Loan Default Prediction — End-to-End Machine Learning Project

Complete Data Science Workflow, Analysis & Model Deployment

---

1. Project Overview

This project develops a complete end-to-end machine learning system to predict loan default risk. The workflow includes data loading, cleaning, feature engineering, exploratory data analysis (EDA), preprocessing, model training, evaluation, and deployment of a ready-to-use prediction pipeline.

---

2. Dataset Summary

Rows: 255,347
Columns: 18
Memory: 126 MB before optimization
Missing Values: None
Duplicates: None
Dropped Column: LoanID (identifier)
Target Imbalance: ~11.6% default

Key Numeric Statistics:

- **Age:** Mean 43 (Range: 18–69)
- **Income:** Mean 82,499 (Range: 15k–149k)
- **LoanAmount:** Mean 127,578
- **CreditScore:** Mean 574 (Range: 300–849)

---

3. Data Cleaning

- Verified absence of missing and duplicate data
- Converted categorical columns to category dtype
- Ensured numeric columns were properly typed
- No significant outliers detected
- Memory usage reduced from ~126MB to ~25MB

---

4. Feature Engineering

New features were created to enhance model predictive performance:

- **Loan_to_Income = LoanAmount / Income**
- **Employment_Stability = MonthsEmployed / 12**
- **CreditLines_per_Year = NumCreditLines / (EmploymentYears + 0.1)**

- **High_Risk_Loan = 1 if DTI > 0.6 or CreditScore < 500**

Feature Groups:

- **Numeric:** Age, Income, LoanAmount, CreditScore, etc.
- **Categorical:** Education, EmploymentType, LoanPurpose, etc.

---

5. Exploratory Data Analysis (EDA)

Key Insights:

- Target imbalance: Only 11.6% are defaulters
- Defaulters typically have:
  - Lower CreditScore
  - Higher LoanAmount
  - Lower Employment Stability
  - Higher-risk loan purposes (Auto, Business)

Correlation Highlights:

- Age ↓ → Default ↑
- CreditScore ↓ → Default ↑
- High_Risk_Loan strongly correlated with default
- LoanAmount and Income show strong positive correlation

---

6. Data Preprocessing

- **Train-Test Split:** 70/30 with stratification
- **Numeric Pipeline:** Median imputation + Standard Scaling
- **Categorical Pipeline:** One-Hot Encoding with handle_unknown='ignore'
- Combined using Column Transformer
- Fitted only on training data to prevent data leakage

---

7. Model Training and Evaluation

Models Trained:

- Logistic Regression
- Random Forest
- XGBoost
- Naive Bayes
- Decision Tree

Evaluation Metrics:
Accuracy, Precision, Recall, F1-score, ROC-AUC, Sensitivity, Specificity

---

8. Model Performance Summary

Model	Accuracy	Precision	Recall	F1	ROC-AUC
Logistic Regression	0.8870	0.6200	0.0691	0.1244	0.7626
Random Forest	0.8859	0.6292	0.0422	0.0790	0.7350
XGBoost	0.8848	0.5235	0.0852	0.1466	0.7422
Naive Bayes	0.8611	0.3437	0.2153	0.2647	0.7398
Decision Tree	0.8044	0.2039	0.2356	0.2186	0.5574

Key Takeaways:

- **Best Overall Model:** Logistic Regression
- **Best Recall:** Naive Bayes (detects more defaulters)
- **Final Model for Deployment:** Logistic Regression

---

9. Deployment Pipeline

Saved Model File: best_model_Logistic_Regression.joblib

Pipeline Components:

- Feature engineering
- Preprocessing (encoding and scaling)
- Trained Logistic Regression model

Prediction Script: pipeline_test.py

- Outputs predicted class and probability

---

10. Project Deliverables

- Full Jupyter notebook (EDA + ML workflow)
- Cleaned dataset
- Engineered dataset
- Modular feature_engineer.py script
- Final ML pipeline
- Test script for inference
- Professional README and documentation

---

11. Author Information

Name: Osama Othman
Email: osmanosamaahmed@gmail.com
LinkedIn: www.linkedin.com