PROJECT 1 - REINFORCEMENT LEARNING

B(E)4M36SMU

21/03/2024

CTU
Arnau Garcia Parise

Index

1. (3 points)

Propose three possible nontrivial reasonable ways how to define the state in the game.

$S_1 = \{(p, d) \mid p \in P, d \in D\}$

Where:

- *p* represents the total value of the player's hand.
- *d* represents the value of the dealer's showing card.
- *P* is the set of possible total values for the player's hand.
  P= {2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21}
- *D* is the set of possible values for the dealer's showing card.
  D= {1,2,3,4,5,6,7,8,9,10,11}

$S_2 = \{(p, d, a) \mid p \in P, d \in D, a \in \{0..4\}\}$

Where:

- *p* represents the total value of the player's hand.
- *d* represents the value of the dealer's showing card.
- *a* represents how many Aces has the Player (max is 4)
- *P* is the set of possible total values for the player's hand.
  P= {2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21}
- *D* is the set of possible values for the dealer's showing card.
  D= {1,2,3,4,5,6,7,8,9,10,11}

$S_3 = \{(p, d, a) \mid p \in P, d \in D, a \in \{0..4\}, n \in \{2..11\}\}$

Where:

- *p* represents the total value of the player's hand.
- *d* represents the value of the dealer's showing card.
- *a* represents how many Aces has the Player (max is 4 because there is only 4 Aces)

- *n* represents number of cards in player's hand (max is 11 because the sum of the minimum values are 1+1+1+1+2+2+2+2+3+3+3 = 21)
- *P* is the set of possible total values for the player's hand.
  P= {2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21}
- *D* is the set of possible values for the dealer's showing card.
  D= {1,2,3,4,5,6,7,8,9,10,11}

2. (3 points)

For each state-space representation from 1, provide a rough estimate of the overall number of states.
You cannot just guess a number; you have to justify it somehow (e.g., by calculation).
You do not need to provide an exact number; however, your estimate should not be too far from the true count. If you are not sure how to calculate the number of states, you can write a program that counts them for you and submit the code together with your report.

$$S_1 = \{(p, d) \mid p \in P, d \in D\}$$

For the player, there are |P| = 20 possible values (P = {2..21}) and for the dealer there are |D| = 11 (D = {1..11}). The overall number of states is 20 * 11 = **220**.

$$S_2 = \{(p, d, a) \mid p \in P, d \in D, a \in \{0..4\}\}$$

For the player, there are |P| = 20 possible values, for the dealer there are |D| = 11 and for the aces, you can have 0 or a maximum of 4 aces in the hand. The overall number of states is 20 * 11 * 5 = **1100**.

$$S_3 = \{(p, d, a) \mid p \in P, d \in D, a \in \{0..4\}, n \in \{2..11\}\}$$

For the player, there are |P| = 20 possible values, for the dealer there are |D| = 11 and the total number of cards is |n| = 10 (n = {2..11}). The overall number of states is 20 * 11 * 10 = **2200**.

2. (3 points)

Pick one of the state space representations you proposed in 1. Use this representation from now on. Please, explain why you consider it the best one and answer the following questions. Does this representation capture all information that can be used for agent decisions? Or is there any simplification? If yes, will the simplification influence the result (final policy, utility values)? If yes, how much will the result be influenced? Can you use exact methods (value iteration/policy iteration) to solve the game? If yes, how? If not, why?
Briefly explain your choice of discount factor and the number of games that you need to learn the Q-values. Hint you might or might not use: calculate the expected utility in an initial state and compare it with the goal you want to achieve after learning.

State S1 includes the total value of the player's hand and the value of the dealer's showing card. These are critical pieces of information needed for decision-making in blackjack. The player needs to know their hand value relative to the dealer's showing card to make informed decisions.

With only two components (player's hand value and dealer's showing card value), the state space remains relatively small, making it computationally feasible to analyze and learn strategies efficiently.

The simplification may affect the precision of the learned strategies but is unlikely to significantly impact the overall effectiveness. The basic strategy in blackjack is primarily based on the player's hand value and the dealer's showing card, both of which are captured in S1.

Exact methods like value iteration or policy iteration can be applied to solve the game with state representation S1. Since the state space is manageable, these methods can efficiently compute the optimal policy or value function.

A common choice for the discount factor in blackjack might be a high value close to 1 since the game doesn't have a natural termination point until the player decides to stop playing. The number of games needed would depend on the chosen learning algorithm, exploration strategy, and convergence criteria.

6. (7 points)

Test your code and provide an experimental evaluation. Compare the random strategy (provided), the dealer strategy (provided), the result from 4 and the strategy learned by SARSA.
In this question, you should present why your implementation gives valid results, learns, and is well tested.

For example, you may answer the following questions. What is the agent's

expected or average utility? How fast do algorithms implemented in 4 and 5 learn? Does the learned utility contradict your intuition? What is the utility for drawing a card when you have club nine, diamond jack and spades two in your hand, and dealer has club four? What
is the utility of the situation when you have diamond ace and spades five and dealer spades ace? Is it better to draw a card in this situation or not? Did your utility values estimate/ˆQ values converge? Did they converge to the true state values, i.e., U and Q? Does the strategy learned by SARSA follow the recommendation in the section Blackjack strategy of [1]?

In this experimental evaluation, we ran 100,000 episodes for each agent and calculated both the average utility and the win rate for each strategy in the game of blackjack. Here are the results:

1. **Random Agent:**
   - Average Utility: -0.39631
   - Win Rate: 28.1%
2. **Dealer Agent:**
   - Average Utility: -0.08243
   - Win Rate: 40.762%
3. **TD Agent:**
   - Average Utility: -0.07648
   - Win Rate: 41.079%
4. **SARSA Agent:**
   - Average Utility: -0.10848
   - Win Rate: 40.328%

Random Agent: With a negative average utility and a low win rate of 28.1%, the Random Agent performs the worst among all strategies. Random actions are ineffective in achieving favorable outcomes in blackjack.

Dealer Agent: While outperforming the Random Agent, the Dealer Agent still demonstrates suboptimal performance with a slightly negative average utility and a win rate of 40.762%. Its heuristic strategy, mirroring the dealer's actions, falls short of optimal play.

Learning-Based Strategies (TD Agent and SARSA Agent): Both Temporal Difference (TD) learning and SARSA algorithms exhibit improved performance compared to random and heuristic strategies. They achieve similar average utilities and win rates, surpassing the Random Agent and demonstrating slight superiority over the Dealer Agent.

The chosen state representation and feature set may not capture all relevant information for decision-making

Scenario 1: Club nine, diamond jack, and spades two vs. dealer's club four: it's generally recommended to hit against a dealer's low card to improve the hand's potential value without risking busting.

Scenario 2: Diamond ace and spades five vs. dealer's spades ace: Recommends hitting on a hand value of 16 when facing a dealer's high card (10 or ace).