National University of Singapore

CS2109S—Introduction to AI and Machine Learning

Midterm Assessment

Semester 2, 2024/2025

Time allowed: 1 hour 30 minutes

Instructions:

1. Please place your student card or identification document (NRIC, driving license, etc.) on the top right-hand corner of your desk.
2. Please switch off your personal devices with communication features and leave them on the floor next to your desk at all times.
3. If you wish to communicate with an invigilator, go to the washroom, or leave before the end of the assessment, please raise your hand to inform the invigilator.
4. Please follow the other instructions in Examplify.
5. This paper contains the context for the questions in Examplify.
6. This paper contains **Eight (8) pages** including this cover page.
7. This paper should not be submitted.
8. All questions must be answered in Examplify.
9. You may refer to the appendix provided in Examplify.

	Number of questions	Total marks
Uninformed Search	7	9
Informed Search	3	6
Local Search	1	4
Adversarial Search	6	9
Decision Trees	4	8
Linear/Logistic Regression	6	14
Total	27	50

This page is intentionally left blank.

It may be used as scratch paper.

Part 1: Uninformed Search (7 questions, 9 marks)

The Number Transformation Puzzle

You start with an integer N and must transform it into a target integer M using the following operations:

1. **Add 1** (cost: 1)
2. **Subtract 1** (cost: 1)
3. **Multiply by 2** (cost: 2)
4. **Divide by 2** (cost: 2, only allowed if the current number is even).

For example, transforming 5 into 8 could follow the path: 5 – (x2) → 10 – (-1) → 9 – (-1) → 8 (total cost: 4).

Problem Formulation

- **States**: Any integer (unbounded).
- **Initial State**: N.
- **Actions**: Add1, Subtract1, Multiply2, Divide2 (Divide2 is only valid if the current number is even).
- **Transition Function**: Applying an action updates the current integer (e.g., Add1 increases it by 1).
- **Goal Test**: Is the current integer equal to M?
- **Path Cost**: Sum of the costs of the actions taken.

Note:

- A solution may not be optimal.
- An optimal solution is the least-cost solution.

Analyze the search formulation above and answer questions 1A-1G.

1A. [1 mark] Is it true that the search formulation results in a state space where a state **can** be **visited multiple times**? Note: we do not care about the search algorithms in this question since we are asking about the state space, not the search tree.

a. Yes
b. No

1B. [1 mark] Does the search formulation result in **many goal states**?

a. Yes
b. No

1C. [1 mark] Is there **always** a solution when using the search formulation?

a. Yes
b. No

1D. [1 mark] Suppose that you use search without visited memory and queue-based search (e.g., BFS, UCS). Is the search tree finite?

c. Yes
d. No

1E. [1 mark] Suppose that you use search without visited memory and Depth-First Search (DFS). Does the search always terminate?

e. Yes
f. No

1F. [2 marks] Which of the following search (without visited memory) algorithm(s) can we employ such that the search **always finds an answer** (valid solution) if a solution exists?

a. Breadth-First Search (BFS)
b. Depth-First Search (DFS)
c. Uniform-Cost Search (UCS)
d. Depth-Limited Search (DLS) with DFS and max-depth M
e. None of the above

1G. [2 marks] Suppose that we use **search with visited memory**. Which of the following search algorithm(s) is/are the best for the problem?

Best means the algorithm(s) should be complete, optimal, efficient (in terms of big O worst-case space and time complexity), and aware if there is no solution.

a. Depth-First Search (DFS)
b. Uniform-Cost Search (UCS)
c. Depth-Limited Search (DLS) with DFS and max-depth M
d. Iterative Deepening Search (IDS) with DFS
e. None of the above

Solution

1A. **a**

The state space consists of all integers (..., -2, -1, 0, 1, 2, ...). Because you can use "Add 1" and "Subtract 1" moves, you can return to the same integer through different paths (for instance, you can move up from 5 to 6, and then subtract 1 to get back to 5, revisiting that state).

1B. **b**

The goal test is "Is the current integer equal to M?" That is a single condition referring to a single value, M. Therefore, there is only one goal state in this problem.

1C. **a**

Because we have both "Add 1" and "Subtract 1," we can move from any integer N to any other integer M simply by repeated increments or decrements. Although we may use the multiply/divide operations to shorten the path, those are not strictly necessary to reach M; hence a solution always exists.

1D. **a**

Since a solution exists for any N and M, BFS/UCS will find the solution since it explores the path from smallest depth/cost to largest depth/cost.

1E. **b**

In an unbounded and cyclic state space (no visited memory), a naive DFS could keep following paths (e.g., keep subtracting or adding 1, then returning) that lead to cycles or unbounded growth. There is no guarantee it will terminate.

1F. **a**, **c**

Uniform-Cost Search (UCS) and Breadth-First Search (BFS)

BFS explores layer by layer (allowing for repeated expansion of new states without skipping any reachable depth), so if a solution is reachable in finite steps, BFS will eventually find it.

UCS explores paths in increasing order of path cost. Since all action costs are finite and nonnegative, any state reachable via a finite-cost path will eventually be expanded by UCS.

DFS, without visited memory, can get stuck going down an infinite path and may not return to discover a valid solution.

Depth-Limited Search (DLS) uses an incorrect depth. If M is always positive (not true!), the correct depth should be M-N to account for a negative initial state N. However, since M can be negative, DLS with negative max-depth is undefined.

1G. **b**

Uniform-Cost Search (UCS)

Depth-First Search (DFS) and Depth-Limited Search (DLS) fail (same reason as 1F).

IDS may return a solution with higher cost which resides in the same/shallower depth.

Example: Transform N=5 into M=8. IDS may return Path 1.

