Report — Lending Club: Default Prediction & Policy Optimization

Deep Learning (Supervised) + Offline RL (Contextual Bandit)

1. Problem & Data (Business Context)

We act as Research Scientists at a fintech company seeking to improve loan approval decisions. We use the LendingClub accepted loans dataset (2007–2018) to build: (a) a deep learning classifier that predicts default risk; and (b) an offline reinforcement learning (RL) agent that directly learns a profit-maximizing approval policy.

Dataset: accepted_2007_to_2018Q4.csv.gz (~2.26M rows, 151 columns).

Target mapping:

- Fully Paid → 0
- Charged Off/Default/Policy Charged Off → 1.

We keep leakage-free features available at decision time (e.g., loan amount, interest rate, term, employment length, FICO ranges, DTI, revol utilization, purpose, home ownership, state, verification).

2. EDA & Preprocessing (Task 1)

Key engineered features:

- int_rate_num (percentage→fraction)
- revol_util_num (clipped to 150%)
- term_months (36/60)
- emp_len_yrs (10+→10, <1→0.5)
- credit_hist_mths (months between earliest credit line and issue date)

Numeric features were median-imputed and standardized; categorical features were imputed with most-frequent and one-hot encoded. We performed a strict time split: Train ≤ 2016, Val = 2017, Test = 2018.

3. Model 1 — Deep Learning Classifier (Task 2)

Architecture: PyTorch MLP on tabular features (88-dim after preprocessing), with ReLU, BatchNorm, Dropout, and class-imbalance weighting.

Objective: binary cross-entropy on default vs non-default.

Evaluation metrics and results:

Metric	Value	Notes
AUC (TEST)	0.708	Ranking quality (probability a defaulter scores higher risk than a non-defaulter).
F1 (TEST @ t=0.50)	0.369	Precision–recall balance at a fixed threshold; shows classification trade-off.
Threshold selection	t selected on VAL	We picked the threshold on validation (reported 0.50 baseline); see policy section for profit-optimal tuning.

Why AUC & F1?

• AUC is threshold-agnostic and measures overall risk ranking ability, which is essential for prioritizing applicants.
• F1 is suited to imbalanced defaults (~20%), balancing precision (false positive control) and recall (catching defaulters) at a chosen threshold.

4. Model 2 — Offline RL Agent (Task 3)

We cast approval as a one-step contextual bandit:

state = applicant features

actions = {Deny, Approve}

reward = +(loan_amnt × int_rate × term_years) if Fully Paid

−(loan_amnt) if Default, 0 if Deny.

We trained a Direct Method (DM) policy by regressing expected approve-reward with XGBoost, then approving if predicted reward > 0.

Policy evaluation on TEST via Estimated Policy Value (EPV):

Policy	EPV (Mean Reward / Applicant)	Approval Rate
DL @ t=0.50	1,536.59	57.33%
DL @ t=0.95 (profit-optimal)	3,553.38	100.00%
RL (XGB Direct Method)	3,494.66	96.96%

Why EPV?

Estimated Policy Value reflects actual business objective — expected profit per applicant under a policy. Unlike AUC/F1, EPV directly answers: “How much money does this policy make on average?”

5. Policy Comparison & Disagreement Analysis (Task 4)

The DL model implicitly defines a policy via a threshold (approve if predicted default probability < t). The RL policy directly maximizes expected reward. We observed that high thresholds (t≈0.90–0.95) and the RL policy approve most applications but retain positive EPV due to the reward design, which values high interest/longer terms.

Examples where policies disagree (TEST):

• CB Approves / DL@0.50 Denies: n=22361. These are often moderately risky (higher default probability) but have high interest and longer terms, so expected profit remains positive.
• CB Denies / DL@0.50 Approves: n=40. These tend to have low predicted profit (e.g., lower rate/term), so RL avoids them despite low default probability.

6. Limitations & Future Steps

Limitations

• Reward is simplified (no fees/servicing costs/recoveries/prepayment), biasing toward approve-most.
• Logged data bias: we only observe outcomes for historically approved loans; robust off-policy evaluation needs logging propensities.
• No business constraints: real credit policies enforce approval caps, bad-rate limits, capital/exposure constraints, fairness & compliance.

Next Steps

• Constraint-aware optimization: maximize EPV subject to approval rate ≤ X% or bad-rate ≤ Y% (choose threshold on VAL, report on TEST).
• Enhanced rewards: add origination fees, servicing costs, expected recovery curves, and prepayment assumptions.
• Better off-policy evaluation: try Doubly Robust estimators; once versions are stable, compare with CQL/IQL in d3rlpy.
• Interpretability & governance: SHAP for the bandit regressor; fairness audits and stability checks across cohorts/time.

7. Conclusion

The supervised model shows solid ranking (AUC≈0.71) but moderate F1 at a standard threshold. When we align decisions with profit via thresholds or a bandit policy, Estimated Policy Value improves substantially. This underscores a key lesson: in lending, policy tuning around business rewards and constraints matters more than raw classification scores.