Machine learning: lecture 14

Tommi S. Jaakkola
MIT CSAIL
tommi@csail.mit.edu

Topics

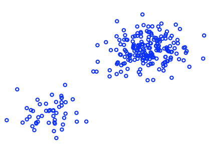
- Gaussian mixtures and the EM-algorithm
  - complete, incomplete, and inferred data
  - EM for mixtures
  - demo
  - EM and convergence
  - regularized mixtures
  - selecting the number of mixture components
  - Gaussian mixtures for classification

Review: mixture densities

- A Gaussian mixture model with m components is defined as

$$p(\mathbf{x}|\theta) = \sum_{j=1}^{m} p_j \, p(\mathbf{x}|\mu_j, \Sigma_j)$$

where $\theta = \{p_1, \ldots, p_m, \mu_1, \ldots, \mu_m, \Sigma_1, \ldots, \Sigma_m\}$ contains all the parameters of the mixture model.



- We have to estimate these models from *incomplete data* involving only $\mathbf{x}$ samples; the assignment to components has to be *inferred*

Types of data: complete

$$p(\mathbf{x}|\theta) = \sum_{j=1}^{m} p_j \, p(\mathbf{x}|\mu_j, \Sigma_j)$$

- When the available data is *complete* each sample contains the setting of all the variables in the model.

$\mathbf{x}$	y			
$\mathbf{x}_1$	0	1	$\ldots$	0
$\mathbf{x}_2$	0	0	$\ldots$	1
$\ldots$			$\ldots$	
$\mathbf{x}_n$	0	1	$\ldots$	0

The parameter estimation problem is in this case straightforward (each component Gaussian can be estimated separately)

Types of data: incomplete

$$p(\mathbf{x}|\theta) = \sum_{j=1}^{m} p_j \, p(\mathbf{x}|\mu_j, \Sigma_j)$$

- *Incomplete* data for a mixture model typically contain only $\mathbf{x}$ samples.

$\mathbf{x}$	y
$\mathbf{x}_1$	
$\mathbf{x}_2$	
$\ldots$	
$\mathbf{x}_n$	

To estimate the parameters we have to infer which component Gaussian was responsible for generating each sample $\mathbf{x}_i$

Types of data: inferred

$$p(\mathbf{x}|\theta) = \sum_{j=1}^{m} p_j \, p(\mathbf{x}|\mu_j, \Sigma_j)$$

- We can infer the values for the missing data based on the current setting of the parameters

$\mathbf{x}$	y			
$\mathbf{x}_1$	$P(y=1\|\mathbf{x}_1,\theta)$	$P(y=2\|\mathbf{x}_1,\theta)$	$\ldots$	$P(y=m\|\mathbf{x}_1,\theta)$
$\mathbf{x}_2$	$P(y=1\|\mathbf{x}_2,\theta)$	$P(y=2\|\mathbf{x}_2,\theta)$	$\ldots$	$P(y=m\|\mathbf{x}_2,\theta)$
$\ldots$			$\ldots$	
$\mathbf{x}_n$	$P(y=1\|\mathbf{x}_n,\theta)$	$P(y=2\|\mathbf{x}_n,\theta)$	$\ldots$	$P(y=m\|\mathbf{x}_n,\theta)$

The parameter estimation problem is again easy if we treat the inferred data as complete data. The solution has to be iterative, however.

The EM-algorithm

Step 0: specify the initial setting of the parameters $\theta = \theta^{(0)}$

$$p(\mathbf{x}|\theta) = \sum_{j=1}^{m} p_j \, p(\mathbf{x}|\mu_j, \Sigma_j)$$

For example, we could

– set each μ_j to $\mathbf{x}$ sampled at random from the training set
– set each Σ_j to be the sample covariance of the whole data
– set mixing proportions p_j to be uniform $p_j = 1/m$.

