DS 102 Data, Inference, and Decisions **Fall 2019**

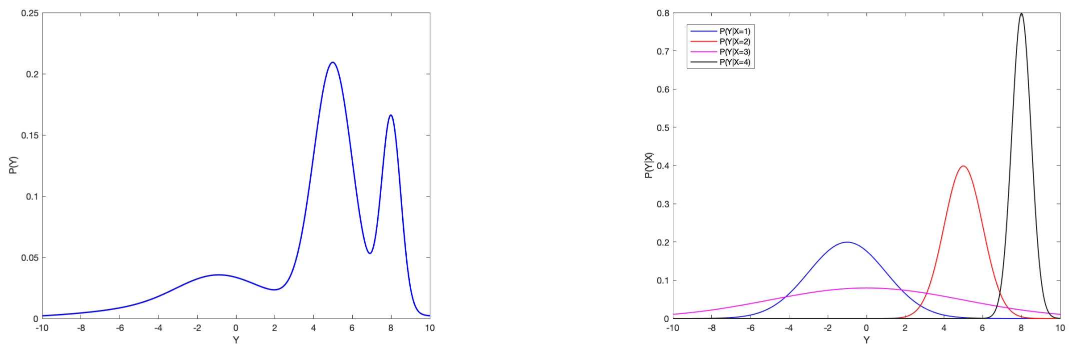
Lecture 9: Bayesian Hierarchical Models

Lecturer: Jacob Steinhardt

In the last lecture, we introduced the idea of modeling hidden structure in our data, and saw an example of a Gaussian mixture model (GMM) where, indeed, accounting for hidden structure made modeling the remaining signal more intuitive. In this lecture, we will begin by revisiting this motivating GMM example in more detail and defining latent variable models in more generality. Then, we will introduce the *Expectation-Maximization* Algorithm as a way of performing inference in such latent variable models.

9.1 Gaussian Mixture Models

We begin by recalling the idea of GMMs. Suppose we have a random variable Y with an unknown distribution $\mathbb{P}(Y)$ as in Figure 9.1(a) that does not appear to be from any class of distributions that we are familiar with. We would still like to come up with a model of the distribution that we can interpret and sample form easily. One common approach, which is simple yet powerful enough to represent complex distributions, is to model the overall distribution as a mixture of Gaussian as illustrated in Figure 9.1(b).



(a) (b)

Figure 9.1: The probability density function of Y and the GMM that describes it.

We saw one situation where a GMM was appropriate in Example 8.6 in the previous lecture.

Example 8.6 revisited. Recall that this example is based on data collected by Galton including the heights, x_i, and genders, z_i, of several individuals. We discussed how modeling the overall distribution of heights as a simple Gaussian is not a very good fit to the observed height data.

Instead, it is better to split on the variable of gender, and then the heights for each gender appear to be roughly Gaussian. In mathematical notation, each person i has gender $z_i \in \{0, 1\}$ and a height $x_i \in \mathbb{E}$, and we choose to model $x_i|z_i \sim N(\mu_{z_i}, \sigma^2)$. We may choose to model gender as being drawn from some Bernoulli distribution with probability π, so $\mathbb{P}(z_i) = \pi^{z_i}(1-\pi)^{1-z_i}$. Here, μ_0, μ_1, and π are hyperparameters.

One useful way to write down statistical models is using a *graphical model*, which represents variables (and hyperparameters) as nodes, and direct relationships between variables (and hyperparameters) as arrows. It is also common to differentiate the nodes in a graphical model which correspond to observed variables by shading. The graphical model representing the statistical model described above for Example 8.6 is given in Figure 9.2.

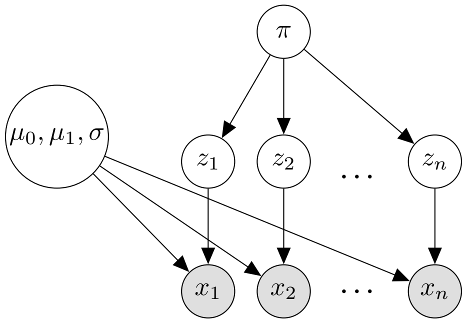


Figure 9.2: The graphical model describing the model for the Galton data in Example 8.6.

