AI Assignment 3: Machine Learning

Alyssa Moore, Jieping Zhao, Keith DeSantis

NOTE******:

All of our group members wrote Assignment 1 in Java, but were not comfortable using our original code and wanted to write in Python, so we received permission from another student, Kush Shah, to use his code for assignment 1.

The code base for A* used was the Assignment 1 code of Kush Shah (kshah2@wpi.edu) and group.

A* Environment:

For reference, the codebase we used ran A* on boards which were denoted through rows and columns, of the following coordinate system:

	C0	C1	C2	C3
R0	**(0,0)**	**(0,1)**	**(0,2)**	**(0,3)**
R1	**(1,0)**	**(1,1)**	**(1,2)**	**(1,3)**
R2	**(2,0)**	**(2,1)**	**(2,2)**	**(2,3)**
R3	**(3,0)**	**(3,1)**	**(3,2)**	**(3,3)**

Explanation of Model:

Features:

Below, we list the features gathered and provide an explanation of what they are and how they were calculated, if necessary.

1. **Row** - the row of the board the agent was located in
2. **Column** - the column of the board the agent was located in
3. **Goal Row** - the row of the board the goal was located in
4. **Goal Column** - the column of the board the goal was located in
5. **Vertical Distance to Goal** - the absolute value of the agent's row minus the goal row

6. **Horizontal Distance to Goal** - the absolute value of the agent's column minus the goal column
7. **Sector Cost** - This feature involved using the agent's current position and the goal's location to calculate the sector of the board that the agent was in. We then used a method to calculate the sum of the three nodes (neighbors of the goal node) that correlated to that sector. Below is a figure displaying how we defined the sectors of the board based on where the goal node was located. In cases where the goal node is located on the edge or corner of the board, the sectors remained the same, though we made a separate method to calculate the "sector cost" in boards with edge-case goal locations, as the number of rows or columns that made up the sector differed from the normal case.

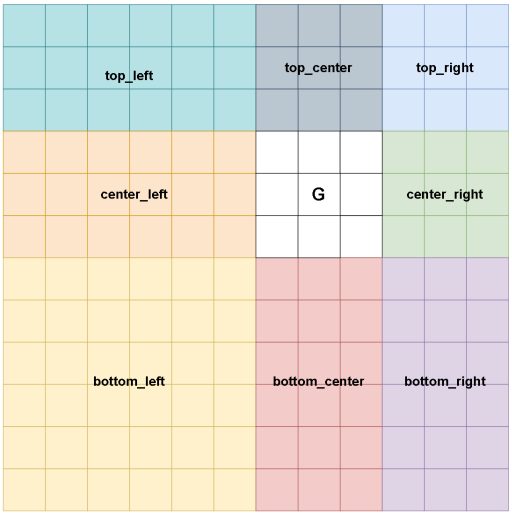

Figure 1: Sector definitions based on the goal node being in the middle of the board

