STA613/CBB540: Statistical methods in computational biology

Mixture Models (2/12/13)

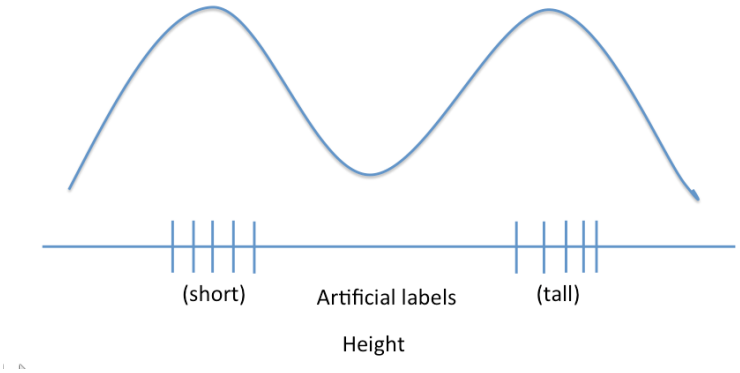
Lecturer: Barbara Engelhardt *Scribe: Brittany Stokes*

1. Mixture Models

Why would you want to include a random variable in your statistical model when those variables are unobserved? Often, a latent variable might help define 'clusters' of the samples: individuals in populations, coding versus non-coding sequence, etc. This clustering might be artificial, but it also still might be useful for interpretability because we can identify the characteristics that define this cluster versus the other clusters (e.g., clinical diagnosis). Often these latent variables simplify the model.

1.1. **Definition.**

Mixture model: a model in which the latent variable takes on one of a finite, unordered set of values. This latent variable represents unobserved subpopulations present in the sample.



Let's say we have a data set with n samples: $D = \{(x_1, ..., x_n)\}$, where x_i are the observed data, and z_i is the latent variable associated with sample i.

1.2. **Multinomial Distribution.**

Our variables z_i are distributed according to a multinomial distribution. This is a generalization of the binomial distribution to n trials with k outcomes (instead of 2 outcomes).

Say $x_i \sim Mult(\pi, n)$. Then each of the k elements of the vector x, x_i, represents the number of times outcome i occurred in n trials.

$$\sum_{i=1}^{k} x_i = n.$$

The probability distribution can be written as follows:

$$P(x_1 = x_1, \ldots, x_k = x_k | n, \pi_1, \ldots, \pi_k) = \frac{n!}{x_1! \ldots x_k!} \pi_1^{x_1} \ldots \pi_k^{x_k},$$

where $0 \leq \pi_i \leq 1$, and $\sum_{i=1}^{k} \pi_i = 1$. The fraction at the beginning accounts for all possible permutations of the n trials.

Example: roll a fair die five times. Here $n = 5$, $k = 6$, and $\pi_i = \frac{1}{6}$. Our vector x might look like this:

$$x = \begin{pmatrix} 1 \\ 2 \\ 0 \\ 2 \\ 0 \end{pmatrix}$$

Then $p = \frac{5!}{1!2!2!} \frac{1}{6}^1 \frac{1}{6}^2 \frac{1}{6}^2$.

For the multinomial distribution, $E[x_i] = n\pi_i$, $var[x_i] = n\pi_i(1 - \pi_i)$, and $cov[x_i, x_j] = -n\pi_i\pi_j$, highlighting the dependence of the different outcomes.

The conjugate prior of the multinomial distribution is the Dirichlet distribution. So, $\pi \sim Dir(\alpha)$ describes the distribution within a k dimensional space, called a *simplex* (Figure 1), constrained so that each element of the k length vector is non-negative and the vector π sums to one.

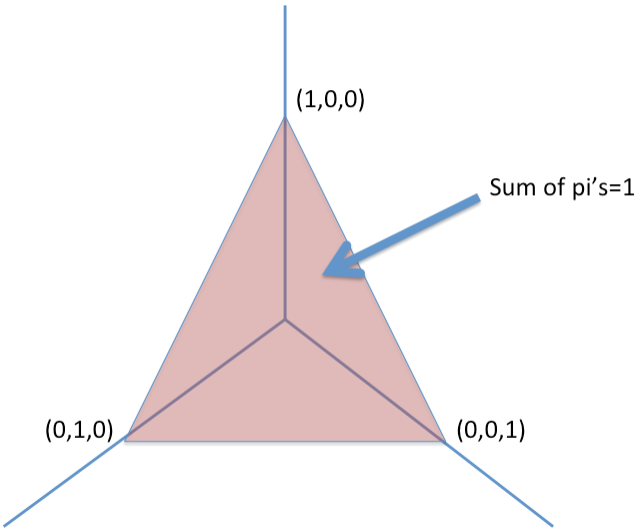


FIGURE 1. Representation of a simplex in three dimensions; every point on the simplex is a vector of length three where each element is non-negative and the three sum to 1.

