pi - agent - report.pdf :

Question 4 (1 point): As in Question 2, this time test your Policy Iteration Agent against each of the provided agents 50 times and report on the results – how many games they won, lost & drew. The other agents are: *random*, *aggressive*, *defensive*.

Against Defensive Agent:

Wins: 44 Draws: 6 Losses: 0

Against Aggressive Agent:

Wins: 50 Draws: 0 Losses: 0

Against Random Agent:

Wins: 50 Draws: 0 Losses: 0

initRandomPolicy :

This method initialises a random policy for the agent. It loops through all valid game states (where it's 'X's turn) and assigns a randomly chosen valid move to each state. Terminal states are skipped as no further actions are possible in those states.

evaluatePolicy(double delta) :

This method performs the policy evaluation step, updating the value of each state under the current policy.

- For each state, the algorithm calculates the expected value of following the current policy by summing over all possible outcomes of the selected move.
- The process repeats until the maximum change in state values (`maxChange`) is smaller than the given threshold `delta`, indicating that the values have converged.

improvePolicy :

This method performs the policy improvement step, attempting to find a better action for each state based on the current value function.

- For each state, the method computes the expected value of all possible actions and selects the one with the highest value.
- If this new best action is different from the current policy's action for that state, the policy is updated, and the policy is marked as unstable (`policyStable = false`).
- The method returns `true` if no changes were made to the policy, indicating it has converged to the optimal policy.

train :

This method alternates between policy evaluation and policy improvement until the policy stabilises:

1. Calls `evaluatePolicy()` to compute the values of states under the current policy.
2. Calls `improvePolicy()` to update the policy based on these values.
3. Repeats the process until the policy no longer changes (policy is stable). Finally, the method initialises the agent's `policy` field with the trained policy (`curPolicy`).