Sneh Shah (sjs413) - Section 2

Rishi Shah (rrs141) - Section 5

Intro to AI Project 2

()

May 19, 2021

Certification of Academic Integrity

I certify that the work done on this project is my own and is not copied or taken from online or any other student's work. - Rishi Shah
I certify that the work done on this project is my own and is not copied or taken from online or any other student's work. -Sneh Shah

Breakdown of Work

Rishi coded the advanced agent functions. He worked on the inference, decision, and efficiency sections of the write-up.

Sneh coded the basic agent functions. She worked on the representation and performance sections of the write-up.

We worked together to come up with the pseudocode for all functions together and split up the coding as mentioned above.

1. Representation

We represented the game board with a matrix. A -1 on the game board represents all the hidden cells, 0-8 represents the clue of a cell, a 9 represents an exploded mine, and a 10 represents a flagged mine.

For the basic agent, we represented the knowledge base as a matrix with the object cell (which keeps track of that cells status, the clue, how many neighbors are identified as safe, how many neighbors are identified as mines, how many hidden neighbors there are, and how many total neighbors there are).

For the advanced agent, we also represent the knowledge base with a matrix with the object advCell (which keeps track of that cells status, the clue, how many neighbors are identified as safe, how many neighbors are identified as mines, a list of the current hidden neighbors, and a list of all neighbors). We also have a list that keeps track of all active clues with each element being (x coordinate, y coordinate, clue). An active clue means that this coordinate still has hidden neighbors and this clue can still be used to identify the status of its hidden neighbors. Lastly, we have the data frame which keeps track of all valid combinations of hidden neighbors given the current list of clues.

2. Inference

When we get a new clue in the basic agent, we set the corresponding coordinate object's status to safe or mine (depending on what type of cell was clicked) and set its clue value equal to the value of the cell we clicked (9 if it's a bomb, 0-8 if regular clue). After that, we update the neighbors of cell, increasing the number of identified mines or the number of identified safe squares by 1 as well as reducing the amount of hidden squares by 1. We deduce everything we can about a given clue before moving on by checking to see if the total number of mines left equals the total number of hidden squares left, or if the total number of safe squares left equals to the number of hidden squares left. This is useful as if any of these cases are true, then we know the identity of the rest of the surrounding squares.

We expand this idea for advanced agent by keeping a knowledge base of valid possible combinations of the hidden squares for all active clues on the board. We model this by using a data frame where each column represents a variable/hidden neighbor. If any column contains only one value (either 0 or 1), then we can conclusively identify that cell as either safe (0) or mine (1). This uses multiple clues at once to make inferences, as we are considering multiple possible combinations at once to see and narrow down which values work and which values don't. We don't always conclude everything about a given clue before moving on. To improve upon this, we could re-run our scan board function if we identify something new from running through the combinations within the data frames.

