HEURISTIC ANALYSIS

By

Asarudheen

For Build an Adversarial Search Agent Project

Udacity

HEURISTIC 1: custom_score

The main idea for creating the heuristic to prioritize the scenario if the number of own moves is greater than opponent moves and scenario with maximum own_moves. Below is the snippet of the custom_score heuristics.

```
if own_moves-opp_moves>0:
    return float((own_moves-opp_moves)*own_moves)
else:
    return float(own_moves - opp_moves)
```

If number of own move is less than number of opposite move then we don't have to prioritize the move to we will return the score as the difference between number of own_moves and opp_moves.

HEURISTIC 2: custom_score_2

Below is the code snippet of the custom_score_2 heuristic.

```
if opp_moves >= 2 and my_moves > 3:
    return float(my_moves*2)
elif opp_moves<2 and my_moves >2:
    return float(my_moves*5)
elif opp_moves <1 and my_moves >= 1:
    return float(my_moves*10)
else:
    return float(my_moves-opp_moves)
```

The main idea behind this heuristic is to defend and reduce the number of moves for the opponent. By increasing the score by 10 times if opponent has less than one move even current player is left with one move. This heuristic helps in taking risky and critical decision.

HEURISTIC 3: custom_score_3

```
if opp_moves > 2:
    return float(own_moves*0.5)
else:
    return float(own_moves*0.7)
```

This Heuristic mainly focus on defending. Based on the number of opponents move will try to reduce the score my multiplying the factor 0.5 or 0.7 with the number of current own move.

CONCLUSION:

The below is the score achieved by the heuristics explained above.

Case 1: NUM_MATCHES = 5

```
                    *************************
                         Playing Matches
                    *************************

Match #    Opponent    AB_Improved    AB_Custom    AB_Custom_2    AB_Custom_3
                       Won | Lost    Won | Lost    Won | Lost    Won | Lost
   1       Random       9  |  1      10  |  0      10  |  0      9  |  1
   2       MM_Open      7  |  3       8  |  2       8  |  2      8  |  2
   3      MM_Center     8  |  2       9  |  1       8  |  2      9  |  1
   4     MM_Improved    8  |  2       8  |  2       7  |  3      7  |  3
   5       AB_Open      5  |  5       4  |  6       5  |  5      4  |  6
   6      AB_Center     7  |  3       5  |  5       4  |  6      6  |  4
   7     AB_Improved    5  |  5       7  |  3       6  |  4      5  |  5
--------------------------------------------------------------------------
          Win Rate:      70.0%        72.9%        68.6%        68.6%
```

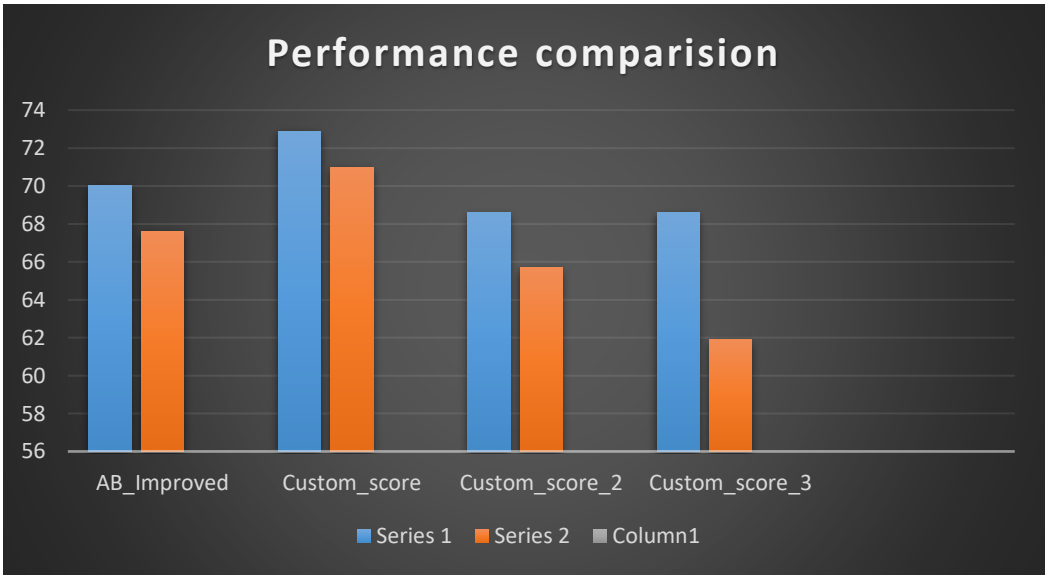
Case 2: Number of Matches increased to 15

```
                    **************************
                         Playing Matches
                    **************************

Match #   Opponent     AB_Improved    AB_Custom    AB_Custom_2   AB_Custom_3
                       Won  | Lost    Won  | Lost   Won  | Lost   Won  | Lost
   1        Random      24  |   6     28  |   2    30  |   0    28  |   2
   2        MM_Open     22  |   8     23  |   7    25  |   5    20  |  10
   3       MM_Center    27  |   3     25  |   5    25  |   5    23  |   7
   4     MM_Improved    21  |   9     23  |   7    21  |   9    17  |  13
   5        AB_Open     16  |  14     15  |  15     9  |  21    14  |  16
   6       AB_Center    17  |  13     19  |  11    15  |  15    14  |  16
   7     AB_Improved    15  |  15     16  |  14    13  |  17    14  |  16
-------------------------------------------------------------------------
          Win Rate:      67.6%        71.0%        65.7%        61.9%
```

From the above two cases its proven that AB_Custom is performing better when compared to AB_Improved heuristics. In AB_Improved we are just finding the difference between the number of own moves and opponent moves but in AB_Custom we are getting the difference and multiplying with number of own_moves if own_moves is greater than opponent moves. This condition will give importance to a move which leads to maximum game play when compared to AB_Improved.

Recommendation about which evaluation function:



From the above visualization we could come to a conclusion that heuristics 1(Custom_score) is performing better than AB_Improved. During all the test run's Custom_score had better win percentage than AB_Improved and While playing against AB_Improved also Custom_score had more winning matches than AB_Improved. With Custom_score we can prioritize the move which leads to maximum number of moves in upcoming turns which indirectly affects the opponent's move.

When we compare Custom_score heuristics with other heuristics, space and time complexity is minimal with maximum win percentage. Need further research to tune the Custom_score_2 to perform better than other heuristics.