



Faculty of Electronic Engineering & Technology (FKTEN)

NMK 40403: ARTIFICIAL INTELLIGENCE

SEM I 2024/2025

MINI PROJECT

LOAN APPROVAL SYSTEM

| NAME | MATRIC NUMBER | COURSE |
|--|--------------------------|--|
| MUHAMMAD AFIQ DANIAL BIN MOHD NOOR | 221302388 | UR6523009- ELECTRONIC NETWORK DESIGN |
| MUHAMMAD AQIL SYARIMAN BIN SAFWAN JAMAL | 2213023390 | UR6523009- ELECTRONIC NETWORK DESIGN |
| MOHAMAD ZARIFF DANIEL BIN ZULCAFLE | 221301700 | UR6523009- ELECTRONIC NETWORK DESIGN |

TABLE OF CONTENTS

| Contents | Pages |
|---|-----------|
| ABSTRACT | 3 |
| TABLE OF CONTENTS | 1 |
| LIST OF FIGURES | 2 |
| | |
| CHAPTER 1: INTRODUCTION | 4 |
| 1.1 Background system | 4 |
| 1.2 Problem statements | 5 |
| 1.3 Objectives | 5 |
| 1.4 Scopes of work | 5 |
| | |
| CHAPTER 2: METHODOLOGY | 7 |
| 2.1 Machine Learning Algorithm | 7 |
| 2.1.1 Support Vector Machine (SVM) | 7 |
| 2.1.2 Naïve Bayes | 7 |
| 2.2 Loan Approval Prediction Dataset | 8 |
| 3.3 Flowchart | 8 |
| 2.3 Loan Approval Calculator | 9 |
| 2.2.1 Load Dataset | 9 |
| 2.2.2 Calculate Total Income | 9 |
| 2.2.3 Calculate Loan Amount | 10 |
| 2.2.4 Loan Status | 10 |
| 2.2.5 Loan Approval Probability Calculator Page | 11 |
| 2.2.6 User Interface Page | 12 |
| | |
| CHAPTER 3: RESULT AND DISCUSSION | 13 |
| 3.1 Loan Approval Probabilitty Calculator Page | 13 |
| 3.2 Approved Loan Status | 14 |
| 3.3 Rejected Loan Status | 15 |
| 3.4 Heat Map | 16 |
| 3.5 Loan Status | 16 |
| 3.6 Loan Status Vs Credit History | 17 |

| | |
|----------------------------------|-----------|
| 3.7 Loan Status Vs Property Area | 17 |
| 3.8 Loan Approval Data | 18 |
| 3.0 Discussion | 19 |
| CHAPTER 4: CONCLUSION | 20 |
| 4.1 Conclusion | 20 |
| 4.2 Future Recommendations | 21 |
| REFERENCES | 22 |

LOAN APPROVAL SYSTEM

ABSTRACT

The loan approval process in financial institutions traditionally involves manual decision-making, which can be time-consuming and prone to human error or bias. In an effort to streamline this process and increase decision-making efficiency, this project presents a Loan Approval System developed in Python. This system utilizes machine learning algorithms, such as Support Vector Machines (SVM), k-Nearest Neighbors (k-NN), Decision Trees, and Naïve Bayes, to predict the likelihood of a loan being approved or rejected based on various factors like applicant income, loan amount, credit history, education, and employment status.

The Loan Approval System offers a user-friendly interface built with Python's Tkinter library. Users can input their details, such as income, loan amount, credit history, and other relevant information, and the system will provide an instant prediction of whether the loan will be approved or rejected. This system analyzes the provided data by calculating probabilities derived from historical loan data, allowing it to output a prediction that reflects the likelihood of approval based on similar past cases.

The Loan Approval System is particularly relevant in the current digital era, where automation is essential for improving operational efficiency and maintaining a competitive edge in the financial industry. By leveraging machine learning, this system not only enhances the speed and accuracy of loan processing but also helps institutions manage risks better by identifying patterns in applicant data that may not be immediately obvious to human decision-makers. The result is a more efficient, transparent, and fair loan approval process that benefits both applicants and institutions alike.

CHAPTER 1

INTRODUCTION

1.1 Background system

In the financial industry, loan approval is a critical process that determines whether an applicant receives the necessary financial support. Traditionally, this process has been handled manually by loan officers who assess applications based on a variety of criteria such as income, credit score, employment status, and other personal factors. However, manual processing can be time-consuming and subject to human errors or biases.

With the rise of digital technologies, financial institutions have started adopting automated systems to make faster and more accurate loan approval decisions. Machine learning and artificial intelligence (AI) have shown significant potential in improving the loan approval process by analyzing vast amounts of data and making predictions based on historical trends.

The Loan Approval System developed in this project leverages machine learning algorithms such as Support Vector Machines (SVM), k-Nearest Neighbors (k-NN), Decision Trees, and Naïve Bayes to automate the loan decision process. This system processes applicant data, evaluates various criteria, and predicts whether a loan will be approved or rejected based on the historical data of previous loan applicants.

1.2 Problem statements

The loan approval process in financial institutions is often inefficient, relying heavily on manual procedures that are both time-consuming and prone to errors. As a result, applicants experience delays, and financial institutions face challenges in processing large volumes of applications. Manual methods also introduce the risk of human bias, where loan officers might unintentionally base decisions on personal judgments rather than objective data, leading to unfair outcomes. Additionally, the increasing number of loan applications puts further strain on traditional systems, making it difficult to scale operations and meet growing demands in a timely manner.

