Report
Udacity AIND Project 3

CustomAgent search function uses an advanced search technique

For this project, I chose option 3. I implemented Monte Carlo tree search, an advanced technique not covered in lecture.

Table documenting the performance of the agent

python run_match.py -f -r 100 -p 4 -a ${AGENT} -o ${TEST_ALGO} -t 150

	Algorithm	
Opponent Agent	Baseline (Minimax)	Custom (Monte Carlo Tree Search)
Random	95.0%	91.0%
Greedy	66.5%	80.0%
Minimax	100.0%	75.5%
Average Score	87.2%	82.2

Baseline

I choose as a baseline Minimax algorithm.

Decide whether or not fair_matches flag should be used

I decided to set the fair_matches flag to true. While none of the evaluated algorithms makes use of an opening book, I think it’s still worth it to make the agents play additional games by switching the opening move, to be able to better quantify the effect of the first move, and therefore have a more reliable overall evaluation.

Answer the required questions

Choose a baseline search algorithm for comparison

I choose Minimax as the baseline search algorithm for comparison.

How much performance difference does your agent show compared to the baseline

It depends on the algorithm used by the opposing agent, but overall the Minimax baseline outperforms MCTS.
On average, including all the test agent algorithms, Minimax(87.2%) is about 5% better than MCTS(82.2%)
Also, one noticeable result is that Minimax wins 100% against another Minimax agent (I don’t have an explanation for this, though).

Why do you think the technique you chose was more (or less) effective than the baseline

I further run other simulations by increasing the time limit, to understand what effects it may have on the performance (See table below).

The observation is that increasing the time limit causes the performance of MCTS to improve, and we see that the trend is reversing: MCTS beats Minimax against a Random and a Greedy agent, but still loses against a Minimax agent.

```
python run_match.py -f -r 100 -p 4 -a ${AGENT} -o ${TEST_ALGO} -t 750
```

	Algorithm	
Opponent Agent	Baseline (Minimax)	Custom (Monte Carlo Tree Search)
Random	92.8%	96.0%
Greedy	71.2%	82.8%
Minimax	100.0%	83.5%
Average Score	88.0%	87.4%