Unsupervised learning or Clustering (cont.) –
K-means
Gaussian mixture models

Machine Learning – 10701/15781
Carlos Guestrin
Carnegie Mellon University
April 5th, 2006
Some Data

nobody tells you what the clusters are...

classification must tell you class

give x \rightarrow \text{must}

(train data) supervised

(x_i, y_i) know label

cluster must
give x \rightarrow \text{tell you cluster}

(train data) unsupervised

(x_i) no labels
K-means

1. Ask user how many clusters they’d like. *(e.g. k=5)*
2. Randomly guess k cluster Center locations
3. Each datapoint finds out which Center it’s closest to.
4. Each Center finds the centroid of the points it owns...
5. ..and jumps there
6. ..Repeat until terminated!
K-means

Randomly initialize $k$ centers

- $\mu^{(0)} = \mu_1^{(0)}, \ldots, \mu_k^{(0)}$

Classify: Assign each point $j \in \{1, \ldots, m\}$ to nearest center:
- $C(t)(j) \leftarrow \arg\min_{\text{points} i} \|\mu_i - x_j\|^2$

Recenter: $\mu_i$ becomes centroid of its point:
- $\mu_i^{(t+1)} \leftarrow \arg\min_{\mu} \sum_{j: C(j) = i} \|\mu - x_j\|^2$

Equivalent to $\mu_i \leftarrow \text{average of its points!}$
What is K-means optimizing?

- Potential function $F(\mu, C)$ of centers $\mu$ and point allocations $C$:
  \[
  F(\mu, C) = \sum_{j=1}^{m} ||\mu_{C(j)} - x_j||^2
  \]
  \[
  = \sum_{i=1}^{K} \sum_{j: C(j) = i} ||\mu_i - x_j||^2
  \]

- Optimal K-means:
  \[
  \min_{\mu} \min_{C} F(\mu, C)
  \]
Does K-means converge?? Part 1

- Optimize potential function:

\[
\min_{\mu} \min_C F(\mu, C) = \min_{\mu} \min_C \sum_{i=1}^{k} \sum_{j: C(j) = i} ||\mu_i - x_j||^2
\]

- Fix \( \mu \), optimize \( C \)

\[
\text{fix } \mu = \mu_i \\
\min_C \sum_{i=1}^{k} \sum_{j: C(j) = i} ||\mu_i - x_j||^2 = \min_C \sum_{j=1}^{m} ||\mu(C(j)) - x_j||^2
\]

\( C(j) \subseteq \arg\min_i ||\mu_i - x_j||^2 \)

achieve minimum.. exactly the classification step in K-means
Does K-means converge?? Part 2

- Optimize potential function:
  \[
  \min_{\mu} \min_{C} F(\mu, C) = \min_{\mu} \min_{C} \sum_{i=1}^{k} \sum_{j: C(j)=i} ||\mu_i - x_j||^2
  \]

- Fix C, optimize \( \mu \)
  \[
  \min_{\mu} \sum_{i=1}^{k} \sum_{j: C(j)=i} ||\mu_i - x_j||^2 = \sum_{i=1}^{k} \min_{\mu_i} \sum_{j: C(j)=i} ||\mu_i - x_j||^2
  \]

  \( \mu_i \) should be centroid

  Recentering step in K-means

fix \( C^{(t+1)} \)

\[\text{opt to get } \mu^{(t+1)}\]

\[F(C^{(t+1)}, \mu^{(t+1)}) \leq F(C^{(t+1)}, \mu^{(t)})\]
Coordinate descent algorithms

Want: \( \min_a \min_b F(a,b) \)

Coordinate descent:
- fix a, minimize b
- fix b, minimize a
- repeat

Converges!!!
- if F is bounded
- to a (often good) local optimum
  - as we saw in applet (play with it!)

