LAB - 6

By

Nishant Rodrigues

2348045

5 MSc Data Science

(Reinforcement Learning)

Introduction

Incorporating AI-driven systems into healthcare and financial markets has immense potential to revolutionize decision-making processes. These domains present unique challenges, such as balancing short-term actions with long-term outcomes, managing uncertainty, and deriving insights from real-time or simulated data streams. Temporal Difference (TD) Learning, a core reinforcement learning (RL) technique, offers a robust framework for tackling these challenges. By iteratively improving its value estimation, TD Learning enables agents to adapt to dynamic environments and optimize decision-making strategies.

In this document, we explore two distinct applications of TD Learning:

1. **Healthcare System for Real-Time Patient Monitoring**: Developing an AI-driven healthcare system that adjusts treatment plans based on real-time health data such as heart rate, blood pressure, and glucose levels. The system aims to balance immediate health improvements with long-term patient well-being using TD Learning.
2. **Reinforcement Learning Trading Agent**: Designing a trading agent to maximize cumulative profits while minimizing transaction costs in the stock market. Using historical or simulated data, the agent applies TD Learning to make optimal buy, sell, or hold decisions in fluctuating market conditions.

Overview of the Tasks

Task 1: Healthcare System for Patient Monitoring

1. **Real-Time Data Integration**: Use real-time medical data (or a simulated dataset) involving metrics such as heart rate, blood pressure, and glucose levels.
2. **Decision-Making with TD Learning**: Implement a reinforcement learning agent that decides on treatment adjustments, including medication dosage and dietary recommendations, based on the data.
3. **Evaluation Metrics**: Examine the impact of different discount factor (γ) values on short-term and long-term patient outcomes.
4. **Learning Process Analysis**: Monitor the agent's evolving strategy over time and evaluate its performance through visualizations and metrics.

Task 2: Trading Agent for Stock Market

1. **Stock Market Simulation**: Use a real or simulated stock price dataset with historical patterns.
2. **Profit Maximization with TD Learning**: Develop an agent that makes optimal buy, sell, or hold decisions to maximize cumulative profit over a trading period.
3. **Strategy Evolution**: Evaluate how the agent's strategy evolves over time and analyze the impact of γ on its decision-making and performance.

4. **Implementation in OpenAI Gym**: Design and train the agent in a custom OpenAI Gym environment that mimics the stock market dynamics.

Objectives

For the Healthcare System:

1. Build an adaptive AI system capable of monitoring real-time patient data and making treatment adjustments.
2. Utilize TD Learning to balance immediate health improvements with long-term well-being.
3. Assess the impact of varying discount factors (γ) on patient outcomes.
4. Analyze and document the system's learning process, strategy evolution, and overall impact on patient care.

For the Trading Agent:

1. Design a reinforcement learning agent that can effectively trade stocks by maximizing profit and minimizing transaction costs.
2. Leverage TD Learning to enable the agent to learn from fluctuating market conditions.
3. Study the effect of γ on trading strategies and performance metrics.
4. Demonstrate the agent's learning curve and decision-making process with visualizations.

Task 1

Reinforcement Learning Trading Agent Report

Introduction

This report presents the implementation and evaluation of a trading agent using Reinforcement Learning with Temporal Difference (TD) Learning. The agent learns to optimize trading strategies for stocks by making decisions to buy, sell, or hold based on fluctuating market conditions. The goal is to maximize cumulative profit, minimize transaction costs, and avoid unnecessary trades.

Methodology

1. Simulated Stock Market Environment

The stock market environment was simulated using a custom OpenAI Gym environment. The agent observes daily stock prices and makes decisions based on predefined actions:

- Buy

- Sell
- Hold

The environment tracks the agent's net worth, balance, and the number of shares owned, while applying a transaction cost to simulate real-world trading expenses.

2. Temporal Difference Learning Agent

The agent uses the Temporal Difference (TD) Learning method to learn optimal trading strategies. The Q-table stores the expected rewards for state-action pairs, and the Q-values are updated using the formula:

$$Q(s, a) = Q(s, a) + \alpha [r + \gamma \max Q(s', a') - Q(s, a)]$$

Where:
- α (alpha) is the learning rate.
- γ (gamma) is the discount factor.
- r is the reward.
- s and a represent states and actions.

