Heuristic Analysis

Approach

In order to devise the *custom_score* functions logic, the following approach was used:

1. Using self-observation and intuition when playing the game of isolation, I wrote down a set of strategies I noticed myself applying during the game to evaluate the state of my player in the game at any point in time. The following, are those strategies:

   - Comparing the amount of my own moves vs the opponent's: the more moves I had over my opponent's, the better.
   - My ability to block the opponent on their next move: since the goal of the game is to isolate the opponent, blocking their future moves seemed like a natural strategy.
   - My ability to take over the center of the board on my next move: this was also pointed out during the class instructional videos.
   - A relative measure of how close my player was to the center as compared to the opponent: in general, if my player was closer to the center than the opponent, the better it seemed, as it creates the notion of "controling" the center.
   - A simple count of the number of possible moves for my player: this is also captured in the out-of-the-box open-move strategy.

   There were other strategies I came up with during my brainstorming, but they were later discarded as irrelevant after further evaluation.

2. All of these strategies were later coded individually or in combination in the *custom_score* functions available in `game_agent.py`. When coded in combination, a similar approach to the one used when creating a Machine Learning model was utilized. I.e., a score function composed of the summation of the weighted features (strategies in this case) was applied. The general equation of the resulting function is:

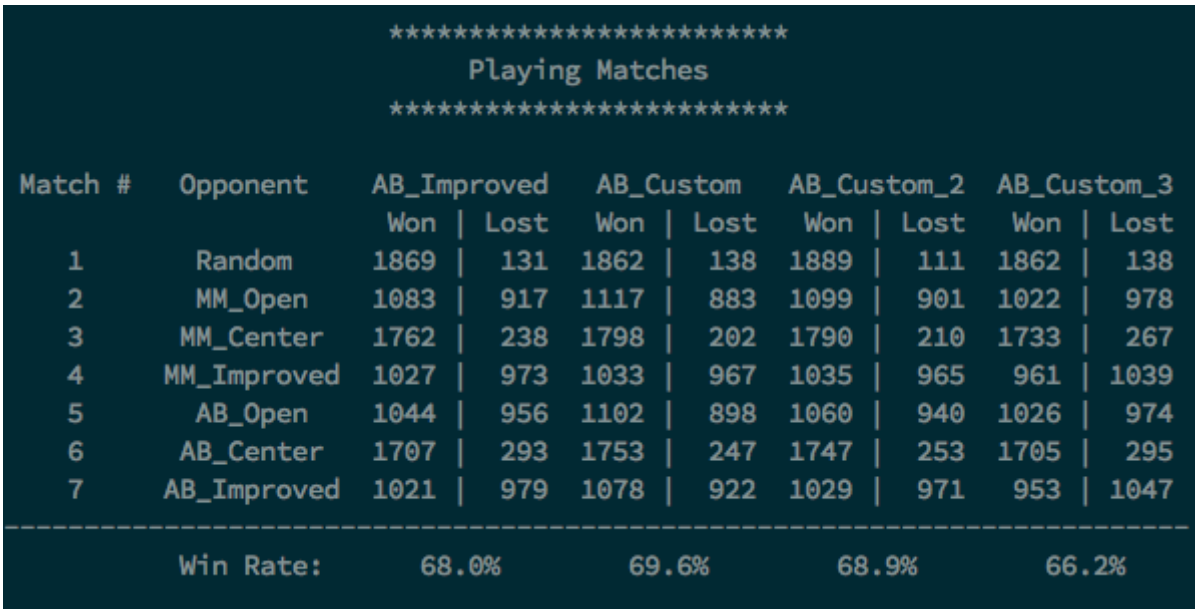
$$score = \sum_{i=1}^{n} \omega_i * x_i \quad , where\ n = number\ of\ strategies$$

3. Afterwards, their performance were evaluated by running `tournament.py` over sevaral runs. The evaluation included observing the following characteristics in the results:

   - noting the total *win rate* for each test player.
   - comparing all *AB_Custom** players to each opponent's result (paying attention to the opponent's strategy).
   - comparing all *AB_Custom*** *players results to the* AB_Improved* test player's for each opponent.

4. In an iterative fashion, each *custom_score* function was updated, by changing their param weights (ω_i) and adding or removing features to the equation (x) and further evaluated. This process allowed for the fine-tuning and optimization of the functions.

5. After several iterations, the final version of the *custom_score* functions was produced. As the last verification step, a *tournament* consisting of "`NUM_MATCHES = 1000`" was executed. The following screen shot summarizes the final results.



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

Match #   Opponent    AB_Improved   AB_Custom   AB_Custom_2  AB_Custom_3
                       Won | Lost   Won | Lost   Won | Lost   Won | Lost
   1       Random     1869 |  131  1862 |  138  1889 |  111  1862 |  138
   2       MM_Open    1083 |  917  1117 |  883  1099 |  901  1022 |  978
   3      MM_Center   1762 |  238  1798 |  202  1790 |  210  1733 |  267
   4     MM_Improved  1027 |  973  1033 |  967  1035 |  965   961 | 1039
   5       AB_Open    1044 |  956  1102 |  898  1060 |  940  1026 |  974
   6      AB_Center   1707 |  293  1753 |  247  1747 |  253  1705 |  295
   7     AB_Improved  1021 |  979  1078 |  922  1029 |  971   953 | 1047
--------------------------------------------------------------------------
          Win Rate:      68.0%        69.6%        68.9%        66.2%
```

Results of tournament final version of the custom_score functions over 1000 matches

The results show both *AB_Custom* and *AB_Custom_2* perform slightly better than the *AB_Improved* test agent. Also, it is interesting to note that the strategy implemented by *AB_Custom* (`custom_score()`) tends to perform really well against the out-of-the-box "center" strategy (as shown by the performance against *MM_Center* and *AB_Center*); however, not as well, yet still better, when compared against the "open-move" and the "improved" strategies.

This suggests that certain strategies work better than others under certain situations, depending on what the opponent's strategy is. In other words, there may not be a silver-bullet strategy that works well on all situations. A better, and more intuitive approach is to use an adaptive stragegy - one that can dynamically adapt to the opponent's behavior, much like a smart player would act.

In conclusion, it is recommended that the *AB_Custom* evaluation function be used, because:

a) It performs better than the rest of the test agents under all provided scenarios.

b) It is relatively simple, as it only combines 2 basic strategies (as described below).

c) It can easily be fine-tuned, since it only has 2 features.

Strategies utilized per custom_score function

The following are the strategies utilized for each *custom_score* function:

- **`custom_score()`**:
  - `opponent_block_ability`
  - `relative_mobility`
- **`custom_score_2()`**:
  - `center_ability`
  - `opponent_block_ability`

    - own_mobility
    - relative_center_domination
    - relative_mobility
- **custom_score_3()**:
    - opponent_block_ability
    - own_mobility

Name	**Strategy**
center_ability	Ability to capture board center
opponent_block_ability	Ability to block the opponent
own_mobility	Player # of next moves
relative_center_domination	Player's distance to center vs opponent's
relative_mobility	Player's moves vs opponent's

Strategy names as used in the *custom_score* functions