K-means is a coordinate descent algorithm!
(One) bad case for k-means

- Clusters may overlap
- Some clusters may be “wider” than others

![Diagram showing a bad case for k-means with overlapping clusters and note: can't do this with k-means!!]
Gaussian Bayes Classifier Reminder

\[ P(y = i \mid x_j) = \frac{p(x_j \mid y = i)P(y = i)}{p(x_j)} \]

\[ P(y = i \mid x_j) \propto \frac{1}{(2\pi)^{m/2} \| \Sigma_i \|^{1/2}} \exp \left[ -\frac{1}{2} (x_j - \mu_i)^T \Sigma_i^{-1} (x_j - \mu_i) \right] P(y = i) \]

\[ p(x_j \mid y = i) \propto N(\mu_i, \Sigma_i) \]

\[ \mu_i = \begin{pmatrix} 10 \\ -5 \\ 200 \end{pmatrix} \]

\[ \Sigma_i = \begin{pmatrix} \sigma_{i1}^2 & \sigma_{i2} & \sigma_{i3} \\ \sigma_{i2} & \sigma_{i2}^2 & \sigma_{i3} \\ \sigma_{i3} & \sigma_{i3} & \sigma_{i3}^2 \end{pmatrix} \]

\[ p(y = A \mid x_j) \propto 0.4 \quad 0.8 \]

\[ p(y = B \mid x_j) \propto 0.1 \quad 0.2 \]
Predicting wealth from age

wealth = poor
(prior = 0.760718)

<table>
<thead>
<tr>
<th>1</th>
<th>mean</th>
<th>cov</th>
</tr>
</thead>
<tbody>
<tr>
<td>age</td>
<td>37.374</td>
<td>198.935</td>
</tr>
</tbody>
</table>

wealth = rich
(prior = 0.239282)

<table>
<thead>
<tr>
<th>1</th>
<th>mean</th>
<th>cov</th>
</tr>
</thead>
<tbody>
<tr>
<td>age</td>
<td>44.7727</td>
<td>111.618</td>
</tr>
</tbody>
</table>

density
Predicting wealth from age
Learning modelyear, mpg --- maker

Σ is 2x2 [2 continuous features]

\[ \Sigma = \begin{pmatrix}
\sigma_{11} & \sigma_{12} & \cdots & \sigma_{1m} \\
\sigma_{21} & \sigma_{22} & \cdots & \sigma_{2m} \\
\vdots & \vdots & \ddots & \vdots \\
\sigma_{m1} & \sigma_{m2} & \cdots & \sigma_{mm}
\end{pmatrix} \]

- maker = america
  - prior = 0.625
  - mean: mpg = 20.0335, mod_year = 75.5918
  - cov: mpg = 41.4785, mod_year = 15.2912

- maker = asia
  - prior = 0.201531
  - mean: mpg = 30.4506, mod_year = 77.443
  - cov: mpg = 37.0887, mod_year = 12.6427

- maker = europe
  - prior = 0.173489
  - mean: mpg = 27.6029, mod_year = 75.6765
  - cov: mpg = 43.2988, mod_year = 11.3562

Tilted because \( \sigma \) is \( \neq 0 \)
General: $O(m^2)$ parameters

$$
\Sigma = \begin{pmatrix}
\sigma_{11} & \sigma_{12} & \cdots & \sigma_{1m} \\
\sigma_{12} & \sigma_{22} & \cdots & \sigma_{2m} \\
\vdots & \vdots & \ddots & \vdots \\
\sigma_{1m} & \sigma_{2m} & \cdots & \sigma_{mm}
\end{pmatrix}
$$
Aligned: $O(m)$ parameters

\[ \Sigma = \begin{pmatrix} \sigma^2_1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & \sigma^2_2 & 0 & \cdots & 0 & 0 \\ 0 & 0 & \sigma^2_3 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & \sigma^2_{m-1} & 0 \\ 0 & 0 & 0 & \cdots & 0 & \sigma^2_m \end{pmatrix} \]

Each class has only variance parameters.

