Solutions

10-601 Machine Learning	**Name:**
Fall 2021	**Andrew Email:**
Exam 3 Practice Problems	**Room:**
December 2, 2021	**Seat:**
Time Limit: N/A	**Exam Number:**

Instructions:

- Fill in your name and Andrew ID above. Be sure to write neatly, or you may not receive credit for your exam.
- Clearly mark your answers in the allocated space **on the front of each page.** If needed, use the back of a page for scratch space, but you will not get credit for anything written on the back of a page. If you have made a mistake, cross out the invalid parts of your solution, and circle the ones which should be graded.
- No electronic devices may be used during the exam.
- Please write all answers in pen.
- You have N/A to complete the exam. Good luck!

Instructions for Specific Problem Types

For "Select One" questions, please fill in the appropriate bubble completely:

Select One: Who taught this course?

● Matt Gormley

○ Marie Curie

○ Noam Chomsky

If you need to change your answer, you may cross out the previous answer and bubble in the new answer:

Select One: Who taught this course?

● Matt Gormley

○ Marie Curie

~~●~~ Noam Chomsky

For "Select all that apply" questions, please fill in all appropriate squares completely:

Select all that apply: Which are scientists?

■ Stephen Hawking

■ Albert Einstein

■ Isaac Newton

□ I don't know

Again, if you need to change your answer, you may cross out the previous answer(s) and bubble in the new answer(s):

Select all that apply: Which are scientists?

■ Stephen Hawking

■ Albert Einstein

■ Isaac Newton

~~■~~ I don't know

For questions where you must fill in a blank, please make sure your final answer is fully included in the given space. You may cross out answers or parts of answers, but the final answer must still be within the given space.

Fill in the blank: What is the course number?

10-601

10-~~7~~601

1 Reinforcement Learning

1.1 Markov Decision Process

Environment Setup (may contain spoilers for Shrek 1)

Lord Farquaad is hoping to evict all fairytale creatures from his kingdom of Duloc, and has one final ogre to evict: Shrek. Unfortunately all his previous attempts to catch the crafty ogre have fallen short, and he turns to you, with your knowledge of Markov Decision Processes (MDP's) to help him catch Shrek once and for all.

Consider the following MDP environment where the agent is Lord Farquaad:

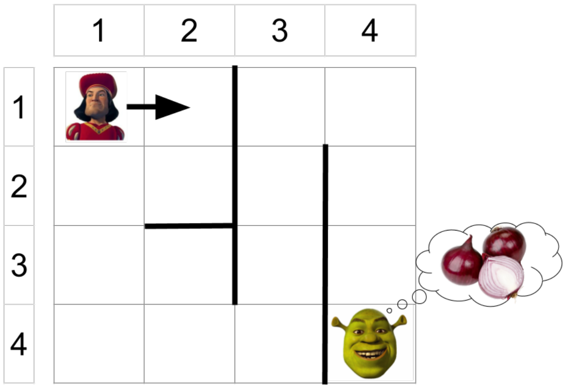


Figure 1: Kingdom of Duloc, circa 2001

Here's how we will define this MDP:

- S **(state space):** a set of states the agent can be in. In this case, the agent (Farquaad) can be in any location (row, col) and also in any orientation $\in \{N, E, S, W\}$. Therefore, state is represented by a three-tuple (row, col, dir), and $S =$ all possible of such tuples. Farquaad's start state is $(1, 1, E)$.
- A **(action space):** a set of actions that the agent can take. Here, we will have just three actions: turn right, turn left, and move forward (turning does not change *row* or *col*, just *dir*). So our action space is $\{R, L, M\}$. Note that Farquaad is debilitatingly short, so he cannot travel through (or over) the walls. Moving forward when facing a wall results in no change in state (but counts as an action).
- $R(s, a)$ **(reward function):** In this scenario, Farquaad gets a reward of 5 by moving into the swamp (the cell containing Shrek), and a reward of 0 otherwise.
- $p(s'|s, a)$ **(transition probabilities):** We'll use a deterministic environment, so this will bee 1 if s' is reachable from s and by taking a, and 0 if not.

1. What are $|S|$ and $|A|$ (size of state space and size of action space)?

   $|S| = 4$ rows × 4 columns × 4 orientations $= 64$
   $|A| = |\{R, L, M\}| = 3$

2. Why is it called a "Markov" decision process? (Hint: what is the assumption made with p?)

   $p(s'|s, a)$ assumes that s' is determined only by s and a (and not any other previous states or actions).

3. What are the following transition probabilities?

$$p((1,1,N)|(1,1,N),M) =$$
$$p((1,1,N)|(1,1,E),L) =$$
$$p((2,1,S)|(1,1,S),M) =$$
$$p((2,1,E)|(1,1,S),M) =$$

$$p((1,1,N)|(1,1,N),M) = 1$$
$$p((1,1,N)|(1,1,E),L) = 1$$
$$p((2,1,S)|(1,1,S),M) = 1$$
$$p((2,1,E)|(1,1,S),M) = 0$$

4. Given a start position of $(1, 1, E)$ and a discount factor of $\gamma = 0.5$, what is the expected discounted future reward from $a = R$? For $a = L$? (Fix $\gamma = 0.5$ for following problems).

   For $a = R$ we get $R_R = 5 * (\frac{1}{2})^{16}$ (it takes 17 moves for Farquaad to get to Shrek, starting with $R, M, M, M, L...$)

   For $a = L$, this is a bad move, and we need another move to get back to our original orientation, from which we can go with our optimal policy. So the reward here is:

   $R_L = (\frac{1}{2})^2 * R_R = 5 * (\frac{1}{2})^{18}$

5. What is the optimal action from each state, given that orientation is fixed at E? (if there are multiple options, choose any)

R	R	M	R
R	R	L	R
M	R	L	R
M	M	L	-

(some have multiple options, I just chose one of the possible ones)

6. Farquaad's chief strategist (Vector from Despicable Me) suggests that having $\gamma = 0.9$ will result in a different set of optimal policies. Is he right? Why or why not?

