www.kiet.edu
Delhi-NCR, Ghaziabad

KIET GROUP OF INSTITUTIONS
Connecting Life with Learning

Assessment Report

on

"Predict Loan Default"

submitted as partial fulfillment for the award of

BACHELOR OF TECHNOLOGY DEGREE

SESSION 2024-25

in

CSE(AI -B)

By Group No. 8

Atharv Tayal (202401100300078)
Aviral Dixit (202401100300081)
Ayush Goyal (202401100300083)
Dhairya Goel (202401100300099)

Under the supervision of

"Mr. Shivansh Prashad"

KIET Group of Institutions, Ghaziabad

May, 2025

---

1. Introduction

In today’s lending industry, accurately assessing the risk of loan default is critical to maintaining the financial health of lending institutions. Loan default occurs when a borrower fails to meet the legal obligations of the loan agreement, which can lead to significant financial losses. With the increasing availability of financial data, machine learning offers powerful tools to improve risk assessment models by uncovering complex patterns that traditional methods might overlook.

This project focuses on building a predictive model to classify whether a loan applicant is likely to default. We use a variety of classification algorithms—specifically decision trees and ensemble techniques—to make predictions based on historical applicant and loan data. The primary goals of this project are:

- To develop a reliable machine learning model that can predict loan default.
- To compare different classification algorithms to determine which provides the best performance.
- To visualize feature importance and understand which factors most influence loan default.

By leveraging classification algorithms such as Decision Trees, Random Forests, and XGBoost, and analyzing feature importance, this project aims to aid financial institutions in making data-driven decisions to mitigate risk.

2. Problem Statement

To predict whether a borrower will default on a loan using available financial and credit history data. The classification will help lenders mitigate risk by identifying high-risk applicants.

3. Objectives

- Preprocess the dataset for training a machine learning model.
- Train a Logistic Regression model to classify loan defaults.
- Evaluate model performance using standard classification metrics.
- Visualize the confusion matrix using a heatmap for interpretability.

4. Methodology

- **Data Collection**: The user uploads a CSV file containing the dataset.
- **Data Preprocessing**:
    - Handling missing values using mean and mode imputation.
    - One-hot encoding of categorical variables.
    - Feature scaling using StandardScaler.

- **Model Building**:
    - Splitting the dataset into training and testing sets.
    - Training a Logistic Regression classifier.
- **Model Evaluation**:
    - Evaluating accuracy, precision, recall, and F1-score.
    - Generating a confusion matrix and visualizing it with a heatmap.

5. Data Preprocessing

The dataset is cleaned and prepared as follows:

- Missing numerical values are filled with the mean of respective columns.
- Categorical values are encoded using one-hot encoding.
- Data is scaled using StandardScaler to normalize feature values.
- The dataset is split into 80% training and 20% testing.

6. Model Implementation

Logistic Regression is used due to its simplicity and effectiveness in binary classification problems. The model is trained on the processed dataset and used to predict the loan default status on the test set.

7. Evaluation Metrics

The following metrics are used to evaluate the model:

- **Accuracy**: Measures overall correctness.
- **Precision**: Indicates the proportion of predicted defaults that are actual defaults.
- **Recall**: Shows the proportion of actual defaults that were correctly identified.
- **F1 Score**: Harmonic mean of precision and recall.
- **Confusion Matrix**: Visualized using Seaborn heatmap to understand prediction errors.