Path 1: 5 –(x2)→ 10 –(-1)→ 9 –(-1)→ 8, cost: 2+1+1=4, depth: 3 (suboptimal)

Path 2: 5 –(+1)→ 6 –(+1)→ 7 –(+1)→ 8, cost: 1+1+1=3, depth: 3 (optimal)

UCS is the only algorithm listed that guarantees completeness, optimality, and practical efficiency with visited memory.

Part 2: Informed Search (3 questions, 6 marks)

Amazing Maze

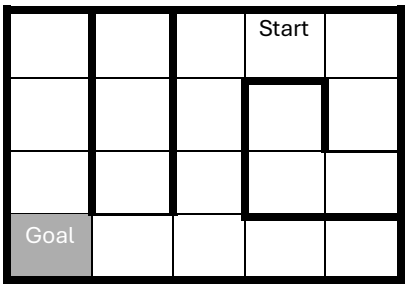


Figure: Example of a maze with start state, goal state, and walls.

You are navigating a two-dimensional maze, where each move has a variable cost depending on your position. The maze is represented as a grid, with walls dividing some of the cells and an enclosing wall surrounding the entire grid.

Let N and M be two positive integers representing the dimensions of the grid. The states are the possible positions and defined as pairs of coordinates (x, y), where $x \in \{0,1,2, \dots, N\}$ and $y \in \{0,1,2, \dots, M\}$. The initial state is some position (x_{start}, y_{start}) within the maze. The goal is the state $(0,0)$. It is guaranteed that from every state, there exists a valid path leading to $(0,0)$.

At each position (x, y), there are the four possible actions: move up, move down, move left, move right, with the corresponding transitions. However, due to the presence of walls, some actions may not be allowed at certain positions. Suppose you are at position (x, y). The costs for each valid action from this position are as follows:

- Move left/right = 2.
- Move up/down = $4y$.

The true cost to reach the goal state from position (x, y) is represented by the unknown function $h^*((x, y))$. Your friend proposes an estimate of the true cost to be

$$h_{est}((x, y)) = x + y^2.$$

You would like to apply the A* algorithm to the search problem. Based on this description, answer questions 2A-2C.

2A. [2 marks] Is h_{est} an admissible heuristic to use for the A* algorithm? Hint: consider a relaxed version of the problem.

a. Yes.
b. No.

2B. [2 marks] Is h_{est} a consistent heuristic to use for the A* algorithm?

a. Yes.
b. No.

2C. [2 marks] Which heuristic is preferred for use with A* without visited memory?

a. h_{est}.
b. $h_{est2}((x, y)) = 4x + 2y^2$.
c. $h_{est3}((x, y)) = 2x + 2y$.
d. $h_{est4}((x, y)) = 2x + 2y^2$.
e. $h_{est5}((x, y)) = 2x + 4y^2$.

Solution

For this set of questions, you must consistently use one assumption: either x describes the left/right direction and y describes the up/down direction (the main assumption) or the other way around (the alternative assumption). Switching between assumptions from one question to another is inconsistent and may suggest a lack of understanding.

2A. **a (Main assumption), b (Alternative assumption)**

First, consider the relaxed maze without the walls inside the grid. For this relaxed problem, the true cost from a state (x, y) to reach the goal state $(0,0)$ is given as follows.

Main assumption: x describes the left/right direction, y describes the up/down direction. In a 2D graphic typically the x direction is the horizontal direction, and the y direction is the vertical direction. However, we allow for the other interpretation as well (with the limitation mentioned in Part C).

The cost of going from x to 0 is $2x$, because each left/right step costs 2 and x steps are taken. An up/down step from y costs $4y$ and y steps are taken to go to 0, hence the cost to the goal is $4\sum_{j=1}^{y} j = \frac{4y(y+1)}{2} = 2y^2 + 2y$. The total cost for the relaxed maze is $h_{relax} = 2x + 2y^2 + 2y$.

Since we considered the relaxed maze, we know that $h_{relax}(x, y) \leq h^*(x, y)$, i.e., it must be smaller or equal than the true cost. Since the friend's estimate $h_{est}(x, y) = x + y^2 \leq h_{relax}(x, y)$ for all (x, y), we have that $h_{est}(x, y) \leq h^*(x, y)$ for all (x, y). Hence, h_{est} is admissible.

Alternative assumption: y describes the left/right direction, x describes the up/down direction.

The cost of going from x to 0 is $4y\ x$, because each up/down step costs $4y$ and x steps are taken. A left/right step from y costs 2 and y steps are taken to go to 0, hence the cost to the goal is 2y. The total cost for the relaxed maze is $h_{relax}(x, y) = 4xy + 2y$.

Since we considered the relaxed maze, we know that $h_{relax}(x, y) \leq h^*(x, y)$, i.e., it must be smaller or equal than the true cost. Since the friend's estimate $h_{est}(x, y) = x + y^2 \geq h_{relax}(x, y)$ for many (x, y), for example $(0,3)$. Hence, h_{est} is not admissible.

2B. **a (Main assumption), b (Alternative assumption)**

Main assumption: The possible actions are move left, right, up, down, and we are given the costs for them. For evaluating consistency, we must check the triangle inequality that says that a consistent heuristic underestimates the cost of making the corresponding move.

For a move left/right, we have $h_{est}(x,y) - h_{est}(x+1,y) = x + y^2 - (x+1) - y^2 = -1 \leq 2$, and $h_{est}(x,y) - h_{est}(x-1,y) = x + y^2 - (x-1) - y^2 = 1 \leq 2$.

For a move up/down, we have $h_{est}(x,y) - h_{est}(x,y+1) = x + y^2 - x - (y+1)^2 = -2y - 1 \leq 4y$, and $h_{est}(x,y) - h_{est}(x,y-1) = x + y^2 - x - (y-1)^2 = 2y - 1 \leq 4y$.

Hence, the heuristic is consistent.

Other assumption: Since h_{est} is not admissible, it cannot be consistent.

