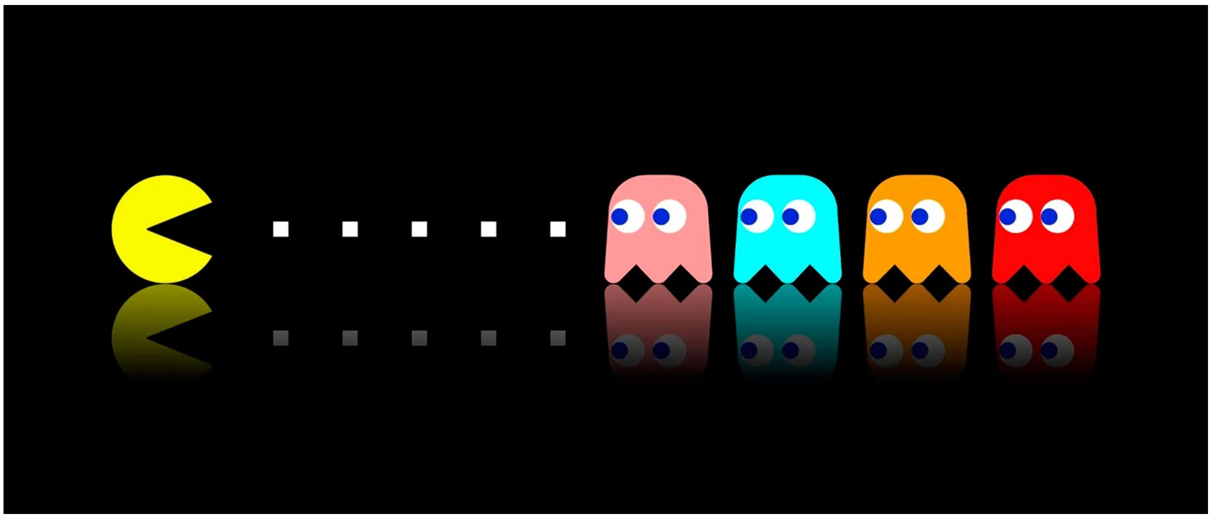
TUTORIAL 1 - MACHINE LEARNING



DONE BY:

ALBERTO CASTELLANO MACIAS with NIA: 100414242

ALEX OSWALD with NIA: 100479018

NARIM WOO with NIA: 100478751

TUTORIAL 1 - MACHINE LEARNING

EXERCISES

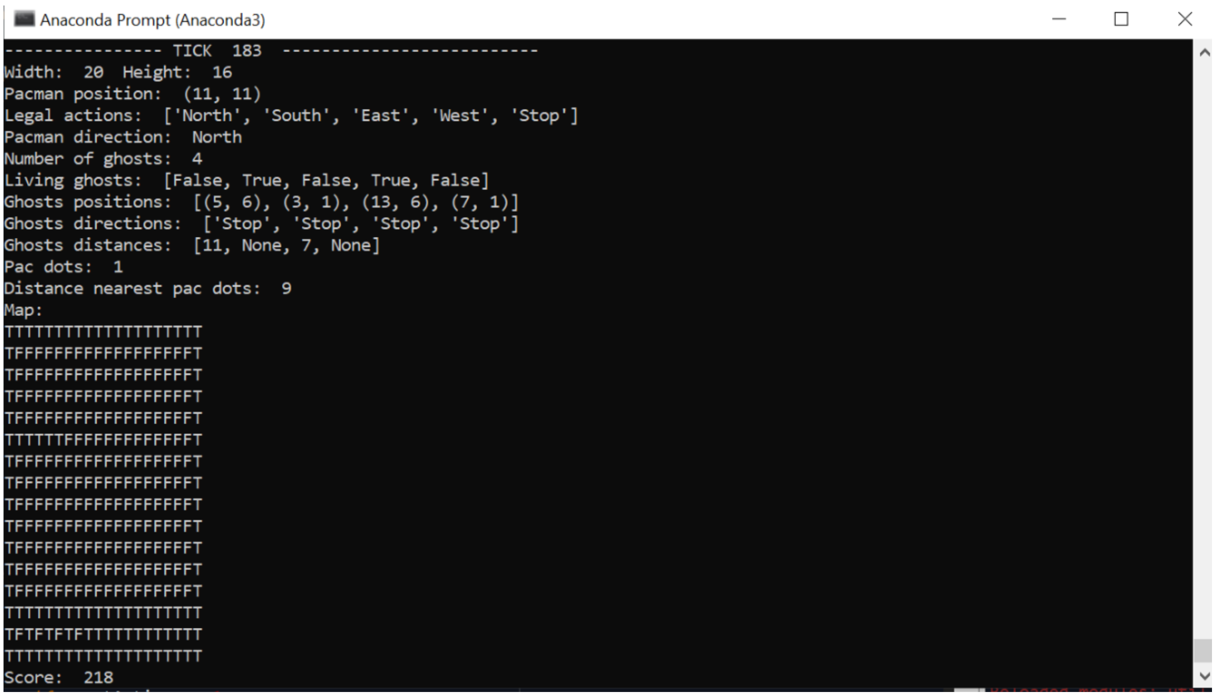
1. What information is shown in the interface? And in the terminal? Which position does Pac-Man occupy initially?