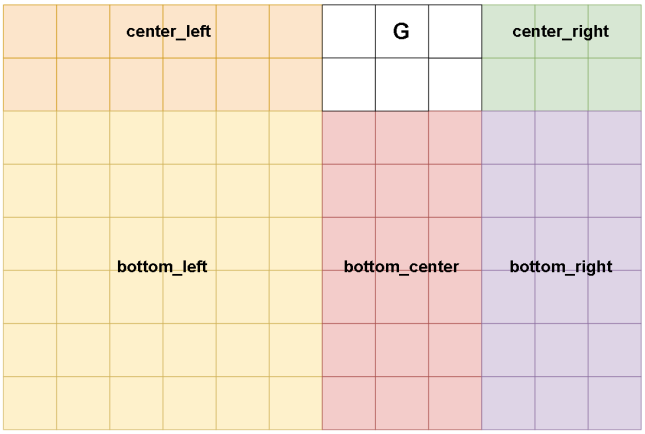

Figure 2: Sector definitions with the goal node being on the edge of the board

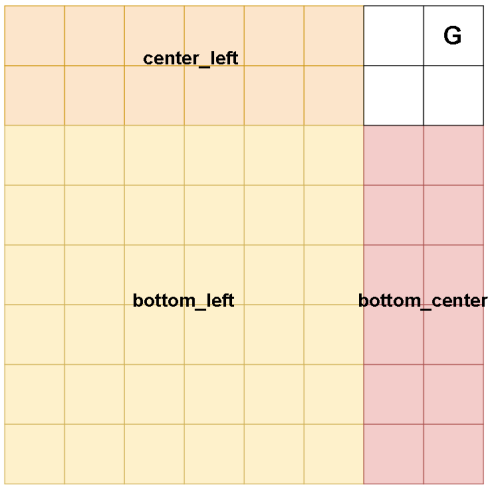

Figure 3: Sector definitions with the goal node being in a corner of the board

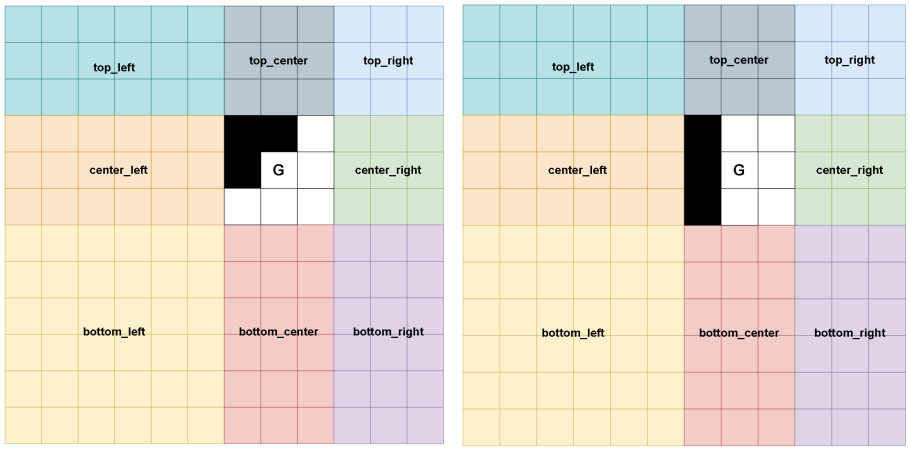

Figure 4: (left) The three neighbors summed if the agent was in the "top_left" sector (right) The three neighbors summed if the agent was in the "center_left" sector

8. **Direction** - The direction the agent is facing on the board. This was stored as a number, such that "north" = 1, "east" = 2, "south" = 3, "west" = 4.
9. **Cost to Goal** - What the learner is trying to estimate

Heuristic 5

$$heuristic_5 = \sqrt{v^2 + h^2} + \frac{v+h}{2}$$

Where:

- v is the vertical distance from the current node to the goal
- h is the horizontal distance from the current node to the goal

This was the heuristic 5 that Alyssa's group used in Assignment 1. The first part of the equation is the pythagorean theorem, used for finding the straight-line distance from the current node to

the goal node. The second part of the equation finds the area of the triangle that the horizontal and vertical distances create between the current and goal nodes. Added together, these values create an admissible heuristic that is better than H4 ($heuristic = v + h$).

Model Creation and Results:

Linear Regression:

The model type we used in this assignment is linear regression. Linear regression is an approach for modeling the relationship between two or more variables. It performs the tasks to predict a dependent target based on the given independent variables. Despite its advantages of being intuitive and robust, we chose this model type because we already had some basic understanding of it during lectures.

Weka:

Weka, the platform that we learned about during class, is a collection of machine learning algorithms for data mining tasks. It is primarily used for data analysis and predictive modeling. Its graphical user interfaces also makes it easy to access the functions. It has a large number of regression algorithms available, and it is fairly simple to use. To implement the linear regression model using weka, we found and picked it under the 'function' folder in the classify tab. We kept the cross-validation as its default 10 folds, before we hit the start button to begin the training.

Returned Model:

According to the dataset generated with the 6 by 6 boards using heuristic 5, (which contained 21,313 observations) we came up with a regression describing the estimated total cost to goal. In the actual model returned, the equation takes into consideration the exact row and column of the current location, the direction of the agent, and the sector cost, with a correlation number of **0.9292**.

