Portfolio Tasks

For AIML429

Document compiled on: May 14, 2024

Contents

The Tasks (Part II)

1: decide whether to make an observation

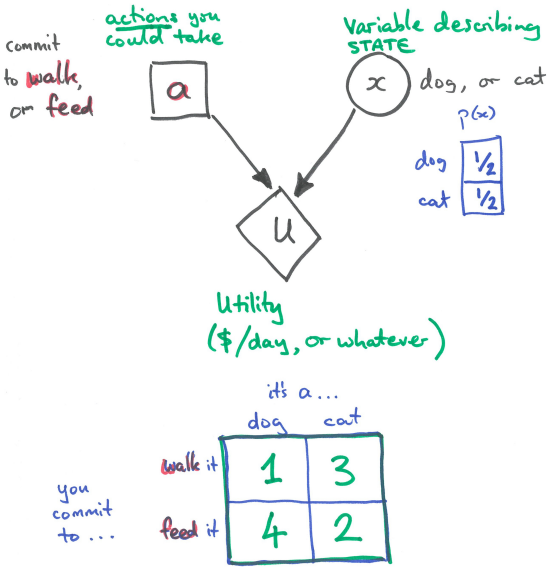
One can add actions and utility nodes to PGMs: together these produce decision networks / influence diagrams, and these enable one to build systems that make decisions about their actions, in service of high expected utility. So:

Suppose someone has a cat, or a dog... but you're not sure which[a]. You commit to either feed or walk, this pet. The utility of each of these depends on whether it's a dog or a cat. I chose the numbers 1,2,3,4 in the table you should see below.

If you *don't* know cat-vs-dog, but believe $\Pr(dog) = 1/2$, the *expected* utility is higher for choosing the action "feed", which would earn you (or the animal, if the utility were its happiness instead of your dollars) 3 utility points ("dollars"), on average. So the system allowed us to decide what to do, under uncertainty.

We can also use such a model to decide *whether to find out* a variable (or perhaps to decide which of several variables to actually observe). Here's a very simple example: what is the expected increase in utility to be gained by observing x before deciding what to do ? This is the value of information , or $VOI(x)$. *i.e.* How much would you pay to find out x before deciding?

[a] NOTHING about this is remotely realistic, but let's leave that aside!



If you don't know x, expected utility

$\bar{u}_{a=\text{walk}} = \frac{1}{2}\cdot 1 + \frac{1}{2}\cdot 3 = 2$

$\bar{u}_{\text{feed}} = \frac{1}{2}\cdot 4 + \frac{1}{2}\cdot 2 = 3$

$\Rightarrow$ you set out to FEED, and expect \$3 /day (or whatever)

2: what to do (continuous action space)

(Generalisation of "archer fish")

Suppose you have a Gaussian prior over the true location of something valuable in a 2-dimensional world: $P(\mathbf{y}) = \mathcal{N}(\mathbf{y}|\boldsymbol{\mu}, \Sigma)$, with mean $\boldsymbol{\mu} = [0, 0]$ and diagonal covariance $\Sigma = [[5, 0], [0, 3]]$. Observations of $\mathbf{y}$ are noise-prone however, due to very noisy sensors: $\hat{\mathbf{y}} = \mathbf{y} + \epsilon$ with $P(\epsilon) = \mathcal{N}(\epsilon|0, \sigma^2\mathbf{1})$ and $\sigma^2 = 0.25$.

Items in this world have a utility which is a function of where they are found: the most valuable sites are near $\mathbf{y} = [-2, 2]$. In general:

$$U(\mathbf{y}) = 100 \exp\left(-\tfrac{1}{2}(\mathbf{y} - \boldsymbol{\mu}_{\text{util}})^T \, \Sigma_{\text{util}}^{-1} \, (\mathbf{y} - \boldsymbol{\mu}_{\text{util}}) \right)$$

♠ in which $\boldsymbol{\mu}_{\text{util}} = [-2, 2]$ and $\Sigma_{\text{util}} = [[1, \tfrac{1}{2}][\tfrac{1}{2}, 1]]$.

