# Heuristic_analysis

- custom_score
  - Analysis
    - Here we are trying to maximizing difference between number of left moves squared
    - Always beat AB_Improved agent
  - Implementation

    my_moves = len(game.get_legal_moves(player))
    opp_moves = len(game.get_legal_moves(game.get_opponent(player)))
    return 1.5 * float(my_moves * my_moves - opp_moves * opp_moves)

- custom_score_2
  - Analysis
    - trying to maximize ratio of difference between my_moves and opp_moves to opp_moves
    - Trying to minimize ratio of difference between opp_moves and my_moves to my_moves
    - Always beat AB_Improved agent
  - implementation

    my_moves = len(game.get_legal_moves(player))
    opp_moves = len(game.get_legal_moves(game.get_opponent(player)))
    return 1.5 * float((my_moves -opp_moves)/ (opp_moves + .000001)) - 1.5 * float((opp_moves -my_moves)/ (my_moves + .000001))

- custom_score_3
  - Analysis
    - trying to increase distance between two players and in the same time trying to increase my_moves
    - Can Beat AB_Improved agent but not all the time
  - implementation

    my_location = game. get_player_location(player)
    opp_location = game.get_player_location(game.get_opponent(player))
    my_moves = len(game.get_legal_moves(player))
    if my_location and opp_location:
        return float(abs(sum(my_location) - sum(opp_location))) + my_moves

```
        elif opp_location:
            return float("-inf")
        else:
            return float("inf")
```

## Results

| Match # | Opponent | AB_Improved | | AB_Custom | | AB_Custom_2 | | AB_Custom_3 | |
|---|---|---|---|---|---|---|---|---|---|
| | | Won | Lost | Won | Lost | Won | Lost | Won | Lost |
| 1 | Random | 27 | 3 | 28 | 2 | 28 | 2 | 29 | 1 |
| 2 | MM_Open | 23 | 7 | 24 | 6 | 23 | 7 | 25 | 5 |
| 3 | MM_Center | 25 | 5 | 24 | 6 | 26 | 4 | 28 | 2 |
| 4 | MM_Improved | 17 | 13 | 23 | 7 | 20 | 10 | 18 | 12 |
| 5 | AB_Open | 16 | 14 | 19 | 11 | 19 | 11 | 15 | 15 |
| 6 | AB_Center | 17 | 13 | 17 | 13 | 18 | 12 | 14 | 16 |
| 7 | AB_Improved | 17 | 13 | 15 | 15 | 19 | 11 | 8 | 22 |
| | Win Rate: | 67.6% | | 71.4% | | 72.9% | | 65.2% | |

## Recommendation

custom_score_2 is recommended for two reasons

- It has best winning rate (72.9 %)
- Always beat AB_Improved agent
- No complex operation so it can go deep in the tree
- Try to beat opponent with from two point of view
    - trying to maximize ratio of difference between my_moves and opp_moves to opp_moves
    - Trying to minimize ratio of difference between opp_moves and my_moves to my_moves