AI Project 2 : Minesweeper

Name: Leena Dighole

Goal: Minesweeper game using AI agent

AI agents:

The minesweeper game is solved using 4 different agents. The project consists of the analysis of the performances of these agents.

- Simple Agent
- Simple Agent with additional knowledge of total number of mines known
- Improved Agent
- Improved Agent with additional knowledge of total number of mines known

Goal: Try to identify maximum no of mine cells for the given mine environment

Simple Agent Algorithm:

The Simple Agent performs following Operations:

1. Select any start cell randomly
2. If the cell is mine, It will add it in no of mines found and keep searching other cells
3. If cell is safe, It will show its clue value (No of mines in its neighborhood)
4. If the clue value = total no of neighbor cells => All neighbors are mines, Add it into KB
5. If the clue value = 0 => All neighbors are safe, Add it into KB
6. If the clue value < total no of neighbor cells => Select the cell randomly

Improved Agent Algorithm:

The Improved Agent performs following Operations:

1. Select any start cell randomly
2. If the cell is mine, It will add it in no of mines found and keep searching other cells
3. If cell is safe, It will show its clue value (No of mines in its neighborhood)
4. If the clue value = total no of neighbor cells => All neighbors are mines, Add it into KB
5. If the clue value = 0 => All neighbors are safe, Add it into KB
6. If the clue value < total no of neighbor cells => Perform some operations on the available KB
7. Collect new inferences with some operations and add it into KB
8. If no new inference is obtained by new operations, Select new cell to be surfed randomly

Bonus Question:

Note that the Simple agent and Improved agent with no of mines known works with the additional piece of information of total no of mines on the board. The significant differences in then performances are analyzed in the later part of the report.

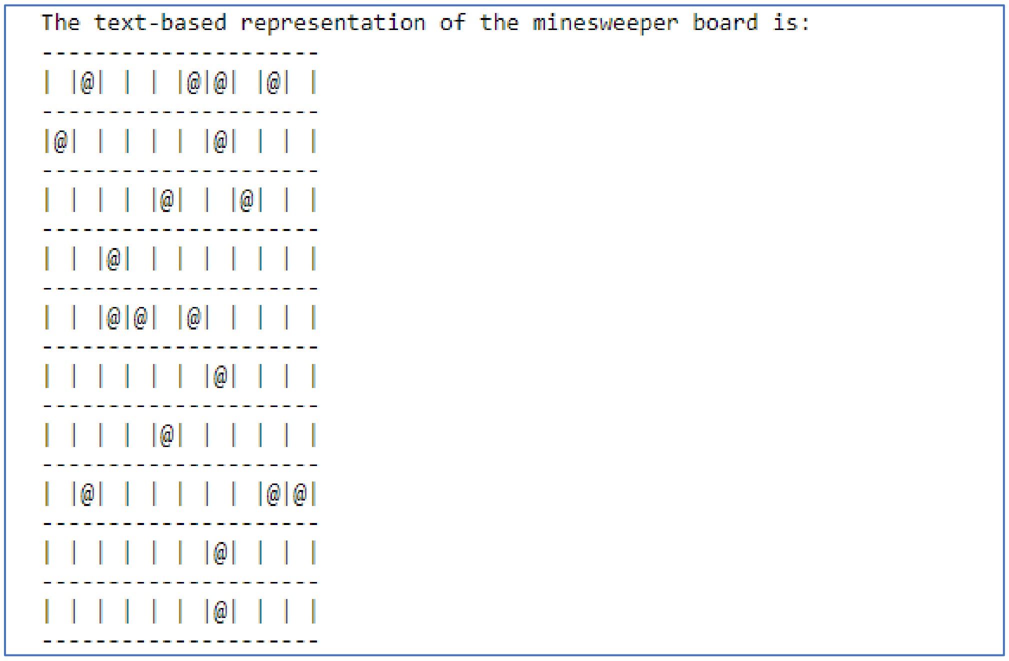
Representation of the mine environment:

Mine Environment

Goal: Mine Sweeper board representation

Algorithm:

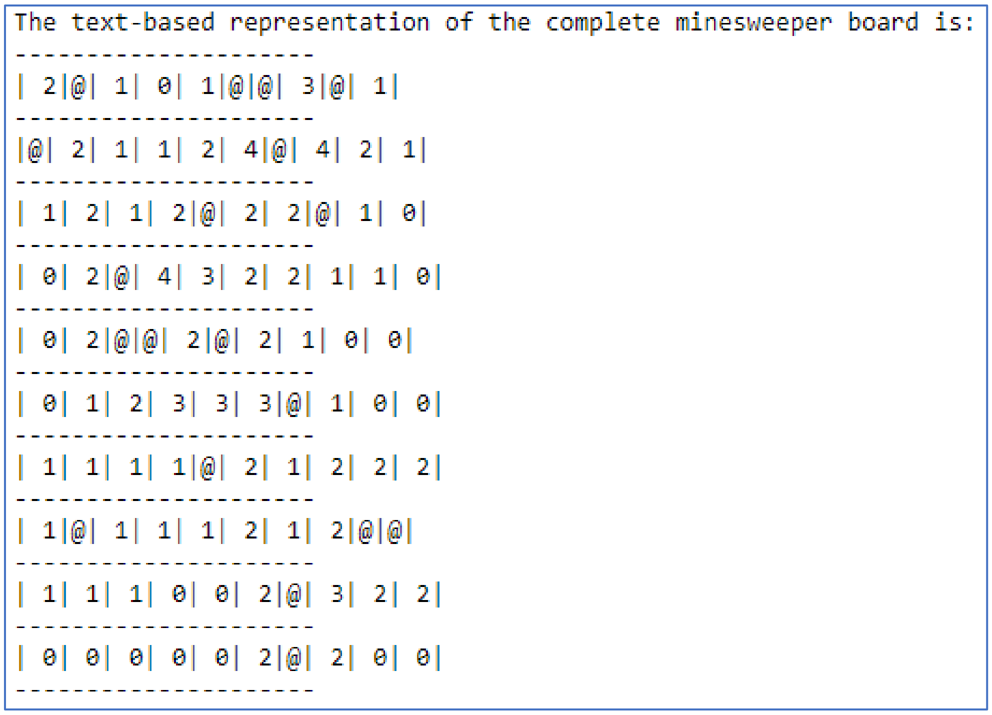
1. Create Board with given dimension and mine density values
2. Print text-Based Representation
3. Check whether cell is safe/ mine
4. calculate Clue value of the safe cell
5. calculate no of hidden squares



```
The text-based representation of the minesweeper board is:
---------------------
| |@| | | |@|@| |@| |
---------------------
|@| | | | | |@| | | |
---------------------
| | | | |@| | |@| | |
---------------------
| | |@| | | | | | | |
---------------------
| | |@|@| |@| | | | |
---------------------
| | | | | | |@| | | |
---------------------
| | | | |@| | | | | |
---------------------
| |@| | | | | | |@|@|
---------------------
| | | | | | |@| | | |
---------------------
| | | | | | |@| | | |
---------------------
```

Figure 1: Mine board representation for d = 10, n=0.2

The text-based representation is used for the mine sweeper board game. The mines are represented by '@' symbol. The clue- mine representation for d = 10 and mine density = 0.2 will look like,



```
The text-based representation of the complete minesweeper board is:
---------------------
| 2|@| 1| 0| 1|@|@| 3|@| 1|
---------------------
|@| 2| 1| 1| 2| 4|@| 4| 2| 1|
---------------------
| 1| 2| 1| 2|@| 2| 2|@| 1| 0|
---------------------
| 0| 2|@| 4| 3| 2| 2| 1| 1| 0|
---------------------
| 0| 2|@|@| 2|@| 2| 1| 0| 0|
---------------------
| 0| 1| 2| 3| 3| 3|@| 1| 0| 0|
---------------------
| 1| 1| 1| 1|@| 2| 1| 2| 2| 2|
---------------------
| 1|@| 1| 1| 1| 2| 1| 2|@|@|
---------------------
| 1| 1| 1| 0| 0| 2|@| 3| 2| 2|
---------------------
| 0| 0| 0| 0| 0| 2|@| 2| 0| 0|
---------------------
```

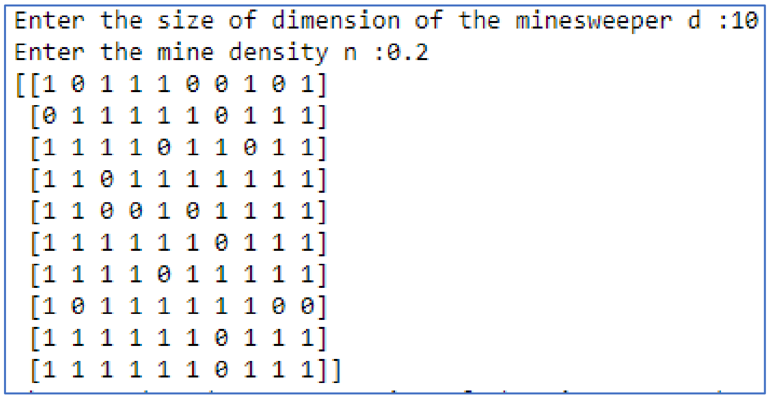
Figure 2:Mine- Clue board representation for d=10, n=0.2

In above figure, the integers represent the clue value of the safe cells which is the total no of mines around that particular safe cell. ‘@’represents mines. In above example with d =10 and n=0.5, we have 19 mines in the board with 100 cells.