These issues contribute to an overall lack of consistency and transparency in loan decision-making, which can undermine both customer trust and the institution's credibility. The absence of automated systems also prevents the ability to analyze large datasets efficiently, hindering the institution's potential to derive valuable insights that could enhance decision-making accuracy. Given these limitations, there is a clear need for an automated solution that can reduce human errors, eliminate bias, and speed up the loan approval process while providing consistent, data-driven decisions.

1.3 Objectives

There are the main objectives that can defined the project outcome:

1. To design and develop an efficient and automated loan approval system using Python and machine learning algorithms.
2. To enhance the accuracy and consistency of loan decisions by reducing human bias.
3. To improve processing efficiency and scalability for handling large volumes of loan applications.

1.4 Scopes of work

This project involves the development of an automated loan approval system that calculates the probability of loan approval based on user inputs. The system will allow users (loan applicants) to enter key details, including the number of dependents, education status, self-employed status, total income, loan amount, loan term (in months), credit history, and property area. These inputs will be processed by machine learning algorithms, primarily using SVM (Support Vector Machine), to predict loan approval or rejection.

In addition to providing loan approval probabilities, the system will include a dataset analysis feature, where users and administrators can explore the historical data used to train the model. The system will also generate a heat map to visualize correlations between different features (e.g., income, loan amount, credit history) and loan approval outcomes, offering valuable insights into the factors influencing loan decisions.

Furthermore, the system will present an SVM graph, which will allow users to visualize how the machine learning model classifies applicants based on their input data. The graph will demonstrate the decision boundary created by the SVM algorithm, helping users understand the classification process in a more interactive and visual way.

The administrator panel will enable management of input data, loan approval predictions, and the generation of statistical visualizations (heat maps and graphs). The system provides a clear decision, indicating whether the loan is approved or not, based on the user's input and the prediction model.

CHAPTER 2

METHODOLOGY

2.1 MACHINE LEARNING ALGORITHM

2.1.1 Support Vector Machine (SVM)

Support Vector Machine (SVM) is a machine learning algorithm used to classify data into different categories. It works by finding a decision boundary, or hyperplane, that separates the data into distinct classes. In the case of loan approval, SVM helps categorize loan applications as either "Approved" or "Rejected." The model is trained using applicant data, such as credit score, income, and loan amount.

Once trained, the SVM model predicts loan approval by determining which side of the decision boundary the new applicant's data falls on. For example, based on the applicant's financial details, the SVM model will classify the loan as "Approved" or "Rejected" based on the established boundary. This process helps automate and improve the accuracy of loan decision-making.

2.1.3 Naïve Bayes

Naïve Bayes is a probabilistic machine learning algorithm used primarily for classification tasks. It is based on Bayes' Theorem, which calculates the probability of a loan being approved or rejected based on the features of the applicant. Naïve Bayes assumes that all features are independent of each other, which simplifies the calculation process. In the case of loan approval, Naïve Bayes calculates the probability of each class (Approved or Rejected) based on the applicant's data, such as income, credit history, and loan amount.

After training on historical loan data, Naïve Bayes predicts the loan status for new applicants by comparing the probabilities of approval and rejection. The class with the higher probability is chosen as the predicted outcome. Despite the assumption of feature independence, Naïve Bayes often performs well in practice and can be an effective tool for predicting loan approvals in cases with large datasets and many features.

| LOAN STATUS | | DEPENDENTS(CUST) | | | | EDUCATION | | SELF EMPLOYED | |
|-------------|---------|------------------|--------|--------|-------|---------------|-------------------|---------------|---------|
| | | 0(339) | 1(105) | 2(103) | 3(51) | GRADUATE(465) | NOT GRADUATE(133) | YES(110) | NO(488) |
| APPROVE | 411/598 | 235 | 67 | 78 | 31 | 329 | 82 | 75 | 336 |
| NOT APPROVE | 187/598 | 104 | 38 | 25 | 20 | 136 | 51 | 35 | 152 |

| TOTAL APPLICANT INCOME (APPLICANT + COAPPLICANT) | | | LOAN AMOUNT | | |
|--|-----------------|------------|-------------|-----------|------|
| <5000(264) | 5001-10000(252) | >10000(82) | <=144 | >144<=400 | >400 |
| 178 | 180 | 53 | 271 | 121 | 9 |
| 85 | 72 | 29 | 103 | 69 | 4 |

| LOAN AMOUNT TERM | | | CREDIT HISTORY | | PROPERTY AREA | | |
|------------------|-------------------|-----------|----------------|--------|---------------|------------|----------------|
| <160(12) | >=160 & <=360(58) | >360(414) | 0(86) | 1(463) | RURAL(175) | URBAN(182) | SEMIURBAN(241) |
| 9 | 389 | 13 | 7 | 368 | 108 | 130 | 173 |
| 3 | 174 | 10 | 79 | 95 | 67 | 52 | 68 |

Figure 2.1: Naïve Bayes Dataset

2.2 Loan Approval Prediction Dataset

The **loanapprovalprediction.csv** dataset contains essential information about loan applicants, including their demographic, financial, and credit-related details. Key features include applicant income, co-applicant income, loan amount, loan term, credit history, education level, dependents, employment status, and property area. The target variable, "Loan_Status," indicates whether a loan was approved ("Y") or rejected ("N").