The EM-algorithm

Step 0: specify the initial setting of the parameters $\theta = \theta^{(0)}$

E-step: complete the incomplete data with the posterior probabilities

$$P(y = j|\mathbf{x}_i, \theta^{(k)}), \;\; j = 1, \ldots, m, \;\; i = 1, \ldots, n$$

The EM-algorithm

Step 0: specify the initial setting of the parameters $\theta = \theta^{(0)}$

E-step: complete the incomplete data with the posterior probabilities

$$P(y = j|\mathbf{x}_i, \theta^{(k)}), \;\; j = 1, \ldots, m, \;\; i = 1, \ldots, n$$

M-step: find the new setting of the parameters $\theta^{(k+1)}$ by maximizing the log-likelihood of the completed (inferred) data

$$\theta^{(k+1)} = \arg\max_{\theta} \sum_{i=1}^{n} \sum_{j=1}^{m} P(y = j|\mathbf{x}_i, \theta^{(k)}) \log \overbrace{\left[p_j \, p(\mathbf{x}_i|\mu_j, \Sigma_j) \right]}^{P(\mathbf{x}_i, y=j|\theta)}$$

Demo

Topics

- Gaussian mixtures and the EM-algorithm
  – complete, incomplete, and inferred data
  – EM for mixtures
  – demo
  – EM and convergence
  – regularized mixtures
  – selecting the number of mixture components
  – Gaussian mixtures for classification

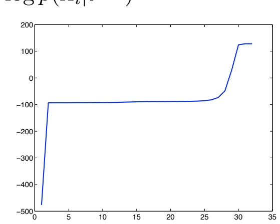
EM-algorithm: convergence

$$p(\mathbf{x}|\theta) = \sum_{j=1}^{m} p_j \, p(\mathbf{x}|\mu_j, \Sigma_j)$$

- The EM-algorithm monotonically increases the log-likelihood of the training data. In other words,

  $l(\theta^{(0)}) < l(\theta^{(1)}) < l(\theta^{(2)}) < \ldots$ until convergence

  $l(\theta^{(k)}) = \sum_{i=1}^{n} \log p(\mathbf{x}_i|\theta^{(k)})$



EM-algorithm: auxiliary objective

- We first introduce possible posterior assignments $\{Q(j|i)\}$ and the corresponding auxiliary likelihood objective:

$$
\begin{aligned}
l(\theta^{(k)}) &= \sum_{i=1}^{n} \log p(\mathbf{x}_i|\theta^{(k)}) \\
&= \sum_{i=1}^{n} \log \sum_{j=1}^{m} p_j^{(k)} p(\mathbf{x}_i|\mu_j^{(k)}, \Sigma_j^{(k)}) \\
&= \sum_{i=1}^{n} \log \sum_{j=1}^{m} Q(j|i) \frac{p_j^{(k)} p(\mathbf{x}_i|\mu_j^{(k)}, \Sigma_j^{(k)})}{Q(j|i)} \\
&\geq \sum_{i=1}^{n} \sum_{j=1}^{m} Q(j|i) \log \frac{p_j^{(k)} p(\mathbf{x}_i|\mu_j^{(k)}, \Sigma_j^{(k)})}{Q(j|i)} \\
&= l(Q; \theta^{(k)})
\end{aligned}
$$

EM-algorithm: auxiliary objective

- The auxiliary objective

$$
l(Q;\theta^{(k)}) = \sum_{i=1}^{n} \sum_{j=1}^{m} Q(j|i) \log \frac{p_j^{(k)} p(\mathbf{x}_i|\mu_j^{(k)}, \Sigma_j^{(k)})}{Q(j|i)} \leq l(\theta^{(k)})
$$

recovers the log-likelihood of the data at the correct posterior assignments. In other words,

$$
\max_Q l(Q;\theta^{(k)}) = l(Q^{(k)};\theta^{(k)}) = l(\theta^{(k)})
$$

where $Q^{(k)}(j|i) = P(y = j|\mathbf{x}_i, \theta^{(k)})$ are the posterior assignments corresponding to parameters $\theta^{(k)}$.