The mines with given mine density (n) are plotted randomly in the d*d dimension board. The actual value based representation of the board has ,

1 = safe cell

0= mine cell



```
Enter the size of dimension of the minesweeper d :10
Enter the mine density n :0.2
[[1 0 1 1 1 0 0 1 0 1]
 [0 1 1 1 1 1 0 1 1 1]
 [1 1 1 1 0 1 1 0 1 1]
 [1 1 0 1 1 1 1 1 1 1]
 [1 1 0 0 1 0 1 1 1 1]
 [1 1 1 1 1 1 0 1 1 1]
 [1 1 1 1 0 1 1 1 1 1]
 [1 0 1 1 1 1 1 1 0 0]
 [1 1 1 1 1 1 0 1 1 1]
 [1 1 1 1 1 1 0 1 1 1]]
```

Figure 3: The actual value representation of the board

Inferences and Decisions

Inferences are drawn based upon the current state of the agent and already explored cells. The data structure –'Sets' in python are used to store the values of new equations and corresponding clue value of the cell. The Inferences for 2 different agents are briefly discussed below:

Simple Agent Inferences

Algorithm:

1. Check whether clue value of cell == no of neighbors => all neighbors are mines
2. Check whether clue value of cell == 0 => all neighbors are safe
3. Check whether clue value of cell < no of neighbors => Pick a random cell for next move

Case scenarios:

The operations performed by simple agent involves updating the KB with knowledge of cell value of a newly surfed cell. Simple agent performs basic inferences.

If a cell has 4 neighbors and its clue value is 8 then it can immediately infer that all neighboring cells are mines. This new piece of information will get updated into the KB.

If A+B+C+D =4, Then A=B=C=D =1 : All are mines

Similarly if a cell has clue value 0 then Simple agent can conclude that all neighboring cells are mines. This new piece of information will get updated into the KB.

If A+B+C =0, Then A=B=C =0 : All are safe

The random cell Is chosen for the next moves if it can not infer any new information.

Improved Agent Inferences

Algorithm:

1. Check whether clue value of cell == no of neighbors => all neighbors are mines
2. Check whether clue value of cell == 0 => all neighbors are safe
3. Check whether clue value of cell < no of neighbors => Perform subtraction operations and add new inferences in KB

Case scenarios:

The operations performed by improved agent involves updating the KB with knowledge of cell value of a newly surfed cell. Improved agent performs basic inferences along with some additional operations.

If a cell has 8 neighbors and its clue value is 8 then it can immediately infer that all neighboring cells are mines. This new piece of information will get updated into the KB.

If A+B+C+D +E+F=6, Then A=B=C=D=E=F =1 : All are mines

Similarly if a cell has clue value 0 then Simple agent can conclude that all neighboring cells are mines. This new piece of information will get updated into the KB.

If $A+B+C+D=0$, Then $A=B=C=D=0$: All are safe

If the number of neighbors is not equal to the clue value then it will try to find the equations with some common cells and try to perform subtraction operations to infer new piece of information.

For example, lets consider following equations,

$A+B+C = 2$

$A+D+C = 1$

Then Improved agent will perform subtraction operation on both equations to check whether we get any new piece of information.

Thus we get $B-D = 1$

Now we can infer that $B = 1$ and $D = 0$ and update our KB using this new discovered cell values.

Decisions

The Inferences involved based upon the equations collected using the clue value and neighboring cells of the cell. The agents performing inferences can try to find whether particular cell is a safe/ mine based upon current KB. But agent needs to take some decisions based upon the current state of the search space. Even if a cell is supposed to be chosen randomly but should it be a immediate random neighbor cell or any random cell, the agent have to make some decisions based upon its own clue value.

The agent can face 5 different cases while deciding any random move:

1. Current cell is a safe cell and has non zero clue value
2. Current cell is a mine
3. If all the neighbors of the cell are mine
4. If all the neighbors of the cell are safe
5. If some neighbors are mines and some are safe

Decision Algorithm:

Start the game for Agent

- start at random cell
- Keep exploring the safe nodes if we have any new inferences
- Else choose a random cell and explore further

If a random cell is to be choosen then agent will decide it based upon its current state.

Decisions according to agents Knowledge:

1. All the neighboring cells of a cell are safe so choose any random neighbor cell for next step
2. All the neighboring cells of a cell are mines so choose any random cell from the board which is not previously explored for next step
3. Agent is unable to decide whether all neighbors are safe/ mines so choose any random neighbor cell for next step
   Note: As the cell has some safe and mine neighbors, it is possible that a randomly chosen cell may or may not be a safe cell. If the mine cell is detected, the KB gets updated with the value and game still get continued until all the mines on the board are searches correctly.
4. If the cell is a mine so choose any random neighbor cell for next step
   Note: The cell is a mine thus it has no clue value and agent can't infer anything regarding its neighbors So choosing a random neighbor makes a sensible move here.
5. Agent can find all mines on the board so terminate the game and try to calculate the final score
   Note: Final score = (safely identified mines / total mines)

Performances:

<u>Case scenario: d =10</u>

Using d = 10 ,the 10*10 minesweeper board game is created and the total no of steps and final score graphs were plot for mine density ranging from n = 0.1 ,0.2...0.9

Note that we will analyze all 3 graphs for all 4 agents.

1. Total no of steps Vs Mine density
2. Total Savings in steps Vs Mine density
3. Final Score Vs Mine density

1.Simple Agent:

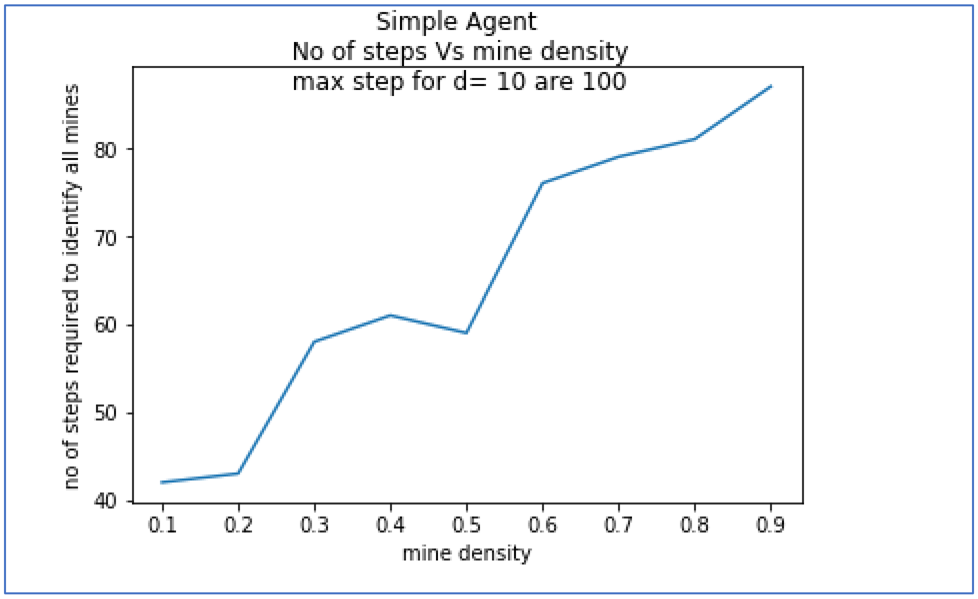


Figure 4:Simple agent graph1

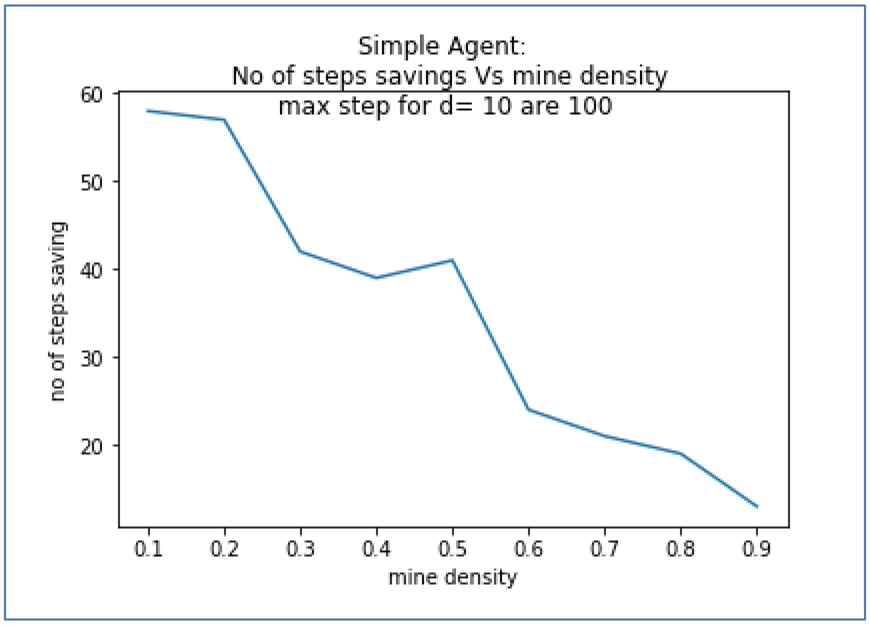


Figure 5:Simple Agent graph 2

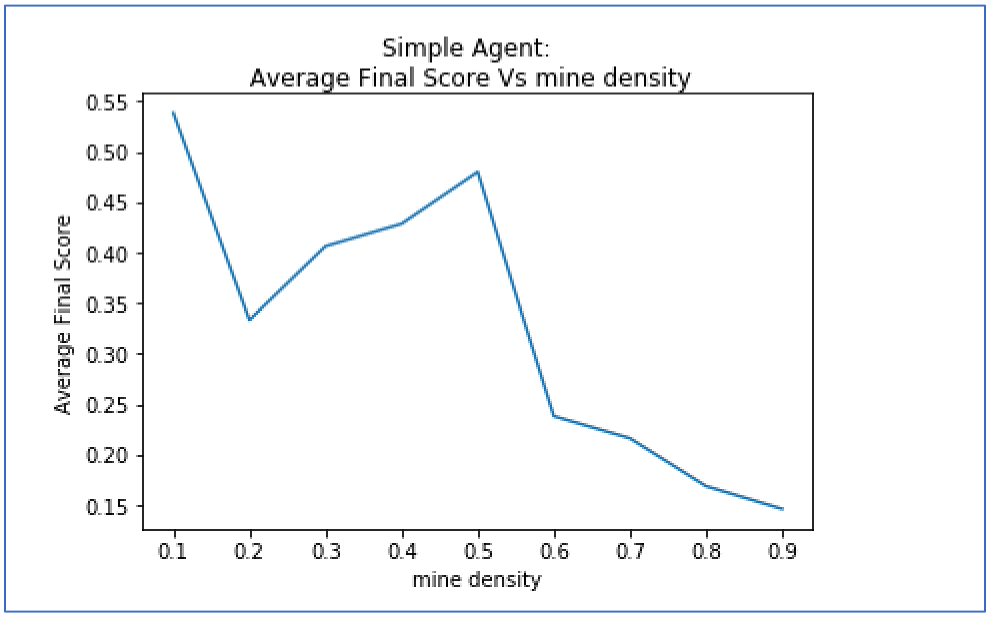


Figure 6:Simple agent Graph 3

Observations:

- We can observe that as the mine density increase, the total number of steps required for Simple agent to identify all the mines of the board also increases.
- The number of steps savings decreases as we increase the mine density.

- The average final score for n =0.1 is high compared to other mine densities as the risk to randomly choose a cell which is a mine is lower. Thus the agent is able to find the maximum mines before stepping on the random cell which is a mine.
- The average final score increases from 0.2 to 0.5 which seems reasonable as number of mine density increases the number of mines increases and safely detected mine number also increases.
- From mine density 0.6 to 0.9 , the final score decreases which seems reasonable as the no of mines increases , the random move has high chances to be a mine cell and thus the final score decreases.
- At mine density = 0.5 , the peak is observed. As the board has equal no of safe and mine cells, the randomly chosen cell has equal probability of getting a mine and a safe cell. Thus the total final score is maximum with a sufficient amount of clue cells and low risk that a random move may end up in getting a mine.

2. Simple Agent with number of mines known:

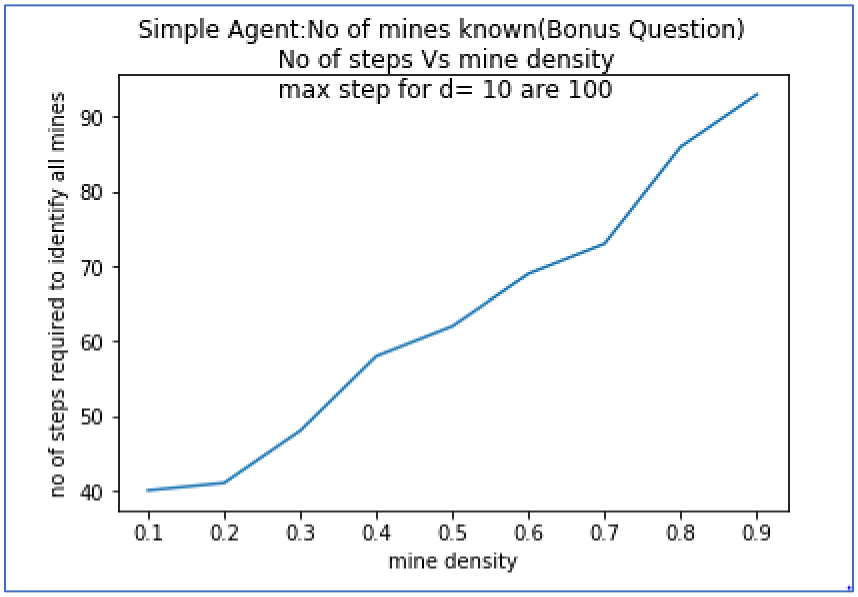


Figure 7:Simple Agent graph 1(Bonus)

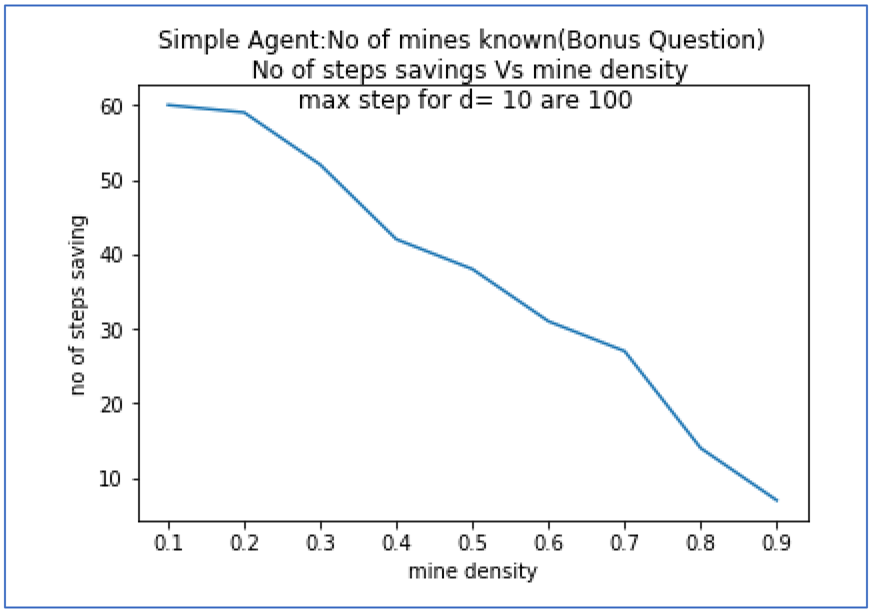


Figure 8:Simple Agent graph 2(Bonus)

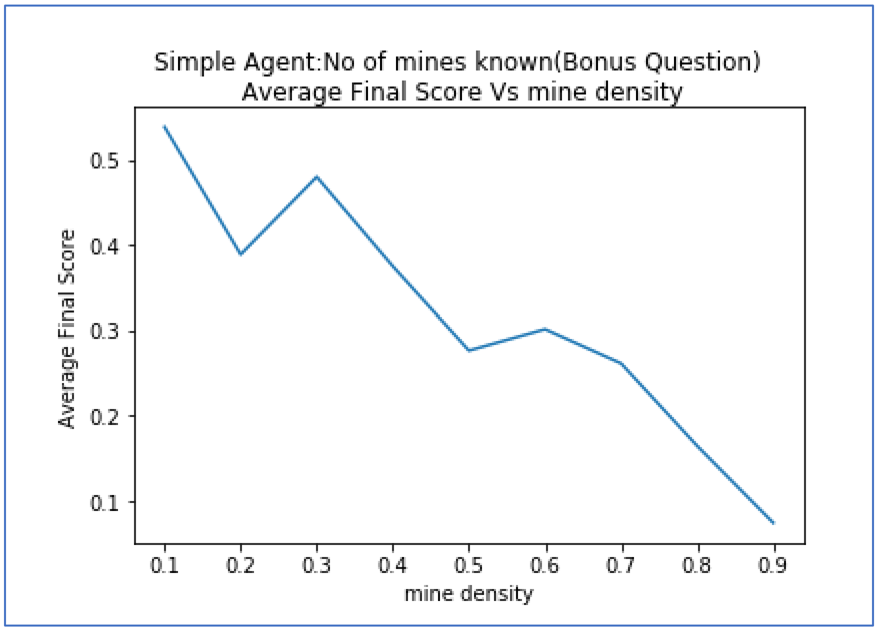


Figure 9:Simple Agent graph 3(Bonus)

Observations:

- We can observe that as the mine density increase, the total number of steps required for Simple agent knowing number of mines to identify all the mines of the board also increases.
- The number of steps savings decreases as we increase the mine density.

- The average final score for n =0.1 is high compared to other mine densities as the risk to randomly choose a cell which is a mine is lower. Thus the agent is able to find the maximum mines before stepping on the random cell which is a mine.
- The average final score increases from 0.2 to 0.3 which seems reasonable as number of mine density increases the number of mines increases and safely detected mine number also increases.
- From mine density 0.4 to 0.9 , the final score decreases which seems reasonable as the no of mines increases , the random move has high chances to be a mine cell and thus the final score decreases.

Its reasonable to have a question about the behavior of Simple agent Vs Simple agent with mines known. Lets see the graph below,

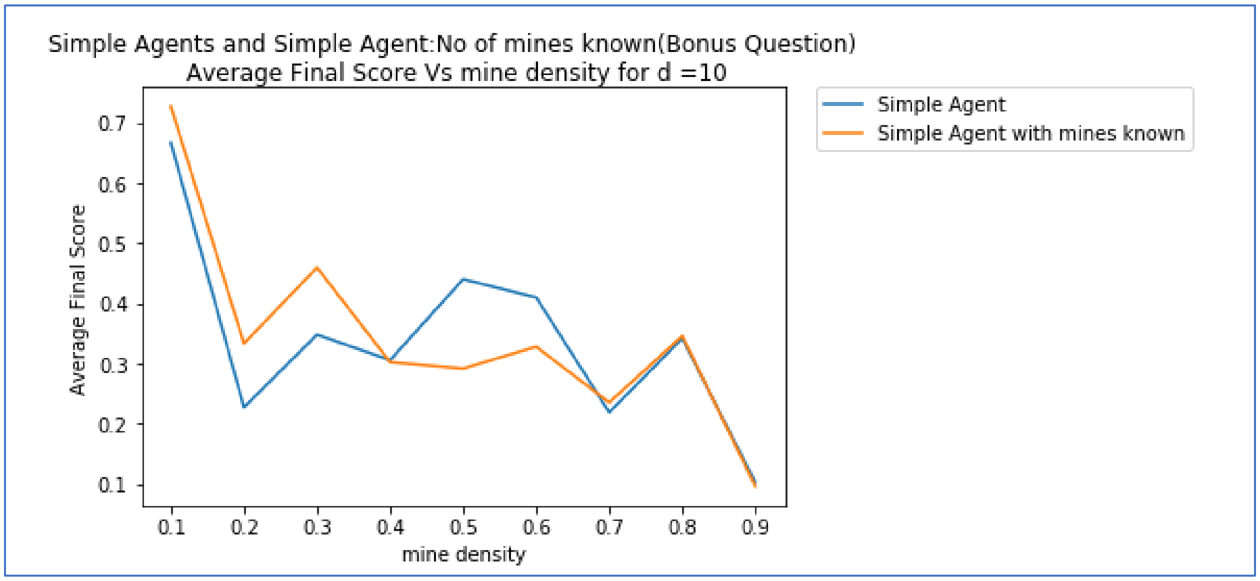


Figure 10:Simple Agent vs simple agent with mines known d =10

Intuitively one can argue that knowing total number of mines of the board must be useful and thus should always outperform simple agent without any knowledge of total no of mines.

