CS 188 Spring 2025 — Intro to Artificial Intelligence Final Exam

Solutions last updated: Sunday, May 18, 2025

PRINT Your Name: ____________________

PRINT Your Student ID: ____________________

PRINT Student name to your left: ____________________

PRINT Student name to your right: ____________________

You have 170 minutes. There are 9 questions of varying credit. (100 points total)

Question:	1	2	3	4	5	6	7	8	9	Total
Points:	10	9	9	14	15	11	8	14	10	100

We reserve the right to deduct points for failing to follow the marking directions below:

For questions with **circular bubbles**, you may select only one choice.

○ Unselected option (Completely unfilled)

◎ Don't do this (it will be graded as incorrect!)

● Only one selected option (completely filled)

For questions with **square checkboxes**, you may select one or more choices.

■ You can select

■ multiple squares

☑ Don't do this (it will be graded as incorrect!)

Anything you write outside the answer boxes or you ~~cross out~~ will not be graded. If you write multiple answers, your answer is ambiguous, or the bubble/checkbox is not entirely filled in, we will grade the worst interpretation.

Read the honor code below and sign your name.

By signing below, I affirm that all work on this exam is my own work. I have not referenced any disallowed materials, nor collaborated with anyone else on this exam. I understand that if I cheat on the exam, I may face the penalty of an "F" grade and a referral to the Center for Student Conduct.

SIGN your name: ____________________

Q1 *Counting Calories* (10 points)

Consider a variant of Pacman where each food dot has a different *calorie count*. Each food dot's calorie count is a strictly positive integer. Each food dot that Pacman eats adds to his total calorie count (C).

When Pacman moves into a square with a dot, he automatically eats the dot as part of that action. All actions cost 1.

Pacman's goal is to eat at least 10 calories in total, i.e. reach a state where $C \geq 10$.

Q1.1 (3 points) Select all admissible heuristics for this problem.

- ☐ C
- ☐ $10 - C$
- ☐ Manhattan distance to the nearest dot.
- ☐ Sum of Manhattan distances to all dots.
- ☐ Euclidean distance to the nearest dot.
- ☐ Sum of Euclidean distances to all dots.
- ○ None of the above

Solution: The correct answer is "None of the above".

None of these heuristics are admissible. The first two result from the fact that they do not connect with the calories on the board (they only consider Pacman's current calories), while you may have encountered the last four during the project.

Consider the goal state, for which all options are non-zero (except the second option). The second is incorrect; consider the counterexample of Pacman being right next to a food dot worth 100 calories. The cost of the true solution is 1, and clearly $100 - 0 = 100$ overestimates that.

For each of the next three subparts, consider the given maze and search algorithms. Select whether each search algorithm will find the optimal solution with **at most 10 states** expanded.

Hint: Recall that expanding a state means calling the successor function on that state.

For all subparts, when calling the successor function, enqueue the Left successor state first, then the Right successor state. This means that for DFS, the Right successor state gets popped off the stack first.

Q1.2 (2 points) A 1×11 grid with 10 dots, each worth 1 calorie. Pacman starts in the leftmost square.

P	•	•	•	•	•	•	•	•	•	•

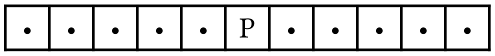
[✓ DFS tree search] ☐ BFS tree search ○ None of the above

Solution: Recall that DFS uses a stack, BFS uses a Queue, and UCS is equivalent to BFS when all costs are equal.

DFS will keep popping a "Right" at the top of the stack. Thus, DFS will go directly right and find the answer in exactly 10 expansions.

BFS (and UCS by extension) will pop from the beginning, and thus "backtrack" to go left after exploring the first "right" action. It will expand more than 100 states.

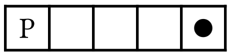
Q1.3 (2 points) A 1×11 grid with 10 dots, each worth 1 calorie. Pacman starts in the middle square.



☐ DFS tree search ☐ BFS tree search ○ None of the above

Solution: The solution for this problem costs more than 10, therefore every algorithm will expand more than 10 states.

Q1.4 (2 points) A 1×5 grid with a single 10-calorie dot. The dot is at the right, and Pacman is at the left.



[✓ DFS tree search] ☐ BFS tree search ○ None of the above

Solution:

DFS tree search will expand the states from left to right, finding the solution with 4 states expanded.

BFS tree search will expand (R=Right, L=Left):
- All cost-1 paths: R
- All cost-2 paths: RL and RR
- All cost-3 paths RLR, RRL, RRR
- All cost-4 paths: RLRL, RLRR, RRLL, RRLR, RRRL, RRRR

RRRR (the solution state) is the 12th state to be dequeued, so by the time you find the goal, 11 states have been expanded.

Q1.5 (1 point) UCS where all actions cost 1 is always equivalent to which search algorithm?

[✓ BFS] ○ DFS ○ A* ○ None

Solution:

UCS explores all cost-1 paths, then all cost-2 paths, etc.

BFS explores all 1-action paths, then all 2-action paths, etc.

If all actions cost 1, these behave exactly the same.

Q2 *Constrained Staff Photos* (9 points)

Four CS 188 TAs want to line up to take staff photos, and Pacman wants to assign each TA to one position.

There are 6 possible positions for TAs to stand, numbered 1 through 6 from left to right. The TAs are the variables, and the positions are the values.

No two TAs can stand in the same position. Not all positions need to be assigned to a TA.

The TAs have special requests:

- Advika (A) does not want anyone to stand directly to her left or right.
- Josh (J) does not want to stand in the leftmost or rightmost position.
- Michael (M) wants to stand directly next to Josh.
- Saathvik (S) does not want to stand directly next to Josh.

Two example assignments are shown below. The invalid assignment violates Advika's, Josh's, and Michael's requests, though Saathvik's request is satisfied.

Valid:

A		M	J		S
1	2	3	4	5	6

Invalid:

J			S	M	A
1	2	3	4	5	6

Q2.1 (1 point) Which type of constraint can be used to represent Advika's request in a single constraint?

○ Unary ○ Binary [✓ Higher-order]

Solution:

To check that nobody is to Advika's left/right, we need to check every TA to make sure that none of them are directly left/right of Advika.

The TAs are variables. Since we have to check more than 2 variables for this constraint, it is a higher-order constraint.

Q2.2 (1 point) Which type of constraint can be used to represent Josh's request in a single constraint?

[✓ Unary] ○ Binary ○ Higher-order

Solution:

This is unary because this constaint only involves 1 variable, Josh. Josh's constraint does not involve checking any other TA's value.

Q2.3 (2 points) For this subpart, no variables are assigned, and every variable's domain is $\{1, 2, 3, 4, 5, 6\}$.

Pacman uses LCV (least constraining value) with forward checking to assign Advika first. Select all value(s) that LCV could assign to A.

