The strategy of MyAgent():

It maintains the opponent's last ten actions to identify any patterns in its moves and use the past actions to predict the opponent's next move if such pattern is found. The prediction is based on the frequency of sequences in the history. It calculates the most common next action following the last nine moves and assumes the opponent will play that action next. Then MyAgent plays the counter action against the prediction according to the rules of Rock-Paper-Scissors. If no patterns are found or the history is not long enough (less than 10 past actions), MyAgent will play randomly.

The strategy of MyAgent is a variant of opponent modeling. Before I implemented this method of opponent pattern learning, I tried fictious play algorithm. With fictious play, the agent lost more than half of the games playing against all 43 opponents. The reasons could be the number of episodes played is relatively small, and the opponents are playing much more complex algorithms which leads to the naïve view of fictious play. Hence instead of the algorithm learns a little from each step, I decided to find out the opponent's pattern first and playing against the pattern to defeat it.

The hyperparameter of the number of last actions to track, 10, is the best trade-off I can find between exploration and exploitation. With this number, it can capture the complex patterns of opponents and it can avoid overfitting since longer patters are less likely to repeat often. This number also leads to quicker computational speed and less memory storage.