   Vector is wrong. While the reward quantity will be different, the set of optimal policies does not change. (it is now $5 * (\frac{9}{10})^{16}$) (one can only assume that Lord Farquaad and Vector would be in kahoots: both are extremely nefarious!)

7. Vector then suggests the following setup: $R(s, a) = 0$ when moving into the swamp, and $R(s, a) = -1$ otherwise. Will this result in a different set of optimal policies? Why or why not?

   It will not. While the reward quantity will be different, the set of optimal policies does not change. (Farquaad will still try to minimize the number of steps he takes in order to reach Shrek)

8. Vector now suggests the following setup: $R(s, a) = 5$ when moving into the swamp, and $R(s, a) = 0$ otherwise, but with $\gamma = 1$. Could this result in a different optimal policy? Why or why not?

   This will change the policy, but not in Lord Farquaad's favor. He will no longer be incentivized to reach Shrek quickly (since $\gamma = 1$). The optimal reward from each state is the same (5) and therefore each action from each state is also optimal. Vector really should have taken 10-301/601...

9. Surprise! Elsa from Frozen suddenly shows up. Vector hypnotizes her and forces her to use her powers to turn the ground into ice. Now the environment is now stochastic: since the ground is now slippery, when choosing the action M, with a 0.2 chance, Farquaad will slip and move two squares instead of one. What is the expected future-discounted rewards from $s = (2, 4, S)$?

   Recall that $R_{exp} = max_a E[R(s, a) + \gamma R_{s'}]$

   (notation might be different than in the notes, but conceptually, our reward is the best expected reward we can get from taking any action a from our current state s.)

In this case, our best action is obviously to move forward. So we get

R_{exp} = (expected value of going two steps) + (expected value of going one step)

$E[2_{steps}] = p((4,4,S)|(2,4,S),M) \times R((4,4,S),(2,4,S),M) = 0.2 \times 5 = 1$

$E[1_{step}] = p((4,3,S)|(2,4,S),M) \times (R((4,3,S),(2,4,S),M) + \gamma R_{(4,3,S)})$

where $R_{(4,3,S)}$ is the expected reward from $(4,3,S)$. Since the best reward from here is obtained by choosing $a = M$, and we always end up at Shrek, we get

$E[1_{step}] = 0.8 \times (0 + \gamma \times 5) = 0.8 \times 0.5 \times 5 = 2$

giving us a total expected reward of $R_{exp} = 1 + 2 = 3$

(I will be very disappointed if this is not the plot of Shrek 5)

1.2 Value and Policy Iteration

1. Which of the following environment characteristics would increase the computational complexity per iteration for a value iteration algorithm? Choose all that apply:

   □ Large Action Space

   □ A Stochastic Transition Function

   □ Large State Space

   □ Unknown Reward Function

   □ None of the Above

   A and C (state space and action space). The computational complexity for value iteration per iteration is $O(|A||S|^2)$

2. Which of the following environment characteristics would increase the computational complexity per iteration for a policy iteration algorithm? Choose all that apply:

   □ Large Action Space

   □ A Stochastic Transition Function

   □ Large State Space

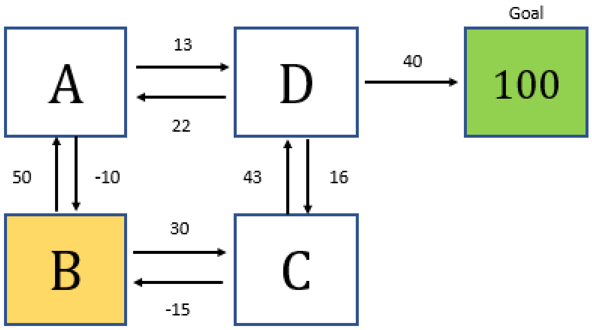
   □ Unknown Reward Function

   □ None of the Above

   A and C again. The computational complexity for policy iteration per iteration is $O(|A||S|^2 + |S|^3)$

3. In the image below is a representation of the game that you are about to play. There are 5 states: A, B, C, D, and the goal state. The goal state, when reached, gives 100 points as reward. In addition to the goal's points, you also get points by moving to different states. The amount of points you get are shown next to the arrows. You start at state

B. To figure out the best policy, you use asynchronous value iteration with a decay (γ) of 0.9.



(i) When you first start playing the game, what action would you take (up, down, left, right) at state B?

Up

(ii) What is the total reward at state B at this time?

50 (immediate reward of 50, and future reward (value at state A) starts at 0)

(iii) Let's say you keep playing until your total values for each state has converged. What action would you take at state B?

C

(iv) What is the total reward at state B at this time?

174 (30 from the immediate action, and 144 from the future reward (value at state C))

4. Let $V_k(s)$ indicate the value of state s at iteration k in (synchronous) value iteration. What is the relation ship between $V_{k+1}(s)$ and $\sum_{s'} P(s'|s,a)[R(s,a,s')+\gamma V_k(s')]$, for any $a \in$ actions? Please indicate the most restrictive relationship that applies. For example, if $x < y$ always holds, please use $<$ instead of $\leq$. Selecting ? means it's not possible to assign any true relationship. (Select the best choice)

$V_{k+1}(s) \,\square\, \sum_{s'} P(s'|s,a)[R(s,a,s') + \gamma V_k(s')]$

○ $=$

○ $<$

○ $>$

○ $\leq$

○ $\geq$

○ ?

E

1.3 Q-Learning

1. For the following true/false, circle one answer and provide a one-sentence explanation:

   (i) One advantage that Q-learning has over Value and Policy iteration is that it can account for non-deterministic policies.

   Circle one: True False

   False. All three methods can account for non-deterministic policies

   (ii) You can apply Value or Policy iteration to any problem that Q-learning can be applied to.

   Circle one: True False

   False. Unlike the others, Q-learning doesn't need to know the transition probabilities (p(s' | s, a)), or the reward function (r(s,a)) to train. This is its biggest advantage.

   (iii) Q-learning is guaranteed to converge to the true value Q* for a greedy policy.

