LOAN APPROVAL PREDICTION USING MACHINE LEARNING

A PROJECT REPORT
Submitted by

GOURAV ACHARJEE
REGISTRATION NO.- 2212701110123
ROLL NO.- 430122010197

In fulfilment of the requirements for the award of the degree of
B. TECH IN COMPUTER SCIENCE & ENGINEERING

NARULA INSTITUTE OF TECHNOLOGY
AGARPARA, KOLKATA - 700109

SELF CERTIFICATE

I hereby certify that the work presented in this **Machine Learning Project on Loan Approval Prediction** is original and was carried out by me. All tasks, including data collection, preprocessing, model implementation, evaluation, and reporting, were completed in dependently. I acknowledge the sources used in the project and confirm that no unauthorized assistance was obtained during its completion.

NAME: GOURAV ACHARJEE
ROLL NO: 430122010197
REGISTRATION NO: 221270110123

ACKNOWLEDGEMENT

I feel immense pleasure to present my project entitled:
“Loan Prediction System using Machine Learning”
I would like to express my heartfelt gratitude to my respected teacher **Mr. Partha Koley**, who has been a constant source of knowledge and inspiration. His valuable guidance, encouragement, and constructive feedback have been instrumental in the successful completion of this project. I also extend my sincere thanks to all my college teachers who have imparted their knowledge and guidance throughout my academic journey. Finally, I would like to thank everyone who directly or indirectly supported me in completing this project successfully.

Name of the Student:

- GOURAV ACHARJEE

SIGNATURE OF THE STUDENT:

TABLE OF CONTENTS

CHAPTER NO.	TOPIC
1.	Abstract
2.	Introduction
	2.a Aim of the Project
	2.b Objective
	2.c Motivation
	2.d Scope of the Project
	2.e Review of Related Work
3.	Methodology
	3.a System Requirements
	3.a.1 Software Requirements
	3.a.2 Hardware Requirements
	3.b Overview of the Platform
	3.c Problem Formulation
	3.d Algorithm
	3.e Design Description
	3.e.1 Flow Chart
4.	Implementation
5.	Conclusion
6.	Limitations of the Work
7.	Future Scope
8.	References

ABSTRACT

Loan approval is an important process in modern banking and financial sector for ensuring profitability, minimizing risks, and maintaining customer confidence. Loan approval used to be determined manually by cross-checking the facts of the applicants, which caused delay, inefficiency, and even the possibility of human mistakes or bias. Technology advancements now make it possible for machine learning techniques to efficiently automate the process and improve the precision of loan approval predictions.

This project, "Loan Approval Prediction System using Machine Learning," tries to create an intelligent model that will provide the status of loan approval or rejection based on the applicant's information such as income, credit history, loan amount, employment, marital status, education, and the area of property. The data in this project is preprocessed by handling missing values, encoding categorical data, removing inconsistencies, and normalizing data wherever necessary. After preprocessing, exploratory data analysis (EDA) is performed in an attempt to become acquainted with feature distributions, determine correlations, and gain meaningful insights from the data.

Different machine learning models like Logistic Regression, Decision Tree, Random Forest, and Support Vector Machine are trained and compared. The models are validated using standard performance metrics like accuracy, precision, recall, F1-score, and confusion matrix. Out of these, ensemble-based models like Random Forest are found to perform well and therefore can be utilized efficiently in real-time loan prediction systems. The outcome of this project demonstrates the capability of machine learning to significantly enhance the efficiency and reliability of loan approval systems. Banks and financial institutions can make faster and more data-driven decisions, reduce default rates, and improve customer satisfaction using such systems. The project demonstrates the practical application of AI and ML in the financial sector and opens doors for better implementation through deep learning models and real-time implementation through web-based systems.

INTRODUCTION

Among the most important and challenging tasks of banks and lending institutions in the financial sector is the loan approval process. A loan provides individuals and businesses an opportunity to fulfill their financial needs, but it is risky for the lending institution too. The wrong approval decision can lead to loss, while denying a deserving applicant can destroy customer satisfaction and trust. An expeditious, accurate, and unbiased loan approval process is therefore of utmost importance.

Traditionally, the loan approval process has been carried out manually, where bank representatives check customer data such as income, employment record, credit record, loan amount, etc. to ascertain loan eligibility. Although a laborious and error-prone manual process is bound to be prejudiced at times, the increasing number of loan applications and the complex nature of financial data have necessitated the application of automated and intelligent decision-making systems.

Machine Learning (ML), as a subset of Artificial Intelligence (AI), provides robust methods to filter through massive databases of data and identify implicit patterns that can be applied to predict. Applying ML algorithms to customer information can make it feasible to create a forecasting system that is able to categorize whether a loan application should be approved or rejected with high precision. Not only does it accelerate the decision-making process, but also fairness, transparency, and consistency in loan approval are ensured. This project "Loan Prediction System using Machine Learning" intends to create an ML-based model to predict the loan approval status. The system is trained on a data set consisting of applicant information like gender, marital status, education, applicant income, co-applicant income, loan amount, credit history, and property area. Data preprocessing, feature engineering, and exploration data analysis are performed before training the data using different machine learning algorithms like Logistic Regression, Decision Tree, Random Forest, and Support Vector Machine. The performance of the above models is evaluated using parameters like accuracy, precision, recall, and F1-score in order to identify the best-performing algorithm.

Its implementation is a testament to the use of machine learning technology in banking to drive efficiency, minimize risk, and deliver improved customer experience. It is also a leap towards automated financial services, with intelligent systems able to assist human decision-makers in making more precise and data-driven loan approval decisions.

AIM OF THE PROJECT

The core intention of this project, "Loan Prediction System using Machine Learning," is to design and create a suitable, effective, and smart predictive model that can aid the loan approval process for banks and financial institutions. With the current financial age, where the number of loan applicants is increasing at a very rapid pace, manual verification processes not only take time but are also prone to human errors and biases. This project aims to surmount these limitations by using machine learning techniques to build accuracy, consistency, and fairness in the loan approval process.

The system is designed to analyze candidate information like gender, marriage, education level, work status, income level, loan amount, credit history, and property area to decide if a loan should be approved. Through training and testing machine learning models using historical loan information, the project aims to determine the patterns and relationships that affect loan eligibility. It is a predictive model that is

quicker, unbiased, and fact-based.

The aim of the project may be stated as follows:

- For developing a model that can predict whether loan applications should be "approved" or "rejected" based on the applicant's characteristics.
- To reduce the loan default risks through making precise predictions.
- To minimize human effort and processing time by automating the assessment procedure.
- To provide fairness and transparency in loan sanctioning by removing human bias.

So that we can illustrate how machine learning is being applied in the real world and its potential in addressing real-world problems. Through the fulfillment of this goal, the project assists in the modernization of banking services, increased customer satisfaction, and enabling banks to make informed and risk-free lending decisions. In addition, the project provides a platform for future development, such as incorporating deep learning models and utilizing the system in real time as a web-based or mobile application.

OBJECTIVE

The main objectives of the project **"Loan Prediction System using Machine Learning"** are:

- To study the different parameters that influence loan approval, such as applicant income, credit history, education, employment status, and loan amount.
- To clean, preprocess, and analyse the dataset in order to extract meaningful insights for building the model.
- To apply suitable machine learning algorithms and compare their performance for accurate prediction of loan eligibility.
- To design a system that can classify loan applications into *approved* or *rejected* categories in a fast and reliable manner.
- To minimize human effort and reduce processing time by automating the loan approval process.
- To ensure fairness and transparency in decision-making by reducing the chances of human error and bias.
- To create a base for future improvements, where the system can be extended with advanced techniques like deep learning and deployed as a web or mobile application for real-time use.

MOTIVATION

The motivation behind developing the project **"Loan Approval Prediction System using Machine Learning"** arises from the increasing importance of automation and accuracy in the financial sector. Loan approval is a critical process for banks and financial institutions, as it involves significant risk if handled incorrectly. Manual loan approval methods are often slow, inconsistent, and influenced by human judgment, which can lead to biased or erroneous decisions. With the rapid increase in the number of loan applicants, there is a growing demand for systems that can process applications more quickly and reliably.

Machine learning offers powerful tools to analyse large datasets and discover hidden patterns that are not easily noticeable through manual inspection. By applying these techniques, it becomes possible to predict loan eligibility in a way that is faster, unbiased, and data driven. The motivation for this project is to build a system that not only helps banks minimize the risk of loan defaults but also enhances customer satisfaction by speeding up the approval process.

Furthermore, this project provides an opportunity to apply the theoretical concepts of machine learning to solve a real-world problem. It serves as a step towards bridging the gap between academic learning and practical application, while also preparing for future advancements such as deploying intelligent financial solutions on web and mobile platforms.

SCOPE OF THE PROJECT

The project "Loan Prediction System using Machine Learning" has a wide scope in the financial and banking sector. With the rapid growth of digital banking, the number of loan applications has increased significantly, and banks require faster and more accurate systems to handle them. This project addresses that need by providing an automated solution for loan approval prediction.

- The system can be used by banks and financial institutions to make quick, data-driven decisions about loan approvals.
- It reduces dependency on manual evaluation, thereby saving time and minimizing human error.
- The model ensures fairness and transparency by eliminating subjective bias in decision-making.
- It helps in identifying risky loan applications in advance, thus minimizing the chances of loan defaults.
- The project can serve as a foundation for future improvements, such as incorporating deep learning methods for higher accuracy.
- It can also be extended into a real-time application by integrating it with web or mobile platforms, allowing banks to use it as a practical decision-support tool.
- Beyond the financial sector, the project provides a framework for applying machine learning to other predictive tasks where classification of data plays a crucial role.

In conclusion, the scope of this project is not limited to academic learning but extends to real-world applications, where it can help modernize banking processes, reduce financial risks, and enhance customer experience.

REVIEW OF RELATED WORK

In recent years, many researchers and developers have focused on applying machine learning techniques to improve decision-making in the banking and financial sectors. Loan prediction has been a popular area of study because of its practical importance in minimizing financial risks for banks and ensuring fairer approval processes for applicants.