Results and Observations

1. Impact of Discount Factor (γ)

The discount factor γ determines the weight of future rewards relative to immediate rewards. The agent's performance was evaluated with different values of γ:
- γ = 0.5: Focuses on immediate rewards, leading to frequent trades and higher transaction costs.
- γ = 0.9: Balances immediate and future rewards, resulting in moderate trading activity and better profits.
- γ = 0.99: Prioritizes long-term rewards, leading to fewer trades but higher cumulative profit.

A higher discount factor generally resulted in a more strategic trading approach with improved profitability.

2. Evolution of Strategy Over Time

Initially, the agent explored various actions randomly due to the high exploration rate (epsilon). Over time, as the agent learned from the environment, it reduced unnecessary trades and focused on maximizing profit. The strategy evolved to:
- Buy at low prices when sufficient balance is available.
- Hold shares during periods of price stability.
- Sell during price spikes to realize profits.

3. Performance Visualization

The following metrics were tracked over the simulation period:
- Net worth over time
- Actions taken by the agent
- Balance and shares owned at the end of the trading period.

Conclusion

The TD Learning-based trading agent successfully learned to maximize cumulative profit over the trading period. Key observations include the impact of the discount factor on strategy and

performance, as well as the gradual evolution of the agent's trading decisions. The agent demonstrated the ability to adapt to market fluctuations and optimize trading outcomes.

```python
# Temporal Difference Agent
class TDTradingAgent:
    def __init__(self, env, alpha=0.1, gamma=0.99, epsilon=0.1):
        self.env = env
        self.alpha = alpha
        self.gamma = gamma
        self.epsilon = epsilon
        self.q_table = defaultdict(lambda: np.zeros(env.action_space.n))

    def choose_action(self, state):
        state = tuple(state)  # Convert numpy array to tuple
        if np.random.rand() < self.epsilon:
            return self.env.action_space.sample()  # Explore
        else:
            return np.argmax(self.q_table[state])

    def train(self, episodes):
        for episode in range(episodes):
            state = tuple(self.env.reset())  # Convert state to tuple
            done = False
            while not done:
                action = self.choose_action(state)
                next_state, reward, done, _ = self.env.step(action)
                next_state = tuple(next_state)  # Convert next_state to tuple
                best_next_action = np.argmax(self.q_table[next_state])

                # Update Q-value
                self.q_table[state][action] = (1 - self.alpha) * self.q_table[state][action] + \
                                              self.alpha * (reward + self.gamma * self.q_table[next_state][best_next_action])

                state = next_state  # Move to the next state
```

```
Step: 67, Price: 6781.707265660906, Net Worth: 80628.45773772332
Step: 68, Price: 6882.430537711001, Net Worth: 81852.54252727273
Step: 69, Price: 6981.140298201791, Net Worth: 83061.22179187388
Step: 70, Price: 7081.863089412807, Net Worth: 84245.73891776334
Step: 71, Price: 7184.939162545739, Net Worth: 85454.41241229554
Step: 72, Price: 7284.867510467519, Net Worth: 86681.32528989072
Step: 73, Price: 7387.9967977791475, Net Worth: 87770.5371170303
Step: 74, Price: 7482.757307570968, Net Worth: 89008.08856476985
Step: 75, Price: 7584.401112579719, Net Worth: 90145.21468227169
Step: 76, Price: 7684.575206716195, Net Worth: 91364.94034237669
Step: 77, Price: 7783.977192015263, Net Worth: 92567.0294720144
Step: 78, Price: 7884.160713568334, Net Worth: 93759.85329560323
Step: 79, Price: 7980.185575739132, Net Worth: 94952.05555424007
Step: 80, Price: 8079.746231963457, Net Worth: 95998.32903811886
Step: 81, Price: 8180.46045710648, Net Worth: 97193.05691281075
Step: 82, Price: 8283.416245195964, Net Worth: 98401.62761452702
Step: 83, Price: 8382.379704759416, Net Worth: 99637.09707160083
Step: 84, Price: 8480.762717553629, Net Worth: 100824.65858636226
Step: 85, Price: 8579.75920346646, Net Worth: 102005.25473989281
Step: 86, Price: 8681.590007701863, Net Worth: 103193.21257084678
Step: 87, Price: 8782.247509921182, Net Worth: 104415.18222167162
Step: 88, Price: 8881.187989513648, Net Worth: 105623.07224830346
Step: 89, Price: 8982.214524379875, Net Worth: 106810.35800341304
Step: 90, Price: 9082.40867947857, Net Worth: 108022.67642180777
Step: 91, Price: 9184.345969459637, Net Worth: 109225.00628299212
Step: 92, Price: 9282.941863271883, Net Worth: 110448.2537627649
Step: 93, Price: 9382.286538978688, Net Worth: 111621.40448851186
Step: 94, Price: 9481.502322672422, Net Worth: 112704.19592128671
Step: 95, Price: 9578.575292776159, Net Worth: 113894.78532561153
Step: 96, Price: 9679.167533330288, Net Worth: 115059.66096685638
Step: 97, Price: 9779.689643874648, Net Worth: 116266.76785350592
Step: 98, Price: 9879.699870787934, Net Worth: 117473.03318003824
Step: 99, Price: 9979.230696521183, Net Worth: 118673.15590299768
```