   Circle one: True False

   False. Q-learning converges only if every state will be explored infinitely. Thus, purely exploiting policies (e.g. greedy policies) will not necessarily converge to Q*, but rather to a local optimum.

2. For the following parts of this problem, recall that the update rule for Q-learning is:

$$\mathbf{w} \leftarrow \mathbf{w} - \alpha \left(q(\mathbf{s}, a; \mathbf{w}) - (r + \gamma \max_{a'} q(\mathbf{s}', a'; \mathbf{w}) \right) \nabla_{\mathbf{w}} q(\mathbf{s}, a; \mathbf{w})$$

   (i) From the update rule, let's look at the specific term $X = (r + \gamma \max_{a'} q(\mathbf{s}', a'; \mathbf{w}))$ Describe in English what is the role of X in the weight update.

   Estimate of true total return (Q*(s,a)). This may get multiple answers, so grade accordingly

(ii) Is this update rule synchronous or asynchronous?

Asynchronous

(iii) A common adaptation to Q-learning is to incorporate rewards from more time steps into the term X. Thus, our normal term $r_t+\gamma*max_{a_{t+1}}q(s_{t+1},a_{t+1};w)$ would become $r_t+\gamma*r_{t+1}+\gamma^2\max_{a_{t+2}}q(\mathbf{s}_{t+2},a_{t+2}:\mathbf{w})$ What are the advantages of using more rewards in this estimation?

Incorporating rewards from multiple time steps allows for a more "realistic" estimate of the true total reward, since a larger percentage of it is from real experience. It can help with stabilizing the training procedure, while still allowing training at each time step (bootstrapping). This type of method is called N-Step Temporal Difference Learning.

3. Let $Q(s,a)$ indicate the estimated Q-value of state-action pair (s,a) at some point during Q-learning. Now your learner gets reward r after taking action a at state s and arrives at state s'. Before updating the Q values based on this experience, what is the relationship between $Q(s,a)$ and $r+\gamma\max_{a'}Q(s',a')$? Please indicate the most restrictive relationship that applies. For example, if $x<y$ always holds, please use $<$ instead of $\leq$. Selecting ? means it's not possible to assign any true relationship. (Select the best choice)

$Q(s,a)\ \square\ r+\gamma\max_{a'}Q(s',a')$

○ $=$

○ $<$

○ $>$

○ $\leq$

○ $\geq$

○ ?

F

4. During standard (not approximate) Q-learning, you get reward r after taking action $North$ from state A and arriving at state B. You compute the sample $r+\gamma Q(B,South)$, where $South=\arg\max_a Q(B,a)$.

Which of the following Q-values are updated during this step? (Select all that apply)

○ Q(A, North)

○ Q(A, South)

○ Q(B, North)

○ Q(B, South)

○ None of the above

A

5. In general, for Q-Learning (standard/tabular Q-learning, not approximate Q-learning) to converge to the optimal Q-values, which of the following are true?

   True or False: It is necessary that every state-action pair is visited infinitely often.

   ○ True

   ○ False

   True or False: It is necessary that the discount γ is less than 0.5.

   ○ True

   ○ False

   True or False: It is necessary that actions get chosen according to $\arg\max_a Q(s,a)$.

   ○ True

   ○ False

   (1) **True**: In order to ensure convergence in general for Q learning, this has to be true. In practice, we generally care about the policy, which converges well before the values do, so it is not necessary to run it infinitely often. (2) **False**: The discount factor must be greater than 0 and less than 1, not 0.5. (3) **False**: This would actually do rather poorly, because it is purely exploiting based on the Q-values learned thus far, and not exploring other states to try and find a better policy.

6. Run (synchronous) value iteration on this environment for two iterations. Begin by initializing the value for all states, $V_0(s)$, to zero.

   Write the value of each state after the first ($k = 1$) and the second ($k = 2$) iterations. Write your values as a comma-separated list of 6 numerical expressions in the alphabetical order of the states, specifically $V(A), V(B), V(C), V(D), V(E), V(T)$. Each of the six entries may be a number or an expression that evaluates to a number. Do not include any max operations in your response.

   There is a space below to type any work that you would like us to consider. Showing work is optional. Correct answers will be given full credit, even if no work is shown.

   $V_1(A), V_1(B), V_1(C), V_1(D), V_1(E), V_1(T)$ (Values for 6 states):

   $10, -1, -1, -1, 1, 0$

   $V_2(A), V_2(B), V_2(C), V_2(D), V_2(E), V_2(T)$ (values for 6 states):

$10, 6.8, -2, -0.4, 1, 0$

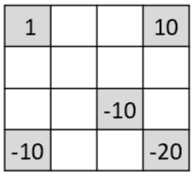
What is the resulting policy after this second iteration? Write your answer as a comma-separated list of three actions representing the policy for states, B, C, and D, in that order. Actions may be Left or Right.

$\pi(B), \pi(C), \pi(D)$ based on V_2 :

Left, Left, Right

Optional work for this problem:

7. Consider a 4x4 Grid World that follows the same rules as Grid World from lecture. Specifically:



1			10
		-10	
-10			-20

- The shaded states have only one action, exit, which leads to a terminal state (not shown) and a reward with the corresponding numerical value printed in that state.
- Leaving any other state gives a living reward, $R(s) = r$.
- The agent will travel in the direction of its chosen action with probability $1 - n$ and will travel in one of the two adjacent directions with probability $n/2$ each.
- If the agent travels into a wall, it will remain in the same state.

Match the MDP setting below with the following optimal policies.

Note: We do not expect you to run value iteration to convergence to compute these policies but rather reason about the effect of different MDP settings.

