EM (cont.)

Machine Learning – 10701/15781

Carlos Guestrin
Carnegie Mellon University

November 26th, 2007

©2005-2007 Carlos Guestrin

Silly Example

Let events be "grades in a class"

w_1 = Gets an A	P(A) = ½
w_2 = Gets a B	P(B) = μ
w_3 = Gets a C	P(C) = 2μ
w_4 = Gets a D	P(D) = ½-3μ

(Note $0 \le \mu \le 1/6$)

Assume we want to estimate μ from data. In a given class there were

a A's
b B's
c C's
d D's

What's the maximum likelihood estimate of μ given a,b,c,d ?

$\hat{\mu}_{MLE}$

©2005-2007 Carlos Guestrin

Trivial Statistics

P(A) = ½ P(B) = μ P(C) = 2μ P(D) = ½-3μ

P(*a,b,c,d* | μ) = K(½)^a(μ)^b(2μ)^c(½-3μ)^d

log P(*a,b,c,d* | μ) = log K + *a*log ½ + *b*log μ + *c*log 2μ + *d*log (½-3μ)

FOR MAX LIKE μ, SET $\frac{\partial \text{LogP}}{\partial \mu} = 0$

$$\frac{\partial \text{LogP}}{\partial \mu} = \frac{b}{\mu} + \frac{2c}{2\mu} - \frac{3d}{1/2 - 3\mu} = 0$$

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

So if class got

A	B	C	D
14	6	9	10

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 μ now?

We can answer this question circularly:

REMEMBER
P(A) = ½
P(B) = μ
P(C) = 2μ
P(D) = ½-3μ

EXPECTATION

If we know the value of μ we could compute the expected value of *a* and *b*

Iterate

Since the ratio a:b should be the same as the ratio ½ : μ

$$\tilde{a} = \frac{1/2}{1/2 + \mu} h \qquad \tilde{b} = \frac{\mu}{1/2 + \mu} h$$

MAXIMIZATION

If we know the expected values of $\tilde{a}$ and $\tilde{b}$ we could compute the maximum likelihood value of μ

$$\mu = \frac{\tilde{b} + c}{6(\tilde{b} + c + d)}$$

E.M. for our Trivial Problem

REMEMBER
P(A) = ½
P(B) = μ
P(C) = 2μ
P(D) = ½-3μ

We begin with a guess for μ
We iterate between EXPECTATION and MAXIMALIZATION to improve our estimates of μ and *a* and *b*.

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

if trueparam was $\mu^{(t)}$

$$\mu^{(0)} = \text{initial guess}$$

$$b^{(t)} = \frac{\mu^{(t)} h}{1/2 + \mu^{(t)}} = \mathrm{E}\left[b \mid \mu^{(t)}\right]$$

E-step

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

M-step

$b =$

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

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

©2005-2007 Carlos Guestrin

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 [OBVIOUS]

So it must therefore converge [OBVIOUS]

In our example, suppose we had
h = 20
c = 10
d = 10
$\mu^{(0)} = 0$

t	$\mu^{(t)}$	$b^{(t)}$
0	0	0
1	0.0833	2.857
2	0.0937	3.158
3	0.0947	3.185
4	0.0948	3.187
5	0.0948	3.187
6	0.0948	3.187

$a^{(t)}$
20

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

©2005-2007 Carlos Guestrin

Back to Unsupervised Learning of GMMs – a simple case

A simple case:

We have unlabeled data $\boldsymbol{x}_1\ \boldsymbol{x}_2 \ldots \boldsymbol{x}_m$

We know there are k classes

We know $P(y_1)\ P(y_2)\ P(y_3) \ldots P(y_k)$

We <u>don't</u> know $\boldsymbol{\mu}_1\ \boldsymbol{\mu}_2 \ldots \boldsymbol{\mu}_k$

We can write P(data | $\boldsymbol{\mu}_1 \ldots \boldsymbol{\mu}_k$)

$$= \mathrm{p}\left(x_1 \ldots x_m \middle| \mu_1 \ldots \mu_k\right)$$

$$= \prod_{j=1}^{m} \mathrm{p}\left(x_j \middle| \mu_1 \ldots \mu_k\right)$$

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

$$\propto \prod_{j=1}^{m} \sum_{i=1}^{k} \exp\left(-\frac{1}{2\sigma^2}\left\|x_j - \mu_i\right\|^2\right) \mathrm{P}(y=i)$$

