CSIS BITS Pilani , K.K. Birla Goa Campus

Artificial Intelligence (CS 407)

Programming Assignment 2 (Report)

Name:- Vishal Vivek Bharambe

Roll No. 2019A7PS0160G

Submission Date:- 4th December 2021

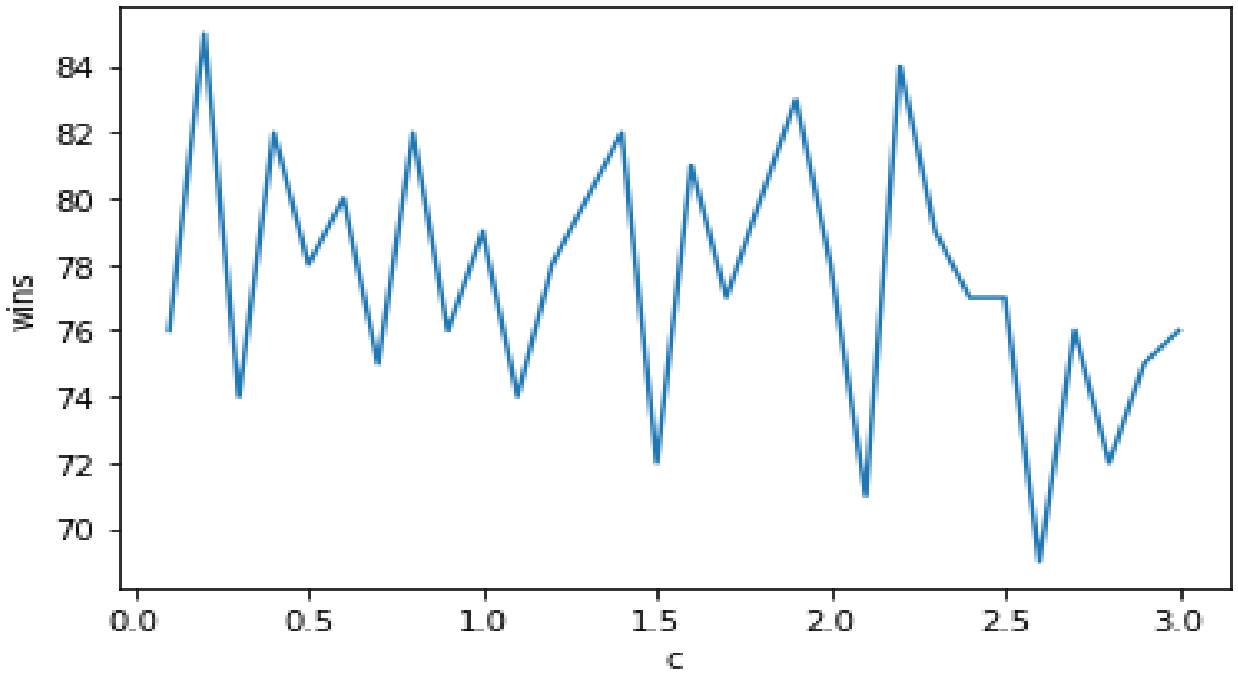
BITS Pilani

K K Birla Goa Campus

1. **MCTS VS MCTS**

   a. **MCTS200vMCTS40**

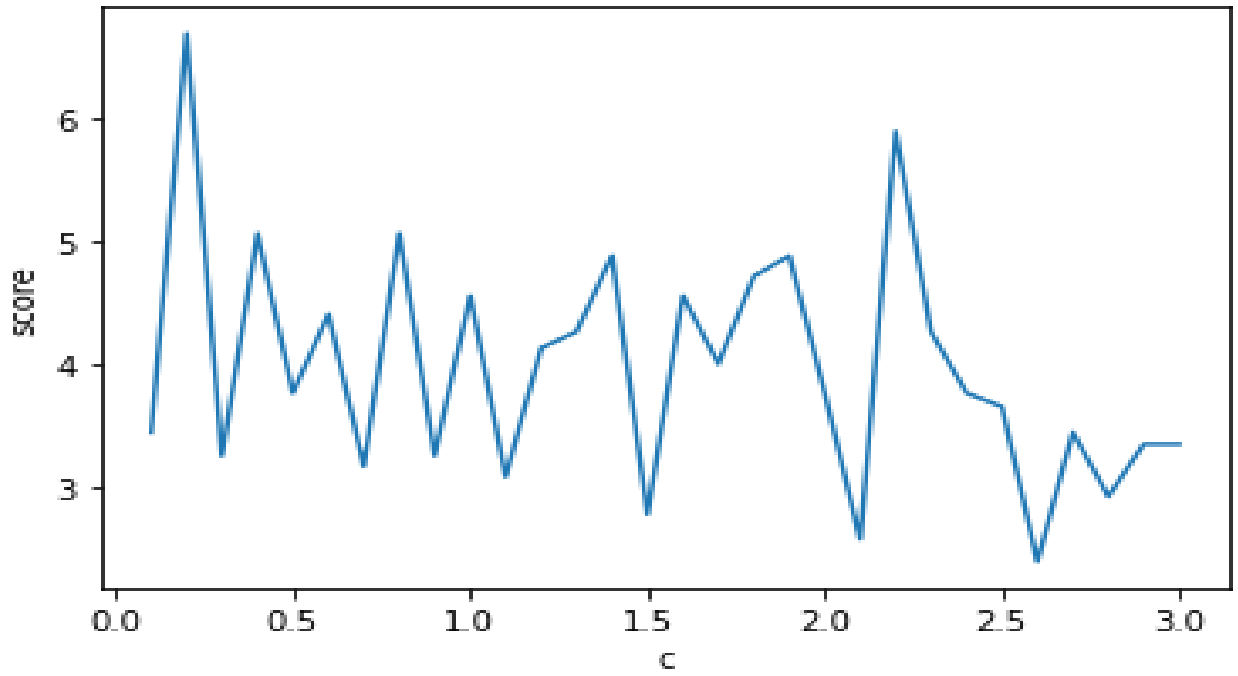
Below is the graph for wins for MCTS200 in MCTS 200 vs MCTS 40 for different values of c



