Why?

What did we do?

The Game Engine (Environment):

We created the game engine (environment) of blackjack. The game involves two players, the dealer, and the agent. We set parameters of the game similar to ones you would find in a casino. At the beginning of each turn, the agent gets two cards facing up, and the dealer gets one card facing up and one card facing down. The agent moves first. It has four options, draw, stand, double, and surrender. If his total number is greater than 21, the game is over, the agent loses. If not, the dealer will not draw until the agent chooses to stand. After the agent decides to stand, the dealer draws based on the following rules. If his total is smaller than 17, he must draw. If the dealer’s total is greater or equal to 17, he cannot draw. And they compare their sum to decide who’s the winner.

The NN Model:

We used a reinforcement learning model. The model has one input layer, three hidden layers, and one output layer.

Input layer: The input layer has 34 input nodes.

The first ten nodes represent the current hand the agent is holding. The first node reports how many A’s it has, the second node represents how many two’s the agent has, etc. For the next ten nodes, it represents the card that the dealer is holding (note: the sum of these ten nodes should be 1 since the agent can only see one card from the agent.) So whichever facing-upward card the dealer has, that specific node would be one. For example: if the agent has a seven and an eight, and the dealer is showing an ace. For the first twenty nodes, Node #7, #8, and #11 would be one. And if the agent has two seven’s, and one ace, and the dealer is showing one ace as well, for the first twenty nodes, Node #1 would be 1, #7 would be 2, #11 would be 1.

In the middle, it has three hidden layers, each with **xxx** amount of nodes, given by this function. The output layer has t

What’s the result?

What does it mean?