2C. **d (Main assumption)**

A preferred heuristic is admissible (theorem for optimality of A* without visited memory) and dominates other admissible heuristics.

Main assumption:

The closest estimate of the true cost we obtain is $h_{relax} = 2x + 2y^2 + 2y$. Hence, when a heuristic is bigger than h_{relax}, we cannot guarantee admissibility. Heuristic h_{est2} is not admissible, since $h_{est2}(x,0) = 4x \geq 2x = h_{relax}(x,0)$. Heuristic h_{est5} is not admissible, since $h_{est5}(0,2) = 4 * 2^2 = 16 > h_{relax}(0,2) = 8 + 4 = 12$.

The remaining heuristics are admissible, and we find that for all x and y it holds that $x + y^2 \leq 2x + 2y^2$ and $2x + 2y \leq 2x + 2y^2$, hence $h_{est4} = 2x + 2y^2$ is preferred.

Other assumption:

The closest estimate of the true cost we obtain is $h_{relax} = 4xy + 2y$. Hence, when a heuristic is bigger than h_{relax}, we cannot guarantee admissibility. For the given heuristics, we see that none of them is admissible. Check the point $(x,0)$, where $h_{relax}(x,0) = 0$, and hence smaller than all the given heuristics for $x > 0$. At this moment, you should have noticed that your assumption was most likely incorrect and checked the main assumption. However, we have decided to be kind and give 4 marks for the combined choice of No for Part A and No for Part B. If that choice is made, automatically no marks for Part C are received. Given that we do not know your assumption, you will receive the max of this marking and the marking arising from the main assumption.

Part 3: Local Search (1 question, 4 marks)

You are tasked with **finding the longest path** starting at *Jurong East MRT Station* in the Singapore MRT network, such that the path **visits the most stations exactly once.**

Which of the following local search formulation is/are reasonable?

In this context, we consider the formulation reasonable if hill-climbing with an infinitely large number of random restarts can reach the global optimum.

a. **State**: Current station
   Initial State: Jurong East
   Goal Test: All stations are visited
   Evaluation Function: Total distance traveled
   Successor Function: Move to any adjacent station
b. **State**: List of visited stations
   Initial State: [Jurong East]
   Goal Test: Path includes 10+ stations
   Evaluation Function: Number of unique stations visited
   Successor Function: Travel to a random unvisited station
c. **State**: Current path (sequence of stations)
   Initial State: [Jurong East]
   Goal Test: No unvisited adjacent stations remain
   Evaluation Function: Length of the path (number of stations)
   Successor Function: Extend the path to an adjacent unvisited station
d. **State**: Current station and visited set
   Initial State: (Jurong East, {Jurong East})
   Goal Test: All stations are in the visited set
   Evaluation Function: Total stations in the visited set
   Successor Function: Move to adjacent stations not in the visited set
e. None of the above.

Solution

3A. **c, d** (main assumption) or **c** (main assumption, alt) or **e** (alternative assumption)

We will accept two assumptions: stochastic (main assumption) and deterministic (alternative assumption) handling of ties.

Main Assumption

Under the stochastic assumption, the correct answer is C and D. Since there is a possible different interpretation of "adjacent station", we will also accept C only (without D).

Formulation C:

- **State**: Current path (sequence of stations).
  This explicitly tracks the path and ensures adjacency (via successor function).
- **Evaluation function**: Path length (number of stations).
- **Successor function**: Extends the path to adjacent unvisited stations.
- Hill-climbing greedily maximizes path length. With infinite restarts, it can explore all possible paths and find the longest valid one.

Formulation D:

- **State**: Current station + visited set.
  Tracks visited stations (prevents revisits) and enforces adjacency (via successor function).
- **Evaluation function**: Number of visited stations.
- **Successor function**: Moves to adjacent unvisited stations.
- The visited set ensures no cycles, and the evaluation function directly aligns with the goal. Infinite restarts allow finding the maximum possible path. The visited set implicitly defines the path's stations (even if the order isn't tracked).

Formulation A and B:

- Do not properly track visited stations or enforce adjacency, leading to invalid paths.
- Formulation 2's successor function allows non-adjacent moves (unrealistic for MRT).

Ambiguity in the "adjacent station" definition for formulation D
It could be interpreted as allowing moves to any adjacent station from any station in the visited set, which would make D incorrect. Therefore, we will also accept only C as the correct answer (without D).

Alternative Assumption

Under the deterministic assumption, the correct answer is E. This is because, in a deterministic scenario, the local search algorithm would be unable to reach the global optimum, even with infinite restarts, as it would always select the same adjacent state in each iteration.

Part 4: Adversarial Search (6 questions, 9 marks)

Race to Eleven

Players A and B alternate turns. A token starts at position **1** on a 1D grid. On each turn, the current player moves the token **+2 or +5** positions.

The only way a player can win is by landing exactly on 11, causing the opponent to lose. In all other scenarios, the game results in a draw. If a player cannot make a valid move—meaning all available moves exceed 11—the other player also cannot make a move, and the game ends in a draw.

Use the Minimax algorithm to solve the game by constructing the complete game tree.

Notes:

- We assume that player A is the first player and is the max player.
- The value of a state is evaluated from the perspective of the max player.
- In terminal states, a win is valued at +1, a loss at -1, and a draw at 0.
- The game tree is *tree* (a child can only have one parent).

Construct the game tree and answer questions 4A-4E.

4A. [1 mark] How many terminal nodes exist in the game tree?

4B. [1 mark] What is the game tree's maximum depth (root node is at depth 0)?