[✓ 1] ☐ 2 ☐ 3 ☐ 4 ☐ 5 [✓ 6]

Solution:

For each possible value, try assigning that value, run forward checking and see how many values get eliminated from other variables' domains. The LCV(s) are the possible value(s) that eliminate the fewest values from other variables' domains.

If Advika picks Position 1, then every TA loses 2 positions from their domain (1 and 2). Likewise, if Advika picks Position 6, then every TA loses 2 positions from their domain (5 and 6).

If she picks any other value, then every TA loses 3 spots. For example, if Advika picks Position 2, then every TA loses 1, 2, and 3 from their domains.

Therefore, the least-constraining values are 1 and 6, since they remove the fewest values from other variables' domains.

For the next two subparts, consider the partial assignment and domains below:

					A
1	2	3	4	5	6

$J : \{2, 3, 4\}$ $M : \{1, 3, 4\}$ $S : \{1, 3, 4\}$

Q2.4 (2 points) Pacman runs arc consistency to check the arc $(S \to J)$.

Select all values that **remain** in the domain of S.

[✓ 1] ☐ 3 [✓ 4] ○ None of the above

Solution: When we process an arc, we remove all values in the tail's domain that would result in the head having no valid values.

- For $S = 1$, J can take the value of 4.
- For $S = 3$, J has no valid values— 2 and 4 would be removed from applying S's request, and 3 would be removed because they cannot stand in the same position.
- For $S = 4$, J can take the value of 2.

We would therefore prune 3 from S's domain, leaving 1 and 4.

Q2.5 (3 points) Suppose Pacman runs the arc consistency algorithm covered in lecture (AC-3). Select all arcs that are enqueued after processing the $(S \to J)$ arc.

[✓ $(J \to S)$] ☐ $(J \to M)$ [✓ $(M \to S)$]

☐ $(S \to J)$ ☐ $(M \to J)$ ☐ $(S \to M)$ ○ None of the above

Solution: With AC-3, you enqueue all neighbors of the tail variable, if any values are pruned from its domain. So we would add all arcs that end with S. Here, that would be $J \to S$ and $M \to S$.

Partial credit is given if you do not prune any domains in 2.4, in which case no arcs would be enqueued, and you should select "None of the above".

As long as you prune any variable in 2.4, the solution to 2.5 is as given.

Q3 *Game o' f(Thrones)* **(9 points)**

Pranav and Catherine are playing a game. Pranav acts as a standard maximizer (represented by triangles), and Catherine is represented by hexagonal nodes. Catherine chooses an action that **minimizes** some function f applied to each of the child nodes.

For example:
- If $f(x) = x$, then Catherine acts as a standard minimizer.
- If $f(x) = -x$, then Catherine acts as a standard maximizer.
- If $f(x) = 0$, then Catherine randomly selects an action, since she will see each of her children as having an equal value of 0.

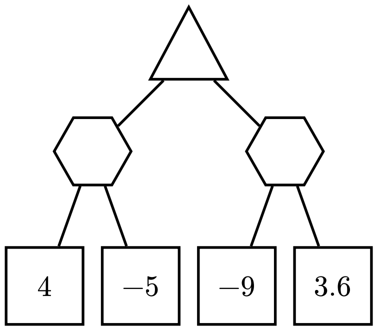
Assumptions:
- Pranav always knows Catherine's strategy and acts optimally.
- Terminal nodes are labeled with Pranav's utility, not Catherine's utility.
- Unless otherwise stated, terminal nodes are unbounded (can take any real value).

Q3.1 (1 point) For this subpart only, consider the tree to the right:

If Catherine chooses an action that minimizes the function

$$f(x) = \begin{cases} 1 \text{ if } x \geq 0 \\ 0 \text{ if } x < 0 \end{cases}$$

what utility does Pranav receive in this game tree, assuming he acts optimally?

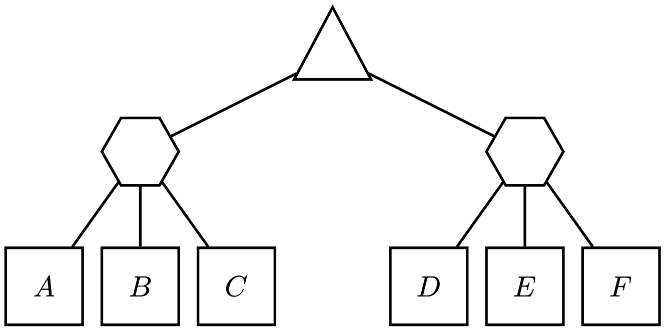


○ 4 [✓ −5] ○ −9 ○ 3.6

Solution: $f(4) = 1, f(-5) = 0, f(-9) = 0, f(3.6) = 1$

Since Catherine minimizes f, she will choose -5 for the left hexagonal node and -9 for the right hexagonal node. Pranav will then choose the maximum of -5 and -9 which is -5.

Q3.2 (2 points) For this subpart only, consider the tree below:



If Catherine chooses an action that minimizes the function

$$f(x) = (x - 188)^2$$

what values of A make it possible to prune B? If no values of A cause B to be pruned, leave both boxes blank and bubble None.

Assume that nodes are evaluated from left to right, and we prune on equality.

$\boxed{188} \leq A \leq \boxed{188}$ ○ None

Solution: The minimum value of f is attained at $x = 188$, so if $A = 188$, the hexagonal node already has minimized f. Since we prune on equality, B (and also C) can be pruned if $188 \leq A \leq 188$.

For the remaining subparts, consider any game tree with alternating maximizer and hexagonal nodes, not necessarily the tree above.

Assumptions for the remaining subparts:
- Both Pranav and Catherine evaluate the entire game tree (no depth-limiting).
- No alpha-beta pruning takes place.

Q3.3 (2 points) Catherine minimizes the same function f as above:

$$f(x) = (x - 188)^2$$

For some given game tree, Pranav's utility when Catherine minimizes $f(x) = (x - 188)^2$ is _______ strictly less than Pranav's utility when Catherine is a standard minimizer.

○ always ○ sometimes [✓ never]

Solution: Proof by contradiction: Assume Catherine can achieve a lower utility by minimizing $f(x)$ compared to acting as a standard minimizer. Then, that means Catherine's optimal policy as a minimizer isn't truly an optimal policy. Since we have a contradiction, Catherine cannot achieve a strictly lower utility by using $f(x)$.

Q3.4 (1 point) Which type of game can model this scenario for **all** functions f?

○ Minimax
[✓ ● Multi-Agent Utilities]
○ Expectimax
○ None of the above

Solution: We can give both agents their own utility to maximize. Pacman can try to maximize his original utility, x. And Catherine can try to maximize a separate utility that we define as $-f(x)$

Q3.5 (3 points) For this subpart only, assume the values at all leaf nodes are strictly positive.

Which of the following functions will cause Catherine to always act like a standard minimizer? Select all that apply.