EM-algorithm: max-max and monotonicity

- We can now rewrite the EM-algorithm in terms of two maximization steps involving the auxiliary objective:

E-step: $Q^{(k)} = \operatorname{argmax}_Q l(Q;\theta^{(k)})$
M-step: $\theta^{(k+1)} = \operatorname{argmax}_\theta l(Q^{(k)};\theta)$

The monotonic increase of the log-likelihood now follows from the facts that 1) the auxiliary objective is monotonically increasing, and 2) it equals the log-likelihood after each E-step

$$
\begin{aligned}
l(\theta^{(k)}) &= l(Q^{(k)};\theta^{(k)}) \\
&\leq l(Q^{(k)};\theta^{(k+1)}) \\
&\leq l(Q^{(k+1)};\theta^{(k+1)}) = l(\theta^{(k+1)})
\end{aligned}
$$

Topics

- Gaussian mixtures and the EM-algorithm
  - complete, incomplete, and inferred data
  - EM for mixtures
  - demo
  - EM and convergence
  - regularized mixtures
  - selecting the number of mixture components
  - Gaussian mixtures for classification

Regularized EM

- Even a single covariance matrix in the Gaussian mixture model involves a number of parameters and can easily lead to over-fitting.

$$
p(\mathbf{x}|\theta) = \sum_{j=1}^{m} p_j \, p(\mathbf{x}|\mu_j, \Sigma_j)
$$

- We can regularize the model by assigning a prior distribution over the parameters, especially the covariance matrices

Regularized EM: prior

- A Wishart prior over each covariance matrix is given by

$$
P(\Sigma|S,n') \propto \frac{1}{|\Sigma|^{n'/2}} \exp\left(-\frac{n'}{2} \text{Trace}(\Sigma^{-1} S) \right)
$$

(written here in a bit non-standard way)

$$
\begin{aligned}
S &= \text{"prior" covariance matrix} \\
n' &= \text{equivalent sample size}
\end{aligned}
$$

The equivalent sample size represents the number of training samples we would have to see in order for the prior and the data to have equal effect on the solution

Regularized EM

- The E-step is unaffected (though the resulting values for the soft assignments will change)
- In the M-step we now maximize a penalized log-likelihood of the weighted training set:

$$\sum_{i=1}^{n}\sum_{j=1}^{m}\overbrace{P(y=j|\mathbf{x}_i,\theta^{(k)})}^{\hat{p}(j|i)}\log\left[p_j p(\mathbf{x}_i|\mu_j,\Sigma_j)\right]+\sum_{j=1}^{m}\log P(\Sigma_j|S,n')$$

Formally the regularization penalty changes the resulting covariance estimates only slightly:

$$\Sigma_j^{(k+1)} \leftarrow \frac{1}{\hat{n}_j+n'}\left[\sum_{i=1}^{n}\hat{p}(j|i)\,(\mathbf{x}_i-\hat{\mu}_j)(\mathbf{x}_i-\hat{\mu}_j)^T+n'S\right]$$

Regularized EM: demo

Topics

- Gaussian mixtures and the EM-algorithm
  - complete, incomplete, and inferred data
  - EM for mixtures
  - demo
  - EM and convergence
  - regularized mixtures
  - selecting the number of mixture components
  - Gaussian mixtures for classification

Model selection and mixtures

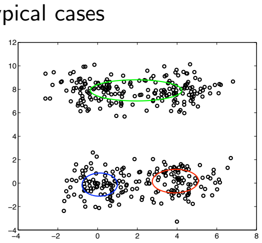
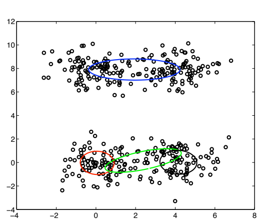
- As a simple strategy for selecting the appropriate number of mixture components, we can find m that minimizes the overall description length (cf. BIC):

$$\text{DL} \approx -\log p(\text{data}|\hat{\theta}_m)+\frac{d_m}{2}\log(n)$$

  - n is the number of training points,
  - $\hat{\theta}_m$ are the maximum likelihood parameters for the m-component mixture, and
  - d_m is the (effective) number of parameters in the m-component mixture.

