neural networks capacity to address complex patterns in loan data. In addition, to guarantee flexibility in response to changing patterns and dynamics in loan application and repayment behaviour, ongoing model monitoring systems will be necessary. We are steadfast in our research to advance ML innovation and bring about revolutionary change in the field of financial decision-making processes as we move forward.

6. REFERENCES

1. Gopinath, Mahankali, K. Srinivas Shankar Maheep, and R. Sethuraman. 2021. “Customer Loan Approval Prediction Using Logistic Regression.” Advances in Parallel Computing. https://doi.org/10.3233/apc210103.
2. Pidikiti Supriya, Myneedi Pavani, Nagarapu Saisushma,Namburi Vimala Kumari, k Vikash,“Loan Prediction by using Machine Learning Models”, International Journal of Engineering and Techniques.Volume 5 Issue 2, Mar-Apr 2019.
3. A. Uzair, T. Aziz, H. Ilyas, S. Asim, B. N. Kadhar, "An Empirical Study on Loan Default Prediction Models" Journal of Computational and Theoretical Nanoscience, Volume 16, Number 8, August 2019, pp. 3483-3488(6). DOI: https://doi.org/10.1166/jctn.2019.8312
4. M. Madaan et al. "Loan default prediction using decision trees and random forest: A comparative study" IOP Conf. Ser.: Mater. Sci. Eng. 2014. doi: 10.1088/1757-899X/1022/1/012042.
5. Alsaleem, M. Y., & Hasoon, S. O. (2020). Predicting bank loan risks using machine learning algorithms. AL-Rafidain J. Comput. Sci. Math., 14(1), 149–158.
6. Ramachandra, H. V., G. Balaraju, R. Divyashree, and Harish Patil. 2021. “Design and Simulation of Loan Approval Prediction Model Using AWS Platform.” 2021 International Conference on Emerging Smart Computing and Informatics (ESCI). https://doi.org/10.1109/esci50559.20219397049.

Data set used <https://www.kaggle.com/datasets/altruistdelhite04/loan-prediction-problem-dataset/.%20>.

1. Singh, Vishal, Ayushman Yadav, Rajat Awasthi, and Guide N. Partheeban. 2021. “Prediction ofModernized Loan Approval System Based on MachineLearning Approach.” 2021 International Conference on IntelligentTechnologies(CONIT).https://doi.org/10.1109/conit51480.2021.9498475.
2. S.m., Karthikeyan, S. M. Karthikeyan, and Pushpa Ravikumar. 2021. “A Comparative Analysis of Feature Selection for Loan Prediction Model.” InternationalJournalofComputerApplications.<https://doi.org/10.5120/ijca2021920992>.
3. Hassan, Amira Kamil Ibrahim and Ajith Abraham. "Modeling consumer loan default prediction using ensemble neural networks" , 2013 International Conference On Computing, Electrical And Electronic Engineering (ICCEEE). IEEE, 2013.
4. .Nitesh Pandey et al. (2022). "Loan Approval Prediction using Machine Learning Algorithms Approach.", IRJMETS, this paper achieved an accuracy of 78.3% using Logistic Regression, Decision Trees, and KNN. It discusses limitations in feature engineering and imbalanced datasets.
5. J. Tejaswini (2022). "Accurate Loan Approval Prediction Based on Machine Learning Approach.", IRJMETS, this paper achieved an accuracy of 88.1% using XGBoost. It emphasizes the importance of careful data pre-processing and feature engineering for optimal performance.
6. Anant Shinde et al. (2022). "Loan Prediction System Using Machine Learning." Published in the field of Finance by IEEE, this paper compares Logistic Regression, Decision Tree, and Random Forest models. It concludes that Decision Tree and Random Forest outperform Logistic Regression due to their non-linearity handling capabilities.
7. Dharavath Sai Kiran. Avula et al. (2023). "Loan Approval Prediction using Adversarial Training and Data Science." Within the domain of Finance, this paper discusses a model that produced low accuracy values and could only handle minimum-sized data.
8. Zhu L, Qiu D, Ergu D, Ying C, Liu K (2019) A study on predicting loan default based on the random forest algorithm. Procedia Comput Sci 162:503–513.
9. Nigmonov A, Shams S, Alam K (2022) Research Macroeconomic determinants of loan defaults: evidence from the U.S. peer-to-peer lending market. Res Int Bus Finan 59:101516.
10. Lee JW, Lee WK, Sohn SY (2021) Graph convolutional network-based credit default prediction utilizing three types of virtual distances among borrowers. Expert Syst Appl 168:114411.
11. Lim S-J, Thiel C, Sehm B, Deserno L, Lepsien J, Obleser J (2022) Distributed networks for auditory memory differentially contribute to recall precision. NeuroImage 256:119227.
12. Fontem B, Smith J (2019) Analysis of a chance-constrained new product risk model with multiple customer classes. Eur J Oper Res 272(3):999–1016 23. Bianco S, Mazzini D, Napoletano P, Schettin R (2019) Multitask painting categorization by deep multibranch neural network. Expert Syst Appl 135:90–101.
13. Wang L, Chen Y, Jiang H, Yao J (2020) Imbalanced credit risk evaluation based on multiple sampling, multiple kernel fuzzy self-organizing map and local accuracy ensemble. Appl Soft Comput 91:106262
14. Vuttipittayamongkol P, Elyan E, Petrovski A (2021) On the class overlap problem in imbalanced data classification. Knowl-Based Syst 212:106631 26. Papouskova M, Hajek P (2019) Two stage consumer credit risk modelling using heterogeneous ensemble learning. Decis Support Syst 118:33–45.
15. Ashofteh A, Bravo JM (2021) A conservative approach for online credit scoring. Expert Syst Appl 176:114835.

[22] N. Uddin, M.K. Uddin Ahamed, M.A. Uddin, Md. Manwarul Islam, Md. Alamin Talukder, and Sunil Aryal, "An ensemble machine learning based bank loan approval predictions system with a smart application," International Journal of Cognitive Computing in Engineering, vol. 4, pp. 327-339, 2023.http://doi.org/10.1016/j.ijcce.2023.09.001.

## 