☐ $\frac{1}{x}$
[✓ $\text{ReLU}(x)$]
[✓ $x + 5$]
[✓ $\log(x)$]
[✓ e^x]
[✓ $x - 5$]
○ None of the above

Solution: Any monotonically increasing function will order the utility of Catherine's children in the same fashion as the identity function $f(x) = x$ which results in her acting like a standard minimizer. $\frac{1}{x}$ is the only function given which is not monotonically increasing on the positive real numbers.

Counterexample: Suppose a hexagon node has two children, 5 and 6. The minimizer chooses 5. If we apply $f(x) = \frac{1}{x}$, then the minimizer chooses 6 because $\frac{1}{6} < \frac{1}{5}$.

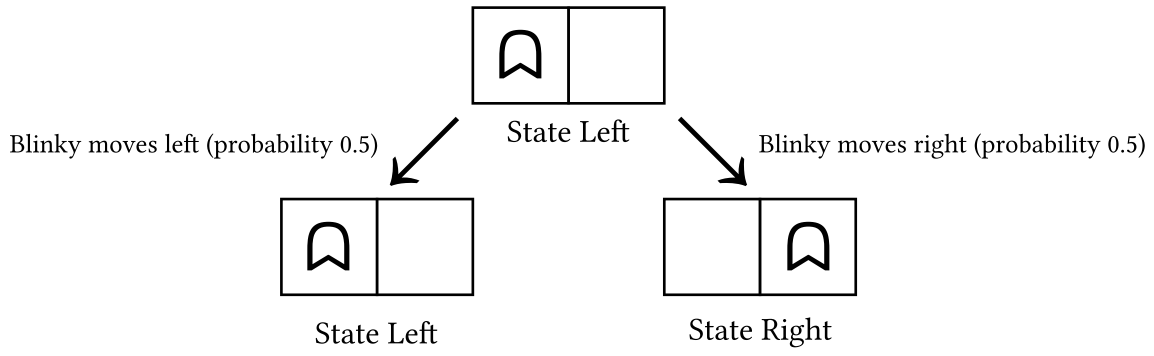
Note: $x - 5$ could cause Catherine to see some negative numbers, but the relative ordering of the numbers is the same, so Catherine still chooses the action that a minimizer would.

Q4 *Busted!* **(14 points)**

Consider a 1×2 grid where Blinky can be in the left or right square. At each time step, Blinky will move left with probability 0.5 or move right with probability 0.5. If the move would result in Blinky moving off the grid, Blinky stays in the same position.

State `Left` represents Blinky in the left square, and state `Right` represents Blinky in the right square.

An example is shown below that starts in state `Left`.


State Left
Blinky moves left (probability 0.5)
Blinky moves right (probability 0.5)
State Left
State Right

Pacman is trying to defeat Blinky and has 3 possible actions: `Bust Left`, `Bust Right`, and `Don't Bust`. Note that Pacman does not have control over Blinky.

`Bust Left` and `Bust Right` give a reward of $+1$ for busting correctly (the same square as Blinky), and -1 for busting incorrectly. The `Bust Left` and `Bust Right` actions transition into a terminal state `X` where no further actions or rewards are available, regardless of whether the bust was correct.

The `Don't Bust` action gives a reward of 0. Then, Blinky moves left or right, and Pacman can take another action.

Q4.1 (4 points) Fill in the table for the transition and reward functions (some rows have been omitted). If a transition (s, a, s') occurs with probability 0, write "N/A" in the box for $R(s, a, s')$.

Solution: The full table is shown below, with rows asked in the question highlighted in yellow.

s	a	s'	$T(s,a,s')$	$R(s,a,s')$
`Left`	`Bust Left`	`Left`	0	N/A
`Left`	`Bust Left`	`Right`	0	N/A
`Left`	`Bust Left`	`X`	1	$+1$
`Left`	`Bust Right`	`Left`	0	N/A
`Left`	`Bust Right`	`Right`	0	N/A
`Left`	`Bust Right`	`X`	1	-1
`Left`	`Don't Bust`	`Left`	0.5	0
`Left`	`Don't Bust`	`Right`	0.5	0
`Left`	`Don't Bust`	`X`	0	N/A
`Right`	`Bust Left`	`Left`	0	N/A
`Right`	`Bust Left`	`Right`	0	N/A
`Right`	`Bust Left`	`X`	1	-1
`Right`	`Bust Right`	`Left`	0	N/A
`Right`	`Bust Right`	`Right`	0	N/A
`Right`	`Bust Right`	`X`	1	$+1$
`Right`	`Don't Bust`	`Left`	0.5	0
`Right`	`Don't Bust`	`Right`	0.5	0
`Right`	`Don't Bust`	`X`	0	N/A

Q4.2 (2 points) If we were to also include the omitted rows, how many rows are in the full table?

Note: Include rows where $T(s, a, s') = 0$.

○ 8 ○ 12 [✓ 18] ○ 24 ○ 30

Solution: 18. There are 2 states with possible actions, 3 actions from those states, and 3 possible successor states, which yields $2 \times 3 \times 3 = 18$. Note that there are no rows where the first column s is `X` because there are no actions available from `X`. (Even if you did do this, you would get $3 \times 3 \times 3 = 27$ which is not an answer choice.)

Q4.3 (3 points) Suppose that at time step $t = 0$, Pacman believes Blinky is in the `Left` square with probability p and is in the `Right` square with probability $1 - p$. Recall that Blinky moves left or right, each with probability 0.5, at each time step.

After one time step, with what probability does Pacman believe Blinky is in the `Left` square?

○ 0 ○ $0.5p$ ○ p [✓ 0.5] ○ $1 - p$ ○ 1

Solution: Note that if Blinky is in the `Left` square, then he moves left with probability 0.5, ending in `Left`, and moves right with probability 0.5, ending in `Right`. The same logic applies if Blinky started in `Right`.

Assume the current time step is t and the state is s_t.

$B(s_t = \text{Left}) = p$

$B(s_t = \text{Right}) = 1 - p$

$P(s_{t+1} = \text{Left} \mid s_t = \text{Left}) = 0.5$

$P(s_{t+1} = \text{Right} \mid s_t = \text{Left}) = 0.5$

$P(s_{t+1} = \text{Left} \mid s_t = \text{Right}) = 0.5$

$P(s_{t+1} = \text{Right} \mid s_t = \text{Right}) = 0.5$

The time update step is

$$\begin{aligned} B'(s_{t+1} = \text{Left}) &= P(s_{t+1} = \text{Left} \mid s_t = \text{Left})B(s_t = \text{Left}) \\ &+P(s_{t+1} = \text{Left} \mid s_t = \text{Right})B(s_t = \text{Right}) \\ &= 0.5p + 0.5(1 - p) \\ &= 0.5 \end{aligned}$$

and similarly for $B'(s_{t+1} = \text{Right}) = 0.5$.

Since there is no evidence, the observation update is not required, and the probabilities are already normalized, so $B(s_{t+1} = \text{Left}) = 0.5$.

For the remaining subparts, suppose Pacman observes an imperfect sensor to gain information about Blinky's location.

