Heuristic Analysis

The default and custom evaluation heuristics that were used to build my agent can be found in the chart below, along with their performance metrics.

Opponent	AB_Improved*	null_score*	open_move_score*	improved_score*	survival_improved_score	aggresive_improved_score	proximity_bias	center_proximinity_bias
Random	34**	25	29	31	33	31	29	30
MM_Open	21	15	15	13	14	15	9	17
MM_Center	20	22	22	24	21	22	25	26
MM_Improved	16	8	16	14	14	15	14	14
AB_Open	19	20	17	20	19	19	17	23
AB_Center	17	17	24	21	27	20	23	21
AB_Improved	16	14	25	18	22	15	13	16
Win Rate	51.1%	43.2%	52.9%	50.4%	53.6%	48.9%	51.4%	52.5%

** Heuristic that is provided by Udacity.*
*** The individual values are the number of victories for the 40 games played against an opponent.*

- **AB_Improved**: a baseline agent, that uses alpha-beta search and iterative deepening.
- **null_score**: presumes no knowledge for non-terminal states, and returns the same uninformative value (0) for all other states.
- **open_move_score**: outputs a score equal to the number of moves available to the user.
- **improved_score**: outputs a score equal to the difference in the number of moves available to the two players.
- **survival_improved_score**: same as the improved_score, except the number of available user moves are weighted twice as great as the number of available opponent moves. This encourages the agent to focus on its survival.
- **agressive_improved_score**: same as the improved_score, except the number of available opponent moves are weighted twice as great as the number of available user moves. This encourages the agent to play aggressive and focus on reducing the number of available moves the opponent has.
- **proximity_bias**: outputs a score equal to 100 minus the Manhattan distance between the user and opponent. This encourages the agent to move as close as possible to the opponent. *Note: 100 is used because it is a large enough value that the Manhattan distance will never be greater than it.*
- **center_proximity_bias**: outputs a score equal to 100 minus the Manhattan distance between the user and opponent plus the Manhattan distance between the user and the centre of the board. This encourages the agent to maintain a location advantage.

Based on the results, the best heuristic is **survival_improved_score**. Three reasons why this function proved best are:

1. It considers the state of both the user and opponent. In an adversarial game, it is necessary to consider all players when making a decision. Without accounting for this information, other players could exploit this lack of attention.
2. It prioritizes survival. Compared to aggressive_improved_score and improved_score, it is clear that it is necessary to consider both players, but it is best to focus on maximizing the number of moves that are available to the user.
3. It's fast. There are very few steps to execute this function so it is able to look at greater depths of the game tree during alpha-beta search. This is evident by the higher win ratio it has against the AB opponents compared to the MM opponents.