4C. [1 mark] If Player A moves to position 3 on their first turn, which player can force a win?

a. A
b. B
c. Game is draw
d. Game continues indefinitely
e. None of the above

4D. [1 mark] Which sequences guarantee a Player B victory? Select all that is/are true.

a. A: +2 → B: +2
b. A: +2 → B: +5
c. A: +2 → B: +2 → A: +2
d. A: +5
e. None of the above

4E. [1 mark] Which of the following action(s) should the first player take? Select all that is/are true.

a. +2
b. +5
c. All actions result in losing the game
d. All actions result in a draw
e. Game continues indefinitely

Consider the following game tree for a two-player game where alpha-beta pruning is applied.

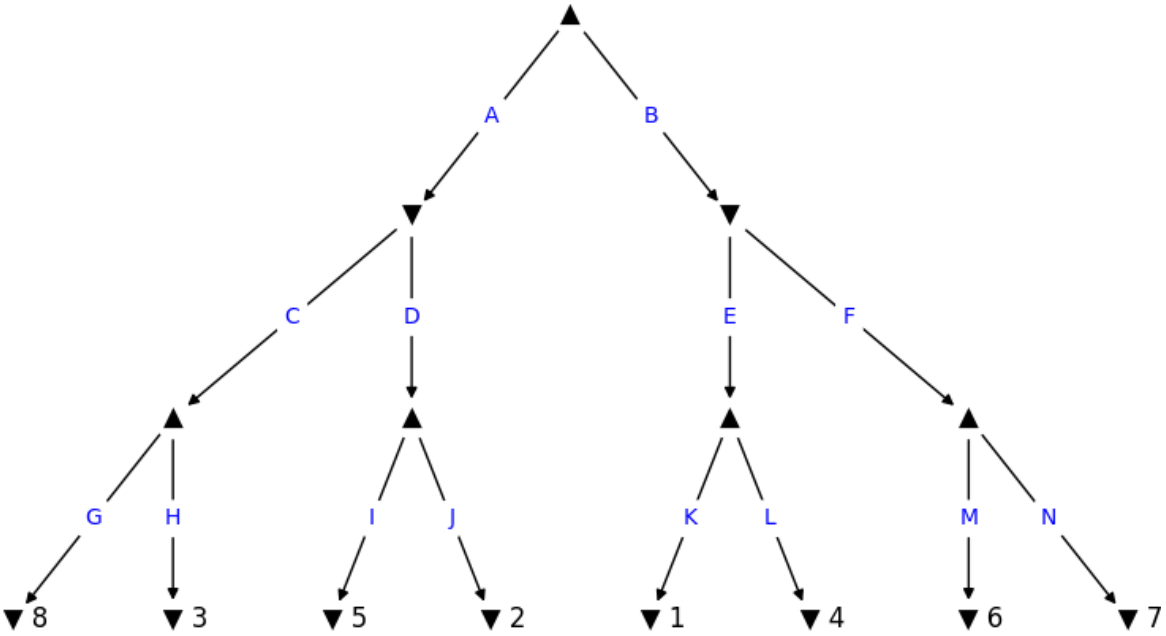


Figure: Game tree for Part 4 Adversarial search - Alpha-Beta pruning.

Note: The symbol ▲ represents the max player's turn, while ▼ indicates the min player's turn.

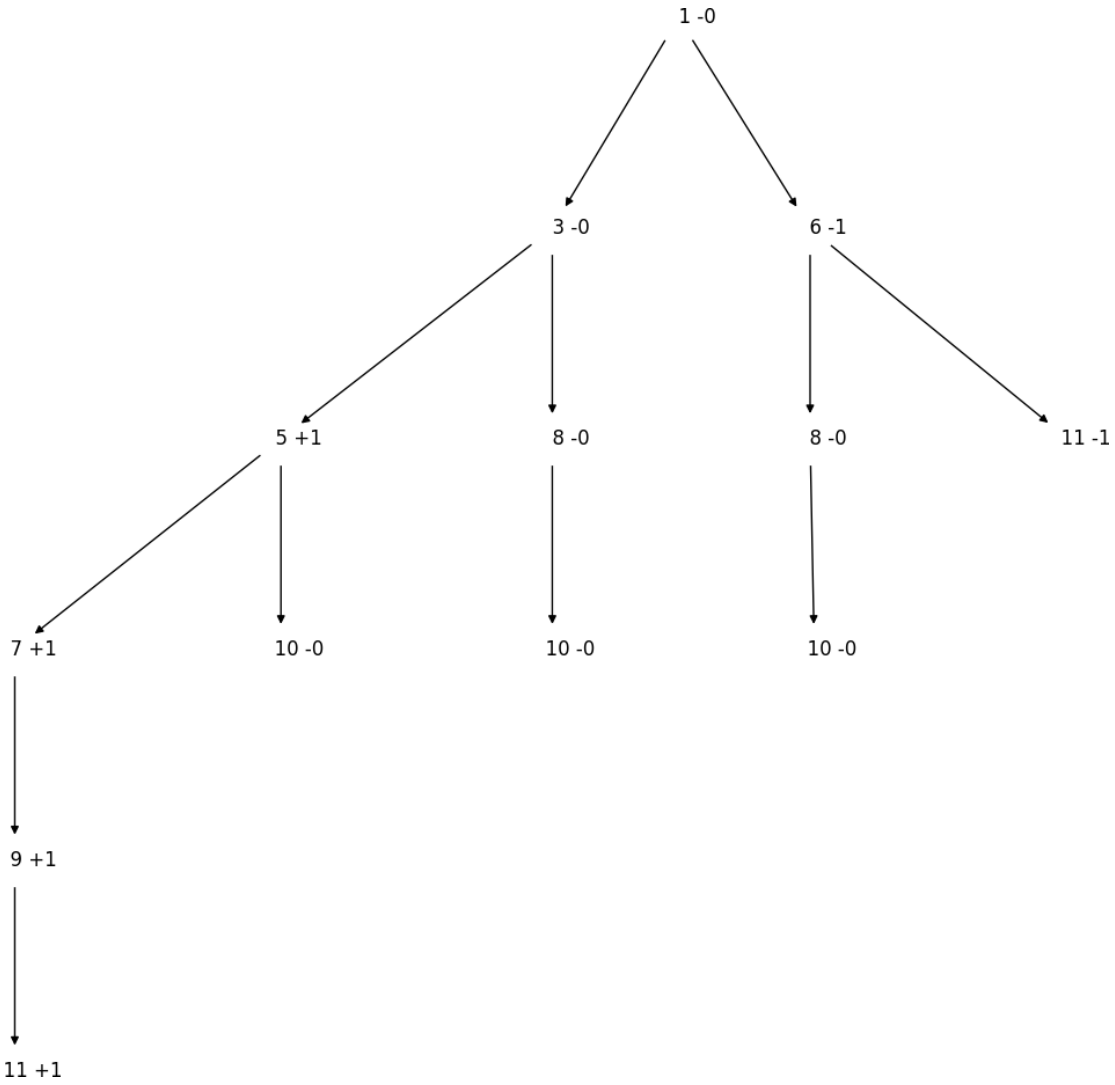
Answer question 4F.