| loan_ID | Gender | Married | Dependent | Education | Self_Emp | ApplicantIn | Coapplicant | LoanAmou | Loan_Amo | Credit_His | Property_A | Loan_Status |
|---------|--------|---------|-----------|-----------|----------|-------------|-------------|----------|----------|------------|------------|-------------|
| P002840 | Female | No | 0 | Graduate | No | 2378 | 0 | 9 | 360 | 1 | Urban | N |
| P001030 | Male | Yes | 2 | Graduate | No | 1299 | 1086 | 17 | 120 | 1 | Urban | Y |
| P001482 | Male | Yes | 0 | Graduate | Yes | 3459 | 0 | 25 | 120 | 1 | Semiurban | Y |
| P001325 | Male | No | 0 | Not Gradu | No | 3620 | 0 | 25 | 120 | 1 | Semiurban | Y |
| P002792 | Male | Yes | 1 | Graduate | No | 5468 | 1032 | 26 | 360 | 1 | Semiurban | Y |
| P001518 | Male | Yes | 1 | Graduate | No | 1538 | 1425 | 30 | 360 | 1 | Urban | Y |
| P001888 | Female | No | 0 | Graduate | No | 3237 | 0 | 30 | 360 | 1 | Urban | Y |
| P001086 | Male | No | 0 | Not Gradu | No | 1442 | 0 | 35 | 360 | 1 | Urban | N |
| P002894 | Female | Yes | 0 | Graduate | No | 3166 | 0 | 36 | 360 | 1 | Semiurban | Y |
| P002979 | Male | Yes | 3 | Graduate | No | 4106 | 0 | 40 | 180 | 1 | Rural | Y |
| P002634 | Female | No | 1 | Graduate | No | 13262 | 0 | 40 | 360 | 1 | Urban | Y |
| P001768 | Male | Yes | 0 | Graduate | Yes | 3716 | 0 | 42 | 180 | 1 | Rural | Y |
| P001430 | Female | No | 0 | Graduate | No | 4166 | 0 | 44 | 360 | 1 | Semiurban | Y |
| P001138 | Male | Yes | 1 | Graduate | No | 5649 | 0 | 44 | 360 | 1 | Urban | Y |
| P002689 | Male | Yes | 2 | Not Gradu | No | 2192 | 1742 | 45 | 360 | 1 | Semiurban | Y |
| P002288 | Male | Yes | 2 | Not Gradu | No | 2889 | 0 | 45 | 180 | 0 | Urban | N |
| P002116 | Female | No | 0 | Graduate | No | 2378 | 0 | 46 | 360 | 1 | Rural | N |
| P001120 | Male | No | 0 | Graduate | No | 1800 | 1213 | 47 | 360 | 1 | Urban | Y |
| P001653 | Male | No | 0 | Not Gradu | No | 4885 | 0 | 48 | 360 | 1 | Rural | Y |
| P002398 | Male | No | 0 | Graduate | No | 1926 | 1851 | 50 | 360 | 1 | Semiurban | Y |
| P001333 | Male | Yes | 0 | Graduate | No | 1977 | 997 | 50 | 360 | 1 | Semiurban | Y |

Figure 2.2: Loan Approval Prediction Dataset

2.3 Loan Approval Calculator

2.3.1 Load Dataset

The code begins by importing the necessary libraries: tkinter for creating the graphical user interface (GUI) and pandas for data handling. The `file_path` variable specifies the location of the dataset file, 'LoanApprovalPrediction.csv,' which contains information relevant to loan approval predictions. Using the `pd.read_csv(file_path)` function from the pandas library, the dataset is read into a DataFrame, which is a table-like structure that allows easy manipulation and analysis of the data. This DataFrame is stored in the variable `data`, making it ready for processing, analysis, and further use in the loan approval prediction system.

```
1 import tkinter as tk
2 from tkinter import ttk
3 import pandas as pd
4
5 # Load the dataset from a CSV file
6 file_path = 'LoanApprovalPrediction.csv' # Replace with the path to your CSV file
7 data = pd.read_csv(file_path)
8
```

Figure 2.3: Load Dataset Coding

2.3.2 Calculate Total Income

The code calculates a new column called `TotalIncome` by adding the `ApplicantIncome` and `CoapplicantIncome` columns from the dataset. This gives the total income for each applicant, which includes both the applicant's and their coapplicant's earnings. Next, the `categorize_total_income` function is defined to categorize the `TotalIncome` into three ranges: "<5000," "5001-10000," and ">10000." This function checks the value of `TotalIncome` and assigns the corresponding category. The `apply()` function is used to apply this categorization to the entire `TotalIncome` column, creating a new column, `TotalIncome_Category`, that stores the income categories for each loan applicant.

```

# Calculate TotalIncome (ApplicantIncome + CoapplicantIncome)
data['TotalIncome'] = data['ApplicantIncome'] + data['CoapplicantIncome']

# Categorize TotalIncome
def categorize_total_income(value):
    if value < 5000:
        return "<5000"
    elif 5001 <= value <= 10000:
        return "5001-10000"
    else:
        return ">10000"

data['TotalIncome_Category'] = data['TotalIncome'].apply(categorize_total_income)

```

Figure 2.4: Calculate Total Income Coding

2.3.3 Calculate Loan Amount

The code categorizes the LoanAmount and Loan_Amount_Term columns into defined ranges for easier analysis. It creates two functions: categorize_loan_amount classifies loan amounts into "<=144," ">144<=400," and ">400," while categorize_loan_term divides loan terms into "<160," ">=160," and ">360." These functions are applied to the respective columns, resulting in two new categorized columns, LoanAmount_Category and Loan_Amount_Term_Category, making the data more organized and easier to interpret.

```

# Categorize LoanAmount
def categorize_loan_amount(value):
    if value <= 144:
        return "<=144"
    elif 145 <= value <= 400:
        return ">144<=400"
    else:
        return ">400"

data['LoanAmount_Category'] = data['LoanAmount'].apply(categorize_loan_amount)

# Categorize Loan_Amount_Term
def categorize_loan_term(value):
    if value < 160:
        return "<160"
    elif 160 <= value <= 360:
        return ">=160"
    else:
        return ">360"

data['Loan_Amount_Term_Category'] = data['Loan_Amount_Term'].apply(categorize_loan_term)