Axis aligned.
Aligned: $O(m)$ parameters

$$\Sigma = \begin{pmatrix}
\sigma^2_1 & 0 & 0 & \cdots & 0 & 0 \\
0 & \sigma^2_2 & 0 & \cdots & 0 & 0 \\
0 & 0 & \sigma^2_3 & \cdots & 0 & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & 0 & \cdots & \sigma^2_{m-1} & 0 \\
0 & 0 & 0 & \cdots & 0 & \sigma^2_m
\end{pmatrix}$$
Spherical: $O(1)$ cov parameters

$$
\Sigma = \begin{pmatrix}
\sigma^2 & 0 & 0 & \ldots & 0 & 0 \\
0 & \sigma^2 & 0 & \ldots & 0 & 0 \\
0 & 0 & \sigma^2 & \ldots & 0 & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & 0 & \ldots & \sigma^2 & 0 \\
0 & 0 & 0 & \ldots & 0 & \sigma^2 
\end{pmatrix}
$$

All features have the same variance.
Spherical: $O(1)$ cov parameters

$$
\Sigma = \begin{pmatrix}
\sigma^2 & 0 & 0 & \ldots & 0 & 0 \\
0 & \sigma^2 & 0 & \ldots & 0 & 0 \\
0 & 0 & \sigma^2 & \ldots & 0 & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & 0 & \ldots & \sigma^2 & 0 \\
0 & 0 & 0 & \ldots & 0 & \sigma^2
\end{pmatrix}
$$

Even simpler
Next… back to **Density Estimation**

What if we want to do density estimation with multimodal or clumpy data?

\[ X \sim \sum_{i=1}^{K} w_i \cdot N(\mu_i, \Sigma_i) \]

Yet don’t know point has \( x, y = i \)

\[ X \sim N(\mu_i, \Sigma_i) \]

One Gaussian for each bump.
But we don’t see class labels!!!

- **MLE:**
  \[
  \arg \max \prod_j P(y_j, x_j)
  \]

  \[
  = \arg \max \log \prod_j P(y_j, x_j) = \arg \max \sum_j \log P(y_j, x_j)
  \]

  *almost always nice!*

- But we don’t know \(y_j\)'s!!!

- Maximize marginal likelihood:
  \[
  \arg \max \prod_j P(x_j) = \arg \max \prod_j \sum_{i=1}^k P(y_j=i, x_j)
  \]

  \[
  = \arg \max \sum_j \log \sum_{i=1}^k P(y_j=i, x_j)
  \]

  *log sum is almost never nice!!*
Special case: spherical Gaussians and hard assignments

\[ P(x_j \mid y = i) = \frac{1}{(2\pi)^{m/2} \left\| \Sigma_i \right\|^{1/2}} \exp \left[ -\frac{1}{2} (x_j - \mu_i)^T \Sigma_i^{-1} (x_j - \mu_i) \right] \sim N(\mu_i, \Sigma_i) \]

- If \( P(X\mid Y=i) \) is spherical, with same \( \sigma \) for all classes:
  \[ P(x_j \mid y = i) \propto \exp \left[ -\frac{1}{2\sigma^2} \|x_j - \mu_i\|^2 \right] \]

- If each \( x_j \) belongs to one class \( C(j) \) (hard assignment), marginal likelihood:
  \[
  \log \prod_{j=1}^{m} \sum_{i=1}^{k} P(x_j, y = i) \propto \prod_{j=1}^{m} \exp \left[ -\frac{1}{2\sigma^2} \|x_j - \mu_{C(j)}\|^2 \right]
  \]
  \[
  = \sum_j \|x_j - \mu_{C(j)}\|^2 + O(1)
  \]

- Same as K-means!!!
The GMM assumption

- There are k components
- Component $i$ has an associated mean vector $\mu_i$
The GMM assumption

- There are k components
- Component $i$ has an associated mean vector $\mu_i$
- Each component generates data from a Gaussian with mean $\mu_i$ and covariance matrix $\sigma^2 I$ spherical

Each data point is generated according to the following recipe:
The GMM assumption

- There are $k$ components
- Component $i$ has an associated mean vector $\mu_i$
- Each component generates data from a Gaussian with mean $\mu_i$ and covariance matrix $\sigma^2 I$

