Capstone Project

Loan Approval Prediction Capstone Project

Machine Learning and FastAPI

Github Link : https://github.com/priyank-rupera-13/Loan-Approval-Prediction-Capstone

Dataset : Loan-Approval-Prediction-Dataset

Created By : Priyank Rupera

Objective

Build a reliable system that predicts whether a loan application should be approved or rejected. The model uses key financial indicators (CIBIL score, annual income, loan amount, tenure in years, asset values, employment details).

To keep decisions realistic, enforce an affordability rule: the EMI at the chosen interest rate must be ≤ 50% of the applicant's monthly income.

- Binary target: 1=Approve, 0=Reject (strict mapping).
- Tenure stored in years In dataset; converted to months for EMI.
- Final decision : Machine Learning prediction gated by EMI policy.

Approach to Solving the Problem

Followed a clear, reproducible ML workflow.
After cleaning column names and mapping the target to {0,1}, split the data with stratification to preserve class balance.

A shared ColumnTransformer imputes numeric values (median) and encodes categorical (one-hot, handle_unknown='ignore'), ensuring identical preprocessing in training and inference.

- Baseline: RandomForest pipeline.
- Comparison: XGBoost pipeline with the same preprocessing.
- Evaluation: Accuracy, Precision, Recall, F1, ROC AUC; Confusion Matrix; ROC curves.
- Artifacts: joblib pipelines and feature_columns.json for API alignment.

Model Summary

Both models are wrapped in pipelines so that feature engineering and encoding are baked into the artifact.

RandomForest offers robustness and straightforward importance scores.
XGBoost is a powerful gradient boosting method that often excels on tabular data and can capture complex interactions.

- Imbalance: consider class_weight (RF) / scale_pos_weight (XGB) if classes are skewed.
- Interpretability: feature_importances_ and permutation importance; optional SHAP & PDPs.
- Reproducibility: fixed random_state and consistent preprocessing.

Results — Test Metrics

The following metrics summarize performance on the hold-out set. Compared RandomForest and XGBoost using the same test data to ensure a fair, unbiased evaluation.

