Mixture Models

- Motivated by soft k-means we developed a "generative model" for clustering.
- Assume there are \( k \) clusters
- Clusters are not required to have the same number of points
- And not required to have the same shape

Mixture of Normals in 1D

Repeat for \( i = 1, \ldots, N \)

- Pick cluster id \( z_i \) from discrete distribution with parameters \( p_1, p_2, \ldots, p_k \)
  
  \[
  \text{Note: } z_i \in \{1, 2, \ldots, k\}
  \]

- Pick the example \( x_i \) from normal distribution with parameters \( \mu_{z_i}, \sigma_{z_i} \)

Example: when \( z_i = 3 \) using \( \mu_3 \) and \( \sigma_3 \)

Given a dataset generated by this process, the clustering task is to identify the parameters \( \{p_j, \mu_j, \sigma_j\}, j = 1, \ldots, k \)

Maximum likelihood estimation

- First analyze assuming \( z_i \) are known
- Convenient notation: represent the number \( z_i \) as a "unit vector" bit sequence

Example: \( k = 4 \)

\[
\begin{align*}
  z_i &= 1 \Rightarrow 1000 \\
  z_i &= 2 \Rightarrow 0100 \\
  z_i &= 3 \Rightarrow 0010 \\
  z_i &= 4 \Rightarrow 0001
\end{align*}
\]

- Notation: \( z_{i,j} \) is \( j \)th bit within \( z_i \)

\[
\begin{align*}
  z_i &= 2 \Rightarrow 0100 \Rightarrow z_{1,2} = 1 \\
  z_{1,3} &= 0
\end{align*}
\]

Maximum likelihood estimation

- The Likelihood

\[
L = \prod_i p(z_i)p(x_i|z_i, \mu_{z_i})
= \prod_i (1/k) \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1}{2\sigma^2}(x_i-\mu_{z_i})^2}
= \prod_i (1/k) \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1}{2\sigma^2} \sum_j z_{i,j} (x_i-\mu_j)^2}
\]

Notation trick: exactly one term remains from the sum!
Maximum likelihood estimation

\[ L = \prod_i (1/k) \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{1}{2}\sum_i z_{i,j}(x_i-\mu_j)^2} \]

\[ \log L = \text{const} - \frac{1}{2\sigma^2} \sum_i \sum_j z_{i,j}(x_i-\mu_j)^2 \]

\[ \frac{\partial \log L}{\mu_j} = \ldots = 0 \]

\[ \mu_j = \frac{\sum_i z_{i,j} x_i}{\sum_i z_{i,j}} \]

We solved this in the last lecture

---

The EM Algorithm

- A general algorithm for maximizing marginal likelihood - i.e. maximizing likelihood when we have hidden random variables
- The algorithm has a simple form when applied to mixture models (and some other models)
- We first explain this special form and then show how to derive it from the general scheme of the EM algorithm

The EM Algorithm

- EM is an iterative algorithm
- Initialize probability model \( p' \)
- Repeat
  - use \( p' \) to calculate an improved model \( p'' \)
  - Set \( p' = p'' \)
- Until no further improvement

EM Algorithm for Mixture Models

- Repeat
  - [E] Calculate using \( p' \)
    \[ f_{i,j} = E_{\mu_i|z_i,x_i}(z_{i|j}) = p(z_i = j|\mu'_i, \text{Data}) \]
  - [M] Estimate \( p'' \) parameters using maximum likelihood replacing the unknown \( z_{i,j} \) by \( f_{i,j} \)

EM for Mixtures in 1D

- [E] Calculate

\[ f_{i,j} = E_{\mu_i|x_i}(z_{i,j}) = \frac{p(z_i = j \text{ and } x_i)}{p(x_i)} = \frac{p(z_i = j \text{ and } x_i)}{\sum_{\ell} p(z_i = \ell \text{ and } x_i)} \]

\[ = \frac{p(z_i = j \text{ and } x_i)}{\sum_{\ell} (1/k) \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{1}{2\sigma}(x_i-\mu'_\ell)^2}} \]