At any given time step t, the sensor's observation (o_t) matches Blinky's location (s_t) with probability 0.7 and does not match Blinky's location with probability 0.3.

Q4.4 (2 points) Fill in the table for $P(O_t \mid S_t)$.

o_t	s_t	$P(o_t \mid s_t)$
`Left`	`Left`	**Solution:** 0.7
`Left`	`Right`	**Solution:** 0.3
`Right`	`Left`	**Solution:** 0.3
`Right`	`Right`	**Solution:** 0.7

Q4.5 (1 point) Suppose the sensor reports that Blinky is in the **`Left`** square and the discount factor is $\gamma < 1$. What is the optimal action given this observation?

[✓ **`Bust Left`**] **`Bust Right`** **`Don't Bust`**

Solution: Since the sensor is correct a majority of the time, the best thing to do given this observation is to **`Bust Left`**, which gives reward $+1$ more often than it gives -1. **`Bust Right`** will more often give -1 than $+1$, and **`Don't Bust`** will always give a reward of 0.

If you **`Don't Bust`**, the transition model means that the current sensor report gives you no additional information about the next state, since Blinky's next state does not depend on his current state. This means its better fo Pacman to act now, instead of allowing γ to decay his future reward.

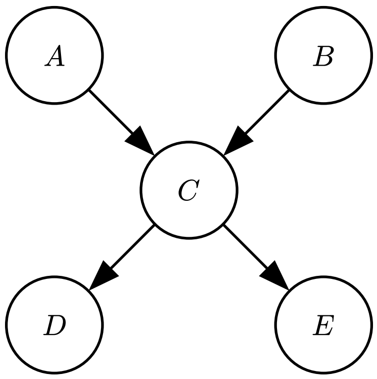
Q4.6 (2 points) Suppose the sensor reports that Blinky is in the **`Left`** square. Assuming $\gamma < 1$, what is the expected return when acting optimally given this observation?

0 0.3 [✓ 0.4] 0.5 0.7 1

Solution: As explained in the previous subpart, the optimal action is to **`Bust Left`**. The expected return is then $(+1) \cdot 0.7 + (-1) \cdot 0.3 = 0.4$. Note that $\gamma < 1$ doesn't affect the return here, as we will terminate immediately following this action.

Q5 *Costco: Inference and Free Samples* (15 points)

Consider the Bayes net below. Each random variable is binary (has two possible values).



Suppose we want to compute $P(B \mid +e)$, using variable elimination.

Q5.1 (1 point) We join and eliminate on A first.

What is the size of the factor generated when we join on A (but before we eliminate on A)?

Note: The size of a factor denotes the number of rows in its CPT.

○ 2^0 ○ 2^1 ○ 2^2 [✓ 2^3] ○ 2^4 ○ 2^5

Solution:

The factors before we start are the CPTs:
- $P(A)$
- $P(B)$
- $P(C \mid A, B)$
- $P(D \mid C)$
- $P(+e \mid C)$

Joining on A gives the factor $f(A, B, C)$, which has table size 2^3 (since there are 3 variables, and each variable is binary).

Q5.2 (1 point) We join and eliminate on D next.

What is the size of the factor generated when we join on D (but before we eliminate on D)?

○ 2^0 ○ 2^1 [✓ 2^2] ○ 2^3 ○ 2^4 ○ 2^5

Solution:

The factors before joining on D are:
- $P(B)$
- $P(D \mid C)$
- $P(+e \mid C)$
- $f(B, C)$

Joining on D gives the factor $f(D, C)$, which has table size 2^2.

Q5.3 (1 point) We join and eliminate on C ~~first~~ next (was clarified on the exam).

What is the size of the factor generated when we join on C (but before we eliminate on C)?

○ 2^0 ○ 2^1 [✓ 2^2] ○ 2^3 ○ 2^4 ○ 2^5

Solution:

The factors before joining on C are:
- $P(B)$
- $P(+e \mid C)$
- $f(B, C)$
- $f(C)$

Joining on C gives the factor $f(B, C, +e)$, which has table size 2^2.

Regardless of your previous answers, suppose that the remaining factors after eliminating A, D, and C are $f(B, +e)$ and $P(B)$. Fill in the expression below to derive the desired probability $P(B \mid +e)$.

$$\frac{\text{(i)}}{\sum_{\text{(ii)}} \text{(iii)}}$$

Q5.4 (1 point) Fill in blank (i).

[✓ $f(B, +e)\ \ P(B)$] ○ $f(B, +e)$ ○ $f(b, +e)\ \ P(b)$ ○ $f(b, +e)$

Q5.5 (1 point) Fill in blank (ii).

[✓ b] ○ b, e ○ e

Q5.6 (2 points) Fill in blank (iii).

○ $f(B, +e)\ P(B)$ ○ $f(B, +e)$ [✓ $f(b, +e)\ P(b)$] ○ $f(b, +e)$

Solution:

By Bayes' rule:

$$P(B \mid +e) = \frac{P(B, +e)}{P(+e)}$$

Expanding the denominator into a sum gives:

$$P(B \mid +e) = \frac{P(B, +e)}{\sum_b P(b, +e)}$$

This expression requires knowing the joint distribution. Multiplying the two remaining factors gives the joint distribution over the remaining variables: $P(B, +e) = f(B, +e)\ \ P(B)$.

Substituting in this joint distribution expression into the formula gives:

$$P(B \mid +e) = \frac{f(B, +e)\ \ P(B)}{\sum_b f(b, +e)\ \ P(b)}$$

Now, suppose we want to estimate $P(B \mid +e)$ using sampling.

Q5.7 (2 points) If we use **rejection** sampling, which of these variable orders can be used to produce a valid sample? Select all that apply.

[✓ A, B, C, D, E] [✓ B, A, C, E, D] ☐ C, B, A, E, D

☐ D, E, A, B, C ☐ E, D, C, B, A ○ None of the above

Solution:

The variables must be sampled in a valid topological order. In other words, when there is an arrow from X to Y, then X must be sampled before Y.

Q5.8 (2 points) Suppose $P(+e \mid +c) = P(+e \mid -c) = 0.001$, and $P(-e \mid +c) = P(-e \mid -c) = 0.999$.

If we use **likelihood weighting**, what are the weights of the samples produced?

- [✓ All samples have weight 0.001.]
- ○ All samples have weight 0.999.
- ○ ~~Some~~ Most (was clarified on the exam) samples have weight 0.001, and the remaining samples all have weight 0.999.
- ○ ~~Some~~ Most (was clarified on the exam) samples have weight 0.999, and the remaining samples all have weight 0.001.

Solution:

Likelihood weighting only accepts samples consistent with the evidence, and weights each sample by the probability of the evidence variable given its parents.

The probability of the evidence $+e$ (given its parent C) is always 0.001, so all samples have weight 0.001.

Q5.9 (2 points) Regardless of your previous answers, consider a scenario where likelihood weighting outputs samples that all have the same weight.