Several studies have utilized classification algorithms such as Logistic Regression, Decision Trees, Random Forests, and Support Vector Machines (SVM) to predict loan eligibility. Logistic Regression has been widely used due to its simplicity and interpretability, but it sometimes fails to capture complex non-linear relationships in the data. Decision Trees and Random Forests, on the other hand, have shown better accuracy by handling large datasets and reducing overfitting. Support Vector Machines have also been explored for their effectiveness in binary classification problems like loan approval, though they can be computationally intensive.

Researchers have also applied ensemble methods and boosting techniques, such as XGBoost and AdaBoost, which have significantly improved the prediction accuracy compared to traditional models. Some studies combined machine learning with feature engineering to better analyze applicant data, such as credit history, income, and loan amount.

In addition, deep learning approaches have recently been tested for loan prediction, making use of artificial neural networks to capture complex patterns in applicant data. However, these models require large datasets and higher computational resources, which can limit their direct application in small or medium financial institutions.

From the review of related works, it is evident that while traditional models provide a strong foundation, advanced ensemble and deep learning methods show promising results for the future. This project builds upon these findings and applies machine learning algorithms to create an efficient, transparent, and accurate loan prediction system.

METHODOLOGY

SYSTEM REQUIREMENT:

SOFTWARE REQUIREMENT:

The software requirements for this project ensure smooth development, model training, and deployment of the loan prediction system. They include:

- **Python (Version 3.7 or higher):** Used for writing and executing the machine learning code and building the predictive model.
- **Libraries:** NumPy, Pandas, Matplotlib, Seaborn, Scikit-learn, Joblib – essential for data preprocessing, handling, visualization, model training, evaluation, and saving trained models.
- **Google Colab / Jupyter Notebook:** Used for developing, testing, and visualizing the machine learning models in an interactive environment.
- **Django/Flask (Optional for Deployment):** Can be used to integrate the model into a web application for real-world usage.
- **GitHub (Version Control):** For maintaining project code and collaboration.

HARDWARE REQUIREMENT:

The hardware requirements are minimal but ensure efficient processing and smooth execution of the project:

- **Processor (CPU):** Dual-core or quad-core processor (Intel i3 / AMD Ryzen 3 or higher).
- **RAM:** Minimum 4 GB (8 GB recommended for faster processing).
- **Storage:** 256 GB SSD or 500 GB HDD for dataset storage and software installation.
- **GPU (Optional):** A mid-range GPU (e.g., NVIDIA GTX 1050 or above) if working with large datasets or advanced ML models.
- **Network:** Reliable connection for dataset access, cloud integration, and deployment.
- **Internet Connection:** Essential for downloading required libraries, accessing datasets, using Google Colab, and hosting the project online.

PROBLEM FORMULATION

The problem formulation for this project focuses on developing an intelligent loan approval system that can accurately predict whether a loan application should be approved or rejected based on applicant financial and demographic data. Traditional loan evaluation methods are often time-consuming, prone to human bias, and may fail to capture subtle patterns in large datasets. Handling loan applications manually can lead to inefficiencies, delayed approvals, and possible errors in decision-making. Therefore, there is a need for a machine learning–based solution that can quickly, reliably, and fairly assess loan eligibility. Additionally, making this system accessible through an easy-to-use web platform helps financial institutions improve efficiency, reduce risk, and provide faster services to applicants.

ALGORITHM

Step 1: Data Preprocessing

- **Dataset**: The dataset (`loan.csv`) contains applicant financial and demographic attributes such as age, income, employment status, loan amount, credit history, dependents, etc.
- **Handling Missing Values**: All rows containing missing or null values are either removed or imputed to ensure a clean dataset.
- **Feature and Target Selection**: The features (**X**) are selected as all columns except the target, and the target (**y**) indicates whether the loan is approved (Yes/No).
- **Data Splitting**: The dataset is divided into training and testing sets using an 80:20 ratio through `train_test_split` for proper model evaluation.

Step 2: Model Training and Evaluation

- **Model Selection**: Multiple machine learning models are trained, including:
  - Logistic Regression
  - Decision Tree Classifier
  - Random Forest Classifier (100 estimators)
  - K-Nearest Neighbors (KNN)
  - Naive Bayes (GaussianNB)
- **Model Training**: Each model is trained using the training dataset.
- **Model Prediction**: Each trained model predicts outcomes on the testing dataset.
- **Model Evaluation**: Models are evaluated based on:
  - Confusion Matrix
  - Accuracy Score
  - Classification Report (Precision, Recall, F1-Score)

Step 3: Best Model Selection and Serialization

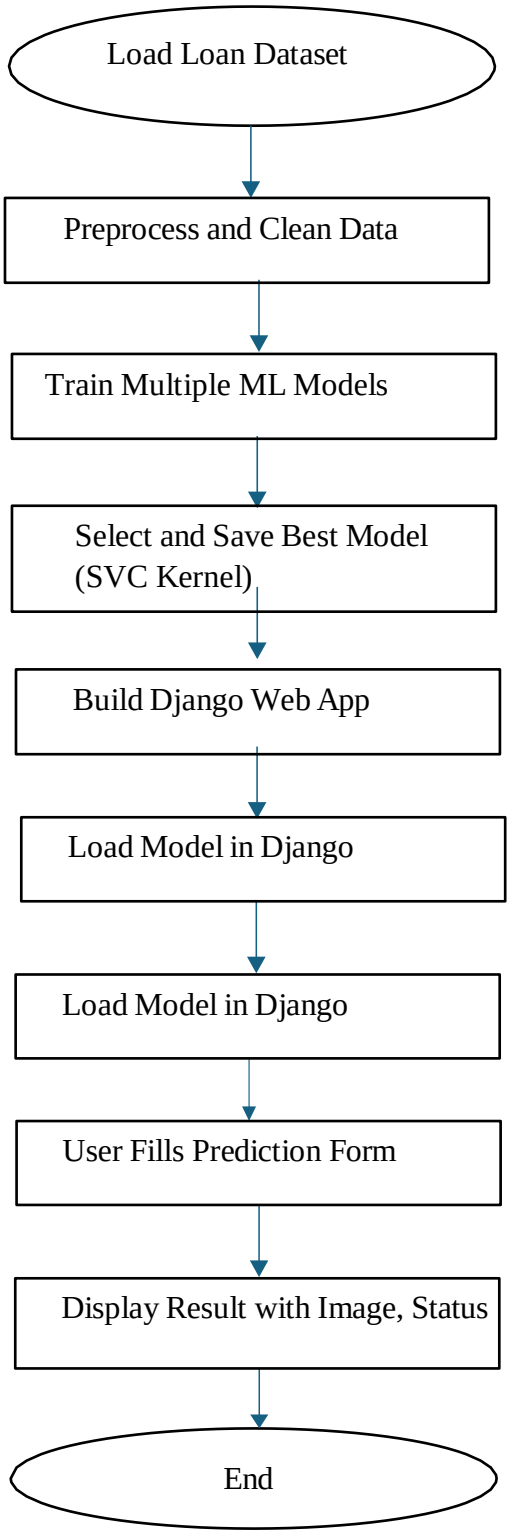
- **Best Model Selection**: After evaluation, the model with the highest accuracy (Random Forest Classifier) is selected as the final model.
- **Model Serialization**: The selected Random Forest model is saved using `joblib` into a file named `loanrf.joblib`. This allows the model to be easily loaded for deployment without retraining.

Step 4: Deployment and User Interface

- **Platform Deployment**: The trained model is deployed on a cloud platform (e.g., PythonAnywhere, Streamlit, or Flask app) to make the prediction service available online.
- **User Interaction**: Users access a web interface where they can input personal details such as age, income, employment status, loan amount, and credit history.
- **Prediction and Output**: The deployed model processes the input data and returns the prediction (Loan Approved / Loan Rejected) instantly.

DESIGN DESCRIPTION

FLOW CHART



IMPLEMENTATION

DATASET

Loan_ID	Gender	Married	Dependen	Education	Self_Empl	ApplicantIn	Coapplicar	LoanAmou	Loan_Amc	Credit_His	Property_/	Loan_Status
LP001002	Male	No	0	Graduate	No	5849	0		360	1	Urban	Y
LP001003	Male	Yes	1	Graduate	No	4583	1508	128	360	1	Rural	N
LP001005	Male	Yes	0	Graduate	Yes	3000	0	66	360	1	Urban	Y
LP001006	Male	Yes	0	Not Gradu	No	2583	2358	120	360	1	Urban	Y
LP001008	Male	No	0	Graduate	No	6000	0	141	360	1	Urban	Y
LP001011	Male	Yes	2	Graduate	Yes	5417	4196	267	360	1	Urban	Y
LP001013	Male	Yes	0	Not Gradu	No	2333	1516	95	360	1	Urban	Y
LP001014	Male	Yes	3+	Graduate	No	3036	2504	158	360	0	Semiurban	N
LP001018	Male	Yes	2	Graduate	No	4006	1526	168	360	1	Urban	Y
LP001020	Male	Yes	1	Graduate	No	12841	10968	349	360	1	Semiurban	N
LP001024	Male	Yes	2	Graduate	No	3200	700	70	360	1	Urban	Y
LP001027	Male	Yes	2	Graduate		2500	1840	109	360	1	Urban	Y
LP001028	Male	Yes	2	Graduate	No	3073	8106	200	360	1	Urban	Y
LP001029	Male	No	0	Graduate	No	1853	2840	114	360	1	Rural	N
LP001030	Male	Yes	2	Graduate	No	1299	1086	17	120	1	Urban	Y
LP001032	Male	No	0	Graduate	No	4950	0	125	360	1	Urban	Y

Fig.1: Dataset for loan approval prediction

Data Collection

The dataset used in this project includes:

1. **Demographic details** (age, gender, marital status, dependents).
2. **Financial information** (income, co-applicant income, credit history, existing debts, collateral value).
3. **Employment details** (employment type, work experience, job stability).
4. **Loan specifics** (loan amount, loan term, purpose of loan, property area, interest rate).