©2005-2007 Carlos Guestrin

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

$$\mathrm{p}\left(y=i \middle| x_j, \mu_1 \ldots \mu_k\right) \propto \exp\left(-\frac{1}{2\sigma^2}\left\|x_j - \mu_i\right\|^2\right) \mathrm{P}(y=i)$$

©2005-2007 Carlos Guestrin

EM for simple case of GMMs: The M-step

- If we know prob. point x_j belongs to class y=i

  → MLE for μ_i is weighted average

  - imagine k copies of each x_j, each with weight $P(y=i|x_j)$:

$$\mu_i = \frac{\sum_{j=1}^{m} P(y=i|x_j)\, x_j}{\sum_{j=1}^{m} P(y=i|x_j)}$$

©2005-2007 Carlos Guestrin

E.M. for GMMs

E-step

Compute "expected" classes of all datapoints for each class

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

Just evaluate a Gaussian at x_j

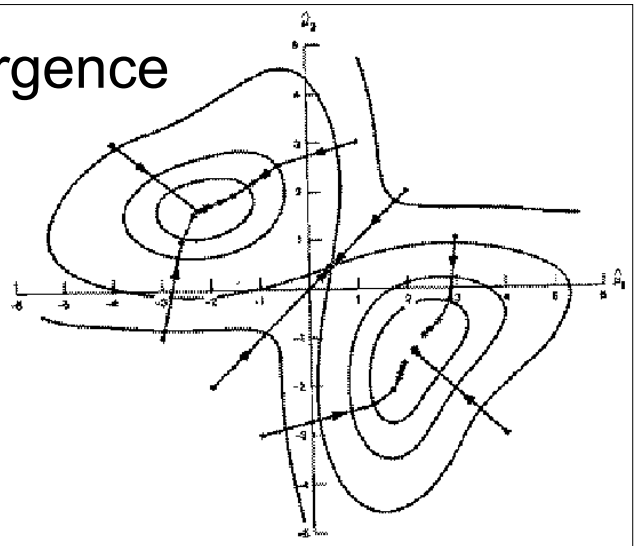
M-step

Compute Max. like **μ** given our data's class membership distributions

$$\mu_i = \frac{\sum_{j=1}^{m} P(y=i|x_j)\, x_j}{\sum_{j=1}^{m} P(y=i|x_j)}$$

©2005-2007 Carlos Guestrin

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

©2005-2007 Carlos Guestrin

E.M. for axis-aligned GMM

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

Iterate. On the *t*'th iteration let our estimates be

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

$p_i^{(t)}$ is shorthand for estimate of $P(y=i)$ on t'th iteration

E-step

Compute "expected" classes of all datapoints for each class

$$P(y = i \mid x_j, \lambda_t) \propto p_i^{(t)} p(x_j \mid \mu_i^{(t)}, \Sigma_i^{(t)})$$

Just evaluate a Gaussian at x_j

M-step

Compute Max. like **μ** given our data's class membership distributions

$$\mu_i^{(t+1)} = \frac{\sum_j P(y = i \mid x_j, \lambda_t) x_j}{\sum_j P(y = i \mid x_j, \lambda_t)}$$

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

m = #records

©2005-2007 Carlos Guestrin

E.M. for General GMMs

$p_i^{(t)}$ is shorthand for estimate of $P(y=i)$ on t'th iteration

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

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

Just evaluate a Gaussian at x_j

M-step

Compute Max. like **μ** given our data's class membership distributions

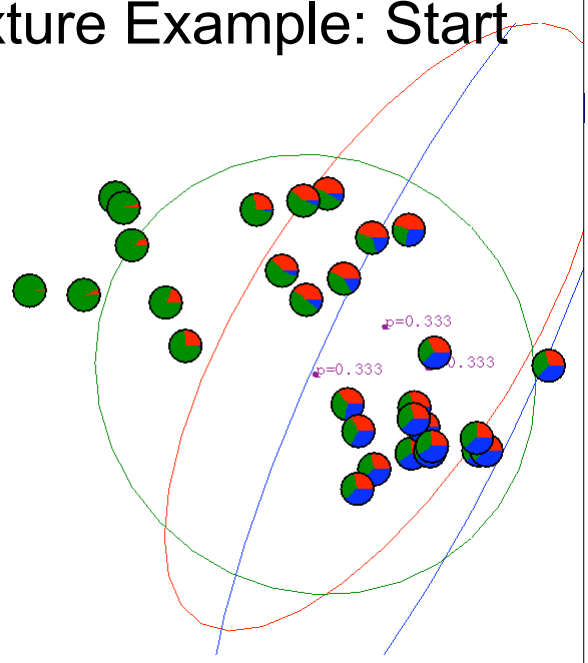
$$\mu_i^{(t+1)} = \frac{\sum_j P(y = i | x_j, \lambda_t) x_j}{\sum_j P(y = i | x_j, \lambda_t)} \qquad \Sigma_i^{(t+1)} = \frac{\sum_j P(y = i | x_j, \lambda_t) [x_j - \mu_i^{(t+1)}][x_j - \mu_i^{(t+1)}]^T}{\sum_j P(y = i | x_j, \lambda_t)}$$