```

Figure 2.5: Calculate Loan Amount Coding

2.3.4 Loan Status

The code filters the dataset into two subsets: `approved_loans` for applications with `Loan_Status = 'Y'` and `rejected_loans` for `Loan_Status = 'N'`. It also defines a function, `calculate_probability`, to compute the probabilities of loan approval or rejection based on specific feature values. This function counts how many applicants with a given feature value have approved or rejected loans, calculates the probabilities by dividing these counts by the total approved and rejected loans, and returns the results. This helps in analyzing the influence of individual features on loan approval outcomes.

```
# Filter only approved (Loan_Status = 'Y')
approved_loans = data[data['Loan_Status'] == 'Y']
rejected_loans = data[data['Loan_Status'] == 'N']

# Define a function to calculate the probability for each feature
def calculate_probability(feature_value, feature_column, total_approved, total_rejected):
    # Count how many applicants in the dataset have the specified feature value and an approved loan
    count_approved = approved_loans[approved_loans[feature_column] == feature_value].shape[0]
    count_rejected = rejected_loans[rejected_loans[feature_column] == feature_value].shape[0]

    # Return the probability (count of approved loans for that feature / total approved loans)
    prob_approved = count_approved / total_approved
    prob_rejected = count_rejected / total_rejected

    return prob_approved, prob_rejected
```

Figure 2.6: Loan Status Coding

2.3.5 Loan Approval Probability Calculator Page

This code creates a graphical user interface (GUI) using Tkinter to calculate the probability of loan approval based on user inputs. The main window, titled "Loan Approval Probability Calculator," collects details such as dependents, education, self-employment status, income, loan amount, loan term, credit history, and property area. It uses predefined functions to calculate probabilities of loan approval and rejection for each feature based on the dataset. By combining these probabilities, the program determines the final loan status as "Approved" or "Not Approved" and displays detailed results in the GUI.

```

# Create the main window using Tkinter
def create_app_window():
    # Create main Tkinter window
    window = tk.Tk()
    window.title("Loan Approval Probability Calculator")

    # Set window size
    window.geometry("600x700")

    # Define variables for user input
    dependents = tk.IntVar()
    education = tk.StringVar()
    self_employed = tk.StringVar()
    total_income = tk.IntVar()
    loan_amount = tk.IntVar()
    loan_amount_term = tk.IntVar()
    credit_history = tk.IntVar()
    property_area = tk.StringVar()

    # Define the function to get input from the user and calculate the probability
    def get_applicant_details_and_calculate():
        total_approved = 411 # Total approved loans in the dataset (411 out of 598)
        total_rejected = 187 # Total rejected loans in the dataset (187 out of 598)

        # Get input values from user
        dep = dependents.get()
        edu = education.get()
        self_emp = self_employed.get()
        income = total_income.get()
        loan_amt = loan_amount.get()
        loan_term = loan_amount_term.get()
        credit_hist = credit_history.get()
        prop_area = property_area.get()