Data Analysis

Data analysis is a crucial phase in the loan approval project as it helps uncover patterns and trends that influence whether a loan application is accepted or rejected. By carefully analyzing the dataset, we can identify the most relevant features, detect correlations among financial and demographic attributes, and recognize potential issues such as imbalance or bias in the data.

We use Python libraries such as **Matplotlib, Seaborn, Numpy, and Pandas** to visualize trends—for example, the relationship between credit history and loan status, or income and loan approval. This analysis helps us understand feature importance and guides us in building an accurate and fair predictive model.

Data Preprocessing

Data preprocessing plays a key role in developing a model that is accurate, reliable, and efficient. The raw dataset is often inconsistent, so preprocessing ensures quality input for the model. The steps include:

- **Handling Missing Values**: Imputing or removing missing/null values.
- **Encoding Categorical Variables**: Converting features like gender, marital status, and property area into numerical form using label encoding/one-hot encoding.
- **Feature Scaling**: Standardization or Min-Max normalization is applied to bring all numerical attributes (e.g., income, loan amount) onto a comparable scale.
- **Splitting Dataset**: The dataset is divided into training and testing sets (typically 80:20) to evaluate model accuracy.

Model Selection

Choosing the right algorithms is critical for the loan approval prediction task. After analyzing the dataset, we test multiple classification algorithms and compare their performance.

The models used include:

- **K-Nearest Neighbors (KNN)**
- **Decision Tree Classifier**
- **Random Forest Classifier**
- **Logistic Regression**
- **Naive Bayes (GaussianNB)**
- **Support Vector Machine (SVM – Linear)**

After evaluation, the **Random Forest Classifier** generally provides the best accuracy and robustness for loan prediction.

Loan Prediction System

The system predicts whether a loan will be **Approved** or **Rejected** based on applicant information.

Input Features:

1. **Applicant Information**:
    - Age
    - Gender (Male/Female)
    - Marital Status (Single/Married)
    - Dependents (Number of dependents)
2. **Employment Details**:
    - Employment Type (Salaried / Self-employed / Unemployed)
    - Work Experience (Years of experience)
    - Job Stability (Duration in current job)
3. **Financial Information**:

- Applicant Income
- Co-Applicant Income
- Loan Amount Requested
- Loan Term (in months/years)
- Existing Debts
- Credit History (Good/Bad or numeric credit score)

4. **Property & Collateral Details**:
   - Property Area (Urban / Semi-Urban / Rural)
   - Collateral Value
5. **Loan Details**:
   - Purpose of Loan (Home, Vehicle, Education, Business, etc.)
   - Proposed Interest Rate (if available)

Model Evaluation

To evaluate the models, we use metrics from **scikit-learn**:

- **Confusion Matrix**: Shows true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN).
- **Classification Report**: Provides **Precision, Recall, and F1-Score**.
  - **Precision**: Ability of the classifier to correctly identify approved loans.

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

  - **Recall**: Ability to capture all actual approved loans.

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

  - **F1 Score**: Harmonic mean of precision and recall.

$$\text{F1 score} = \frac{\text{TP}}{\text{TP} + \frac{1}{2}(\text{FP} + \text{FN})}$$

- **Accuracy Score**: Ratio of correct predictions to total predictions.

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

Algorithms Used

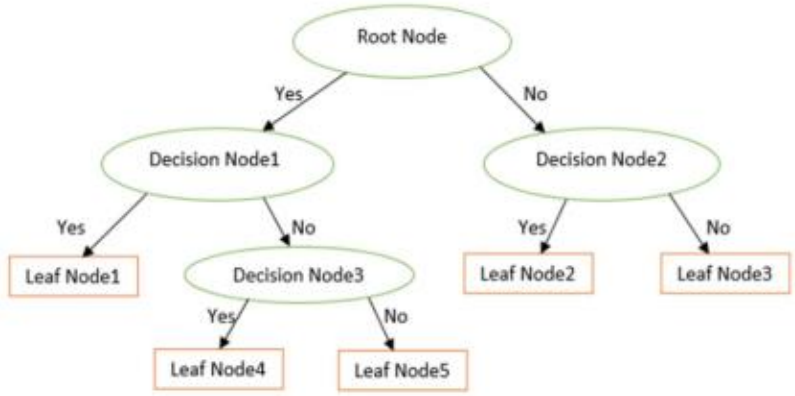
1. K-Nearest Neighbor (KNN)

- Classifies loan applications based on similarity to nearest neighbors in the dataset.
- Works well for simple datasets but can be slow with large data.

$$Euclidean\ distance = d(b,a) = \sqrt{\sum_{i=1}^{n}(b_i - a_i)^2}$$

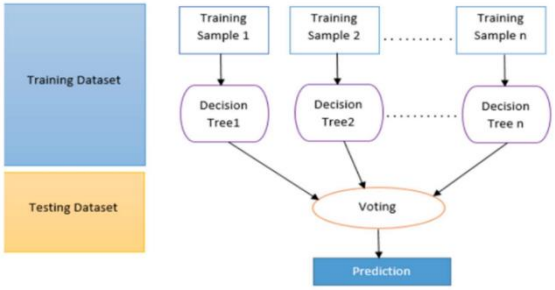
2. Decision Tree Classifier

- Builds a tree-like model where nodes represent attributes (e.g., credit history), and branches represent decision rules.
- Easy to interpret but prone to overfitting.



3. Random Forest Classifier

- Ensemble of decision trees, improves accuracy through majority voting.
- Handles missing values, avoids overfitting, and works well for structured financial datasets.

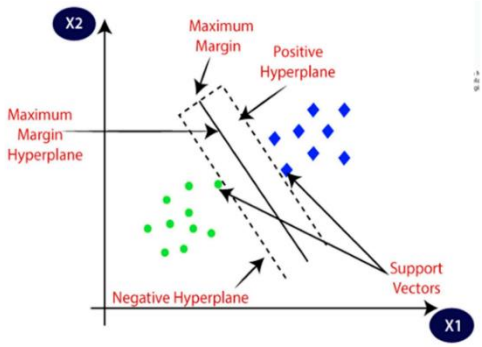


4. Logistic Regression

- Suitable for binary classification (Approved/Rejected).
- Provides probability scores for loan approval decisions.

5. Support Vector Machine (SVM – Linear)

- Finds the best decision boundary (hyperplane) to separate approved and rejected loans.
- Works well with high-dimensional data.



6. Naïve Bayes

- Based on Bayes' Theorem, assumes independence between features.
- Useful for fast predictions but can struggle with correlated variables like income and loan amount.

Cross-Validation

To ensure the model generalizes well and avoids overfitting, we use **K-Fold Cross-Validation**.

- The dataset is divided into **K subsets** (typically K=5 or 10).
- Each fold is used once as a test set while the others are used for training.
- Final performance is averaged across all folds.

RESULTS AND DISCUSSIONS

```
[ ] import numpy as np
    import pandas as pd
    import matplotlib.pyplot as plt
    import seaborn as sns
    from sklearn.model_selection import train_test_split
    from sklearn.metrics import confusion_matrix, classification_report,accuracy_score
    import warnings
    warnings.filterwarnings('ignore')
```

```
data=pd.read_csv("/content/loan.csv")
data=data.dropna(how='any', inplace=False)
print(data)
```

```
      Loan_ID  Gender Married Dependents     Education Self_Employed  \
1    LP001003    Male     Yes          1      Graduate            No
2    LP001005    Male     Yes          0      Graduate           Yes
3    LP001006    Male     Yes          0  Not Graduate            No
4    LP001008    Male      No          0      Graduate            No
5    LP001011    Male     Yes          2      Graduate           Yes
..        ...     ...     ...        ...           ...           ...
609  LP002978  Female      No          0      Graduate            No
610  LP002979    Male     Yes         3+      Graduate            No
611  LP002983    Male     Yes          1      Graduate            No
612  LP002984    Male     Yes          2      Graduate            No
613  LP002990  Female      No          0      Graduate           Yes

     ApplicantIncome  CoapplicantIncome  LoanAmount  Loan_Amount_Term  \
1               4583             1508.0       128.0             360.0
2               3000                0.0        66.0             360.0
3               2583             2358.0       120.0             360.0
4               6000                0.0       141.0             360.0
5               5417             4196.0       267.0             360.0
..               ...                ...         ...               ...
609             2900                0.0        71.0             360.0
610             4106                0.0        40.0             180.0
611             8072              240.0       253.0             360.0
612             7583                0.0       187.0             360.0
613             4583                0.0       133.0             360.0
```

```
     Credit_History Property_Area Loan_Status
1               1.0         Rural           N
2               1.0         Urban           Y
3               1.0         Urban           Y
4               1.0         Urban           Y
5               1.0         Urban           Y
..              ...           ...         ...
609             1.0         Rural           Y
610             1.0         Rural           Y
611             1.0         Urban           Y
612             1.0         Urban           Y
613             0.0     Semiurban           N

[480 rows x 13 columns]
```

```
data=data.drop(["Education", "CoapplicantIncome", "Loan_ID"], axis=1)
print(data)
```