8. Results and Analysis

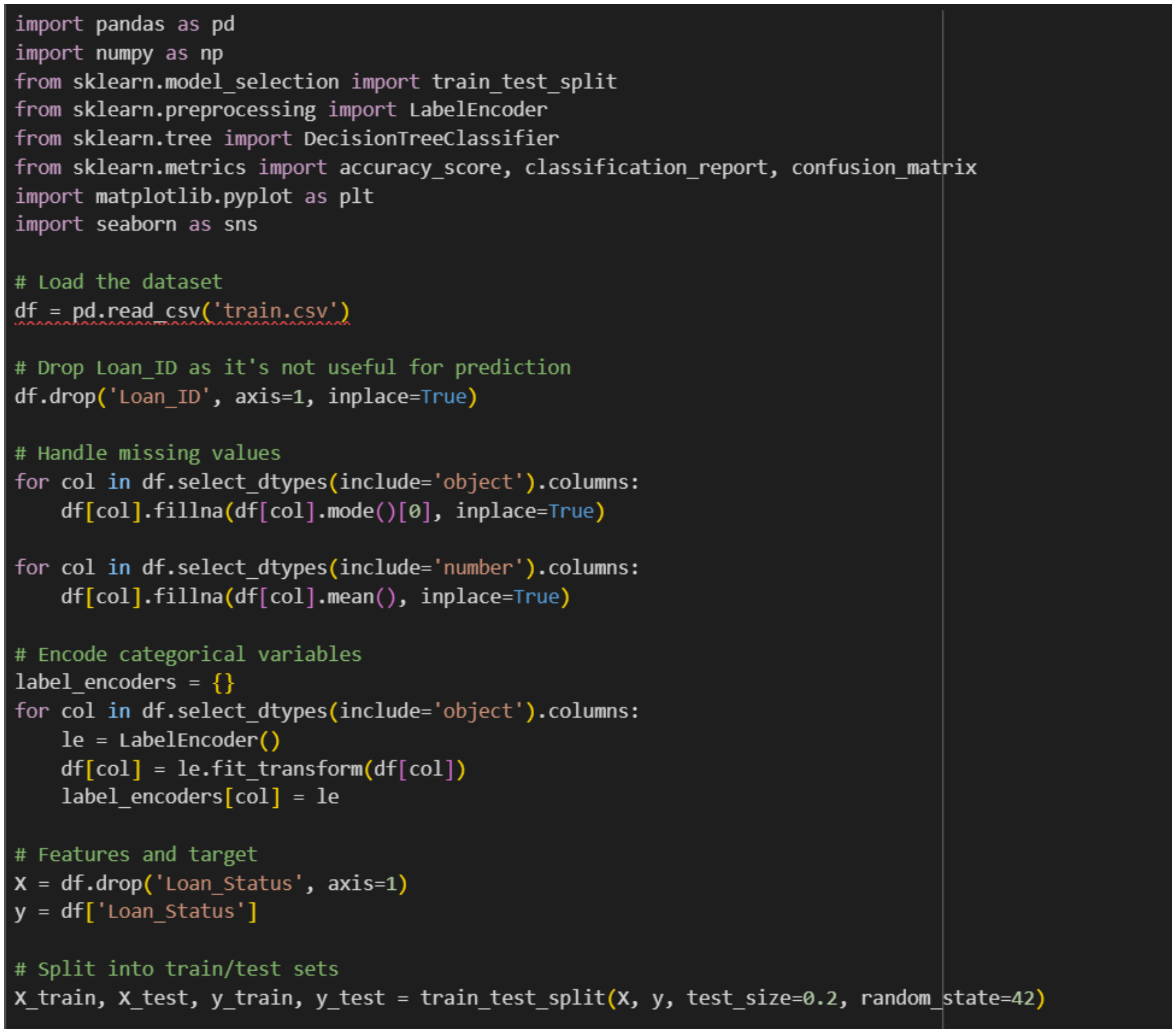
- The model provided reasonable performance on the test set.
- Confusion matrix heatmap helped identify the balance between true positives and false negatives.
- Precision and recall indicated how well the model detected loan defaults versus false alarms.