9.2 Latent Variable Models

In the Galton example, modeling the data with a GMM was straightforward since both height and gender were observed for every individual in the dataset. There are situations where important structure in the data is hidden, but we would still like to take advantage of that structure when we perform statistical inferences. For example, if we did not know the genders of the individuals in Example 8.6, we would still want to discover and leverage that structure in the data. We would then call the heights, x_i, **observed variables** and the variables representing gender, z_i, **latent** or **hidden variables**. This would make the model (e.g. Figure 9.2) one example of a **latent variable model**.

The general form of a latent variable model is given in Figure 9.3.

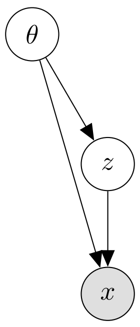


Figure 9.3: General form of a latent variable model, where θ are hyperparameters, z are latent variables, and x are observed variables.

An important special case of the latent variable model is the **Bayesian hierarchical model**, given in Figure 9.4. In a Bayesian hierarchical model, observations are independent given the latent variables, and each observed variable depends only on its corresponding latent variable and the hyperparameters.

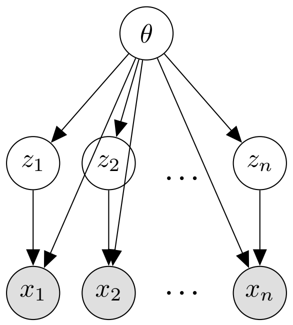
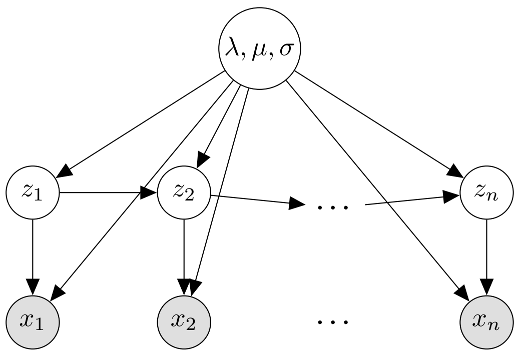


Figure 9.4: General graphical model describing a Bayesian hierarchical model with hyperparameters θ, latent variables z_i, and observed variables x_i.

Hidden Markov models (HMMs) are another example of a specific class of latent variable model. One key difference between an HMM and a hierarchical Bayesian model is that in an HMM each latent variable z_i depends on the previous latent variable z_{i-1}. These additional dependencies mean that, for example, knowing information about x_1 gives some information about x_2. The following example of an HMM further clarifies these properties.

Example 9.1. Suppose $z_1, z_2, \dots z_T$ give the true population of a type of fish in a lake over time. At each time t, we randomly sample various locations in the lake and count the number of fish samples, obtaining measurements $x_1, x_2, \dots, x_T$. A reasonable model for this data: $x_t \sim$ Poisson(λz_t), $z_{t+1} \sim N(z_t, \sigma^2)$, and $z_0 \sim N(\mu, \sigma^2)$. Here, $\theta = (\lambda, \mu, \sigma)$ are hyperparamters, the z_t are latent variables, and the x_t are observed. The corresponding graphical model is:

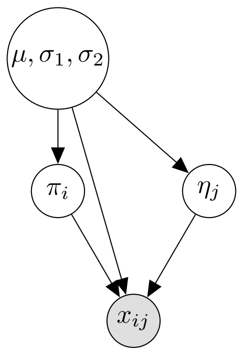


We conclude this section by giving one example where Bayesian hierarchical modeling had a large, real-world impact.

