STA561: Probabilistic machine learning

Exact Inference (9/30/13)

Lecturer: Barbara Engelhardt *Scribes: Jiawei Liang, He Jiang, Brittany Cohen*

1 Validation for Clustering

If we have two centroids, η_1 and η_2, all of the data are clustered around these two centroids. Say we have a new data point, x^*, how do we validate whether or not our model can accurately classify this new unobserved data? One way is to find the RMSE$(x^*, \hat{x}^*)$, where x^* is the value we are looking at and $\hat{x}^*$ is its predicted value.

How do you predict the value of this without seeing the underlying class? We could see what centroid is this x^* associated with. If we see that x^* is associated with centroid η_2, we would predict that $\hat{x}^*$ is equal to η_2.

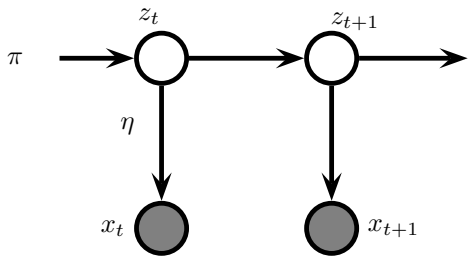
If you use the Gaussian version of this, where you have a mean vector and covariance matrix associated with each of these data points, you can actually compute the *Mahalanobis distance*. You can do this with a soft clustering (soft assignment), assign the data point to η_2 with some probability and compute the Mahalanobis distance and then do the same for η_1. The Mahalanobis distance can be calculated as follows:

$$D_M(x) = \sqrt{(x-\mu)^T \Sigma^{-1} (x-\mu)}$$

where $x = (x_1, x_2, x_3, ..., x_N)^T, \mu = (\mu_1, \mu_2, \mu_3, ..., \mu_N)^T$, and Σ is the covariance matrix.

2 A brief review of Forward-Backward and EM for HMMs

2.1 Forward-Backward



In the hidden Markov model, π represents the initial state, η is the emission probability, and x_t, x_{t+1} have

been observed. We want to find the expected value of a transition from state $z_t^j \rightarrow$ state z_{t+1}^k. The main trick is splitting the data into data before our time point t, and after.

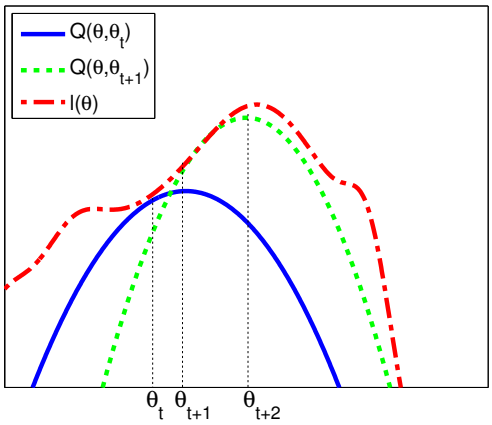
$$
\begin{aligned}
E[z_t^j, z_{t+1}^k] &= p(z_t^j, z_{t+1}^k \mid X_{1:t}, X_{t+1:T}) \\
\text{by Bayes' rule:} \\
&\propto p(X_{t+1:T} \mid Z_t^j, Z_{t+1}^k, X_{1:t}) p(Z_t^j, Z_{t+1}^k \mid X_{1:t}) \\
\text{by conditional independence and the chain rule:} \\
&= p(X_{t+1:T} \mid Z_{t+1}^k) p(Z_t^j, Z_{t+1}^k \mid X_{1:t}) \\
&= p(X_{t+1}, X_{t+2:T} \mid Z_{t+1}^k) p(Z_{t+1}^k \mid Z_t^j, X_{1:t}) p(Z_t^j \mid X_{1:t}) \\
&= p(X_{t+1} \mid Z_{t+1}^k) p(X_{t+2:T} \mid Z_{t+1}^k) p(Z_{t+1}^k \mid Z_t^j) p(Z_t^j \mid X_{1:t}) \\
&= \eta^k \beta_{t+1}(k) A^{jk} \alpha_t(j)
\end{aligned}
$$

2.2 EM

Suppose we have the following data: $\mathcal{D} = \{(x_1, ..., x_T)_1, ..., (x_1, ..., x_T)_n\}$. We have n of these fully observed chains with unknown states on them. (In our running weather example, $(x_1, ..., x_T)$ would be our set of features such as temperature, barometric pressure, wind speed, rain accumulation, etc. at those time points, where time points could be days of the month). We have no idea whether the actual weather was sunny, cloudy, or rainy (we are not given this information). We want to design an HMM that can predict what weather we will be seeing in the current month.