8. $\gamma = 1.0, n = 0.2, r = 0.1$

A)

1	→	←	10
↓	↑	↑	←
↑	←	-10	↑
-10	↑	←	-20

B)

1	←	→	10
↑	↑	↑	↑
↑	↑	-10	↑
-10	↑	←	-20

C)

1	→	→	10
→	↑	↑	↑
↑	↑	-10	↑
-10	↑	←	-20

D)

1	↑	↑	10
←	↑	↑	←
↑	↑	-10	↑
-10	↑	←	-20

E)

1	←	→	10
↑	↑	↑	↑
↑	←	-10	↑
-10	→	←	-20

F)

1	→	↑	10
↓	↑	↑	↑
↑	←	-10	↑
-10	→	←	-20

○ A

○ B

○ C

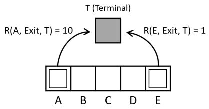
○ D

○ E

○ F

(A). With $\gamma = 1$, the policy will travel to the 10.0 state, and with zero noise, nn, it doesn't have to worry about slipping sideways into negative states.

9. Consider training a robot to navigate the following grid-based MDP environment.



- There are six states, A, B, C, D, E, and a terminal state T.
- Actions from states B, C, and D are Left and Right.
- The only action from states A and E is Exit, which leads deterministically to the terminal state

The reward function is as follows:

- $R(A, Exit, T) = 10$
- $R(E, Exit, T) = 1$
- The reward for any other tuple (s, a, s') equals -1

Assume the discount factor is just 1.

When taking action Left, with 0.8 probability, the robot will successfully move one space to the left, and with 0.2 probability, the robot will move one space in the opposite direction.

When taking action Left, with 0.8 probability, the robot will successfully move one space to the left, and with 0.2 probability, the robot will move one space in the opposite direction.

10. $\gamma = 1.0, n = 0, r = -0.1$

○ A

○ B

○ C

○ D

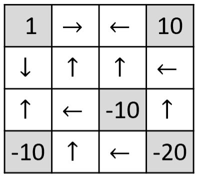
○ E

○ F

(C). With $\gamma = 1$, the policy will travel to the 10.0 state, and with zero noise, nn, it doesn't have to worry about slipping sideways into negative states.

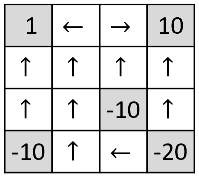
11. Figure repeated for convenience: $\gamma = 0.1, n = 0.2, r = 0.1$

A)



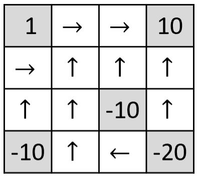
1	→	←	10
↓	↑	↑	←
↑	←	-10	↑
-10	↑	←	-20

B)



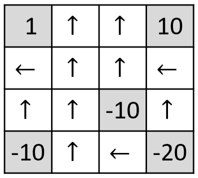
1	←	→	10
↑	↑	↑	↑
↑	↑	-10	↑
-10	↑	←	-20

C)



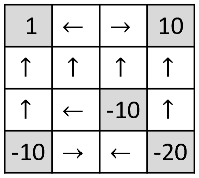
1	→	→	10
→	↑	↑	↑
↑	↑	-10	↑
-10	↑	←	-20

D)



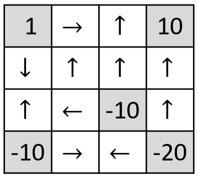
1	↑	↑	10
←	↑	↑	←
↑	↑	-10	↑
-10	↑	←	-20

E)



1	←	→	10
↑	↑	↑	↑
↑	←	-10	↑
-10	→	←	-20

F)



1	→	↑	10
↓	↑	↑	↑
↑	←	-10	↑
-10	→	←	-20

○ A

○ B

○ C

○ D

○ E

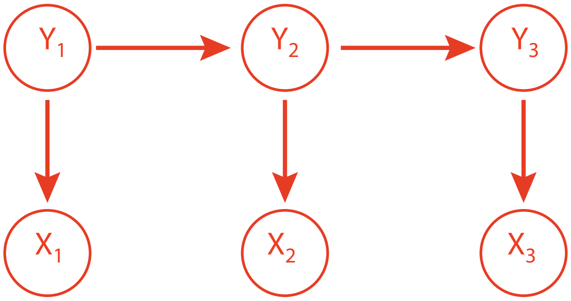
○ F

(E). With low γ, the policy will prefer the closer 1.0 state, but with non-zero noise, it will avoid negative states if at all possible.

2 Hidden Markov Models

1. Recall that both the Hidden Markov Model (HMM) can be used to model sequential data with local dependence structures. In this question, let Y_t be the hidden state at time t, X_t be the observation at time t, $\mathbf{Y}$ be all the hidden states, and $\mathbf{X}$ be all the observations.

   (a) [**2 pts**] Draw the HMM as a Bayesian network where the observation sequence has length 3 (i.e., $t = 1, 2, 3$), labelling nodes with Y_1, Y_2, Y_3 and X_1, X_2, X_3.



   (b) [**2 pts**] Write out the factorized joint distribution of $P(\mathbf{X}, \mathbf{Y})$ using the independencies/conditional independencies assumed by the HMM graph, using terms Y_1, Y_2, Y_3 and X_1, X_2, X_3.
   $P(\mathbf{X}, \mathbf{Y}) =$

   $$P(\mathbf{X}, \mathbf{Y}) = P(Y_1)P(Y_2|Y_1)P(Y_3|Y_2)\prod_{t=1}^{3} P(X_t|Y_t)$$

   (c) [**2 pts**] True or False: In general, we should not include unobserved variables in a graphical model because we cannot learn anything useful about them without observations.
   True **False**

   False.

2. Consider an HMM with states $Y_t \in \{S_1, S_2, S_3\}$, observations $X_t \in \{A, B, C\}$ and parameters $\boldsymbol{\pi} = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$, transition matrix $\boldsymbol{B} = \begin{bmatrix} 1/2 & 1/4 & 1/4 \\ 0 & 1/2 & 1/2 \\ 0 & 0 & 1 \end{bmatrix}$, and emission matrix $\boldsymbol{A} = \begin{bmatrix} 1/2 & 1/2 & 0 \\ 1/2 & 0 & 1/2 \\ 0 & 1/2 & 1/2 \end{bmatrix}$.

   (a) [**3 pts**] What is $P(Y_5 = S_3)$?

$$\begin{aligned} &1 - P(Y_5 = S_1) - P(Y_5 = S_2)\\ =&1 - \frac{1}{16} - 4 \times \frac{1}{32}\\ =&\frac{13}{16}\end{aligned}$$

(b) [**2 pts**] What is $P(Y_5 = S_3 | X_{1:7} = AABCABC)$?