And the graph for

$$score = \frac{wins + \frac{draws}{2}}{losses + \frac{draws}{2}}$$

For MCTS200 in MCTS200 vs MCTS40 for different value of c



The Whole idea of increasing c is to increase the exploration. Where MCTS200 should explore more, and as can be seen that except of one anomaly at c = 0.2 , there are a lot of spikes near the value of 2 .(remember that the formula I have used in my code is where I take square root of c). hence with the slides in the slides,it is supposed to be c= root(2) or somewhere near that

Though it can be seen that graph is very “spikey” and hence it is hard to comment directly but by trying 100 games with the value of c = root(2),Where RED is the MCTS200 Players ,

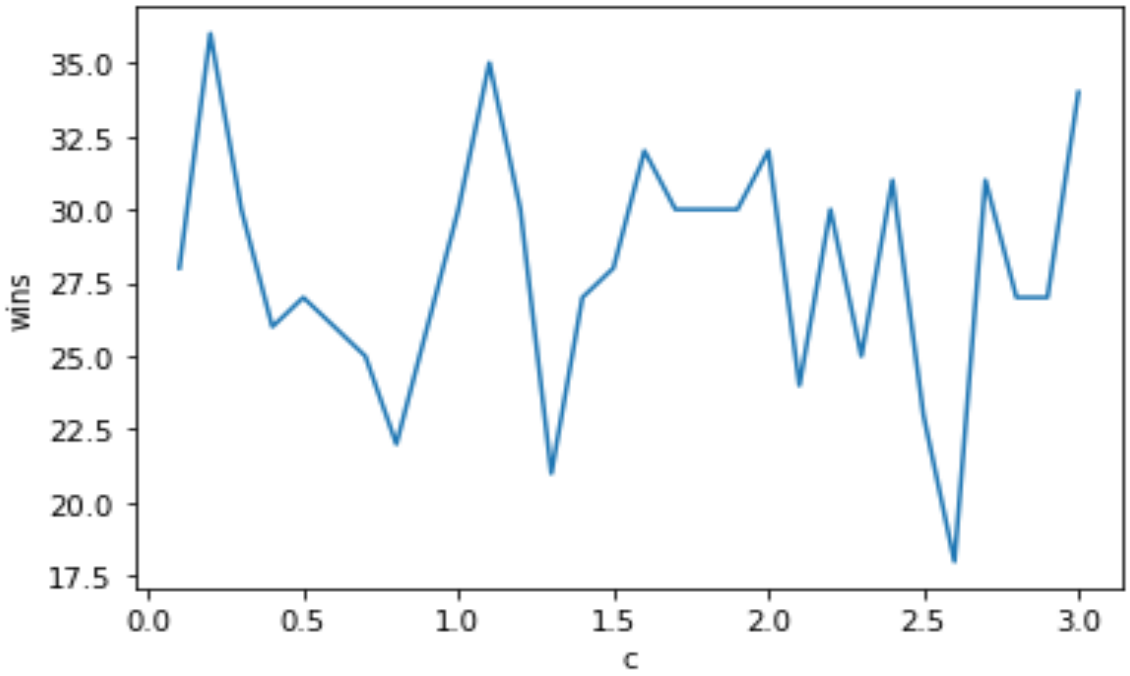
This result was seen which appears to be the best for MCTS200 among the results for other values o**f c**

Whereas Over exploration and under exploration both hurts MCTS200. as can be seen around c = 2,