First part holds for any mixture model.
EM for Mixtures in 1D

- [M] Estimate parameters using max likelihood replacing the unknown $z_{i,j}$ by $f_{i,j}$

$$\mu_j = \frac{\sum_i z_{i,j} x_i}{\sum_i z_{i,j}} \Rightarrow \mu''_j = \frac{\sum_i f_{i,j} x_i}{\sum_i f_{i,j}}$$

General form of EM

- Define an auxiliary function $Q(p',p'')$
- Relative to observed variables $O$ and hidden variables $H$

$$Q(p',p'') = E_{p'(H|O)}[\log p''(H,O)]$$

The EM Algorithm

- EM is an iterative algorithm
- Initialize probability model $p'$
- Repeat
  - Pick $p''$ so as to maximize $Q(p',p'')$
  - Set $p'' = p''$
- Until no further improvement

The EM Algorithm

- Fact:
  - If $Q(p',p'') > Q(p',p')$
    [this means that $p''$ increases the value of $Q()$]
    Then $p'(\text{observed}) > p'(\text{observed})$
  - Therefore iterations of EM increase the likelihood and the algorithm converges to a (possibly local) maximum of the likelihood.

Reminder: Maximum likelihood

$$L = \prod_i \left( \frac{1}{k} \right) \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1}{2\sigma^2} \sum_j z_{i,j}(x_i - \mu_j)^2}$$

$$\log L = \text{const} - \frac{1}{2\sigma^2} \sum_i \sum_j z_{i,j}(x_i - \mu_j)^2$$

$$\frac{\partial \log L}{\mu_j} = \ldots = 0 \Rightarrow \mu_j = \frac{\sum_i z_{i,j} x_i}{\sum_i z_{i,j}}$$

We solved this in the last lecture

EM for Mixtures in 1D

- Develop algorithm from the general template
  - $Q(p',p'') = E_{p'(H|O)}[\log p''(H,O)]$
  - $Q(p',p'')$
    - $E_{p'(Z|X)}[\log p''(Z,X)]$
    - $E_{p'(Z|X)}[\text{const} - \frac{1}{2\sigma^2} \sum_i \sum_j z_{i,j}(x_i - \mu''_{i,j})^2]$

What is the random variable in this expectation? Only $z_{i,j}$
EM for Mixtures in 1D

\[ Q(p', p'') = E_{\mu}(Z|X)[\log p''(Z, X)] \]
\[ = E_{\mu}(Z|X)[\text{const} - \frac{1}{2\sigma^2} \sum_i \sum_j z_{i,j} (x_i - \mu_j^0)^2] \]
\[ = \text{const} - \frac{1}{2\sigma^2} \sum_i \sum_j E_{\mu}(Z|X) [z_{i,j} (x_i - \mu_j^0)^2] \]
\[ = \text{const} - \frac{1}{2\sigma^2} \sum_i \sum_j f_{i,j} (x_i - \mu_j^0)^2 \]

We get exactly the same Eq as before!

EM Algorithm for Mixture Models

• Repeat
  - [E] Calculate using \( p' \)
    \[ f_{i,j} = E_{\mu}(Z|X, \{\mu_k^0\})[z_{i,j}] = p(z_i = j|\mu_k^0), \text{Data} \]
  - [M] Estimate \( p'' \) parameters using max likelihood replacing the unknown \( z_{i,j} \) by \( f_{i,j} \)

Using the same methodology on any mixture model (not just Gaussian) yields the same template.

Semi-Supervised Naïve Bayes Model

• Naïve Bayes: Probabilistic model with strong simplifying assumptions
• Illustrating application: text categorization where we have data for (document, label)
• What if we have many documents but labels for only a few of them?
• Can the unlabeled documents help?