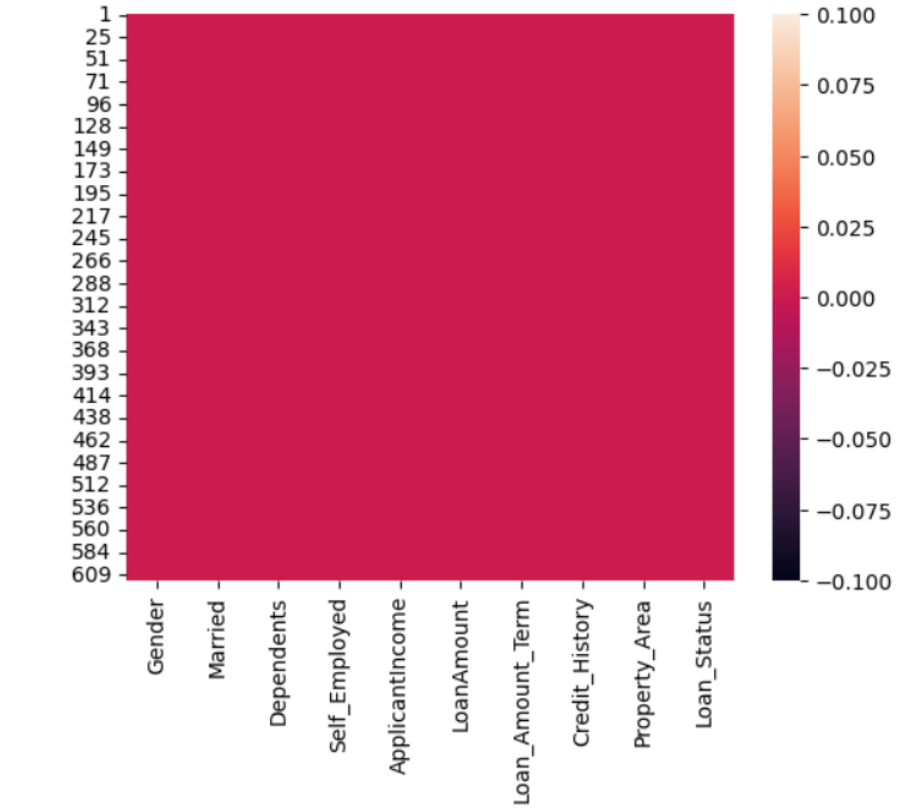
```
     Gender Married Dependents Self_Employed  ApplicantIncome  LoanAmount  \
1      Male     Yes          1            No             4583       128.0
2      Male     Yes          0           Yes             3000        66.0
3      Male     Yes          0            No             2583       120.0
4      Male      No          0            No             6000       141.0
5      Male     Yes          2           Yes             5417       267.0
..      ...     ...        ...           ...              ...         ...
609  Female      No          0            No             2900        71.0
610    Male     Yes         3+            No             4106        40.0
611    Male     Yes          1            No             8072       253.0
612    Male     Yes          2            No             7583       187.0
613  Female      No          0           Yes             4583       133.0

     Loan_Amount_Term  Credit_History Property_Area Loan_Status
1               360.0             1.0         Rural           N
2               360.0             1.0         Urban           Y
3               360.0             1.0         Urban           Y
4               360.0             1.0         Urban           Y
5               360.0             1.0         Urban           Y
..                ...             ...           ...         ...
609             360.0             1.0         Rural           Y
610             180.0             1.0         Rural           Y
611             360.0             1.0         Urban           Y
612             360.0             1.0         Urban           Y
613             360.0             0.0     Semiurban           N

[480 rows x 10 columns]
```

```
[ ] data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 480 entries, 1 to 613
Data columns (total 10 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   Gender             480 non-null    object 
 1   Married            480 non-null    object 
 2   Dependents         480 non-null    object 
 3   Self_Employed      480 non-null    object 
 4   ApplicantIncome    480 non-null    int64  
 5   LoanAmount         480 non-null    float64
 6   Loan_Amount_Term   480 non-null    float64
 7   Credit_History     480 non-null    float64
 8   Property_Area      480 non-null    object 
 9   Loan_Status        480 non-null    object 
dtypes: float64(3), int64(1), object(6)
memory usage: 41.2+ KB
```

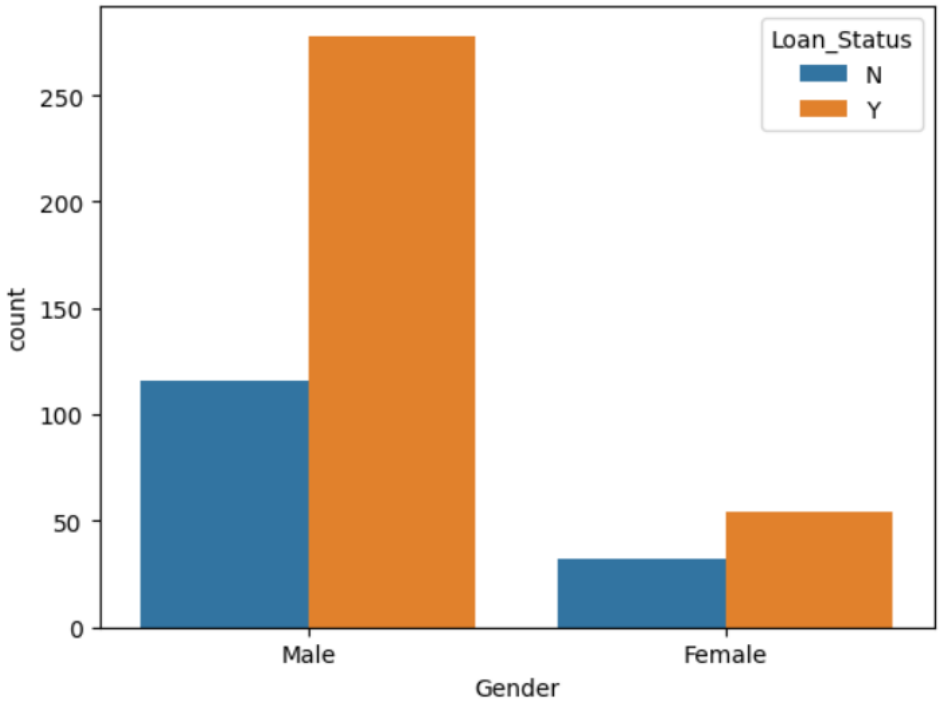
```
sns.heatmap(data.isnull())
```

```
<Axes: >
```



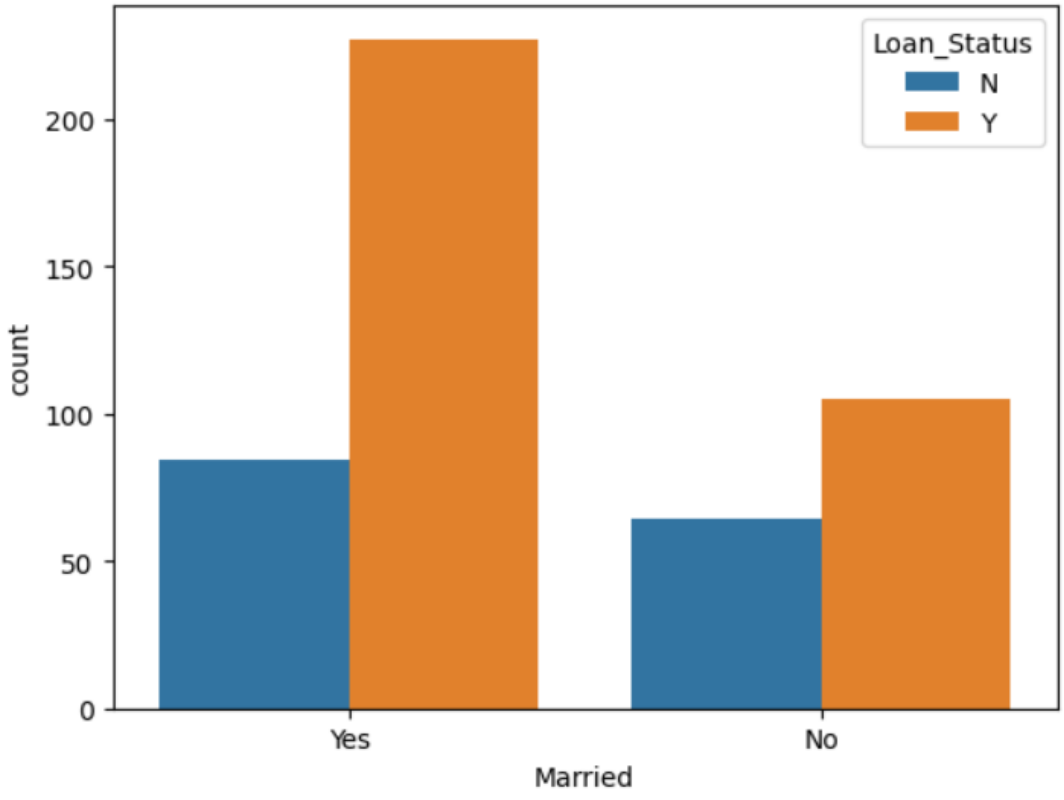
```
sns.countplot(x='Gender', data=data, hue='Loan_Status')
```

```
<Axes: xlabel='Gender', ylabel='count'>
```



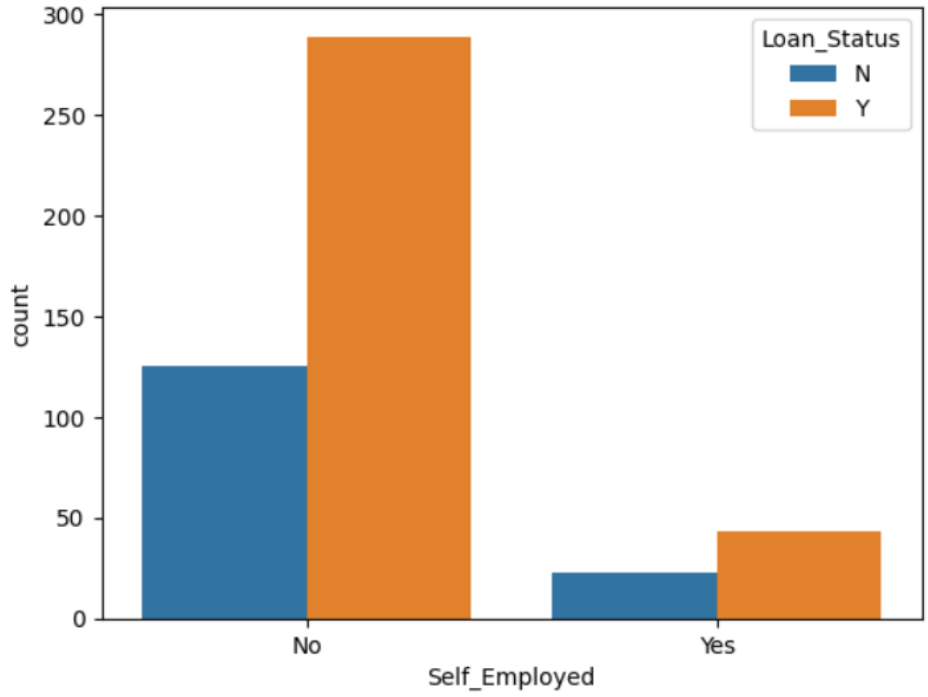

```
sns.countplot(x='Married', data=data, hue='Loan_Status')
```

```
<Axes: xlabel='Married', ylabel='count'>
```



