As suggested on <https://stackoverflow.com/q/1291377/618018> I started with simple and dumb heuristics. Complexities can be added to fun if they give better results. The heuristics in sample\_players.py are not complicated.

Is it worth to add different variables?

I started with simple heuristic function which had legal moves for both players. This approach gave results in range of 60% to 70%.

Next I move to add blank space as next variable to heuristic. Counting blank spaces seems to be natural but it made results go south up to 50%.

3 chosen heuristics are simple and they depend on legal moves.

def custom\_score(game, player): \_:

my\_score = len(game.get\_legal\_moves(player)) + 1

opponent\_score = len(game.get\_legal\_moves(game.get\_opponent(player))) + 1

return float(my\_score/opponent\_score)

return value of > 1 implies opponent has less moves to play. Length is increased for both players to avoid division by 0 or return value as 0.

def custom\_score\_2(game, player):

my\_score = len(game.get\_legal\_moves(player))

opponent\_score = len(game.get\_legal\_moves(game.get\_opponent(player)))

return my\_score - 2. \* opponent\_score

return value is calculated by subtracting opponent moves by factor of two from our player.

def custom\_score\_3(game, player):

my\_score = len(game.get\_legal\_moves(player))

opponent\_score = len(game.get\_legal\_moves(game.get\_opponent(player)))

return float(my\_score^2 - opponent\_score^2)

return value is calculated by taking diff in sum of scores. This will pump up the difference by factor of 2.

My result text has, “Your agents forfeited 169.0 games while there were still legal moves available to play.” It means my agent is returning no moves when there are moves and my minimax/alpha beta function is not able to find them. Instead of returning (-1, -1) I returned first legal move. This helped to remove this error.

Final result in next page,