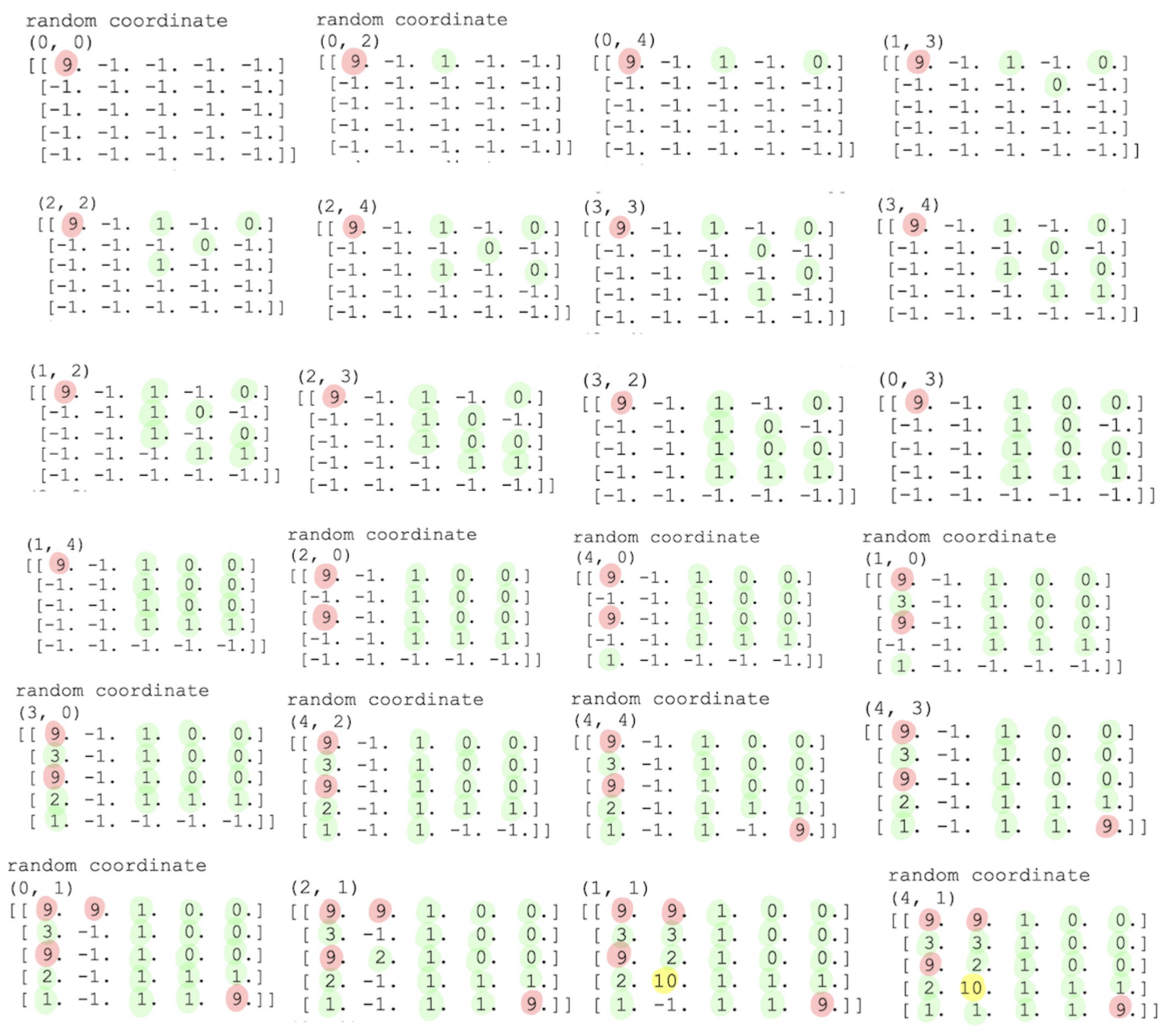
3. Decisions

Our program decides what cell to search next by having a fringe of all not yet explored, but identified safe cells. It pops from that fringe to choose what cell to go to next, as that for sure won't be a mine. If there is nothing in the fringe, then it would choose a random coordinate. For scan board, the knowledge base is sorted by having the most constrained square first. Basically, the higher the clue value is, the earlier it will be searched in the knowledge base. For clues that are more constrained like this, we are more likely to figure information about unknown cells, as the valid options are more limited.