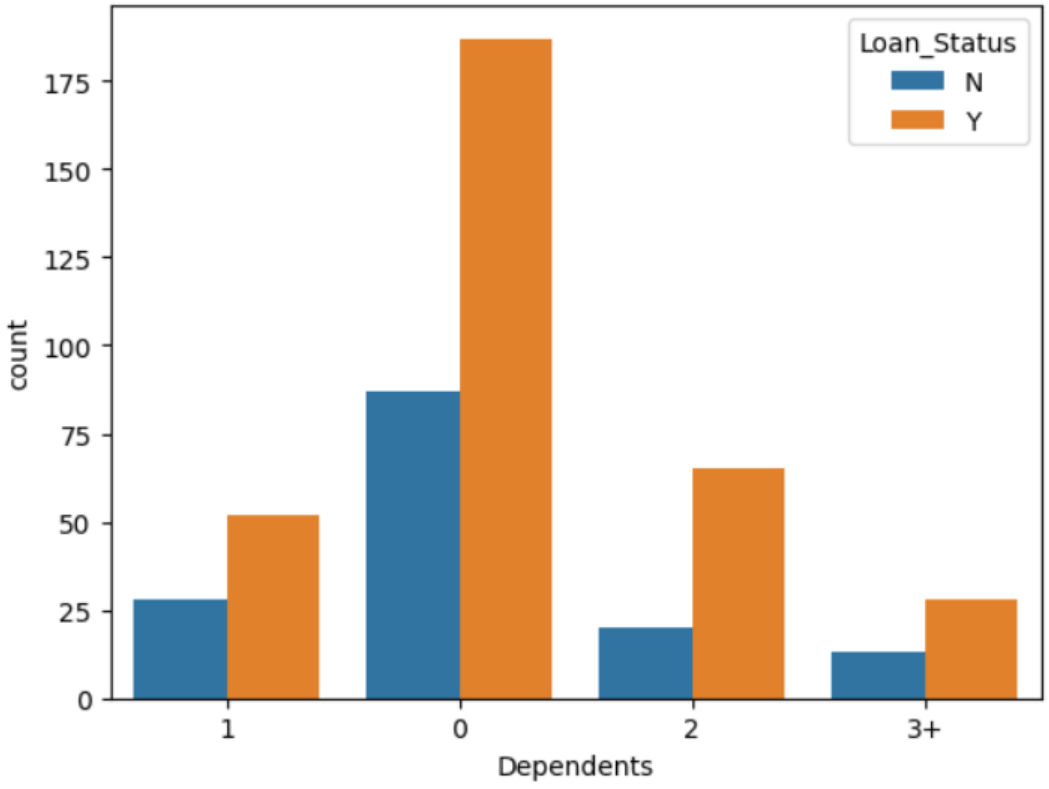
```
sns.countplot(x='Self_Employed', data=data, hue='Loan_Status')
```

```
<Axes: xlabel='Self_Employed', ylabel='count'>
```



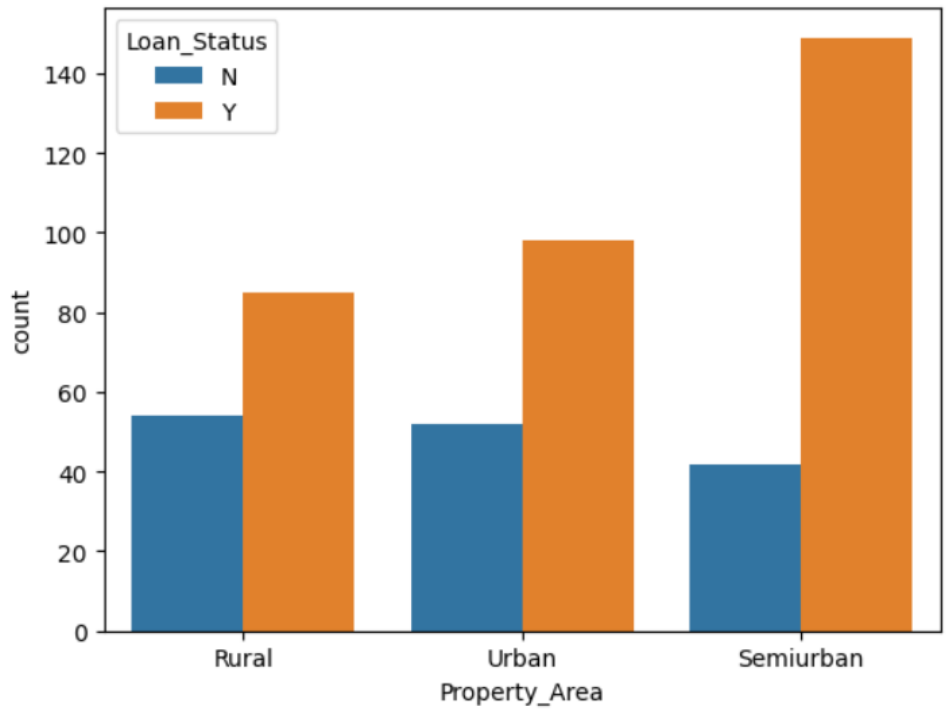

```
sns.countplot(x='Dependents', data=data, hue='Loan_Status')
```

```
<Axes: xlabel='Dependents', ylabel='count'>
```



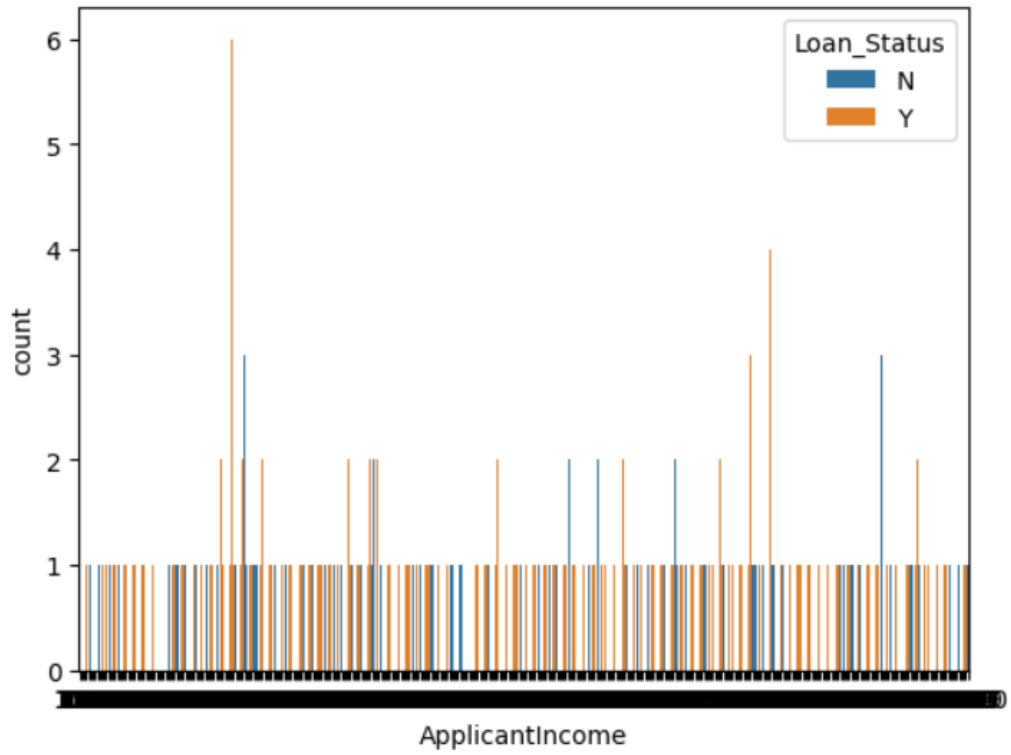
```
sns.countplot(x='Property_Area', data=data, hue='Loan_Status')
```

```
<Axes: xlabel='Property_Area', ylabel='count'>
```