1.3. **Mixture model.**

Returning to the mixture model and our samples, let's say that we have, for each sample x_j, a latent variable:

$$z_j = \begin{pmatrix} 0 \\ 0 \\ . \\ . \\ . \\ 1 \\ . \\ . \\ . \\ 0 \\ 0 \end{pmatrix}$$

where $z_j^i = 1$ at the i^{th} position of the vector, and zero everywhere else. This notation is called a *multinomial vector*, for k possible outcomes. Thus, for a single sample x:

$$\begin{aligned} P(x|\theta) &= \sum_{i=1}^{k} P(x|z^i = 1, \theta_i) P(z^i = 1||\pi_i) \\ &= \sum_{i=1}^{k} \pi_i P(x|z^i = 1, \theta_i) \end{aligned}$$

$P(x|z^i = 1, \theta_i)$ is the likelihood of data x coming from the i^{th} group; this term is called the *mixture component*. $P(z^i = 1|\pi_i) = \pi_i$ because of the multinomial distribution; this terms is called the *mixture proportion*, and represents the proportion of samples that are expected to be in cluster i.

We have not yet said how the mixture components are distributed; whatever their distribution, we use θ to represent their parameterization.

One way to think about this model is as if we are marginalizing over our latent variables, z:

$$\begin{aligned} P(x|\theta) &= \sum_{i=1}^{k} P(x, z^i = 1|\theta) \\ &= \sum_{i=1}^{k} P(x|z^i = 1, \theta_i) P(z^i = 1|\pi_i) \end{aligned}$$

1.4. **Gaussian Mixture Model.**

Let us assume now that the mixture components are each described by a multivariate Gaussian distribution; then θ_i, the parameters for each mixture component, is defined as (μ_i, Σ_i), where μ_i is the mean of the samples coming from cluster i, and Σ_i is the covariance of the samples coming from cluster i.

Each of the clusters in a mixture model are modeled separately, and have their own parameters. We assume, though, that each component is Gaussian distributed:

$P(x|\theta) = \sum_{i=1}^{k} \pi_i \mathcal{N}(x|\mu_{z^i}, \Sigma_{z^i})$.

We want to estimate μ, Σ, and π from our data. The number of components, k, is usually unknown (methods like `mclust` can be used to try to estimate it); in general, you need to try a number of

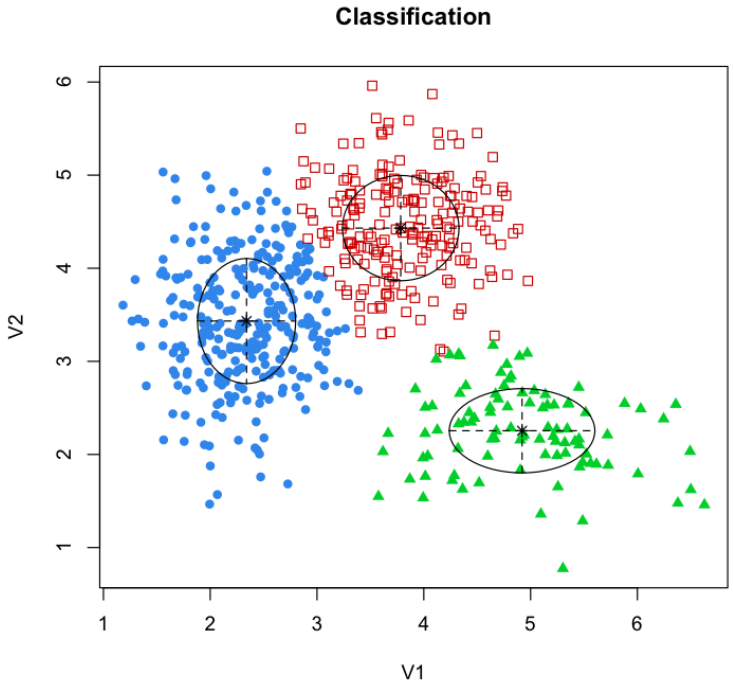


FIGURE 2. Three-cluster model for bivariate Gaussian data. The means and covariances are shown for each cluster.

different values to find the right k. The number of components changes the dimension of the latent structure.