Learned Heuristic Equation (dataset generated on 6x6 boards with H5):

$$Cost\ to\ Goal = -0.1364 * Row + 0.0183 * Column + 0.0616 * Goal\ Row + 2.7938 * Vertical\ Distance\ to\ Goal + 2.918 * Horizontal\ Distance\ to\ Goal - 0.2633 * Direction + 0.0375 * Sector\ Cost - 0.0396$$

An interesting note is that the learner gave the feature "goal column" a coefficient of 0. This indicates that the learner determined that the goal's column was not a significant enough factor in

the cost to goal to take into account. This surprised us at first, but as described below, the results of testing our model's heuristic were very positive.

The features and their learned coefficients are shown below in descending order of the coefficient's magnitude for ease of reading:

Feature	**Coefficient**
Horizontal Distance to Goal	2.918
Vertical Distance to Goal	2.7938
Direction	-0.2633
Row	-0.1364
Goal Row	0.0616
Constant Value (Not a feature)	-0.0396
Sector Cost	0.0375
Column	0.0183
Goal Column	0

Trial Data:

Heuristic 5:

	Number Nodes Expanded	**Solution Cost**	**Number Actions**	**Effective Branching Factor**
Trial 1	5	3	3	1.71
Trial 2	62	11	8	1.68
Trial 3	163	16	9	1.76
Trial 4	8	7	3	2.0
Trial 5	4	3	3	1.59
Trial 6	70	15	6	2.03

Trial 7	829	20	11	1.84
Trial 8	421	17	11	1.73
Trial 9	43	10	5	2.12
Trial 10	245	19	11	1.65
Average	**185.0**	**12.1**	**7.0**	**1.81**

Heuristic 6:

	Number Nodes Expanded	**Solution Cost**	**Number Actions**	**Effective Branching Factor**
Trial 1	3	3	3	1.44
Trial 2	14	11	8	1.39
Trial 3	18	18	9	1.38
Trial 4	3	7	3	1.44
Trial 5	4	3	3	1.59
Trial 6	10	15	6	1.47
Trial 7	13	23	10	1.29
Trial 8	19	17	11	1.31
Trial 9	15	10	5	1.72
Trial 10	28	19	11	1.35
Average	**12.7**	**12.6**	**6.9**	**1.44**

Heuristic 7:

	Number Nodes Expanded	**Solution Cost**	**Number Actions**	**Effective Branching Factor**

Trial 1	4	3	3	1.59
Trial 2	16	11	8	1.41
Trial 3	23	18	9	1.42
Trial 4	4	7	3	1.59
Trial 5	4	3	3	1.59
Trial 6	15	15	6	1.57
Trial 7	43	20	11	1.41
Trial 8	23	17	11	1.33
Trial 9	13	10	5	1.67
Trial 10	72	19	11	1.48
Average	**21.7**	**12.3**	**7.0**	**1.50**

Analysis:

For reference, here are the average measurements:

AVERAGES	Nodes Expanded	Solution Cost	Eff. Branching Factor
Heuristic 5 (Optimal)	**185.0**	**12.1**	**1.81**
Heuristic 6	**12.7**	**12.6**	**1.44**
Heuristic 7	**21.7**	**12.3**	**1.50**

Based on the results, it seems that heuristic 7 (H7) is non-admissible, as it sometimes found non-optimal solutions when compared to heuristic 5 (H5), however H7 still greatly outperformed heuristic 6 (H6).

H7 was non-admissible since its average solution cost was greater than H5's, but H7 still managed to find better paths than H6 in some cases, as shown by it having a lower average solution cost than H6. The trade off for this improved performance was a very minimal increase in the number of nodes expanded and effective branching factor. However, the increase in nodes expanded was very small (+3 nodes on average), especially compared to the 185 nodes that H5 expanded on average. It follows that H7 would have a slightly higher branching factor than H6, which we do observe, however the difference is relatively small, only around 0.06 on average for

this experiment. Both H7 and H6, as expected, have a lower effective branching factor than H5. H7 also ran similarly fast compared to H6, but often found better paths.

All this indicates that while H7 is non-admissible and can overestimate heuristics, it is likely that it will not usually make overestimates as egregious as H6. In other words, in many cases the real cost to goal vs. calculated heuristic for the three would look something like this:

Figure 5. Heuristic estimates of cost to goal vs. real cost to goal

This observation shows H7 often overestimating, but missing the mark by less than H6. In general, we believe H7 is the best of these heuristics to use for practical purposes, as it is able to run *significantly faster* than H5, and will often give relatively good paths in terms of score.