Lecture 8:

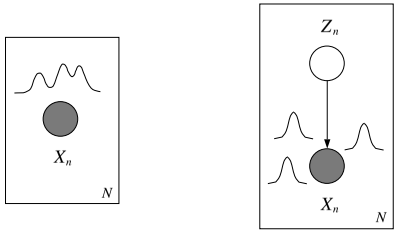
Latent Variable Models

Sam Roweis

January 30, 2002

Latent Variables

- What to do when a variable $\mathbf{Z}$ is *always* unobserved?
  Depends on where it is. If it is a leaf in a directed model, we can just forget about it. But if it is an interior node we need to model it.
- We may also want to *intentionally* introduce latent variables to model complex dependencies between variables without using links directly between them.

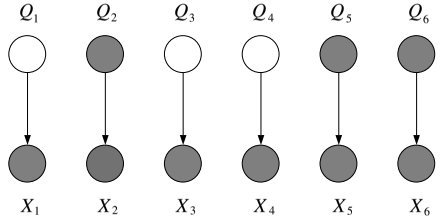


(a) (b)

This can actually simplify the model (e.g. Naive Bayes, mixtures).

Unobserved Variables

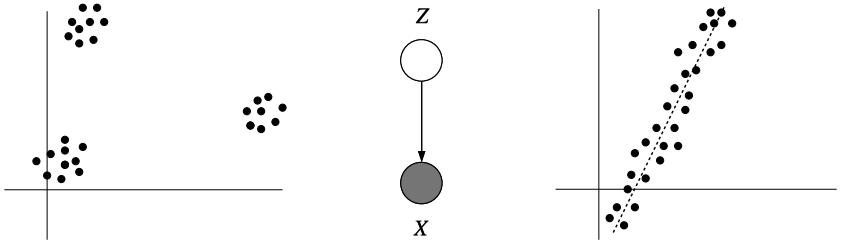
- Certain variables Q in our models may be *unobserved*, either some of the time (missing) or always (latent), either at training time or at test time.



  Graphically, we will use shading to indicate observation.
- For partially missing values, we can model the joint distribution and *integrate out* the missing values at training or test time.
  [Recall that $p(x) = \sum_q p(x, q)$.]

Clustering vs. Classification
Latent Factor Models vs. Regression

- You can think of clustering as the problem of classification with missing class labels.



- You can think of factor models (such as factor analysis, PCA, ICA, etc.) as linear or nonlinear regression with missing inputs.

Why is Learning Harder?

- In fully observed settings, the probability model is a product thus the log likelihood is a sum where terms decouple.
  (At least for directed models.)

$$\begin{aligned}\ell(\theta;\mathcal{D}) &= \log p(\mathbf{x},\mathbf{z}|\theta) \\ &= \log p(\mathbf{z}|\theta_z) + \log p(\mathbf{x}|\mathbf{z},\theta_x)\end{aligned}$$

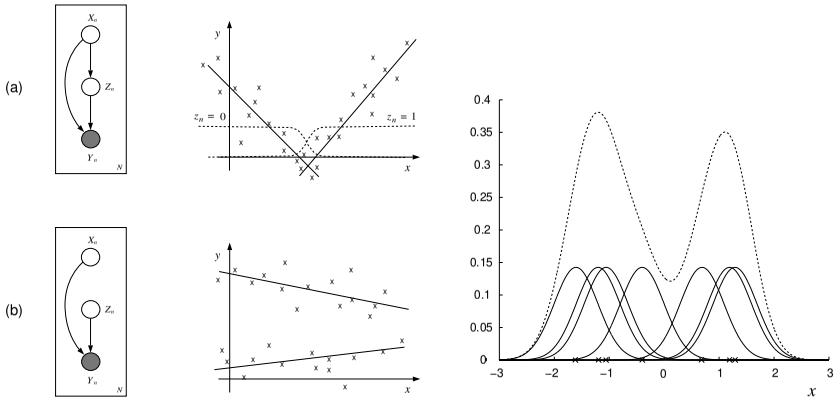
- With latent variables, the probability already contains a sum, so the log likelihood has all parameters coupled together:

$$\begin{aligned}\ell(\theta;\mathcal{D}) &= \log \sum_{\mathbf{z}} p(\mathbf{x},\mathbf{z}|\theta) \\ &= \log \sum_{\mathbf{z}} p(\mathbf{z}|\theta_z) p(\mathbf{x}|\mathbf{z},\theta_x)\end{aligned}$$

  (Just as with the partition function in undirected models.)

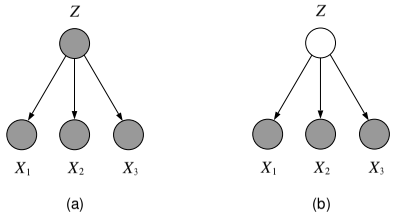
Mixture Models

- Most basic latent variable model with a single discrete node z.
- Allows different "experts" or "submodels" to contribute to the density model in different parts of the space. Divide and conquer. (e.g. multimodal densities, or piecewise-linear regressions).



Learning with Latent Variables

- Likelihood $\ell(\theta;\mathcal{D}) = \log \sum_{\mathbf{z}} p(\mathbf{z}|\theta_z) p(\mathbf{x}|\mathbf{z},\theta_x)$ couples parameters:



- We can treat this as a black box probability function and just try to optimize the likelihood as a function of θ. However, sometimes taking advantage of the latent variable structure can make parameter estimation easier.
- Good news: soon we will see the *EM algorithm* which allows us to treat learning with latent variables using fully observed tools.
- Basic trick: guess the values you don't know.
  Basic math: use convexity to lower bound the likelihood.