- **Stock Prices**: The "Price" column represents the simulated stock price at each step. The stock prices are generated as a cumulative sum of random values and fluctuate over time, starting low and increasing gradually.

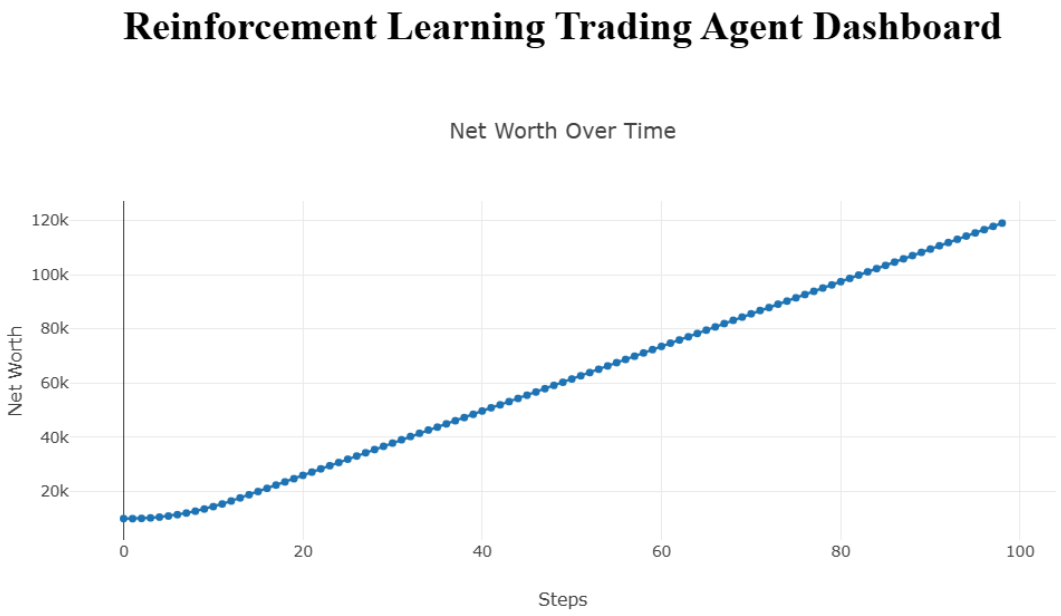
- **Net Worth**: The "Net Worth" column tracks the agent's financial standing at each step. It is calculated as:

Net Worth=Balance+(Number of Shares Held×Current Stock Price)\text{Net Worth} = \text{Balance} + (\text{Number of Shares Held} \times \text{Current Stock Price})Net Worth=Balance+(Number of Shares Held×Current Stock Price)

The net worth grows steadily, indicating that the agent's trading strategy has been profitable.

