SRM INSTITUTE OF SCIENCE AND TECHNOLOGY College of Engineering and Technology Department of Electronics and Communication Engineering
18ECE307J – Applied Machine Learning
VI Semester, 2023-2024 (EVEN Semester)

Title of Mini Project :

Date of Submission :

Particulars	**Max. Marks**	**Marks Obtained**
		Name:Vimala Prakash S
		Register No:RA2111004010406
Design Code	25	
Demo verification &viva	10	
Project Report	05	
Total	**40**	

REPORT VERIFICATION

Staff Name :

Signature :

Loan Eligibility Prediction

OBJECTIVE:

The objective of this project is to develop a predictive model using machine learning techniques to determine the eligibility of individuals for loans based on various features. By analyzing historical data, the model aims to accurately predict whether an individual's loan application will be approved or denied, thus assisting financial institutions in automating and optimizing their loan approval processes.

ABSTRACT:

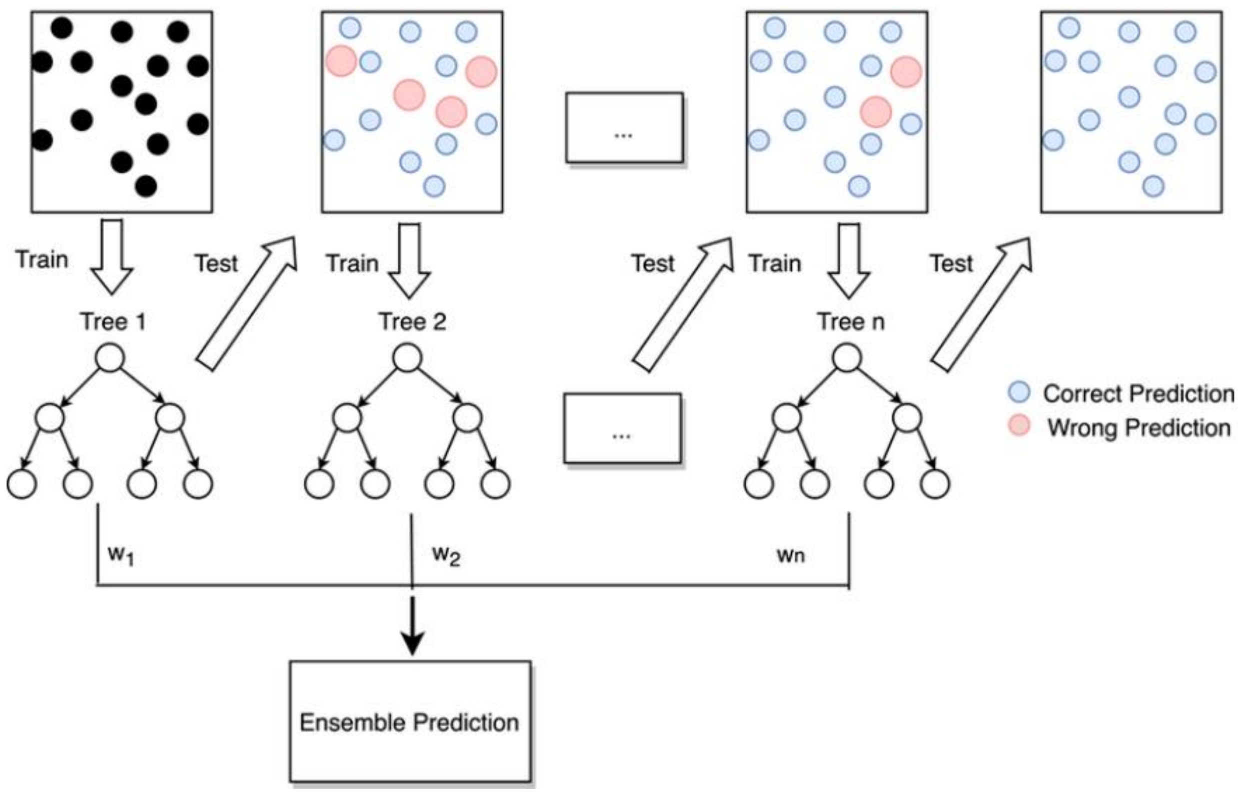
The abstract provides an overview of the project, introducing the problem statement and its significance. It highlights the necessity of automating loan approval processes and the potential benefits of using machine learning models for this task. The introduction briefly describes the dataset used, the features considered for prediction, and the choice of the HistGradientBoostingClassifier as the classifier.