9. Conclusion

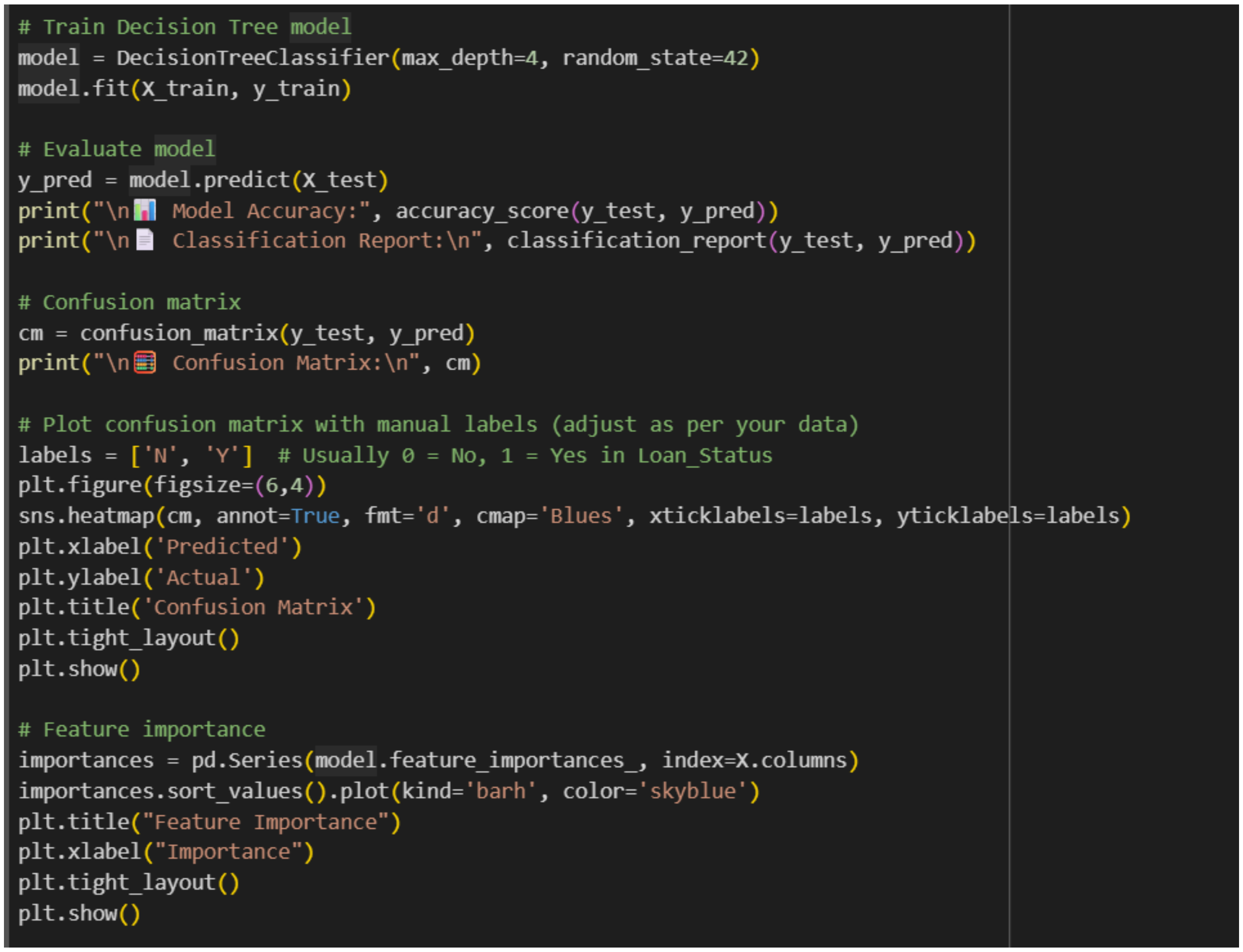
The logistic regression model successfully classified loan defaults with satisfactory performance metrics. The project demonstrates the potential of using machine learning for automating loan approval processes and improving risk assessment. However, improvements can be made by exploring more advanced models and handling imbalanced data.