Each data point is generated according to the following recipe:

1. Pick a component at random:
   Choose component $i$ with probability $P(y=i)$
The GMM assumption

- There are $k$ components
- Component $i$ has an associated mean vector $\mu_i$
- Each component generates data from a Gaussian with mean $\mu_i$ and covariance matrix $\sigma^2 I$

Each data point is generated according to the following recipe:

1. Pick a component at random: Choose component $i$ with probability $P(y=i)$
2. Datapoint $\sim N(\mu, \sigma^2 I)$
The General GMM assumption

- There are $k$ components
- Component $i$ has an associated mean vector $\mu_i$
- Each component generates data from a Gaussian with mean $\mu_i$ and covariance matrix $\Sigma_i$

Each data point is generated according to the following recipe:

1. Pick a component at random: Choose component $i$ with probability $P(y = i)$
2. Datapoint $\sim N(\mu, \Sigma)$
Unsupervised Learning: not as hard as it looks

Sometimes easy

Sometimes impossible

and sometimes in between

IN CASE YOU'RE WONDERING WHAT THESE DIAGRAMS ARE, THEY SHOW 2-d UNLABELED DATA (X VECTORS) DISTRIBUTED IN 2-d SPACE. THE TOP ONE HAS THREE VERY CLEAR GAUSSIAN CENTERS
Marginal likelihood for general case

\[ P(x_j \mid y = i) = \frac{1}{(2\pi)^{m/2} \| \Sigma_i \|^{1/2}} \exp \left[ -\frac{1}{2} (x_j - \mu_i)^T \Sigma_i^{-1} (x_j - \mu_i) \right] \]

- Marginal likelihood:
  \[
  \prod_{j=1}^{m} P(x_j) = \prod_{j=1}^{m} \sum_{i=1}^{k} P(x_j, y = i) \\
  = \prod_{j=1}^{m} \sum_{i=1}^{k} \frac{1}{(2\pi)^{m/2} \| \Sigma_i \|^{1/2}} \exp \left[ -\frac{1}{2} (x_j - \mu_i)^T \Sigma_i^{-1} (x_j - \mu_i) \right] P(y = i)
  \]
Special case 2: spherical Gaussians and soft assignments

- If \( P(X|Y=i) \) is spherical, with same \( \sigma \) for all classes:
  \[
P(x_j \mid y = i) \propto \exp \left[ -\frac{1}{2\sigma^2} \left\| x_j - \mu_i \right\|^2 \right]
  \]

- Uncertain about class of each \( x_j \) (soft assignment), marginal likelihood:
  \[
  \log \left[ \prod_{j=1}^{m} \sum_{i=1}^{k} P(x_j, y = i) \right] \propto \sum_{j=1}^{m} \log \sum_{i=1}^{k} \exp \left[ -\frac{1}{2\sigma^2} \left\| x_j - \mu_i \right\|^2 \right] P(y = i)
  \]
  \[
  \log \sum_{i=1}^{k} \exp \left[ \frac{-1}{2\sigma^2} \left\| x_j - \mu_i \right\|^2 \right] P(y = i)
  \]
Unsupervised Learning: Mediumly Good News

We now have a procedure s.t. if you give me a guess at $\mu_1, \mu_2, \ldots, \mu_k$ I can tell you the prob of the unlabeled data given those $\mu$'s.

Suppose $x$'s are 1-dimensional.

There are two classes; $w_1$ and $w_2$

$P(y_1) = 1/3$ $P(y_2) = 2/3$ $\sigma = 1$.

There are 25 unlabeled datapoints

$x_1 = 0.608$
$x_2 = -1.590$
$x_3 = 0.235$
$x_4 = 3.949$
$\vdots$
$x_{25} = -0.712$

(From Duda and Hart)
Duda & Hart’s Example

We can graph the prob. dist. function of data given our \( \mu_1 \) and \( \mu_2 \) estimates.

We can also graph the true function from which the data was randomly generated.