The interface shows the code used to run the Pac-man game.

As we can see in the image below, the terminal shows many pieces of information about the ongoing game.
For every tick, it shows the following information:

- Width and height of the map
- The position of the Pac-man: (x, y)
- The allowed movements the Pac-man can make for the following tick
- The movement the Pac-man just made
- The number of initial ghosts in the game
- Information about whether each ghost is alive or dead (T/F) (the first index corresponds to the Pac-man and is always False)
- The position of each ghost (x, y)
- The movement of each ghost (always Stop)
- The Manhattan distance between the Pac-man and each of the ghosts (None if a ghost is already dead)
- Number of Pac dots remaining
- The Manhattan distance between the Pac-man and the Pac dot
- A T/F list showing the positions of the walls in the map
- The score of the ongoing game

TUTORIAL 1 - MACHINE LEARNING



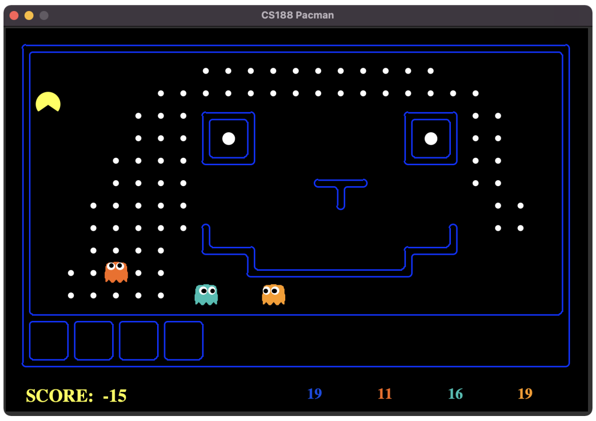
```
--------------- TICK  183 -------------------------
Width:  20  Height:  16
Pacman position:  (11, 11)
Legal actions:  ['North', 'South', 'East', 'West', 'Stop']
Pacman direction:  North
Number of ghosts:  4
Living ghosts:  [False, True, False, True, False]
Ghosts positions:  [(5, 6), (3, 1), (13, 6), (7, 1)]
Ghosts directions:  ['Stop', 'Stop', 'Stop', 'Stop']
Ghosts distances:  [11, None, 7, None]
Pac dots:  1
Distance nearest pac dots:  9
Map:
TTTTTTTTTTTTTTTTTTTT
TFFFFFFFFFFFFFFFFFFT
TFFFFFFFFFFFFFFFFFFT
TFFFFFFFFFFFFFFFFFFT
TFFFFFFFFFFFFFFFFFFT
TTTTTTFFFFFFFFFFFFFT
TFFFFFFFFFFFFFFFFFFT
TFFFFFFFFFFFFFFFFFFT
TFFFFFFFFFFFFFFFFFFT
TFFFFFFFFFFFFFFFFFFT
TFFFFFFFFFFFFFFFFFFT
TFFFFFFFFFFFFFFFFFFT
TFFFFFFFFFFFFFFFFFFT
TTTTTTTTTTTTTTTTTTTT
TFTFTFTFTTTTTTTTTTTT
TTTTTTTTTTTTTTTTTTTT
Score:  218
```

The initial position of the Pac-man is (12, 10)

2. In your opinion, which data could be useful to decide what Pac-Man should do on each tick?
In our opinion, the distance between the Pac-man and each of the ghosts and Pac dots should determine the next move the Pac-man should make, to make the most efficient path possible.

3. Take a look at the layout folder. How are the mazes defined? Create a new maze, save it and execute the game on it. Include a screenshot of the maze in the document.