The EM algorithm for HMMs, also known as the Baum-Welch algorithm (Baum et al. 1970), can be briefly written as:

- Initialize parameters:
  - Transition probabilities A
  - Initial state π
  - Emission probabilities η. In the case of Gaussian, $\eta = \{\mu_k, \Sigma_k\}$, where μ_k, Σ_k are cluster-specific parameters of Gaussian
- E Step: use the forward-backward algorithm to compute the expected sufficient statistics
  - $\mathrm{E}[Z_1^k]$
  - $\mathrm{E}[Z_t^j, Z_{t+1}^k]$
  - To compute these, we need to run the forward-backward algorithm. We already initialized values for A, π, η (call these the parameters at step $s = 0$). So we can just plug those values and compute the expectations, conditional on these parameter values.
- M Step:
  - Compute MLE or MAP estimates of A, π, η, given the expected sufficient statistics from the E step. These updates can be derived, e.g., via the expected complete log likelihood.
- Iterate EM steps until convergence



Since our Z values are unobserved, we are selecting a likelihood function from all sets of possible likelihoods, as shown in the figure below (from Figure 11.16 of the Murphy textbook). To find the MLE $\hat{\theta}$, we need a concave likelihood function. In the E step, we choose one of these trajectories of the complete log likelihood, selected by choosing the value of the expected sufficient statistics. Then in the M step, given we have this concave description of complete log likelihood, it is easy to find the global maximum. EM is called a *coordinate ascent algorithm* because this method will continue to increase the expected complete log likelihood values in a monotone way, iterating between reevaluating the latent variables (the E-step) and the model parameters (the M-step), until a local maxima is achieved.

3 Exact Inference

3.1 Introduction

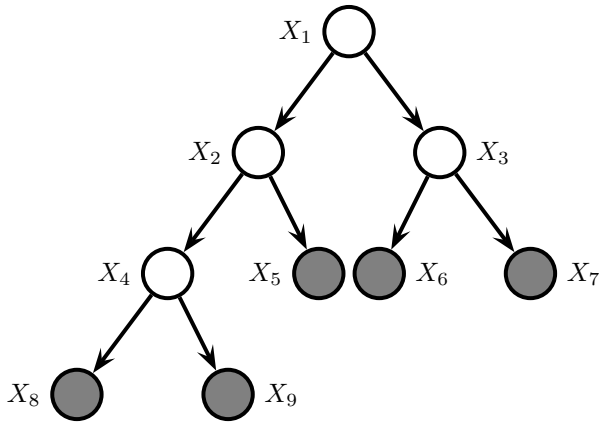
In the previous Section, we discussed the forward-backward algorithm in HMMs, which is an example of *Belief Propagation* (BP), or the *sum product* on chains. In this section we expand this idea to general trees. Then we will begin to discuss variable elimination on arbitrary graphs. These methods all encapsulate methods for estimating parameters exactly according to the MLE or the MAP estimates in the context of missing observations. We will generally compute the marginal value of specific latent variables by marginalizing out the remaining latent variables directly, incorporating the observations as we go.

3.2 Belief Propagation

Just like the forward-backward algorithm, there are two basic steps to performing BP in trees:

- Leaves $\rightarrow$ root, to collect evidence;
- Root $\rightarrow$ leaves, to distribute evidence.

Consider the following tree:



This can be thought of as an evolutionary tree (a *phylogeny*) where X represents the number of toes each species at the leaves of the tree has. Then we might be interested in computing $P(X_1|X_{5:9})$, which is probability of the number of toes in the most recent common ancestor given the observed leaf values. In the same example we might also be interested in computing $P(X_2|X_{5:9})$, or the probability of the number of toes of the most recent ancestor of X_5 and X_8 and X_9 given all of the observations.

First, we can write out the factorization of the joint probability of this model:

$$P(X_{1:9}) = P(X_1)P(X_2|X_1)P(X_3|X_1)P(X_4|X_2)P(X_5|X_2)P(X_6|X_3)P(X_7|X_3)P(X_8|X_4)P(X_9|X_4).$$

By the definition of conditional probabilities, we have

$$\begin{aligned} P(X_1|X_{5:9}) &= \frac{\sum_{X_2X_3X_4} P(X_{1:9})}{P(X_{5:9})} \\ &\propto \sum_{X_2X_3X_4} P(X_{1:9}) \\ &= \sum_{X_2X_3X_4} P(X_1)P(X_2|X_1)P(X_3|X_1)P(X_4|X_2)P(X_5|X_2)P(X_6|X_3)P(X_7|X_3)P(X_8|X_4)P(X_9|X_4). \end{aligned}$$

If this is a multinomial model, and each node has three different possibilities given its ancestor, then the size of CPT (conditional probability table) would be 3^9, which has too many states to sum over (even for a tree of this size). So instead we introduce an algorithm to compute the information locally from leaves to root, and propagate this information back to the leaves. This *belief propagation* algorithm, also known as the *peeling* algorithm on trees, produces exact marginal probabilities, and is a single upward and downward pass of the messages.