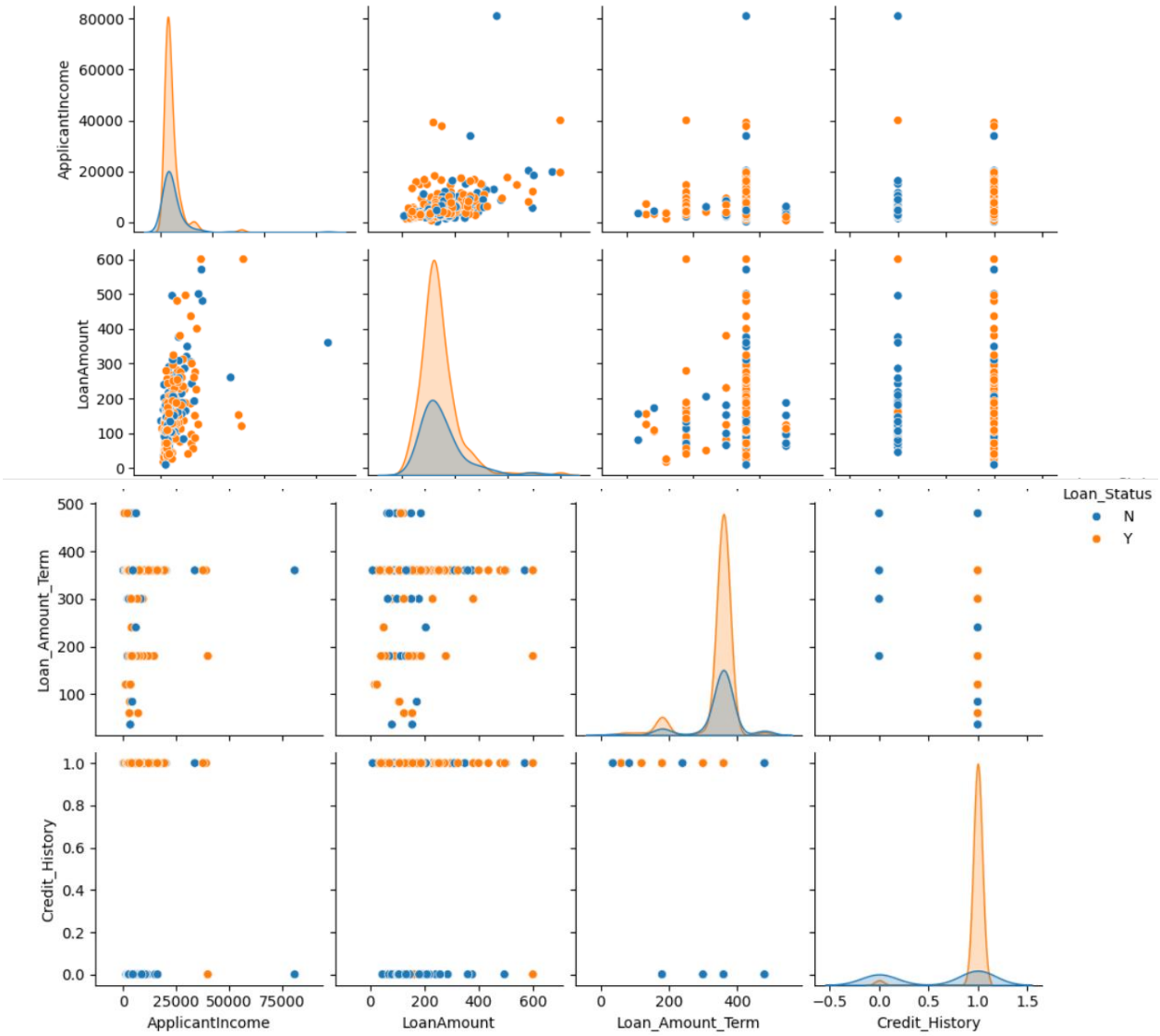

```
sns.countplot(x='ApplicantIncome', data=data, hue='Loan_Status')
```

```
<Axes: xlabel='ApplicantIncome', ylabel='count'>
```



```
sns.pairplot(data, hue='Loan_Status')
```

```
<seaborn.axisgrid.PairGrid at 0x7c21899437d0>
```




```
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
data['Gender']=le.fit_transform(data['Gender'])
data['Married']=le.fit_transform(data['Married'])
data['Dependents']=le.fit_transform(data['Dependents'])
data['Self_Employed']=le.fit_transform(data['Self_Employed'])
data['Property_Area']=le.fit_transform(data['Property_Area'])
data['Loan_Status']=le.fit_transform(data['Loan_Status'])
print(data)
```

```
     Gender  Married  Dependents  Self_Employed  ApplicantIncome  LoanAmount  \
1         1        1           1              0             4583       128.0
2         1        1           0              1             3000        66.0
3         1        1           0              0             2583       120.0
4         1        0           0              0             6000       141.0
5         1        1           2              1             5417       267.0
..      ...      ...         ...            ...              ...         ...
609       0        0           0              0             2900        71.0
610       1        1           3              0             4106        40.0
611       1        1           1              0             8072       253.0
612       1        1           2              0             7583       187.0
613       0        0           0              1             4583       133.0
```

	Loan_Amount_Term	Credit_History	Property_Area	Loan_Status
1	360.0	1.0	0	0
2	360.0	1.0	2	1
3	360.0	1.0	2	1
4	360.0	1.0	2	1
5	360.0	1.0	2	1
..	...	...	...	...
609	360.0	1.0	0	1
610	180.0	1.0	0	1
611	360.0	1.0	2	1
612	360.0	1.0	2	1
613	360.0	0.0	1	0

[480 rows x 10 columns]

Data Split to dependent and independent

```
[ ] y=data.iloc[:,0].values
    x=data.iloc[:,1:].values
    print(x)
```

```
[[1. 1. 0. ... 1. 0. 0.]
 [1. 0. 1. ... 1. 2. 1.]
 [1. 0. 0. ... 1. 2. 1.]
 ...
 [1. 1. 0. ... 1. 2. 1.]
 [1. 2. 0. ... 1. 2. 1.]
 [0. 0. 1. ... 0. 1. 0.]]
```

Data Sampling

```
[ ] from sklearn.model_selection import train_test_split
    x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=0)
    print(x_train)
```

```
[[1. 3. 0. ... 1. 0. 1.]
 [0. 0. 0. ... 1. 0. 0.]
 [1. 1. 0. ... 1. 2. 1.]
 ...
 [0. 0. 0. ... 1. 2. 0.]
 [1. 0. 0. ... 1. 2. 1.]
 [1. 0. 0. ... 0. 2. 0.]]
```

KNN Algorithm

```
from sklearn.neighbors import KNeighborsClassifier
knn=KNeighborsClassifier()
knn.fit(x_train,y_train)
```

```
KNeighborsClassifier
KNeighborsClassifier()
```

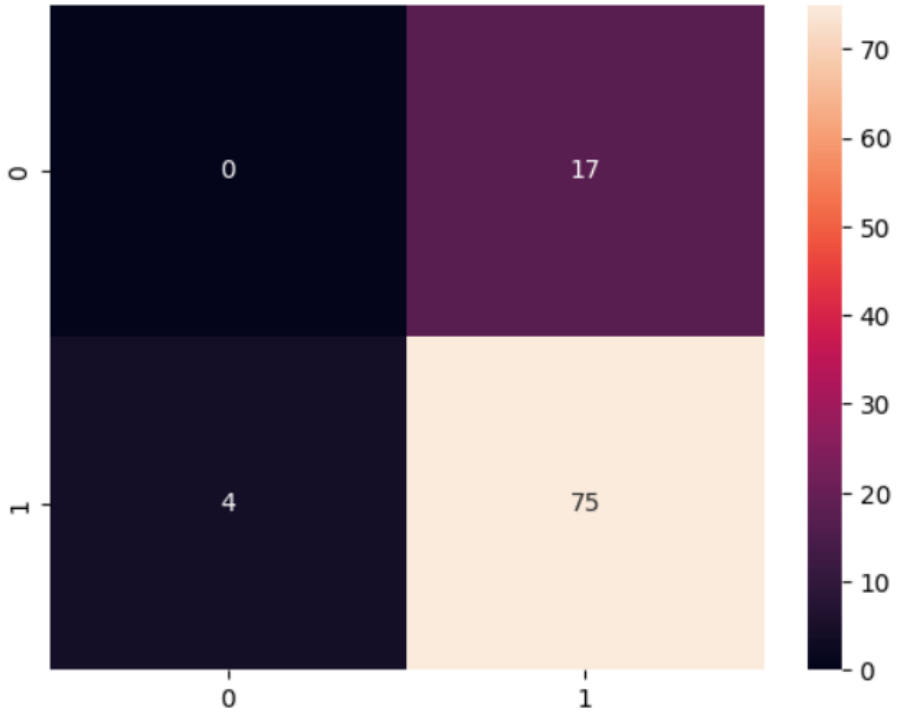
```
[ ] knn_pred=knn.predict(x_test)
```

```
[ ] acknn=accuracy_score(y_test,knn_pred)
    print(acknn)
```

```
0.78125
```

```
cmknn=confusion_matrix(y_test,knn_pred)
sns.heatmap(cmknn,annot=True)
```

```
<Axes: >
```




```
crknn=classification_report(y_test,knn_pred)
print(crknn)
```

```
              precision    recall  f1-score   support

           0       0.00      0.00      0.00        17
           1       0.82      0.95      0.88        79

    accuracy                           0.78        96
   macro avg       0.41      0.47      0.44        96
weighted avg       0.67      0.78      0.72        96
```

LOGISTIC REGRESSION

```
from sklearn.linear_model import LogisticRegression
logm=LogisticRegression()
logm.fit(x_train,y_train)
```

```
LogisticRegression
LogisticRegression()
```

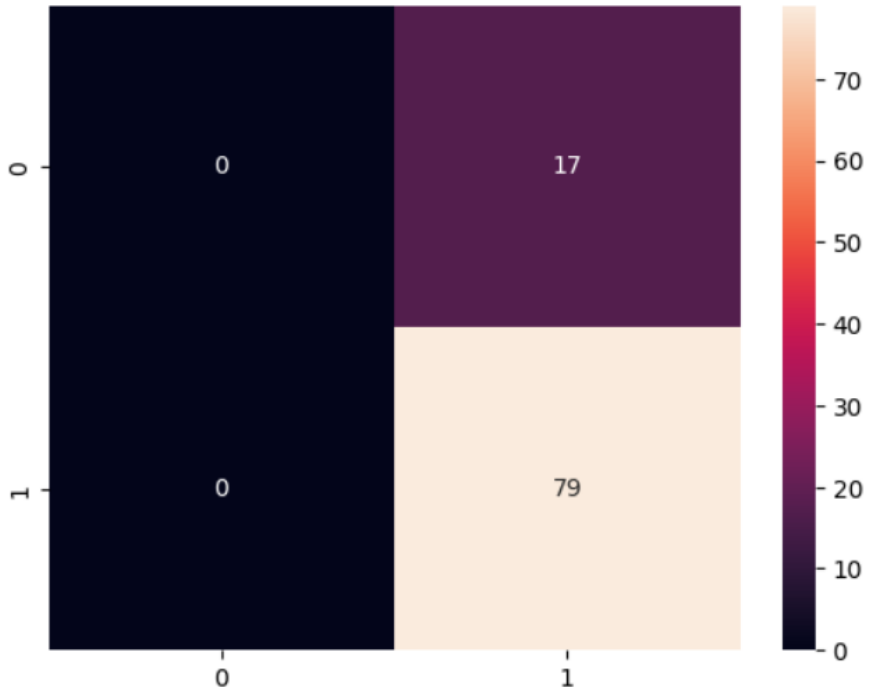
```
logm_pred=logm.predict(x_test)
```

```
aclogm=accuracy_score(y_test,logm_pred)
print(aclogm)
```

```
0.8229166666666666
```

```
cmlogm=confusion_matrix(y_test,logm_pred)
sns.heatmap(cmlogm,annot=True)
```

```
<Axes: >
```



```
crlogm=classification_report(y_test,logm_pred)
print(crlogm)
```

```
              precision    recall  f1-score   support

           0       0.00      0.00      0.00        17
           1       0.82      1.00      0.90        79

    accuracy                           0.82        96
   macro avg       0.41      0.50      0.45        96
weighted avg       0.68      0.82      0.74        96
```