0, since it is impossible for S_3 to output A.

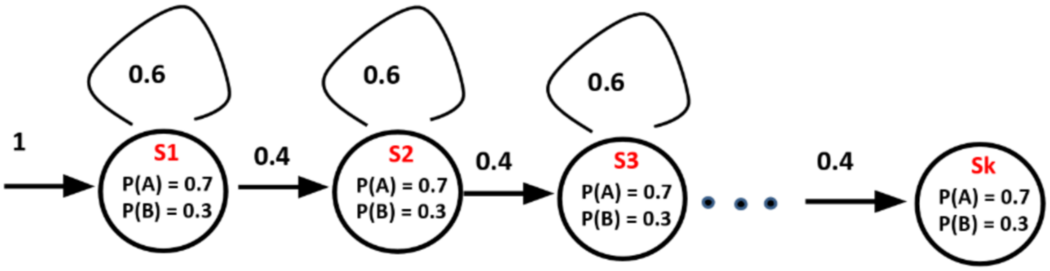
(c) [**4 pts**] Fill in the following table assuming the observation $AABCABC$. The α's are values obtained during the forward algorithm: $\alpha_t(i) = P(X_1, ..., X_t, Y_t = i)$.

t	$\alpha_t(1)$	$\alpha_t(2)$	$\alpha_t(3)$
1			
2			
3			
4			
5			
6			
7			

t	$\alpha_t(1)$	$\alpha_t(2)$	$\alpha_t(3)$
1	$1/2$	0	0
2	$1/8$	$1/16$	0
3	$1/32$	0	$1/32$
4	0	$1/2^8$	$5/2^8$
5	0	$1/2^{10}$	0
6	0	0	$1/2^{12}$
7	0	0	$1/2^{13}$

(d) [**3 pts**] Write down the sequence of $Y_{1:7}$ with the maximal posterior probability assuming the observation $AABCABC$. What is that posterior probability?
$S_1S_1S_1S_2S_2S_3S_3$

posterior probability = 1



3. Consider the HMM in the figure above. The HMM has k states $(s_1, ..., s_k)$. s_k is the terminal state. All states have the same emission probabilities (shown in the figure). The HMM always starts at s_1 as shown. Transition probabilities for all states except s_k are also the same as shown. Once a run reaches s_k it outputs a symbol based on the s_k state emission probability and terminates.

   1. [**5 pts**] Assume we observed the output AABAABBA from the HMM. Select all answers below that COULD be correct.

      ○ $k > 8$

      ○ $k < 8$

      ○ $k > 6$

      ○ $k < 6$

      ○ $k = 7$

      BCDE. It cannot be more that 8 since if it was we would have more than 8 values in the output.

   2. [**9 pts**] Now assume that $k = 4$. Let $P('AABA')$ be the probability of observing AABA from a full run of the HMM. For the following equations, fill in the box with $>, <, =$ or ? (? implies it is impossible to tell).

      (a) $P('AAB')$ □ $P('BABA')$

      $<$, since we must have at least 4 outputs, $P('AAB') = 0$

      (b) $P('ABAB')$ □ $P('BABA')$

      $=$, since all states are the same, it does not matter where the Bs come from in terms of probability

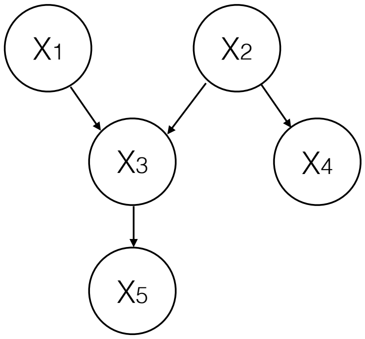
(c) $P('AAABA')$ □ $P('BBAB')$

$>$, $P('BBAB') = 0.4^3 \times 0.3^4 \times 0.7$, and $P('AAABA')$ is a sum over 3 possibilities (we need to stay twice in one of the three states). So $P('AAABA') = 3 \times 0.4^3 \times 0.6 \times 0.7^4 \times 0.3$

3 Graphical Models [16 pts]

1. Consider the following two Bayesian networks.

   (a) Answer whether the following conditional independence is true.



[2 pts] $X_1 \perp X_2 \mid X_3$?
Circle one: Yes No
No.

[2 pts] $X_1 \perp X_4$?
Circle one: Yes No
Yes.

[2 pts] $X_5 \perp X_2 \mid X_3$?
Circle one: Yes No
Yes.

   (b) **[4 pts] Write out the joint probability in a form that utilizes as many independence/conditional independence assumptions contained in the graph as possible. Answer:** $P(X_1, X_2, X_3, X_4, X_5) =$
$P(X_1, X_2, X_3, X_4, X_5) = P(X_1)P(X_2)P(X_3|X_1, X_2)P(X_4|X_2)P(X_5|X_3)$

   (c) **[2 pts]** In the Hidden Markov Model (HMM), a state depends only on the corresponding observation and its previous state.

Circle one: True False

True.

   (d) **[2 pts]** In a graphical model, if $X_1 \perp X_2$, then $X_1 \perp X_2|Y$ for every node Y in the graph.

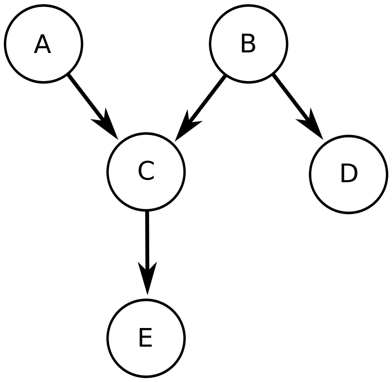
Circle one: True False

False. Consider $X_1 \rightarrow Y \leftarrow X_2$.

(e) **[2 pts]** In a graphical model, if $X_1 \perp X_2 | Y$ for some node Y in the graph, it is always true that $X_1 \perp X_2$.