4F. [4 marks] Suppose we traverse this tree using (depth-first) alpha-beta pruning from **left to right**. Select **all** the link(s) that would be pruned by alpha-beta pruning algorithm. Select only the links that are directly pruned and not those that are indirectly pruned because they are in a subtree of a pruned link.

Which of the following link(s) is/are pruned? Select all that is/are true.

a. A
b. B
c. C
d. D
e. E
f. F
g. G
h. H
i. I
j. J
k. K
l. L
m. M
n. N
o. None of the above

Solution



For this set of questions, you must consistently use one assumption: either all players play optimally (the main assumption) or players may play sub-optimally (the alternative assumption). Switching between assumptions from one question to another is inconsistent and may suggest a lack of understanding.

4A. **5** (tree) or **3** (graph)

4B. **5**

4C. **c** (main assumption) or **e** (alternative assumption)

If player B plays optimally (main assumption), the game results in a draw, therefore, C is the correct answer.

If player B plays sub-optimally, B will choose +2, allowing player A to force a win. In other words, there is an action (picking +2) that will ensure A wins the game.

Under the assumption that players may play sub-optimally (the alternative assumption), there are two possible outcomes: either player A can force a win if player B plays sub-optimally, or the game will result in a draw if player B plays optimally. Thus, the correct answer is e.

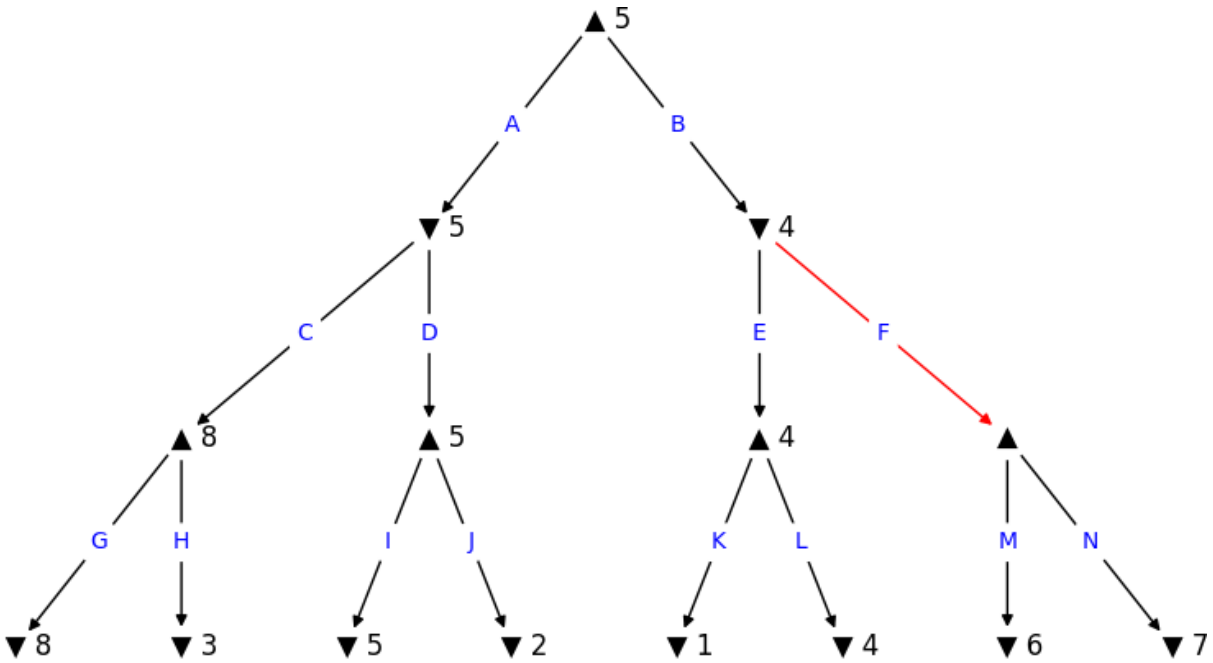
4D. **d** (main assumption) or **e** (alternative assumption)

If we assume that player B plays optimally (main assumption), then only +5 will lead to Player B winning. Otherwise (alternative assumption), then even with +5, B may take +2 which results in a draw.

4E. **a**

Between the option of losing and draw, the player will choose draw.

4F. **f**



Part 5: Decision Trees (4 questions, 8 marks)

You have been tasked with hiring for a tech startup. Leveraging your expertise, you create a decision tree based on past data given in Table 1.

