EM and mixture models

D.A. Forsyth, UIUC
Example problems

- Examples:
  - Estimate a mean in the presence of outliers
  - Segmentation using a mixture of Gaussians
  - A topic model for words (or image patches)

- Characteristic:
  - Missing information which would simplify the problem, if known
    - Which data points are outliers
    - Which segment each pixel comes from
    - Which topic each word comes from
Mean in the presence of outliers

- Observations:
  - either 1D data or 1D outlier
  - data is normal, with known variance
  - mixture model, which is generative
    - i.e. a model of how data is generated

\[
P(x|\mu, \sigma, \pi) = \pi P_1(x|\mu, \sigma) + (1 - \pi) P_2(x)
\]
Loglikelihood

- Unpromising, because a sum of logs of sums

\[
\sum_i \log P(x_i | \mu, \sigma, \pi) = \sum_i \log [\pi P_1(x_i | \mu, \sigma) + (1 - \pi) P_2(x_i)]
\]
Models of data, outliers

- **Data:**
  - simplest is normal
- **Outliers:**
  - simplest is a constant
  - duck issues with normalization for the moment
- doesn’t help

\[
\sum_i \log P(x_i | \mu, \sigma, \pi) = \sum_i \log \left[ \pi K_1 \exp \frac{-(x_i - \mu)^2}{2\sigma^2} + (1 - \pi) K_2 \right]
\]
Much easier if...

- We knew where each data item came from
  - Introduce a “switch” $d$ which is 0 if it came from data, 1 if outlier
  - we don’t know these, but pretend we do
  - now the probability of a data item $x$, $d$ is:

$$P(x, \delta|\mu, \sigma, \pi) = \left[ \pi K_1 \exp \frac{-(x_i - \mu)^2}{2\sigma^2} \right]^{(1-\delta)} \left[ (1 - \pi)K_2 \right]^\delta$$
Complete data loglikelihood

- Pretend we knew a $d$ for each $x$
  - the loglikelihood would be

$$
\sum_i \log (P(x_i, \delta_i | \mu, \sigma, \pi)) = \sum_i \left\{ (1 - \delta_i) \left[ \log \pi + \log K_1 + \frac{-(x_i - \mu)^2}{2\sigma^2} \right] + \delta_i \left[ \log(1 - \pi) + \log K_2 \right] \right\}
$$

- (for reference, expression from previous slide below)

$$
P(x, \delta | \mu, \sigma, \pi) = \left[ \pi K_1 \exp \frac{-(x_i - \mu)^2}{2\sigma^2} \right]^{(1-\delta)} \left[ (1 - \pi) K_2 \right]^\delta
$$
Complete data loglikelihood

- This is a function of parameters
  - (and all the d, because we don’t know them)
  - write as

\[ L_c(\theta, \delta) \]

For our example, this has the form:

\[
\sum_i \log(P(x_i, \delta_i | \mu, \sigma, \pi)) = \sum_i \left\{ (1 - \delta_i) \left[ \log \pi + \log K_1 + \frac{-(x_i - \mu)^2}{2\sigma^2} \right] + \delta_i \left[ \log(1 - \pi) + \log K_2 \right] \right\}
\]
Expectation-Maximization

• Notice:
  • with an estimate of the parameters, can estimate whether x is outlier, noise
    • this could be a soft estimate
    • we could plug this in, then reestimate the parameters

• Formal procedure:
  • parameters are theta
  • start with estimate theta\(^n\)
  • form Q function, below (The E-step)
    • maximise in parameters (The M-step)
  • Iterate

\[
Q(\theta; \theta^{(n)}) = E_{P(\delta|x,\theta^{(n)})} (L_c(\theta, \delta))
\]
Expectation-Maximization

- Notice:
  - $Q$ incorporates all we know about $d$ given our current estimate of $\theta$
  - so
    - estimate $\theta = \theta^{(n)}$
    - now average out $d$ using this estimate
      - but leaving $\theta$ alone
      - to get a new function of $\theta$
    - now this contains all we know about $d$ given estimate, so maximize
  - repeat till convergence.