```

Figure 2.7: Loan Approval Probability Calculator Page

2.3.6 User Interface Page

This section of the code defines the user interface (UI) components for the loan approval probability calculator. It uses Tkinter widgets like Label, Entry, and Combobox to create input fields for various loan applicant details such as dependents, education, employment status, income, loan amount, loan term, credit history, and property area. Each field is labeled and arranged neatly in the main window using the grid layout. A "Calculate" button is provided, which triggers the probability calculation function. The results, including the loan approval status and associated probabilities, are displayed in a Label widget below the input fields.

```

# create UI components
tk.Label(window, text="Enter number of dependents (e.g., 0, 1, 2, ...):").grid(row=0, column=0, sticky=tk.W, padx=10, pady=5)
tk.Entry(window, textvariable=dependents).grid(row=0, column=1, padx=10, pady=5)

tk.Label(window, text="Enter education status (graduate/not graduate):").grid(row=1, column=0, sticky=tk.W, padx=10, pady=5)
tk.Combobox(window, textvariable=education, values=["graduate", "not graduate"]).grid(row=1, column=1, padx=10, pady=5)

tk.Label(window, text="Enter self-employed status (yes/no):").grid(row=2, column=0, sticky=tk.W, padx=10, pady=5)
tk.Combobox(window, textvariable=self_employed, values=["yes", "no"]).grid(row=2, column=1, padx=10, pady=5)

tk.Label(window, text="Enter total income (ApplicantIncome + CoapplicantIncome):").grid(row=3, column=0, sticky=tk.W, padx=10, pady=5)
tk.Entry(window, textvariable=total_income).grid(row=3, column=1, padx=10, pady=5)

tk.Label(window, text="Enter loan amount:").grid(row=4, column=0, sticky=tk.W, padx=10, pady=5)
tk.Entry(window, textvariable=loan_amount).grid(row=4, column=1, padx=10, pady=5)

tk.Label(window, text="Enter loan amount term (in months):").grid(row=5, column=0, sticky=tk.W, padx=10, pady=5)
tk.Entry(window, textvariable=loan_amount_term).grid(row=5, column=1, padx=10, pady=5)

tk.Label(window, text="Enter credit history (1 for good, 0 for bad):").grid(row=6, column=0, sticky=tk.W, padx=10, pady=5)
tk.Entry(window, textvariable=credit_history).grid(row=6, column=1, padx=10, pady=5)

tk.Label(window, text="Enter property area (Urban/Semiurban/Rural):").grid(row=7, column=0, sticky=tk.W, padx=10, pady=5)
tk.Combobox(window, textvariable=property_area, values=["Urban", "Semiurban", "Rural"]).grid(row=7, column=1, padx=10, pady=5)

# calculate button
calculate_button = tk.Button(window, text="Calculate Loan Approval Probability", command=get_applicant_details_and_calculate)
calculate_button.grid(row=8, column=0, colspan=2, padx=20)

# Result label
result_label = tk.Label(window, text="", justify=tk.LEFT)
result_label.grid(row=9, column=0, colspan=2, padx=10, pady=10)

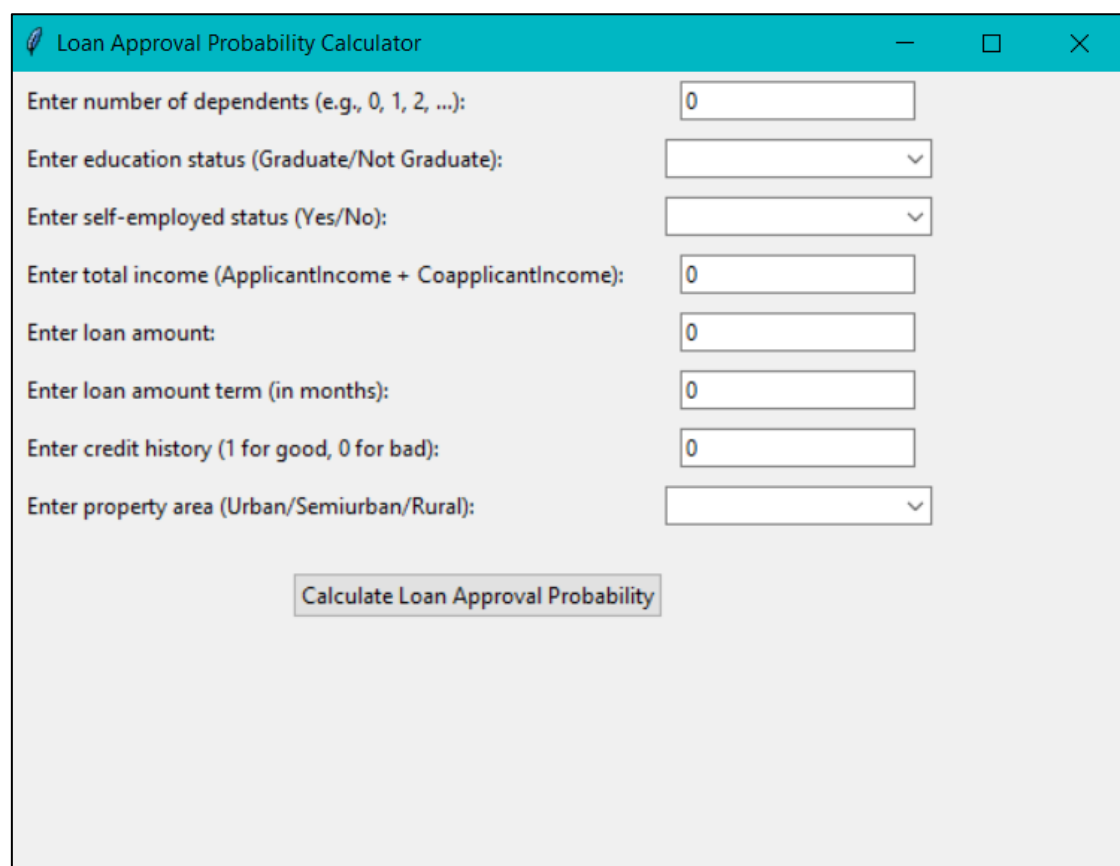
```

Figure 2.8: Loan Approval Probability Calculator Page

CHAPTER 3

RESULT AND DISCUSSION

3.1 Loan Approval Probabililty Calculator Page



The screenshot displays a web application window titled "Loan Approval Probability Calculator". The interface includes several input fields for user data, each followed by a small downward arrow indicating a dropdown menu. The inputs are: "Enter number of dependents (e.g., 0, 1, 2, ...):" with the value "0"; "Enter education status (Graduate/Not Graduate):"; "Enter self-employed status (Yes/No):"; "Enter total income (ApplicantIncome + CoapplicantIncome):" with the value "0"; "Enter loan amount:" with the value "0"; "Enter loan amount term (in months):" with the value "0"; "Enter credit history (1 for good, 0 for bad):" with the value "0"; and "Enter property area (Urban/Semiurban/Rural):". A "Calculate Loan Approval Probability" button is positioned below the input fields.

| Field Label | Value |
|---|-------|
| Enter number of dependents (e.g., 0, 1, 2, ...): | 0 |
| Enter education status (Graduate/Not Graduate): | |
| Enter self-employed status (Yes/No): | |
| Enter total income (ApplicantIncome + CoapplicantIncome): | 0 |
| Enter loan amount: | 0 |
| Enter loan amount term (in months): | 0 |
| Enter credit history (1 for good, 0 for bad): | 0 |
| Enter property area (Urban/Semiurban/Rural): | |

Calculate Loan Approval Probability

Figure 3.1: Loan Approval Probability Calculator Page Interface

3.2 Approved Loan Status

Loan Approval Probability Calculator

Enter number of dependents (e.g., 0, 1, 2, ...):

1

Enter education status (Graduate/Not Graduate):

Graduate

Enter self-employed status (Yes/No):

Yes

Enter total income (ApplicantIncome + CoapplicantIncome):

7500

Enter loan amount:

150

Enter loan amount term (in months):

170

Enter credit history (1 for good, 0 for bad):

1

Enter property area (Urban/Semiurban/Rural):

Urban

Calculate Loan Approval Probability

Calculated probabilities for loan approval based on the input details:

Dependents (Approved/Rejected): 0.1630/0.2032

Education (Approved/Rejected): 0.8005/0.7273

Self Employed (Approved/Rejected): 0.1825/0.1872

Total Income (Approved/Rejected): 0.4380/0.3850

Loan Amount (Approved/Rejected): 0.2944/0.3690

Loan Term (Approved/Rejected): 0.9465/0.9305

Credit History (Approved/Rejected): 0.8954/0.5080

Property Area (Approved/Rejected): 0.3163/0.3636

Final Probability of Loan Approval (P(Y)): 0.00082299

Final Probability of Loan Rejection (P(N)): 0.00067549

Loan Status: Approved

Figure 3.2: Approved Loan Status Interface

3.3 Rejected Loan Status

Loan Approval Probability Calculator

Enter number of dependents (e.g., 0, 1, 2, ...):

1

Enter education status (Graduate/Not Graduate):

Graduate

Enter self-employed status (Yes/No):

Yes

Enter total income (ApplicantIncome + CoapplicantIncome):

7500

Enter loan amount:

150

Enter loan amount term (in months):

170

Enter credit history (1 for good, 0 for bad):

1

Enter property area (Urban/Semiurban/Rural):

Urban

Calculate Loan Approval Probability

Calculated probabilities for loan approval based on the input details:

Dependents (Approved/Rejected): 0.1630/0.2032

Education (Approved/Rejected): 0.8005/0.7273

Self Employed (Approved/Rejected): 0.1825/0.1872

Total Income (Approved/Rejected): 0.4380/0.3850

Loan Amount (Approved/Rejected): 0.2944/0.3690

Loan Term (Approved/Rejected): 0.9465/0.9305

Credit History (Approved/Rejected): 0.8954/0.5080

Property Area (Approved/Rejected): 0.3163/0.3636

Final Probability of Loan Approval (P(Y)): 0.00082299

Final Probability of Loan Rejection (P(N)): 0.00067549

Loan Status: Approved

Figure 3.3: Not Approved Loan Status Interface

3.4 Heat Map

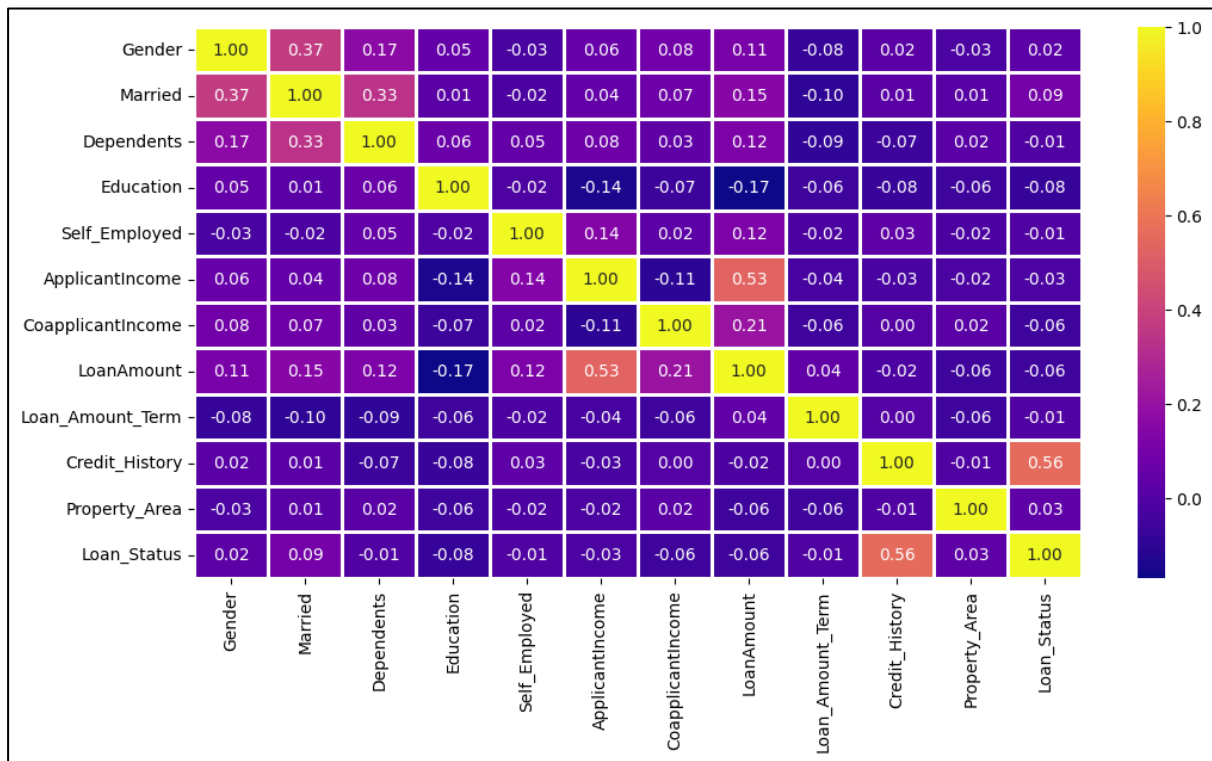


Figure 3.4: Heat Map

3.5 Loan Status

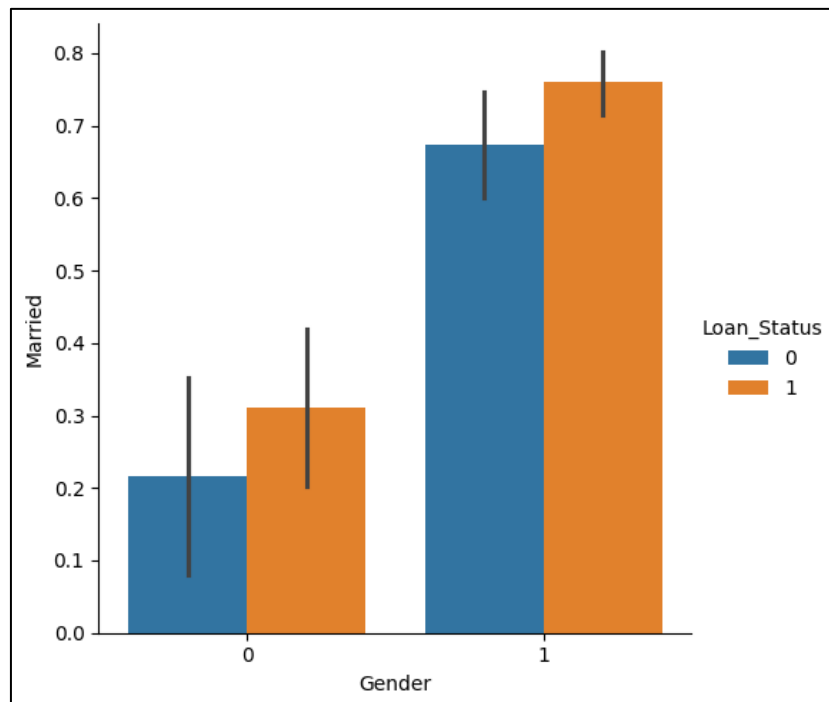


Figure 3.5: Heat Map

3.6 Loan Status vs Credit History

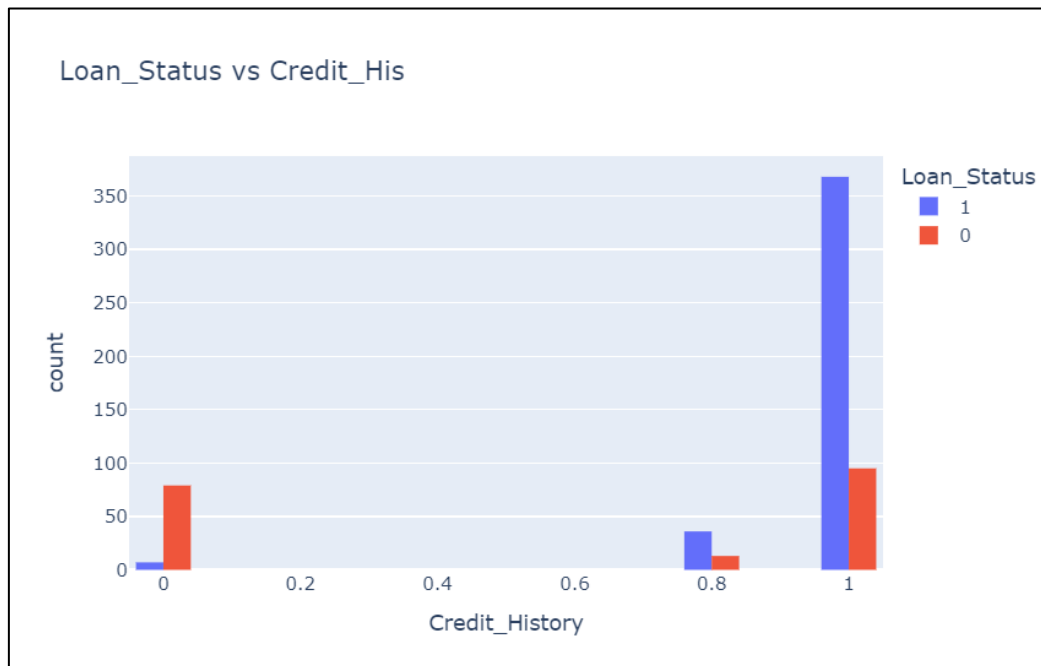


Figure 3.6: Loan Status versus Credit History Graph

3.7 Loan Status vs Property Area



Figure 3.7: Loan Status versus Property Area

3.8 Loan Approval Data

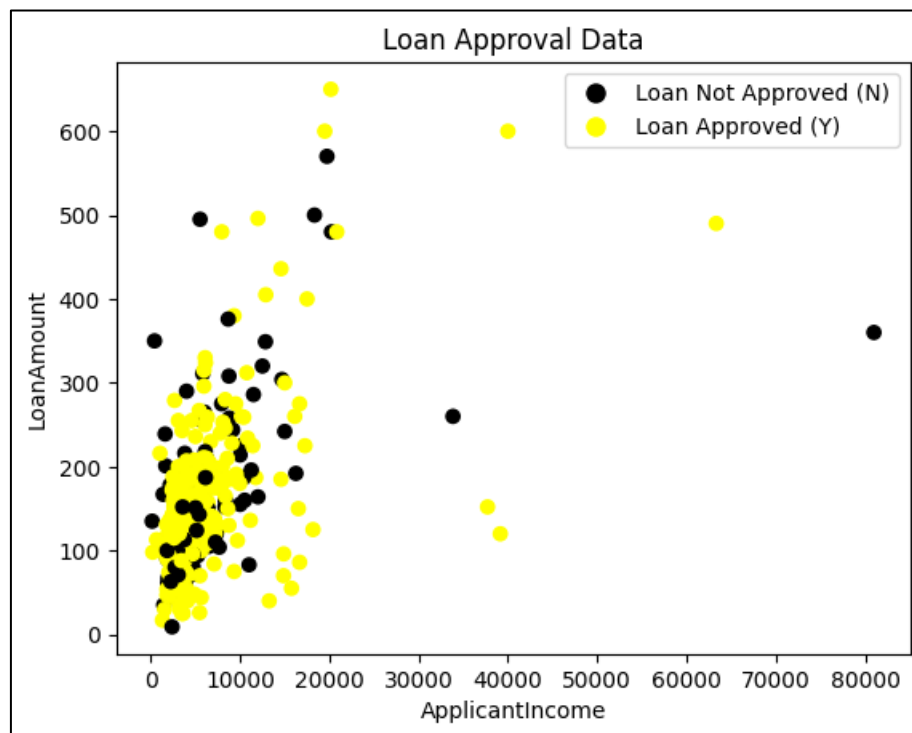


Figure 3.8: Loan Approval

3.9 Discussion

The Loan Approval System was developed using Python programming language, along with Tkinter and Pandas libraries. Various programming environments and tools, such as Visual Studio Code, Eclipse, and Jupyter Notebook, are available to build similar systems. However, Python was chosen for this project due to its simplicity, versatility, and strong library support for data analysis and user interface development. Tkinter was used as the GUI toolkit for building an intuitive interface, while Pandas was selected for its robust data manipulation capabilities, allowing efficient management of the loan dataset. These tools together provide a comprehensive platform for implementing a user-friendly and functional system.

Additionally, the system leverages probability-based calculations to assess loan approval status. The dataset is analyzed to predict the probability of loan approval or rejection using Naive Bayes-like techniques. Applicants enter their details, such as dependents, education, income, and credit history, into the system. The system then calculates approval probabilities based on predefined criteria. Users can also view detailed insights into the decision-making process, ensuring transparency. Despite challenges in aligning the calculations with real-world scenarios and ensuring accuracy, the system performs as expected. It efficiently reduces manual effort, enhances decision-making, and simplifies the loan approval process for financial institutions.

One of the key strengths of this system is its ability to process and analyze large datasets quickly. By automating the process of loan application assessment, financial institutions can save significant time and resources, which would otherwise be spent on manual reviews. The system also provides a level of objectivity by relying on data-driven decisions, reducing human biases that may influence the approval or rejection of applications.

Additionally, users are provided with clear insights into why a particular decision was made, adding to the system's transparency and trustworthiness. Furthermore, the Loan Approval System is scalable and flexible, making it suitable for different types of financial institutions, ranging from small credit unions to large banks. By incorporating machine learning techniques, such as probability-based modeling, the system can be easily adapted to new datasets or refined to improve its accuracy over time.. The system can also be integrated with existing databases and business workflows, providing seamless automation of the loan approval process.

CHAPTER 4

CONCLUSION

4.1 Conclusion