But from above graph, we can see that its not the always case. The Simple agent do outperform the Simple agent with mines known knowledge for mine densities 0. 4 to 0.6.

For mine densities,

0.4-0.6 : The Simple agent outperforms Simple agent with mines known

0-0.3 and 0.7 to 0.9 : The Simple agent with additional knowledge of total no of mines outperforms the simple agent.

This shows that the knowledge of no of mines is not necessarily useful for all mine densities. It may not give us any useful information as we can see it in the graph for range 0.4 to 0.6.

We have also simulated the minesweeper game board for d = 20. We tried to cross check our analysis for d =10.

<u>Case scenario: d =20</u>

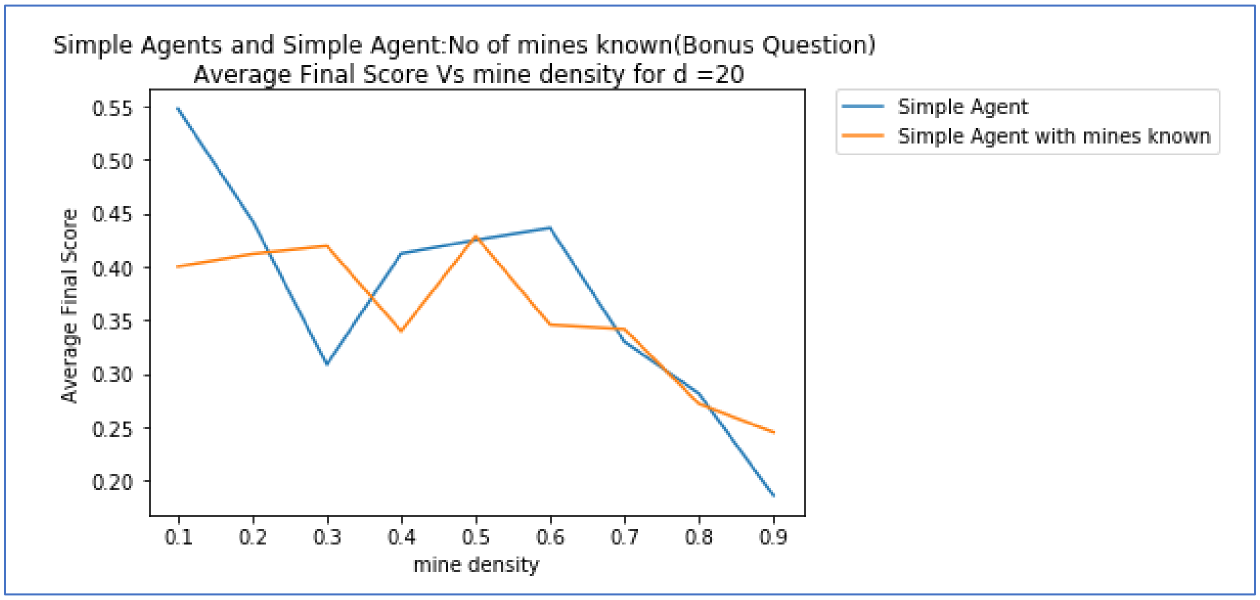


Figure 11: Simple Agent vs simple agent with mines known d =20

The above graph for d =20 gives us the result consistent with d=10. The simple agent outperforms simple agent with no of mines known for the range of the mine densities: 0.4 to 0.6.

The reason behind this phenomenon possibly can be the no of safe and mine cells. The board game with probably equal no of mine and safe cells find it equally likely to predict new random cell as mine or a safe cell. In this equal probability situation we no longer gain any new inference based upon knowing no of total mines.

Similarly for n = 0.4 and 0.6, the randomness of cell choice with most of the cells being equally likely as safe or mine doesn't give additional knowledge. Thus we can say that due to randomness of a cell choice amongst equally likely safe or a mine cell, the simple agent outperforms Simple agent with additional knowledge of total no of mines.

So we can say that knowing the total number of mines does not necessarily give additional knowledge to solve the game. **Thus Simple agent outperforms the simple agent with knowledge of total no of mines!**

Note that the simple agent is able to determine more than 50% mines correctly is a great achievement in terms of performances. The normal range of correctly identified mines is between 30-50 %. Thus we can say that the basic inference is capable enough for determining 30-40 % of total mines correctly.

We also need an improved agent with a intuitively better performance than the simple agent. This can be done using some operations on the current knowledge base of the agent.

3.Improved agent:

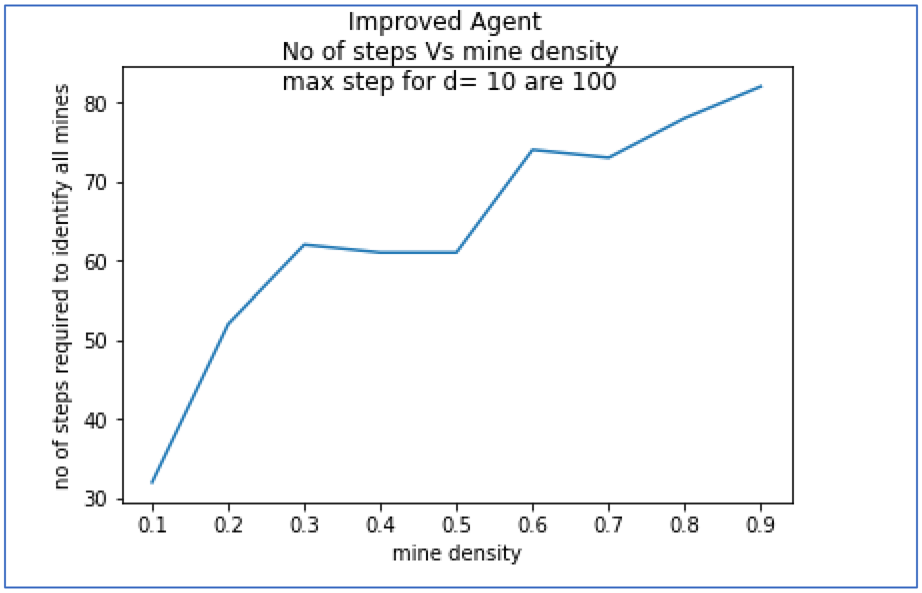


Figure 12:Improved agent graph 1 for d =10

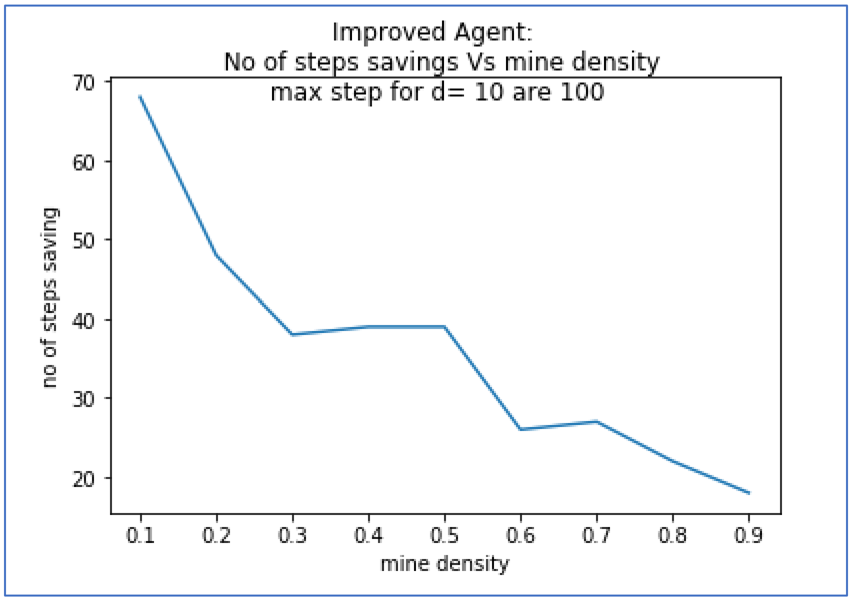


Figure 13:Improved agent graph 2 for d=10

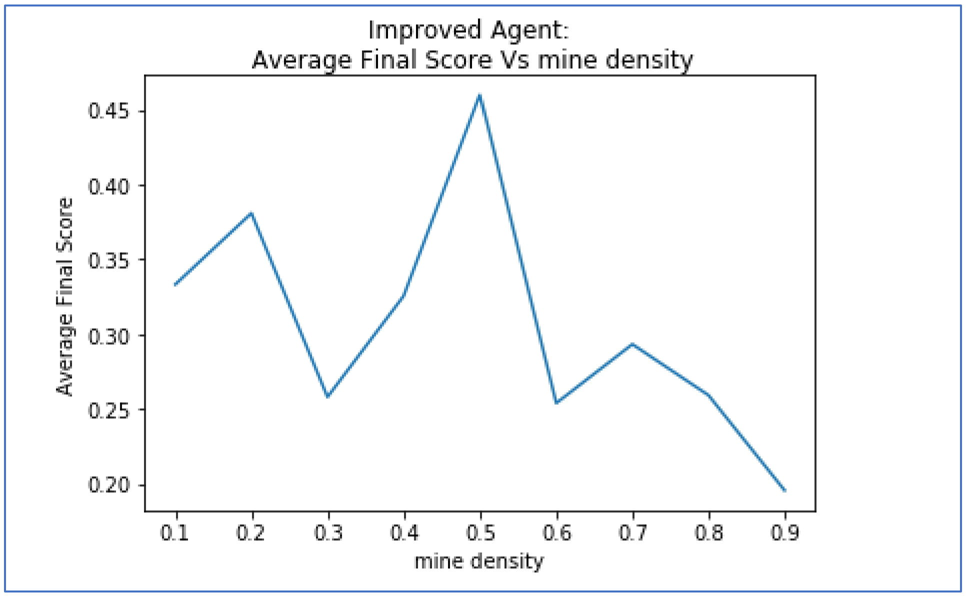


Figure 14:Improved Agent Graph 3 for d =10

Observations:

- We can observe that as the mine density increase, the total number of steps required for Improved agent to identify all the mines of the board also increases.
- The number of steps savings decreases as we increase the mine density.
- The average final score increases from 0.3 to 0.5 which seems reasonable as number of mine density increases the number of mines increases and safely detected mine number also increases.
- From mine density 0.6 to 0.9 , the final score decreases which seems reasonable as the no of mines increases , the random move has high chances to be a mine cell and thus the final score decreases.
- At mine density = 0.5 , the peak is observed. As the board has equal no of safe and mine cells, the randomly chosen cell has equal probability of getting a mine and a safe cell. Thus the total final score is maximum with a sufficient amount of clue cells and low risk that a random move may end up in getting a mine.

4.Improved agent with number of mines known:

Improved agent is having additional information about total no of mines in the game. The agent is using this key information to find out the total no of mines on the board.

Intuitively the improved agent must be able to find more no of mines safely compared to the basic inferences of a simple agent.

Lets analyze these results for cases with d=10 and d=20

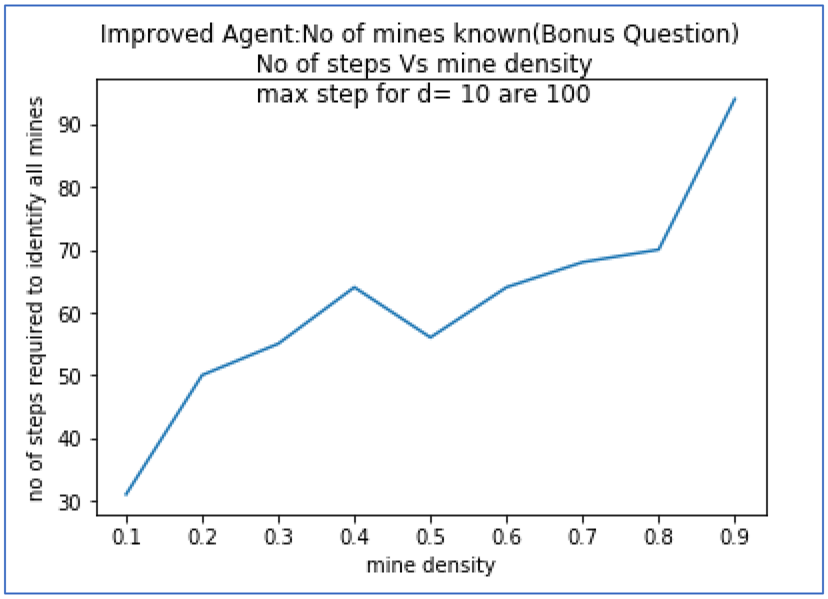


Figure 15:Improved agent (Bonus) graph 1, d=10

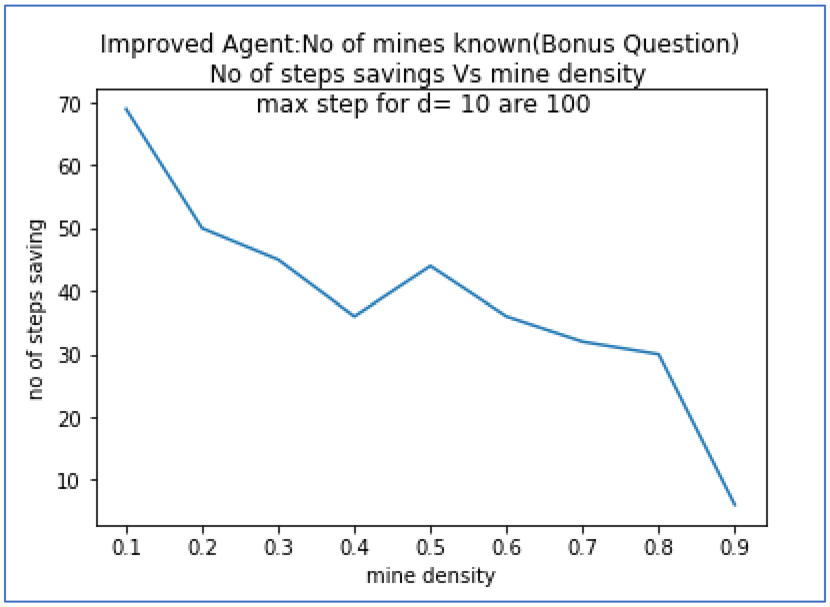


Figure 16: Improved agent (Bonus) graph 2, d=10

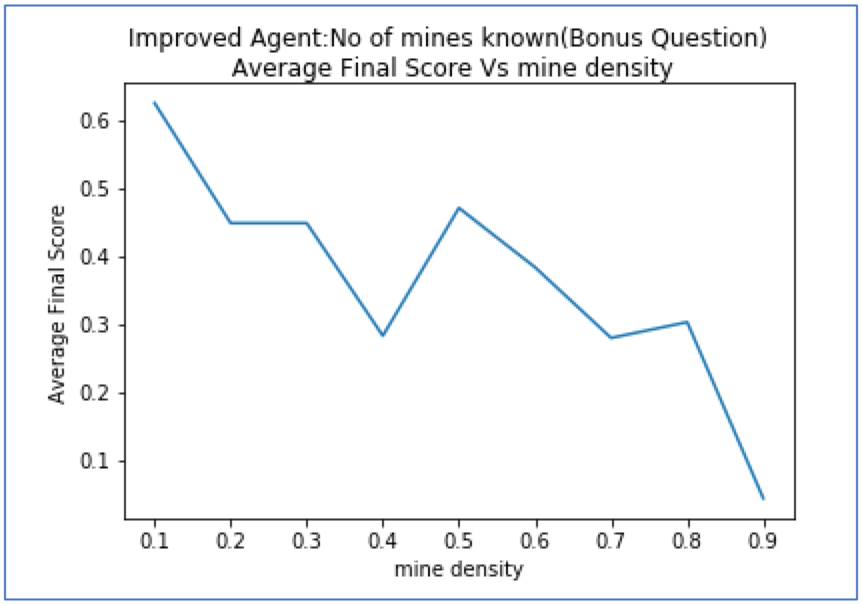


Figure 17: Improved agent (Bonus) graph 3, d=10

Observations:

- We can observe that as the mine density increase, the total number of steps required for Improved agent knowing number of mines to identify all the mines of the board also increases.
- The number of steps savings decreases as we increase the mine density.
- The average final score for n =0.1 is high compared to other mine densities as the risk to randomly choose a cell which is a mine is lower. Thus the agent is able to find the maximum mines before stepping on the random cell which is a mine. Its almost 66% Thus our improved agent is able to find 66% of the mines safely.
- The average final score decreases from 0.2 to 0.4 which seems reasonable as number of mine density increases the number of mines increases and chances of getting a randomly chosen cell as a mine also increases.
- From mine density 0.6 to 0.9 , the final score decreases which seems reasonable as the no of mines increases , the random move has high chances to be a mine cell and thus the final score decreases.
- At mine density = 0.5 , the peak is observed. As the board has equal no of safe and mine cells, the randomly chosen cell has equal probability of getting a mine and a safe cell. Thus the total final score is maximum with a sufficient amount of clue cells and low risk that a random move may end up in getting a mine.