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

m = #records

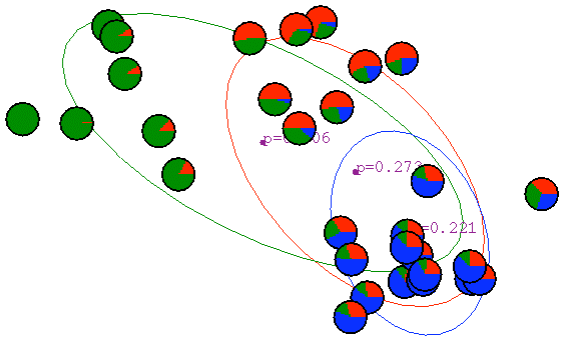
©2005-2007 Carlos Guestrin

Gaussian Mixture Example: Start



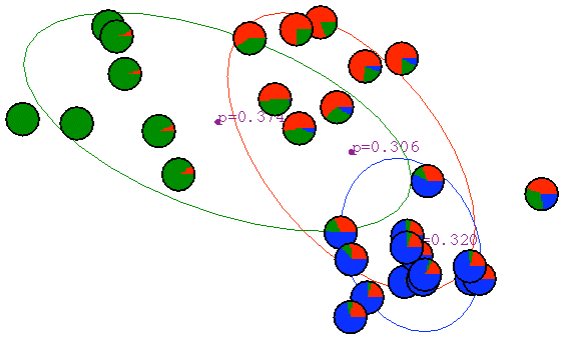
©2005-2007 Carlos Guestrin

After first iteration



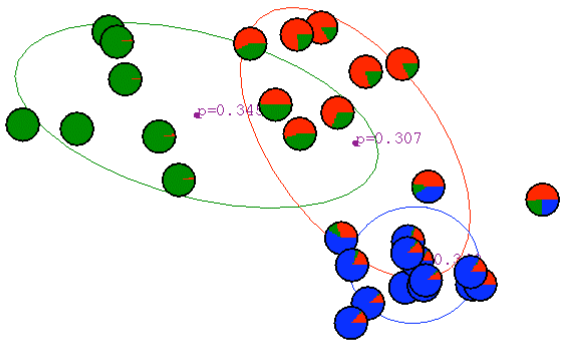
©2005-2007 Carlos Guestrin

After 2nd iteration

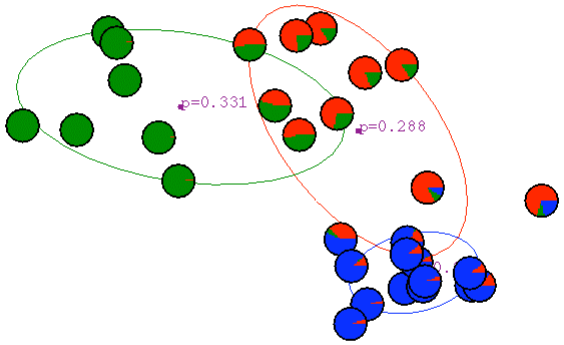


©2005-2007 Carlos Guestrin

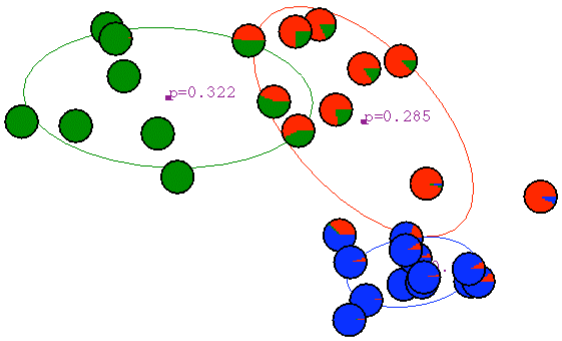
