♟ Hexapawn ♟

At this point in precalculus we have just covered some basics of combinatorics including combinations and permutations. I am using the attached problem sets to introduce students to some fundamentals of machine learning as it relates to discrete mathematics.

Hexapawn is a classic machine learning demonstration popularized by Martin Gardner (`http://cs.williams.edu/~freund/cs136-073/GardnerHexapawn.pdf`).

The lesson sequence is roughly this:

Day 1

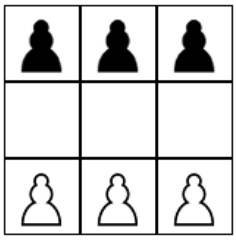
- Introduce the game of Hexapawn and let students play to get accustom to the rules and start creating strategies. Students can create simple boards by tearing or cutting out pieces of paper.
- Have students find solutions to Problem Set 1.
- Review solutions (possibly at the start of day 2).

Day 2

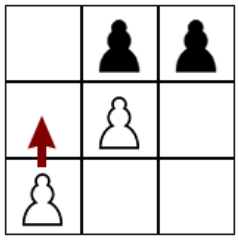
- Introduce the idea of a game tree and how a machine can "learn" to play by recording games in a tree and then using the known outcomes to make decisions about game play. Mention Arthur Samuel and how he was able to create a machine that eventually was able to outplay its creator.
- Have students find solutions to Problem Set 2. One of the important ideas in this problem set is to encourage students to think algorithmically so that they can start to realize the mechanics of a machine that can improve in task performance with experience.
- Review solutions.

♟ Hexapawn ♟

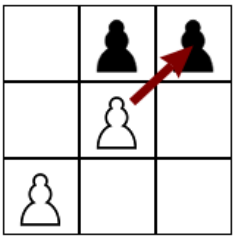
Hexapawn is a turn-based game using six chess pawns on a 3×3 board. White moves first. A piece can either move forward if the space in front of it is empty, or capture an opponent's piece diagonally.



(a) Starting position



(b) A forward move



(c) Capture diagonally

A player wins when:

(1) One of their pawns reaches the other side
(2) They capture all of their opponent's pieces
(3) It is their opponenet's turn and there is no legal move

Hexapawn is a *solved* game, meaning that we can calculate an optimal move for any given position. Get started playing and see if you can "solve" the game before attempting the questions below.

1. Is it possible to draw (tie) in hexapawn?
2. Does one side have the advantage?
3. What is the shortest game possible (fewest moves)?
4. What is the longest game possible (most moves)?
5. How many unique positions are possible after the first move (only white moves)?
6. How many unique positions are possible after the first two moves (white moves then black moves)?
7. Give an example of a position that is impossible to reach.
8. If white's first move is to move their left-most pawn forward, then what is black's worst response? What is black's best response?
9. If white's first move is to move their center pawn forward, then what is black's worst response? What is black's best response?
10. How many unique games are possible? Here, a game is represented by a sequence of moves.

♟ Hexapawn ♟

Browse to `jedediyah.github.io/data/hexapawn`. Try playing a few games! Let the machine play itself until it solves the game.

As the tree grows, there is more information recorded about the possible game outcomes. The machine is programmed to follow a path of least resistance when it moves. That is, it will make the move that leads to the least bad results that we know of in the tree.

1. As the machine plays itself, how do you know when the game has been solved?
2. What is the shortest game in the tree? What is the longest game in the tree?
3. After letting the machine play itself until one side always wins, there are still several games in the tree where each side has won. So how is it that one side will always win now? In other words, why are some previously reachable results now un-reachable?
4. Why doesn't the tree expand completely? In other words, why won't every possible game get played?
5. Find a game that hasn't yet been recorded in the tree, and play it into the tree.
6. Refresh the website, and start the machine playing itself again. Does it generate the same tree each time? What might it say about the algorithm if the tree is the same or different?
7. Refresh the website. Running one Machine vs. Machine game at a time, use the blank chart to keep track of the machine's score after each game. Each time black wins, the score increases by one. Each time white wins, the score decreases by one. Play enough games to solve Hexapawn.
   (a) How many games did it take before black always won?
   (b) On average how many games does it take before black always wins? What was the fewest number of games required? The most? It might be useful to ask your classmates to share their data.
   (c) How many wins total did white have?
   (d) On average how many wins did white have? What was the fewest number of wins for white? The most?

Game the System

Another student resets the tree and says that they are going to train the machine by playing games into the tree manually. Remember that when the machine makes a move, it can make a decision based only on the information currently in the tree.

8. How *could* each of the following affect the machine's decision making?
   (a) The student enters games into the tree such that the first move is always the left pawn.
   (b) The student enters one game for each possible first move (then stops).
   (c) The student enters games where they always try to make the worst possible move.
9. If you were going to play against one of the trees above, which would you choose in order to give yourself the best possibility of winning (as white)?
10. Machine learning and algorithms are commonly used to make decisions about lots of things. When a machine is making a decision, how can you decide whether or not to trust it?

Machine Score
-10
-9
-8
-7
-6
-5
-4
-3
-2
-1
0
1
2
3
4
5
6
7
8
9
10
11
5
10
15
20
25
30
35
Games Played