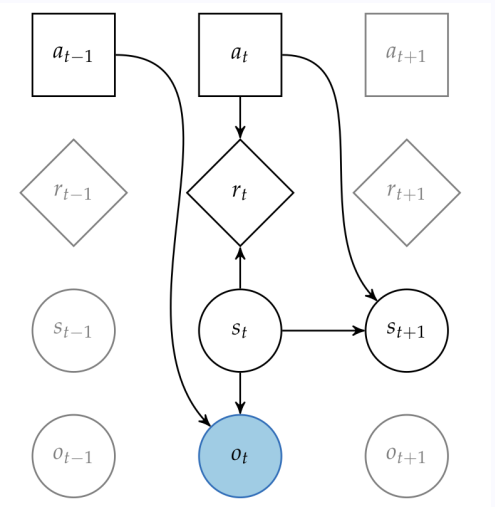
If you detect an item, you can take a shot and try to hit it. Your shots are also not 100 percent on-target however: they actually arrive at a point $\mathbf{y}_{\text{shot}} = \mathbf{y}_{\text{aim}} + \epsilon_{\text{shot}}$ with $P(\epsilon_{\text{shot}}) = \mathcal{N}(\epsilon|0, \sigma^2_{\text{shot}}\mathbf{1})$ and $\sigma^2_{\text{shot}} = 0.05$.

Assume you are "risk neutral".

If an observation is made $\hat{\mathbf{y}} = [1, 3]$, and you had one chance to "shoot" at a spot, what spot $\mathbf{y}_{\text{aim}}$ would it be? Explain your reasoning.

(there are different ways to answer this question, e.g. by noticing simplifying features of the functions being used and arguing mathematically, or by something more brute-force, since the problem is only 2d.)

3: what to do (sequence)



♠

The picture shows part of the PGM describing a Markov model in which the underlying state s unfolds over time, influenced by actions a, which are chosen from a discrete set of options. *(Note that the arrows all repeat and only one of each is shown here).* Here, the observations o are modelled as arising from the current state only (the previous action can sometimes play a role in what is seen too – hence the arrow – but this is not always included). Carrying out an action while in a state may produce an immediate reward r.

You can assume the following: *(i)* at the time of deciding action a_t the agent knows about all previous observations $o_1..o_{t-1}$, actions, and the rewards (if any) that resulted; *(ii)* the sequence is going to end at some fixed time index τ; *(iii)* any parameters of the model have been learned by some sensible procedure – this question is not about learning.

Noting that the bottom two layers consist of an HMM, outline how you might approach the question of how to make the decision, a_t, using the current model. Try to be clear and succinct. You might find parts of Chapter 19 of Algorithms for Decision Making useful in thinking this through.

This is quite a tricky question - at the end of the day there is a surprising depth to questions about how to act under uncertainty. Don't try to "solve it" perfectly, but instead aim to give an account of the main issues at play.

4: different mixture model

Briefly describe and investigate [a] the idea of a mixture model in which the mixture components are not Gaussians, but are instead some other distribution.
In particular, can the EM algorithm still be used to learn parameters of your model from data?

[a] e.g. you might do so experimentally, by adapting the GMM demo notebook we looked at – or choose another way you prefer.