You are given many samples generated from likelihood weighting. Select all strategies that compute a consistent estimate for $P(+b \mid +e)$.

- [✓ Number of samples with $+b$, divided by total number of samples.]
- ☐ Number of samples with $+e$, divided by total number of samples.
- [✓ Sum of weights of all samples with $+b$, divided by sum of weights of all samples.]
- ☐ Sum of weights of all samples with $+e$, divided by sum of weights of all samples.
- ○ None of the above

Solution:

The third option is always correct, because it's a weighted count of the samples with $+b$.

Because all samples have the same weight, a count-based estimate gives the same value as a weighted count.

The second and fourth option are incorrect, since all samples generated with likelihood weighting are consistent with the evidence. Therefore, these two options always compute 1.0.

Q5.10 (2 points) For this subpart only, suppose we want to estimate $P(+e \mid +a)$ using **rejection** sampling.

We generate samples by sampling each variable, one at a time. Which of the following variables, when sampled first, results in samples being rejected as early as possible?

[✓ A] ○ B ○ C ○ D ○ E

Solution: Sampling A first makes it so that samples are rejected as early as possible, since our evidence is $+a$.

Q6 *Scandal* (11 points)

Blinky is taking an exam and considers the utility of cheating. We model his thought process as follows:

1. Blinky chooses his level of cheating L, which is a real number from 0 to 1.

   For example, $L = 0$ represents no cheating, $L = 1$ represents the most cheating, and $L = 0.7$ represents a lot of cheating.

2. After the exam, Blinky is either caught ($+c$) or not caught ($-c$). The value of C is sampled from a probability distribution $P(C \mid L)$.
   The probability that Blinky is caught is $P(+c \mid L) = L$, and $P(-c \mid L) = 1 - L$.

3. If Blinky is caught, he receives utility -50. If Blinky is not caught, he receives utility $10 + 100L$.

Q6.1 (1 point) What is Blinky's expected utility if $L = 0$?

○ -50 ○ -40 ○ 5 [✓ 10] ○ 60 ○ 100 ○ 110

Q6.2 (1 point) What is Blinky's expected utility if $L = 1$?

[✓ -50] ○ -40 ○ 5 ○ 10 ○ 60 ○ 100 ○ 110

Q6.3 (2 points) What is Blinky's expected utility if $L = 0.5$?

○ -50 ○ -40 [✓ 5] ○ 10 ○ 60 ○ 100 ○ 110

Solution: For parts 6.1 - 6.3, we plug in the given value of L into the expected utility expression: $L \times (-50) + (1 - L) \times (10 + 100L)$.

6.1: $(0 \times -50) + (1 - 0) \times (10 + 100 \times 0) = 0 + 10 = 10$

6.2: $(1 \times -50) + (1 - 1) \times (10 + 100 \times 0) = -50 + 0 = -50$

6.3: $(0.5 \times -50) + (1 - 0.5) \times (10 + 100 \times 0.5) = -25 + 0.5(60) = -25 + 30 = 5$

Q6.4 (3 points) What L maximizes Blinky's utility?

Hint: A quadratic $ax^2 + bx + c$ achieves its maximum (assuming one exists) at $x = -\frac{b}{2a}$.

0.2

Solution:

In real life, the penalty for getting caught is much more severe than -50, so even though the optimal cheating level is 0.2 in this question, you shouldn't cheat!

The expected utility, given some cheating level L, is represented by:

$$\begin{aligned} &L(-50) + (1 - L)(10 + 100L) \\ &= -50L + 10 + 100L - 10L - 100L^2 \\ &= -100L^2 + 40L + 10 \end{aligned}$$

To find the L that maximizes the utility, we just need to find the value of L that maximizes this expression.

Following the hint, this quadratic achieves its maximum at:

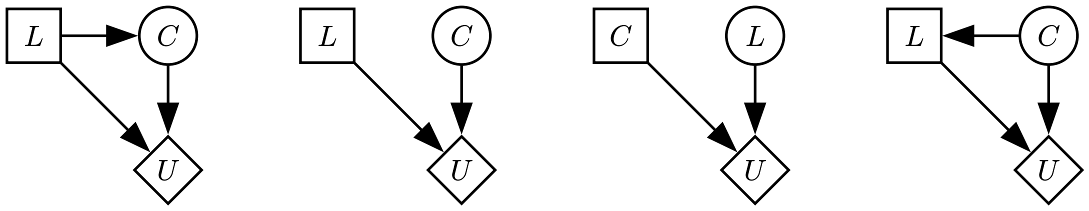
$$\frac{-b}{2a} = \frac{-40}{-200} = 0.2$$

Substituting $L = 0.2$ back into the quadratic yields an MEU of 14 as in the next subpart.

Q6.5 (2 points) Blinky's MEU (maximum expected utility) is 14. What does this number represent?

- [✓ If Blinky takes the exam many times, all with the optimal L, he receives utility 14 on average.]
- ○ If Blinky takes the exam many times, all with the optimal L, he always receives utility 14.
- ○ When Blinky takes the exam with optimal L, the highest utility he can ever receive is 14.
- ○ When Blinky takes the exam with optimal L, the lowest utility he can ever receive is 14.

Q6.6 (2 points) Which decision network best models this scenario?



[✓ Network (i)] ○ Network (ii) ○ Network (iii) ○ Network (iv)

Solution:

Recall square nodes are action nodes; circle nodes are chance nodes; and diamond nodes are utility nodes.

In our scenario:
- L represents an action that Blinky takes (choosing a level of cheating L).
- C represents the probability that Blinky is caught, and it is dependent on his cheating (L).
- U is the utility, which is dependent on whether or not Blinky is caught, and how much cheating he did (U is dependent on L and C.)

Network (i) correctly shows these relationships.

Network (ii) does not show that Blinky's cheating level affects his probability of being caught.

Network (iii) incorrectly labels C, the chance of being caught, as an action node, and also mislabels L, the choice of cheating level, as a chance node.

Network (iv) incorrectly shows L as being dependent of C. Blinky's choice of cheating level L is not probabilistically dependent on his chance of being caught C. Recall that edges represent conditional relationships, just like Bayes Nets.

Q7 *On the Run* **(8 points)**

Oh no! Blinky was caught cheating on his exam and is now trying to escape! At any given time, Blinky's location (L) is either Dwinelle, VLSB, or Stanford.

We want to compute a belief distribution over Blinky's location using particle filtering.

Q7.1 (2 points) Suppose we use 10 particles. We pause the particle filtering algorithm immediately after an observation update.

Which statement implies that $L =$ Stanford with probability 0.9 at this time step?

- [✓ There are exactly 9 particles at Stanford.]
- ○ We add up the un-normalized weights of all the particles at Stanford, and the sum is 9.0.
- ○ We add up the un-normalized weights of all the particles at Stanford, and the sum is 0.9.
- [✓ We add up the normalized weights of all the particles at Stanford, and the sum is 0.9.]