\[
Q(\theta; \theta^{(n)}) = E_{P(\delta|x,\theta^{(n)})} (L_c(\theta, \delta))
\]
In our case ....

\[
E_{P(\delta|x, \theta^{(n)})} L_c(\theta, \delta) = \sum_{\delta} P(\delta|x, \theta^{(n)}) \log \left( P(x, \delta|\mu, \sigma, \pi) \right)
\]

\[
= \sum_{\delta} P(\delta|x, \theta^{(n)}) \left\{ \sum_i \log \left( P(x_i, \delta_i|\mu, \sigma, \pi) \right) \right\}
\]

\[
= \sum_i \left[ P(\delta_i = 0|x, \theta^{(n)}) \log P(x_i, \delta_i = 0|\mu, \sigma, \pi) + P(\delta_i = 1|x, \theta^{(n)}) \log P(x_i, \delta_i = 1|\mu, \sigma, \pi) \right]
\]
In our case...

\[
P(\delta_i = 1|\mathbf{x}_i, \theta^{(n)}) = \frac{P(\mathbf{x}, \delta_i = 1|\theta^{(n)})}{P(\mathbf{x}_i|\theta^{(n)})} = \frac{P(\mathbf{x}, \delta_i = 1|\theta^{(n)})}{P(\mathbf{x}_i, \delta_i = 0|\theta^{(n)}) + P(\mathbf{x}_i, \delta_i = 1|\theta^{(n)})}
\]

\[
= \frac{P(\mathbf{x}|\delta_i = 1, \theta^{(n)})P(\delta_i = 1|\theta^{(n)})}{\left[ P(\mathbf{x}_i|\delta_i = 0, \theta^{(n)})P(\delta_i = 0|\theta^{(n)}) + P(\mathbf{x}_i|\delta_i = 1, \theta^{(n)})P(\delta_i = 1|\theta^{(n)}) \right]}
\]

\[
= \frac{K_2(1 - \pi^{(n)})}{K_1 \exp \frac{-(\mathbf{x}_i - \mu^{(n)})^2}{2\sigma^2} \pi(n) + K_2(1 - \pi^{(n)})}
\]
\[ \sum_{i} \left[ P(\delta_i = 0|x, \theta^{(n)}) \log P(x_i, \delta_i = 0|\mu, \sigma, \pi) + \\
\quad P(\delta_i = 1|x, \theta^{(n)}) \log P(x_i, \delta_i = 1|\mu, \sigma, \pi) \right] \]

Substitute expression below

\[ P(x, \delta|\mu, \sigma, \pi) = \left[ \pi K_1 \exp \frac{-(x_i - \mu)^2}{2\sigma^2} \right]^{(1-\delta)} [(1 - \pi) K_2]^\delta \]

To get:

\[ \sum_{i} \left[ P(\delta_i = 0|x, \theta^{(n)}) \log \left[ \pi K_1 \exp \frac{-(x_i - \mu)^2}{2\sigma^2} \right] + \\
\quad P(\delta_i = 1|x, \theta^{(n)}) \log [(1 - \pi) K_2] \right] \]
In our case, E-step

- Obtain $P(d|x, \theta^n)$ terms
  - often called soft weights
- Substitute to get a Q function that looks like:

\[
\sum_i \left[ w_i^{(n)} \left[ \log \pi + \log K_1 + \frac{-(x_i - \mu)^2}{2\sigma^2} \right] + (1 - w_i^{(n)}) \left[ \log(1 - \pi) + \log K_2 \right] \right]
\]
Maximizing

• For the m-step, the P(lltheta^n) terms are constants

\[ \sum_i w_i^{(n)} \left[ \log \pi + \log K_1 + \frac{-(x_i - \mu)^2}{2\sigma^2} \right] + (1 - w_i^{(n)}) \left[ \log(1 - \pi) + \log K_2 \right] \]

• Gradient, set to zero

\[ \pi^{(n+1)} = \frac{\sum_i w_i^{(n)}}{N} \]
\[ \mu^{(n+1)} = \frac{\sum_i w_i^{(n)} x_i}{N} \]
All together...

