Lab 7: Reinforcement Learning - Value Iteration

T-504: Introduction to Machine Learning

Fall, 2018

1 Introduction

The goal of this lab is to get a better understanding of dynamic programming approaches for finding the optimal policy for a problem, by implementing value iteration for a specific problem.

2 Handing in

Hand in **a report (PDF document)** with the answers to all the questions below including the respective Python code that generated the plot or numbers for your answer, if any.

3 Environment

Consider the game of Blackjack played by a single player against the dealer. The game is played with a standard deck of 52 cards (2 to ace of the 4 suits). Each card has a value. The value of cards 2 to 10 is their pip value, face cards (Jack, Queen, and King) are worth 10 points and an Ace can be worth 1 or 11 points, as the player chooses. A hands' value is the sum of the card values. The goal of the game is to get a hand worth more than the opponent's hand, but not more than 21.

There are several variations on the rules of the game. For the purpose of this assignment, use the following rules:

To start with, the player gets two cards and the dealer gets one card from the deck (all face up). Now, as long as his hand value is below 21, the player can "hit", i.e., get one additional card in his hand. Alternatively, he can "stand" to end his turn. If the hand value of the player is over 21 the dealer wins. Otherwise, the dealer will now play according to this fixed strategy: A dealer will "hit" as long as his hands' value is less than 17 and "stand" otherwise. The dealer always counts aces as 11 points, unless that would bring his hands' value above 21. After the dealers turn, the outcome of the game is determined as follows (in this order):

- If the players' hand has a value above 21, he looses.
- If the dealers' hand has a value above 21, the player wins.
- If player and dealer have hands with equal value the game ends in a draw.
- The player with the higher valued hand wins.

4 Tasks

1. Modeling the environment *(Total points: 50)*

   For this task, assume that the game is played with a deck consisting of infinitely many copies of each of the 52 cards. That is, even after having seen a card, the probability of drawing a specific card is still $\frac{1}{52}$.

   (a) (20 points) How would you describe a state of the problem? What are the possible actions, successor states, rewards and transition probabilities? Try to reduce the state space as much as possible (combine states whose values and best action do not differ), but make sure to keep the Markov property.

(b) (30 points) Implement your model of the problem by filling out the relevant functions in BlackjackMDP in `lab7.py`. Test your model by generating some simulations of random game play and checking whether the resulting states make sense.

How many reachable states does your model have?

2. Value Iteration *(Total points: 50)*

(a) (25 points) Implement Value Iteration for an arbitrary MDP (any object derived from the MDP class in the code). Decide on an appropriate stopping criterion and run your code on the Blackjack MDP.

(b) (10 points) What is the expected outcome of playing Blackjack with the optimal policy? Who will win the the long run, the player or the dealer? Why?

(c) (15 points) Visualize the resulting value function and policy. For example, you could use a table similar to the one on Wikipedia, depending on your choice of state space.

(d) **(5 bonus points)** Suppose the player had an additional action "Double down". Upon "Double down" the player will get exactly one more card and end his turn. He will also double his initial bet, that is, win or loose double the normal amount. Show how having this additional action changes the optimal policy and expected outcome of the game.

(e) **(10 bonus points)** Playing with an infinite deck of cards is somewhat unrealistic. How does the value of the game change, if we assume to only play with one deck of cards (52 cards)? How many different states do you need to consider in this case?