Policy Optimization for Financial Decision-Making

Objective:
This project is designed to assess your ability to handle a real-world machine learning problem from start to finish. You will take a public dataset, perform analysis, frame it for both supervised learning and reinforcement learning, build and train models, and—most importantly—critically analyze and compare their behaviors and outcomes.

This task will test your end-to-end skills in:

- Data analysis and feature engineering.
- Building and evaluating deep learning models for prediction.
- Framing a supervised problem as an offline RL problem.
- Applying modern offline RL frameworks to learn a decision-making policy.
- Synthesizing results and communicating your findings and future strategy.

The Business Context:
Imagine you are a Research Scientist at a fintech company. The company wants to improve its loan approval process. You have access to a historical dataset of all loans granted in the past, including applicant details and whether they ultimately defaulted or paid off the loan.

The goal is to develop an intelligent system that can decide whether to **approve** or **deny** a new loan application to maximize the company's financial return.

The Dataset: LendingClub Loan Data

For this project, you will use the well-known **LendingClub Loan Data** dataset. It contains information on accepted and rejected loans, along with borrower information.

- **You can download the dataset directly from Kaggle:** LendingClub Loan Data
- **Focus:** Please use the accepted_2007_to_2018.csv file. Due to its size, you may want to sample it for faster iteration, but your final models should be trained on a substantial portion of the data.

The Core Tasks

Task 1: Exploratory Data Analysis (EDA) and Preprocessing

Before building any models, you must understand and clean the data.

1. **Analyze the Data:** Perform an initial EDA to understand the features, identify missing values, and understand the distribution of key variables (e.g., loan status, interest rate, applicant income).
2. **Feature Engineering & Selection:** The dataset has many columns. Select a meaningful subset of features that you believe would be predictive of loan default. Justify your choices.
3. **Data Cleaning:** Preprocess your selected features. This will involve handling missing values, encoding categorical variables, and feature scaling. Document your steps clearly.

Task 2: Model 1 - The Predictive Deep Learning Model

Your first model will be a standard supervised classifier that predicts the risk of default.

1. **Define the Target:** Your target variable will be loan_status. You will need to convert this into a binary target: {0: Fully Paid, 1: Defaulted}. (You can consider "Charged Off" as defaulted).
2. **Build and Train:** Implement a deep learning model (e.g., a Multi-Layer Perceptron using PyTorch or TensorFlow) to predict the probability of default given the applicant's features.
3. **Evaluate:** Train your model and evaluate its performance using standard classification metrics. You must report the **AUC (Area Under the ROC Curve)** and **F1-Score** on a held-out test set.

Task 3: Model 2 - The Offline Reinforcement Learning Agent

Now, frame this as an offline RL problem where an agent learns a *policy* to approve or deny loans.

1. **Define the RL Environment (from the static dataset):**
    - **State (s):** The vector of preprocessed features for a given loan applicant.
    - **Action (a):** A discrete action space of {0: Deny Loan, 1: Approve Loan}.
    - **Reward (r):** This is the crucial part you need to engineer. A simple reward structure could be:
        - If action == Deny: reward = 0 (No risk, no gain).
        - If action == Approve and the loan was Fully Paid: reward = + (loan_amnt * int_rate) (Profit from interest).
        - If action == Approve and the loan was Defaulted: reward = - loan_amnt (Loss of the principal).
2. **Train an Offline RL Agent:**
    - Use a modern offline RL framework to train an agent.
    - Choose a suitable offline RL algorithm.
    - Train your agent on the dataset you've prepared.

Task 4: Analysis, Comparison, and Future Steps

This is the most important part of the project. Your goal is to synthesize your findings into a coherent analysis.

1. **Present Your Results:** In your final report, clearly present the key metrics for both models (AUC/F1 for the DL model, and the **Estimated Policy Value** for the RL agent, which can be calculated using your chosen library).
2. **Explain the Difference in Metrics:**
    - Why are AUC and F1-Score the right metrics for the DL model? What do they tell us about its capabilities?
    - Why is "Estimated Policy Value" the key metric for the RL agent? What does it represent in the context of our business problem?
3. **Compare the Policies:**
    - Your DL model implicitly defines a policy (e.g., "approve if predicted default probability < threshold"). Your RL agent learns a policy directly.
    - Find examples of applicants where the two models would make **different decisions**. For instance, find a high-risk applicant that the DL model would flag but the RL agent still approves. Why might the RL agent do this? (Hint: Think about reward.)
4. **Propose Future Steps:**
    - Based on your findings, what would you do next? Would you deploy one of these models?
    - What are the limitations of your approach?
    - What other data would you want to collect? What other algorithms might you explore?

How to Submit

You will be asked to provide the following three items:

1. **Your Resume/CV (PDF upload)**
2. **Link to your GitHub Repository**
    - Your repository must be public and well-organized.
    - It should contain all the source code (as Jupyter Notebooks or .py scripts) for your EDA, DL model, and RL agent.
    - It **must** include a README.md file with clear, step-by-step instructions on how to set up the environment (e.g., via a requirements.txt file) and run your code to reproduce your findings.
3. **Final Report (PDF upload)**
    - A concise (2-3 page) report that walks through your entire process.
    - This document should provide detailed answers to all the questions outlined in **Task 4**. This report is the primary artifact we will evaluate to understand your thinking.

Evaluation Criteria

You will be evaluated on:

- **Analytical Rigor:** The quality and thoughtfulness of your EDA and data preprocessing choices.
- **Technical Execution:** The correctness of your model implementations and the reproducibility of your code via the GitHub repo.
- **Depth of Analysis:** This is the most critical criterion. How deeply you analyze your results, explain the "why" behind the numbers, and compare the two modeling paradigms in your report.
- **Communication:** The overall clarity of your README.md file and your final report.