Roan Urquhart
COMP 560

Assignment 2

Questions

1. How did changing the initial value for exploration (starting close to 1 versus starting closer to 0) change the resulting computed policy?

   When the initial value for exploration is set closer to 1, the utility values will vary more each time the learning is done. This is because the program will try other paths instead of just following the best policy found so far. When the initial value is set near zero the utility values will remain more stable as the program will choose to follow the best policy found until that point.

2. How did you decide when to stop learning?

   I decided to stop learning once the utility values reached convergence

3. How did changing the discount value (starting close to 1 versus starting closer to 0) change the resulting policy?

   As you make the discount value closer to 1, each utility value for the policy becomes more and more skewed. Due to the skew, there is a more noticeable difference between the utility of each state, so the chosen policy is the same consistently. As the discount value is set closer to 0, the difference between each utility value decreases. Since the values are closer to each other, the policy is more likely to change.

4. How did changing the value for epsilon change the resulting policy?

   Changing the value for epsilon affects how much of an effect exploration and exploitation have the on the result. If the epsilon value allows for the iterations to continue for a longer period of time the results will be more consistent due to the exploitation strategy. However if the iterations continue for a short time, the exploration strategy has a greater effect on the results, causing them to be less consistent.

Implementation:

Main Execution

The file **proj_execution.py** handles the calling of all necessary function for the program to run.

Model Based

The file **model_based.py** contains the implementation of the model-based reinforcement learning. The most important function in this file is **model_based_rl()** which handles calculation as well as the necessary function calls to complete the learning process.

Model Free

The file **model_free.py** contains the implementation of the model-free reinforcement learning. The most important function in this file is **model_free_rl()** which handles calculation as well as the necessary function calls to complete the learning process.

Utility

The file **utility.py** contains the implementation for input parsing, printing, and a variety of helper functions. The most important helper function is **calc_utility(state, discount, reward, explore_val)**. This function does the necessary calculation for one iteration of utility value calculation. This function is called until the values converge in both reinforcement learning functions. The comments within the code can give better insight into exactly how this function works.

Contribution:

I, Roan Urquhart, worked alone on this project. Therefore all of the code is written by me.