- Start with estimate of mean, $\pi$
  - get soft weights from estimate
  - reestimate mean, $\pi$
  - until (convergence)
Mixture models and segmentation

- In k-means, we clustered pixels using hard assignments
  - each pixel goes to closest cluster center
  - but this may be a bad idea
    - pixel may help estimate more than one cluster
- We will build a probabilistic mixture model

\[
P(x | \mu_1, \ldots, \mu_k, \pi_1, \ldots, \pi_k, \Sigma) = \sum_i \pi_i P(x | \mu_i, \Sigma)
\]
Mixture model

- Interpretation:
  - obtain pixel by
    - choosing a mixture component
    - given that component, choosing pixel
- Natural to have each mixture component be a Gaussian

\[
P(x | \mu_1, \ldots, \mu_k, \pi_1, \ldots, \pi_k, \Sigma) = \sum_i \pi_i P(x | \mu_i, \Sigma)
\]
Mixture components

- Gaussians
  - are oriented “blobs” in the feature space
  - we will assume covariance is known, and work with mean
  - expression below

\[ P(x|\mu_i, \Sigma) \propto \exp \left( -\frac{(x - \mu_i)\Sigma^{-1}(x - \mu_i)}{2} \right) \]
Problem: Learning and IDLL

- We must estimate the mixture weights and means
- Maximising likelihood is very hard
  - in this form, sometimes known as incomplete data log-likelihood

$$L(\theta) = \sum_i \log P(x_i|\theta) = \sum_i \log \left( \sum_j \pi_j P(x_i|\mu_j, \Sigma) \right)$$
Complete Data Log-likelihood

- Learning would be easy if we knew which blob each data item came from
  - weights: count what fraction came from which blob
  - means: average data items
- Introduce a set of variables
  - to tell which mixture component a data item came from
  - $d_{ij}$ is 1 if data item $i$ comes from blob $j$
    - 0 otherwise

\[ P(x_i | \delta_i, \theta) = \prod_j P(x_i | \mu_j)^{\delta_{ij}} \]
Complete Data Log-likelihood

• Write the probability for $x_i, d_{ij}$ conditioned on params

\[
P(x_i, \delta_i | \theta) | \theta = P(x_i | \delta_i, \theta) P(\delta_i | \theta) | \theta
\]

\[
= \prod_j \left[ P(x_i | \mu_j) \pi_{ij} \right]^{\delta_{ij}}
\]
Complete Data Log-likelihood

- Log likelihood of $x$, allocation, conditioned on parameters
- Not obviously helpful
  - but notice the useful form - think of $d$ as switches

$$
\log P(x, \delta | \theta) = \sum_i \log P(x_i, \delta_i | \theta)
$$
$$
= \sum_{ij} [\delta_{ij} \{\log P(x_i | \mu_j) + \log \pi_j\}]
$$
Learning and CDLL

- Introduce hidden variables to get complete data log-likelihood
  - $d_{ij}$ is 1 if data item $i$ comes from blob $j$
    - 0 otherwise
  - Learning would be easy if we knew which blob each data item came from
    - weights: count what fraction came from which blob
    - means: average data items
  - But we don’t

$$L_c(\theta, \delta_{ij}) = \sum_{ij} \delta_{ij} \log (\pi_j P(x_i | \mu_j, \Sigma))$$
Working with CDLL

• Notice:
  • with an estimate of the parameters, can estimate blobs data came from
    • this could be a soft estimate
    • we could plug this in, then reestimate the parameters