3.2.1 Upward Message

The upward message passes the information at the leaves of the tree up to the root of the tree. Let's rewrite the expression above with indicator values for evidence $P(X_{5:9})$; we then push the summation into the

factorized probability distribution:

$$\begin{aligned}P(X_1|X_{5:9}) = \sum_{X_{2:9}} & P(X_1)P(X_2|X_1)P(X_3|X_1)P(X_4|X_2)P(X_5|X_2)\mathbb{1}(X_5 = x_5) \\ & P(X_6|X_3)\mathbb{1}(X_6 = x_6)P(X_7|X_3)\mathbb{1}(X_7 = x_7)P(X_8|X_4)\mathbb{1}(X_8 = x_8)P(X_9|X_4)\mathbb{1}(X_9 = x_9) \\ = P(X_1) & \sum_{X_2} P(X_2|X_1) \sum_{X_3} P(X_3|X_1) \sum_{X_4} P(X_4|X_2) \sum_{X_5} P(X_5|X_2)\mathbb{1}(X_5 = x_5) \\ & \sum_{X_6} P(X_6|X_3)\mathbb{1}(X_6 = x_6) \sum_{X_7} P(X_7|X_3)\mathbb{1}(X_7 = x_7) \sum_{X_8} P(X_8|X_4)\mathbb{1}(X_8 = x_8) \sum_{X_9} P(X_9|X_4)\mathbb{1}(X_9 = x_9)\end{aligned}$$

Message Passing: Denote the message from node b to node a to be

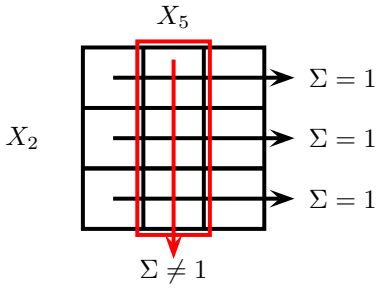
$$\begin{aligned}m_{ba}(X_b) &= P(X_b|X_{des(b)}) \\ &= \prod_{i\in \text{Ch}(b)} \sum_{x_i \in X_i} \psi(X_b, X_i) m_{ib}(X_i)\end{aligned}$$

where $\psi(X_b, X_i)$ is the unnormalized probability of X_b, X_i, called the *potential* in undirected graphs. m_{ci} is multiplied by $\mathbb{1}(X_c = x_c)$ when X_c is observed (as having value x_c).

Consider

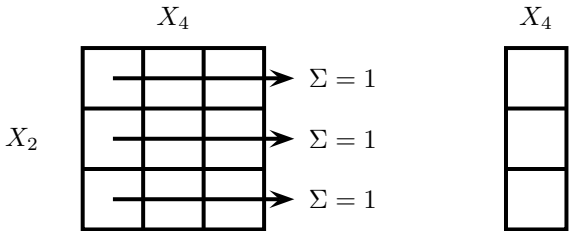
$$\begin{aligned}m_{21}(X_2) &\propto P(X_2|X_4, X_5) \\ &= \big[\sum_{X_4} P(X_4|X_2)m_{42}(X_4)\big]\big[\sum_{X_5} P(X_5|X_2)\mathbb{1}(X_5 = x_5)\big].\end{aligned}$$

We draw the CPT in the second bracket $\sum_{X_5} P(X_5|X_2)\mathbb{1}(X_5 = x_5)$ as follows:



As shown in the graph, each row of X_2 is normalized and sum up to one. Each column is not necessarily normalized and we only select the column where $X_5 = x_5$.

We can also draw the CPT in the first bracket $\sum_{X_4} P(X_4|X_2)m_{42}(X_4)$:



The assumptions are the same as the previous graph. We select each column in the left table and multiply it by the upward message (the right vector) and sum them up.

The message at root:

$$m_{root}(X_1) = \big[\sum_{X_2} P(X_2|X_1)m_{21}(X_2)\big]\big[\sum_{X_3} P(X_3|X_1)m_{31}(X_3)\big]$$

Then:

$$P(X_1|X_2, ..., X_9) \propto P(X_1)m_{root}(X_1)$$

3.2.2 Downward Messages

After we get $P(X_1|X_{2:9})$ through upward messages, we pass these observations down in order to find $P(X_2|X_{5:9})$. Our downward pass now will incorporate observations not descendant from a hidden node into the conditional probability for that node. Note that $P(X_2|X_{5:9})$ is not the same thing as $P(X_2|X_5, X_8, X_9)$. Currently, we have $P(X_2|X_5, X_8, X_9)$, which is $m_{21}(X_2)$. We want to include in this conditional probability all the information that X_1 has received from every leaf, and combine the non-descendant leaves with the information that X_2 received from its descendants on the upward pass.

