Assessment Report

on

“Predict Loan Default”

submitted as partial fulfillment for the award of

BACHELOR OF TECHNOLOGY DEGREE

SESSION 2024-25

in

CSE(AIML)

By

Name : Vikrant Baliyan

Roll Number : 202401100400211

Section: C

Under the supervision of

“ABHISHEK SHUKLA”

KIET Group of Institutions, Ghaziabad

May, 2025

1. Introduction

This report outlines a machine learning approach to predict loan default using financial history and credit scores. The goal is to classify whether a borrower will default on a loan based on features such as income, employment status, credit history, and other financial indicators. Accurate prediction of loan default can help financial institutions minimize risk and improve lending decisions.

2. Methodology

The methodology used in this classification problem consists of the following steps:

1. **Data Loading**: The dataset is loaded from a CSV file.
2. **Data Preprocessing**:
    - Removal of the non-informative identifier column (LoanID).
    - Encoding of categorical features using LabelEncoder.
3. **Train-Test Split**: The dataset is split into 80% training and 20% testing sets.
4. **Model Training**: A RandomForestClassifier is trained on the training data.
5. **Model Evaluation**:
    - A confusion matrix is used to visualize classification performance.
    - Accuracy, precision, recall, and F1-score are reported.

3. CODE

```
import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn.model_selection import train_test_split

from sklearn.preprocessing import LabelEncoder

from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import confusion_matrix, classification_report


# Load dataset

df = pd.read_csv("1. Predict Loan Default.csv")

if 'LoanID' in df.columns:

    df = df.drop(columns=['LoanID'])


# Encode categorical variables

categorical_cols = df.select_dtypes(include='object').columns

for col in categorical_cols:

    le = LabelEncoder()

    df[col] = le.fit_transform(df[col])


# Features and target

X = df.drop(columns=['Default'])

y = df['Default']

# Train-test split
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Model training
clf = RandomForestClassifier(random_state=42)
clf.fit(X_train, y_train)
y_pred = clf.predict(X_test)

# Evaluation
print(confusion_matrix(y_test, y_pred))
print(classification_report(y_test, y_pred))
```

---

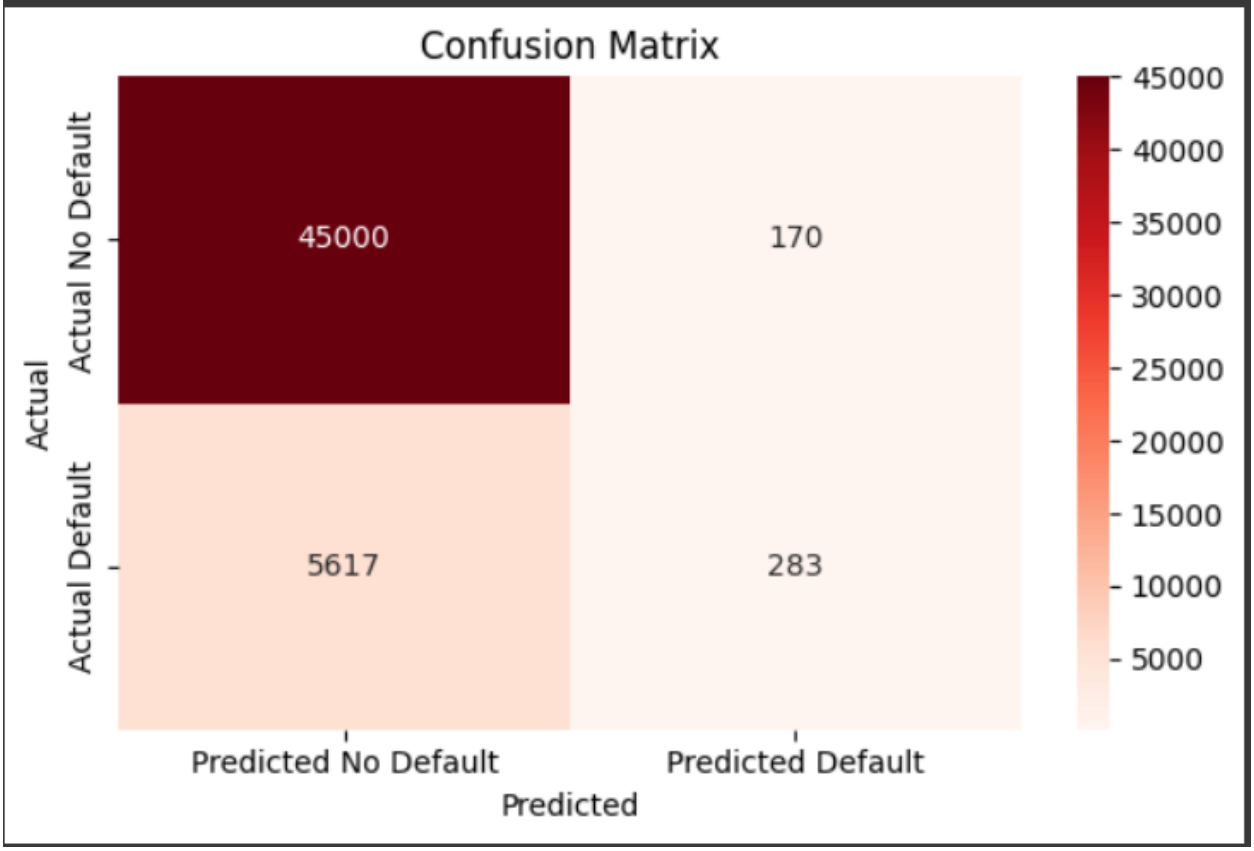
4. Output

Confusion Matrix:

	Predicted No Default	Predicted Default
Actual No Default	45,000	170
Actual Default	5,617	283

Classification Report:

	precision	recall	f1-score	support
0	0.89	1.00	0.94	45170
1	0.62	0.05	0.09	5900
accuracy			0.89	51070
macro avg	0.76	0.52	0.51	51070
weighted avg	0.86	0.89	0.84	51070



5. References

- Scikit-learn documentation
- Seaborn documentation
- Matplotlib documentation
- Dataset: Provided by user