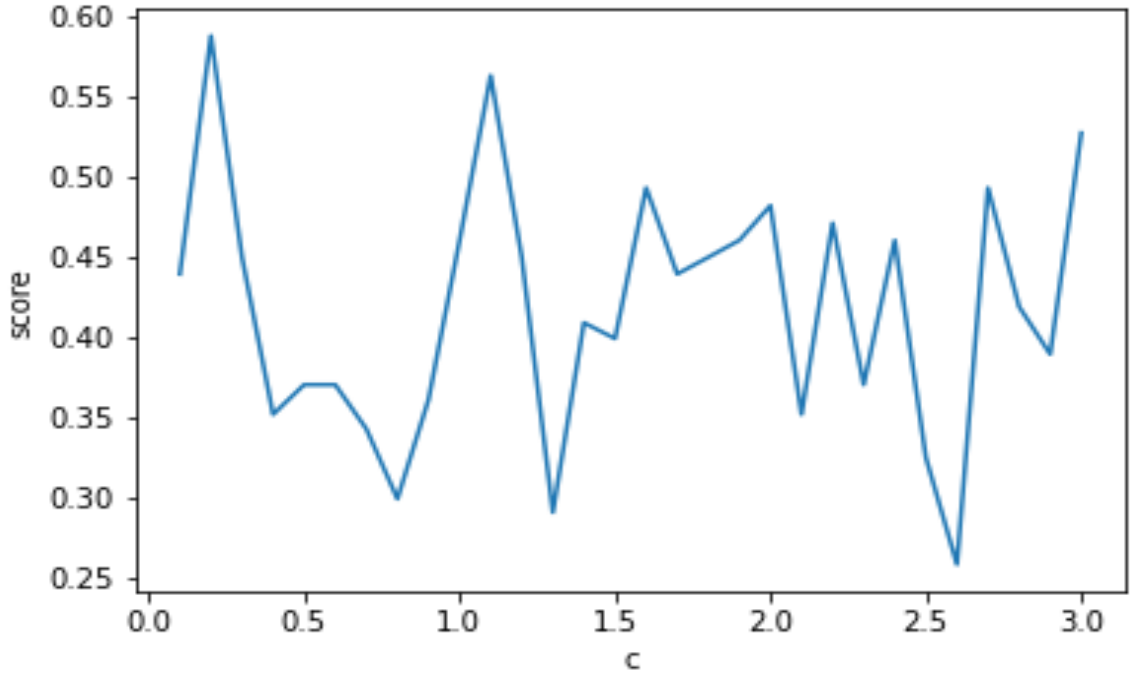
Which match with the statistics we get by playing 100 games and it’ll lose its Hold over MCTS40 for that c

b. MCTS40 vs MCTS200

Below is the graph for MCTS40vMCTS400, where wins account for wins of MCTS40



And the plot for the score as the formula was stated above



It can be seen from the score that , MCTS40 perform well for smaller c (basically MCTS200 performs suboptimally) and it can also be seen from the observations.

The drop in MCTS200 wins can be seen below as compared to it being the first player can be explained by winning bias of the first player. Though due to lesser information , still , Majority of the games are won by MCTS200 As can be seen through the picture below.

```
RED WINS: 30
BLACK WINS: 69
DRAW: 1
```

2. **Q Learning.**

   I wasn't able to do this properly due to amount of errors I was getting and then the explosion of values that I couldn't find the error for. Though I was able to get some results for QLearning vs a Random Agent where QLearningAgent is the First player(Other configurations lead to problems and errors I could not fix) and accordingly I have uploaded the .dat.gz file as well for the given scenario where the QL agent wins most of the times.

   How the Q- Table was made:

   Using defaultdict from collections , every state is assigned a value of 0.2 (positive expections),

   For mapping a state to the value, we hash the state using ternary hashing as it falls under int64's range and it also wont have any collisions with any other state.

   Here, the use of afterstates can be useful as we can read a place from distinct positions and rather that estimating value for a state action pair, we can estimate value for a state that we might be going to,

   With a stochastic environment , the idea of afterstates might not be that good as the agent won't know the next state, here the tabulation of action is necessary as it would keep the expected reward for the action from the state, but if the environment is deterministic , we can directly move to a state with highest approximated Q Value.

   As I could not perform experiments over hyperparameters on Q learing, I would not be commenting on that.

   (FOR Extracting the .dat.gz file, dill library would be needed as pickle doesn't allow defaultdict)

3. We can make the QLearningAgent win in the minimum amount of moves by giving it a suitable discounting factor. As the effect of the reward decreases as more number of steps are taken, in the end it will try to ensure that minimum number of steps are taken to win , similarly another not so good approach is to give it a small negative reward which might lead to similar results though the first option should be preffered.