In conclusion, the Loan Approval System successfully automates the process of loan application assessment, offering a more efficient and objective approach compared to traditional manual methods. By utilizing Python, Tkinter, and Pandas, the system is not only user-friendly but also capable of handling large datasets and performing complex probability-based calculations. The integration of probability models, like Naive Bayes, ensures that the loan approval process is data-driven and transparent, allowing users to gain insights into how decisions are made.

The system's ability to simplify the decision-making process for financial institutions is a significant advantage, particularly in reducing human biases, saving time, and improving operational efficiency. With its flexible architecture, the system can be scaled and adapted to suit various financial institutions, from small businesses to large banks. While challenges such as ensuring data security and keeping the model updated with changing loan criteria persist, the system provides a solid foundation for improving loan approval processes and could be further developed with machine learning enhancements and stronger security measures.

Overall, the Loan Approval System not only meets the objectives set at the beginning of the project but also demonstrates the potential for automation in the financial sector, offering both financial institutions and applicants a more efficient, accurate, and transparent approach to loan approvals.

4.2 Future Recommendations

In the future, the Loan Approval System can be enhanced by incorporating more advanced machine learning algorithms, such as decision trees or support vector machines, to improve the accuracy and predictive power of the loan approval predictions. Currently, the system uses probability-based calculations, but machine learning techniques could enable the model to learn from historical data and adapt to new patterns, improving its decision-making capabilities over time. This would make the system more robust and reliable, helping to reduce errors in loan approvals and increasing its overall efficiency.

Another potential improvement is the integration of a real-time data validation and fraud detection system. Currently, the system relies on predefined inputs and doesn't assess the authenticity of the information provided by applicants. Future versions could incorporate checks against external databases or APIs to verify applicant details, such as employment history, income, and credit scores, to help detect fraudulent activities. This would significantly enhance the security and trustworthiness of the system, providing a more secure platform for both lenders and borrowers.

Finally, the system could be made more accessible by developing a mobile application or web-based platform. While the current version provides a desktop interface through Tkinter, expanding it to a web or mobile interface would allow users to apply for loans, check approval statuses, and view insights from any device, enhancing user accessibility and convenience. This would also enable financial institutions to reach a wider audience and streamline their loan processing across various platforms, contributing to better user engagement and higher efficiency.

4.3 REFERENCES

1. *Loan-Approval-Prediction-Dataset*. (2023, July 16). Kaggle.
<https://www.kaggle.com/datasets/architsharma01/loan-approval-prediction-dataset>
2. GeeksforGeeks. (2025, January 27). *Loan Approval Prediction using Machine Learning*. GeeksforGeeks. <https://www.geeksforgeeks.org/loan-approval-prediction-using-machine-learning/>
3. Prasertcbs. (n.d.). *basic-dataset/Loan-Approval-Prediction.csv at master · prasertcbs/basic-dataset*. GitHub. <https://github.com/prasertcbs/basic-dataset/blob/master/Loan-Approval-Prediction.csv>
4. Keithkamson. (n.d.). *GitHub - keithkamson/loan-prediction-ML: Loan Approval Prediction Model using the dataset from Kaggle*. GitHub.
<https://github.com/keithkamson/loan-prediction-ML>