**Assignment 0**

**Overview:**

The first assignment in UNR’s Machine Learning class was to build a Connect 4 game implementing a MiniMax tree search with AlphaBeta pruning for the Artificial Intelligence. Special attention was focused on the static evaluator.

**Static Evaluator:**

I had the hardest time coming up with an appropriate static evaluator. Initially, my approach to the evaluation function was all wrong. Instead of analyzing the overall state of the board after each move, I was analyzing the effect of each move on the board.

Once I corrected this assumption, I was able to come up with a function that increasingly valued greater number of the player’s pieces in a row, which is quite obvious and intuitive. I got a little hung up on blocking moves, as I wasn’t sure if the AI was going to handle this directly. I did build blocking into my static evaluator by making any sequences of the player’s pieces a greater penalty than could be offset by the computer-sequential pieces (unless there was a winning move for the computer). The following is a table of the evaluator’s strengths and weaknesses.

An important note is that my moves list starts with the center column, since most of the game is played starting from the middle. That feature lends to the nature of the weaknesses described below.

**Table 1: Static Evaluator Characteristics**

|  |  |
| --- | --- |
| **Strengths** | **Weaknesses** |
| 1. Greater depth causes blocks to be made sooner | 1. Fills up center row even when vertical win is not possible |
| 1. Greater depth causes faster wins | 1. Greater depth causes blindness to threats in outer columns if middle columns are dense |