CS 189/289A Introduction to Machine Learning
Fall 2022 Jennifer Listgarten, Jitendra Malik

DIS10

1 PGMs: Sleeping in Class

In this question, you'll be reasoning about a Dynamic Bayesian Network (DBN), a form of a Probabilistic Graphical Model.

Your favorite discussion section TA wants to know if their students are getting enough sleep. Each day, the TA observes the students in their section, noting if they fall asleep in class or have red eyes. The TA makes the following conclusions:

1. The prior probability of getting enough sleep, S, with no observations, is 0.7.
2. The probability of getting enough sleep on night t is 0.8 given that the student got enough sleep the previous night, and 0.3 if not.
3. The probability of having red eyes R is 0.2 if the student got enough sleep, and 0.7 if not.
4. The probability of sleeping in class C is 0.1 if the student got enough sleep, and 0.3 if not.

(a) Formulate this information as a dynamic Bayesian network that the professor could use to filter or predict from a sequence of observations. If you were to reformulate this network as a hidden Markov model instead (that has only a single observation variable), how would you do so? Give a high-level description (probability tables for the HMM formulation are not necessary.)

Solution: Our Bayesian Network has three variables: S_t, whether the student gets enough sleep, R_t, whether they have red eyes in class, and C_t, whether the student sleeps in class. S_t is a parent of S_{t+1}, R_t, and C_t. The network can be provided pictorally, or fully through conditional probability tables (CPTs.) The CPTs for this problem are given by:
$P(s_0) = 0.7$
$P(s_{t+1}|s_t) = 0.8$
$P(s_{t+1}|\neg s_t) = 0.3$
$P(r_t|s_t) = 0.2$
$P(r_t|\neg s_t) = 0.7$
$P(c_t|s_t) = 0.1$
$P(c_t|\neg s_t) = 0.3$

To reformulate this problem as an HMM with a single observation node, we can combine the 2-valued variables r_t and c_t into a single 4-valued variable, multiplying together the emission probabilities.

(b) Consider the following evidence values at timesteps 1, 2, and 3:

(a) e_1 = not red eyes, not sleeping in class

(b) e_2 = red eyes, not sleeping in class

(c) e_3 = red eyes, sleeping in class

Compute state estimates for timesteps t at 1, 2, and 3; that is, calculate $P(S_t|e_{1:t})$. Assume a prior on $P(s_0)$ that is consistent with the prior in the previous part; that is, $P(s_0) = 0.7$. **Solution:** We can apply the filtering (forward computation) method. We walk through the computation step-by-step:

$$
\begin{aligned}
P(S_0) &= \langle 0.7, 0.3 \rangle \\
P(S_1) &= \sum_{s_0} P(S_1|s_0)P(s_0) \\
&= \langle 0.8, 0.2 \rangle 0.7 + \langle 0.3, 0.7 \rangle 0.3 \\
&= \langle 0.65, 0.35 \rangle \\
P(S_1|e_1) &= \alpha P(e_1|S_1)P(S_1) \\
&= \alpha \langle 0.8 * 0.9, 0.3 * 0.7 \rangle \langle 0.65, 0.35 \rangle
\end{aligned}
$$

After normalizing, we get the following (the rest of the solution(s) will normalize implicitly.)

$$
\begin{aligned}
&= \langle 0.8643, 0.1357 \rangle \\
P(S_2|e_1) &= \sum_{s_1} P(S_2|s_1)P(s_1|e_1) \\
&= \langle 0.7321, 0.2679 \rangle \\
P(S_2|e_1 : e_2) &= \alpha P(e_2|S_2)P(S_2|e_1) \\
&= \langle 0.5010, 0.4990 \rangle \\
P(S_3|e_1 : e_2) &= \sum_{s_2} P(S_3|s_2)P(s_1|e_1 : e_2) \\
&= \langle 0.5505, 0.4495 \rangle \\
P(S_3|e_1 : e_3) &= \alpha P(e_3|S_3)P(S_3|e_1 : e_2) \\
&= \langle 0.1045, 0.8955 \rangle
\end{aligned}
$$

(c) Compute smoothing estimates $P(S_t|e_{1:3})$ for each timestep, using the same evidence as the previous part.

Solution:

First, we do the backwards computations:

$$\begin{aligned}
P(e_3|S_3) &= \langle 0.2 * 0.1, 0.7 * 0.3 \rangle \\
&= \langle 0.02, 0.21 \rangle \\
P(e_3|S_2) &= \sum_{s_3} P(e_3|s_3) P(s_3|S_2) \\
&= \langle 0.02 * 0.8 + 0.21 * 0.2, 0.02 * 0.3 + 0.21 * 0.7 \rangle \\
&= \langle 0.0588, 0.153 \rangle \\
P(e_2 : e_3|S1) &= \sum_{s_2} P(e_2|s_2) P(e_3|s_2) P(s_2|S_1) \\
&= \langle 0.0233, 0.0556 \rangle
\end{aligned}$$

Now, we can combine them with the forwards computation and normalize.

$$\begin{aligned}
P(S_1|e_1 : e_3) &= \alpha P(S_1|e_1) P(e_2 : e_3|S_1) \\
&= \langle 0.7277, 0.2723 \rangle \\
P(S_2|e_1 : e_3) &= \alpha P(S_2|e_1 : e_2) P(e_3|S_2) \\
&= \langle 0.2757, 0.7243 \rangle \\
P(S_3|e_1 : e_3) &= \langle 0.1045, 0.8955 \rangle
\end{aligned}$$

(d) Compare, in plain English, the filtered estimates you computed for timesteps 1 and 2 with the smoothed estimates. How do the two analyses differ?