Example 9.2. Suppose we are in charge of 2016 election forecasting, and we would like to know the fraction of people in each state who will vote for Clinton. Each of the 50 states has some number of polls, and we assume each poll has a large enough sample size that we can treat its error as Normally distributed (by the Central Limit Theorem). Thus, we could use a model where we have an independent Gaussian margin of error in each state, sample the fraction of Clinton supporters in each state accordingly, and look at how often Clinton wins to forecast the outcome of the overall election. A model very similar to this predicted 90% for Clinton in 2016, but Trump ended up winning the election.

What is wrong with this analysis that could have contributed to overestimating Clinton's chances? One important issue is that we have assumed independent margins of error, whereas in reality pollster bias is likely to be consistent. Specifically, if a given pollster is biased in one direction in one state, that pollster is likely to be biased the same way in another state. This means that errors are actually highly correlated. If we model errors as independent when in fact they are correlated, we will likely be overconfident, since under a model with independence assumptions it would be very unlikely to be wrong in the same direction many times.

What could we do better? Say for each state there is a fraction of Clinton supporters, $\pi_1, \pi_2, \ldots, \pi_5 0$, and that there are k pollsters, where the j^{th} pollster has some bias η_j. Denote by x_{ij} the result of pollster j in state i. One potential model which accounts for pollster bias is to suppose $\pi_i \sim N(\mu, \sigma_1^2)$, $\eta_j \sim N(0, \sigma_2^2)$, and $x_{ij} \sim N(\pi_i + \eta_j, \sigma_{ij}^2)$. In practice, one would likely compute σ_{ij} based on each poll's sample size (which may require some additional hyperparameters), but for the purposes of this example we ignore these details. Here, $\theta = (\mu, \sigma_1, \sigma_2)$ are the hyperparameters, the π_i and η_j are latent variables, and the x_{ij} are observed variables.



Nate Silver from FiveThirtyEight used a model somewhat similar to this (but which incorporated some additional factors beyond just pollster bias) to forecast the 2016 election.

9.3 Inference on Latent Variable Models

At this point, we have seen several example of useful latent variable models. How do we actually perform inference in these models?

One potential method is to place a prior on the parameters and sample $\mathbb{P}(\theta, z|x)$. We will talk more about this approach in the next lecture. In this lecture, we will focus on a slightly different method wherein we aim to maximize $\log \mathbb{P}(x|\theta) = \log\left(\sum_z \mathbb{P}(x, z|\theta)\right)$. This approach can be thought of as being "half Bayesian" since we effectively do maximum likelihood estimation over θ, but still treat the latent z's probabilistically.

As stated, this "half-Bayesian" method has a significant drawback in that we need to sum over all the latent variables z. In Example 8.6, even if there are only 100 individuals in the Galton dataset, there are $2^100 \approx 10^30$ possible ways to assign binary genders to each person. This sum over z is thus too large to straightforwardly compute, even in this moderate case. We need a better approach for solving this problem in an efficient manner. We will see that the *Expectation-Maximization* (EM) algorithm addresses precisely this problem.

As further motivation for the EM algorithm, we will look more closely at posteriors for Bayesian hierarchical models. Recall we have latent variables $z_1, z_2, \ldots, z_n$ and observed variables $x_1, x_2, \ldots, x_n$.

First suppose we know all the hyperparameters θ. We claim that it is not too difficult to compute the conditional distribution $\mathbb{P}(z_1, \ldots, z_n|x_1, \ldots, x_n, \theta)$. In particular, since the z_i are conditionally independent and each x_i only depends on its corresponding latent variable z_i, the posterior factors as the product of individual terms which each only depend on one latent variable:

$$\begin{aligned}\mathbb{P}(z_1, z_2, \ldots, z_n|x_1, x_2, \ldots, x_n, \theta) &\propto \mathbb{P}(z_1, z_2, \ldots, z_n|\theta)\mathbb{P}(x_1, x_2, \ldots, x_n|z_1, z_2, \ldots, z_n, \theta)\\ &= \prod_{i=1}^n \mathbb{P}(z_i|\theta) \prod_{i=1}^n \mathbb{P}(x_i|z_i, \theta).\end{aligned}$$

Since the posterior decomposes, we can separately compute each independent posterior $p(z_i|x_i, \theta) \propto p(z_i|\theta)p(x_i|z_i, \theta)$, and then sample them independently. This is the reason why posteriors that decompose like this are efficiently computable, and this decomposition will happen whenever we have a hierarchical model. (As an aside, although it is slightly more complicated due to the time dependence structure, there are also efficient algorithms for working with HMMs.)