Model selection: example

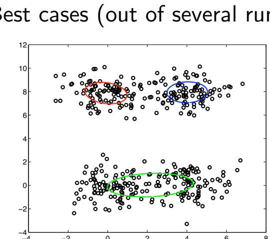
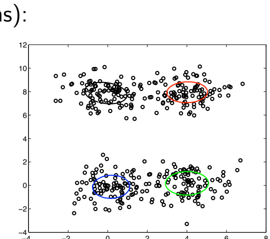
- Typical cases

```
m=1, -logP(data)=2017.38, penalty=14.98, DL=2032.36
m=2, -logP(data)=1712.69, penalty=32.95, DL=1745.65
m=3, -logP(data)=1711.40, penalty=50.93, DL=1762.32
m=4, -logP(data)=1682.06, penalty=68.90, DL=1750.97
```


Model selection: example

- Best cases (out of several runs):

```
m=1, -logP(data)=2017.38, penalty=14.98, DL=2032.36
m=2, -logP(data)=1712.69, penalty=32.95, DL=1745.65
m=3, -logP(data)=1678.56, penalty=50.93, DL=1729.49
m=4, -logP(data)=1649.08, penalty=68.90, DL=1717.98
```

Topics

- Gaussian mixtures and the EM-algorithm
  - complete, incomplete, and inferred data
  - EM for mixtures
  - demo
  - EM and convergence
  - regularized mixtures
  - selecting the number of mixture components
  - Gaussian mixtures for classification

Classification example

- A digit recognition problem (8x8 binary digits)
  Training set $n = 100$ (50 examples of each digit).
  Test set $n = 400$ (200 examples of each digit).
- We'd like to estimate class conditional mixture models (and prior class frequencies) to solve the classification problem

Classification example

- A digit recognition problem (8x8 binary digits)
  Training set $n = 100$ (50 examples of each digit).
  Test set $n = 400$ (200 examples of each digit).
- We'd like to estimate class conditional mixture models (and prior class frequencies) to solve the classification problem

  For example:

  Class 1: $P(y=1),\ p(\mathbf{x}|\theta_1),$ (e.g., a 3-component mixture)

  Class 0: $P(y=0),\ p(\mathbf{x}|\theta_0),$ (e.g., a 3-component mixture)

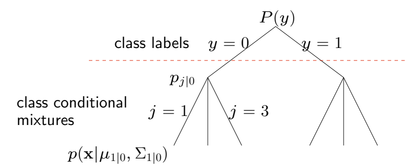
  A new test example $\mathbf{x}$ would be classified according to

  $$\text{Class} = 1 \text{ if } \quad \log \frac{\hat{P}(y=1)p(\mathbf{x}|\hat{\theta}_1)}{\hat{P}(y=0)p(\mathbf{x}|\hat{\theta}_0)} > 0$$

  and Class $= 0$ otherwise.

Classification example

- A digit recognition problem (8x8 binary digits)
  Training set $n = 100$ (50 examples of each digit).
  Test set $n = 400$ (200 examples of each digit).
- We'd like to estimate class conditional mixture models (and prior class frequencies) to solve the classification problem



$$p(\mathbf{x}|\theta_0) = \sum_{j=1}^{3} p_{j|0}\, p(\mathbf{x}|\mu_{j|0}, \Sigma_{j|0})$$

(a hierarchical mixture model)

Classification example

- A digit recognition problem (8x8 binary digits)
  Training set $n = 100$ (50 examples of each digit).
  Test set $n = 400$ (200 examples of each digit).
- The figure gives the number of missclassified examples on the test set as a function of the number of mixture components in each class-conditional model