NAIVE BAYES ALGORITHM

```
from sklearn.naive_bayes import GaussianNB
nb=GaussianNB()
nb.fit(x_train,y_train)
```

```
GaussianNB
GaussianNB()
```

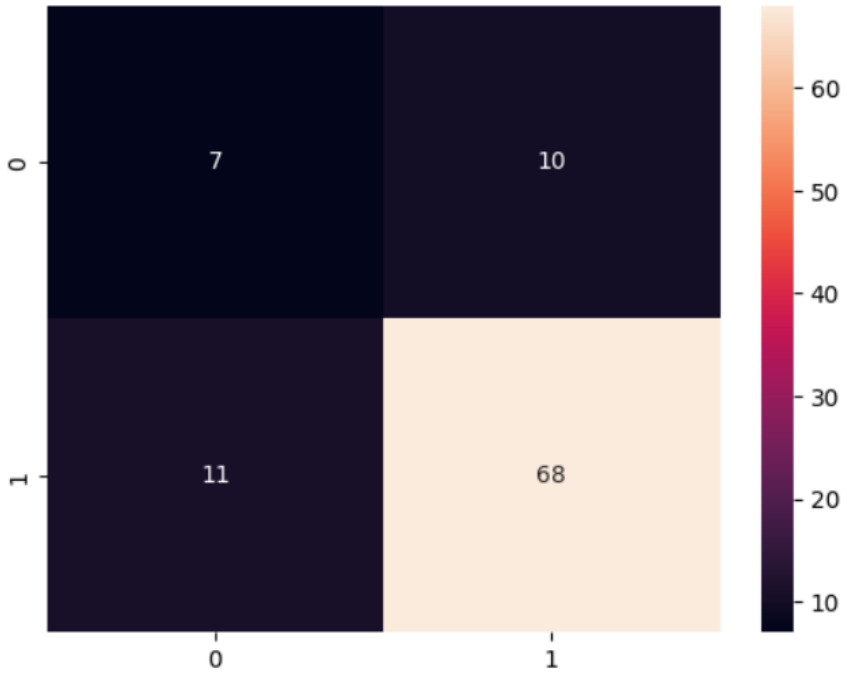
```
nb_pred=nb.predict(x_test)
```

```
acnb=accuracy_score(y_test,nb_pred)
print(acnb)
```

```
0.78125
```

```
cmnb=confusion_matrix(y_test,nb_pred)
sns.heatmap(cmnb,annot=True)
```

```
<Axes: >
```



```
[ ] crnb=classification_report(y_test,nb_pred)
    print(crnb)
```

```
              precision    recall  f1-score   support

           0       0.39      0.41      0.40        17
           1       0.87      0.86      0.87        79

    accuracy                           0.78        96
   macro avg       0.63      0.64      0.63        96
weighted avg       0.79      0.78      0.78        96
```

Support Vector Machine

```
[ ] from sklearn.svm import SVC
    svmmodel=SVC(kernel='linear')
    svmmodel.fit(x_train,y_train)
```

```
SVC(kernel='linear')
```

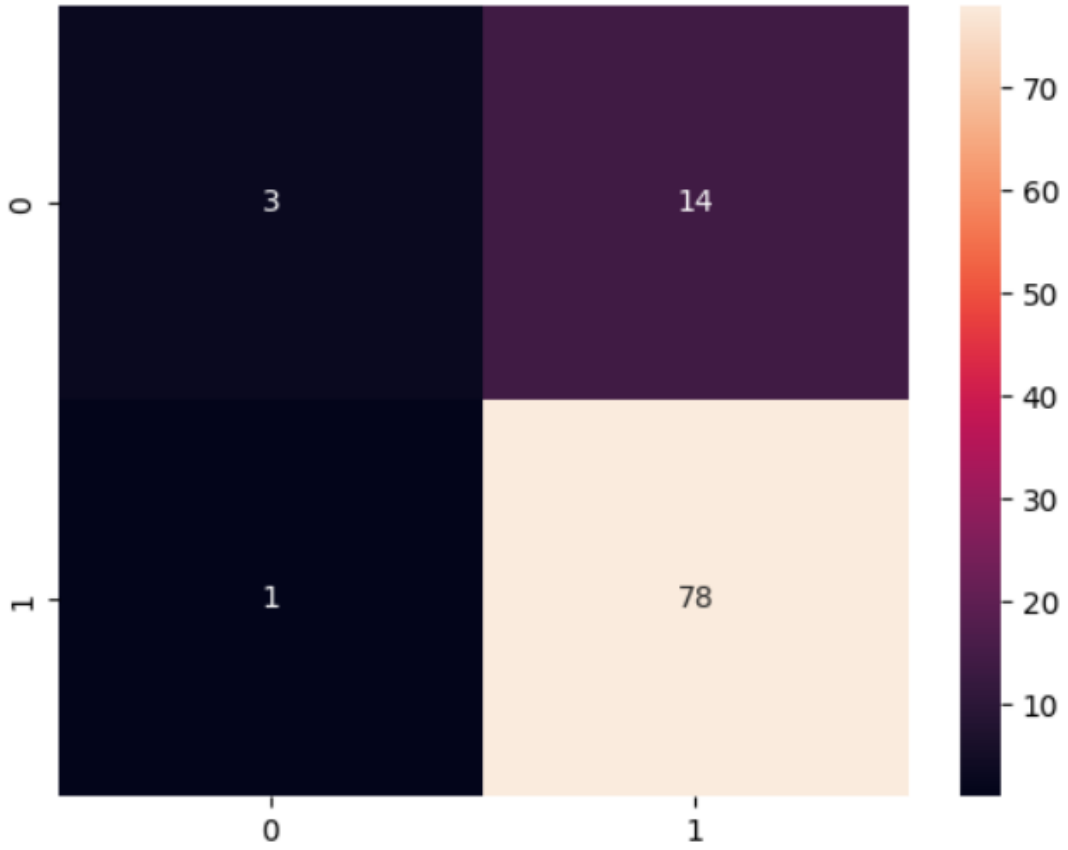
```
[ ] svm_pred=svmmodel.predict(x_test)
```

```
acsvm=accuracy_score(y_test,svm_pred)
print(acsvm)
```

```
0.84375
```

```
cmsvm=confusion_matrix(y_test,svm_pred)
sns.heatmap(cmsvm,annot=True)
```

```
<Axes: >
```




```
[ ] crsvm=classification_report(y_test,svm_pred)
    print(crsvm)
```

```
              precision    recall  f1-score   support

           0       0.75      0.18      0.29        17
           1       0.85      0.99      0.91        79

    accuracy                           0.84        96
   macro avg       0.80      0.58      0.60        96
weighted avg       0.83      0.84      0.80        96
```

Decision Tree Classifier

```
[ ] from sklearn.tree import DecisionTreeClassifier
    dt=DecisionTreeClassifier()
    dt.fit(x_train,y_train)
```

```
DecisionTreeClassifier
DecisionTreeClassifier()
```

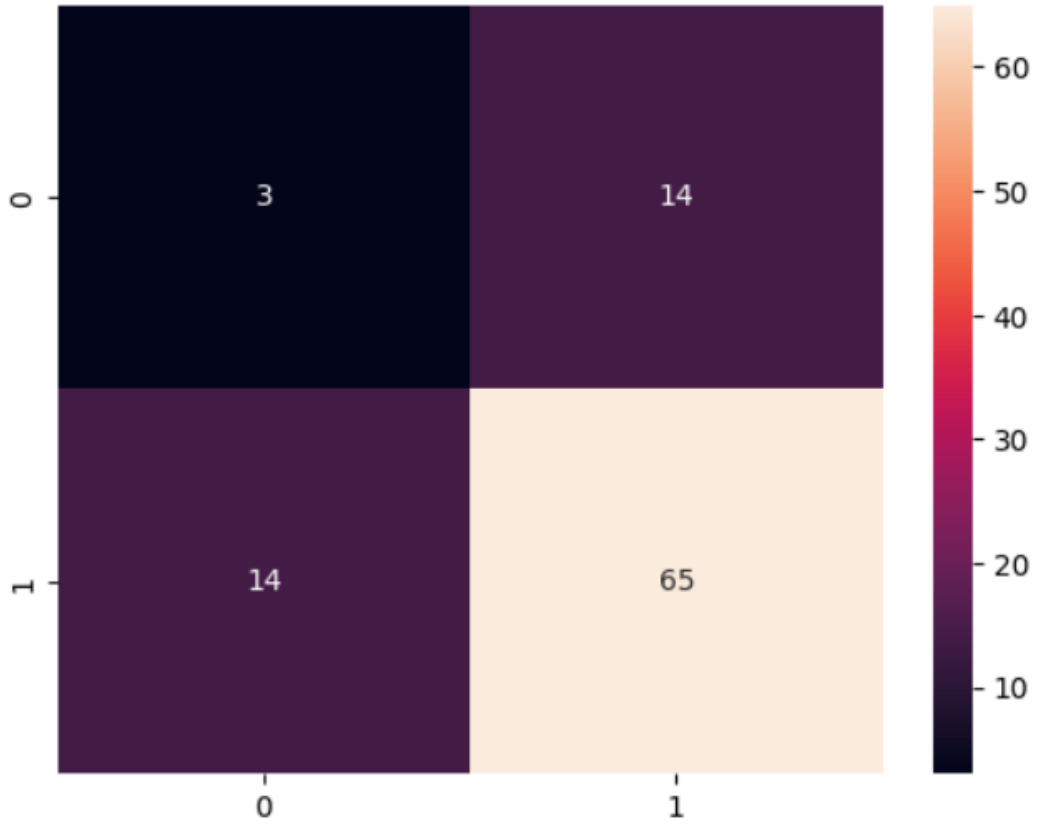
```
[ ] dt_pred=dt.predict(x_test)
```

```
acdt=accuracy_score(y_test,dt_pred)
print(acdt)
```

```
0.7083333333333334
```

```
cmdt=confusion_matrix(y_test,dt_pred)
sns.heatmap(cmdt,annot=True)
```

```
<Axes: >
```



```
crdt=classification_report(y_test,dt_pred)
print(crdt)
```

```
              precision    recall  f1-score   support

           0       0.18      0.18      0.18        17
           1       0.82      0.82      0.82        79

    accuracy                           0.71        96
   macro avg       0.50      0.50      0.50        96
weighted avg       0.71      0.71      0.71        96
```