4. Performance Play by Play

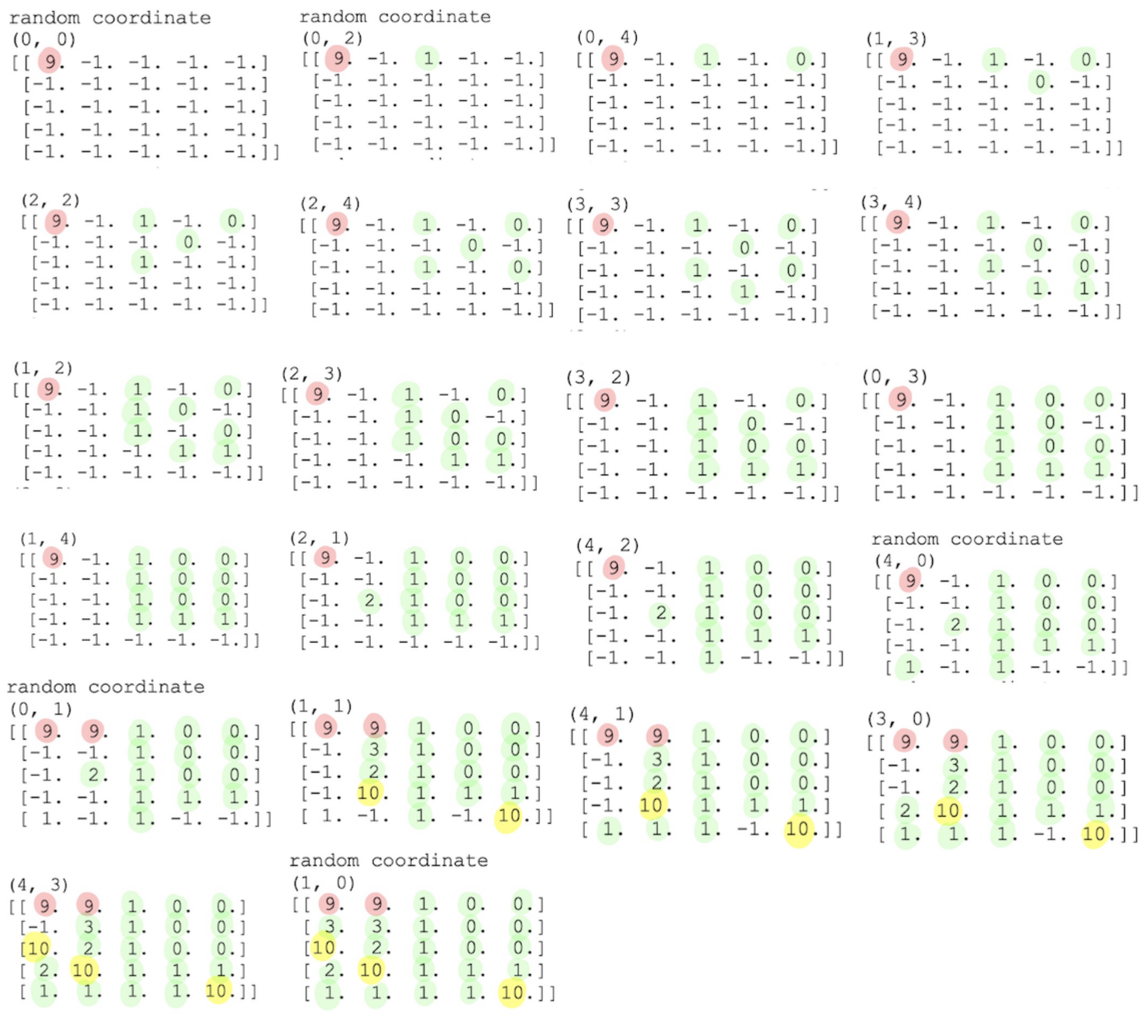
Below shows a play by play for a 5x5 board with 5 mines.