Its reasonable to have a question about the behavior of Improved agent Vs Improved agent with mines known. Lets see the graph below,

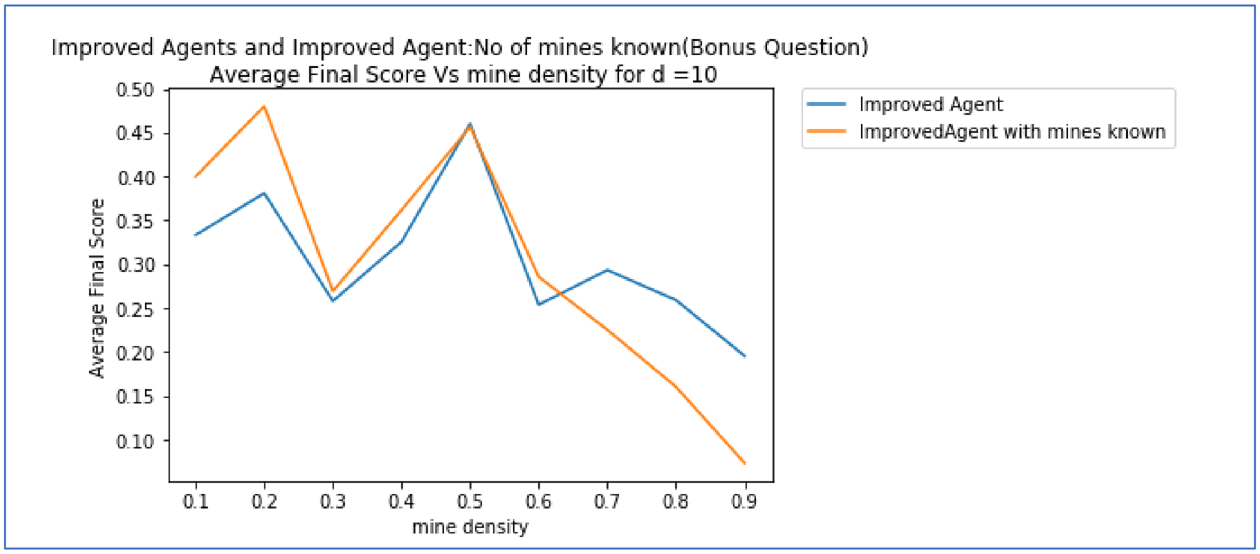


Figure 18: Simple Agent vs simple agent with mines known d =10

Intuitively one can argue that knowing total number of mines of the board must be useful and thus should always outperform Improved agent without any knowledge of total no of mines.

But from above graph, we can see that its not the always case. The Improved agent do outperform the Improved agent with mines known knowledge for mine densities 0. 7 to 0.9.

For mine densities,

0.7-0.9 : The Improved agent outperforms Improved agent with mines known

0-0.6 : The Improved agent with additional knowledge of total no of mines outperforms the Improved agent.

0.5: The peak is observed in both cases.

This shows that the knowledge of no of mines is not necessarily useful for all mine densities. It may not give us any useful information as we can see it in the graph for range 0.7 to 0.9.

The reason behind this phenomenon possibly can be the tradeoff between no of safe and mine cells. The board game with probably equal no of mine and safe cells find it equally likely to predict new random cell as mine or a safe cell. In this equal probability situation we get a peak.

Similarly for n = 0.7 and 0.9, the randomness of cell choice with most of the cells being mine is more than the no of safe cells and potentially doesn't give additional knowledge. Thus we can say that due to randomness of a cell choice amongst more likely a mine cell compared to a safe cell, the Improved agent outperforms Improved agent with additional knowledge of total no of mines.

There is still uncertainty about these results as we need to check whether this holds true in higher dimension values of d. The reason for these phenomenon is still not clear but tried to address one of the possible ways in above paragraph.

So we can say that knowing the total number of mines does not necessarily give additional knowledge to solve the game. **Thus Improved agent outperforms the Improved agent with knowledge of total no of mines!**

Lets analyze the performances of Simple agent Vs Improved agent:

Case 1: for d =10

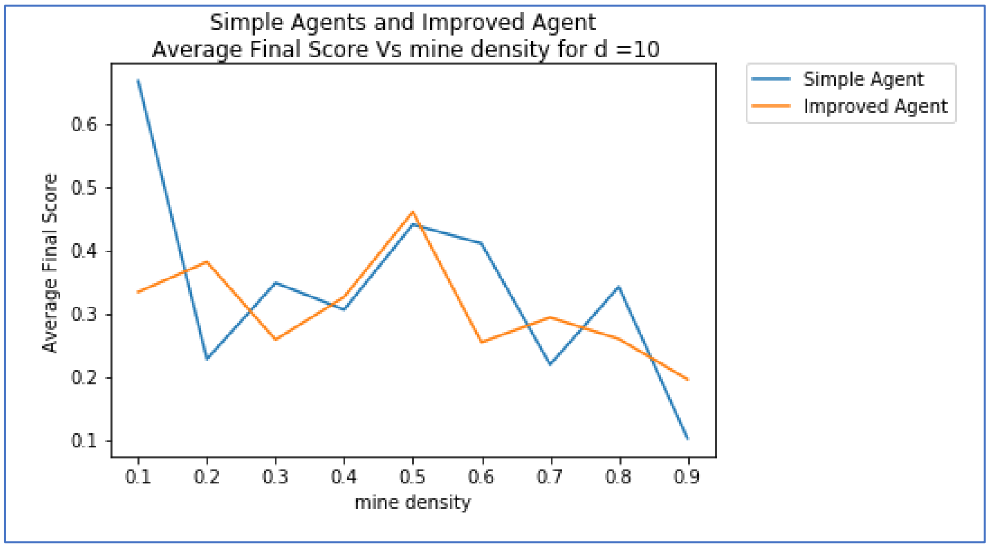


Figure 19: Simple agent Vs Improved agent for d=10

Our intuition is to check whether Improved agent always outperforms the simple agent. Also study the possible conditions where simple agent can perform better than Improved agent.

From the above graph for d=10, one can notice that Improved agent performs better than simple agent which is as expected in most of the cases. But there exists the cases where Simple agent outperforms Improved agent.

For mine density = 0.3,0.6,0.8 the Simple agent outperforms improved agent.

To verify whether the results holds in higher dimension, we simulated environment for d =20,

The results are surprising!

For n = 0.3, both d =10 and d=20 simple agents outperforms the corresponding improved agent.

For n = 0.5 to 0.7, the d=20 simple agent outperforms the improved agent.

These uncertainty increases the demand of testing the behavior in higher dimensions such as d=50 and so on.

The specific reason behind this behavior is under the water and a good scope of brainstorming.

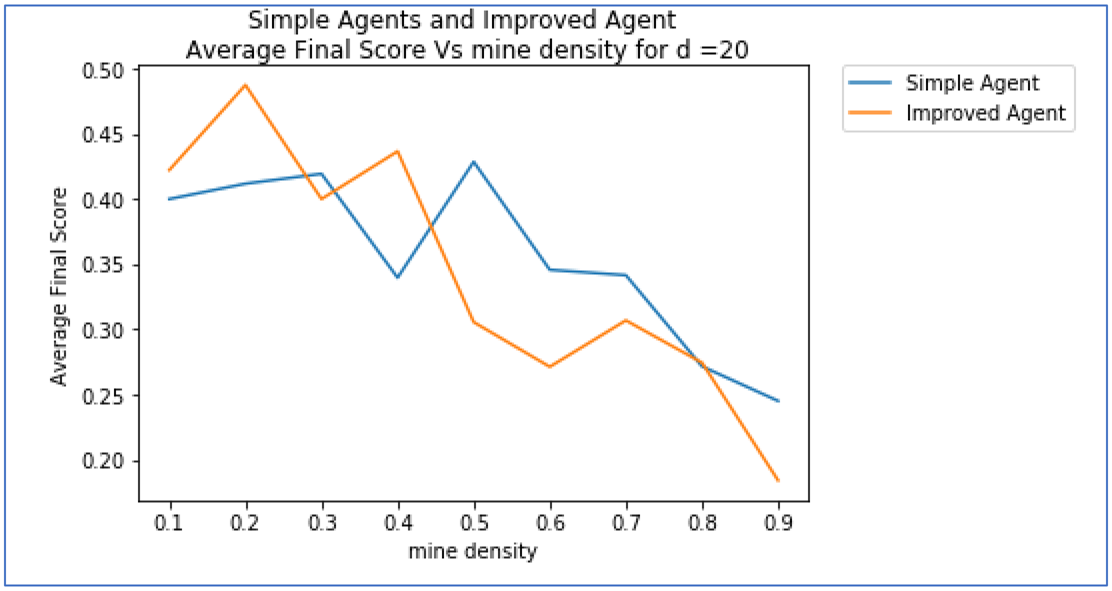


Figure 20: Simple agent Vs Improved agent for d=20

Lets see whether the Simple agent Vs Improved agent with additional information of no of mines known performs different than above cases.

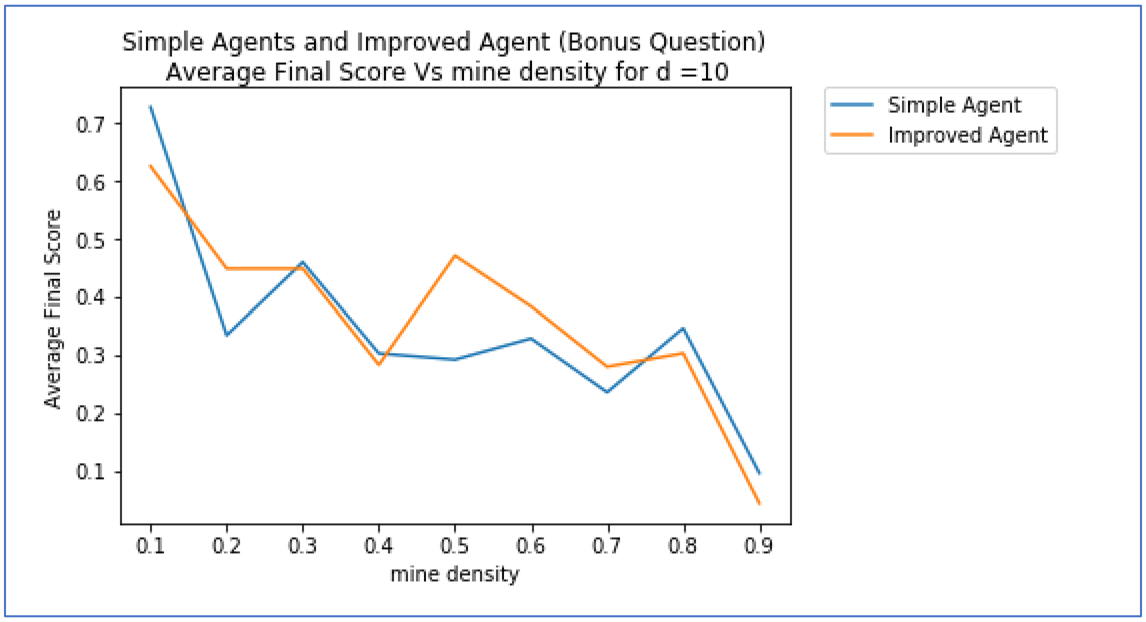


Figure 21: Simple agent Vs Improved agent for d=10

For mine density 0.3 and 0.8,the simple agent outperforms the improved agent given a knowledge of total no of mines present on the board. This behavior is inline with the Simple agent Vs Improved agent without any additional information for d=10.