Accuracy	0.9824
Precision	0.985
Recall	0.9868
F1-score	0.9859
ROC AUC	0.9984

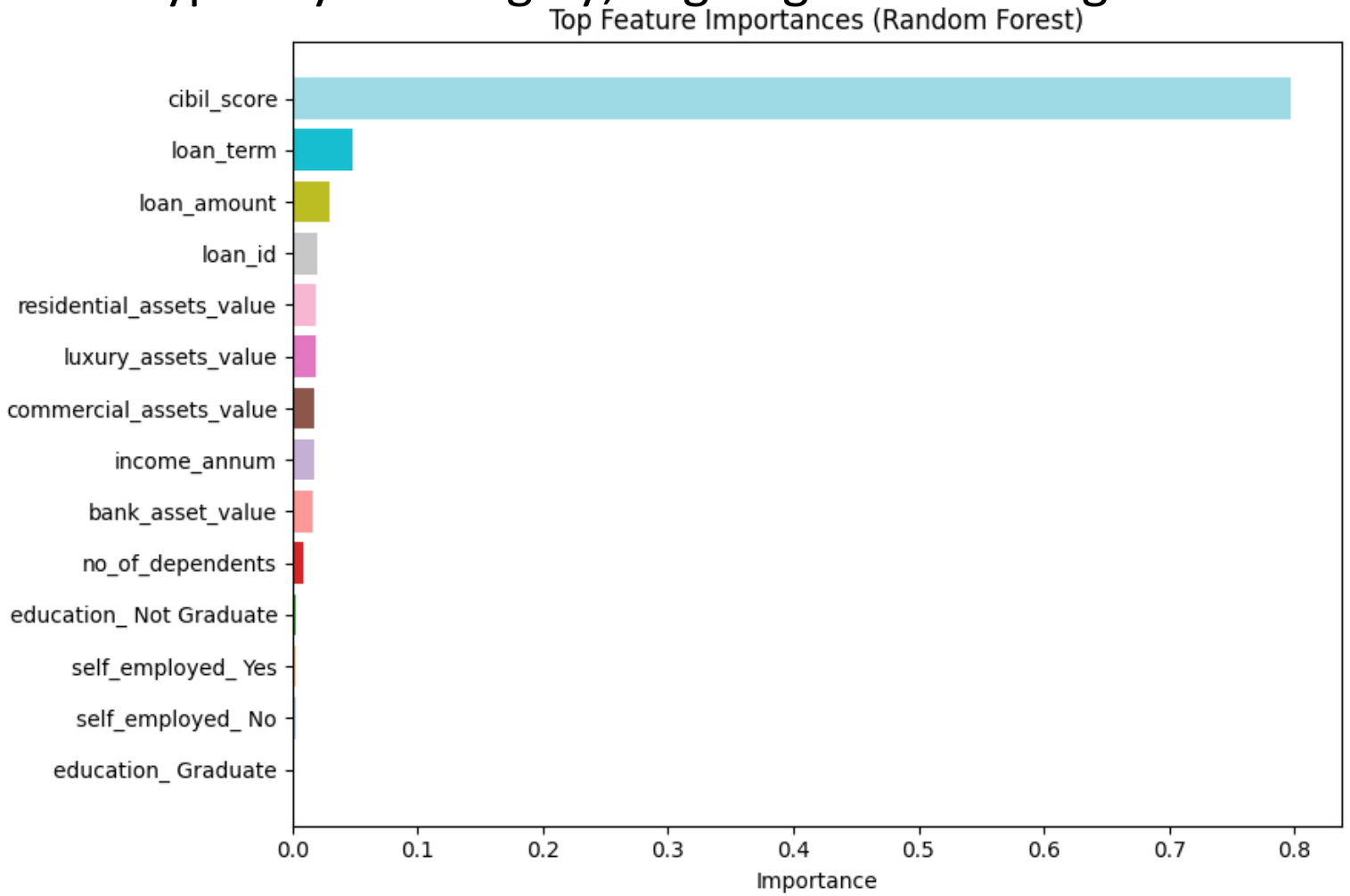
Classification Report

	Precision	Recall	F1-Score	Support
Reject	0.98	0.98	0.98	323
Approve	0.98	0.99	0.99	531

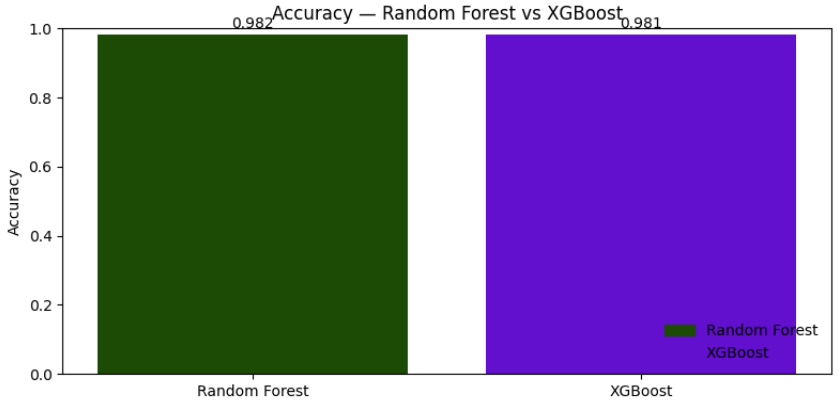
Accuracy			0.98	854
Macro Avg	0.98	0.98	0.98	854
Weighted Avg	0.98	0.98	0.98	854

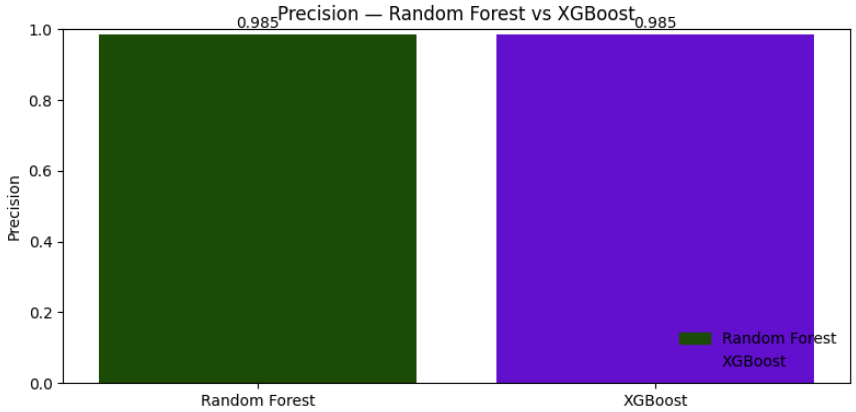
Feature Importance (Top Drivers)