10. References

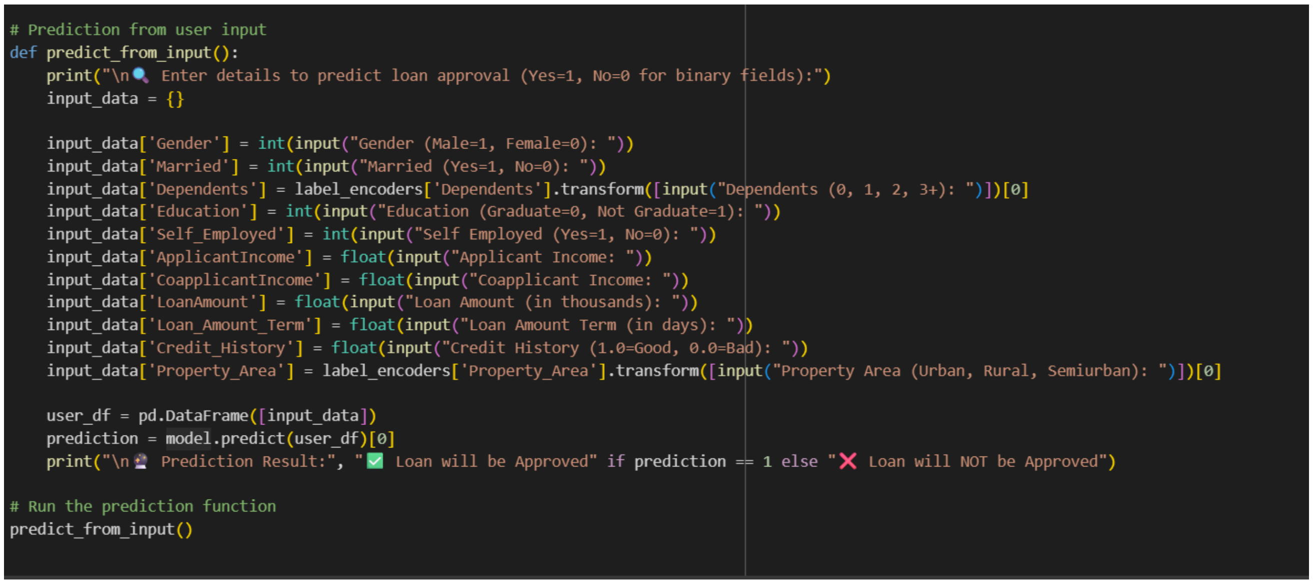
- scikit-learn documentation
- pandas documentation
- Seaborn visualization library
- Research articles on credit risk prediction



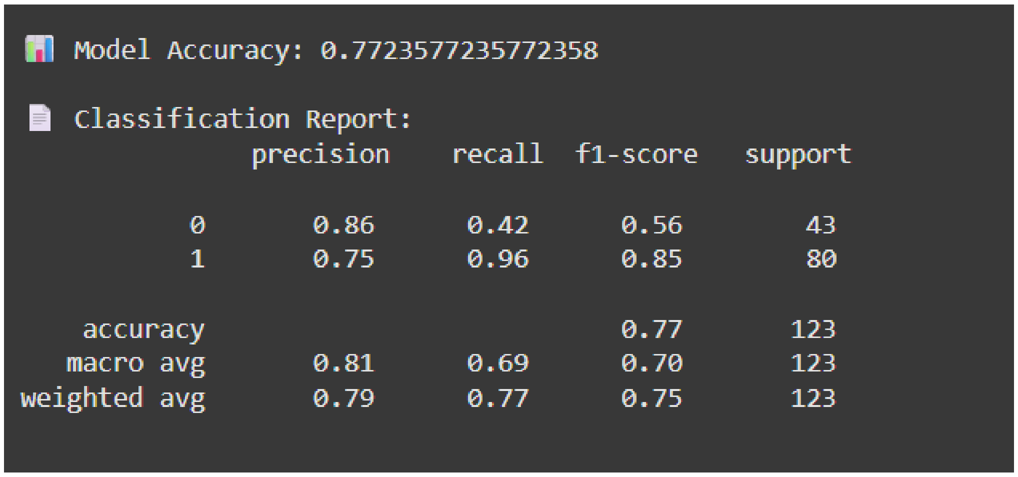
```python
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
import matplotlib.pyplot as plt
import seaborn as sns

# Load the dataset
df = pd.read_csv('train.csv')

# Drop Loan_ID as it's not useful for prediction
df.drop('Loan_ID', axis=1, inplace=True)

# Handle missing values
for col in df.select_dtypes(include='object').columns:
    df[col].fillna(df[col].mode()[0], inplace=True)

for col in df.select_dtypes(include='number').columns:
    df[col].fillna(df[col].mean(), inplace=True)

# Encode categorical variables
label_encoders = {}
for col in df.select_dtypes(include='object').columns:
    le = LabelEncoder()
    df[col] = le.fit_transform(df[col])
    label_encoders[col] = le

# Features and target
X = df.drop('Loan_Status', axis=1)
y = df['Loan_Status']

# Split into train/test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```