Recall: Naïve Bayes Model

• Each class induces a distribution over features.
• Features are conditionally independent given the class
• On slides I use model for binary features

Recall: Naïve Bayes Model

\[ p(z_i = j) = p_j \]
\[ p(x_{i,\ell} = 1|\text{class } j) = q_{j,\ell} \]
\[ p(x_{i,\ell}|\text{class } j) = \prod_{\ell} q_{j,\ell}^{x_{i,\ell}} (1 - q_{j,\ell})^{1-x_{i,\ell}} \]
\[ p(z_i = j \text{ and } x_i) = p_j \prod_{\ell} q_{j,\ell}^{x_{i,\ell}} (1 - q_{j,\ell})^{1-x_{i,\ell}} \]
\[ p(z_i \text{ and } x_i) = \prod_{j} \left[ p_j \prod_{\ell} q_{j,\ell}^{x_{i,\ell}} (1 - q_{j,\ell})^{1-x_{i,\ell}} \right]^{z_{i,j}} \]
### Recall: Maximum Likelihood

\[ p_j = p(z_i = j) = \frac{\text{number of examples with class } j}{\text{number of examples}} \]

\[ q_{j,i} = p(x_{i,t} = 1 | z_i = j) = \frac{\text{num of ex with class } j \text{ and } x_{i,t} = 1}{\text{number of examples with class } j} \]

### Naïve Bayes for Text Classification

- **Classes:** Yes, No
- **Features:** word-slot in document has value word-\(i\) in lexicon
- **Features have a huge number of values**
- **Number of features is document length**

In the application the features are not binary.
In the application we assumed that all features (word slots) have the same distribution.
We keep to the binary model in the slides.

### Naïve Bayes as Mixture Model

Repeat for \(i = 1, \ldots, N\)

1. Pick cluster Id \(z_i\) from discrete distribution with parameters \(p_1, p_2, \ldots, p_k\)
2. Pick the example \(x_i\) from Naive Bayes distribution with parameters \(q_{z_i}\)

### EM Algorithm

- **Complete Data Likelihood**

\[ L = \prod_{i} \prod_{j} \left[ p_j \prod_{\ell} q_{j,\ell}^{x_{i,\ell}} (1 - q_{j,\ell})^{(1-x_{i,\ell})} \right]^{z_{i,j}} \]

- **Log Likelihood**

\[ \log L = \sum_{j} z_{i,j} \left( \log p_j + \sum_{\ell} x_{i,\ell} \log q_{j,\ell} + (1-x_{i,\ell}) \log (1-q_{j,\ell}) \right) \]

What is the random variable in this expectation? Only \(z_{i,j}\)
**Semi-Supervised Naïve Bayes Model**

- What if we have many documents but labels for only a few of them?
- Can the unlabeled documents help?
- Use EM: for examples where $z_i$ is known use $f_{i,j} = z_{i,j}$ instead of estimating it
- Nothing else changes in the algorithm!

**EM Algorithm**

- Same steps as before
- Calculate $f_{i,j}$
- Solve Maximum likelihood using $f_{i,j}$ instead of $z_{i,j}$

**EM Algorithm**

- E Step: Calculating $f_{i,j}$

$$f_{i,j} = E_p(z_i | x_i) = \frac{p'(z_i = j \text{ and } x_i)}{\sum_c p'(z_i = c \text{ and } x_i)} = \frac{p'_{j,i} \prod \ell q'_{\ell,\ell}(1 - q'_{\ell,\ell}) (1 - x_{i,\ell})}{\sum_c p'_{c,i} \prod \ell q'_{\ell,\ell}(1 - q'_{\ell,\ell}) (1 - x_{i,\ell})}$$

**EM Algorithm for Naïve Bayes**

- Repeat
  - Calculate:
    $$f_{i,j} = \frac{p'_{j,i} \prod \ell q'_{\ell,\ell}(1 - q'_{\ell,\ell}) (1 - x_{i,\ell})}{\sum_c p'_{c,i} \prod \ell q'_{\ell,\ell}(1 - q'_{\ell,\ell}) (1 - x_{i,\ell})}$$
  - Calculate:
    $$p'_j = \frac{\sum_i f_{i,j}}{N}$$
    $$q'_{j,\ell} = \frac{\sum_i f_{i,j} x_{i,\ell}}{\sum_i f_{i,j}}$$
  - Assign: $p' \leftarrow p''$ and $q' \leftarrow q''$