The following is the basic agent playing the game.



```
random coordinate
(0, 0)
[[ 9. -1. -1. -1. -1.]
 [-1. -1. -1. -1. -1.]
 [-1. -1. -1. -1. -1.]
 [-1. -1. -1. -1. -1.]
 [-1. -1. -1. -1. -1.]]

random coordinate
(0, 2)
[[ 9. -1.  1. -1. -1.]
 [-1. -1. -1. -1. -1.]
 [-1. -1. -1. -1. -1.]
 [-1. -1. -1. -1. -1.]
 [-1. -1. -1. -1. -1.]]

(0, 4)
[[ 9. -1.  1. -1.  0.]
 [-1. -1. -1. -1. -1.]
 [-1. -1. -1. -1. -1.]
 [-1. -1. -1. -1. -1.]
 [-1. -1. -1. -1. -1.]]

(1, 3)
[[ 9. -1.  1. -1.  0.]
 [-1. -1. -1.  0. -1.]
 [-1. -1. -1. -1. -1.]
 [-1. -1. -1. -1. -1.]
 [-1. -1. -1. -1. -1.]]

(2, 2)
[[ 9. -1.  1. -1.  0.]
 [-1. -1. -1.  0. -1.]
 [-1. -1.  1. -1. -1.]
 [-1. -1. -1. -1. -1.]
 [-1. -1. -1. -1. -1.]]

(2, 4)
[[ 9. -1.  1. -1.  0.]
 [-1. -1. -1.  0. -1.]
 [-1. -1.  1. -1.  0.]
 [-1. -1. -1. -1. -1.]
 [-1. -1. -1. -1. -1.]]

(3, 3)
[[ 9. -1.  1. -1.  0.]
 [-1. -1. -1.  0. -1.]
 [-1. -1.  1. -1.  0.]
 [-1. -1. -1.  1. -1.]
 [-1. -1. -1. -1. -1.]]

(3, 4)
[[ 9. -1.  1. -1.  0.]
 [-1. -1. -1.  0. -1.]
 [-1. -1.  1. -1.  0.]
 [-1. -1. -1.  1.  1.]
 [-1. -1. -1. -1. -1.]]

(1, 2)
[[ 9. -1.  1. -1.  0.]
 [-1. -1.  1.  0. -1.]
 [-1. -1.  1. -1.  0.]
 [-1. -1. -1.  1.  1.]
 [-1. -1. -1. -1. -1.]]

(2, 3)
[[ 9. -1.  1. -1.  0.]
 [-1. -1.  1.  0. -1.]
 [-1. -1.  1.  0.  0.]
 [-1. -1. -1.  1.  1.]
 [-1. -1. -1. -1. -1.]]

(3, 2)
[[ 9. -1.  1. -1.  0.]
 [-1. -1.  1.  0. -1.]
 [-1. -1.  1.  0.  0.]
 [-1. -1.  1.  1.  1.]
 [-1. -1. -1. -1. -1.]]

(0, 3)
[[ 9. -1.  1.  0.  0.]
 [-1. -1.  1.  0. -1.]
 [-1. -1.  1.  0.  0.]
 [-1. -1.  1.  1.  1.]
 [-1. -1. -1. -1. -1.]]

(1, 4)
[[ 9. -1.  1.  0.  0.]
 [-1. -1.  1.  0.  0.]
 [-1. -1.  1.  0.  0.]
 [-1. -1.  1.  1.  1.]
 [-1. -1. -1. -1. -1.]]

random coordinate
(2, 0)
[[ 9. -1.  1.  0.  0.]
 [-1. -1.  1.  0.  0.]
 [ 9. -1.  1.  0.  0.]
 [-1. -1.  1.  1.  1.]
 [-1. -1. -1. -1. -1.]]

random coordinate
(4, 0)
[[ 9. -1.  1.  0.  0.]
 [-1. -1.  1.  0.  0.]
 [ 9. -1.  1.  0.  0.]
 [-1. -1.  1.  1.  1.]
 [ 1. -1. -1. -1. -1.]]

random coordinate
(1, 0)
[[ 9. -1.  1.  0.  0.]
 [ 3. -1.  1.  0.  0.]
 [ 9. -1.  1.  0.  0.]
 [-1. -1.  1.  1.  1.]
 [ 1. -1. -1. -1. -1.]]

random coordinate
(3, 0)
[[ 9. -1.  1.  0.  0.]
 [ 3. -1.  1.  0.  0.]
 [ 9. -1.  1.  0.  0.]
 [ 2. -1.  1.  1.  1.]
 [ 1. -1. -1. -1. -1.]]

random coordinate
(4, 2)
[[ 9. -1.  1.  0.  0.]
 [ 3. -1.  1.  0.  0.]
 [ 9. -1.  1.  0.  0.]
 [ 2. -1.  1.  1.  1.]
 [ 1. -1.  1. -1. -1.]]

random coordinate
(4, 4)
[[ 9. -1.  1.  0.  0.]
 [ 3. -1.  1.  0.  0.]
 [ 9. -1.  1.  0.  0.]
 [ 2. -1.  1.  1.  1.]
 [ 1. -1.  1. -1.  9.]]

(4, 3)
[[ 9. -1.  1.  0.  0.]
 [ 3. -1.  1.  0.  0.]
 [ 9. -1.  1.  0.  0.]
 [ 2. -1.  1.  1.  1.]
 [ 1. -1.  1.  1.  9.]]

random coordinate
(0, 1)
[[ 9.  9.  1.  0.  0.]
 [ 3. -1.  1.  0.  0.]
 [ 9. -1.  1.  0.  0.]
 [ 2. -1.  1.  1.  1.]
 [ 1. -1.  1.  1.  9.]]

(2, 1)
[[ 9.  9.  1.  0.  0.]
 [ 3. -1.  1.  0.  0.]
 [ 9.  2.  1.  0.  0.]
 [ 2. -1.  1.  1.  1.]
 [ 1. -1.  1.  1.  9.]]

(1, 1)
[[ 9.  9.  1.  0.  0.]
 [ 3.  3.  1.  0.  0.]
 [ 9.  2.  1.  0.  0.]
 [ 2. 10.  1.  1.  1.]
 [ 1. -1.  1.  1.  9.]]

random coordinate
(4, 1)
[[ 9.  9.  1.  0.  0.]
 [ 3.  3.  1.  0.  0.]
 [ 9.  2.  1.  0.  0.]
 [ 2. 10.  1.  1.  1.]
 [ 1.  1.  1.  1.  9.]]
```

The following is the advanced agent playing the game.



