Mission 4

Adam Morris – Computational Social Cognition Bootcamp, July 2017
thatadammorris.com thatadammorris@gmail.com

Your agent is unstoppable, and the casino is desperate. In a last-ditch attempt to salvage their evil plan, they trot out Wyatt, a veteran dealer. He slowly strides up to the blackjack table, squeaks back the chair, stares at you with his one remaining eye, and growls: "It's time to play real blackjack."

Congratulations on completing Mission 3. If you want to compare your code to mine, my solution is available in "solo_blackjack.m" (online here).

In your final mission, you'll simulate your agent playing (almost) real blackjack. The game is similar to before, except now there's a dealer. The dealer starts by dealing themselves two cards, and dealing you two cards. You go through exactly the same decision process as in Part B of the last mission.

But now, the reward conditions are different. You're now trying to "beat the dealer". After you make your decisions, it's the dealer's turn. The dealer reveals their two cards. If the sum is less than 17, the dealer must "hit" and draw an extra card. The dealer repeats this process until their sum is greater than or equal to 17.

You win if and only if your sum is strictly greater than the dealer's sum – without your sum going over 21. (If your sum is over 21, you lose no matter what.) The dealer wins ties. For example, if the dealer's cards sum to 17 and yours sum to 19, you win. If both your cards and the dealer's cards sum to 20, you lose. If the dealer's cards sum to 25 and yours sum to 22, you lose (because your sum is over 21). Make sense?

If the agent wins, it receives a reward of 1; if it loses, it receives a reward of 0.

Part A

Simulate a Q-learning agent playing this game. (Assume for now that the dealer's cards are both face down during the agent's decision process.) The only thing you should have to change is the simulation of the reward function. Plot the agent's final value estimates. (To do this, use `plotResults(Qvalues, 4)`. Again, make sure to label "hit" as action 1, and "stay" as action 2.)

Part B

Now, assume that your agent can see the dealer's first card before making its decisions. How would your agent incorporate this information? (Hint: State spaces can be multi-dimensional.) Simulate an agent that does, and compare its average award to the agent in Part A. Who performs better? Why?

Plot the agent's final value estimates. (Your `Qvalues` matrix should be three-dimensional!) What does the agent do differently based on the dealer's card? In essence, the agent has modeled some kind of social information by incorporating it into the state space. What kind of social-psychological phenomenon could this approach be used to model?

(My code for this mission is in "real_blackjack.m", online here.)