Project 3: Better, Smarter, Faster

16:198:520

Consider the same environment as in Project 2 - a circle of 50 nodes, with random edges. Again we have the predator, the agent, and the prey. The prey transitions as before, choosing a neighbor at random each time it moves. The predator transitions in accordance with the Distracted Predator Model from the previous project. In this project, we want to build an agent that captures the prey as efficiently as possible.

> How many distinct states (configurations of predator, agent, prey) are possible in this environment?

For a given state s, let $U^*(s)$ be the *minimal expected number of rounds to catch the prey* for an optimal agent, assuming movement order as in Project 2.

- What states s are easy to determine U^* for?
- How does $U^*(s)$ relate to U^* of other states, and the actions the agent can take?

Write a program to determine U^* for every state s, and for each state what action the algorithm should take. Describe your algorithm in detail.

- Are there any starting states for which the agent will not be able to capture the prey? What causes this failure?
- Find the state with the largest possible finite value of U^*, and give a visualization of it.
- Simulate the performance of an agent based on U^*, and compare its performance (in terms of steps to capture the prey) to Agent 1 and Agent 2 in Project 2. How do they compare?
- Are there states where the U^* agent and Agent 1 make different choices? The U^* agent and Agent 2? Visualize such a state, if one exists, and explain why the U^* agent makes its choice.

Build a model to predict the value of $U^*(s)$ from the state s. Call this model V.

- How do you represent the states s as input for your model? What kind of features might be relevant?
- What kind of model are you taking V to be? How do you train it?
- Is overfitting an issue here?
- How accurate is V?

Once you have a model V, simulate an agent based on the values V instead of the values U^*.

> How does its performance stack against the U^* agent?

In Project 2, we were also interested in the partial information situation, where we may not know exactly where the prey or predator is. Consider the unknown prey position case - here, the state of the agent may be represented by the position of the agent, the position of the predator, and a vector of probabilities $\underline{p}$. Because there are infinite possible belief states, the optimal utility is hard to solve for. In this case, we might estimate the utility of a given state as the expected utility based on where the prey might be.

$$U_{\text{partial}}(s_{\text{agent}}, s_{\text{predator}}, \underline{p}) = \sum_{s_{\text{prey}}} p_{s_{\text{prey}}} U^*(s_{\text{agent}}, s_{\text{predator}}, s_{\text{prey}}). \tag{1}$$

Simulate an agent based on U_{partial}, in the partial prey info environment case from Project 2, using the values of U^* from above. How does it compare to Agent 3 and 4 from Project 2? Do you think this partial information agent is optimal?

Build a model V_{partial} to predict the value U_{partial} for these partial information states. Use as the training data states (including belief states) that actually occur during simulations of the U_{partial} agent.

- How do you represent the states $s_{\text{agent}}, s_{\text{predator}}, \underline{p}$ as input for your model? What kind of features might be relevant?
- What kind of model are you taking V_{partial} to be? How do you train it?
- Is overfitting an issue here? What can you do about it?
- How accurate is V_{partial}? How can you judge this?
- Is V_{partial} more or less accurate than simply substituting V into equation (1)?
- Simulate an agent based on V_{partial}. How does it stack against the U_{partial} agent?

Bonus:

- **Bonus 1:** Build a model to predict the actual utility of the U_{partial} agent. Where are you getting your data, and what kind of model are you using? How accurate is your model, and how does its predictions compare to the estimated U_{partial} values?
- **Bonus 2:** Build an optimal partial information agent.