Loan Approval Prediction Report

Project Title:

Building a Prediction Model for Banking Loan Approval

Problem Statement:

Banks often struggle to approve the right loan applicants despite rigorous screening. The goal of this project is to develop a machine learning model to accurately predict whether a loan should be **approved (1)** or **rejected (0)**.

Dataset Description:

The dataset contains various applicant-related attributes. Key variables include:

Feature	Description
APP_ID	Application ID
CIBIL_SCORE_VALUE	Credit Score: 0 = Bad, 1 = OK, 2 = Good
NEW_CUST	New to Credit: Yes/No
EMPLOYMENT_TYPE	1 = Salaried, 0 = Self-employed
AGE	Age of applicant
SEX	Gender: M/F
NO_OF_DEPENDENTS	Number of dependents
MARITAL_Status	Marital status: 1 = Married, 0 = Not Married
EDU_QUA	Education: 1 = Educated, 0 = Not educated
P_RESTYPE	Residence type: 1 = Own, 0 = Rented
EMPLOYEE_TYPE	0 = Private, 1 = Temp, 2 = Govt
MON_IN_OCC	Months in current occupation
INCOME_EXP_GMI	Income-to-expense rating
ASSET_LOAN_RATIO	Asset to loan value ratio
TENURE	Loan tenure in months (12 to 48)
STATUS	Target variable: 1 = Approved, 0 = Rejected

Data Preprocessing:

- Encoded categorical variables using Label Encoding.
- Filled missing values.
- Split data into 70% training and 30% testing sets.

Model: Random Forest Classifier

Evaluation Results:

Accuracy: 0.6473684210526316

Classification Report:

```
Accuracy: 0.6473684210526316

Classification Report:
              precision    recall  f1-score   support

           0       0.68      0.82      0.74      2474
           1       0.56      0.36      0.44      1516

    accuracy                           0.65      3990
   macro avg       0.62      0.59      0.59      3990
weighted avg       0.63      0.65      0.63      3990
```

Confusion Matrix:

	Predicted Rejected	Predicted Approved
Actual Rejected	2039	435
Actual Approved	972	544

The model performs better at identifying rejections than approvals. Class imbalance may be a contributing factor.

Google Colab Code:

Loan Approval Prediction - Random Forest Classifier

Step 1: Import Libraries

import pandas as pd

```python
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 accuracy_score, classification_report, confusion_matrix

# Step 2: Load Dataset
from google.colab import files
uploaded = files.upload()
df = pd.read_csv('your_dataset.csv')

# Step 3: Preprocessing
label_encoders = {}
for column in df.columns:
    if df[column].dtype == object:
        le = LabelEncoder()
        df[column] = le.fit_transform(df[column].astype(str))
        label_encoders[column] = le

X = df.drop(columns=['APP_ID', 'STATUS'])
y = df['STATUS']

# Step 4: Split the Data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

# Step 5: Train the Model
model = RandomForestClassifier(n_estimators=100, random_state=42)
model.fit(X_train, y_train)
```

Step 6: Predictions

y_pred = model.predict(X_test)

Step 7: Evaluation

print("Accuracy:", accuracy_score(y_test, y_pred))

print(classification_report(y_test, y_pred))

Step 8: Confusion Matrix

cm = confusion_matrix(y_test, y_pred)

plt.figure(figsize=(6,4))

sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', xticklabels=["Rejected", "Approved"], yticklabels=["Rejected", "Approved"])

plt.title('Confusion Matrix')

plt.xlabel('Predicted')

plt.ylabel('Actual')

plt.show()

Conclusion:

We built a Random Forest model with 64.7% accuracy for predicting loan approvals. Future improvements include handling class imbalance and model tuning for better recall of approved loans.

Confusion Matrix Plot (Output):

```
Confusion Matrix:
[[2039  435]
 [ 972  544]]
```