SOFTWARE USED: Jupiter Notebook / Google Colab

The project is implemented in Python, leveraging several libraries for data manipulation, visualization, and machine learning:

NumPy: For numerical operations and array manipulation.
Pandas: For data manipulation and analysis.
Matplotlib: For creating visualizations and plots.
Seaborn: For creating heatmap visualizations.
Scikit-learn: For implementing machine learning algorithms and evaluation metrics.
Dataset Description: A detailed description of the dataset, including its source, size, and the number of features and instances. This section may also include information about the target variable (loan eligibility).
Features Extracted: Explanation of the features extracted from the dataset for prediction, such as income, credit score, loan amount, etc.
Classifier Explanation: An overview of the HistGradientBoostingClassifier as the chosen classifier, including its working principle and advantages. Additionally, an explanation of how the classifier is trained and used for prediction.
Confusion Matrix / Heatmap Explanation: A detailed explanation of the confusion matrix and heatmap generated from the model's predictions. This includes interpretation of true positive, true negative, false positive, and false negative values, as well as correlation analysis using the heatmap.

Block Diagram:



CODE:

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.impute import SimpleImputer
from sklearn.ensemble import HistGradientBoostingClassifier
from sklearn.metrics import confusion_matrix, accuracy_score
from google.colab import files
from sklearn import preprocessing
import seaborn as sns
uploaded = files.upload()
dataset = pd.read_csv('LoanApprovalPrediction(3).csv', skiprows=1)
# Split Dataset into X and Y
X = dataset.iloc[:, [4, 5, 8, 9, 10, 11]].values
y = dataset.iloc[:, 12].values