After 3rd iteration



After 4th iteration

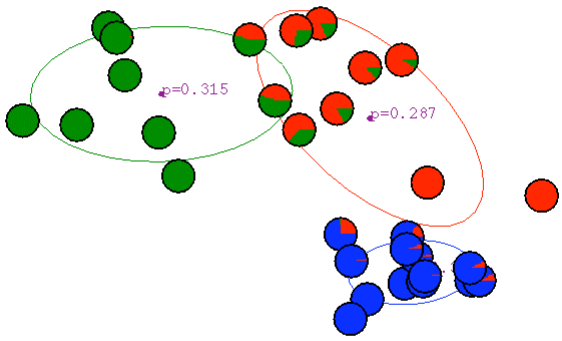


After 5th iteration



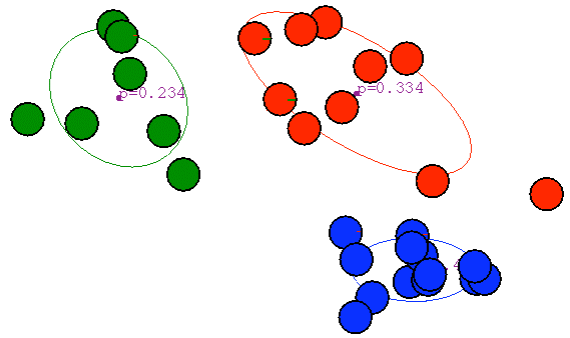
©2005-2007 Carlos Guestrin

After 6th iteration



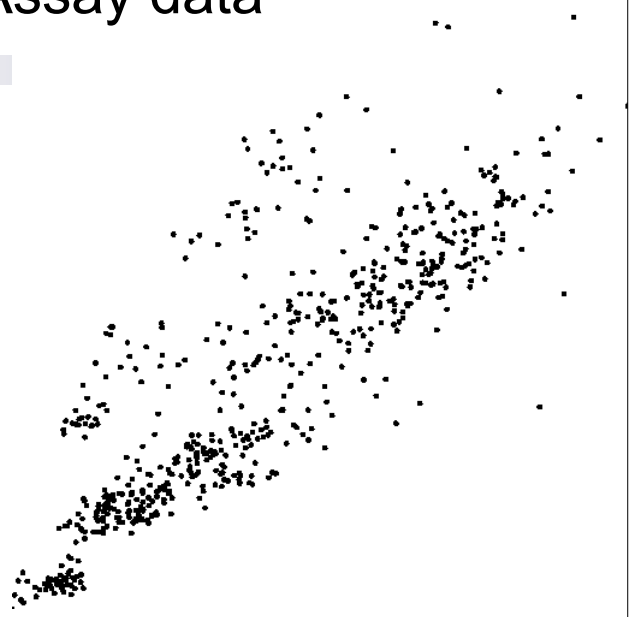
©2005-2007 Carlos Guestrin

After 20th iteration



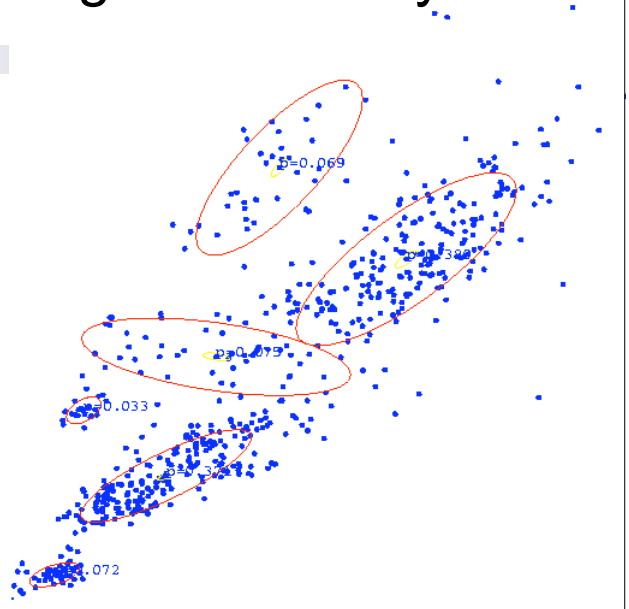
©2005-2007 Carlos Guestrin

Some Bio Assay data

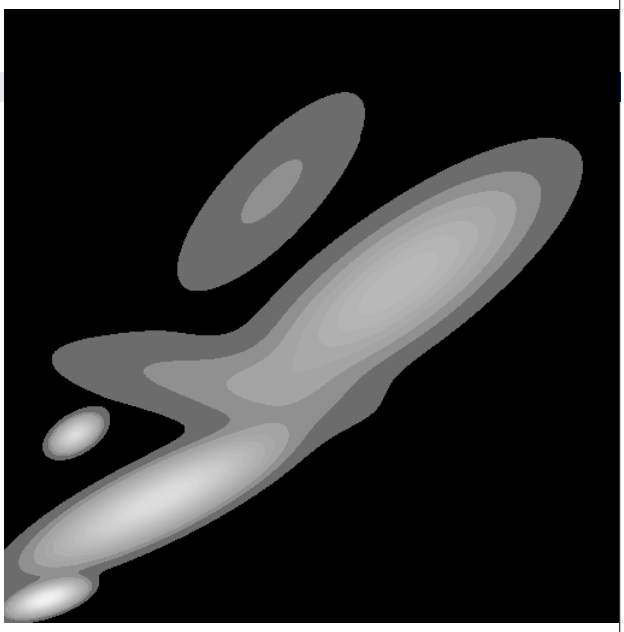


©2005-2007 Carlos Guestrin

