Assignment 4

Name: Diwakar Yalpi

UIN: 01127089

Github link: https://github.com/diwakaryalpi/AI-Assignment4

EVALUATION FUNCTION:

I have implemented an evaluation function for minimax algorithm in order to calculate the value which is being used by the algorithm to determine the next best possible move. The computer will choose the highest value obtained from executing the evaluation function and this value points to the exact location in Connect4. In the evaluation function, it will check if it can get consecutive fours, then threes and lastly twos.

Below formula calculates the total number of possible fours, threes and twos that the human makes and then subtract the total number of possible fours, threes and twos that the computer makes.

utility_value = (my_fours * 10 + my_threes * 5 + my_twos * 2) - (comp_fours *10 + comp_threes * 5 + comp_twos * 2)

Once the values are calculated, the column which has the highest corresponding value is chosen and then the move is played.

Minimax algorithm:

minimax.py is a program contains an implementation of the minimax algorithm for Connect Four. This file also has methods for creating states of the board and functions for checking status of a given state. minimax algorithm is used to predict the next best possible moves while playing connect 4 game.

Alphabeta pruning algorithm:

I have used alphabeta pruning to enhance the functionality. It's a condition where alpha value and beta value gets initialized and updated as program traverses through the branches, this avoids the unnecessary branching so that the program gets a better move. And when alpha>=beta we no longer need to go through the otherbranches from that particular node.

In the code, as it is hard to go till the leaf nodes, I'm looking only till 6 steps ahead, it takes for normal minimax algorithm to make a move, whereas alpha beta pruning does it in no time. You can change the depth to 4 to immediately see the move of the computer.

Analysis:

I tried to win over computer, but I am unable to do that. Computer won most of the times. The output.txt file has the whole output saved when the files minimax.py and alphabeta.py files are executed.

Run time:

- Initially when there are almost no coins on the board, simple minimax algorithm with a depth of 6 took almost 12sec to output a move whereas the alphabeta pruning algorithm just took 4sec.
- We can see a significant decrease in the run time for the algorithm. It's an indication of how many branches it avoided.