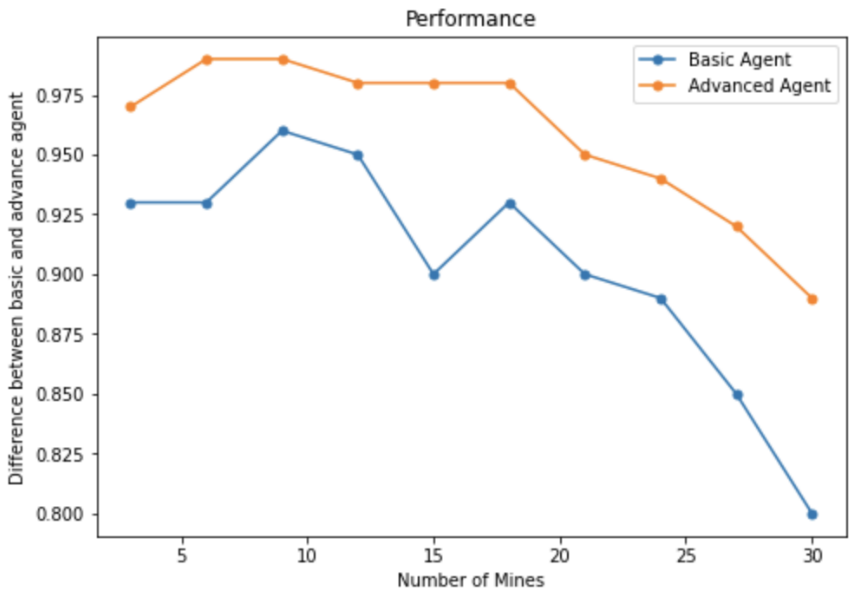
```
random coordinate
(0, 0)
[[ 9. -1. -1. -1. -1.]
 [-1. -1. -1. -1. -1.]
 [-1. -1. -1. -1. -1.]
 [-1. -1. -1. -1. -1.]
 [-1. -1. -1. -1. -1.]]

random coordinate
(0, 2)
[[ 9. -1.  1. -1. -1.]
 [-1. -1. -1. -1. -1.]
 [-1. -1. -1. -1. -1.]
 [-1. -1. -1. -1. -1.]
 [-1. -1. -1. -1. -1.]]

(0, 4)
[[ 9. -1.  1. -1.  0.]
 [-1. -1. -1. -1. -1.]
 [-1. -1. -1. -1. -1.]
 [-1. -1. -1. -1. -1.]
 [-1. -1. -1. -1. -1.]]

(1, 3)
[[ 9. -1.  1. -1.  0.]
 [-1. -1. -1.  0. -1.]
 [-1. -1. -1. -1. -1.]
 [-1. -1. -1. -1. -1.]
 [-1. -1. -1. -1. -1.]]

(2, 2)
[[ 9. -1.  1. -1.  0.]
 [-1. -1. -1.  0. -1.]
 [-1. -1.  1. -1. -1.]
 [-1. -1. -1. -1. -1.]
 [-1. -1. -1. -1. -1.]]

(2, 4)
[[ 9. -1.  1. -1.  0.]
 [-1. -1. -1.  0. -1.]
 [-1. -1.  1. -1.  0.]
 [-1. -1. -1. -1. -1.]
 [-1. -1. -1. -1. -1.]]

(3, 3)
[[ 9. -1.  1. -1.  0.]
 [-1. -1. -1.  0. -1.]
 [-1. -1.  1. -1.  0.]
 [-1. -1. -1.  1. -1.]
 [-1. -1. -1. -1. -1.]]

(3, 4)
[[ 9. -1.  1. -1.  0.]
 [-1. -1. -1.  0. -1.]
 [-1. -1.  1. -1.  0.]
 [-1. -1. -1.  1.  1.]
 [-1. -1. -1. -1. -1.]]

(1, 2)
[[ 9. -1.  1. -1.  0.]
 [-1. -1.  1.  0. -1.]
 [-1. -1.  1. -1.  0.]
 [-1. -1. -1.  1.  1.]
 [-1. -1. -1. -1. -1.]]

(2, 3)
[[ 9. -1.  1. -1.  0.]
 [-1. -1.  1.  0. -1.]
 [-1. -1.  1.  0.  0.]
 [-1. -1. -1.  1.  1.]
 [-1. -1. -1. -1. -1.]]

(3, 2)
[[ 9. -1.  1. -1.  0.]
 [-1. -1.  1.  0. -1.]
 [-1. -1.  1.  0.  0.]
 [-1. -1.  1.  1.  1.]
 [-1. -1. -1. -1. -1.]]

