Policy Optimization for Financial Decision-Making

Final Report

Author: Mayank Jyani (229303081)
Date: December 2025
Project: Lending Club Loan Default Prediction — Supervised Learning vs Offline RL

Executive Summary

This project compares a supervised deep-learning classifier and an offline RL Q-agent to decide loan approvals with the business objective of maximizing realized financial return. On the test set the RL Q-agent achieved an average reward of 0.042508 per loan (total test reward = 1700.34) with an approve rate of 61.07%. The supervised deep model attains an F1 = 0.7090 on the approve/paid class. Recommendation: validate uplift with bootstrap CI, then run a staged A/B trial using safety overrides before production deployment.

---

1. Data Analysis & Preprocessing

Dataset overview
- Source: Lending Club (accepted_2007_to_2018.csv), processed into project splits under `data/processed`.
- Splits used: train / val / test (parquet files).
- Default / charged-off rate: reported in EDA notebook (`src/eda_notebook.ipynb`).

Feature selection & engineering
- Features used: standard credit features (loan_amnt, int_rate, annual_inc, dti, fico-like indicators), plus engineered ratios (income_to_loan, installment_to_income).
- Rationale: These features correlate with repayment capacity and the loan's economic value (loan amount × interest).

Preprocessing
- Missing values: numeric median, categorical mode (documented in preprocessing pipeline).
- Encoding: label / ordinal encoding for categories; one-hot where appropriate.
- Scaling: StandardScaler applied to numeric features for MLP input.
- Final feature list validated against `data/rl/feature_cols.csv`.

---

2. Model 1 — Supervised Deep Learning

Setup
- Architecture: MLP (input → 256 → 128 → 64 → 1), dropout ≈ 0.3, BCEWithLogitsLoss.
- Optimizer: Adam; early stopping on validation AUC.

Training & evaluation
- Target: binary (0 = Fully Paid, 1 = Default/Charged Off).
- Test metrics (threshold = 0.5 unless noted):
- F1 (approve / paid class): **0.7090**
- Accuracy (example eval): ≈ **0.5894**
- AUC-ROC: (compute from eval logs; insert exact value) — recommended to report here.

Interpretation
- AUC measures discrimination independent of threshold.

- F1 balances precision and recall and is appropriate for imbalanced classification where both false positives and false negatives matter.
- As a decision policy: supervised model → "approve if P(default) < threshold". This policy does not directly maximize profit.

---

3. Model 2 — Offline Reinforcement Learning

RL formulation
- State: preprocessed feature vector for an applicant.
- Action: {0 = Deny, 1 = Approve}.
- Reward:
  - Deny: r = 0
  - Approve & Paid: r = + loan_amnt × (int_rate / 100)
  - Approve & Default: r = - loan_amnt

Agent & training
- Agent: MLP Q-network (2 outputs) trained offline on logged data (Q-learning style). Conservative algorithms (CQL/IQL) are recommended for production to limit OOD overestimation.
- Checkpoint: `models/rl_q/q_agent_best.pth`
- Evaluation: `src/eval_q_agent.py` produces `models/rl_q/q_agent_test_predictions.csv`

Results (test)
- Avg reward (per loan, RL policy): **0.042508**
- Total reward (test): **1700.34**
- Approve rate (RL): **61.07%**
- Bootstrap script included: `src/bootstrap_policy_test.py` to compute CI for RL vs supervised reward differences.

Interpretation
- Estimated policy value directly measures expected monetary return — the core business objective.
- RL policy approves loans that are profitable in expectation (considering loan_amnt and int_rate), even if they have higher default probability.

---

4. Comparison & Case Analysis

Key differences
- Supervised model focuses on predicting default probability (AUC / F1).
- RL optimizes expected reward (profit) and naturally weighs loan amount and interest rate.

Example behaviors
- High-risk, high-interest loan:
  - Supervised: may DENY due to high P(default).
  - RL: may APPROVE if EV = (1 − p_default) × interest_gain − p_default × loan_amnt > 0.
- Low-stake low-interest loan:
  - Supervised: may APPROVE (low default probability).
  - RL: may DENY if expected interest profit is too small to justify risk.

Empirical policy disagreement
- Disagreements cluster where loan amount or interest rate strongly shifts expected value.
- These cases are where RL adds most value but also require careful business validation.

---

5. Discussion, Limitations & Recommendations

Limitations

- Offline data bias: historical data mostly contains approved loans → limited counterfactuals for denied actions.
- Reward simplification: recovery rates, partial repayments, fees, and collection costs are not modeled.
- Temporal drift: dataset (2007–2018) may not reflect current macroeconomics.
- Fairness: demographic biases must be evaluated and mitigated.

Recommendations
1. Run the bootstrap script (`src/bootstrap_policy_test.py`) and confirm 95% CI for RL vs supervised avg reward excludes zero.
2. If positive, deploy RL policy in a staged A/B experiment with safety overrides:
   - Safety rule: block RL approvals when supervised P(default) > threshold (e.g., 0.7).
3. Use conservative offline-RL (CQL / IQL) and importance-sampling / FQE for more robust evaluation.
4. Expand reward model with recovery rates, fees, and cost-of-capital to match business P&L.
5. Add fairness audits and regulatory documentation before production.

---

6. Conclusion

- The RL agent demonstrates a measurable uplift in estimated financial return (avg reward = **0.042508** per loan) and approves ~61% of applicants — a profitable, outcome-driven policy compared to a classifier tuned for predictive accuracy (F1 = **0.7090**).
- The appropriate production approach is a hybrid: adopt RL for profit optimization, with supervised-model-based safety checks, conservative training algorithms, and phased A/B rollout with monitoring.