	Experience	Interview	Potential	Hiring decision
1	> 3 years	Bad	High	Yes
2	1-3 years	Bad	High	Yes
3	1-3 years	Good	High	Yes
4	1-3 years	Good	Mid	Yes
5	< 1 year	Bad	Mid	No
6	< 1 year	Good	High	No
7	< 1 year	Good	High	Yes

Consider the decision tree learning algorithm using information gain. In the event of a tie in information gain, the priority for selecting attributes to construct the tree is as follows: Experience (most preferred), followed by Potential, and finally Interview (least preferred). If a decision at a leaf node is unclear, majority voting is applied, returning "No/Yes" in the case of a tie or when no examples are available.

Based on this description, answer questions 5A-5D.

Notes:

- The entropy for a given probability distribution p_i, for $i = 1, \ldots, n$ is given by:

$$Entropy = -\sum_{i=1}^{n} p_i \log_2(p_i).$$

- Remember the log is in base 2!
- Values for the log: $\log_2(1) = 0; \log_2(2) = 1; \log_2(3) = 1.585; \log_2(4) = 2; \log_2(5) = 2.322; \log_2(6) = 2.585; \log_2(7) = 2.807; \log_2(8) = 3; \log_2(9) = 3.170; \log_2(10) = 3.322.$
- For rounding to two decimal places, round to the closest number with two decimal places. For example, 1.234 is rounded to 1.23 and 1.237 is rounded to 1.24.

5A: [2 mark] What is the entropy of the Hiring Decision (yes/no) in the table, rounded to two decimal places?

5B: [2 marks] What is the information gain of selecting "Experience" as the root node of the Hiring Decision (yes/no) in the table, rounded to two decimal places?

5C: [2 marks] Given that Experience has the highest information gain, you build the remaining decision tree given the context information. According to this decision tree, what is the Hiring Decision for the following candidate?

Candidate Information: Experience: < 1 year / Interview: Good / Potential: Mid

a) Yes
b) No
c) No/Yes

5D: [2 mark] Please prune the decision tree from the previous question ensuring that **each** leaf node contains at least 2 training data points. According to your pruned decision tree, what is the Hiring Decision for the following candidate?

Candidate Information: Experience: <1 year / Interview: Good / Potential: Mid

a) Yes
b) No
c) No/Yes

Solution

5A. 0.86

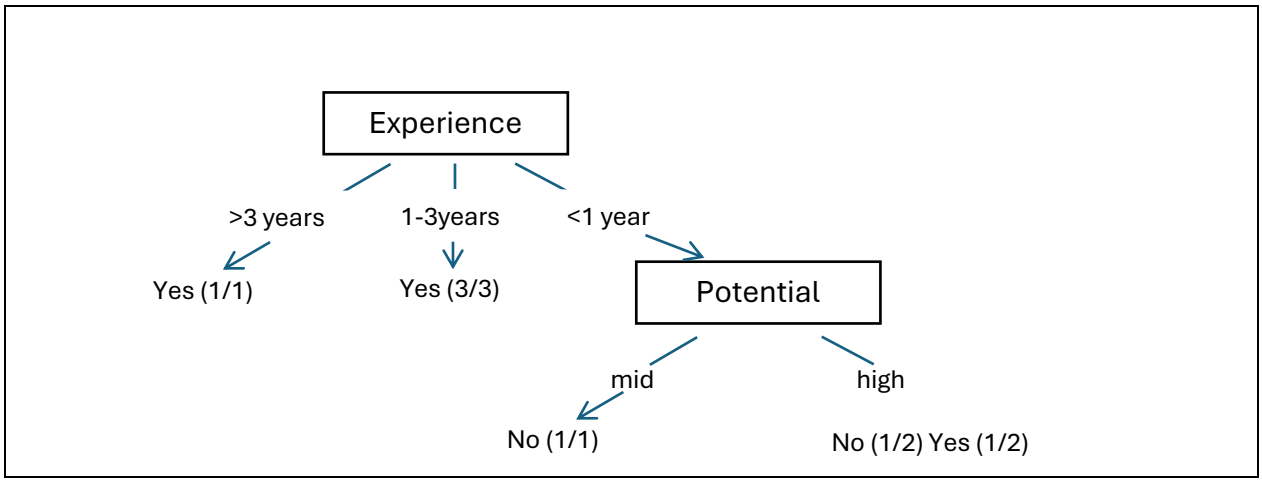
Explanation: $-\frac{5}{7}\log_2\left(\frac{5}{7}\right) - \frac{2}{7}\log_2\left(\frac{2}{7}\right) = 0.863.$

5B. 0.47

The remainder when splitting according to experience is $remainder(Exp) = -\frac{1}{7}1\log_2(1) - \frac{3}{7}1\log_2(1) - \frac{3}{7}\left(-\frac{2}{3}\log_2\left(\frac{2}{3}\right) - \frac{1}{3}\log_2\left(\frac{1}{3}\right)\right) = 0 + 0 + 0.39355$. Hence the information gain is $0.863 - 0.39355 = 0.4695 = 0.47$.

5C. **b**

The decision tree is as follows:



Experience <1 year and Potential mid leads to No for the decision.

5D. **a**

Explanation: Everything is pruned because the leaf node Experience >3 years only has 1 training points. Based on the majority at that level, the decision is Yes.

Part 6: Linear and Logistic Regression (6 questions, 14 marks)

6A. [2 marks] Consider the following housing data.

Original data

# bedrooms	Size sqm
4	113
3	102
3	100
3	84
3	112
2	68
2	53
3	122
3	150
3	90

After standardization of both features, which data set will you have?

A:

# bedrooms	Size sqm
0.62	376.20
0.06	71.92
0.06	16.60
0.06	-425.99
0.06	348.53
-0.51	-868.57
-0.51	-1283.49
0.06	625.15
0.06	1399.67
0.06	-260.02

B:

# bedrooms	Size sqm
1.94	0.49
0.18	0.09
0.18	0.02
0.18	0.56
0.18	0.46
1.59	1.14
1.59	1.68
0.18	0.82
0.18	1.83
0.18	0.34

C:

# bedrooms	Size sqm
1.94	0.49
0.18	0.09
0.18	0.02
0.18	-0.56
0.18	0.46
-1.59	-1.14
-1.59	-1.68
0.18	0.82
0.18	1.83
0.18	-0.34

Hint: The problem can be solved without calculation.

a) A.
b) B.
c) C.