Solution:

The smoothed analysis shows that the time the student started sleeping poorly is one timestep earlier than filtering only computation by incorporating future observations that indicated lack of sleep at the last step.

2 Markov Decision Processes and Value Computations

In this question, you'll be reasoning about maximizing reward when sequentially making decisions in a Markov Decision Process (MDP), as well as about the Bellman equation - the central equation to solving and understanding MDPs.

Consider the classic gridworld MDP, where an agent starts in cell (1, 1) and navigates around its environment:

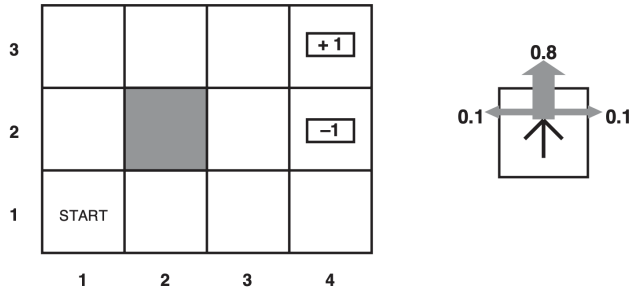

Figure 1: Gridworld MDP with stochastic transition probabilities.

In this world, the agent can take 4 actions in each cell: Up, Down, Left, or Right. The cells are indexed by (horizontal, vertical); that is, cell (4, 1) is in the bottom-right corner. The transition probabilities of the world work as follows: if the agent takes an action, it will move to the cell in the action's direction with probability 0.8, and it will slip to the action's relative right or left direction with probability 0.1 each. If the action (or slipping direction) is into a cell with no traversable tile (i.e., either a border or the wall in cell (2,2)), that action keeps the agent at the cell it is currently at. For example, if the agent is in (3, 1), and it takes the action Up, it will land in cell (3, 2) with probability 0.8, in cell (2, 1) with probability 0.1, and (4, 1) with probability 0.1. If the agent is in cell (1, 3) and takes the action Right, it will land in cell (2, 3) with probability 0.8, in cell (1, 2) with probability 0.1, and (1, 3) with probability 0.1. When the agent reaches either of the defined reward states, at cells (4, 2) and (4, 3), the agent incurs the corresponding reward and the episode terminates.

Recall the Bellman equation for computing the *value*, $V(s)$ of each state in an MDP, where we have a set of actions A, a set of states S, a reward value for each state $R(s)$, the transition dynamics of our world $P(s'|s,a)$, and a discount factor γ:

$$V(s) = R(s) + \gamma \max_{a \in A} \sum_{s' \in S} P(s'|s,a)V(s') \tag{1}$$

Lastly, we'll refer to policies as $\pi(s) = a$, where a policy π prescribes an action to take when in a given state.

(a) Consider an agent that starts in cell (1, 1) and takes the actions Up, Up in timesteps 1 and 2, respectively. Calculate which cells can be reached in each timestep from this action sequence and with what probabilities. How is this calculation similar to the prediction task for an HMM?

Solution:

For the action sequence (Up, Up), we have the following probabilities of being in a cell at timesteps 0, 1, and 2:

Cell	t = 0	t = 1	t = 2
(1, 1)	1	0.1	0.02
(1, 2)	0	0.8	0.24
(1, 3)	0	0	0.64
(2, 1)	0	0.1	0.09
(3, 1)	0	0	0.01

(b) Consider the reward function, $R(s)$, for all states that currently don't have a reward assigned to them (every cell except for (4, 2) and (4, 3).) Define what an optimal policy would be for an agent given the following reward values: (i.) $R(s) = 0$, (ii.) $R(s) = -2.0$, and (iii.) $R(s) = 1.0$. You may assume the discount factor to be a number arbitrarily close to 1, e.g. 0.9999. It may be helpful to draw out the gridworld and actions that should be taken at each state (remember that policies are defined over all states in an MDP!)

Solution:

In part (i), the agent does not incur a penalty for "existing" in the environment, but it does not incur a reward, either. Thus, the agent will be incentivized to avoid the penalty cell at (4, 2) at all costs, but is not in any rush to get to the reward cell at (4, 3.) An optimal corresponding policy (note that there are many, and this is just one example) is thus as follows: Take the action Left in cells (2, 1), (3, 1), and (3, 2); take the action Down in cell (4, 1), take the action Up in cells (1, 1) and (1, 2), and take Right in (1, 3), (2, 3), (3, 3.)

In part (ii), a large penalty is incurred simply for existing in the environment. The penalty is so much larger than the penalty for reaching cell (4, 2), in fact, that the agent is willing to get to that state just to terminate the episode and avoid incurring more huge penalty just for being in the world. The corresponding policy is to take the Right action in cells (1, 1), (2, 1), (3, 1), (1, 3), (2, 3), and (3, 3); and take the Up action in cells (1, 2) and (4, 1).

In part (iii), life is good for the agent for simply existing in the environment; so much so, that the agent is incentivized to never terminate the episode and reach either of the terminal cells. An optimal policy here (note that there are many, and this is just one example) becomes to take the Left action if in cells (3, 2) or (3, 3), the Down action in cells (4, 1), and any action in all of the other cells.

(c) Sometimes MDPs are formulated with a reward function $R(s, a)$ that depends on the action taken, or with a reward function $R(s, a, s')$ that also depends on the outcome state. Write out the Bellman equations for these formulations.

Solution: The tricky part in this problem is to make sure the max and summations are in the right place. For $R(s, a)$, the formulation is:

$$V(s) = \max_{a \in A}[R(s, a) + \gamma \sum_{s' \in S} P(s'|s, a)V(s)]$$

And for $R(s, a, s')$:

$$V(s) = \max_{a \in A} \sum_{s' \in S} P(s'|s, a)[R(s, a, s') + \gamma V(s)]$$