

FINAL REPORT

1. Presentation of Results

In this project, I tried to compare two different ways of making loan-approval decisions:

1. A Deep Learning (DL) model that predicts the probability of default.
2. An Offline Reinforcement Learning (RL) agent trained with Conservative Q-Learning (CQL) to directly maximize long-term expected profit.

Here are the results I got:

Deep Learning Model

- **AUC:** 0.7099
- **F1-Score (0.5 threshold):** 0.3124
- **Best F1 (optimized threshold):** ~0.34

These scores suggest the model can reasonably separate risky borrowers from safe ones, although not perfectly (dataset imbalance probably makes it hard).

Reinforcement Learning Agent

- **Estimated Policy Value (EPV):** 166700.2

EPV basically means:

If we use this RL policy in real life, the expected average profit would be around 166k.

2. Why These Metrics Matter

Why AUC + F1 for a DL classifier?

- **AUC** tells how well the model ranks borrowers by risk. Since defaults are much fewer than non-defaults, accuracy wouldn't make sense, so AUC becomes a better choice.
- **F1-Score** cares about precision and recall, which is important because mistakes cost money on both sides:
 - Approving a risky borrower → big loss
 - Rejecting a good borrower → lost revenue

So AUC tells how well the model distinguishes risk, and F1 tells how well it catches actual defaulters. Together they give a good idea of the model's skill.

Why EPV for the RL agent?

The RL model doesn't try to classify at all. Its goal is just:

“Choose the action (approve/deny) that leads to maximum profit.”

So the only metric that matters is **Estimated Policy Value**, which directly measures how much money the policy expects to make.

This matches the actual business goal much better, because lenders care about profit, not classification accuracy.

3. Comparing the Policies: DL vs RL

DL Model's Decision Rule

The DL model follows a simple rule:

Approve the loan if predicted default probability < threshold.

It doesn't care about profit, interest rate, or reward — it only cares about “will default or not.”

RL Model's Decision Rule

The RL model is different. It learns a policy by understanding the financial reward:

- If borrower pays: **+profit (loan_amount × interest_rate)**
- If borrower defaults: **−loan_amount**
- If denied: **0**

So sometimes RL might approve a borrower who looks risky if the potential interest earnings are high enough.

Example Where DL Rejects but RL Approves

Borrower Profile (example):

Feature	Value
FICO Score	670
DTI	25%
Loan Amount	20,000
Interest Rate	18%
Employment	2 years

DL Model:

Default probability = 0.42 → Threshold = 0.34 → **Reject**

RL Agent:

Expected Value \approx

$(0.8 \times 3600) - (0.2 \times 20000) = +880 \rightarrow$ **Approve**

So RL approves because it sees net positive expected profit, while DL just sees a high probability of default and says no.

This shows how RL may sometimes make “smarter” financial decisions.

4. Future Steps & What I'd Do Next

Which Model Should Be Deployed?

Honestly, I would **deploy the DL model first** because:

- It's way simpler to explain
- It's more stable
- It's less risky for regulators

And I'd use the RL model more like a **decision-support tool** or simulator right now instead of something directly in production.

Limitations in This Project

Some limits I noticed:

- Not much behavioral or time-series data
- Reward function is simplified (no operational costs or regulation rules)
- RL is single-step, so it doesn't model long-term borrower behavior
- EPV is based only on the historical labels, not a realistic simulation environment

Data I Wish I Had

- Borrower payment history over time
- Bank transaction patterns
- Changes in employment and income
- Macroeconomic indicators
- Early repayment behavior
- More detailed costs of servicing loans

With these, the RL model could be way better.

Future Algorithms Worth Trying

For DL:

- XGBoost / LightGBM
- TabNet
- Uncertainty models like Bayesian NNs

For RL:

- Batch-Constrained Q-Learning (BCQ)
- BRAC
- SAC (Soft Actor Critic)
- Risk-sensitive RL (like CVaR)

These would help reduce extreme risky approvals and give more stable profits.

Final Summary

- DL Model:
 - AUC = **0.7099**
 - F1 = **0.3124**
 - Good at separating risky borrowers
- RL Model:
 - EPV = **166700.2**
 - Focuses on maximizing profit directly
- The two models don't always agree. RL sometimes approves borrowers that DL rejects because RL looks at profit, not just probability of default.
- For now, DL is safer for deployment, while RL is promising but needs more validation and better data.