GMM clustering of the assay data

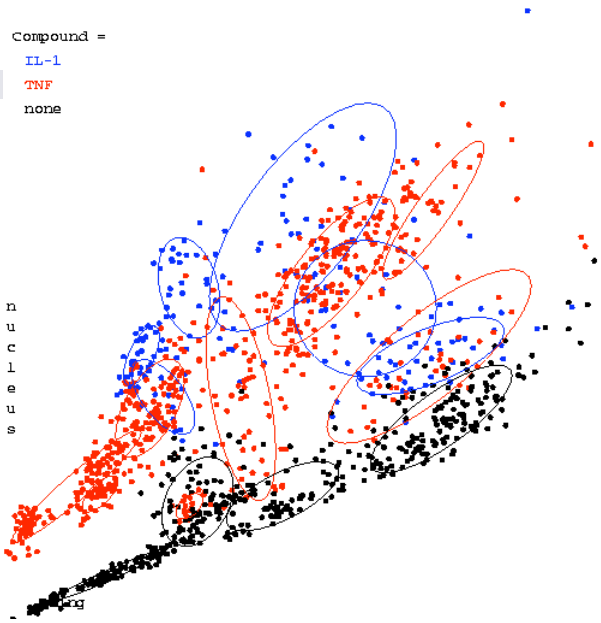


Resulting Density Estimator



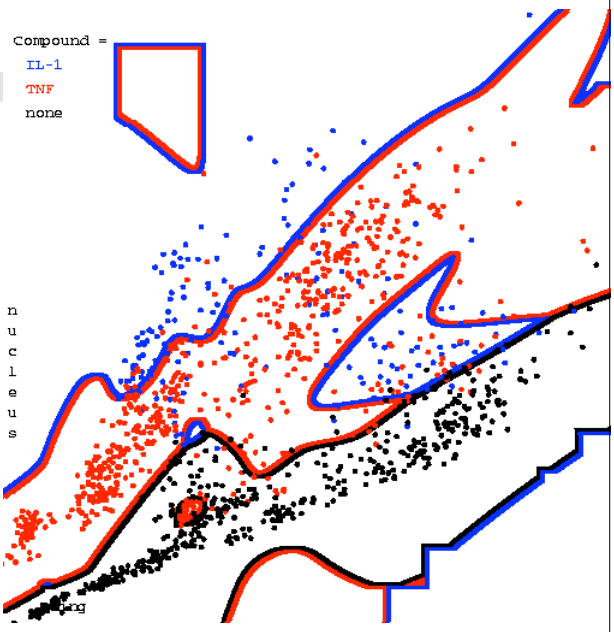
Three classes of assay

(each learned with it's own mixture model)



©2005-2007 Carlos Guestrin

Resulting Bayes Classifier

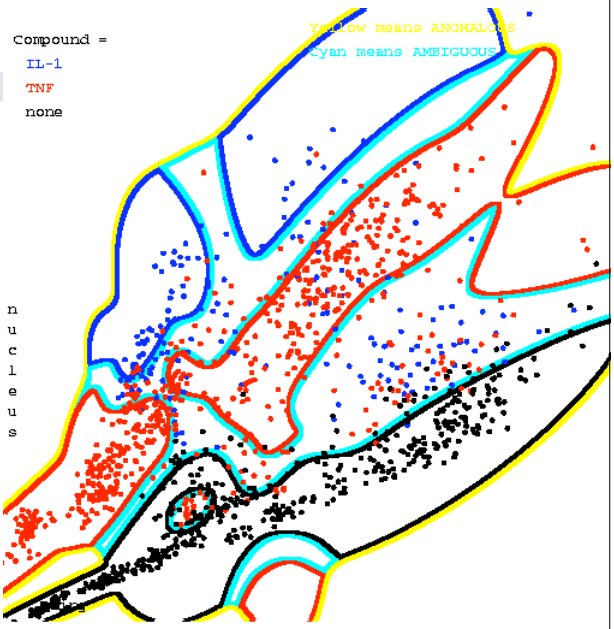


©2005-2007 Carlos Guestrin

Resulting Bayes Classifier, using posterior probabilities to alert about ambiguity and anomalousness