6B. [4 marks] Take the linear regression hypothesis $h(x) = w_0 + w_1 x$, where $\begin{bmatrix} w_0 \\ w_1 \end{bmatrix} = \begin{bmatrix} 0.5 \\ 1 \end{bmatrix}$. We have the data points $x^{(0)} = 0, y^{(0)} = 0$ and $x^{(1)} = 2, y^{(1)} = 1,$ and the loss function is $\frac{1}{2}\left(h\left(x^{(0)}\right) - y^{(0)}\right)^2 + \frac{1}{2}\left(h\left(x^{(1)}\right) - y^{(1)}\right)^2$. Perform a single update to the vector $\begin{bmatrix} w_0 \\ w_1 \end{bmatrix}$ using the gradient descent update rule with the learning rate $1/4$. The resulting weights $\begin{bmatrix} w_{0,new} \\ w_{1,new} \end{bmatrix}$ are given by which of the options?

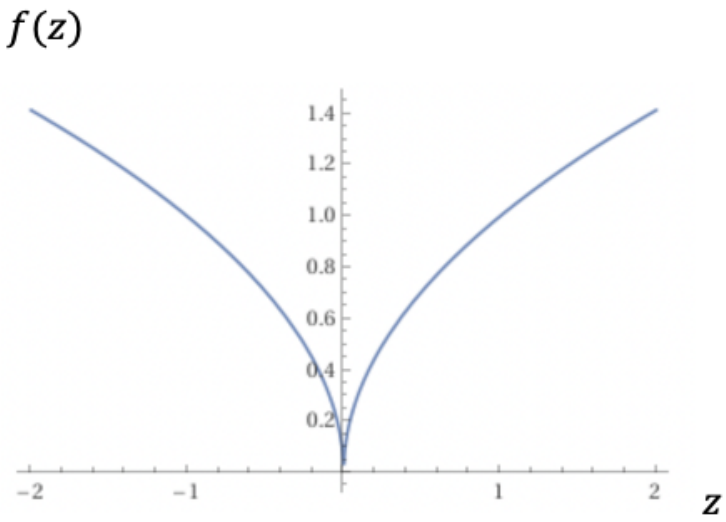
a) $\begin{bmatrix} -1/2 \\ 1/4 \end{bmatrix}$
b) $\begin{bmatrix} 0 \\ -1/4 \end{bmatrix}$
c) $\begin{bmatrix} 0 \\ -1/2 \end{bmatrix}$
d) $\begin{bmatrix} 1/2 \\ 1/4 \end{bmatrix}$

e) $\begin{bmatrix} 0 \\ 1/2 \end{bmatrix}$
f) $\begin{bmatrix} -1/4 \\ 1/2 \end{bmatrix}$
g) $\begin{bmatrix} 0 \\ 1/4 \end{bmatrix}$
h) $\begin{bmatrix} 1/4 \\ 1/2 \end{bmatrix}$
i) None of the above.

6C. [2 marks] You are given a data set that consists of 4K images, where each pixel of every image has a grayscale value $g \in [0,1]$. You create a feature vector x, where each feature corresponds to the greyscale value of a pixel. Additionally, each image has an associated real-valued target y. You plan to use a linear regression model $h(x)$, and you aim to minimize the MSE loss to predict the target value for new images. Given these choices, select true statements about the regression and learning algorithms?

a) The problem setting implies that the normal equation will be the preferred method over gradient descent.
b) Since this problem has uneven features, min-max scaling will improve the learning.
c) Stochastic gradient descent will take the most direct path to the optimal solution.
d) Since we transform features before applying linear regression, the model is overparameterized.
e) None of the statements are true.

6D. [2 marks] For a linear regression problem, you decide to use an unusual loss function. The loss function used for each data point is $f(h(x) - y)$, where $f(z)$ is graphed below for $z = h(x) - y$. You average the loss over the data set in the same way as the MSE. Given this choice, what is the best learning algorithm to use? (Hint: Best is here considered in terms of speed and finding a good minimum.)



a) Normal equation.
b) Gradient descent.
c) Stochastic gradient descent.
d) Decision tree learning.

6E. [2 marks] Let us be given 4 key attributes of a smartphone. You would like to predict the failure probability of a phone given a data set of phone failures. What is the best hypothesis class to choose?

a) $h_w(x) = w_0x_0 + w_1x_1 + \sigma(w_2x_2 + w_3x_3 + w_4x_4)$

b) $h_w(x) = w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4$

c) $h_w(x) = w_0 + w_1x_1 + w_2x_2 + w_3x_3 + x_4$

d) $h_w(x) = w_0 + w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4$

e) $h_w(x) = \sigma(w_0x_0 + w_1x_1) + \sigma(w_2x_2 + w_3x_3 + w_4x_4)$

f) $h_w(x) = \sigma(w_0 + w_1x_1 + w_2x_2 + w_3x_3 + x_4)$

g) $h_w(x) = \sigma(w_0x_0 + w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4)$

h) $h_w(x) = \sigma(w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4)$

6F. [2 marks] Consider a logistic regression model for multi-class classification with three classes: Cat, Dog, and Rabbit. We are given the following weight vectors for "One vs. Rest" classifiers, where the $h_{A/B}(x)$ represents the probability of the class A instead of B and contains the weight vector $w_{A/B}$. The weight vectors for each classifier includes the bias term as the first element in each weight vector. For example, -1 is the bias for $w_{Rabbit/Cat}$.

$$w_{Dog/Rabbit} = \begin{bmatrix} 0 \\ 0.4 \\ 0.1 \end{bmatrix},$$

$$w_{Rabbit/Cat} = \begin{bmatrix} -1 \\ 0.2 \\ -0.4 \end{bmatrix},$$

$$w_{Cat/Dog} = \begin{bmatrix} 2 \\ -0.5 \\ 0.3 \end{bmatrix}.$$

Let the decision threshold be 0.5. Given the input $x = \begin{bmatrix} 3 \\ 2 \end{bmatrix}$, determine which class the model predicts.

a) Dog
b) Cat
c) Rabbit
d) None of the above.