• Formal procedure:
  • Quite general
  • start with estimate
  • form Q function, below (The E-step)
    • maximise in parameters (The M-step)
  • Iterate

\[
Q(\theta; \theta^{(n)}) = E_{P(\delta|x,\theta^{(n)})} (L_c(\theta, \delta))
\]
The E-step for Gaussian mixtures

• Notice that the expression for the Q function simplifies

\[
E_{P(\delta|x, \theta^{(n)})} \left( L_c(\theta, \delta) \right) = E_{P(\delta|x, \theta^{(n)})} \left( \sum_{ij} \delta_{ij} \left[ \log \pi_j + \log P(x_i|\mu_j, \Sigma) \right] \right)
\]

\[
= \left( \sum_{ij} P(\delta_{ij} = 1|x, \theta^{(n)}) \left[ \log \pi_j + \log P(x_i|\mu_j, \Sigma) \right] \right)
\]
The E-step for Gaussian mixtures

- We rearrange using probability identities

\[
P(\delta_{ij} = 1|x, \theta^{(n)}) = \frac{P(x, \delta_{ij} = 1|\theta^{(n)})}{P(x|\theta^{(n)})} = \frac{P(x|\delta_{ij} = 1, \theta^{(n)})P(\delta_{ij} = 1|\theta^{(n)})}{P(x|\theta^{(n)})} = \frac{P(x|\delta_{ij} = 1, \theta^{(n)})P(\delta_{ij} = 1|\theta^{(n)})}{P(x|\delta_{ij} = 1, \theta^{(n)})P(\delta_{ij} = 1|\theta^{(n)}) + P(x, \delta_{ij} = 0|\theta^{(n)})}
\]
The E step for Gaussian mixtures

• And substitute

\[
P(x_i, \delta_{ij} = 0|\theta^{(n)}) = \sum_{k \neq j} \pi_k \frac{1}{Z} \exp \left( -\frac{(x_i - \mu_k^{(n)})\Sigma^{-1}(x_i - \mu_k^{(n)})}{2} \right)
\]

\[
P(x_i|\delta_{ij} = 1, \theta^{(n)}) = \frac{1}{Z} \exp \left( -\frac{(x_i - \mu_j^{(n)})\Sigma^{-1}(x_i - \mu_j^{(n)})}{2} \right)
\]

\[
P(\delta_{ij} = 1|\theta^{(n)}) = \pi_j
\]
The M step for Gaussian mixtures

- We must maximise

\[
\sum_{ij} P(\delta_{ij} = 1|x, \theta^{(n)}) \left[ \log \pi_j + \log P(x_i|\mu_j, \Sigma) \right] =
\sum_{ij} P(\delta_{ij} = 1|x, \theta^{(n)}) \left[ \log \pi_j - \frac{(x_i - \mu_j)^T \Sigma^{-1} (x_i - \mu_j)}{2} - \log Z \right]
\]

- in the mixture weights and in the means
  - we can drop log Z
Two ways

- differentiate, set to zero, etc.
- regard the expectations as “soft counts”
  - so mixture weights from soft counts as:

\[
\pi_j = \frac{\sum_i P(\delta_{ij} = 1|x_i, \theta^{(n)})}{\sum_{i,j} P(\delta_{ij} = 1|x_i, \theta^{(n)})}
\]

- and means from soft counts as:

\[
\mu_j = \frac{\sum_i x_i P(\delta_{ij} = 1|x_i, \theta^{(n)})}{\sum_{i,j} P(\delta_{ij} = 1|x_i, \theta^{(n)})}
\]
Segmentation with EM

Another Application

- Clustering binary codes
  - or modelling their source
    - document clustering
    - image clustering
    - detecting objects in attribute lists
- Example: observe ->
  - these bits probably aren’t independent
  - but what do we do?
    - two independent sources, mixed

```
1001 x8
0110 x8
1011 x1
1101 x1
0111 x1
1110 x1
```
Model