Circle one: True False

False. Consider $X_1 \leftarrow Y \rightarrow X_2$.



2. Consider the graphical model shown above for questions (a)-(f). Assume all variables are boolean-valued.

(a) [2 pt.] (Short answer) Write down the factorization of the joint probability $P(A, B, C, D, E)$ for the above graphical model, as a product of the five distributions associated with the five variables.

$P(A, B, C, D, E) = P(A)P(B)P(C|A, B)P(D|B)P(E|C)$

(b) [2 pt.] **T or F**: Is C conditionally independent of D given B (i.e. is $(C \perp D)|B$)? Yes

(c) [2 pt.] **T or F**: Is A conditionally independent of D given C (i.e. is $(A \perp D)|C$)? No

(d) [2 pt.] **T or F**: Is A independent of B (i.e. is $A \perp B$)? Yes

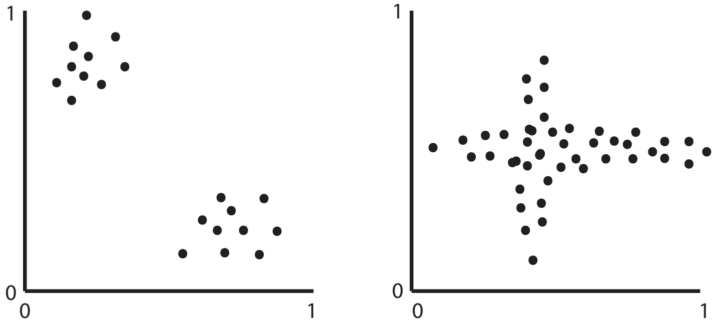
(e) [4 pt.] Write an expression for $P(C = 1|A = 1, B = 0, D = 1, E = 0)$ in terms of the parameters of Conditional Probability Distributions associated with this graphical model.

$$P(C = 1|A = 1, B = 0, D = 1, E = 0) = \frac{P(A = 1, B = 0, C = 1, D = 1, E = 0)}{\sum_{c=0}^{1} P(A = 1, B = 0, C = c, D = 1, E = 0)}$$

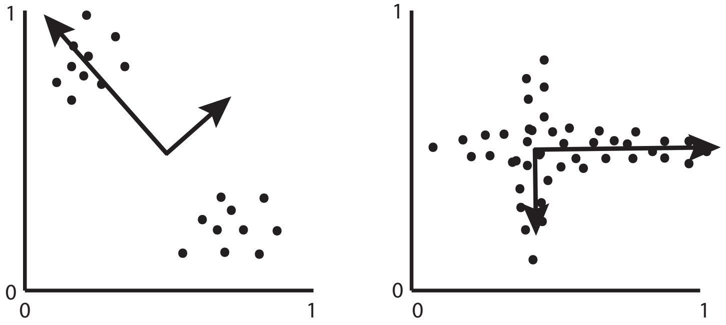
$$= \frac{P(A = 1)P(B = 0)P(C = 1|A = 1, B = 0)P(D = 1|B = 0)P(E = 0|C = 1)}{\sum_{c=0}^{1} P(A = 1)P(B = 0)P(C = c|A = 1, B = 0)P(D = 1|B = 0)P(E = 0|C = c)}$$

4 Principal Component Analysis

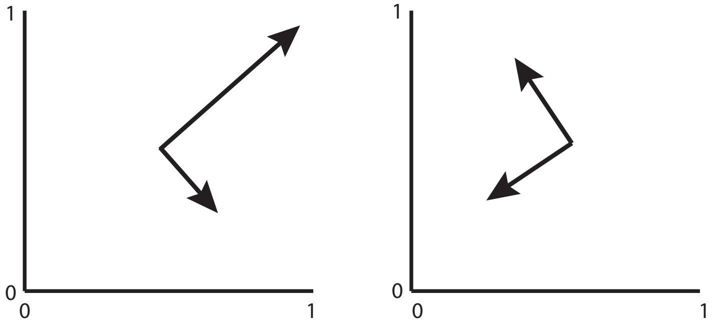
1. (i) [**5 pts**] Consider the following two plots of data. Draw arrows from the mean of the data to denote the direction and relative magnitudes of the principal components.



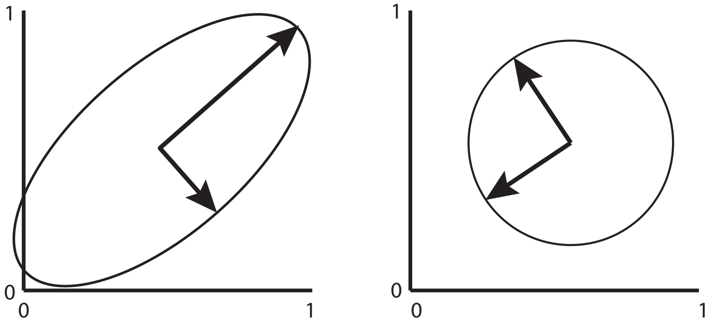
Solution:



(ii) [**5 pts**] Now consider the following two plots, where we have drawn only the principal components. Draw the data ellipse or place data points that could yield the given principal components for each plot. Note that for the right hand plot, the principal components are of equal magnitude.



Solution:



2. Circle one answer and explain.

In the following two questions, assume that using PCA we factorize $X \in \mathbb{R}^{n\times m}$ as $Z^T U \approx X$, for $Z \in \mathbb{R}^{m\times n}$ and $U \in \mathbb{R}^{m\times m}$, where the rows of X contain the data points, the rows of U are the prototypes/principal components, and $Z^T U = \hat{X}$.

(i) [**2 pts**] Removing the last row of U and Z will still result in an approximation of X, but this will never be a better approximation than $\hat{X}$.

Circle one: True False

True.

(ii) [**2 pts**] $\hat{X}\hat{X}^T = Z^T Z$.

Circle one: True False

True.

(iii) [**2 pts**] The goal of PCA is to interpret the underlying structure of the data in terms of the principal components that are best at predicting the output variable.

Circle one: True False

False

(iv) [**2 pts**] The output of PCA is a new representation of the data that is always of lower dimensionality than the original feature representation.