Solution

6A. **c**

C is the correct standardized data. Without calculation, we can rule out A, as the two features are not on the same scale, a key aspect of standardization. The two features are on even more distinct scales in A. We can rule out B based on the absence of negative values: Standardization subtracts the mean (and divides by the standard deviation) and for these data we would expect that some training points are smaller than the mean hence the standardized data should have examples with negative feature values.

6B. **g**

The loss function is $J(w) = \frac{1}{2}\left(h\left(x^{(0)}\right) - y^{(0)}\right)^2 + \frac{1}{2}\left(h\left(x^{(1)}\right) - y^{(1)}\right)^2$. First, note that $h\left(x^{(0)}\right) - y^{(0)} = 0.5$ and $h\left(x^{(1)}\right) - y^{(1)} = 0.5 + 2 - 1 = 1.5$. The partial derivatives are $\frac{\partial}{\partial w_0} J(w) = \left(h\left(x^{(0)}\right) - y^{(0)}\right) + \left(h\left(x^{(1)}\right) - y^{(1)}\right) = 0.5 + 1.5 = 2$ and $\frac{\partial}{\partial w_1} J(w) = \left(h\left(x^{(0)}\right) - y^{(0)}\right)x^{(0)} + \left(h\left(x^{(1)}\right) - y^{(1)}\right)x^{(1)} = 0 + 1.5 * 2 = 3$. The gradient update step is

$$\begin{bmatrix} w_{0,new} \\ w_{1,new} \end{bmatrix} = \begin{bmatrix} w_0 \\ w_1 \end{bmatrix} - \gamma \begin{bmatrix} \frac{\partial}{\partial w_0} J(w) \\ \frac{\partial}{\partial w_1} J(w) \end{bmatrix}$$, which becomes $\begin{bmatrix} w_{0,new} \\ w_{1,new} \end{bmatrix} = \begin{bmatrix} 0.5 \\ 1 \end{bmatrix} - \frac{1}{4}\begin{bmatrix} 2 \\ 3 \end{bmatrix} = \begin{bmatrix} 0.5 \\ 1 \end{bmatrix} - \begin{bmatrix} 0.5 \\ 0.75 \end{bmatrix} = \begin{bmatrix} 0 \\ 1/4 \end{bmatrix}$.

6C. **e**

The problem setting implies that gradient descent will be the preferred method over the normal equation, because we have many features (4K implies around 8 million pixels). In lecture we have discussed that solving a linear system (matrix inversion) comes at a cost of d^3 hence is prohibitive.

The features are even $[0,1]$, hence min-max scaling will not change much.

Stochastic gradient descent takes a randomized path to the solution. Gradient descent takes the path in the greatest decrease of the function hence is the most direct path.

The sentence “Since we transform features before applying linear regression, the model is overparameterized.” is non-sensical in this problem.

Hence, none of the statements are true.

6D. **b or c**

The loss function is non-convex. There are line segments between two points that are below the function. We discussed the property of Stochastic Gradient Descent to be

able to overcome being stuck in local minima/saddle points arising in non-convex landscapes in class. From these two observations, c) is the main correct answer.

While the function is non-convex, it is in fact quasi-convex. For quasi-convex functions, a version of gradient descent can be used in certain cases. It is called normalized gradient descent, see https://arxiv.org/abs/1507.02030 and references therein. Hence, we also allow the choice of option b).

Note on overshooting the minimum: We have introduced in class the concept of the learning rate as a hyperparameter and choosing a small learning rate (with the tradeoff of slow convergence). In addition, tutorial 4 mentions a learning rate scheduler, which can be used to decrease the learning rate during gradient descent. The question asks about the best method where "Best is here considered in terms of speed and finding a good minimum". A small learning rate during the later stage of gradient descent will prevent overshooting by large amounts, and will find a good enough minimum.

Option a) cannot be used as we cannot derive a normal equation, due to the presence of the function f. Option d) applies to classifiers with a decision tree model only.

6E. **g**

From the problem statement, we require a logistic model with 4 features. In addition, the bias improves a model. Adding the dummy feature x_0 gives the hypothesis $h_w(x) = \sigma(w_0x_0 + w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4)$.

6F. **b**

Computing the dot products (using the dummy feature) gives:

$$a_{D/R} = \begin{bmatrix} 0 \\ 0.4 \\ 0.1 \end{bmatrix}^T \begin{bmatrix} 1 \\ 3 \\ 2 \end{bmatrix} = 0.8 + 0.2 = 1., \; a_{R/C} = \begin{bmatrix} -1 \\ 0.2 \\ -0.4 \end{bmatrix}^T \begin{bmatrix} 1 \\ 3 \\ 2 \end{bmatrix} = -1 + 0.6 - 0.8 = -1.2.$$

$$a_{Cat/Dog} = \begin{bmatrix} 2 \\ -0.5 \\ 0.3 \end{bmatrix}^T \begin{bmatrix} 1 \\ 3 \\ 2 \end{bmatrix} = 2 - 1.5 + 0.6 = 1.1.$$

Since the sigmoid is monotonically increasing, these dot products tell us that Dog obtains 1 vote and Cat obtains 2 votes, hence Cat is the answer.

—END OF PAPER —