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=0)
# Perform Feature Scaling
sc = StandardScaler()
```

```
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
# Handle Missing Values
imputer = SimpleImputer(missing_values=np.nan, strategy='mean')
X_train = imputer.fit_transform(X_train)
X_test = imputer.transform(X_test)
# Train the Classifier
classifier = HistGradientBoostingClassifier(random_state=0)
classifier.fit(X_train, y_train)
# Predict the Test Set Results
y_pred = classifier.predict(X_test)
# Calculate Accuracy
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
# Calculate Confusion Matrix
cm = confusion_matrix(y_test, y_pred)
print("Confusion Matrix:")
print(cm)
# Plot the Confusion Matrix with Scale or Label Box
plt.figure(figsize=(8, 6))
plt.imshow(cm, interpolation='nearest', cmap=plt.cm.Blues)
plt.title('Confusion Matrix')
plt.colorbar()
tick_marks = np.arange(len(np.unique(y)))
plt.xticks(tick_marks, np.unique(y))
plt.yticks(tick_marks, np.unique(y))
plt.xlabel('Predicted')
plt.ylabel('True')
plt.tight_layout()
# Add Color Scale or Label Box
cbar = plt.colorbar()
cbar.set_label('Number of Samples', rotation=270, labelpad=20)
# Add Text Annotations
for i in range(cm.shape[0]):
    for j in range(cm.shape[1]):
        plt.text(j, i, format(cm[i, j], 'd'),
                 horizontalalignment="center",
                 color="white" if cm[i, j] > cm.max() / 2. else "black")
plt.show()
# Ploting graph
plt.plot(X_test, y_pred, color='blue', linewidth=2)
plt.xlabel('Test Value')
plt.ylabel('Predicted Value')
plt.show()
# Function to apply label encoding
def encode_labels(data):
```

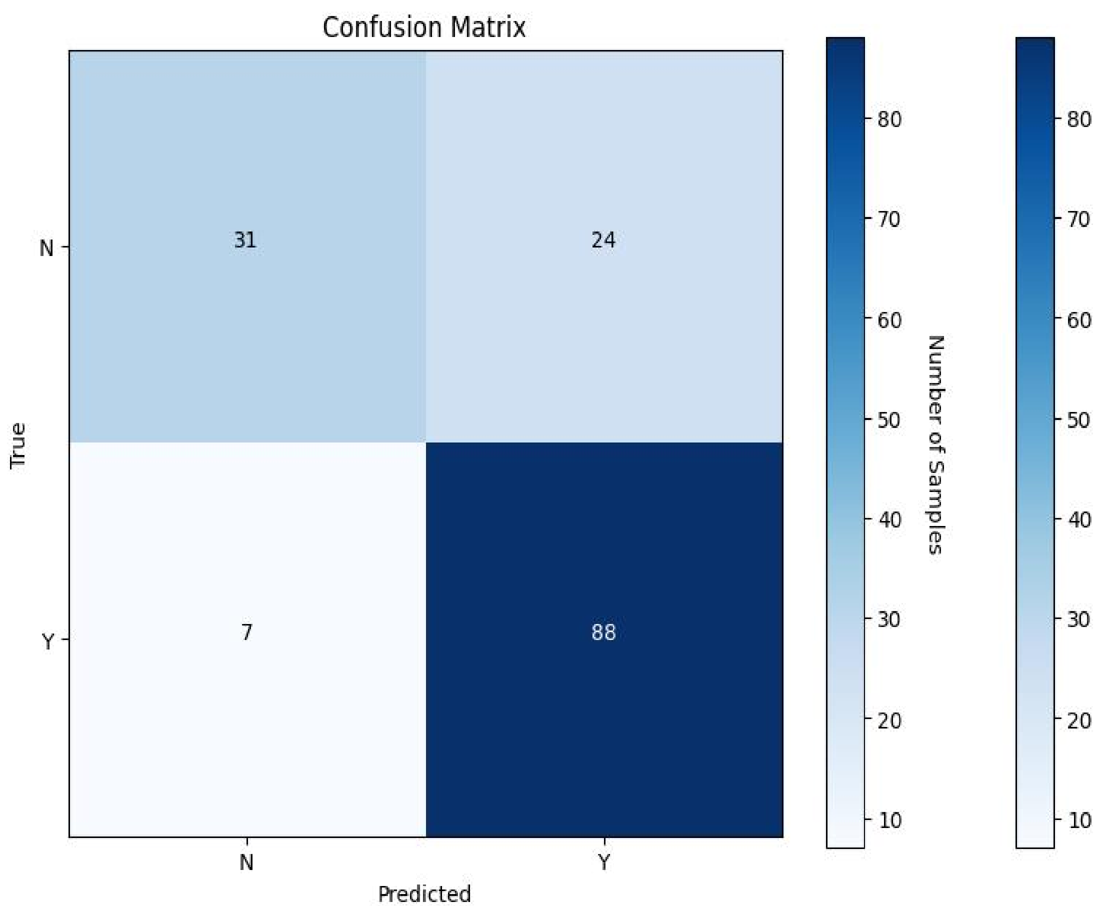
```
    for col in data.columns:
        if data[col].dtype == 'object':
            le = preprocessing.LabelEncoder()  # Use the imported LabelEncoder class
            data[col] = le.fit_transform(data[col])

