**Loan Default Prediction — Lending Club Dataset**

**📌 Problem Statement**

Predict whether a borrower will **default** on a loan using Lending Club historical loan application data. The dataset is **imbalanced**, making sensitivity (recall) and ROC-AUC critical evaluation metrics.

**🎯 Objective**

* Build a machine learning model to predict loan defaults.
* Handle imbalance in the dataset.
* Evaluate performance using **Sensitivity (Recall)** and **ROC-AUC**.

**📂 Dataset**

* Source: Lending Club loan data (loan\_data.csv)
* Target column: not.fully.paid
  + 0: Loan repaid
  + 1: Loan defaulted

**Dataset balance:**

* Repaid (0): ~84%
* Defaulted (1): ~16%

**🔧 Steps Performed**

1. **Load dataset** and inspect.
2. **Check for nulls** → no missing values found.
3. **Class distribution check** → confirmed strong imbalance.
4. **Handle imbalance** → applied **SMOTE oversampling**.
5. **Plot distributions** → visualized original vs balanced dataset.
6. **Encode categoricals** → one-hot encoding applied via pandas.
7. **Train/test split** with stratification; standardized numeric features.
   * Model used: **Scikit-learn MLPClassifier** (small NN).
   * Balanced logistic regression was also tested for comparison.
8. **Evaluate model** with Sensitivity, Precision, and ROC-AUC.
   * ROC Curve plotted.
   * Best threshold found via **Youden’s J statistic**.

**📊 Results**

* **Best Threshold (Youden J):** 0.4777
* **Sensitivity (Recall):** 0.697
* **Precision:** 0.738
* **ROC AUC:** 0.805
* **Train/Test split:** 80/20 (Train: 12,872 | Test: 3,218)

⚠️ Note: The model showed a **convergence warning** (didn’t fully converge in 20 iterations).  
Increasing max\_iter would likely improve training stability, but recall and AUC are already strong.

**📈 Interpretation**

* AUC of **0.805** indicates strong discriminatory power between repaid vs default loans.
* Recall of **~70%** means the model successfully catches ~70% of true defaulters.
* Precision of **~74%** shows most predicted defaults are correct.
* Class imbalance was effectively handled by SMOTE, leading to balanced training data.

**✅ Conclusion**

* A **neural network via scikit-learn (MLPClassifier)** provided robust performance without the instability of TensorFlow/Keras on limited resources.
* The model is a good baseline for **loan risk prediction**.
* Further improvements could include:
  + Hyperparameter tuning (increase max\_iter, adjust hidden layers).
  + Feature selection / engineering.
  + Comparing tree-based ensembles (e.g., XGBoost, RandomForest).