Solution: Since the weights of the particles are normalized at the end of the observation update step, the number of particles represents the likelihood of each location, which means option 1 is correct. The middle two options do not normalize, which is problematic since they could add up to a number greater than 1. Normalizing gives a valid probability distribution, which leads us to the last option also being correct. Credit was given for selecting either option 1 or option 4.

Suppose there are two cameras (C_1, C_2) that are conditionally independent given L. We are only able to observe Blinky's location using these cameras.

Q7.2 (2 points) Which graph always represents the scenario where C_1 and C_2 are conditionally independent given L?

$$\begin{tikzcd} & L & \\ C_1 \arrow[ur] \arrow[rr] & & C_2 \arrow[ul] \end{tikzcd}$$

○ Graph (i)

$$\begin{tikzcd} & L \arrow[dl] & \\ C_1 \arrow[rr] & & C_2 \arrow[ul] \end{tikzcd}$$

○ Graph (ii)

$$\begin{tikzcd} & L \arrow[dl] \arrow[dr] & \\ C_1 & & C_2 \end{tikzcd}$$

[✓ Graph (iii)]

$$\begin{tikzcd} & L & \\ C_1 \arrow[ur] & & C_2 \arrow[ul] \end{tikzcd}$$

○ Graph (iv)

Solution: In Graph (i), C_2 is directly dependent on C_1.

Graph (ii) is not a valid Bayes Net (it is cyclical); also, C_2 is directly dependent on C_1.

Graph (iv) is incorrect, because C_1 and C_2 would be dependent given L (observing a common effect of two variables makes them dependent).

Q7.3 (2 points) In the observation update, how do we calculate the weight w of a particle?

○ $w \leftarrow P(\ell \mid c_1) \cdot P(\ell \mid c_2)$ [✓ $w \leftarrow P(c_1 \mid \ell) \cdot P(c_2 \mid \ell)$]

○ $w \leftarrow P(c_1, c_2 \mid \ell) \cdot P(\ell)$ ○ $w \leftarrow P(c_1 \mid \ell) \cdot P(c_2 \mid c_1)$

Solution: The observation update sets the weight of the particle, which represents a specific location, to "the chance of seeing those observations given the particle's state"— in other words, how likely would we see these observations if this particle were correct?

Since we know the cameras are conditionally independent given L, we can write the answer as $w \leftarrow P(C_1 \mid L) \cdot P(C_2 \mid L)$.

Q7.4 (2 points) At some time during the particle filtering algorithm, we have the four weighted particles shown on the right.

During the re-sampling step, what is the probability that a newly re-sampled particle is at Stanford?

[✓ 1/8] ○ 1/4 ○ 1/2 ○ 1

Solution:

Sum of weights is $0.4 + 0.2 + 0.9 + 0.1 = 1.6$.

Sum of weights on Stanford, s, is 0.2.

Thus, the probability that a resampled particle is in state s is $0.2/1.6 = 1/8$.

	Location (L)	Weight
Particle 1:	Dwinelle	0.4
Particle 2:	Stanford	0.2
Particle 3:	Dwinelle	0.9
Particle 4:	VLSB	0.1

Q8 *Yet Another Machine Learns* **(14 points)**

Q8.1 (2 points) Select all true statements about the Sigmoid function (shown below).

$$\text{Sigmoid}(x) = \frac{1}{1 + e^{-x}}$$

- [✓ Sigmoid outputs values between 0 and 1.]
- ☐ Sigmoid is linear.
- [✓ Sigmoid is non-negative.]
- [✓ Sigmoid is monotonically increasing.]
- ○ None of the above

Solution: The first option is correct, as the denominator will always be strictly greater than 1.

Sigmoid is nonlinear, so the second option is incorrect.

Sigmoid can never be negative, so the third option is correct.

As x increases, $-x$ decreases, so $\exp(-x)$ decreases, and the denominator $1 + \exp(-x)$ decreases. Since we are dividing by a smaller number as x increases, Sigmoid's value will increase as x increases. This is monotonically increasing, so the fourth option is correct.

For the next two subparts, a perceptron classifier has the following weight vectors for three classes ("Sports", "Music", "Books"):

$$w_{\text{Sports}} = [5, 4, 5] \qquad w_{\text{Music}} = [3, 8, 2] \qquad w_{\text{Books}} = [2, 3, 4]$$

Now, the classifier is updated a single time, using a new data point with feature vector $[1, -1, 2]$ and class "Books".

Q8.2 (2 points) What are the new weights for w_{Music} after the update?

○ $[6, 3, 7]$ ○ $[4, 5, 3]$ [✓ $[3, 8, 2]$] ○ $[3, 2, 6]$

○ $[5, 4, 5]$ ○ $[4, 7, 4]$ ○ $[2, 9, 0]$ ○ $[2, 3, 4]$

Solution:

First, attempt to classify the new data point:

- Sports score: $[5, 4, 5] \cdot [1, -1, 2] = 11$
- Music score: $[3, 8, 2] \cdot [1, -1, 2] = -1$
- Books score: $[2, 3, 4] \cdot [1, -1, 2] = 7$

Therefore, we classify the data point as Sports. However, the true classification is Books, so we need to subtract $[1, -1, 2]$ from Sports, and add $[1, -1, 2]$ to Books.

The Music data point is left unchanged.

New weight vectors:

$$\begin{aligned} w_{\text{sports}} &= [5, 4, 5] - [1, -1, 2] = [4, 5, 3] \\ w_{\text{music}} &= [3, 8, 2] \\ w_{\text{books}} &= [2, 3, 4] + [1, -1, 2] = [3, 2, 6] \end{aligned}$$

Q8.3 (2 points) What are the new weights for w_{Books} after the update?

○ $[6, 3, 7]$ ○ $[4, 5, 3]$ ○ $[3, 8, 2]$ [✓ $[3, 2, 6]$]

○ $[5, 4, 5]$ ○ $[4, 7, 4]$ ○ $[2, 9, 0]$ ○ $[2, 3, 4]$

Solution:

See solution to previous subpart.

In the next two subparts, consider a naive Bayes classifier for emails, just like the one from lecture. The classes are “Ham” and “Spam”.

Each email is featurized into a 3-element feature vector using the binary bag-of-words model with words “Free”, “Money”, and “Now”. For example, the email “hello money” is featurized into $[0, 1, 0]$, and the email “now now money” is featurized into $[0, 1, 1]$.

Q8.4 (2 points) We want to add a new feature word "Ring" to the classifier. Select all information about the training dataset needed to compute $P(\text{Ring} = 1 \mid \text{class} = \text{Spam})$.

The choices are not independent, e.g. selecting all choices means you need all 4 values. Assume you don't know the total number of Ham and Spam emails.