Circle one: True False

False

5 Ensemble Methods

1. **[3pts]** In the AdaBoost algorithm, if the final hypothesis makes no mistakes on the training data, which of the following is correct?

   Select all that apply:

   □ Additional rounds of training can help reduce the errors made on unseen data.

   □ Additional rounds of training have no impact on unseen data.

   □ The individual weak learners also make zero error on the training data.

   □ Additional rounds of training always leads to worse performance on unseen data.

   A. AdaBoost is empirically robust to overfitting and the testing error usually continues to reduce with more rounds of training.

2. **[2pt] True or False:** In AdaBoost weights of the misclassified examples go up by the same multiplicative factor.

   ○ True

   ○ False

   True, follows from the update equation.

Round	$D_t(A)$	$D_t(B)$	$D_t(C)$	$D_t(D)$	$D_t(E)$	$D_t(F)$
1	?	?	$\frac{1}{6}$	?	?	?
2	?	?	?	?	?	?
…						
219	?	?	?	?	?	?
220	$\frac{1}{14}$	$\frac{1}{14}$	$\frac{7}{14}$	$\frac{1}{14}$	$\frac{2}{14}$	$\frac{2}{14}$
221	$\frac{1}{8}$	$\frac{1}{8}$	$\frac{7}{20}$	$\frac{1}{20}$	$\frac{1}{4}$	$\frac{1}{10}$
…						
3017	$\frac{1}{2}$	$\frac{1}{4}$	$\frac{1}{8}$	$\frac{1}{16}$	$\frac{1}{16}$	0
…						
8888	$\frac{1}{8}$	$\frac{3}{8}$	$\frac{1}{8}$	$\frac{2}{8}$	$\frac{3}{8}$	$\frac{1}{8}$

3. **[12pts]** In the last semester, someone used AdaBoost to train some data and recorded all the weights throughout iterations but some entries in the table are not recognizable. Clever as you are, you decide to employ your knowledge of Adaboost to determine some of the missing information.

   Below, you can see part of table that was used in the problem set. There are columns for the Round # and for the weights of the six training points (A, B, C, D, E, and F) at the start of each round. Some of the entries, marked with "?", are impossible for you to read.

In the following problems, you may assume that non-consecutive rows are independent of each other, and that a classifier with error less than $\frac{1}{2}$ was chosen at each step.

(a) **[3pts]** The weak classifier chosen in Round 1 correctly classified training points A, B, C, and E but misclassified training points D and F. What should the updated weights have been in the following round, Round 2? Please complete the form below.

Round	$D_2(A)$	$D_2(B)$	$D_2(C)$	$D_2(D)$	$D_2(E)$	$D_2(F)$
2						

$\frac{1}{8}, \frac{1}{8}, \frac{1}{8}, \frac{1}{4}, \frac{1}{8}, \frac{1}{4}$

(b) **[3pts]** During Round 219, which of the training points (A, B, C, D, E, F) must have been misclassified, in order to produce the updated weights shown at the start of Round 220? List all the points that were misclassified. If none were misclassified, write 'None'. If it can't be decided, write 'Not Sure' instead.

Not sure

(c) **[3pts]** You observes that the weights in round 3017 or 8888 (or both) cannot possibly be right. Which one is incorrect? Why? Please explain in one or two short sentences.

○ Round 3017 is incorrect.

○ Round 8888 is incorrect.

○ Both rounds 3017 and 8888 are incorrect.

NOTE: Please do not change the size of the following text box, and keep your answer in it. Thank you!

C. 3017: weight cannot be 0; 8888: sum of weights should be 1.

4. [**3 pts**] What condition must a weak learner satisfy in order for boosting to work?
   Short answer:

   The weak learner must classify above chance performance.

5. [**3 pts**] After an iteration of training, AdaBoost more heavily weights which data points to train the next weak learner? (Provide an intuitive answer with no math symbols.)
   Short answer:

   The data points that are incorrectly classified by weak learners trained in previous iterations are more heavily weighted.

6. [**3 pts extra credit**] Do you think that a deep neural network is nothing but a case of boosting? Why or why not? Impress us.
   Answer:

   Both viewpoints can be argued. One may view passing a linear combination through a nonlinear function as a weak learner (e.g., logistic regression), and that the deep neural network corrects for errors made by these weak learners in deeper layers. Then again, every layer of the deep neural network is optimized in a global fashion (i.e., all weights are updated simultaneously) to improve performance, which could possibly capture dependencies which boosting could not.

   Almost all coherent answers should be accepted, with full points to those who strongly argue their position with ML ideas.

6 K-Means

1. For **True or False** questions, circle your answer and justify it; for **QA** questions, write down your answer.

   (i) For a particular dataset and a particular k, k-means always produce the same result, if the initialized centers are the same. Assume there is no tie when assigning the clusters.

   ○ True

   ○ False

   Justify your answer:

   True. Every time you are computing the completely same distances, so the result is the same.

   (ii) k-means can always converge to the global optimum.

   ○ True

   ○ False

   Justify your answer:

   False. It depends on the initialization. Random initialization could possibly lead to a local optimum.

   (iii) The cluster assignments for all data points may not change at all between two consecutive iterations in k-means.

   ○ True

   ○ False

   Justify your answer:

   True. This will happen when k-means reaches the global or local optimum.

   (iv) k-means is not sensitive to outliers.

   ○ True

   ○ False

   Justify your answer:

False. k-means is quite sensitive to outliers, since it computes the cluster center based on the mean value of all data points in this cluster.

(v) k in k-nearest neighbors and k-means has the same meaning.

○ True

○ False

Justify your answer:

False. In knn, k is the number of data points we need to look at when classifying a data point. In k-means, k is the number of clusters.

(vi) What's the biggest difference between k-nearest neighbors and k-means?

Write your answer in one sentence:

knn is a supervised algorithm, while k-means is unsupervised.