(0, 3)
[[ 9. -1.  1.  0.  0.]
 [-1. -1.  1.  0. -1.]
 [-1. -1.  1.  0.  0.]
 [-1. -1.  1.  1.  1.]
 [-1. -1. -1. -1. -1.]]

(1, 4)
[[ 9. -1.  1.  0.  0.]
 [-1. -1.  1.  0.  0.]
 [-1. -1.  1.  0.  0.]
 [-1. -1.  1.  1.  1.]
 [-1. -1. -1. -1. -1.]]

(2, 1)
[[ 9. -1.  1.  0.  0.]
 [-1. -1.  1.  0.  0.]
 [-1.  2.  1.  0.  0.]
 [-1. -1.  1.  1.  1.]
 [-1. -1. -1. -1. -1.]]

(4, 2)
[[ 9. -1.  1.  0.  0.]
 [-1. -1.  1.  0.  0.]
 [-1.  2.  1.  0.  0.]
 [-1. -1.  1.  1.  1.]
 [-1. -1.  1. -1. -1.]]

random coordinate
(4, 0)
[[ 9. -1.  1.  0.  0.]
 [-1. -1.  1.  0.  0.]
 [-1.  2.  1.  0.  0.]
 [-1. -1.  1.  1.  1.]
 [ 1. -1.  1. -1. -1.]]

random coordinate
(0, 1)
[[ 9.  9.  1.  0.  0.]
 [-1. -1.  1.  0.  0.]
 [-1.  2.  1.  0.  0.]
 [-1. -1.  1.  1.  1.]
 [ 1. -1.  1. -1. -1.]]

(1, 1)
[[ 9.  9.  1.  0.  0.]
 [-1.  3.  1.  0.  0.]
 [-1.  2.  1.  0.  0.]
 [-1. 10.  1.  1.  1.]
 [ 1. -1.  1. -1. 10.]]

(4, 1)
[[ 9.  9.  1.  0.  0.]
 [-1.  3.  1.  0.  0.]
 [-1.  2.  1.  0.  0.]
 [-1. 10.  1.  1.  1.]
 [ 1.  1.  1. -1. 10.]]

(3, 0)
[[ 9.  9.  1.  0.  0.]
 [-1.  3.  1.  0.  0.]
 [-1.  2.  1.  0.  0.]
 [ 2. 10.  1.  1.  1.]
 [ 1.  1.  1. -1. 10.]]

(4, 3)
[[ 9.  9.  1.  0.  0.]
 [-1.  3.  1.  0.  0.]
 [10.  2.  1.  0.  0.]
 [ 2. 10.  1.  1.  1.]
 [ 1.  1.  1.  1. 10.]]

random coordinate
(1, 0)
[[ 9.  9.  1.  0.  0.]
 [ 3.  3.  1.  0.  0.]
 [10.  2.  1.  0.  0.]
 [ 2. 10.  1.  1.  1.]
 [ 1.  1.  1.  1. 10.]]
```