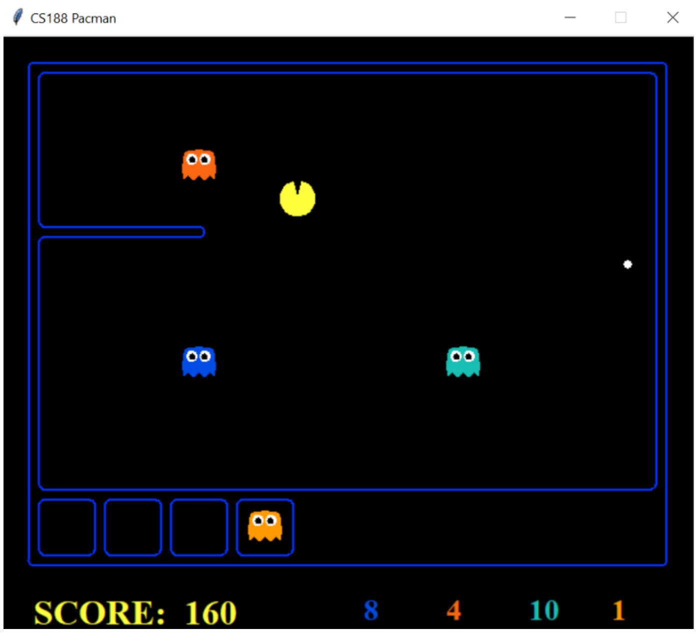
We created a new maze, `newlayout.lay`. A figure of it is shown below:



TUTORIAL 1 - MACHINE LEARNING

4. Execute the agent BasicAgentAA with the following command: python busters.py -p BasicAgentAA. Describe which information is shown on the screen about the game state. Which information do you consider the most useful?

As we can see in the screenshot below, the score of the game is shown in the lower left corner. Also, the Manhattan distance to each of the ghosts is shown in the lower right corner. Each distance is differentiated by their color, corresponding to each ghost.
In addition, the blue lines correspond to the walls of the map and the squares below the map contain a ghost if such has already been eaten, or are empty if it is still alive.



We believe that the distances to each ghost are useful to decide what movement to make next, probably towards the nearest ghost. The score is also useful as you try to maximize it during the game.

TUTORIAL 1 - MACHINE LEARNING

5. Program a method called printLineData() inside the BasicAgentAA agent from the bustersAgents.py file. This method should return a string with the information from the Pac-Man state you consider relevant. Then, you call this method from the main loop of the game in Game.py to write the information to a file. For each tick, you should store a line with all the considered information, splitting each data with commas. Moreover, each time a new game starts, the new lines must be appended below the old ones. You should not rewrite the file when a new game starts.

As seen below, we decided to store information about the game score, Pac-man's position, remaining food counts, remaining capsule counts, and the coordinate positions of each Ghost (living status indicated), and the game score for each tick, separated by commas and as strings.

To do this, we implemented the following method to save the game state on each tick:

```python
def printLineData(self, gameState):
    # game & pacman stats
    score = f"{gameState.getScore()}"
    pacman_pos = f"{gameState.getPacmanPosition()[0]},{gameState.getPacmanPosition()[1]}"

    # use manhattan distances list to check if ghost is dead -> put in
    # current coordinates or 'None' if it's dead.
    ghost_dists_test = gameState.data.ghostDistances
    if (type(ghost_dists_test[0]) == int):
             ghost0_pos = f"{gameState.getGhostPositions()[0][0]},{gameState.getGhostPositions()[0][1]}"
    else:    ghost0_pos = f"{None},{None}"
    if (type(ghost_dists_test[1]) == int):
             ghost1_pos = f"{gameState.getGhostPositions()[1][0]},{gameState.getGhostPositions()[1][1]}"
    else:    ghost1_pos = f"{None},{None}"
    if (type(ghost_dists_test[2]) == int):
             ghost2_pos = f"{gameState.getGhostPositions()[2][0]},{gameState.getGhostPositions()[2][1]}"
    else:    ghost2_pos = f"{None},{None}"
    if (type(ghost_dists_test[3]) == int):
             ghost3_pos = f"{gameState.getGhostPositions()[3][0]},{gameState.getGhostPositions()[3][1]}"
    else:    ghost3_pos = f"{None},{None}"

    # get food & capsule stats
    food = f"{gameState.getNumFood()}"
    capsules = f"{len(gameState.getCapsules())}"

    # compile shortened statistic variables to one string to be appended to csv
    state = f"{score},{food},{capsules},{pacman_pos},{ghost0_pos},{ghost1_pos},{ghost2_pos},{ghost3_pos}"

    return state
```

Usage:

```
USAGE: rm ex.csv && py busters.py -p BasicAgentAA -g RandomGhost -l openClassic
```