- [✓] Number of "Spam" emails containing the word "Ring".
- [ ] Number of "Ham" emails containing the word "Ring".
- [✓] Number of "Spam" emails **not** containing the word "Ring".
- [ ] Number of "Ham" emails **not** containing the word "Ring".
- ○ None of the above

Solution:

Naive Bayes classifies by a count estimate, the simple ratio of $\frac{\text{\# Spam emails containing Ring}}{\text{\# Spam emails in total}}$.

Since we do not have the total number of Spam emails, we need the number of "Spam" emails not containing "Ring", in order to derive the number of "Spam" emails in total.

The table below shows the results of running the classifier on a given test dataset:

Predicted Class	True Class	$P(\text{Predicted Class} \mid \text{True Class})$
Spam	Spam	0.6
Spam	Ham	0.2
Ham	Spam	0.4
Ham	Ham	0.8

In the test dataset, 10% of the data points are "Spam".

Q8.5 (1 point) What is the **precision** of the classifier, treating "Spam" as the positive class?

Hint from lecture slides: Precision $= \frac{\text{TP}}{\text{TP} + \text{FP}}$

○ 0.2 [✓ 0.25] ○ 0.4 ○ 0.6 ○ 0.75 ○ 0.8

Q8.6 (1 point) What is the **recall** of the classifier, treating "Spam" as the positive class?

Hint from lecture slides: Recall $= \frac{\text{TP}}{\text{TP} + \text{FN}}$

○ 0.2 ○ 0.25 ○ 0.4 [✓ 0.6] ○ 0.75 ○ 0.8

Solution:
- TP = True Positive (predicted "Spam", true class is "Spam")
- FP = False Positive (predicted "Spam", true class is Ham)
- FN = False Negative (predicted Ham, true class is "Spam")

Note: A previous version of this solution did not properly account for the proportion of "Spam" in the test dataset, and incorrectly noted 0.75 as the answer to 8.5. This solution has been updated.

Weighting by the proportion of the classes in the test dataset (0.1 "Spam", 0.9 "Ham") in order to get the number of TP, FP, FN, we get:

Precision: $\frac{0.1 \times 0.6}{0.1 \times 0.6 + 0.9 \times 0.2} = \frac{0.06}{0.24} = 0.25.$

Recall: $\frac{0.1 \times 0.6}{0.1 \times 0.6 + 0.1 \times 0.4} = \frac{0.06}{0.1} = 0.6$

The next two subparts are from the Transformers lecture.

Q8.7 (1 point) True or False: In multi-head attention, each attention head can specialize on extracting different information from the input sequence.

[✓ True] ○ False

Solution: This is true, based on empirical research.

If each attention head was extracting the same information from the input sequence, there wouldn't be a benefit to using more attention heads!

Q8.8 (1 point) True or False: Beam search decoding allows LLMs to investigate multiple potential sequences in parallel and select the best one based on a heuristic.

[✓ True] ○ False

Solution: True. Beam search decoding allows LLMs to choose a high likelihood sequence of tokens, while using a heuristic to reduce the amount of searching needed.

The next two subparts are from the guest lectures.

Q8.9 (1 point) Which of these are areas where we see bias in AI? Select all that apply.

[✓ Racial] [✓ Language] ○ None of the above

Solution: AI has a large amount of bias, often a result of the training data used being inherently biased. It is an active field of research to mitigate these problems.

Bias shows up in all outputs, especially image generation and text outputs.

Q8.10 (1 point) Which of the following is **not** a feature of a good model editing technique?

○ Reliability [✓ Incoherence] ○ Generalization ○ Locality

Solution: Model editing is a technique to update a model's knowledge or behavior.

Incoherence is what we try to avoid with model editing; we want coherent outputs and avoid catastrophic forgetting or reduced reasoning capabilities.

Reliability measures a model's ability to recall a fact accurately.

Generalization measures if a model can recall a fact under a variety of circumstances and contexts.

Locality ensures that other facts and capabilities remain consistent after editing; i.e. the editing is "local" and does not affect unintended, other parts of the model.

Q9 *Did You Pay Attention?* (10 points)

Q9.1 (1 point) For unit vectors

$$a = \begin{bmatrix} a_1 \\ \vdots \\ a_n \end{bmatrix} \text{ and } b = \begin{bmatrix} b_1 \\ \vdots \\ b_n \end{bmatrix},$$

what does the dot product $a \cdot b$ represent?

○ The difference in magnitude between a and b.

[✓ ● The similarity between a and b.]

○ The sum of a and b.

Solution: $a \cdot b = \|a\| \, \|b\| \cos\theta$, where θ is the angle between a and b. Since a and b are unit vectors, this reduces to $\cos\theta$, which is higher as the vectors point closer to the same direction, meaning the vectors are more similar.

Q9.2 (2 points) Which of the following is true about Softmax (shown below)?

Note: $\exp(z) = e^z$.

$$\text{Softmax}\left(\begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}\right) = \begin{bmatrix} \frac{\exp(x_1)}{\sum_{i=1}^{n} \exp(x_i)} \\ \vdots \\ \frac{\exp(x_n)}{\sum_{i=1}^{n} \exp(x_i)} \end{bmatrix}$$

[✓ ■ The values in the output of Softmax form a valid probability distribution.]

[✓ ■ If $x_i < x_j$, then after applying Softmax, the ith element is less than the jth element.]

□ Softmax linearly scales all inputs to between 0 and 1.

[✓ ■ Softmax can be used to introduce a non-linearity in a neural network.]

○ None of the above

Solution:

Option 1: Any softmaxed vector will sum to 1 with each entry being between 0 and 1, so that is a valid probability distribution.

Option 2: Softmax maintains the relative ordering of its elements since the exponential function does and so does scaling by a constant factor (the sum of each of the exponentiated individual inputs).

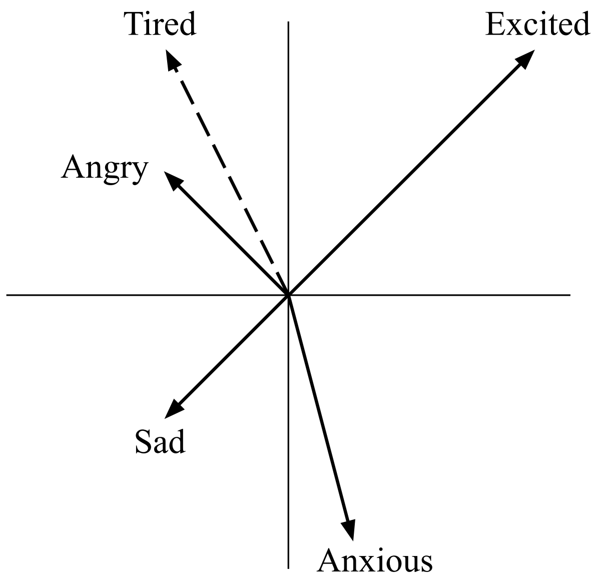
Option 3: Though Softmax does transform its inputs between 0 and 1, it does so nonlinearly.

Option 4: Softmax is nonlinear, so it can serve this purpose.