If you get a new data point x^*, in which component should it be clustered? We can write out the posterior and apply Bayes' rule:

$$P(z^i = 1|x^*, \hat{\theta}) = \frac{P(x^*|z^i = 1, \hat{\theta})\pi_i}{\sum_k \pi_k P(x^*|z^k = 1, \hat{\theta})},$$

where $P(z^i = 1|x^*, \hat{\theta})$ is the posterior probability of latent variable z^i, x^* is the new observation, and $\hat{\theta}$ are the parameters estimated from the data x, $P(x^*|z^i = 1, \hat{\theta})$ is the likelihood of the new observation, and π_i is the prior probability of component i. Then, $\sum_k \pi_k P(x * |z^k = 1, \hat{\theta})$ is the normalizer.

We can now consider writing out the likelihood (L) and the log likelihood (l) of a set of data $D = \{x_1, \ldots, x_n\}$:

$$L(D|\theta) = \Pi_{j=1}^{n} P(x_j|\theta)$$

$$l(D|\theta) = \sum_{j=1}^{k} \log \left\{ \sum_{i=1}^{k} \pi_i \mathcal{N}(x_j|\mu_i, \sigma_i) \right\}.$$

General problem: if you know the values for z then you could compute μ_i and Σ_i trivially; similarly, if you knew μ and Σ, you could compute the values for z trivially. But we do not know either.

1.5. **K-Means.**

In K-means, each cluster is described by a single mean, or *centroid*, so as not to confuse this model with an actual probabilistic model. There is no underlying probability model. The goal is to group data into k clusters

algorithm

Let z_j^i be a matrix of hard assignments $(n \times k)$, and μ_k be a matrix of centroids $(k \times d)$.

(1) set k; choose initial centroids (for example, by selecting k data points)

(2) assign data x_j to a cluster i: $z_j^i = 1$, by a distance function (e.g., squared L_2, distance)

$$z_j^i = \begin{cases} 1 & \text{if } i = argmin_k ||x_j - \mu_k||^2; \\ 0 & \text{otherwise.} \end{cases}$$

(3) calculate the positions of the centroids:

$$\mu_i = \frac{\sum_(j = 1)^n z_j^i x_j}{\sum_(j = 1)^n z_j^i}$$

(4) iterate 2 and 3 until the assignments, z, no longer change.

Note that the distance function in the cluster assignment step can be chosen specifically for your problem, and is arbitrary.

1.6. **Expectation Maximization (EM).**

We would like to take the intuition from K-means, specifically, iterating from assignments of data points to clusters, and the updating the cluster-specific parameters, and make them probabilistic. In particular, let the z_j^i are now soft assignment variables, meaning they take a value between 0 and 1 and the vector z_j sums to one. In particular, we can now consider the posterior probability of a data point being generated from each class, and set our z_j variables to this posterior probability. We can also incorporate the probability of a data point belonging to each class π_i, and consider this the prior probability of the class (although we are estimating it from data). Finally, we can describe each mixture component in terms of a Gaussian distribution with its own parameters, and compute the likelihood of the data point given its assignment into a component.

In particular, let $\tau_j^i(t) = P(z_j^i = 1 | x_j, \hat{\theta}) = E[z_j^i | x_j, \hat{\theta}]$, and then:

$$\hat{\mu_i} = \frac{\sum_{j=1}^n \tau_j^i x_j}{\sum_{j=1}^n \tau_j^i}.$$

Soft assignment: probability vector describing the posterior probability of x_j in each cluster. Found by averaging all of the points in the cluster, weighted according to their soft assignment to that cluster. Contribution of a single point to the parameters of a single component is proportional to their assignment to that cluster.