We can see that the advanced agent identifies more mines than the basic agent. Up until the point (1,4), the basic agent and advanced agent play the same way since they are going through the neighbors of coordinates that have 0 as their clue. After (1,4) is selected, the basic agent chooses a random coordinate while the advanced agent explores (2,1) and (4,2) because it can identify that these coordinates are safe with the current active clues. For example, the advanced agent recognizes that since either (0,1) or (1,1) is a mine, (2,1) must be safe.

The advanced agent flagging (4,2) as safe was initially surprising but it is able to do so because it looks through all active clues. Since (3,4), (3,3), and (3,2) need a neighboring mine, (4,3) or (4,4) must be a mine and (4,2) must be safe. If (4,2) was a mine then the clue for (3,3) would have to be a 2.

The basic agent picks the random coordinate (0,1) towards the end of the game, but if we were playing the game, we would not choose this coordinate to explore next since it is a neighbor of a cell with a clue of 3.

5. Performance



The above graph depicts the performance of the advanced agent and the basic agent. We can see that as the mine density increases, the performance of both basic agent and advanced agent decreases. This trend makes sense since with a larger mine density there is a greater chance of randomly picking a mine. If there are more mines and less clues, it becomes harder to identify which cells are safe and which cells are mines. We also see that the advanced agent always performs better than the basic agent. This also makes sense since the advanced agent uses clues globally and can discover more while the basic agent just uses clues locally.

6. Efficiency

The main space constraint for this project was keeping track of the knowledge base. We heavily utilized neighboring hidden cells when solving for our advanced agent, and to keep track of coordinates, we used a data frame. This data frame continued to grow in size, as we added more and more hidden neighbors of which we could not identify anything about. This is a problem-specific constraint, as solving minesweeper in an advanced way requires a lot of bookkeeping to see how multiple constraints can be satisfied.

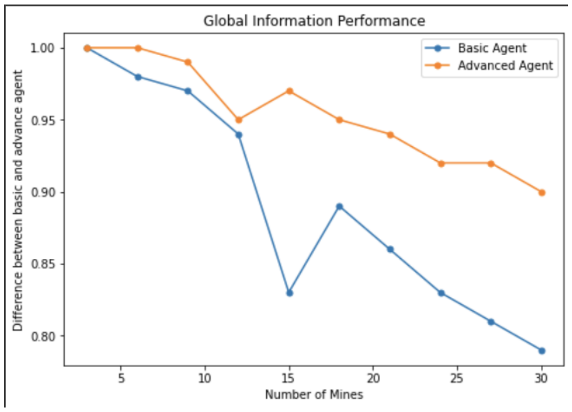
With time, we did run into some issues with other algorithms that we attempted. This was primarily of an implementation-specific issue, as we were continuously testing all the possible binary combinations for all the hidden neighboring cells in our knowledge base simultaneously. This number kept getting exponentially bigger as our code ran, taking an unnecessary long amount of time. Because of that issue, we had to change our original algorithm to what it is now, and it is definitely a lot faster than before. We could still further improve upon this by not testing as many combinations once we know for sure that certain patterns won't work.

7. Clever Acronym

basic agent - SCMD (single clue mine detector)
advanced agent - MCMD (multiple clue mine detector)

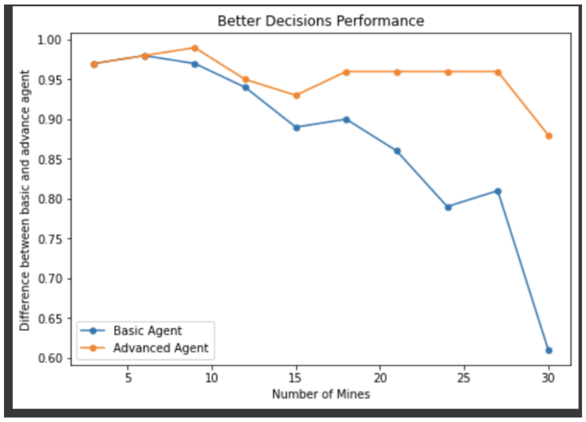
8. Global Information

If we know how many mines there are on the board before starting the game, we can use this number to break out of the loop earlier. We keep a list of all mines (exploded and flagged in a list). If the length of this list is equal to the number of mines, we can stop the game and open all remaining unexplored cells since we can conclude that they are safe. Additionally, if the length of the unexplored coordinates is equal to total mines - the mines flagged/exploded, we know that all remaining hidden coordinates are mines and can stop the game early.



9. Better Decisions

Our better random selection method chooses random cells by first checking to see if any open cells surround any random squares. If a certain cell does not have any open neighbors, then that random cell is valid to click on. However, if that is not the case, we total the number of remaining mines that the neighboring cells have and pick the random cell that has the least sum value. This essentially means that this cell is least likely to be a mine. We essentially use a probability based approach here to pick our random coordinate better.


Better Decisions Performance
Basic Agent
Advanced Agent
Difference between basic and advance agent
Number of Mines