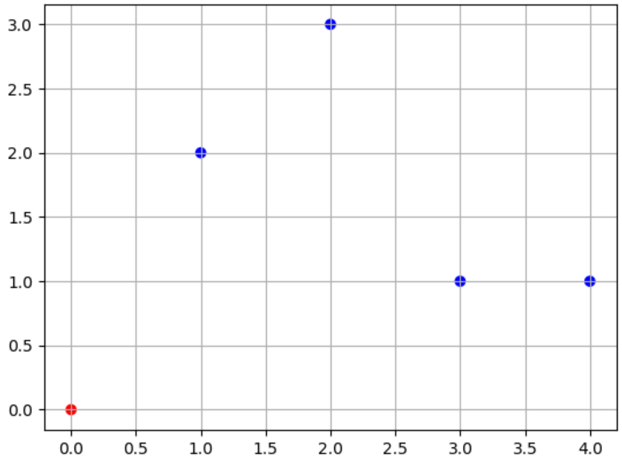
2. In k-means, random initialization could possibly lead to a local optimum with very bad performance. To alleviate this issue, instead of initializing all of the centers completely randomly, we decide to use a smarter initialization method. This leads us to k-means++.

The only difference between k-means and k-means++ is the initialization strategy, and all of the other parts are the same. The basic idea of k-means++ is that instead of simply choosing the centers to be random points, we sample the initial centers iteratively, each time putting higher probability on points that are far from any existing center. Formally, the algorithm proceeds as follows.

```
**Given:** Data set $x^{(i)}, i = 1, \ldots, N$
**Initialize:**
    $\mu^{(1)} \sim \text{Uniform}(\{x^{(i)}\}_{i=1}^{N})$
    For $j = 2, \ldots, k$
        Computing probabilities of selecting each point
            $p_i = \frac{\min_{j'<j} \|\mu^{(j')} - x^{(i)}\|_2^2}{\sum_{i'=1}^{N} \min_{j'<j} \|\mu^{(j')} - x^{(i')}\|_2^2}$
        Select next center given the appropriate probabilities
            $\mu^{(j)} \sim \text{Categorical}(\{x^{(i)}\}_{i=1}^{N}, \mathbf{p}_{1:N})$
```

Note: n is the number of data points, k is the number of clusters. For cluster 1's center, you just randomly choose one data point. For the following centers, every time you initialize a new center, you will first compute the distance between a data point and the center closest to this data point. After computing the distances for all data points, perform a normalization and you will get the probability. Use this probability to sample for a new center.

Now assume we have 5 data points (n=5): (0, 0), (1, 2), (2, 3), (3, 1), (4, 1). The number of clusters is 3 (k=3). The center of cluster 1 is randomly choosen as (0, 0). These data points are shown in the figure below.



(i) [**5 pts**] What is the probability of every data point being chosen as the center for cluster 2? (The answer should contain 5 probabilities, each for every data point)

(0, 0): 0
(1, 2): 0.111
(2, 3): 0.289
(3, 1): 0.222
(4, 1): 0.378

(ii) [**1 pts**] Which data point is mostly liken chosen as the center for cluster 2?

(4, 1) is mostly likely chosen.

(iii) [**5 pts**] Assume the center for cluster 2 is chosen to be the most likely one as you computed in the previous question. Now what is the probability of every data point being chosen as the center for cluster 3? (The answer should contain 5 probabilities, each for every data point)

(0, 0): 0
(1, 2): 0.357
(2, 3): 0.571
(3, 1): 0.071
(4, 1): 0

(iv) [**1 pts**] Which data point is mostly liken chosen as the center for cluster 3?

(2, 3) is mostly likely chosen.

(v) [**3 pts**] Assume the center for cluster 3 is also chosen to be the most likely one as you computed in the previous question. Now we finish the initialization for all 3 centers. List the data points that are classified into cluster 1, 2, 3 respectively.

cluster 1: (0, 0)
cluster 2: (1, 2), (2, 3)
cluster 3: (3, 1), (4, 1)

(vi) **[3 pts]** Based on the above clustering result, what's the new center for every cluster?

center for cluster 1: (0, 0)
center for cluster 2: (1.5, 2.5)
center for cluster 3: (3.5, 1)

(vii) **[2 pts]** According to the result of (ii) and (iv), explain how does k-means++ alleviate the local optimum issue due to initialization?

k-means++ tends to initialize new cluster centers with the data points that are far away from the existing centers, to make sure all of the initial cluster centers stay away from each other.

7 Clustering and Lloyd's Algorithm [18 pts]

7.1 True/False

Circle True or False for the questions below. **If your answer is False, provide a one line justification.**

1. [**2 pts**] In Lloyd's algorithm, the cost always drops after one update step.

   Circle one: True False

   True.

2. [**2 pts**] Lloyd's algorithm is more likely to pick the wrong centers when number of clusters k increases.

   Circle one: True False

   True.

3. [**2 pts**] When α in k-means++ becomes 0, it means random sampling.

   Circle one: True False

   True.

7.2 Lloyd's Method

Consider a dataset with seven points $\{x_1, \ldots, x_7\}$. Given below are the distances between all pairs of points.

	x_1	x_2	x_3	x_4	x_5	x_6	x_7
x_1	0	5	3	1	6	2	3
x_2	5	0	4	6	1	7	8
x_3	3	4	0	4	3	5	6
x_4	1	6	4	0	7	1	2
x_5	6	1	3	7	0	8	9
x_6	2	7	5	1	8	0	1
x_7	3	8	6	2	9	1	0

[**4 pts**] Assume that $k = 2$, and the cluster centers are initialized to x_3 and x_6. Which of the following shows the two clusters formed at the end of the first iteration of Lloyd's algorithm? Circle the correct option.

(a) $\{x_1, x_2, x_3, x_4\}$, $\{x_5, x_6, x_7\}$

(b) $\{x_2, x_3, x_5\}$, $\{x_1, x_4, x_6, x_7\}$

(c) $\{x_1, x_2, x_3, x_5\}$, $\{x_4, x_6, x_7\}$

(d) $\{x_2, x_3, x_4, x_7\}$, $\{x_1, x_5, x_6\}$

Solution: (b).