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

Yash Varshney (202401100400217)

KIET Group of Institutions, Ghaziabad

April, 2025

Introduction

In this report, we aim to predict loan default using machine learning techniques. Loan default is a critical issue for financial institutions, as it can lead to significant financial losses. By predicting whether a borrower will default on their loan, institutions can take preventive actions to mitigate risk. The dataset used in this analysis contains various features of loan applicants, including demographic, financial, and historical information, and is used to build a model that predicts the likelihood of loan default.

The goal of this analysis is to develop a predictive model using a Random Forest classifier, evaluate its performance, and provide insights into the loan default prediction. We will evaluate the model based on accuracy, precision, and recall metrics, with a focus on how well the model detects defaults. The results from this analysis can be used to inform decisions in risk management.

Methodology

1. Data Preprocessing

The dataset used in this study is loaded and processed using the following steps:

1. **Data Cleaning**: The first step involves inspecting the data for any missing or irrelevant values. In this case, the Loan ID column is an identifier and is dropped from the dataset, as it does not contribute to the prediction model.
2. **Categorical Feature Encoding**: The dataset contains categorical features (e.g., gender, education level, etc.). These features are encoded using Label Encoding, which assigns a unique integer value to each category. This transformation ensures that the categorical variables can be used as inputs in the machine learning model.
3. **Sampling**: To speed up the training process, the dataset is randomly sampled, and only 10% of the data is used for model training and evaluation. This reduces computational time while maintaining the dataset's diversity.
4. **Feature and Target Separation**: The dataset is divided into the features (X) and the target variable (y). The target variable in this case is the Default column, indicating whether a loan was defaulted or not.

2. Model Selection and Training

The primary machine learning model used in this report is the **Random Forest classifier**. This model was selected due to its ability to handle both numerical and categorical data and its robustness in classification tasks.

1. **Training and Test Split**: The data is split into training and testing sets using an 80-20 split, where 80% of the data is used for training, and 20% is used for evaluation. This ensures the model is evaluated on unseen data.
2. **Model Training**: The Random Forest classifier is trained on the training set using 100 estimators. The model is trained to predict the probability of loan default based on the input features.

3. Model Evaluation

Once the model is trained, its performance is evaluated using the following metrics:

- **Accuracy**: The overall proportion of correct predictions made by the model.
- **Precision**: The proportion of predicted defaults that are actually defaults. This metric is important when the cost of false positives (predicting a loan will default when it does not) is high.

- **Recall**: The proportion of actual defaults that are correctly predicted by the model. This is crucial when the cost of false negatives (failing to predict a default when it occurs) is high.

Additionally, a confusion matrix is generated to visualize the performance of the model. This matrix shows the number of true positives, true negatives, false positives, and false negatives, providing insights into the model's predictive capabilities.

4. Data Visualization

The results of the confusion matrix are visualized using a heatmap, which makes it easy to interpret the model’s performance in terms of false positives and false negatives.

