Hongyu Li
Loi Pham
Philip Kim

Assignment 2: Reinforment Learning

In this assigment, we use two different approaches to find the optimal policy for playing golf. In order to obtain the optimal policy, we need to calculate the utilities for each action at each state. The two approaches we used are Model-Based Learning and Model-Free Learning.

Model-Based Learning:

In Model-Based learning, we need to calculate the probabilities for each starting state, action, and resulting state triplet and with the probability transition table, we can calculate the utilities with bellman equation.

Given an environment, an input file, we can extract the starting states, actions, and resulting states, and the probabilities of each triplet. As confirmed by Dr. Palay and TA, we know the starting states, actions, and the possible resulting states given a starting state and an action taken, so this program assumes we do know the resulting actions. The only thing we do not know is the probabilities.

In the program, we extract a 1-d array of all starting states, a 2-d array of all actions taken at each state, and a 3-d array of all possible resulting states from each state and action, and another 3-d array of probabilities of each triplet. With these arrays, we can easily use *choices* function, given all possible resulting states and corresponding probabilities, to get a resulting state. By doing this many times (5000, in the program), we can get all possible results with how many times of each and calculate the probability for each resulting state given a starting state and an action.

In the second part, we can calculate utility table from the probability table. For each starting state and an action, there is a corresponding utility value. We can first build a utility table same size as the action table, with all values 0 and after the first iteration, all values will be one except for “In”. Then we can use probability table and bellman equation to calculate further utility and by doing this multiple times (5000), we can get a utility table.

With the utility table, we can decide the optimal policy. For each state, we can choose the action with minimum utility as the best choice for this state. Therefore, we have an optimal policy.

Model-Free Learning:

In Model-Free Learning, we don’t need to calculate the probabilities for each resulting state. We can calculate the q-value given the current state and the resulting state.

Given the environment, we have the same arrays as above. This time, we can define a q-table, same size as 2-d actions, with all values set to zero. We can start from “Fairway”, and get the resulting state from there, and start from the resulting state. Every time, we have a starting state and a resulting state. Based on the q-value of the starting state and q-value of the resulting state (minimum q-value), we can calculate a new q-value for the starting state (current state) with a learning rate (0.1) and a discount (0.9) and replace the old value in q-table with the new value. We do this process many times and we get a-q table with updated values. Each q-value is the utility value for each state.

In this way, we can determine the optimal policy by choosing each state and the action with minimum utility.

Exploration and Exploitation:

We used both exploration and exploitation in both approaches when we choose the next action. When doing exploration, we choose a random action, for example, in model-based learning, when choosing an action to calculate the utility values of a resulting state, we would choose the utility value of a random action. In model-free learning, when choosing an action to calculate the q value of the resulting state, we would choose a random action. When doing exploitation, we choose an action with minimum utility value for each state, for example in model-based learning, when choosing an action to calculate the utility values of a resulting state, we would choose the action with minimum utility, same in model-free learning. In this way, we can make sure we encounter all possible actions and choose the best one in later iterations.

In order to achieve that, we can use an epsilon value, which is between 0 and 1. If it is closer to 1, then we do more exploration and if it is closer to 0, we do more exploitation. Every time we choose an action, we first generate a random value between 0 and 1, if it's greater than epsilon, we choose the minimum utility action, otherwise, we choose a random utility. In the program, we choose to make epsilon decay after the first half of iterations, epsilon gradually decays as more and more iterations so that the program is more and more likely to do exploitation, till epsilon value is 1. In the program, we set the initial epsilon value to 0.5.

Analysis:

Changing initial value for exploration, which is changing epsilon, would affect how fast we can get a usable utility table. Greater epsilon values tend to do more exploitation, so that it is more likely to choose "best action" at each state, which would lead to faster results, but with less exploration. We may not be able to explore all possible actions at each state, leading to inaccurate utility values. Similarly, smaller epsilon values tends to do more exploration, which could lead to longer process but accurate results.

We used 5000 iterations in both model-based learning and model-free learning. It was not easy to decide when to stop the program. We can usually stop learning when the utility values are relatively stable, so we can decide which one is the "best action".

Changing discount values would not affect the optimal policy but will largely affect utility values. Greater discount values would lead to greater utility values, as we put more weight on the utility values of the resulting states.

Contributions:

Hongyu Li: wrote the model-based learning; wrote and assisted with model-free learning.

Loi Pham: wrote the parser for model-based and model-free learning; assisted with model-free learning.

Philip Kim: debugged the parser to make it work with model-free and model-based learning; assisted with model-free learning.