Now, suppose instead that x and z were are both known. In this case, we would just like to do maximum likelihood estimation on θ:

$$\underset{\theta}{\operatorname{argmax}} \log \mathbb{P}(x, z|\theta).$$

It turns out that we can also do this efficiently for hierarchical models.

To summarize, we have seen that if θ is known, it is often easy to to compute the posterior $\mathbb{P}(z|\theta, x)$, and if instead z is known, it is often easy to find $\operatorname{argmax}_\theta \log \mathbb{P}(x, z|\theta)$. The EM algorithm, defined in the next section, combines these two pieces into a larger algorithm that solves our original maximum likelihood problem:

$$\underset{\theta}{\operatorname{argmax}} \log \mathbb{P}(x|\theta).$$

9.4 Expectation-Maximization

The EM algorithm alternates between updating two variables, θ and q, where q is chosen to match $\mathbb{P}(z|x, \theta)$, and $\theta = \operatorname{argmax}_\theta \mathbb{E}_{z\sim q(z)}[\log \mathbb{P}(x, z|\theta)]$. Formally, initialize $\theta^{(1)}$ arbitrarily. Then, for iterations $t = 1, 2, \ldots, T$:

1. $q^{(t)} \leftarrow \mathbb{P}(z|x, \theta^{(t)})$
2. $\theta^{(t+1)} \leftarrow \operatorname{argmax}_\theta \mathbb{E}_{z\sim q^{(t)}(z)}[\log \mathbb{P}(x, z|\theta)]$

Step 1 is known as the E-step, since it imputes a new distribution over z based on your current estimates of the parameters and the observed data. Step 2 is called the M-step, since it updates the parameters by maximizing the likelihood.

Remark 9.3. The EM algorithm can be shown to maximize a lower bound on the log-likelihood of the data $\mathbb{P}(x|\theta)$ at each iteration, meaning that as the algorithm runs we have more and more confidence that the log-likelihood of the data is improving. This formalizes the relationship between EM and our original problem of solving $\operatorname{argmax}_\theta \log \mathbb{P}(x|\theta)$.

9.5 EM for Gaussian

We will now show how the abstract EM algorithm works out in the Gaussian case. Specifically, we will present a method of learning the parameters of a GMM, $\theta_i = (\mu_i, \sigma_i^2)$, and π_i, from observed data $x_1, \ldots, x_n$.

Suppose we know that there are d underlying Gaussians. If we knew the value of z_j for each x_j we could write the likelihood of the data as:

$$\mathbb{P}(x_j, z_j|\theta_1, ..., \theta_d, \pi_1, ..., \pi_d) = \mathbb{P}(z_j|\pi_1, ..., \pi_d)\mathbb{P}(x_j|z_j, \theta_1, ..., \theta_d) = \prod_{i=1}^{d}(\pi_i \mathcal{N}(x_j; \theta_i))^{\mathbb{I}(z_j=i)}.$$

The log likelihood of all of the data would therefore be given by:

$$\ell(x, z_j; \theta_1, ..., \theta_d, \pi_1, ..., \pi_d) = \sum_{j=1}^{n}\sum_{i=1}^{d} \mathbb{I}(z_j = i)(\log(\pi_i) + \log(\mathcal{N}(x_j; \mu_i, \sigma_i^2)).$$

which we could maximize over all $\theta_1, ..., \theta_d$ and $\pi_1, ..., \pi_d$.

