Heuristic Analysis

```
This script evaluates the performance of the custom_score evaluation
function against a baseline agent using alpha-beta search and iterative
deepening (ID) called `AB_Improved`. The three `AB_Custom` agents use
ID and alpha-beta search with the custom_score functions defined in
game_agent.py.

                    **************************
                         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     2  |   8    10  |   0     8  |   2
    2       MM_Open     6  |   4     0  |  10     9  |   1     7  |   3
    3      MM_Center    9  |   1     3  |   7     8  |   2    10  |   0
    4     MM_Improved   8  |   2     3  |   7     9  |   1     7  |   3
    5       AB_Open     7  |   3     1  |   9     6  |   4     5  |   5
    6      AB_Center    8  |   2     2  |   8     7  |   3     5  |   5
    7     AB_Improved   5  |   5     0  |  10     7  |   3     2  |   8
--------------------------------------------------------------------------
          Win Rate:      74.3%        15.7%        80.0%        62.9%
```

Based the above output from the tournament, we recommend using **AB_Custom_2** evaluation function, with a win rate of 80%, for the following reasons:

- It implements the strategy of maximizing the chances of active player having most number of moves available in the end to win the game.
- The active layer with the most number of moves than the opponent has an advantage, and this is represented by assigning a score of +4. When player is not active but has more number of moves than opponent, there is a slight disadvantage but it's still favorable to the player. This is shown by assigning a score of +2. The most disadvantaged position for the player is when he is inactive and has least number of moves than the opponent. Such moves are penalized by assigning a score of -4.
- When looking at the win rates just for AB_* opponent matches, the win rate for AB_Custom_2 and AB_Improved are very similar. This could be due to the nature of both evaluation functions where they both look at the relative advantage of the player in-terms of moves, and improved score has more variation in-terms of magnitude of improvement, whereas AB_Custom_2 assigns fixed scores regardless of magnitude of variation.

AB_Custom_3 is the second function among the three custom functions with a win rate of approx.63%. AB_Custom_3 score looks at if the current player can attack the opposition by looking at overlapping moves and moving there on the consequent turn.

AB_Custom is the least performing evaluation function with a win rate of approx.16%. This was meant to be an implementation of weighted scoring functions, and it will need further revision in-terms of appropriate weights tuning and assignment to make it more competitive.