Bayesian Statistics

Exercises for week 04

1 Cooking with Bayesian inference

5 pt(s)

In this exercise, you'll translate a 'realistic' scenario into a hierarchical Bayesian model. Part of the exercise requires you to search online for function names and ways of specifying data structures in `R`. Oftentimes, you'll find that there are different ways of achieving a particular goal.

Consider the following setting. I'm making dinner according to $K = 4$ different cooking shows. Let's refer to the shows as Gordon, Jamie, Julia and Nigella. Unfortunately, the quality of each tv show's recipes can differ.

- I'm trying N dinners.
- A dinner i is either tasty ($x_i = 1$) or a failure ($x_i = 0$).
- Each cooking show k has its own success probability θ.
- The label $z_i \in \{1, \ldots, K\}$ indicates on which show the dinner i is based.
- The success of the dinner depends only on the success probability θ and the label z.[1]
- Each show has its own 'pseudo-success count' a and 'pseudo-failure count' b.
- For every dinner, I pick a tv show uniformly at random, that is, by drawing a random tv show z from a *categorical* distribution with parameter $\boldsymbol{\alpha} = 1/K \times \mathbf{1}_K$, where $\mathbf{1}_K$ means a vector of length K containing only ones (i.e. each show is equally likely in the prior).

Construct the model:

1. The first step in modeling this scenario is to draw a graphical model (using plate notation). You'll need to think about which parameters depend on which other parameters, and, conversely, which parameters are *independent*, and which parameters are repeating. Also, some parameters will require subscript indices, which are not always shown in the outline above![2]
2. The second step is to specify by which probability distribution each of the dependencies in your model may be described. Clearly indicate the name of the distribution, what parameters go in, what parameter comes out, and any (if applicable) constraints on the outcomes. Use the *generative model* notation as in the lecture (or see e.g. the next exercise).
3. With your model, you can now generate N simulated observations $x_1, \ldots, x_N$. Write an `R` script that does this, for $N = 100$. Begin the script by setting the hyperparameters for each tv show. Using the hyperparameters, you can draw the next layer of parameters (for each..?). With those parameters, you can draw the actual observations $\mathbf{x}$. Summarize your output as follows: show for each tv show how many dinners it has in the simulated data, and how many of these were a success. Show this output for the following three cases, and show which hyperparameter settings you used to encode the following beliefs:
   (a) You have strong prior beliefs (hope?) that the recipes by Jamie were a success.
   (b) You think it doesn't matter in which tv show I watched; you think it'll be a failure anwyay.

[1] And not on my cooking skills, obviously.

[2] Tip: the `.tex` file for this exercise is also available on Blackboard. You can use it to see how I draw the diagram in the next exercise, using `Tikz`.

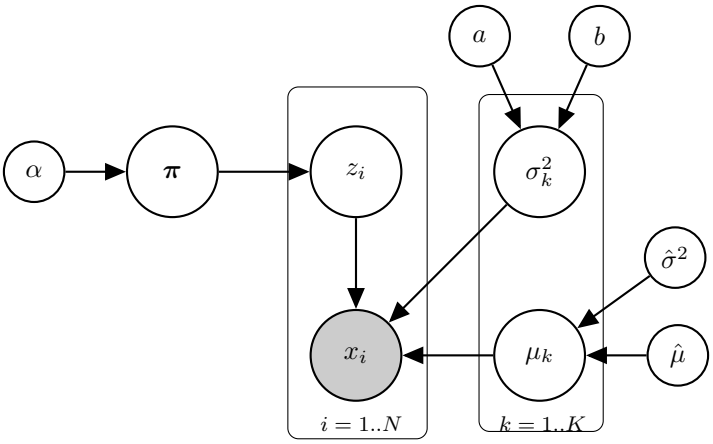

Figure 1: A one-dimensional Gaussian mixture model.

(c) You have absolutely no idea what these tv dinner shows are, so you have uniform prior beliefs about each of their success probabilities.

Verify that the generated data is as you would expect for your hyperparameter settings!

Some tips: `R` does not (by default) contain an implementation for the Categorical distribution. However, you can use the function `sample`.

2 A 1-D Gaussian mixture model

5 pt(s)

The following model is quite famous, as it is often used in modeling data. It is known as the Gaussian mixture model and describes that K distributions together explain the N data points. Any one data point is still generated by a single distribution. The model is used as a *clustering algorithm* (see the Data Mining course!), because we can use it to probabilistically learn to which cluster (= mixture component = a single Gaussian) each data point belongs. The algorithm can be seen as the probabilistic/Bayesian generalization of the *K-means algorithm*.

The model is fully described by the equations below, and further illustrated by the diagram in Fig. 1. For now, we consider the one-dimensional case (for a D-dimensional case, the x's and μ's become vectors of length D and the σ^2's become positive definite matrices of size $D \times D$).

$$\begin{aligned}
\sigma_k^2 | a, b &\sim \text{Gamma}\left(\sigma_k^2 \mid a, b\right) && k = 1..K \\
\mu_k | \hat{\mu}, \hat{\sigma}^2 &\sim \text{Gaussian}\left(\mu_k \mid \hat{\mu}, \hat{\sigma}^2\right) && k = 1..K \\
\boldsymbol{\pi} | \alpha &\sim \text{Dir}\left(\pi \mid \boldsymbol{\alpha}\right) && \boldsymbol{\alpha} = \alpha \times \mathbf{1}_K \\
z_i | \boldsymbol{\pi} &\sim \text{Cat}\left(z_i \mid \boldsymbol{\pi}\right) && i = 1..N \\
x_i | \mu_k, \sigma_k^2, z_i &\sim \text{Gaussian}\left(x_i \mid \mu_{z_i}, \sigma_{z_i}^2\right) && i = 1..N \ ,
\end{aligned}$$

in which $\mathbf{1}_K$ means a vector of length K, with 1 in each element.

On Blackboard, you'll find an `R` script that is able to generate data from this model (`exercise04_2.R`). You can use it to explore the effect of changing the hyperparameters $\alpha, a, b, \hat{\mu}, \hat{\sigma}^2$. Be sure to revert all hyperparameters to their original settings before moving to the next sub-question.

1. Typically, the parameter of interest in such a model, is $\boldsymbol{\pi}$. We want to learn this, given the actual data. What is the role of the vector $\boldsymbol{\pi}$ in this model? How does it change if you change α from a value smaller than one, to one, to a value (much) larger than one? There is a question on the Blackboard discussion board, which has an answer that includes a link that may be very helpful here.

2. What is the effect of the hyperparameter a? How does the generated data change if you adjust it?

3. What is the effect of the hyperparameter $\hat{\sigma}^2$? What happens if you change it to a very small value? If the actual data looked like that, what would be the consequence for learning $\boldsymbol{\pi}$?