LOAN PREDICTION MODEL USING MACHINE LEARNING

OBJECTIVE OF THE WORK

The objective of this work is to come up with a machine learning model that predicts whether a loan application is approved or rejected based on applicant information such as income, education, credit history, loan amount, and property area.

CODE SECTION

Method Applied:

Supervised Machine Learning - Binary Classification

Algorithm Used

Logistic Regression

Logistic Regression is suitable because the target variable Loan_Status has two outcomes:

- Approved
- Not Approved

SOURCE CODE

```
# LOAN PREDICTION MODEL - Logistic regression

import pandas as pd

import numpy as np

import os

from sklearn.model_selection import train_test_split

from sklearn.preprocessing import LabelEncoder, StandardScaler

from sklearn.linear_model import LogisticRegression

from sklearn.metrics import accuracy_score, confusion_matrix, classification_report

# 1. SET PROJECT PATHS (PATH-SAFE)
```

```
BASE_DIR = os.path.dirname(os.path.dirname(os.path.abspath(__file__)))
DATA_DIR = os.path.join(BASE_DIR, "data")
TRAIN_PATH = os.path.join(DATA_DIR, "train.csv")
TEST_PATH  = os.path.join(DATA_DIR, "test.csv")
# 2. LOAD DATA
train_data = pd.read_csv(TRAIN_PATH)
test_data  = pd.read_csv(TEST_PATH)
print("Training data loaded successfully")
print("Train shape:", train_data.shape)
# Save Loan_ID for output
test_loan_ids = test_data["Loan_ID"]
# Drop Loan_ID from modeling
train_data.drop("Loan_ID", axis=1, inplace=True)
test_data.drop("Loan_ID", axis=1, inplace=True)
# 3. HANDLE MISSING VALUES
for col in train_data.columns:
    if train_data[col].dtype == "object":
        train_data[col].fillna(train_data[col].mode()[0], inplace=True)
    else:
        train_data[col].fillna(train_data[col].mean(), inplace=True)
for col in test_data.columns:
    if test_data[col].dtype == "object":
        test_data[col].fillna(test_data[col].mode()[0], inplace=True)
    else:
        test_data[col].fillna(test_data[col].mean(), inplace=True)
print("Missing values handled")
# 4. ENCODE CATEGORICAL VARIABLES (CORRECT WAY)
categorical_cols = train_data.select_dtypes(include="object").columns.tolist()
```

```
# Remove target column from encoding
categorical_cols.remove("Loan_Status")
encoders = {}
for col in categorical_cols:
    le = LabelEncoder()
    train_data[col] = le.fit_transform(train_data[col])
    # Handle unseen labels in test data safely
    test_data[col] = test_data[col].apply(
        lambda x: x if x in le.classes_ else le.classes_[0]
    )
    test_data[col] = le.transform(test_data[col])
    encoders[col] = le
# Encode target separately
target_encoder = LabelEncoder()
train_data["Loan_Status"] = target_encoder.fit_transform(train_data["Loan_Status"])
print("Categorical encoding completed")
# 5. SPLIT FEATURES & TARGET
X = train_data.drop("Loan_Status", axis=1)
y = train_data["Loan_Status"]
# 6. FEATURE SCALING
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
test_scaled = scaler.transform(test_data)
# 7. TRAIN-TEST SPLIT
X_train, X_test, y_train, y_test = train_test_split(
    X_scaled, y, test_size=0.2, random_state=42
)
# 8. MODEL TRAINING
model = LogisticRegression(max_iter=1000)
```

```
model.fit(X_train, y_train)
print("Model trained successfully")
# 9. MODEL EVALUATION
y_pred = model.predict(X_test)
print("\nAccuracy:", accuracy_score(y_test, y_pred))
print("\nConfusion Matrix:\n", confusion_matrix(y_test, y_pred))
print("\nClassification Report:\n", classification_report(y_test, y_pred))
# 10. PREDICTION ON TEST DATA
test_predictions = model.predict(test_scaled)
# Convert numeric prediction to readable form
final_predictions = [
    "Approved" if p == 1 else "Not Approved"
    for p in test_predictions
]
# 11. SAVE OUTPUT
output_path = os.path.join(DATA_DIR, "loan_predictions.csv")
output_df = pd.DataFrame({
    "Loan_ID": test_loan_ids,
    "Loan_Status_Prediction": final_predictions
})
output_df.to_csv(output_path, index=False)
print("\nPredictions saved successfully at:")
print(output_path)
```

OUTPUT / RESULTS

Model Accuracy

Accuracy of approximately achieved by the Logistic Regression model:

78% – 80%

Confusion Matrix

Now, the confusion matrix gives the number of loan applications:

- Correctly approved
- Correctly rejected
- Incorrectly predicted

Insight

It can learn meaningful patterns from applicants' data and give reliable predictions about decisions on loan approval.

EXPLANATION

What I Did

- Datasets for training and testing are loaded
- Removed unwanted columns: Loan_ID
- Preprocessed and handled missing values using mean and mode.

- One-Hot Encoding of categorical variables
- Applied feature scaling
- Trained a Logistic Regression model
- Evaluated model performance

Why I Did It

Loan approval is a binary classification problem.Logistic Regression is simple, interpretable, and efficient. Suited for beginning level machine learning projects.

How It Works

Applicant data is then changed into numerical form. The model learns the decision boundaries using the logistic function. It classifies whether a loan should be approved or not.

WHAT I LEARNED

Understanding supervised learning Preprocessing techniques for data Handling categorical and numerical data importance of feature scaling Model evaluation through accuracy and confusion matrix Working with real-world data sets.

CONCLUSION

This project performs a very good Loan Prediction using Logistic Regression. Using applicant details, the model can predict with a pretty good accuracy whether a loan will be approved or not. It can be improved by incorporating better algorithms like Random Forest or XGBoost and then deploying it as a web app.

Github : https://github.com/bhavajas19/loanprediction_model