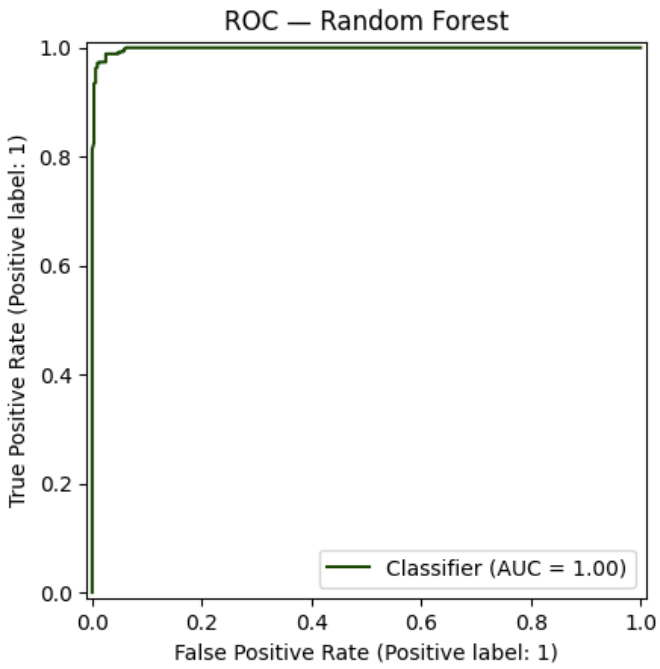
Feature importance helps identify which inputs most influence the prediction. In our case, variables like CIBIL score, loan term, and loan amount typically rank highly, aligning with lending intuition.

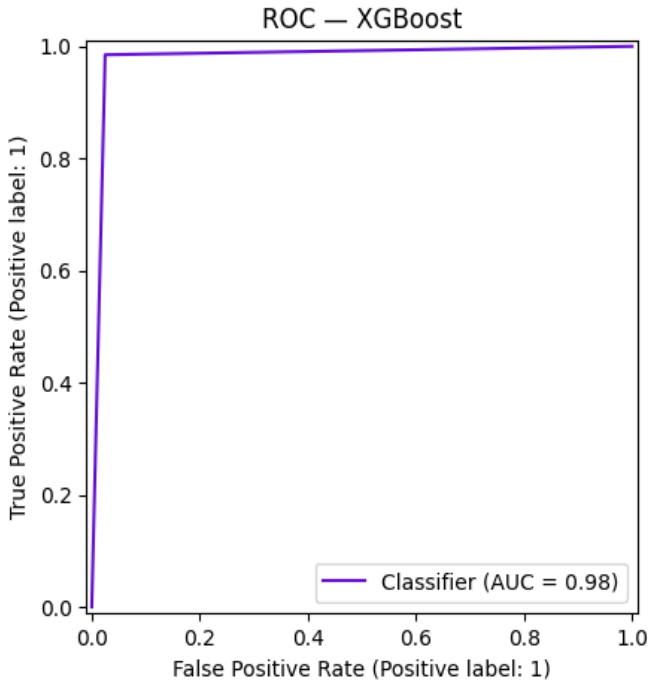


ROC Curves — RF vs XGB









Comparison table and Confusion Matrix

	Accuracy	Precision	Recall	F1	ROC_AUC
Random Forest	0.9824	0.9850	0.9868	0.9859	0.9984
XGBoost	0.9813	0.9849	0.9849	0.9849	0.9978

Random Forest Matrix

	Pred Reject	Pred Approve
Actual Reject	315	8
Actual Approve	7	524

XGBoost Confusion Matrix

	Pred Reject	Pred Approve
Actual Reject	315	8
Actual Approve	8	523

Inference (What We Learned)

Both models generalize well on unseen data.
The most influential features align with domain knowledge: credit quality and repayment capacity.

To avoid over-optimistic approvals, we enforce an EMI affordability gate so that monthly payments remain manageable relative to income.

- Model score + EMI rule = safer, more transparent decisions.
- Reports include EMI, monthly income, threshold, and a human-readable reason.

FastAPI Overview

The API loads the saved pipeline and the exact training feature list, aligns incoming JSON to training columns, computes a probability, and applies the EMI rule before returning the final decision.

Interest rate can be passed as 8.2 (percent) or 0.082 (fraction); tenure is provided in years and converted to months for EMI.

- Endpoints: /ping, /expected_features, /predict, /predict_batch.
- Outputs: model_pred, final_pred, prob_approve, EMI, monthly_income, threshold, reason.

Other Comments — Fine-tuning & Next Steps

To push performance and trust further, consider tuning hyperparameters, calibrating probabilities if thresholds matter, and adding model-agnostic explainability such as permutation importance and SHAP values.

In production, monitor approval rates and drift, and retrain on fresh data as needed.

- RandomizedSearchCV / GridSearchCV for RF & XGB.
- Cross-validation and probability calibration (e.g., Platt/Isotonic).
- SHAP/PDPs for interpretability; cost-sensitive thresholding.
- Monitoring: data/label drift, approval rate shifts; scheduled retraining.

References

- Scikit-learn — https://scikit-learn.org/
- XGBoost — https://xgboost.readthedocs.io/
- FastAPI — https://fastapi.tiangolo.com/
- Kaggle — Loan Prediction dataset

Thank you !