On the other hand, if we knew all the values of $\theta_1, ..., \theta_d$ and $\pi_1, ..., \pi_d$ we could find the posterior distribution over z_j for each data point x_j. This posterior is given by:

$$\mathbb{P}(z_j = i|x_j) = \frac{\pi_i \mathcal{N}(x_j; \mu_i, \sigma_i^2)}{\sum_{k=1}^{d} \pi_k \mathcal{N}(x_j; \mu_k, \sigma_k^2)}.$$

In this setting, the EM algorithm will proceed as follows:

1. Randomly initialize θ_i and π_i.
2. Given fixed θ_i and π_i, for each data point x_j approximate the probability that z_j comes from Gaussian i, denoted $q_j(i) = \mathbb{P}(z_j = i|x_j)$.
3. Given fixed distributions q_j find the values of θ_i and π_i that maximize the expected likelihood of the data (over the distributions $q_j(i)$):

   $$(\pi^*, \theta^*) = \underset{\pi_1,...,\pi_d,\theta_1,...,\theta_d}{\text{argmax}} \mathbb{E}_q[\ell(x; \theta_1, ..., \theta_n)]$$

   Since $\mathbb{E}[\mathbb{I}(z_j = i)] = \mathbb{P}(z_j = i|x_i) = q_j(i)$, this simplifies to:

   $$(\pi^*, \theta^*) = \underset{\pi_1,...,\pi_d,\theta_1,...,\theta_d}{\text{argmax}} \sum_{j=1}^{n}\sum_{i=1}^{d} q_j(i)(\log(\pi_i) + \log(\mathcal{N}(x_j; \mu_i, \sigma_i^2))$$

   .
4. Iterate between the two sub-problems until convergence.

We now outline the EM algorithm with unknown μ_i, σ_i^2, π_i for a mixture of d Gaussians given $x_1, ...x_n$.

Algorithm 1 Expectation-Maximization Algorithm for Gaussian Mixture Models.

```
Input: Data: $x_1, ..., x_n$, Number of Gaussians in the mixture $d$, number of iterations $T$
Output: $(\pi_i, \mu_i, \sigma_i^2)$ for $i = 1, ..., d$.
Randomly Initialize $(\pi_i^{(0)}, \mu_i^{(0)}, \sigma_i^{(0)})$ for $t = 1$ to $T$ do
    E-Step: for $j = 1$ to $n$ do
        for $i = 1$ to $d$ do
            $q_j(i) \leftarrow \frac{\pi_i^{(t-1)} N(x_j; \mu_i^{(t-1)}, (\sigma_i^{(t-1)})^2)}{\sum_{k=1}^{d} \pi_k^{(t-1)} N(x_j; \mu_k^{(t-1)}, (\sigma_k^{(t-1)})^2)}$
        end
    end
    M-Step: for $i = 1$ to $d$ do
        $N_i^{(t)} \leftarrow \sum_{j=1}^{n} q_j(i)$.
        $\mu_i^{(t)} \leftarrow \frac{1}{N_i^{(t)}} \sum_{j=1}^{n} q_j(i) x_j$.
        $\sigma_i^{(t)} \leftarrow \frac{1}{N_i^{(t)}} \sum_{j=1}^{n} q_j(i)(x_j - \mu_i^{(t)})^2$.
        $\pi_i^{(t)} \leftarrow \frac{N_i^{(t)}}{n}$.
    end
end
```

Note that the update for μ_i and σ_i^2 are both the maximizers of the expected likelihood using typical maximum likelihood approaches from previous lectures and discussions. The update for π_i, however, requires maximizing the expected likelihood while constraining $\sum_{i=1}^{d} \pi_i = 1$. This derivation requires using solving a constrained optimization problem which is outside the scope of this class.

Remark 9.4. Note that the EM algorithm can be very sensitive to the initialization, and is not guaranteed to converge to the same solution from any initialization.