**Predicting Loan Default Risk: A Data Science Case Study**

Introduction

Credit risk assessment is one of the most critical challenges in the financial industry. Institutions must accurately identify which borrowers are likely to default, in order to minimize losses and improve financial inclusion.

In this project, I set out to build a machine learning model that predicts whether a loan applicant will default based on their demographic details, loan history, and performance records.

The project followed a structured workflow, guided by a set of instructions from my tutor and project guidelines. The pipeline included:  
 1. Data collection and merging.  
 2. Data cleaning and preprocessing.  
 3. Exploratory Data Analysis (EDA).  
 4. Baseline modeling.  
 5. Feature engineering.  
 6. Feature selection (Variance Threshold, RFE, SHAP).  
 7. Robust modeling with multiple algorithms.  
 8. Visualization in Power BI.  
 9. Deployment preparation.

Along the way, I encountered challenges, made corrections (especially with dataset merging), and refined my approach. This article documents the full process.

**Step 1: Data Sources**

Three datasets were provided:  
 1. Demographics dataset: customer details (e.g., birthdate, bank account type, employment status, education).  
 2. Loan performance dataset: loan number, loan amount, total due, term days, good/bad flag.  
 3. Previous loans dataset: approval date, creation date, closed date, first due date, first repaid date.

Correction I made later: Initially, I merged all three datasets directly, but later realized datasets 2 and 3 shared duplicate columns. The correct approach was:  
 • Merge dataset 1 and 2 on customer\_id.  
 • Then, only merge the unique columns from dataset 3 (closeddate, firstduedate, firstrepaiddate).

**Step 2: Data Cleaning**

Each dataset was cleaned separately before merging. Key steps included:  
 • Standardizing column names (lowercase, underscores).  
 • Handling missing values:  
 • Categorical → filled with “Unknown”.  
 • Numerical → filled with median values.  
 • Dropping duplicates.  
 • Converting date columns to date-time format.

Example:

df1[“birthdate”] = pd.to\_datetime(df1[“birthdate”], errors=”coerce”)

**Step 3: Exploratory Data Analysis (EDA)**

EDA was done to understand data patterns and distributions:  
 • Target distribution: “Good” vs “Bad” loans. Imbalanced (majority were “Good”).  
 • Numerical distributions: Loan amounts, total due, term days.  
 • Categorical analysis: Employment status, bank type, education level, etc.  
 • Group comparisons: Default rates across different demographics.  
 • Correlation heatmaps: Identified weak correlations but useful signals from repayment patterns.

Key finding: Certain banks and employment statuses had higher default rates.

**Step 4: Baseline Modeling**

Before feature engineering, I tested three models on the raw dataset:  
 • Logistic Regression  
 • Random Forest  
 • XGBoost

Results (ROC AUC):  
 • XGBoost→ 0.6031 (best baseline)  
 • Random Forest→ 0.5879  
 • Logistic Regression → 0.5847 (performed poorly, likely due to imbalance and weak features).

Interpretation: Even without feature engineering, tree-based models outperformed linear models.

**Step 5: Feature Engineering**

Based on guidelines, I created new features:  
 1. Loan-to-Income Ratio → approximated from available loan amount fields.  
 2. Risk Bands → categorized loan amounts into Low, Medium, High, Very High.  
 3. Payment Behavior Score → calculated repayment delays (firstduedate vs firstrepaiddate).  
 • Negative delay = early repayment → score 4.  
 • On-time = score 4.  
 • Short delay = score 3.  
 • Long delay = score 2 or 1.

These features improved interpretability and predictive power.

**Step 6: Feature Selection**

With thousands of features after one-hot encoding, feature selection was necessary.  
 1. Variance Threshold: removed near-constant features.  
 • Dropped one irrelevant feature (risk\_band\_Very High).  
 2. Recursive Feature Elimination (RFE): kept top 20 features.  
 • Selected features like payment\_behavior\_score, Bank\_Wema Bank, Employment\_Retired.  
 3. SHAP Values: interpreted feature importance globally.  
 • Showed repayment behavior and certain bank affiliations were the strongest predictors.

**Step 7: Robust Modeling**

Following the guidelines, I tested five models with hyperparameter tuning (GridSearchCV):  
 1. Logistic Regression  
 2. Random Forest  
 3. XGBoost  
 4. LightGBM  
 5. Neural Networks (MLP)

Final Results (best performing model from robust phase):  
 • MLP Classifier (Neural Network):  
 • Accuracy: 0.57  
 • Precision: 0.27  
 • Recall: 0.56  
 • F1: 0.36  
 • ROC AUC: 0.61

Interpretation: Performance dropped compared to baseline because of class imbalance and feature sparsity. The dataset may need better balancing (SMOTE/undersampling) or richer income/credit data for better results.

**Step 8: Visualization with Power BI**

To complement the machine learning models, I built a Power BI dashboard to make the results interpretable for business stakeholders. The dashboard focused on understanding who the defaulters are, their demographic distribution, and trends over time.

Key Visualizations Included:  
 1. Defaulters by Age Group : Showed how default rates vary across different age brackets, highlighting high-risk segments.  
 2. Defaulters by Bank Account Type : Compared defaults among customers with savings vs current accounts.  
 3. Defaulters by Employment Status : Identified which employment categories (e.g., employed, self-employed, retired) are most associated with defaults.  
 4. Default Trend Over Time : Displayed how defaults evolved month by month, useful for spotting seasonal or systemic risk.  
 5. Loan Amount vs Age Group : Compared loan sizes taken across age categories, uncovering which groups tend to borrow more.  
 6. Loan Disbursal Overview : Showed the volume of loans disbursed over time, helping contextualize default levels.  
 7. Default Rates by Bank Name : Highlighted which banks had higher or lower risk profiles based on client performance.

Insights from Dashboard:  
 • People within the age group (36–45) tended to default more frequently.  
 • Certain banks had disproportionately higher default rates.  
 • Employment status was a key driver of repayment ability.  
 • Defaults fluctuated over time, suggesting possible external (economic/seasonal) factors.

With these interactive visuals, stakeholders could filter by demographics (age, employment, bank type) and quickly identify risk hotspots.

**Step 9: Deployment (Optional)**

The model was prepared for deployment using Streamlit.  
Steps included:  
 1. Saving the trained model as a .pkl file using joblib.  
 2. Writing a streamlit\_app.py script to load the model and make predictions.  
 3. Running the app locally via streamlit run streamlit\_app.py.

Due to environment constraints, deployment wasn’t fully finalized but the framework was established.

**Key Learnings & Shortcomings**  
 • Data merging mistake: Initially merged full datasets instead of only unique columns → fixed later.  
 • Imbalanced data: Caused poor performance in some models (Logistic Regression, Neural Network).  
 • Feature engineering was crucial: repayment behavior stood out as a strong predictor.  
 • Deployment challenges: Faced environment and package issues but outlined a deployment pipeline.

**Conclusion**

This project demonstrated the end-to-end process of solving a loan default risk problem:  
 • Data cleaning, merging, and preprocessing.  
 • Insightful EDA and feature engineering.  
 • Baseline vs robust modeling comparisons.  
 • Model interpretation with SHAP.  
 • Business-facing dashboard in Power BI.

While the models achieved moderate performance, the project highlighted areas for improvement: richer features, better handling of imbalance, and improved deployment readiness.