```python
# Train Decision Tree model
model = DecisionTreeClassifier(max_depth=4, random_state=42)
model.fit(X_train, y_train)

# Evaluate model
y_pred = model.predict(X_test)
print("\n📊 Model Accuracy:", accuracy_score(y_test, y_pred))
print("\n📄 Classification Report:\n", classification_report(y_test, y_pred))

# Confusion matrix
cm = confusion_matrix(y_test, y_pred)
print("\n🧮 Confusion Matrix:\n", cm)

# Plot confusion matrix with manual labels (adjust as per your data)
labels = ['N', 'Y']  # Usually 0 = No, 1 = Yes in Loan_Status
plt.figure(figsize=(6,4))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', xticklabels=labels, yticklabels=labels)
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix')
plt.tight_layout()
plt.show()

# Feature importance
importances = pd.Series(model.feature_importances_, index=X.columns)
importances.sort_values().plot(kind='barh', color='skyblue')
plt.title("Feature Importance")
plt.xlabel("Importance")
plt.tight_layout()
plt.show()
```

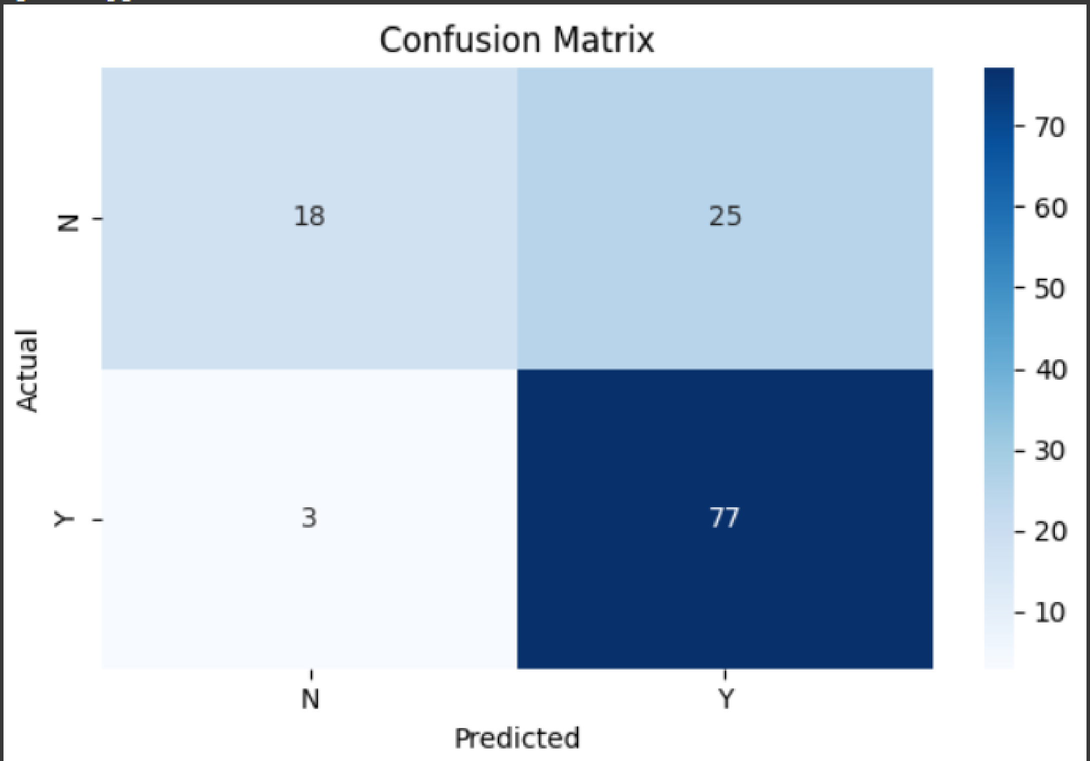


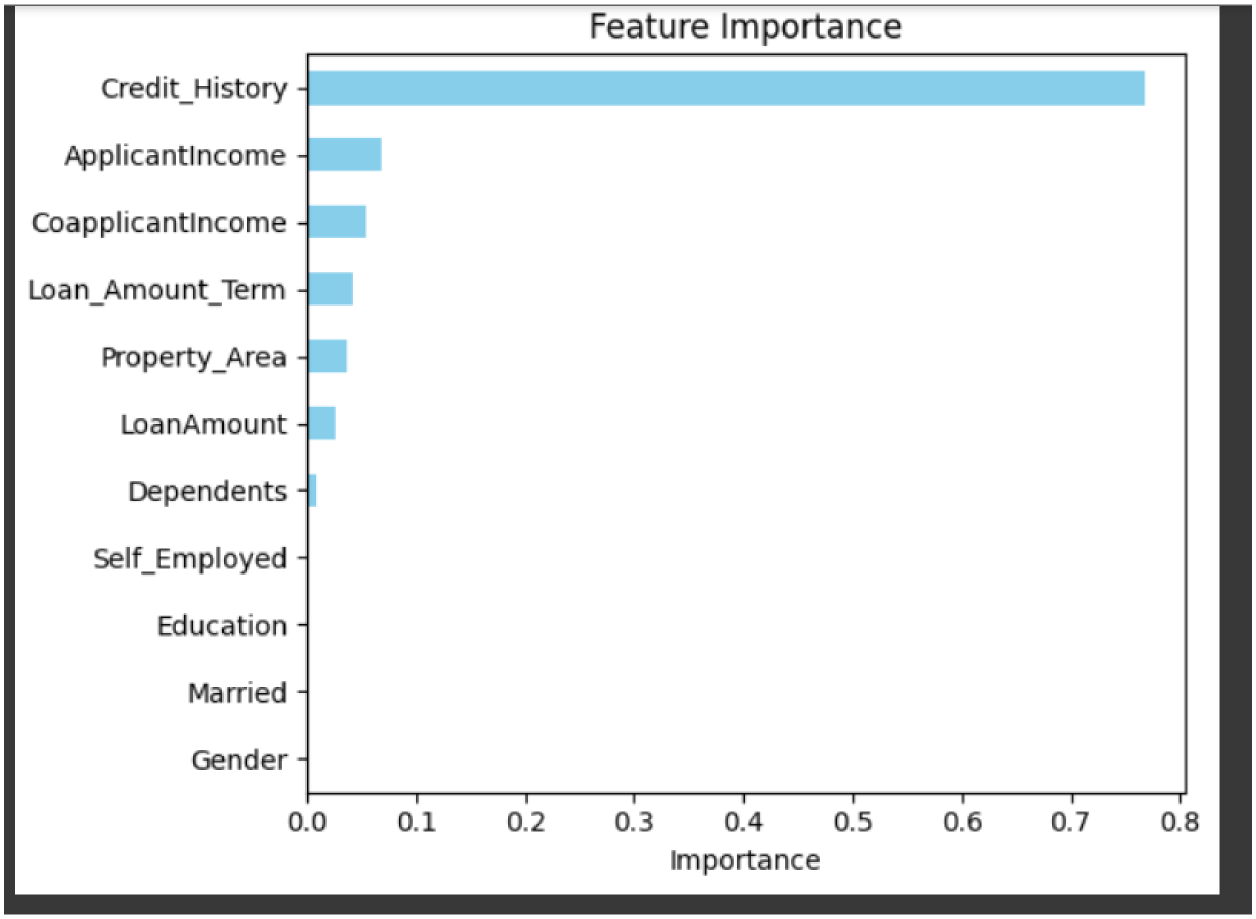
```python
# Prediction from user input
def predict_from_input():
    print("\n🔍 Enter details to predict loan approval (Yes=1, No=0 for binary fields):")
    input_data = {}