Still the question remain same whether the same behavior can be observed in the higher values of d and if not then what can be the possible reasons for the same.

Thus we tried to analyze the results of following cases for d =10 and 20:

1. Simple Agent
2. Improved agent
3. Simple agent with no of mines known
4. Improved agent with no of mines known
5. Simple agent Vs Simple agent with no of mines known
6. Improved agent Vs Improved agent with no of mines known
7. Simple agent Vs Improved agent
8. Simple agent Vs Improved agent (no of mines known)

Note: I am currently working on the d =50 simulation for above 8 cases. Due to total no of cells =2500, the computational time for d=50 is relatively high and time consuming. The most of the uncertain questions can get addressed if we have its performance analysis. I will update the results once done with simulation for d=50.

The d=20 took almost 7 hours of simulation to get the graph of final score for different mine densities.

New Performance analysis and observations:

The following observations are based for d=10, calculated average based upon 100 iterations.

Kindly consider these new versions of the analysis for your reference!

Lets observe the number of steps required for agents Vs mine densities.

For all agents, as the mine density increases, the total number of steps required to detect all mines successfully also increases. Intuitively this result is as expected.

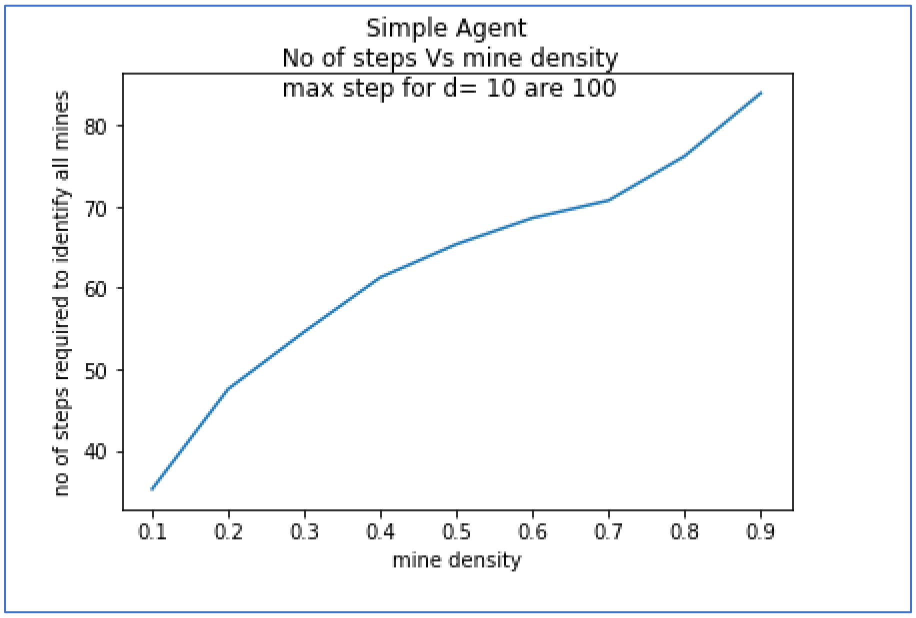


Figure22: Total no of steps Vs mine density

For d=10, total no of cells are 100. The steps savings can be calculated as the difference between total cells and calculated average of steps for 100 times iterations. As the total search steps are increasing, the total savings decreases as mine density increases.

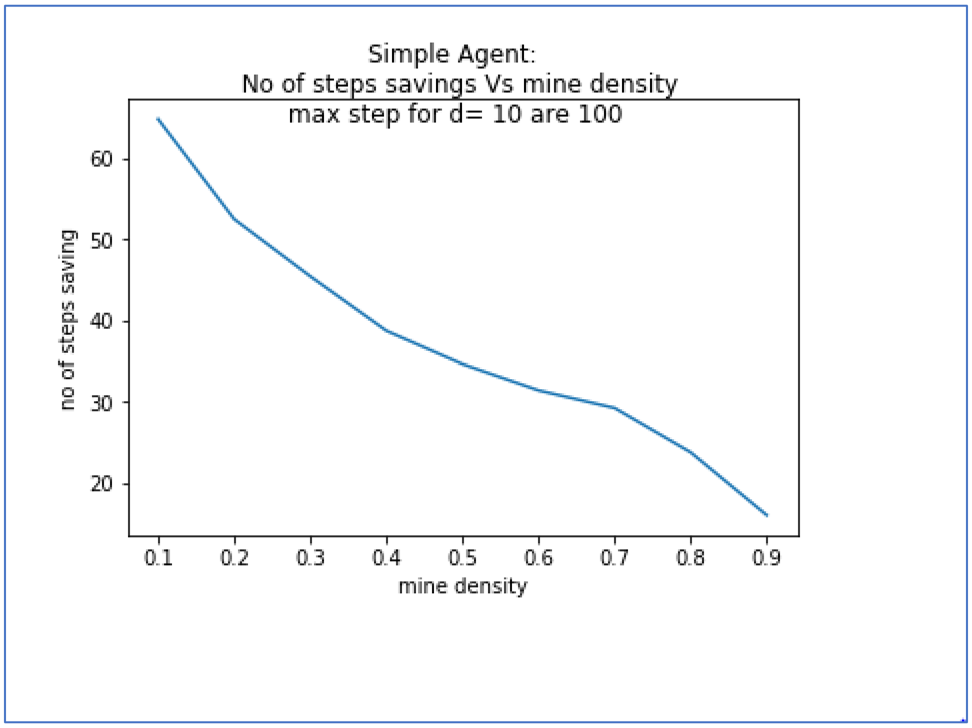


Figure 23: Total no of savings Vs mine density

Lets observe the performances of the simple agent Vs improved agent.

The 100 iteration results for d=10 are shown in below graph.

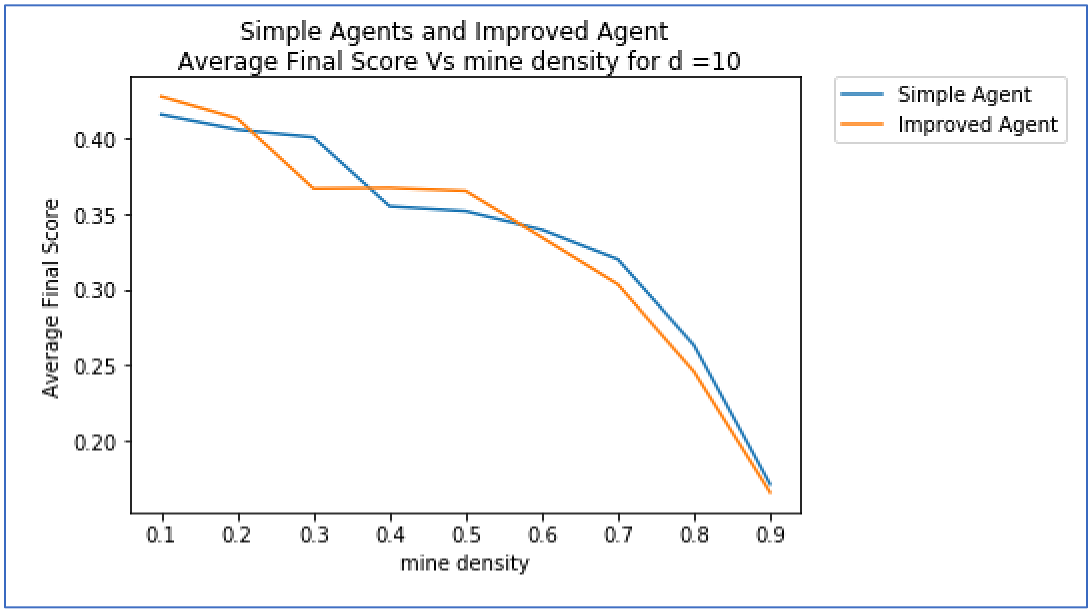


Figure 24: Simple Agent Vs Improved Agent

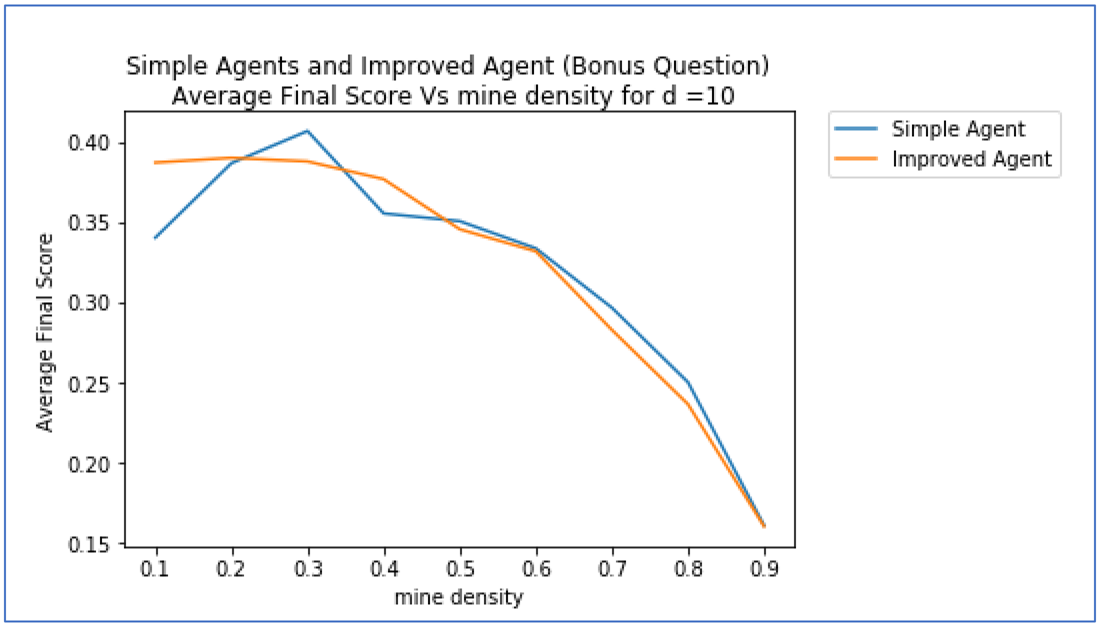


Figure 25 : Simple Agent Vs Improved Agent (Knowing no of total mines on board)