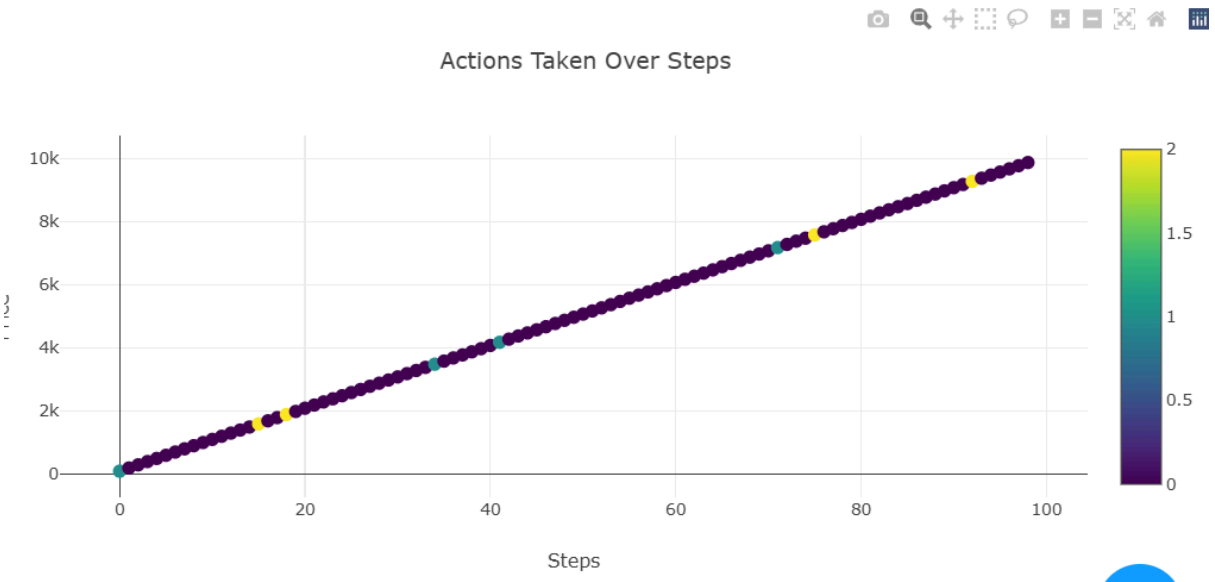
🎬 **Final Results**:

- At **Step 99**:
    - **Price**: 9979.23
    - **Net Worth**: 118673.16 This shows the agent ended with a significant increase in net worth compared to the initial balance of 10,000.



🎬 The steady rise in net worth suggests the TD learning algorithm successfully learned a policy to trade stocks profitably.

🎬 The logs demonstrate that the agent's trades (buy/sell) aligned well with price trends, effectively leveraging opportunities.



Initial Balance	Final Balance	Final Net Worth	Total Shares
10000	450.58861456035265	119006.98706401557	12

1. **Impact of Discount Factor (γ) on Performance**:
    - The discount factor, γ\gammaγ, determines how much importance the agent gives to future rewards.
        - **Higher γ\gammaγ Values**: The agent places greater emphasis on long-term rewards. This can result in more strategic, forward-looking decisions but may delay immediate rewards. In a stock trading scenario, the agent would aim to maximize the overall portfolio growth over time rather than short-term gains.
        - **Lower γ\gammaγ Values**: The agent prioritizes immediate rewards, leading to potentially short-sighted decisions. This might result in more frequent trades aimed at quick profits, potentially increasing transaction costs and risks.
    - γ\gammaγ thus needs to balance between capturing future rewards and responding effectively to short-term opportunities.
    - In stock trading, this could mean the agent moves from erratic buying/selling to well-timed actions aligned with market trends and price fluctuations.

Effect of Discount Factor γ\gammaγ on my Agent:

        - A **higher γ\gammaγ** will make the agent consider the net worth changes over the long term (future steps). For example, the agent might hold stocks longer to benefit from a predicted price increase, even if it means forgoing short-term rewards.
        - A **lower γ\gammaγ** might lead the agent to prioritize immediate improvements in net worth. This could result in frequent buying and selling, increasing transaction costs.
    - In my case, with γ=0.99\gamma = 0.99γ=0.99, the agent is forward-looking, trying to optimize net worth over the long term, which is suitable for the stock

Task 2

This dataset contains medical data that is commonly used in heart disease diagnosis. Each row in the dataset represents a patient, and each column contains specific attributes related to the patient's health and medical history. Below is the description of each feature:

Features Description:

1. **age**:
    - The age of the patient (in years).
2. **sex**:
    - The gender of the patient, encoded as:
        - 1 = Male
        - 0 = Female
3. **cp (Chest Pain Type)**:
    - Type of chest pain experienced by the patient, encoded as:
        - 0 = Typical angina
        - 1 = Atypical angina
        - 2 = Non-anginal pain
        - 3 = Asymptomatic
4. **trestbps (Resting Blood Pressure)**:

- Resting blood pressure (in mm Hg) at the time of admission to the hospital.