**20 newsgroups data**

[Graph showing classification accuracy of traditional naive Bayes (no unlabeled data) with an EM algorithm for different amounts of labeled training data. Each of these results is statistically significant (a plateau, having unlabeled data does not help nearly as much, because there is a lot of labeled data, and the naive Bayes learning curve is close to 70%). This represents a 30% reduction in classification error. Note that EM also performs well even with a very small number of labeled documents; with only 20 documents (a single labeled document per class), naive Bayes obtains 20%, EM 35%. As expected, EM performs significantly better. For example, with 300 labeled documents (15% classification accuracy), naive Bayes requires 2000 labeled examples, while EM requires only 600 labeled examples to achieve the same accuracy. These results demonstrate that EM finds parameter estimates that improve classification accuracy and reduce the need for labeled training examples. For example, with 300 labeled documents, 15% classification accuracy, naive Bayes requires 2000 labeled examples, while EM achieves 66% accuracy.]
Figure 3. Classification accuracy while varying the number of unlabeled documents. The effect is shown on the 20 Newsgroups data set, with 5 different amounts of labeled documents, by varying the amount of unlabeled data on the horizontal axis. Having more unlabeled data helps. Note the dip in accuracy when a small amount of unlabeled data is added to a small amount of labeled data. We hypothesize that this is caused by extreme, almost 0 or 1, estimates of component membership, \( P(\hat{\alpha}) \), for the unlabeled documents (as caused by naive Bayes' word independence assumption).

In Figure 3 we consider the effect of varying the amount of unlabeled data. For five different quantities of labeled documents, we hold the number of labeled documents constant, and vary the number of unlabeled documents in the horizontal axis. Naturally, having more unlabeled data helps, and it helps more when there is less labeled data. Notice that adding a small amount of unlabeled data to a small amount of labeled data actually hurts performance. We hypothesize that this occurs because the word independence assumption of naive Bayes leads to overly-confident \( P(\hat{\alpha}) \) estimates in the E-step, and the small amount of unlabeled data is distributed too sharply. (Without this bias in naive Bayes, the E-step would spread the unlabeled data more evenly across the classes.) When the number of unlabeled documents is large, however, this problem disappears because the unlabeled set provides a large enough sample to smooth out the sharp discreteness of naive Bayes' overly-confident classification.

We now move on to a different data set. To provide some intuition about why EM works, we present a detailed trace of one example from the WebKB data set. Table 3 shows the evolution of the classifier over the course of two EM iterations. Each column shows the ordered list of words that the model indicates are most "predictive" of the \( \text{course} \) class. Words are judged to be "predictive" using a weighted log likelihood ratio.

The symbol \( D \) indicates an arbitrary digit. At Iteration 0, the parameters are estimated from a randomly-chosen single labeled document per class. Notice that the document seems to be about a specific Artificial Intelligence course at Dartmouth. After two EM iterations with 2500 unlabeled documents, we see that EM has used the unlabeled data to find words that are more generally course-related. Without this bias in naive Bayes, the E-step would spread the unlabeled data more evenly across the classes. When the number of unlabeled documents is large, however, this problem disappears because the unlabeled set provides a large enough sample to smooth out the sharp discreteness of naive Bayes' overly-confident classification.

We now move on to a different data set. To provide some intuition about why EM works, we present a detailed trace of one example from the WebKB data set. Table 3 shows the evolution of the classifier over the course of two EM iterations. Each column shows the ordered list of words that the model indicates are most "predictive" of the course class. Words are judged to be "predictive" using a weighted log likelihood ratio.

The symbol \( D \) indicates an arbitrary digit. At Iteration 0, the parameters are estimated from a randomly-chosen single labeled document per class. Notice that the document seems to be about a specific Artificial Intelligence course at Dartmouth. After two EM iterations with 2500 unlabeled documents, we see that EM has used the unlabeled data to find words that are more generally course-related. Without this bias in naive Bayes, the E-step would spread the unlabeled data more evenly across the classes. When the number of unlabeled documents is large, however, this problem disappears because the unlabeled set provides a large enough sample to smooth out the sharp discreteness of naive Bayes' overly-confident classification.