EM Algorithm

(1) Initialize the $\mu_k^{(0)}$, $\Sigma_k^{(0)}$, $\pi_k^{(0)}$variables

(2) Expectation step: find the expected values of the assignment variables

$$(\tau_j^i)^{(t+1)} = \frac{\pi_i^{(t)} \mathcal{N}(x_j|\mu_i^{(t)}\Sigma_i^{(t)})}{\sum_{k=1}^K \pi_k^{(t)} \mathcal{N}(x_j|\mu_k^{(t)}\Sigma_k^{(t)})}$$

(3) Maximization step: calculate the means and covariance matrices $\mu_k^{(t)}$, $\Sigma_k^{(t)}$ of each mixture component, and the mixture components $\pi_i^{(t)}$, conditioned on x_j and z_j, e.g.:

$$\begin{aligned}
\hat{\mu_i}^{(t+1)} &= \frac{\sum_{j=1}^n (\tau_j^i)^{(t+1)} x_j}{\sum_{j=1}^n (\tau_j^i)^{(t+1)}} \\
\hat{\Sigma_i}^{(t+1)} &= \frac{\sum_{j=1}^n (\tau_j^i)^{(t+1)} (x_j - \mu_j^{(t+1)})(x_j - \mu_j^{(t+1)})^T}{\sum_{j=1}^n (\tau_j^i)^{(t+1)}} \\
\hat{\pi_i}^{(t+1)} &= \frac{\sum_{j=1}^n (\tau_j^i)}{n}.
\end{aligned}$$

3

(4) Iterate steps 2 and 3 until they have converged.

K-means versus EM:

- EM is a soft assignment versus a hard assignment to a cluster in K-means
- π takes into account the cluster membership in assigning a point to a cluster
- Component parameters based on maximum likelihood estimates in EM, distance-based measure in K-means
- EM guaranteed to converge to a local maxima; no such guarantee in K-means.

Expected complete log likelihood (ECLL): Let's assume that we have actually observed assignment values z for each data point:

$$\begin{aligned}
D_c &= \{(x_1, z_1), \ldots, (x_n, z_n)\} \\
l_c(D_c|\theta) &= \sum_n \log P(x_n, z_n|\theta) \\
&= \sum_n \log \prod_{i=1}^k [\pi_i \mathcal{N}(x_n \mu_i, \sigma_i)]^(z_n^i) \\
&= \sum_n \sum_{i=1}^k z_n \log[\pi_i \mathcal{N}(x_n|\mu_i, \sigma_i)]
\end{aligned}$$

This is called the *complete log likelihood*. We can take the expectation of this in terms of our parameters (call them jointly θ):

$$
\begin{aligned}
< l_c(D_c|\theta) >_{\theta^{(t)}} &= \left\langle \sum_n \sum_{i=1}^{k} z_n^i \log[\pi_i N(x_n|\mu_i, \sigma_i)] \right\rangle_{\theta^{(t)}} \\
&= \sum_n \sum_{i=1}^{k} < z_n^i >_{\theta^{(t)}} \log[\pi_i N(x_n|\mu_i, \sigma_i)] \\
&= \sum_n \sum_{i=1}^{k} \tau_n^i \log[\pi_i N(x_n|\mu_i, \sigma_i)],
\end{aligned}
$$

because τ_n^i is the expected value of z_n^i with respect to the parameters θ.

The ECLL gives us the means to derive the EM updates. For the E-step, we need to compute the expected value of z_n^i in this case; if we also had the term $< z_n^i z_n^j >$ in the ECLL, we would have to compute this as well.

To derive the Maximization step, take the derivative of the ECLL with respect to each of the parameters to update, set to zero, and solve, e.g., $\frac{\partial < l_c(D_c|\theta) >_{\theta^{(t)}}}{\partial(\mu)}$.

1.7. **T. Bailey Paper.** This paper used mixture models, with parameters fitted using EM, to identify motifs (unknown *a priori*) in sequence data.

Different model types (OOPS, ZOOPS, TCM) were developed and used to find motifs in a set of DNA sequences when motifs are unobserved. We can use prior knowledge to help find unknown motifs, but this prior knowledge is not required.

The three different models included:

- OOPS: one occurrence per sequence
- ZOOPS: zero or one occurrence per sequence
- TCM: two component mixture

Given set of genomic sequence, we might want one occurrence of the motif per sequence (OOPS). The parameters include the background frequency of nucleotides, the frequency of the nucleotides within the motif (for each base of the motif) and the locations on each sequence where the motif starts (in expectation). z_i are the starting points of each motif.

The model used a multinomial distribution to describe the distribution of emissions (nucleotides) from the two clusters: motif or background.

MEME was used to find new motifs in a sequence of proteins and identify new members of protein families. Without prior information the model chooses the motif width. Under these conditions OOPS performs better than ZOOPS. The model performs better with prior information. EM performs better than Gibbs sampling, as shown by cross-validation.