Observations: Simple Agent Vs Improved Agent

We can comment 3 points here based upon mine densities

- **case1: 0.4<= n <=0.6,** the improved agent is outperforming the simple agent significantly which is as expected
- **case 2: n<0.3 and n> 0.7,** the cases where simple agent is outperforming the improved agent is occurred due to randomness in the original board. Ideally, they both should perform similar in these mine densities which expected as per our intuition.
- In general, an Improved agent must be equal to or better than the simple agent. The cases where we are getting the simple agent outperforming improved agent is simply due to the fact of algorithm in which same board is not passed for simple and improved agent algorithm. Randomness led to these differences.

Note: The improved agent performs greater than or equal to the simple agent which is as expected. The differences can be observed significantly for range (0.4,0.6)

The similar observations are observed with the simple agent and improved agent knowing no of mines (Bonus Question is attempted to compare the result similarities in both approaches).

The results are as expected :)

Bonus Question Analysis:

Now lets try to analyze the results of Simple agent Vs Simple agent with total number of mines known.

Similar algorithm is used for improved agent along with simple agent.

The following graph is calculated for d=10 with 100 iterations.

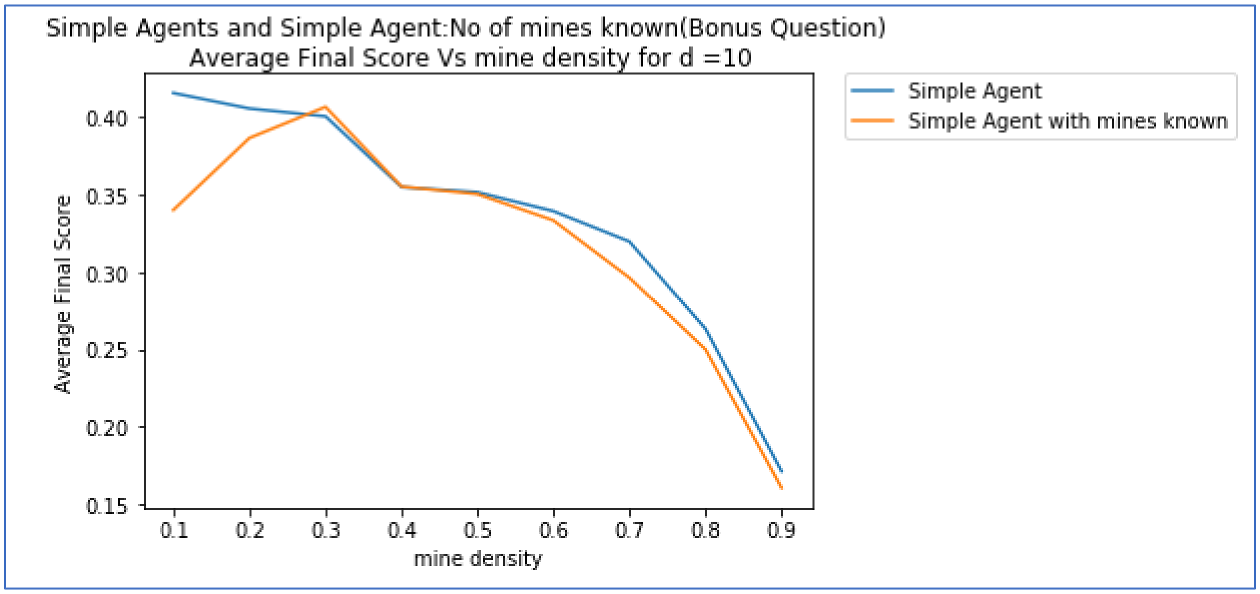


Figure 26: Simple Agent Vs Simple Agent with total no of mines known

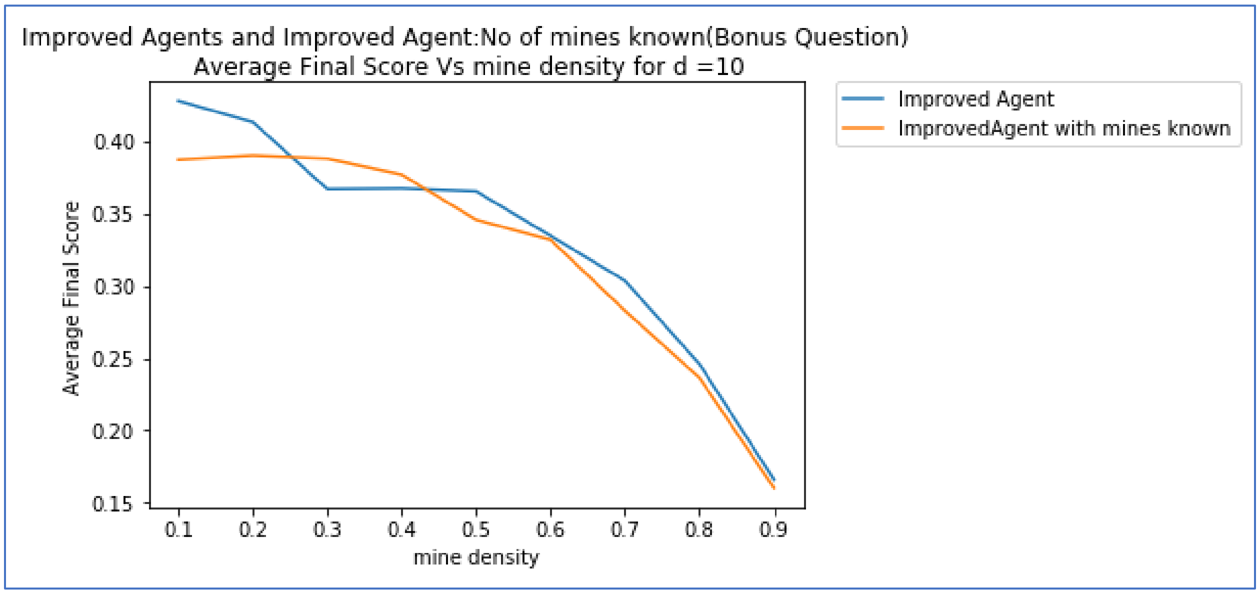


Figure 27: Improved Agent Vs Improved Agent with total no of mines known

Observations: Simple Agent Vs Simple Agent with mines known

We can comment 3 points here based upon mine densities

- **case-1 : n<0.3** :As intuitively the no of mines are very less which supports the result that knowing no of mines for these lower densities doesn't add up much information to reduce the no of searches

- **case-2: 0.3<=n <0.5:** Intuitively we can observe presence of sufficient number of higher clue values compared to lower mine densities to be able to guess new mines knowing no of total mines. Thus, the no of mines adds up to useful information for these mine densities.
- **case-3 : n>= 0.5** :As n increases, number of mines increases and so does the risk to step on a mine cell when choose randomly. This supports our result that knowing no of mines at higher densities doesn't add up to considerable new information
- In general, the information with no of mines on the board is significant for certain mine densities**(0.2,0.5)** and not always proves useful information which is as expected as our intuition.
- **At n=0.4**, simple agent and agent with no of mines known almost performs equally (interesting similarity at n=0.4)

Note: The Improved Agent Vs no of mines known follows the similar results of simple agent in terms of observations.

Brainstorming:

- The optimal result of the performances between simple agent Vs improved agent. The value of d which gives us the accurate results.
- The spike and result similarity at 0.4 mine density in the graph of agent Vs agent with total number of mines known.
- The possible performances of minesweeper algorithm at higher dimensions.
- Is it possible to design an agent which will always win the minesweeper game? What schemes must be implemented to make its accuracy 100%
- Will it be computationally more expensive? How to manage the tradeoff between computational cost and higher dimension minesweeper game solving challenges?

Future Scope:

- We can try to improve our improved agent by using the probability approach in it. The case where an improved agent makes the decision of exploring a random cell can be more specifically chosen amongst its neighboring cells possibly by analyzing the probability of each cell being a mine cell.
- The intuition behind this approach is that out smart agent must be able to detect all mines on the board with 100% accuracy.
- The interesting thing behind its implementation will be checking whether we get 100% accuracy with the new smart agent exploration. If we get 100% accuracy then will it be same for all mine density values? If not then what can be possible reasons? And how can we improve the scheme to make it always win the game regardless of any d and n value?

Note: I am planning to implement the combined approach of Improved agent along with probability approach and will update its results soon!