- They are close. Good.
- The 2\textsuperscript{nd} solution tries to put the “2/3” hump where the “1/3” hump should go, and vice versa.
- In this example unsupervised is almost as good as supervised. If the \( x_1 \ldots x_{25} \) are given the class which was used to learn them, then the results are \((\mu_1=-2.176, \mu_2=1.684)\). Unsupervised got \((\mu_1=-2.13, \mu_2=1.668)\).
Duda & Hart’s Example

Graph of \( \log P(x_1, x_2 \ldots x_{25} | \mu_1, \mu_2) \) against \( \mu_1 (\rightarrow) \) and \( \mu_2 (\uparrow) \)

Max likelihood = \( (\mu_1 = -2.13, \mu_2 = 1.668) \)

Local minimum, but very close to global at \( (\mu_1 = 2.085, \mu_2 = -1.257) \)*

* corresponds to switching \( y_1 \) with \( y_2 \)
Finding the max likelihood $\mu_1, \mu_2.. \mu_k$

We can compute $P(\text{data} | \mu_1, \mu_2.. \mu_k)$

How do we find the $\mu_i$'s which give max. likelihood?

- The normal max likelihood trick:
  
  Set $\frac{\partial}{\partial \mu_i} \log \text{Prob (....)} = 0$

  and solve for $\mu_i$'s.

  # Here you get non-linear non-analytically-solvable equations

- Use gradient descent
  
  Slow but doable

- Use a much faster, cuter, and recently very popular method…
Expectation Maximalization
We’ll get back to unsupervised learning soon.

But now we’ll look at an even simpler case with hidden information.

The EM algorithm

- Can do trivial things, such as the contents of the next few slides.
- An excellent way of doing our unsupervised learning problem, as we’ll see.
- Many, many other uses, including inference of Hidden Markov Models (future lecture).
Silly Example

Let events be “grades in a class”

\[ w_1 = \text{Gets an A} \quad P(A) = \frac{1}{2} \]
\[ w_2 = \text{Gets a B} \quad P(B) = \mu \]
\[ w_3 = \text{Gets a C} \quad P(C) = 2\mu \]
\[ w_4 = \text{Gets a D} \quad P(D) = \frac{1}{2} - 3\mu \]

(Note \( 0 \leq \mu \leq \frac{1}{6} \))

Assume we want to estimate \( \mu \) from data. In a given class there were

\[ a \text{ A's} \]
\[ b \text{ B's} \]
\[ c \text{ C's} \]
\[ d \text{ D's} \]

What’s the maximum likelihood estimate of \( \mu \) given \( a, b, c, d \)?

\[
\hat{\mu} = \arg\max_{\mu} P(a, b, c, d | \mu)
\]

\[
= \arg\max_{\mu} \left( \frac{1}{2} \right)^a \cdot (\mu)^b \cdot (2\mu)^c \cdot \left( \frac{1}{2} - 3\mu \right)^d
\]
Trivial Statistics

\[ P(A) = \frac{1}{2} \quad P(B) = \mu \quad P(C) = 2\mu \quad P(D) = \frac{1}{2} - 3\mu \]

\[ P(a,b,c,d \mid \mu) = K(\frac{1}{2})^a(\mu)^b(2\mu)^c(\frac{1}{2} - 3\mu)^d \]

\[ \log P(a,b,c,d \mid \mu) = \log K + a\log \frac{1}{2} + b\log \mu + c\log 2\mu + d\log (\frac{1}{2} - 3\mu) \]

FOR MAX LIKE \( \mu \), SET \( \frac{\partial \log P}{\partial \mu} = 0 \)

\[ \frac{\partial \log P}{\partial \mu} = \frac{b}{\mu} + \frac{2c}{2\mu} - \frac{3d}{1 - 2\mu} = 0 \]

Gives max like \( \mu = \frac{b + c}{6(b + c + d)} \)

So if class got

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>14</td>
<td>6</td>
<td>9</td>
<td>10</td>
</tr>
</tbody>
</table>

Max like \( \mu = \frac{1}{10} \)

Boring, but true!
Same Problem with Hidden Information

Someone tells us that

Number of High grades (A’s + B’s) = \( h \)
Number of C’s = \( c \)
Number of D’s = \( d \)

What is the max. like estimate of \( \mu \) now?

\[
\arg \max_{\mu} \sum_{a,b : a+b=h} p(a,b,c,d \mid \mu)
\]
Same Problem with Hidden Information

Someone tells us that
- Number of High grades (A’s + B’s) = \( h \)
- Number of C’s = \( c \)
- Number of D’s = \( d \)

What is the max. like estimate of \( \mu \) now?

