Recommendation System using Reinforcement Learning (RL)

Summary Report

🔧 1. RL Setup

- **Environment**: A custom OpenAI Gym-style environment (RecommendationEnv) was built to simulate user responses to marketing actions based on real-like user profile features.
- **State Space**: Each user is represented by 10 numerical features:
    - recency, frequency, monetary_value, engagement_score
    - discount_sensitive, free_shipping_preferred
    - email_opens, sms_response, push_response, last_purchase_days
- **Action Space**: A discrete list of recommendation strategies (e.g., sending a discount via email or SMS).
- **Reward Function**: Designed to simulate user satisfaction:
    - +0.5 for correct targeting (e.g., discounts for sensitive users)
    - +0.3 for matching preferred channels (e.g., email for high open rates)
    - Clipped total reward ∈ [0, 1]
- **Algorithm Used**: [Proximal Policy Optimization (PPO)] from Stable-Baselines3
    - Chosen for its reliability in discrete action spaces and ease of tuning.
- **Vectorized Environment**: DummyVecEnv used to support batched training.

2. Key Takeaways

- **Personalization**: RL effectively learned to recommend personalized strategies based on user behavior profiles.
- **Reward Optimization**: The agent quickly learned to maximize rewards by aligning actions with user preferences (e.g., discount + email for sensitive users with high open rate).
- **Interpretability**: The state and actions remain interpretable, useful for debugging and business insights.

3. Sample Outputs

Example simulation for a trained agent:

User Features:

recency: 15.00

frequency: 3.00

monetary_value: 200.00

engagement_score: 0.78

discount_sensitive: 1.00

free_shipping_preferred: 0.00

email_opens: 0.87

sms_response: 0.12

push_response: 0.05

last_purchase_days: 8.00

Agent Action Chosen:

Action #2 → ['discount', 15%, 'email']

Reward Achieved: 0.9

The agent correctly detected discount sensitivity and matched the preferred communication channel (email), leading to a high reward.

4. Suggestions for Scaling to Real Users

- **Data Collection**: Integrate with real e-commerce user logs to train on actual user behavior rather than simulated profiles.
- **Reward Tuning**: Incorporate actual user engagement metrics (click-through rates, conversion, etc.) into the reward function.
- **Action Expansion**: Support more complex actions such as bundling, timing optimization, or dynamic discounts.
- **Model Interpretability**: Add SHAP or LIME-based explainability modules to help non-technical stakeholders understand recommendations.
- **Continuous Learning**: Deploy in an online learning setup where the model adapts based on real-time feedback from users.

5. Future Work

- Integrate deep user embeddings using unsupervised learning to enrich state space.
- Test with other RL algorithms like A2C, DQN for comparison.
- Deploy as a REST API for real-time recommendation serving.
- Explore multi-agent setups for group recommendations or campaign optimization.