    input_data['Gender'] = int(input("Gender (Male=1, Female=0): "))
    input_data['Married'] = int(input("Married (Yes=1, No=0): "))
    input_data['Dependents'] = label_encoders['Dependents'].transform([input("Dependents (0, 1, 2, 3+): ")])[0]
    input_data['Education'] = int(input("Education (Graduate=0, Not Graduate=1): "))
    input_data['Self_Employed'] = int(input("Self Employed (Yes=1, No=0): "))
    input_data['ApplicantIncome'] = float(input("Applicant Income: "))
    input_data['CoapplicantIncome'] = float(input("Coapplicant Income: "))
    input_data['LoanAmount'] = float(input("Loan Amount (in thousands): "))
    input_data['Loan_Amount_Term'] = float(input("Loan Amount Term (in days): "))
    input_data['Credit_History'] = float(input("Credit History (1.0=Good, 0.0=Bad): "))
    input_data['Property_Area'] = label_encoders['Property_Area'].transform([input("Property Area (Urban, Rural, Semiurban): ")])[0]

    user_df = pd.DataFrame([input_data])
    prediction = model.predict(user_df)[0]
    print("\n🔮 Prediction Result:", "✅ Loan will be Approved" if prediction == 1 else "❌ Loan will NOT be Approved")

# Run the prediction function
predict_from_input()
```

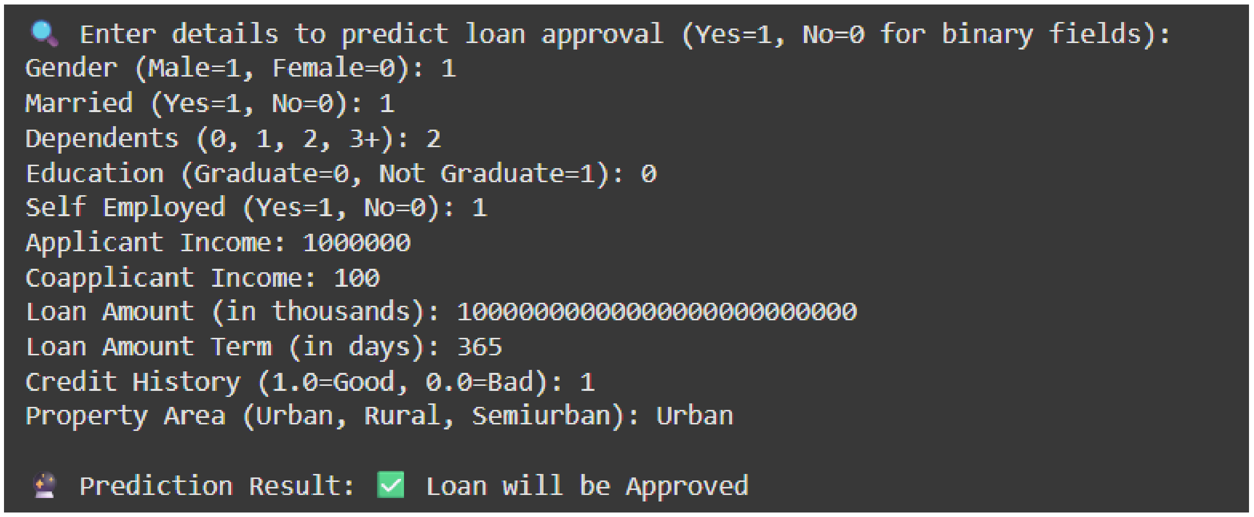


```
📊 Model Accuracy: 0.7723577235772358

📄 Classification Report:
              precision    recall  f1-score   support

           0       0.86      0.42      0.56        43
           1       0.75      0.96      0.85        80

    accuracy                           0.77       123
   macro avg       0.81      0.69      0.70       123
weighted avg       0.79      0.77      0.75       123
```

```
... 🧮 Confusion Matrix:
 [[18 25]
 [ 3 77]]
```







```
🔍 Enter details to predict loan approval (Yes=1, No=0 for binary fields):
Gender (Male=1, Female=0): 1
Married (Yes=1, No=0): 1
Dependents (0, 1, 2, 3+): 2
Education (Graduate=0, Not Graduate=1): 0
Self Employed (Yes=1, No=0): 1
Applicant Income: 1000000
Coapplicant Income: 100
Loan Amount (in thousands): 10000000000000000000000000
Loan Amount Term (in days): 365
Credit History (1.0=Good, 0.0=Bad): 1
Property Area (Urban, Rural, Semiurban): Urban

🔮 Prediction Result: ✅ Loan will be Approved
```