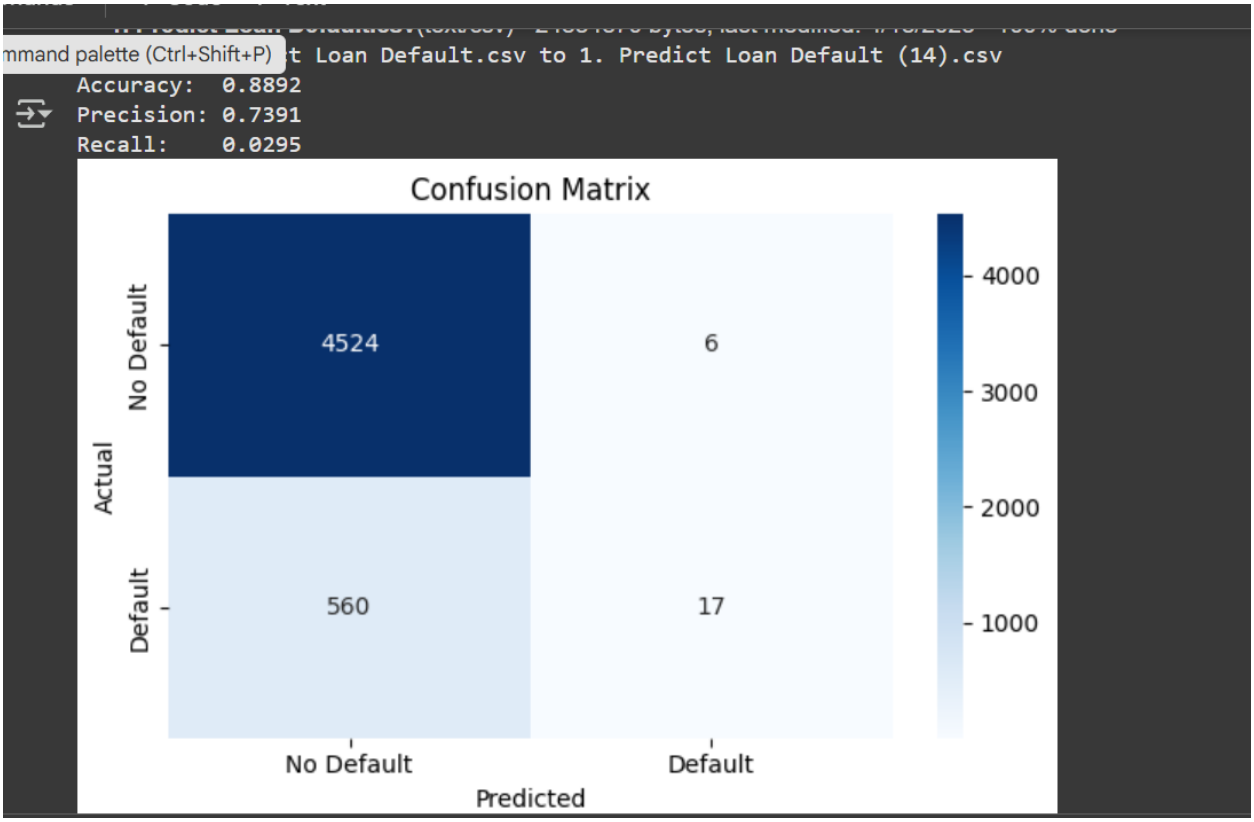
CODE

```
# Install necessary libraries
!pip install seaborn scikit-learn --quiet

# Import libraries
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, precision_score, recall_score, confusion_matrix

# Mount Google Drive (if your file is stored there)
from google.colab import files
uploaded = files.upload()

# Load dataset
df = pd.read_csv(next(iter(uploaded)))

# Drop identifier
df_model = df.drop(columns=["LoanID"])

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

# Sample 10% of the dataset to speed up training
df_sampled = df_model.sample(frac=0.1, random_state=42)
```

```
# Features and target
X = df_sampled.drop(columns=['Default'])
y = df_sampled['Default']

# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Train Random Forest classifier
clf = RandomForestClassifier(n_estimators=100, random_state=42)
clf.fit(X_train, y_train)

# Predictions
y_pred = clf.predict(X_test)

# Metrics
accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred)
recall = recall_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)

# Print evaluation metrics
print(f"Accuracy:  {accuracy:.4f}")
print(f"Precision: {precision:.4f}")
print(f"Recall:    {recall:.4f}")

# Plot heatmap
plt.figure(figsize=(6, 4))
sns.heatmap(conf_matrix, annot=True, fmt="d", cmap="Blues",
            xticklabels=["No Default", "Default"],
            yticklabels=["No Default", "Default"])
plt.title("Confusion Matrix")
plt.xlabel("Predicted")
plt.ylabel("Actual")
```

```
plt.tight_layout()

plt.show()
```

OUTPUT



References

- W3Schools. (n.d.). Python Data Analysis. Retrieved from https://www.w3schools.com/python/pandas