Analysing custom_score3

First the custom score implement to include three factors in calculation:

1. 60% of score: number of moves for player/opponents
2. 20% of score: player near to center give more score
3. 20% of score: opponent near to corners give more score

The tournament test performed to compare it with AB_Improved (player moves - opponent mvoe) and the results printed as following with two different runs:

Run1

Opponent	AB_Improved Won	AB_Improved Lost	AB_Custom_3 Won	AB_Custom_3 Lost
Random	9	1	10	0
MM_Open	7	3	4	6
MM_Center	7	3	9	1
MM_Improved	5	5	4	6
AB_Open	5	5	5	5
AB_Center	7	3	6	4
AB_Improved 4	5	5	6	
Win Rate:	65.7%		62.9%	

Run2

Opponent	AB_Improved Won	AB_Improved Lost	AB_Custom_3 Won	AB_Custom_3 Lost
Random	9	1	9	1
MM_Open	6	4	5	5
MM_Center	9	1	8	2
MM_Improved	4	6	8	2
AB_Open	4	6	6	4
AB_Center	5	5	4	6
AB_Improved	5	5	5	5
Win Rate:	60.0%		64.3%	

AB_Custom_3 performed avarage number of wining against testing agents (63.25) while AB_Improved perfomed avarage number of wining (62.85)

In seconed test the custom heuristic function performed better results as well as the avarage result of both cases.

Analysing custom_score2

Custom score 2 calculation made to make agent in attack mode: (player moves - opponent mvoe*2)

The tournament performed two different times with following results:

Run1

Opponent	AB_Improved Won	AB_Improved Lost	AB_Custom_2 Won	AB_Custom_2 Lost
Random	9	1	10	0
MM_Open	7	3	4	6
MM_Center	7	3	9	1
MM_Improved	5	5	5	5
AB_Open	5	5	4	6
AB_Center	7	3	4	6
AB_Improved 4	5	5	6	4
Win Rate:	65.7%		60.0%	

Run2

Opponent	AB_Improved Won	AB_Improved Lost	AB_Custom_2 Won	AB_Custom_2 Lost
Random	9	1	10	0
MM_Open	6	4	6	4
MM_Center	9	1	10	0
MM_Improved	4	6	4	6
AB_Open	4	6	5	5
AB_Center	5	5	6	4
AB_Improved	5	5	6	4
Win Rate:	60.0%		67.1%	

AB_Custom_2 performed avarage number of wining against testing agents (63.5) while AB_Improved perfomed avarage number of wining (62.85)

The avarage result of AB_Custom_2 perfomed better than AB_Improved

Analysing custom_score1

AB_Custom made with combination of previous two score functions, it use the calculation with percantage similar to AB_Custom_3 but editing point 1 (number of moves for player/opponents) as following:

70% of score for number of moves:

- if (player1 have more than two moves): player moves - opponent mvoe*2

- if (player1 have more than two moves): player moves - opponent mvoe

In addtion, another factor added to return infinity if opponent about lose while player still have moves.

Three testing runs perfomed where **custom_score1** made better results in both cases:

Run1

Opponent	AB_Improved		AB_Custom		AB_Custom_2		AB_Custom_3	
	Won	Lost	Won	Lost	Won	Lost	Won	Lost
Random	8	2	10	0	8	2	8	2
MM_Open	7	3	7	3	6	4	8	2
MM_Center	9	1	8	2	6	4	7	3
MM_Improved	5	5	6	4	5	5	7	3
AB_Open	6	4	8	2	4	6	3	7
AB_Center	4	6	5	5	5	5	5	5
AB_Improved	6	4	5	5	4	6	5	5
Win Rate:	64.3%		70.0%		54.3%		61.4%	

Run2

Opponent	AB_Improved		AB_Custom		AB_Custom_2		AB_Custom_3	
	Won	Lost	Won	Lost	Won	Lost	Won	Lost
Random	9	1	9	1	9	1	9	1
MM_Open	5	5	6	4	5	5	5	5
MM_Center	9	1	8	2	7	3	8	2
MM_Improved	6	4	5	5	5	5	5	5
AB_Open	5	5	5	5	4	6	5	5
AB_Center	4	6	5	5	7	3	6	4
AB_Improved	6	4	5	5	6	4	6	4
Win Rate:	62.9%		61.4%		61.4%		62.9%	

AB_Custom_2 performed avarage number of wining against testing agents (65.7) while AB_Improved perfomed avarage number of wining (63.6)

Final analysis

Based on the results (**custom_score1**) selected to be my **ID_Improved** agent because:

1- Combine three strategy to calculate the score.

2- The score of function help to detect winner path better than other functions.

3- Ease of implementation.