Mixture Densities

- Exactly like a classification model but the class is unobserved and so we sum it out. What we get is a perfectly valid density:

$$\begin{aligned}p(\mathbf{x}|\theta) &= \sum_{k=1}^{K} p(z=k|\theta_z) p(\mathbf{x}|z=k,\theta_k) \\ &= \sum_{k} \alpha_k p_k(\mathbf{x}|\theta_k)\end{aligned}$$

  where the "mixing proportions" add to one: $\sum_k \alpha_k = 1$.
- We can use Bayes' rule to compute the posterior probability of the mixture component given some data:

$$p(z=k|\mathbf{x},\theta) = \frac{\alpha_k p_k(\mathbf{x}|\theta_k)}{\sum_j \alpha_j p_j(\mathbf{x}|\theta_j)}$$

  these quantities are called *responsibilities*.

Mixtures of Experts

- Also called conditional mixtures. Exactly like a class-conditional model but the class is unobserved and so we sum it out again:

$$p(\mathbf{y}|\mathbf{x},\theta) = \sum_{k=1}^{K} p(z=k|\mathbf{x},\theta_z)p(\mathbf{y}|z=k,\mathbf{x},\theta_k)$$
$$= \sum_k \alpha_k(\mathbf{x}|\theta_z)p_k(\mathbf{y}|\mathbf{x},\theta_k)$$

  where $\sum_k \alpha_k(\mathbf{x}) = 1 \quad \forall \mathbf{x}$.
- Harder: must learn $\alpha(\mathbf{x})$ (unless chose z independent of $\mathbf{x}$).
- We can still use Bayes' rule to compute the posterior probability of the mixture component given some data:

$$p(z=k|\mathbf{x},\mathbf{y},\theta) = \frac{\alpha_k(\mathbf{x})p_k(\mathbf{y}|\mathbf{x},\theta_k)}{\sum_j \alpha_j(\mathbf{x})p_j(\mathbf{y}|\mathbf{x},\theta_j)}$$

  this function is often called the *gating function*.

Example: Mixture of Linear Regression Experts

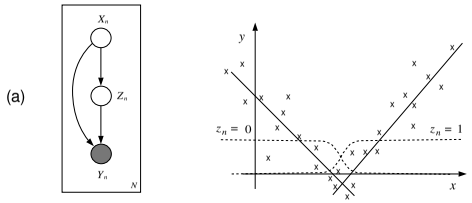
- Each expert generates data according to a linear function of the input plus additive Gaussian noise:

$$p(y|\mathbf{x},\theta) = \sum_k \alpha_k(\mathbf{x})\mathcal{N}(y|\beta_k^\top \mathbf{x}, \sigma_k^2)$$

- The "gate" function can be a softmax classification machine:

$$\alpha_k(\mathbf{x}) = p(z=k|\mathbf{x}) = \frac{e^{\eta_k^\top \mathbf{x}}}{\sum_j e^{\eta_j^\top \mathbf{x}}}$$

- We are not modeling the density of the inputs $\mathbf{x}$.

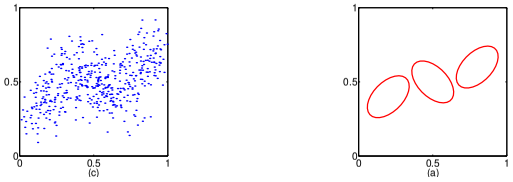


Example: Gaussian Mixture Models

- Consider a mixture of K Gaussian components:

$$p(\mathbf{x}|\theta) = \sum_k \alpha_k \mathcal{N}(\mathbf{x}|\mu_k, \Sigma_k)$$
$$p(z=k|\mathbf{x},\theta) = \frac{\alpha_k \mathcal{N}(\mathbf{x}|\mu_k,\Sigma_k)}{\sum_j \alpha_j \mathcal{N}(\mathbf{x}|\mu_k,\Sigma_k)}$$
$$\ell(\theta;\mathcal{D}) = \sum_n \log \sum_k \alpha_k \mathcal{N}(\mathbf{x}^n|\mu_k,\Sigma_k)$$

- As a clustering machine, the probabilistic semantics provides us with the familiarity signal, the assignment rule, and the cost function.



Parameter Constraints

- If we want to use general optimizations (e.g. conjugate gradient) to learn latent variable models, we often have to make sure parameters respect certain constraints. (e.g. $\sum_k \alpha_k = 1$, Σ_k pos.definite).
- A good trick is to reparameterize these quantities in terms of unconstrained values. For mixing proportions, use the softmax:

$$\alpha_k = \frac{\exp(q_k)}{\sum_j \exp(q_j)}$$

- For covariance matrices, use the Cholesky decomposition:

$$\Sigma^{-1} = A^\top A$$
$$|\Sigma|^{-1/2} = \prod_i A_{ii}$$

  where A is upper diagonal with positive diagonal:

$$A_{ii} = \exp(r_i) > 0 \qquad A_{ij} = a_{ij} \quad (j > i) \qquad A_{ij} = 0 \quad (j < i)$$

Logsum

- Often you can easily compute $b_k = \log p(\mathbf{x}|z = k, \theta_k)$, but it will be very negative, say -10^6 or smaller.
- Now, to compute $\ell = \log p(\mathbf{x}|\theta)$ you need to compute $\log \sum_k e^{b_k}$.
- Careful! Do not compute this by doing `log(sum(exp(b)))`. You will get underflow and an incorrect answer.
- Instead do this:
  - Add a constant exponent B to all the values b_k such that the largest value comes close to the maxiumum exponent allowed by machine precision: `B = MAXEXPONENT-log(K)-max(b)`.
  - Compute `log(sum(exp(b+B)))-B`.