Yellow means anomalous

Cyan means ambiguous



©2005-2007 Carlos Guestrin

Announcements

- Project:
  - Poster session: NSH Atrium, Friday 11/30, 2-5pm
    - Print your poster early!!!
      - SCS facilities has a poster printer, ask helpdesk
      - Students from outside SCS should check with their departments
      - It's OK to print separate pages
    - We'll provide pins, posterboard and an easel
      - Poster size: 32x40 inches
    - Invite your friends, there will be a prize for best poster, by popular vote
- Last lecture:
  - Thursday, 11/29, 5-6:20pm, Wean 7500

©2005-2007 Carlos Guestrin

The general learning problem with missing data

- Marginal likelihood – **x** is observed, **z** is missing:

$$\begin{aligned}\ell(\theta : \mathcal{D}) &= \log \prod_{j=1}^{m} P(\mathbf{x}_j \mid \theta) \\ &= \sum_{j=1}^{m} \log P(\mathbf{x}_j \mid \theta) \\ &= \sum_{j=1}^{m} \log \sum_{\mathbf{z}} P(\mathbf{x}_j, \mathbf{z} \mid \theta)\end{aligned}$$

©2005-2007 Carlos Guestrin

E-step

- **x** is observed, **z** is missing
- Compute probability of missing data given current choice of θ
  - Q(**z**|$\mathbf{x}_j$) for each $\mathbf{x}_j$
    - e.g., probability computed during classification step
    - corresponds to "classification step" in K-means

$$Q^{(t+1)}(\mathbf{z} \mid \mathbf{x}_j) = P(\mathbf{z} \mid \mathbf{x}_j, \theta^{(t)})$$

©2005-2007 Carlos Guestrin

Jensen's inequality

$$\ell(\theta : \mathcal{D}) \;=\; \sum_{j=1}^{m} \log \sum_{\mathbf{z}} P(\mathbf{z} \mid \mathbf{x}_j) P(\mathbf{x}_j \mid \theta)$$

- **Theorem**: log $\sum_{\mathbf{z}}$ P(**z**) f(**z**) ≥ $\sum_{\mathbf{z}}$ P(**z**) log f(**z**)

©2005-2007 Carlos Guestrin

Applying Jensen's inequality

- Use: log $\sum_{\mathbf{z}}$ P(**z**) f(**z**) ≥ $\sum_{\mathbf{z}}$ P(**z**) log f(**z**)

$$\ell(\theta^{(t)} : \mathcal{D}) \;=\; \sum_{j=1}^{m} \log \sum_{\mathbf{z}} Q^{(t+1)}(\mathbf{z} \mid \mathbf{x}_j) \frac{P(\mathbf{z}, \mathbf{x}_j \mid \theta^{(t)})}{Q^{(t+1)}(\mathbf{z} \mid \mathbf{x}_j)}$$

©2005-2007 Carlos Guestrin

The M-step maximizes lower bound on weighted data

- Lower bound from Jensen's:

$$\ell(\theta^{(t)} : \mathcal{D}) \;\geq\; \sum_{j=1}^{m} \sum_{\mathbf{z}} Q^{(t+1)}(\mathbf{z} \mid \mathbf{x}_j) \log P(\mathbf{z}, \mathbf{x}_j \mid \theta^{(t)}) + m.H(Q^{(t+1)})$$

- Corresponds to weighted dataset:
  - <$\mathbf{x}_1$,$\mathbf{z}$=1> with weight $Q^{(t+1)}(\mathbf{z}=1|\mathbf{x}_1)$
  - <$\mathbf{x}_1$,$\mathbf{z}$=2> with weight $Q^{(t+1)}(\mathbf{z}=2|\mathbf{x}_1)$
  - <$\mathbf{x}_1$,$\mathbf{z}$=3> with weight $Q^{(t+1)}(\mathbf{z}=3|\mathbf{x}_1)$
  - <$\mathbf{x}_2$,$\mathbf{z}$=1> with weight $Q^{(t+1)}(\mathbf{z}=1|\mathbf{x}_2)$
  - <$\mathbf{x}_2$,$\mathbf{z}$=2> with weight $Q^{(t+1)}(\mathbf{z}=2|\mathbf{x}_2)$
  - <$\mathbf{x}_2$,$\mathbf{z}$=3> with weight $Q^{(t+1)}(\mathbf{z}=3|\mathbf{x}_2)$
  - …