Sample CSV output from above usage…

```
score,num_food,num_capsules,pacman_x,pacman_y,ghost0_x,ghost0_y,ghost1_x,ghost1_y,ghost2_x,ghost2_y,ghost3_x,ghost
3_y
0,121,2,4,12,9,3,11,7,19,12,22,9
-1,121,2,5,12,9,4,10,7,19,11,22,9
-2,121,2,5,12,9,3,10,6,18,11,23,9
97,120,2,5,11,10,3,9,6,18,11,23,9
196,119,2,5,10,10,3,9,6,17,11,23,8
```

```
195,119,2,5,10,10,3,9,6,18,11,22,8
194,119,2,5,10,10,3,9,7,18,10,21,8
293,118,2,6,10,10,3,9,6,18,11,21,8
292,118,2,6,9,10,4,9,6,18,10,21,7
291,118,2,6,9,10,5,9,5,18,11,21,8
390,117,2,6,8,11,5,9,4,18,11,21,7
489,116,2,6,7,11,4,10,4,17,11,22,7
588,115,2,7,7,11,3,10,3,18,11,22,8
687,114,2,7,6,11,4,10,3,18,11,23,8
786,113,2,8,6,11,3,10,3,17,11,23,9
785,113,2,8,5,11,4,10,3,18,11,22,9
784,113,2,9,5,12,4,10,4,18,11,21,9
783,113,2,9,5,12,4,9,4,17,11,21,8
982,113,2,9,4,13,4,None,None,16,11,21,9
1081,112,2,10,4,13,3,None,None,16,11,21,8
1180,111,2,11,4,13,4,None,None,16,12,21,8
1279,110,2,12,4,14,4,None,None,16,12,22,8
1278,110,2,13,4,14,5,None,None,15,12,23,8
1277,110,2,13,4,14,5,None,None,15,12,22,8
1276,110,2,13,4,13,5,None,None,16,12,23,8
1475,110,2,13,5,None,None,None,None,16,11,23,7
1474,110,2,13,6,None,None,None,None,16,10,22,7
1473,110,2,13,7,None,None,None,None,16,11,21,7
1472,110,2,13,8,None,None,None,None,15,11,21,8
1471,110,2,13,9,None,None,None,None,15,10,22,8
1470,110,2,14,9,None,None,None,None,15,11,22,9
1469,110,2,14,10,None,None,None,None,15,12,23,9
1468,110,2,14,11,None,None,None,None,15,11,22,9
1767,109,2,15,11,None,None,None,None,None,None,22,9
1866,108,2,16,11,None,None,None,None,None,None,22,8
1965,107,2,17,11,None,None,None,None,None,None,22,7
1964,107,2,18,11,None,None,None,None,None,None,22,8
1963,107,2,19,11,None,None,None,None,None,None,21,8
1962,107,2,19,10,None,None,None,None,None,None,21,8
1961,107,2,19,10,None,None,None,None,None,None,21,9
2060,106,2,20,10,None,None,None,None,None,None,21,9
2059,106,2,20,10,None,None,None,None,None,None,21,8
2158,105,2,20,9,None,None,None,None,None,None,20,8
2257,104,2,20,8,None,None,None,None,None,None,20,7
```

6. Program an “intelligent” Pac-Man. To do so, modify the method chooseAction() from the BasicAgentAA class. As you can see, by now it just act randomly. Your new Pac-Man should chase and eat all the ghosts in the screen. Compare the results obtained by executing the game with the static and random ghosts (-g RandomGhost). Play several games and determine the number of ticks (on average) that your agent needs to eat the ghosts.
ty

Using the command “rm ex.csv && py busters.py -p BasicAgentAA -g RandomGhost -l openClassic”, 4 times, the average amount of moves it took for the pacman to beat the game was 35.25 moves (of set: 33+31+34+43).

7. The agent you just programmed does not use any machine learning technique. What advantages do you think machine learning may have to control Pac-Man?

Machine learning might add many advantages to controlling our Pac-man. It can improve its decision making by taking into account many more variables, such as the direction each ghost takes or the Pac-man itself, instead of using a fixed algorithm. This might allow the Pac-man to redirect its route during the game depending on these many variables and making a more efficient path.

CONCLUSIONS

Overall, we have learned that the Pac-man program can be modified in many ways to obtain the desired outputs, by playing with the different layouts, changing the number of ghosts, implementing different algorithms for the Pac-man to function, etc. Principally, we modified the game.py to alter the functionality of the game and siphon data from the gameplay to a CSV file which records relevant information for later use; in the future similar datafiles can be used to train a machine learning model. Additionally, we modified the bustersAgents.py file to customize our own class of agent for the Pacman with custom functionality, and we learned how to customize our own game maps by creating custom .lay files, and using various different ASCII characters to add different functionality and aspects to the game. However, we have also realized that by using only the pre-built methods and definitions of the BasicAgentAA file might not be the most efficient or versatile way to program the game. In the future we can make the Pac-man learn from its surroundings and change its way of functioning by utilizing more data from different ghost agents and the Grid & Actions classes. All of these discoveries reveal the potential of machine learning and how far we will be able to go in designing and creating our models for the Pacman game.