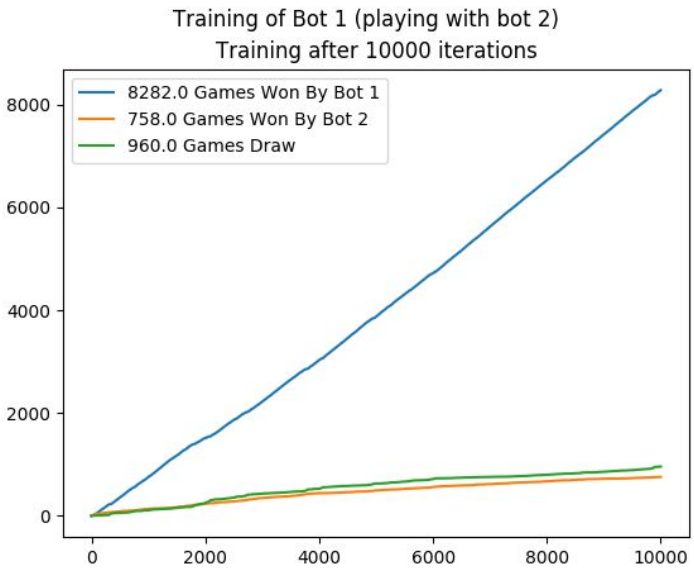
Develop an agent for playing tic-tac-toe game using Reinforcement Learning

An agent (agent 1) is taught by training with another agent (agent 2). In the training, agent 1 plays the chess game with agent 2 and reward is allocated at the end of the game. This training repeats for the no of iterations. When an agent wins, it is rewarded with +1 reward, -1 on a loss and 0 on a draw. The state-action value is updated after the game ends and reward is allotted. All the states are updated in the sequence with a decaying reward (with a decayRate=0.9). The agent follows an ε greedy (ε =0.3, can be changed in the code) approach for choosing an action, balancing the exploration and exploitation part of choosing an action to learn to win the game.
After training the bot (agent 1) is ready to play with a human and it also learns after each game.
For variation in epsilon (iterations=10000) we see

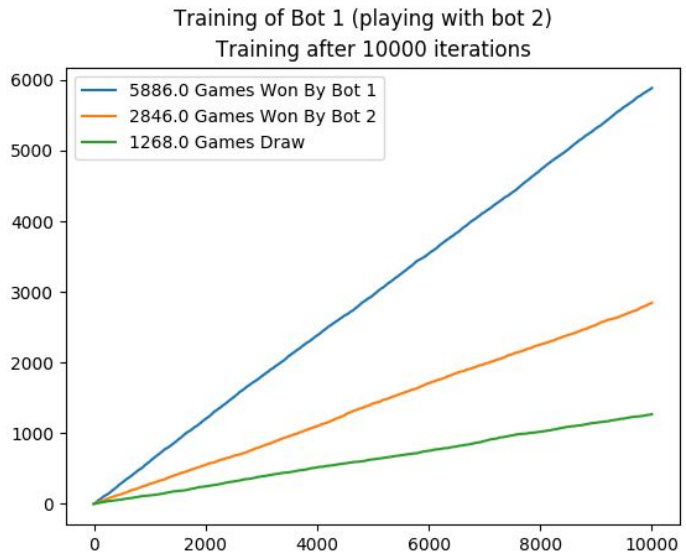
- ε = 0.1

It focuses more on exploitation and very little on exploration and hence it gets only introduced to a handful of optimal learning experience. Hence it is able to win more in training but in a match with a human, performs significantly bad.



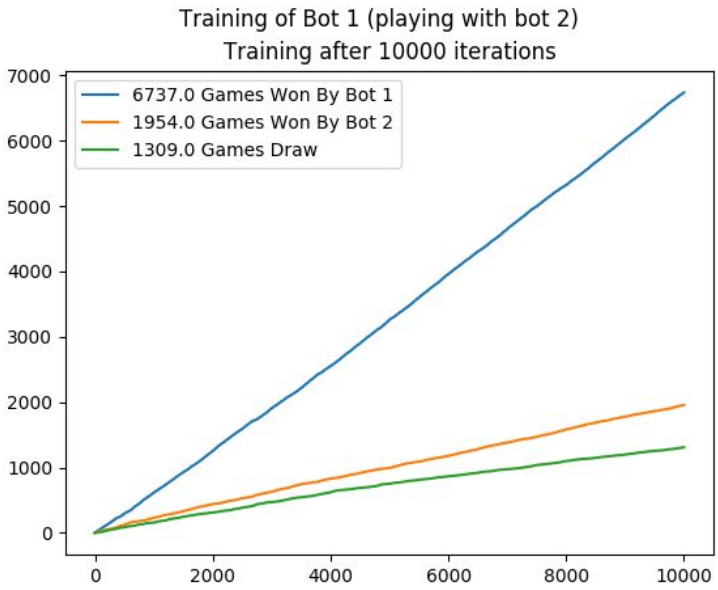
- ε = 0.8 ,

It focuses more on exploration than the exploitation and the agent may have discovered a lot of actions for a state during training . Hence it is not able to win as much in training but in a match with a human, performs significantly good.



- ε = 0.4

It balances exploration and exploitation appropriately and plays smartly against a human.



The effect of iterations on learning is pretty straightforward, more no of iterations means more learning and with right value of epsilon, better learning too.