5: sigmoids, generative vs discriminative

- in discriminative training of a single perceptron, we can interpret the sigmoidal activation function

$$y = \frac{1}{1 + \exp(-(\mathbf{w} \cdot \mathbf{x} + w_0))}$$

  as giving[a] a probability of the "1" class, namely $p(y = 1 \mid \mathbf{x}, \mathbf{w})$. In attempted to match targets t in a training set of $(t, \mathbf{x})$ pairs, solution weights can be found by gradient descent of the "log loss" $\log \mathcal{L} = \sum_n t_n \log y_n + (1 - t_n) \log(1 - y_n)$, leading to the weight change rule $\delta w_i = \eta(t_n - y_n)x_{i,n}$

- in a mixture-of-two-Gaussians generative model, the Gaussian parameters θ are straightforward to find in the fully observed (supervised learning) case. If the Gaussians are "spherical"[b], then somewhat beautifully $p(y = 1 \mid \mathbf{x}, \theta)$ turns out to be the same sigmoid function, with weights and bias that are a direct function of the θ.

Would the weights be the **same, or different** in the two cases?
By a clear argument, or a demonstration, defend your answer.

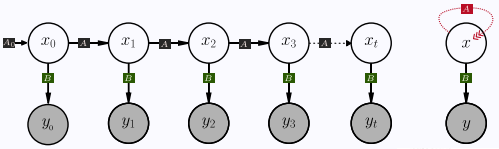
[a] w_0 is the bias weight
[b] also called "isotropic"

6: KL divergence

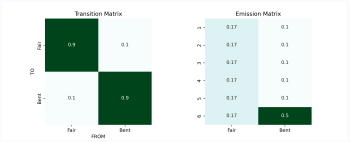
Explain how minimizing the KL divergence is sometimes described as "mode seeking", and at other times as "mode covering" instead. These are very different behaviours!

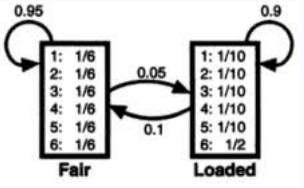
7: Gibbs sampling

HMM:

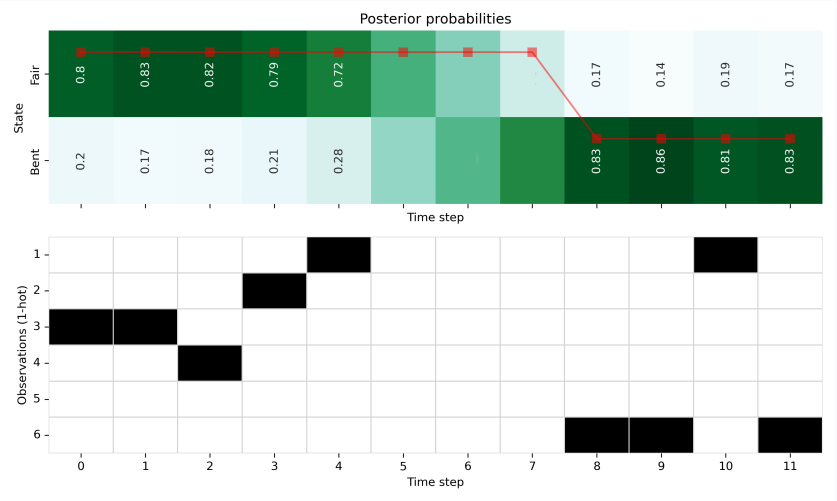


Parameters:





Inference via the 'forward-backward' algorithm (a.k.a. 'belief propagation'):



Use Gibbs Sampling to estimate the hidden state probabilities at $t = 6$.

Hint (non-essential but interesting): in sparse PGMs Gibbs sampling can be very efficiently implemented since one only has to consider the "Markov Blanket"[1] of a node in order to update it. For this question however, an efficient implementation is **not** a priority, and you can evaluate

$$\begin{aligned} p(x_i \mid \mathbf{x}_{\text{rest}}) &\propto p(x_i, \mathbf{x}_{\text{rest}}) \\ &= p(\mathbf{x}) \\ &= \prod_k p(x_k \mid \text{parents}_k) \end{aligned}$$

Provided you keep the model small, $p(\mathbf{x})$ won't be vanishingly tiny and you won't need to do this in log-space.

[1] its parents, its children, and (due to the collider effect) its children's *other* parents.