Isolation Game Heuristic Analysis:

A good heuristic function must be capable of capturing the following:

1. Number of remaining moves of both player and Opponent
2. Distance(Simple Vs Euclidean) of player and opponent from center of board since the center provides positional advantages for the winner.
3. Sum of Maximum moves over Minimal number of remaining squares a player or opponent can possibly make before they are cornered. i.e exploiting the maximum number of moves of a player/opponent over the minimum number of available squares or positions.

Thus, we have implemented our Heuristic functions(custom_score, custom_score_2, custom_score_3) to capture a combination of these different properties
of a heuristic and selected the best combination we believe from data showed our best player performance.

Given center position to of the isolation board to be (cx, cy) and a player position (x, y) then:

1) Simple Distance = abs(x-cx) + abs(y-cy)
2) Euclidean Distance = sqrt(| x-cx|^2 + |y-cy|^2)

COMBINATION I

custom_score:

It combines both:
1) The number of player + opponent moves
2) The distance(Simple) measure to capture positional advantage
3) Sum of Maximum moves over Minimal number of remaining squares.

custom_score_2:

It combines both:
1) Sum of Maximum moves over Minimal number of remaining squares

custom_score_3:

It combines both:
1) The distance measure(Simple) to capture positional advantage
2) Sum of Maximum moves over Minimal number of remaining squares

Result for Combination 1:

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

Match #	Opponent	AB_Improved		AB_Custom		AB_Custom_2		AB_Custom_3	
		Won	Lost	Won	Lost	Won	Lost	Won	Lost
1	Random	8	2	10	0	5	5	7	3
2	MM_Open	5	5	7	3	8	2	4	6
3	MM_Center	8	2	8	2	6	4	7	3
4	MM_Improved	4	6	7	3	5	5	5	5
5	AB_Open	5	5	7	3	6	4	3	7
6	AB_Center	6	4	6	4	7	3	3	7
7	AB_Improved	5	5	4	6	1	9	4	6
	Win Rate:	58.6%		70.0%		54.3%		47.1%	

Your ID search forfeited 176.0 games while there were still legal moves available to play.

COMBINATION II

custom_score:

It combines:

1) The number of player + opponent moves
2) The distance(Simple) measure to capture positional advantage
3) Sum of Maximum moves over Minimal number of remaining squares

custom_score_2:

It combines:

1) Number of Player/Opponent remaining moves
2) Sum of Maximum moves over Minimal number of remaining squares

custom_score_3:

It combines both:

1) The distance(Euclidean) measure to capture positional advantage
2) Sum of Maximum moves over Minimal number of remaining squares

Result for Combination II:

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

Match #	Opponent	AB_Improved Won | Lost	AB_Custom Won | Lost	AB_Custom_2 Won | Lost	AB_Custom_3 Won | Lost
1	Random	6 | 4	7 | 3	10 | 0	5 | 5
2	MM_Open	5 | 5	6 | 4	6 | 4	7 | 3
3	MM_Center	6 | 4	9 | 1	8 | 2	7 | 3
4	MM_Improved	6 | 4	6 | 4	6 | 4	2 | 8
5	AB_Open	6 | 4	5 | 5	7 | 3	3 | 7
6	AB_Center	7 | 3	7 | 3	6 | 4	6 | 4
7	AB_Improved	6 | 4	4 | 6	3 | 7	7 | 3
	Win Rate:	60.0%	62.9%	65.7%	52.9%

Your ID search forfeited 178.0 games while there were still legal moves available to play.

COMBINATION III

custom_score:

combines:

1) The number of player + opponent moves
2) The distance(Simple) measure to capture positional advantage
3) Sum of Maximum moves over Minimal number of remaining squares

custom_score_2:

combines:

1) Number of Player/Opponent remaining Moves!
2) Sum of Maximum moves over Minimal number of remaining squares

custom_score_3:

combines:

1) The distance(Simple) measure to capture positional advantage
2) Sum of Maximum moves over Minimal number of remaining squares

Result for Combination III:

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

Match #   Opponent        AB_Improved   AB_Custom   AB_Custom_2   AB_Custom_3
                          Won | Lost    Won | Lost  Won | Lost    Won | Lost
    1       Random          7 |  3        8 |  2      8 |  2        7 |  3
    2       MM_Open         7 |  3        7 |  3      6 |  4        3 |  7
    3       MM_Center       8 |  2        6 |  4      8 |  2        7 |  3
    4       MM_Improved     8 |  2        6 |  4      7 |  3        6 |  4
    5       AB_Open         5 |  5        6 |  4      7 |  3        4 |  6
    6       AB_Center       8 |  2        4 |  6      7 |  3        6 |  4
    7       AB_Improved     2 |  8        5 |  5      6 |  4        4 |  6
                      ------------------------------------------------------------
            Win Rate:      64.3%        60.0%        70.0%        52.9%
```

Your ID search forfeited 171.0 games while there were still legal moves available to play.

COMBINATION IV

custom_score:

combines:

1) The number of player + opponent moves
2) The distance(Euclidean) measure to capture positional advantage
3) Sum of Maximum moves over Minimal number of remaining squares

custom_score_2:

combines:

1) Number of Player/Opponent remaining moves!
2) Sum of Maximum moves over Minimal number of remaining squares
3) The distance(Simple) measure to capture positional advantage

custom_score_3:

combines:

1) Number of player/opponent remaining moves!
2) The distance(Simple) measure to capture positional advantage.

Result for Combination IV:

****
Playing Matches
****

Match #	Opponent	AB_Improved	AB_Custom	AB_Custom_2	AB_Custom_3
		Won | Lost	Won | Lost	Won | Lost	Won | Lost
1	Random	8 | 2	10 | 0	9 | 1	8 | 2
2	MM_Open	6 | 4	8 | 2	8 | 2	6 | 4
3	MM_Center	8 | 2	6 | 4	7 | 3	8 | 2
4	MM_Improved	8 | 2	6 | 4	6 | 4	8 | 2
5	AB_Open	7 | 3	6 | 4	7 | 3	7 | 3
6	AB_Center	5 | 5	6 | 4	6 | 4	6 | 4
7	AB_Improved	5 | 5	4 | 6	6 | 4	4 | 6
	Win Rate:	67.1%	65.7%	70.0%	67.1%

Your ID search forfeited 160.0 games while there were still legal moves available to play.

COMBINATION V

custom_score_3:

combines:

1) The number of player + opponent moves
2) The distance(Simple) measure to capture positional advantage
3) Sum of Maximum moves over Minimal number of remaining squares

custom_score_2:

combines:

1) Number of Player/Opponent remaining moves!
2) Sum of Maximum moves over Minimal number of remaining squares
3) The distance(Simple) measure to capture positional advantage

custom_score:

combines:

1) Number of Player/Opponent remaining moves!
2) The Simple distance measure to capture positional advantage
3) The Euclidean distance measure to capture positional (L-shaped) advantage

Result for Combination V:

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

Match #   Opponent    AB_Improved   AB_Custom   AB_Custom_2  AB_Custom_3
                      Won | Lost   Won | Lost   Won | Lost   Won | Lost
    1     Random        7  |  3      8  |  2      8  |  2      9  |  1
    2     MM_Open      10  |  0      9  |  1      9  |  1      6  |  4
    3     MM_Center     6  |  4      9  |  1      6  |  4      9  |  1
    4     MM_Improved   7  |  3      6  |  4      7  |  3      6  |  4
    5     AB_Open       8  |  2      5  |  5      6  |  4      6  |  4
    6     AB_Center     5  |  5      8  |  2      5  |  5      6  |  4
    7     AB_Improved   6  |  4      6  |  4      6  |  4      2  |  8
                  --------------------------------------------------------------
          Win Rate:     70.0%       72.9%       67.1%       62.9%

Your ID search forfeited 158.0 games while there were still legal moves available to play.
```

Conclusion:

After trying several combinations, it became self evident that,
the heuristic function was mostly influenced by the Number of Player/Opponent Remaining moves and a combining it with any of the other factors(introducing the other property if the result from a given properties led to a draw e.g Number of remaining moves produced a draw or tie game).

Looking at custom_score_3 heuristic function of **COMBINATION II** and **COMBINATION III,** we observed that whether we used Euclidean or Simple distance the heuristic performance was the same.

Using **COMBINATION V** as the final implementation, we believe custom_score heuristic function takes into account all these separate contributions and gives the most stable performance if the same combination is ran repeatedly several times. So we consider **custom_score** our best heuristic function although even custom_score_2 or custom_score_3 could equally be our chosen heuristic function.

The heuristic implementation(custom_score_3, custom_score_2, custom_score) which led to the best performance as shown in the code.