©2005-2007 Carlos Guestrin

The M-step

$$\ell(\theta^{(t)} : \mathcal{D}) \;\geq\; \sum_{j=1}^{m} \sum_{\mathbf{z}} Q^{(t+1)}(\mathbf{z} \mid \mathbf{x}_j) \log P(\mathbf{z}, \mathbf{x}_j \mid \theta^{(t)}) + m.H(Q^{(t+1)})$$

- Maximization step:

$$\theta^{(t+1)} \leftarrow \arg\max_{\theta} \sum_{j=1}^{m} \sum_{\mathbf{z}} Q^{(t+1)}(\mathbf{z} \mid \mathbf{x}_j) \log P(\mathbf{z}, \mathbf{x}_j \mid \theta)$$

- Use expected counts instead of counts:
  - If learning requires Count($\mathbf{x}$,$\mathbf{z}$)
  - Use $E_{Q(t+1)}$[Count($\mathbf{x}$,$\mathbf{z}$)]

©2005-2007 Carlos Guestrin

Convergence of EM

- Define potential function F(θ,Q):

$$\ell(\theta : \mathcal{D}) \;\geq\; F(\theta, Q) = \sum_{j=1}^{m} \sum_{\mathbf{z}} Q(\mathbf{z} \mid \mathbf{x}_j) \log \frac{P(\mathbf{z}, \mathbf{x}_j \mid \theta)}{Q(\mathbf{z} \mid \mathbf{x}_j)}$$

- EM corresponds to coordinate ascent on F
  - Thus, maximizes lower bound on marginal log likelihood

©2005-2007 Carlos Guestrin

M-step is easy

$$\theta^{(t+1)} \leftarrow \arg\max_{\theta} \sum_{j=1}^{m} \sum_{\mathbf{z}} Q^{(t+1)}(\mathbf{z} \mid \mathbf{x}_j) \log P(\mathbf{z}, \mathbf{x}_j \mid \theta)$$

- Using potential function

$$F(\theta, Q^{(t+1)}) \;=\; \sum_{j=1}^{m} \sum_{\mathbf{z}} Q^{(t+1)}(\mathbf{z} \mid \mathbf{x}_j) \log P(\mathbf{z}, \mathbf{x}_j \mid \theta) + m.H(Q^{(t+1)})$$

©2005-2007 Carlos Guestrin

E-step also doesn't decrease potential function 1

- Fixing θ to $\theta^{(t)}$:

$$\ell(\theta^{(t)} : \mathcal{D}) \;\geq\; F(\theta^{(t)}, Q) = \sum_{j=1}^{m} \sum_{\mathbf{z}} Q(\mathbf{z} \mid \mathbf{x}_j) \log \frac{P(\mathbf{z}, \mathbf{x}_j \mid \theta^{(t)})}{Q(\mathbf{z} \mid \mathbf{x}_j)}$$

©2005-2007 Carlos Guestrin

KL-divergence

- Measures distance between distributions

$$KL(Q||P) = \sum_{z} Q(z) \log \frac{Q(z)}{P(z)}$$

- KL=zero if and only if Q=P

©2005-2007 Carlos Guestrin

E-step also doesn't decrease potential function 2

- Fixing θ to $\theta^{(t)}$:

$$\ell(\theta^{(t)} : \mathcal{D}) \geq F(\theta^{(t)}, Q) = \ell(\theta^{(t)} : \mathcal{D}) + \sum_{j=1}^{m} \sum_{\mathbf{z}} Q(\mathbf{z} \mid \mathbf{x}_j) \log \frac{P(\mathbf{z} \mid \mathbf{x}_j, \theta^{(t)})}{Q(\mathbf{z} \mid \mathbf{x}_j)}$$

$$= \ell(\theta^{(t)} : \mathcal{D}) - m \sum_{j=1}^{m} KL\left(Q(\mathbf{z} \mid \mathbf{x}_j) || P(\mathbf{z} \mid \mathbf{x}_j, \theta^{(t)})\right)$$