We can answer this question circularly:

**EXPECTATION**

If we know the value of \( \mu \) we could compute the expected value of \( a \) and \( b \)

Since the ratio \( a:b \) should be the same as the ratio \( \frac{1}{2} : \mu \)

\[
a = \frac{1}{2} \cdot \frac{1}{\frac{1}{2} + \mu} h
\]

\[
b = \frac{\mu}{1/2 + \mu} h
\]

**MAXIMIZATION**

If we know the expected values of \( a \) and \( b \) we could compute the maximum likelihood value of \( \mu \)

\[
\mu = \frac{b+c}{6(b+c+d)}
\]
E.M. for our Trivial Problem

We begin with a guess for $\mu$
We iterate between EXPECTATION and MAXIMALIZATION to improve our estimates of $\mu$ and $a$ and $b$.

Define $\mu^{(t)}$ the estimate of $\mu$ on the $t$'th iteration

$$b^{(t)} = \frac{\mu^{(t)}h}{1/2 + \mu^{(t)}} = \mathbb{E}[b | \mu^{(t)}]$$

$$\mu^{(t+1)} = \frac{b^{(t)} + c}{6(b^{(t)} + c + d)}$$

$$= \text{max like est. of } \mu \text{ given } b^{(t)}$$

Continue iterating until converged.
Good news: Converging to local optimum is assured.
Bad news: I said “local” optimum.

REMEMBER

$P(A) = 1/2$
$P(B) = \mu$
$P(C) = 2\mu$
$P(D) = 1/2 - 3\mu$
E.M. Convergence

Convergence proof based on fact that Prob(data | µ) must increase or remain same between each iteration [NOT OBVIOUS]

But it can never exceed 1 [OBIOSUS]
So it must therefore converge [OBIOSUS]

In our example, suppose we had

h = 20

\( c = 10 \)

\( d = 10 \)

\( \mu^{(0)} = 0 \)

Convergence is generally linear: error decreases by a constant factor each time step.
Remember:

- We have unlabeled data $x_1, x_2, \ldots, x_m$
- We know there are $k$ classes
- We know $P(y_1), P(y_2), P(y_3), \ldots, P(y_k)$
- We don’t know $\mu_1, \mu_2, \ldots, \mu_k$

We can write $P(\text{data} | \mu_1, \ldots, \mu_k)$

$$= p(x_1, \ldots, x_m | \mu_1, \ldots, \mu_k)$$

$$= \prod_{j=1}^{m} p(x_j | \mu_1, \ldots, \mu_k)$$

$$= \prod_{j=1}^{m} \sum_{i=1}^{k} p(x_j | \mu_i) P(y = i)$$

$$\propto \prod_{j=1}^{m} \sum_{i=1}^{k} \exp \left( -\frac{1}{2\sigma^2} \|x_j - \mu_i\|^{2} \right) P(y = i)$$
EM for simple case of GMMs: The E-step

- If we know $\mu_1, \ldots, \mu_k$ → easily compute prob. point $x_j$ belongs to class $y=i$

$$p(y = i|x_j, \mu_1, \ldots, \mu_k) \propto \exp\left(-\frac{1}{2\sigma^2}\|x_j - \mu_i\|^2\right)P(y = i)$$
EM for simple case of GMMs: The M-step

