Task 4: Analysis, Comparison, and Future Steps

This project utilizes two modeling techniques to predict loan default risk and optimize lending decisions using historical LendingClub data:

- A **supervised learning classifier (MLP)** trained to predict loan default risk (focuses on predicting the likelihood of default based on historical labels)
- An **offline reinforcement learning agent (DQN)** trained to learn a policy that maximizes financial return (an approach that learns decision strategies (approve/deny loans) by optimizing for long-term cumulative rewards)

Exploratory Data Analysis and Preprocessing

- **Data Overview:**

The dataset comprised applicant-level loan information with **numerical and categorical features** representing **demographic, financial and behavioural attributes**. The target variable was **binary**:
- **1 :** Loan Default
- **0 :** Loan Fully Paid

The dataset was **moderately imbalanced** with defaults representing a smaller proportion of the total records - motivating class imbalance adjustment in the supervised model and reward-driven training in the RL agent. As a result - a **stratified subset of 100,000 applicant records** was used to ensure balanced representation across default and non-default classes while maintaining computational efficiency.

- **Handling Missing Data:**

Multiple continuous variables (such as annual income and credit-to-loan ratios) contained missing entries. Missing values were handled via:
- **Median imputation** for numerical columns using fillna(),
- **Mode-based imputation** for categorical variables where applicable.

This ensured the model did not lose data diversity while maintaining consistent input dimensionality.

- **Feature Scaling and Encoding:**

All **continuous values** were standardized using **StandardScaler**, transforming them to **zero mean and unit variance**.
Categorical features were **label-encoded** to integer form and cast using astype('int') to ensure compatibility with the neural network and DQN inputs.

This normalization was crucial for both techniques:
- The **MLP classifier** relies on uniform input scales for stable gradient updates.
- The **DQN agent** benefits from scaled states that improve Q-value convergence.

- **Dataset Partitioning**

For the supervised model:

- **Training/Validation split:** 80:20 ratio.
- **Batch size:** 256 samples.
- **Shuffling:** Enabled to reduce bias across mini-batches.

For the reinforcement learning setup:

- The dataset was transformed into **MDP (Markov Decision Process)** format.
- **Synthetic "denied" actions** were added to balance the action space (approve, deny) and allow the agent to evaluate counterfactual outcomes.

- **Data Cleaning**

The original **loan_status** contained multiple categories (e.g., *Current*, *Late*, *In Grace Period*, *Fully Paid*, *Charged Off*, *Default*).
For this analysis, only **“Fully Paid”**, **“Charged Off”**, and **“Default”** entries were retained to create a clear binary classification target representing loan outcome.
Intermediate statuses were excluded because they indicate ongoing loans without finalized repayment outcomes, which could introduce label ambiguity into both the supervised and reinforcement learning formulations.

Model 1: Supervised Deep Learning Classifier

A Multilayer perceptron with:

- **Input layer :** 128 -> ReLU -> Dropout (0.3)
- **Hidden layer :** 64 -> ReLU -> Dropout (0.3)
- **Output layer :** 1 logit

- **Loss:** BCEWithLogitsLoss with class imbalance adjustment
- **Optimizer:** Adam (lr=1e-3)
- **Batch size:** 256
- **Epochs:** 10

Performance (Validation set)

Table 1

Metric	Value
AUC	0.9990
F1-score	0.9891
Final Train Loss	5.3112

Confusion Matrix

Table 2

Actual / Predicted	No default (0)	Default (1)
No default (0)	5566 (TN)	10 (FP)
Default (1)	21 (FN)	1402 (TP)

This model demonstrates a near-perfect classification performance with minimal misclassification.

Model 2: Offline Reinforcement Learning Agent

Components:

- **State (s):** Preprocessed applicant features
- **Action (a):** {0: Deny Loan, 1: Approve Loan}
- **Reward (r):**

- Approve and fully paid : +loan_amnt x int_rate
- Approve and Defaulted : -loan_amnt
- Deny : 0

- **Algorithm:** Deep Q-Network (DQN) via d3rlpy
- **Dataset:** MDP format with synthetic denied loans added
- **Training steps:** 50,000
- **Loss** reduced from 0.0109 to 0.000095 over 5 epochs.

Evaluation metrics:

Table 3

Metric	**Value**
Estimated Policy Value (Avg Predicted Reward)	0.8235
Approvals Predicted	78
Denials Predicted	296
Avg Reward for Approvals	-915.60
Avg Reward for Denials	0.000

Analysis :

Table 4

Model	Metric	Value	Interpretation
Deep learning Classifier	AUC	0.9990	Near-accurate ability to distinguish between defaulters and non-defaulters
	F1-score	0.9891	Good balance between precision and recall
Offline RL agent	Estimated Policy Value	0.8235	Average expected return per decision under learned policy (reward-weighted loan profitability)

The **Deep Learning model** is evaluated using **AUC and F1-score** which are ideal for binary classification tasks:

- **AUC (Area under the ROC curve):** measure the model's ability to rank applicants by risk. An AUC close to 1 (0.9990 here; **see Table 1**) means that the classifier can almost accurately separate good borrowers from defaulters - which is crucial when decisions are threshold based.
- **F1-score (0.9891; see table 1):** balances **precision** (i.e. avoiding false positives, i.e. rejecting reliable customers) and **recal**l(catching likely defaulters) , which is crucial in imbalance datasets like loan defaulters. In a loan setting, **high precision** protects the customer trust, while a **high recall** value minimizes financial losses due to defaults.

Estimated Policy Value for the RL agent:

The Estimated Policy Value (0.8235; **see Table 3**) measures the **average predicted return** per loan decision under the learned policy.

- It captures both profit from successful loans and penalties from defaults, thus aligning directly with the company's financial objective: maximizing overall lending profitability.
- This metric is a **direct business analogue** AUC/F1 - but for decision optimization instead of probability estimation.

Comparison :

- The **DL mode**l defines an implicit policy:
  approve if **predicted default probability < threshold(eg: 0.5)**. It is against risk and optimized for classification accuracy.

- The **RL agent** learns a policy directly from reward signals. It may approve high-risk applicants if the **expected reward (loan_amnt x int_rate)** outweighs the potential loss.

Example Divergence:

On one hand, an applicant with a **low FICO score, high DTI and a short employment history** might be rejected by the **DL model** due to **high predicted risk.**
On the other hand, the **RL agent** may still approve if the **loan amount is small and the interest rate is high**, resulting in a **net positive expected reward**.

Future Steps:

Given the higher accuracy and interpretability, the **DL model** is well-suited for initial deployment. It offers a safe, risk-aware strategy ideal for production use.
The **RL agent**, while more reward-driven, could complement the DL model in a hybrid setup - especially for borderline or high interest cases.
So ideally **deploying the DL model** will be considered first, with the RL agent being reserved for further testing and refinement.

Limitations:

- RL agent trained on static data with no exploration.
- Reward function does not capture late payments or long term value
- The DL model focuses on prediction, not financial outcomes.

Next Steps:

- Collecting richer repayment and borrower behaviour data.
- Trying advanced offline RL methods like CQL or IQL
- Simulating deployment to test real-world impact.