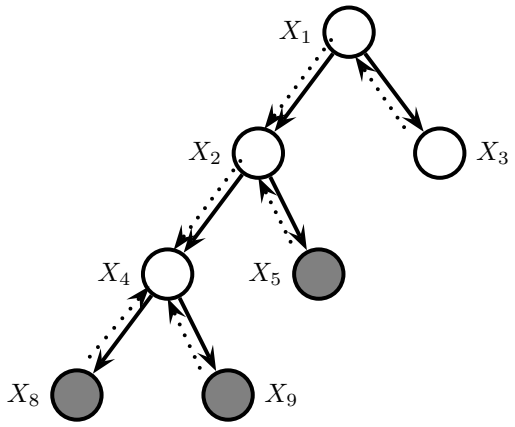
For downward messages from X_2 to X_4,

$$m_{24}(X_4) \propto P(X_4|X_{\text{non-desc}(4)}) = P(X_4|X_5, X_6, X_7)$$

Consider node t, with parent r. To compute the belief state for t, we need to combine the bottom-up belief from its children s and c together with a top-down message from r, which summarizes all the non-descendant information from the rest of the graph:

$$m_{ts}(X_s) = \sum_{X_t} P(X_s|X_t)[\prod_{c\in \text{Ch}(t), c\neq s} m_{ct}(X_t)]m_{rt}(X_t)$$

Combining upward and downward messages, we can compute $p(X_4|X_{5:9})$ as follows (see graph for representation):



where $P(X_4|X_{5:9}) \propto m_{42}(X_4)m_{24}(X_4)$, the product of the upward message of its descendent and downward message of its non-descendent.

3.3 Variable Elimination

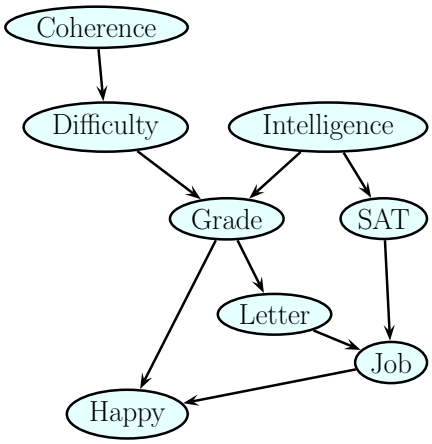


Figure 1: The student DGM. Based on Figure 9.8 of (Koller and Friedman 2009); variable names are indicated by the first .

Variable elimination is an extension of belief propagation to any arbitrary DAGs or even undirected graphs. Consider the directed graph in the figure below. In this method, as in Belief Propagation, we compute the exact marginal probability of a variable in this model. The example model above, from (Koller and Friedman 2009), relates categorical random variables pertaining to a single student. The corresponding joint has the following factorized form:

$$P(C, D, I, G, S, L, J, H) = P(C)P(D|C)P(I)P(G|I, D)P(S|I)P(L|G)P(J|L, S)P(H|G, J).$$

Now suppose we want to compute $p(J|G, S)$, the probability that a person will get a job given his grade and SAT score. Since we have eight categorical variables, we could simply enumerate over all possible assignments to all the variables (except for J, G and S), adding up the probability of each joint instantiation:

$$P(J|G, S) \propto \sum_{C,D,I,L,H} P(C, D, I, G, S, L, J, H)$$

We can be smarter by *pushing sums inside products*. In our example, we get

$$\begin{aligned} P(J|G, S) &\propto \sum_{C,D,I,L,H} P(C, D, I, G, S, L, J, H) \\ &= \sum_{C,D,I,L,H} P(C)P(D|C)P(I)P(G|I, D)P(S|I)P(L|G)P(J|L, S)P(H|G, J) \end{aligned}$$

push in the sums as far as possible:

$$= \sum_{L} p(J|L, S)p(L|G) \sum_{H} p(H|G, J) \sum_{I} p(S|I)p(I) \sum_{D} p(G|I, D) \sum_{C} p(C)p(D|C)$$

This is the key idea behind the *variable elimination* algorithm (Zhang and Poole 1996). We can eliminate variables in the order of C, D, I, H, L. The run time of this algorithm is proportional to the size of the largest

message, and the order we eliminate variables determines the size of the largest message in this method. Thus, the elimination order is critical for feasible approaches to computing these marginal probabilities exactly.

The downside of variable elimination (VE) is that we cannot easily reuse our 'messages', and thus we will design specialized elimination orderings for each marginal probability.