- If we know prob. point $x_j$ belongs to class $y=i$
  \[ \rightarrow \text{MLE for } \mu_i \text{ is weighted average} \]
- Imagine $k$ copies of each $x_j$, each with weight $P(y=i|x_j)$:

\[
\mu_i = \frac{\sum_{j=1}^{m} P(y=i|x_j) x_j}{\sum_{j=1}^{m} P(y=i|x_j)}
\]
E.M. for GMMs

**E-step**

Compute “expected” classes of all datapoints for each class

\[ p(y = i | x_j, \mu_1...\mu_k) \propto \exp\left( -\frac{1}{2\sigma^2} \| x_j - \mu_i \|^2 \right) P(y = i) \]

**M-step**

Compute Max. like \( \mu \) given our data’s class membership distributions

\[ \mu_i = \frac{\sum_{j=1}^{m} P(y = i | x_j) x_j}{\sum_{j=1}^{m} P(y = i | x_j)} \]
E.M. Convergence

- EM is coordinate ascent on an interesting potential function
- Coord. ascent for bounded pot. func. → convergence to a local optimum guaranteed
- See Neal & Hinton reading on class webpage

- This algorithm is REALLY USED. And in high dimensional state spaces, too. E.G. Vector Quantization for Speech Data
E.M. for General GMMs

Iterate. On the \( t \)'th iteration let our estimates be

\[
\lambda_t = \{ \mu_1^{(t)}, \mu_2^{(t)} \ldots \mu_k^{(t)}, \Sigma_1^{(t)}, \Sigma_2^{(t)} \ldots \Sigma_k^{(t)}, p_1^{(t)}, p_2^{(t)} \ldots p_k^{(t)} \}
\]

**E-step**

Compute “expected” classes of all datapoints for each class

\[
P(y = i|x_j, \lambda_t) \propto p_i^{(t)} p\left(x_j | \mu_i^{(t)}, \Sigma_i^{(t)}\right)
\]

**M-step**

Compute Max. like \( \mu \) given our data’s class membership distributions

\[
\mu_i^{(t+1)} = \frac{\sum_j P(y = i|x_j, \lambda_t) x_j}{\sum_j P(y = i|x_j, \lambda_t)}
\]

\[
\Sigma_i^{(t+1)} = \frac{\sum_j P(y = i|x_j, \lambda_t) [x_j - \mu_i^{(t+1)}] [x_j - \mu_i^{(t+1)}]^T}{\sum_j P(y = i|x_j, \lambda_t)}
\]

\[
p_i^{(t+1)} = \frac{\sum_j P(y = i|x_j, \lambda_t)}{m}
\]

\( p_i^{(t)} \) is shorthand for estimate of \( P(y=i) \) on t’th iteration.

Just evaluate a Gaussian at \( x_j \).
Gaussian Mixture Example: Start
After first iteration
After 2nd iteration
After 3rd iteration
After 4th iteration
After 5th iteration

\[ p = 0.322 \]

\[ p = 0.285 \]
After 6th iteration
After 20th iteration
Some Bio Assay data
GMM clustering of the assay data
Resulting Density Estimator
Three classes of assay
(each learned with its own mixture model)
Resulting Bayes Classifier
Resulting Bayes Classifier, using posterior probabilities to alert about ambiguity and anomalousness.

Yellow means ANOMALOUS
Cyan means AMBIGUOUS
What you should know

- K-means for clustering:
  - algorithm
  - converges because it’s coordinate ascent

- EM for mixture of Gaussians:
  - How to “learn” maximum likelihood parameters (locally max. like.) in the case of unlabeled data

- Be happy with this kind of probabilistic analysis

- Understand the two examples of E.M. given in these notes

- Remember, E.M. can get stuck in local minima, and empirically it DOES
Acknowledgements

- K-means & Gaussian mixture models presentation contains material from excellent tutorial by Andrew Moore:
  - http://www.autonlab.org/tutorials/

- K-means Applet:
  - http://www.elet.polimi.it/upload/matteucc/Clustering/tutorial_html/AppletKM.html

- Gaussian mixture models Applet:
  - http://www.neurosci.aist.go.jp/%7Eakaho/MixtureEM.html