This question continues on the next page.

Blinky has a word bank containing {"Angry", "Anxious", "Excited", "Sad"}. He uses a feature extraction function f to convert the words into feature vectors, shown below.

Blinky builds a model that takes in a new word "Tired", featurizes it into $f(\text{Tired})$, and wants to find the most similar word in the word bank.



Words in the bank:

$$f(\text{Angry}) = \begin{bmatrix} -1 \\ 1 \end{bmatrix}$$

$$f(\text{Anxious}) = \begin{bmatrix} 0.5 \\ -2 \end{bmatrix}$$

$$f(\text{Excited}) = \begin{bmatrix} 2 \\ 2 \end{bmatrix}$$

$$f(\text{Sad}) = \begin{bmatrix} -1 \\ -1 \end{bmatrix}$$

Blinky's new word:

$$f(\text{Tired}) = \begin{bmatrix} -1 \\ 2 \end{bmatrix}$$

Q9.3 (2 points) Blinky's model uses the following equation, where w is a word in the word bank.

$$\text{SimilarityScore}(\text{Tired}, w) = \frac{\exp(f(\text{Tired}) \cdot f(w))}{\sum_{w' \in \text{ bank}} \exp(f(\text{Tired}) \cdot f(w'))}$$

Which w has the highest SimilarityScore with "Tired"?

[✓ Angry] Anxious Excited Sad

Solution: Though it is possible to calculate this directly, notice that the SimilarityScore is a softmax of the dot product of the feature vectors of the two words. This means that it will be highest with the most similar vector to $f(\text{Tired})$, which is "Angry".

Q9.4 (2 points) For this subpart only, Xavier creates a XavierSimilarityScore model

$$\text{XavierSimilarityScore}(\text{Tired}, w) = \frac{\exp(A\, f(\text{Tired}) \cdot f(w))}{\sum_{w' \in \text{ bank}} \exp(A\, f(\text{Tired}) \cdot f(w'))}$$

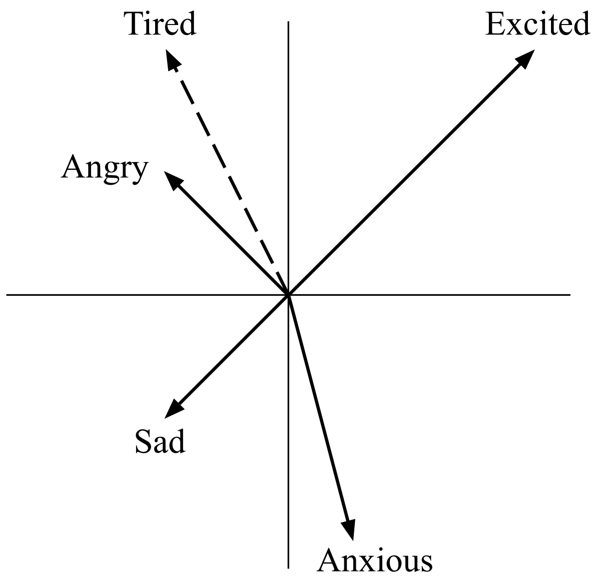
where A is a 2×2 matrix that transforms Tired's feature vector from $f(\text{Tired})$ to $A\, f(\text{Tired})$.

What choice of A causes "Anxious" to have the highest XavierSimilarityScore to "Tired"?

- ○ $\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$, which leaves $f(\text{Tired})$ unchanged.
- ○ $\begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}$, which rotates $f(\text{Tired})$ 90 degrees clockwise.
- [✓ $\begin{pmatrix} -1 & 0 \\ 0 & -1 \end{pmatrix}$, which rotates $f(\text{Tired})$ 180 degrees.]
- ○ $\begin{pmatrix} 2 & 0 \\ 0 & 2 \end{pmatrix}$, which scales $f(\text{Tired})$ by 2.

Solution: Flipping $f(\text{Tired})$ will cause it to end up most similar to $f(\text{Anxious})$.

The diagram is reprinted for your convenience.



Words in the bank:

$$f(\text{Angry}) = \begin{bmatrix} -1 \\ 1 \end{bmatrix}$$

$$f(\text{Anxious}) = \begin{bmatrix} 0.5 \\ -2 \end{bmatrix}$$

$$f(\text{Excited}) = \begin{bmatrix} 2 \\ 2 \end{bmatrix}$$

$$f(\text{Sad}) = \begin{bmatrix} -1 \\ -1 \end{bmatrix}$$

Blinky's new word:

$$f(\text{Tired}) = \begin{bmatrix} -1 \\ 2 \end{bmatrix}$$

Q9.5 (2 points) For this subpart only, Noah creates a NoahSimilarityScore model:

$$\text{NoahSimilarityScore}(\text{Tired}, w) = \frac{\exp(f(\text{Tired}) \cdot B\, f(w))}{\sum\limits_{w' \in \text{ bank}} \exp(f(\text{Tired}) \cdot B\, f(w'))}$$

where B is a 2×2 matrix that transforms each bank word's feature vector from $f(w)$ to $B\,f(w)$.

What choice of B causes "Sad" to have the highest NoahSimilarityScore to "Tired"?

- ○ $\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$, which leaves $f(w)$ unchanged.
- [✓ ● $\begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}$, which rotates $f(w)$ 90 degrees clockwise.]
- ○ $\begin{pmatrix} -1 & 0 \\ 0 & -1 \end{pmatrix}$, which rotates $f(w)$ 180 degrees.
- ○ $\begin{pmatrix} 2 & 0 \\ 0 & 2 \end{pmatrix}$, which scales $f(w)$ by 2.

Solution: Rotating each of the word bank feature vectors 90 degrees clockwise will cause "Sad" to be most similar to "Tired".

Q9.6 (1 point) For this subpart only, consider an AttentionScore model:

$$\text{AttentionScore}(\text{Tired}, w) = \frac{\exp(f(\text{Tired}) \cdot f(w))}{\sum_{w' \in \text{ bank}} \exp(f(\text{Tired}) \cdot f(w'))} \, C \, f(w)$$

where C is a 1×2 matrix.

True or False: There exists a choice of C such that "Anxious" has the highest AttentionScore with "Tired" out of all the words in the word bank.

[✓ True] False

Solution: "Anxious" is the only vector with positive x-coordinate and negative y-coordinate. The SimilarityScore is a positive number between 0 and 1.

Note that $Cf(w)$ is essentially a dot product, since it multiplies a 1×2 matrix with a 2×1 matrix, the same as a row vector by a column vector.

If we positively weight the x-coordinate and negatively weight the y-coordinate by a large amount, every vector except for $f(\text{Anxious})$ will have at least one of its components weighted negatively, and $f(\text{Anxious})$ will have both components weighted positively, making Anxious's AttentionScore significantly higher than every other word's.

For example, $C = \begin{bmatrix} 188 & -188 \end{bmatrix}$ satisfies the requirements.