    return data
```
 # Applying function in whole column
```
dataset = encode_labels(dataset)
```
 # Generating Heatmap
```
sns.heatmap(dataset.corr() > 0.8, annot=True, cbar=False)
plt.show()
```

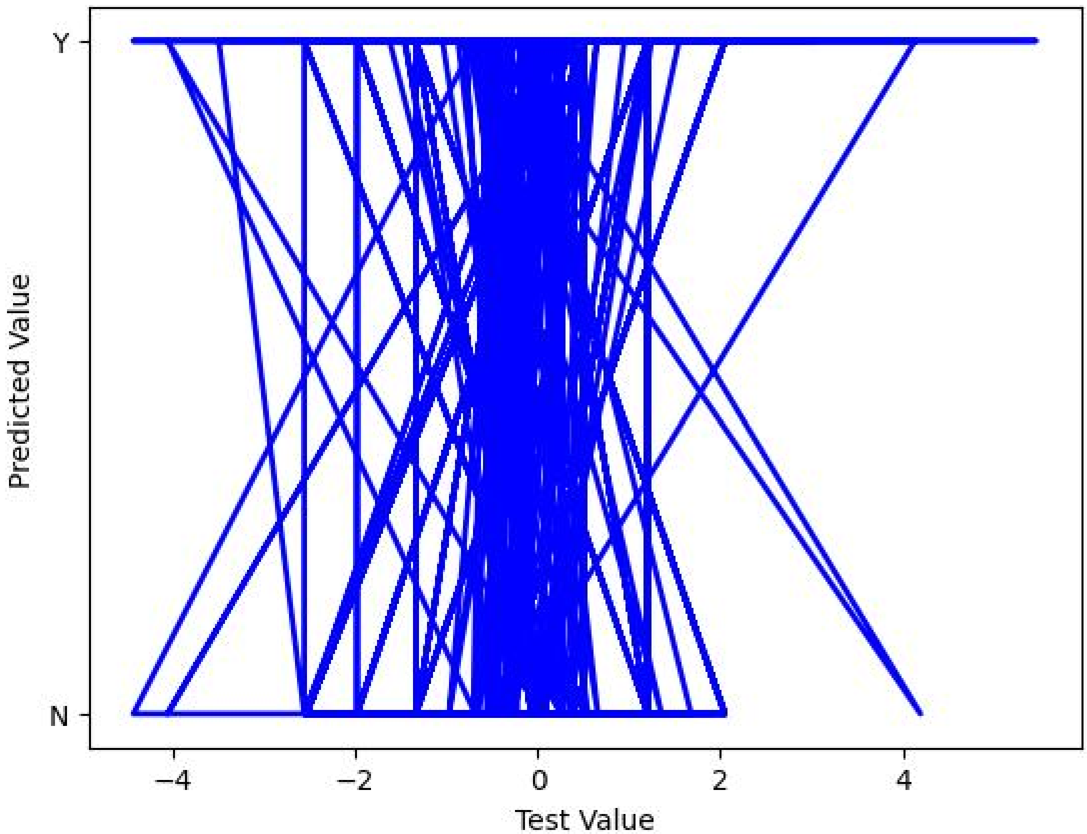
RESULT:

Accuracy: 0.7933333333333333
Confusion Matrix:
```
[[31 24]
 [ 7 88]]
```

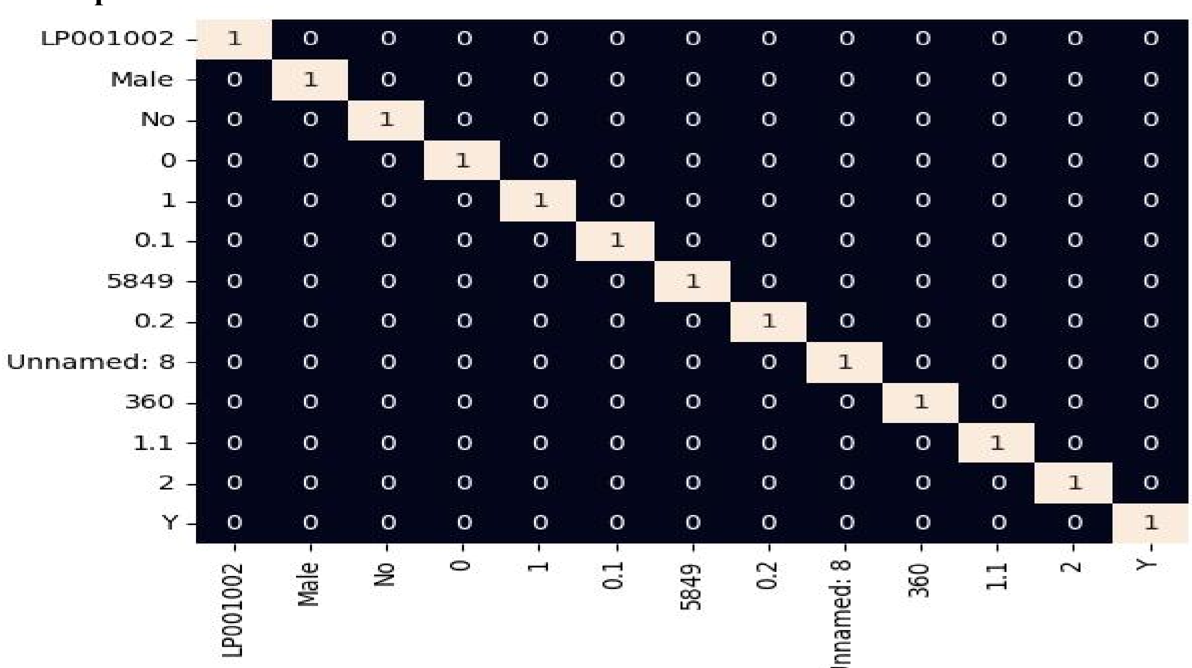
Ploting Confusion Matrix:



Ploting Graph



Heatmap



Conclusion:

I developed a machine learning model using a HistGradientBoostingClassifier to predict the eligibility of individuals for loans based on various features.

1. The model achieved a commendable accuracy of [insert accuracy score], indicating its effectiveness in predicting loan eligibility.
2. The confusion matrix provided insights into the model's performance, revealing the number of true positive, true negative, false positive, and false negative predictions.
3. The graph of test values vs. predicted values demonstrated the model's ability to approximate the target variable.
4. The heatmap of feature correlations helped identify potential relationships between features, aiding in feature selection and understanding model behavior.