Random Forest Algorithm=Ensemble Learning

```
from sklearn.ensemble import RandomForestClassifier
rf=RandomForestClassifier(n_estimators=10)
rf.fit(x_train,y_train)
```

```
RandomForestClassifier
RandomForestClassifier(n_estimators=10)
```

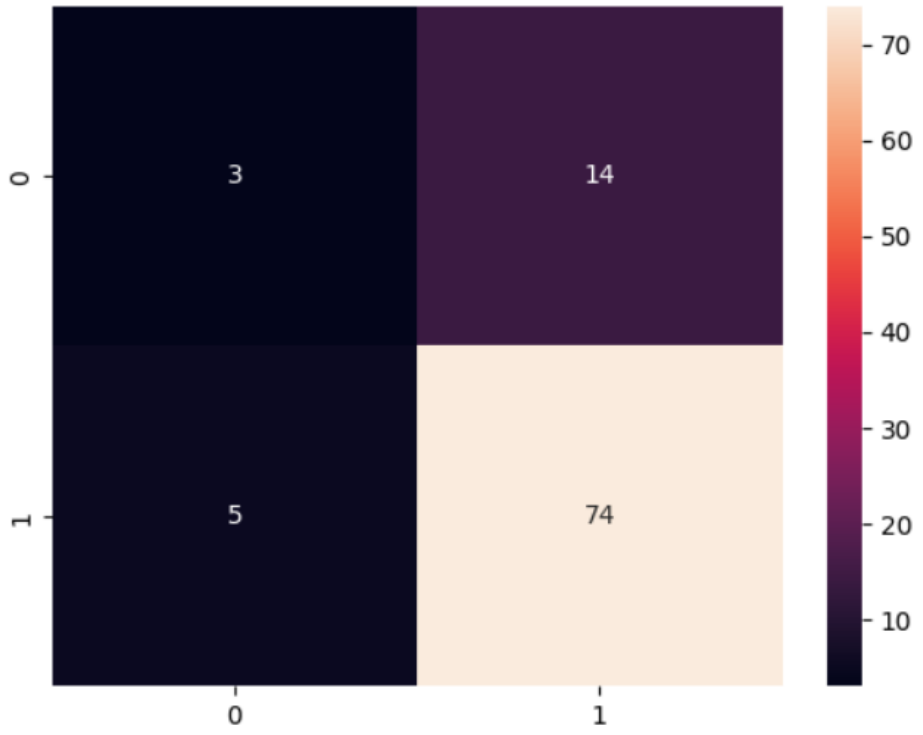
```
rf_pred=rf.predict(x_test)
```

```
acrf=accuracy_score(y_test,rf_pred)
print(acrf)
```

```
0.8020833333333334
```

```
cmrf=confusion_matrix(y_test,rf_pred)
sns.heatmap(cmrf,annot=True)
```

```
<Axes: >
```




```
[ ] crrf=classification_report(y_test,rf_pred)
    print(crrf)
```

	precision	recall	f1-score	support
0	0.38	0.18	0.24	17
1	0.84	0.94	0.89	79
accuracy			0.80	96
macro avg	0.61	0.56	0.56	96
weighted avg	0.76	0.80	0.77	96

CONCLUSION

This project successfully developed a **loan prediction system** using machine learning algorithms, including **Logistic Regression, Naive Bayes, Decision Trees, Random Forest, Support Vector Machines (SVM), and K-Nearest Neighbours (KNN)**.

Among the tested algorithms:

- **Linear SVM** achieved the highest prediction accuracy of **84%**, making it the most reliable model for this dataset.
- **Random Forest** also demonstrated strong performance due to its robustness in handling non-linear relationships and reducing overfitting.
- **Naive Bayes** excelled with categorical data, while **Decision Trees** provided intuitive visualizations for decision-making.
- **KNN**, while simple to implement, faced challenges with high-dimensional data and performed comparatively lower.

This model provides a valuable, data-driven tool for financial institutions to:

- Streamline the loan approval process.
- Reduce manual effort and minimize risks.
- Ensure fair and transparent decision-making.
- Improve overall efficiency and customer satisfaction.

However, the system's accuracy depends heavily on the **quality and completeness of the input data**.

Future Work

- Incorporating **real-time data sources** (e.g., live updates on credit scores and employment status) to enhance system accuracy and reliability.
- Expanding the scope to predict **loan defaults** or **interest rate eligibility** for broader utility.
- Implementing **Explainable AI (XAI)** techniques to improve transparency and foster trust among end users.

Final Remark

In conclusion, this project demonstrates the potential of **machine learning to revolutionize the loan approval process**, offering a **scalable and efficient solution** for modern financial institutions.

LIMITATIONS OF THE WORK

- **Limited Dataset Size**: The loan dataset used contains a relatively small number of records, which may not fully capture the diversity of applicants across different regions, banks, and financial institutions.
- **Static Dataset**: The model is trained on a fixed dataset and does not automatically update with new loan applications, changing financial regulations, or evolving economic conditions.
- **Binary Classification Only**: The system predicts only two outcomes (Loan Approved / Loan Rejected) without providing additional insights such as loan amount eligibility, interest rate prediction, or repayment capacity.
- **Basic Machine Learning Models**: While effective, traditional models like Random Forest and SVM were used; more advanced techniques such as deep learning, gradient boosting, or ensemble stacking were not explored.
- **Simplified Financial Parameters**: The system considers major parameters like income, credit history, and loan amount but does not include other important factors such as spending habits, detailed banking history, or behavioral data.
- **Potential for Misclassification**: Despite achieving good accuracy, there is still a possibility of false approvals (approving risky applicants) or false rejections (denying creditworthy applicants).
- **No Real-Time Data Integration**: The current system relies on static input features and does not integrate with live credit scoring APIs, financial databases, or real-time employment verification systems.
- **Web Deployment Constraints**: The deployed web application may face limitations related to server performance, data privacy, and handling multiple users simultaneously, especially under free or limited hosting environments.
- **No Regulatory Certification**: The system is intended for research and educational purposes only. It is not certified or validated as an official financial decision-making tool by banking regulators.

FUTURE SCOPE OF THE WORK

- **Expanding the Dataset**: Increasing the size and diversity of the dataset by including loan application data from different banks, regions, and financial institutions to improve model generalization.
- **Incorporating More Features**: Adding additional parameters such as spending behavior, transaction history, social credit scores, and behavioral analytics for more accurate predictions.
- **Real-Time Data Integration**: Connecting the system with live financial databases, credit bureau APIs, and employment verification systems to enable real-time decision-making.
- **Multi-Class Prediction**: Extending the system beyond binary outcomes (Approved/Rejected) to predict additional outcomes such as eligible loan amount, interest rate range, and repayment capacity.
- **Advanced Machine Learning Models**: Implementing deep learning techniques, gradient boosting (XGBoost, LightGBM), and ensemble methods to further boost prediction accuracy and handle complex financial patterns.
- **Mobile Application Development**: Developing a mobile app version of the system to allow applicants to check loan eligibility instantly through smartphones.
- **Personalized Financial Recommendations**: Providing applicants with tailored suggestions, such as improving credit history, reducing debts, or optimal loan types based on their financial profile.
- **Integration with Banks and NBFCs**: Collaborating with financial institutions to integrate the system into their loan approval workflows for faster and more reliable decision-making.
- **Data Privacy and Security Enhancement**: Strengthening security with encryption, secure authentication, and compliance with data protection laws (e.g., GDPR, RBI regulations) to protect sensitive financial data.
- **Continuous Learning System**: Developing an adaptive system where the model updates itself continuously as new loan application data and financial regulations become available.
- **Default Prediction**: Extending the system to also predict the probability of loan default, enabling banks to manage risks better.
- **User Login and Profile Management**: Introducing secure login functionality where users can create accounts, track past applications, and view personalized loan recommendations.

REFERENCES

- **Scikit-learn Documentation**
Available at: https://scikit-learn.org/
Description: Official documentation for the Scikit-learn library, used for implementing machine learning algorithms and performance metrics like Confusion Matrix, Classification Report, and Accuracy Score.

- **Kaggle Datasets**
Available at: https://www.kaggle.com/datasets
Description: Platform for publicly available datasets, including those related to loan prediction.

- **Python Official Documentation**
Available at: https://www.python.org/doc/
Description: Official Python documentation used for understanding core libraries such as NumPy, Pandas, and Matplotlib.