5. **chol (Serum Cholesterol)**:
    - Serum cholesterol level (in mg/dL).
6. **fbs (Fasting Blood Sugar)**:
    - Fasting blood sugar level (> 120 mg/dL), encoded as:
        - 1 = True
        - 0 = False
7. **restecg (Resting Electrocardiographic Results)**:
    - Resting electrocardiographic measurement, encoded as:
        - 0 = Normal
        - 1 = Having ST-T wave abnormality (e.g., T wave inversions, ST elevation or depression > 0.05 mV)
        - 2 = Showing probable or definite left ventricular hypertrophy
8. **thalach (Maximum Heart Rate Achieved)**:
    - Maximum heart rate achieved during exercise.
9. **exang (Exercise-Induced Angina)**:
    - Angina induced by exercise, encoded as:
        - 1 = Yes
        - 0 = No
10. **oldpeak**:
    - ST depression induced by exercise relative to rest (a measure of stress on the heart).
11. **slope (Slope of the Peak Exercise ST Segment)**:
    - Slope of the ST segment during the peak exercise, encoded as:
        - 0 = Upsloping

        - 1 = Flat
        - 2 = Downsloping

12. **ca (Number of Major Vessels Colored by Fluoroscopy)**:
    - Number of major blood vessels (0–4) colored by fluoroscopy (a technique used for imaging blood flow in arteries).
13. **thal (Thalassemia)**:
    - Type of thalassemia (a blood disorder), encoded as:
        - 0 = Normal
        - 1 = Fixed defect
        - 2 = Reversible defect
14. **target (Diagnosis of Heart Disease)**:
    - The target label indicating whether the patient has heart disease:
        - 1 = Presence of heart disease
        - 0 = Absence of heart disease

Agent Definition (TDHealthcareAgent)

- **Temporal Difference Learning**:
    - Combines exploration (random actions) and exploitation (choosing the best-known action).
    - **Q-Table**: A lookup table where state-action pairs are stored along with their corresponding Q-values (expected long-term rewards).
    - **Parameters**:
        - alpha (Learning Rate): Controls how much new information overrides old.
        - gamma (Discount Factor): Determines the importance of future rewards.

        - epsilon (Exploration Rate): Balances exploration vs. exploitation.

- **Training**:
    - The agent interacts with the environment for multiple episodes, learns optimal actions, and updates the Q-Table using the **Bellman Equation**.
- **Testing**:
    - The agent uses the learned Q-Table to make decisions and demonstrates its strategy.

Visualization (Dash Application)

- A **Dash GUI** visualizes the RL agent's learning process and treatment strategies:
    - **Reward Graph**: Shows rewards obtained over time (steps).
    - **Action Scatter Plot**: Displays the actions taken by the agent over time.
    - **Summary Table**: Tabular view of the step, action, and reward data for easy interpretation.

Interpretation of the Code

Purpose:

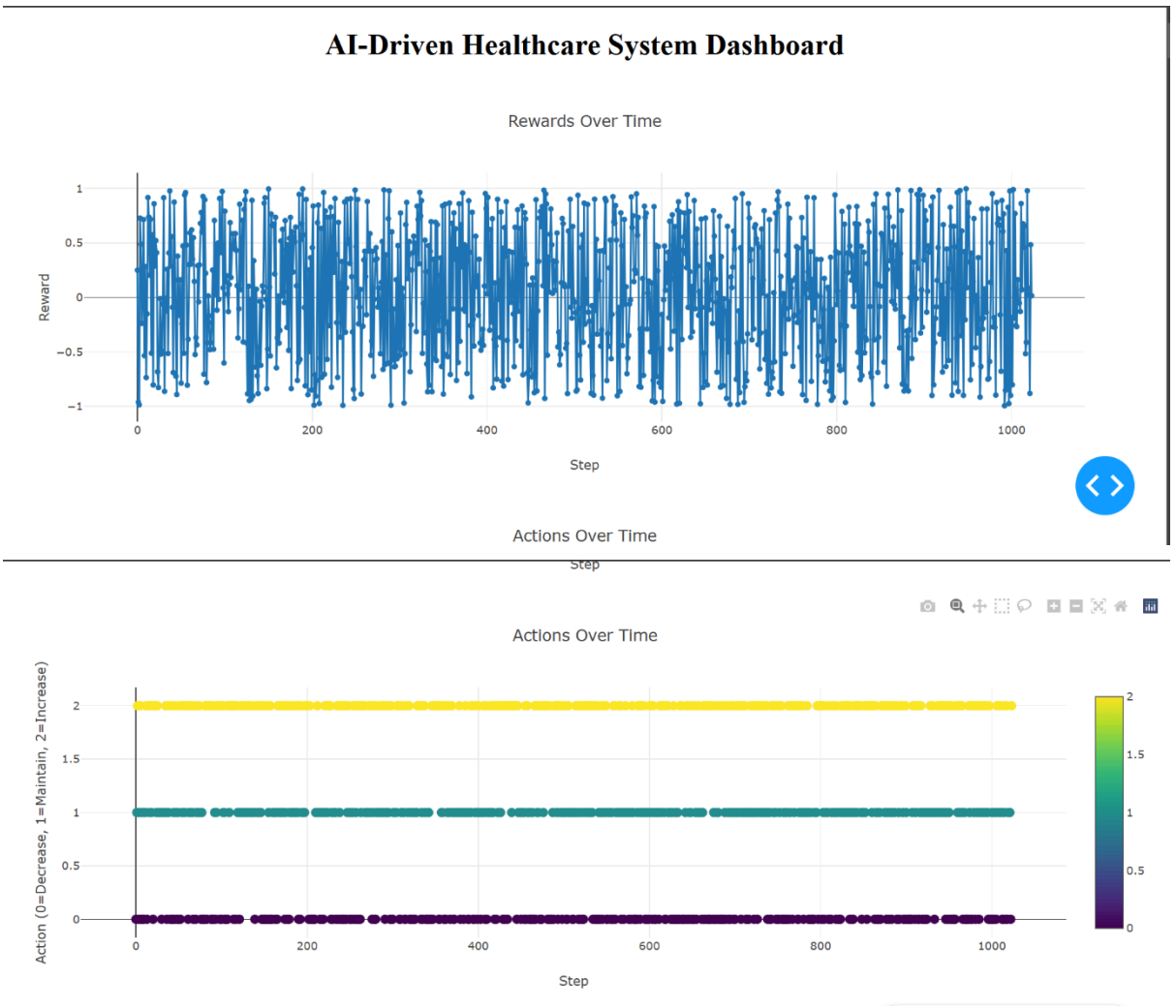
The code simulates how an AI agent can suggest optimal dosage adjustments for patients based on medical data.

Agent's Learning Process:

1. **Exploration Phase**:
    - During training, the agent explores different actions (due to epsilon) and observes their effects on the patient's reward.
2. **Exploitation Phase**:
    - Over time, the agent starts favoring actions with higher Q-values, which are indicative of better long-term rewards.

Agent's Strategy:

- The agent learns to identify patterns in patient data that result in higher rewards.




```
 2.          1.          ], Reward: 0.9474367401428547
Step: 852, State: [0.47916667 1.         0.         0.32075472 0.29452055 0.
 1.         0.6870229  1.         0.         2.         0.25
 3.         0.         ], Reward: 0.30461750580128943
Step: 853, State: [0.79166667 1.         0.         0.24528302 0.23515982 0.
 0.         0.44274809 1.         0.41935484 1.         0.5
 3.         0.         ], Reward: -0.08689798462599296
Step: 854, State: [0.75       1.         3.         0.41509434 0.35616438 1.
 0.         0.78625954 0.         0.22580645 1.         0.25
 2.         0.         ], Reward: 0.4888873277387764
Step: 855, State: [0.35416667 1.         1.         0.06603774 0.16210046 1.
 1.         0.64885496 0.         0.         2.         0.
 3.         1.         ], Reward: -0.9599591002360308
Step: 856, State: [0.8125     0.         2.         0.24528302 0.19406393 0.
 0.         0.33587786 0.         0.24193548 1.         0.
 2.         1.         ], Reward: -0.80925797833638
Step: 857, State: [0.29166667 1.         0.         0.19811321 0.40410959 0.
 1.         0.83969466 0.         0.19354839 1.         0.
```

1. State Transitions:

- The State values change after every step, reflecting the agent moving to the next patient in the dataset.

- The agent uses these state transitions to learn how specific features (like cholesterol, heart rate, etc.) influence rewards.

2. Rewards:

- Positive rewards (e.g., 0.4888873277387764) suggest that the agent's chosen action (e.g., adjusting dosage) likely benefited the patient.
- Negative rewards (e.g., -0.9599591002360308) suggest that the chosen action had adverse effects.
- These rewards guide the agent in learning an optimal treatment policy.