KIET
GROUP OF INSTITUTIONS
Connecting Life with Learning

An
Assessment Report
on
"Predict Loan Default"

submitted as partial fulfillment for the award of

BACHELOR OF TECHNOLOGY DEGREE

SESSION 2024-25

in

CSE (AI & ML)

By

Shashi Ranjan(202401100400171)

Under the supervision of

"Abhishek Shukla"

KIET Group of Institutions, Ghaziabad

Introduction

Loan default prediction is a crucial task in the financial sector. By predicting whether a borrower will default on a loan, financial institutions can reduce risk and make informed lending decisions. In this task, we use a dataset containing financial and demographic information of borrowers to build a classification model that predicts the likelihood of a loan default. Visualization and evaluation of results help validate the model performance.

Methodology

1. **Data Loading:** Loaded the dataset `1. Predict Loan Default.csv`.
2. **Preprocessing:**
    a. Dropped irrelevant columns (e.g., LoanID).
    b. Encoded categorical variables using Label Encoding.
3. **Sampling:** Sampled 20,000 rows from the dataset for faster model training.
4. **Splitting:** Divided the data into training and testing sets (80%-20%).
5. **Model Training:** Trained a Random Forest classifier with 10 estimators.
6. **Evaluation:** Calculated accuracy, precision, recall and plotted a confusion matrix heatmap.

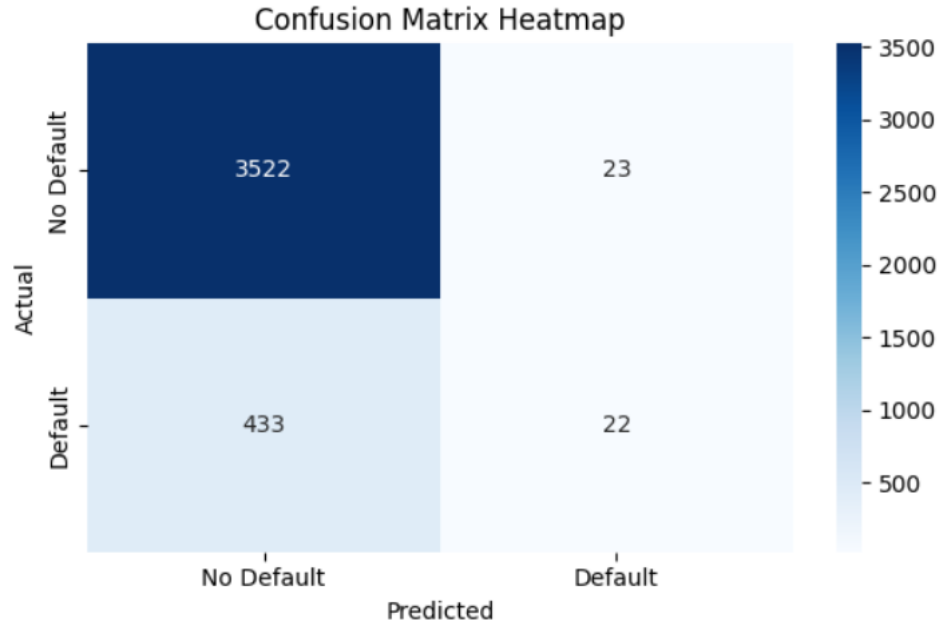
Code

```
import pandas aspd
importmatplotlib.pyplotasplt
importseabornassns
fromsklearn.model_selectionimporttrain_test_split
fromsklearn.preprocessingimportLabelEncoder
fromsklearn.ensembleimportRandomForestClassifier
fromsklearn.metricsimportconfusion_matrix, accuracy_score,
precision_score, recall_score

# Load the dataset
df = pd.read_csv("Predict Loan Default.csv")

# Drop the LoanID column
df = df.drop(columns=["LoanID"])

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

# Sample a subset for faster training
df_sampled = df.sample(n=20000, 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, stratify=y
)
# Train a Random Forest classifier
clf = RandomForestClassifier(n_estimators=10,
random_state=42)
clf.fit(X_train, y_train)

# Predictions
y_pred = clf.predict(X_test)

# Evaluation metrics
cm = confusion_matrix(y_test, y_pred)
accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred)
recall = recall_score(y_test, y_pred)

# Print metrics
print("Accuracy:", round(accuracy, 4))
print("Precision:", round(precision, 4))
print("Recall:", round(recall, 4))

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

OUTPUT

```
Accuracy: 0.886
Precision: 0.4889
Recall: 0.0484
```



References/Credits

- Dataset: Provided for AI MSE
- Libraries used: pandas, matplotlib, seaborn, scikit-learn
- Implementation: Done in Google Colab