©2005-2007 Carlos Guestrin

E-step also doesn't decrease potential function 3

$$\ell(\theta^{(t)} : \mathcal{D}) \geq F(\theta^{(t)}, Q) = \ell(\theta^{(t)} : \mathcal{D}) - m \sum_{j=1}^{m} KL\left(Q(\mathbf{z} \mid \mathbf{x}_j) || P(\mathbf{z} \mid \mathbf{x}_j, \theta^{(t)})\right)$$

- Fixing θ to $\theta^{(t)}$
- Maximizing $F(\theta^{(t)},Q)$ over Q → set Q to posterior probability:

$$Q^{(t+1)}(\mathbf{z} \mid \mathbf{x}_j) \leftarrow P(\mathbf{z} \mid \mathbf{x}_j, \theta^{(t)})$$

- Note that

$$F(\theta^{(t)}, Q^{(t+1)}) = \ell(\theta^{(t)} : \mathcal{D})$$

©2005-2007 Carlos Guestrin

EM is coordinate ascent

$$\ell(\theta : \mathcal{D}) \;\geq\; F(\theta, Q) = \sum_{j=1}^{m} \sum_{\mathbf{z}} Q(\mathbf{z} \mid \mathbf{x}_j) \log \frac{P(\mathbf{z}, \mathbf{x}_j \mid \theta)}{Q(\mathbf{z} \mid \mathbf{x}_j)}$$

- **M-step**: Fix Q, maximize F over θ (a lower bound on $\ell(\theta : \mathcal{D})$):

$$\ell(\theta : \mathcal{D}) \;\geq\; F(\theta, Q^{(t)}) = \sum_{j=1}^{m} \sum_{\mathbf{z}} Q^{(t)}(\mathbf{z} \mid \mathbf{x}_j) \log P(\mathbf{z}, \mathbf{x}_j \mid \theta) + m.H(Q^{(t)})$$

- **E-step**: Fix θ, maximize F over Q:

$$\ell(\theta^{(t)} : \mathcal{D}) \geq F(\theta^{(t)}, Q) \;=\; \ell(\theta^{(t)} : \mathcal{D}) - m \sum_{j=1}^{m} KL\left(Q(\mathbf{z} \mid \mathbf{x}_j) || P(\mathbf{z} \mid \mathbf{x}_j, \theta^{(t)})\right)$$

  - "Realigns" F with likelihood:

$$F(\theta^{(t)}, Q^{(t+1)}) \;=\; \ell(\theta^{(t)} : \mathcal{D})$$

©2005-2007 Carlos Guestrin

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
- Remember, E.M. can get stuck in local minima, and empirically it <u>DOES</u>
- EM is coordinate ascent
- General case for EM

©2005-2007 Carlos Guestrin

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

©2005-2007 Carlos Guestrin

Dimensionality Reduction

Machine Learning – 10701/15781

Carlos Guestrin
Carnegie Mellon University

November 26th, 2007

©2005-2007 Carlos Guestrin

Dimensionality reduction

- Input data may have thousands or millions of dimensions!
  - e.g., text data has
- **Dimensionality reduction**: represent data with fewer dimensions
  - easier learning – fewer parameters
  - visualization – hard to visualize more than 3D or 4D
  - discover "intrinsic dimensionality" of data
    - high dimensional data that is truly lower dimensional

©2005-2007 Carlos Guestrin

Feature selection

- Want to learn f:**X**→Y
  - **X**=<X_1,…,X_n>
  - but some features are more important than others
- **Approach**: select subset of features to be used by learning algorithm
  - **Score** each feature (or sets of features)
  - **Select** set of features with best score

©2005-2007 Carlos Guestrin

Simple greedy **forward** feature selection algorithm

- Pick a dictionary of features
  - e.g., polynomials for linear regression
- Greedy heuristic:
  - Start from empty (or simple) set of features $F_0 = \emptyset$
  - Run learning algorithm for current set of features F_t
    - Obtain h_t
  - Select **next best feature X_i**
    - e.g., X_j that results in lowest cross-validation error learner when learning with $F_t \cup \{X_j\}$
  - $F_{t+1} \leftarrow F_t \cup \{X_i\}$
  - Recurse

©2005-2007 Carlos Guestrin

Simple greedy **backward** feature selection algorithm

- Pick a dictionary of features
  - e.g., polynomials for linear regression
- Greedy heuristic:
  - Start from all features $F_0 = F$
  - Run learning algorithm for current set of features F_t
    - Obtain h_t
  - Select **next worst feature X_i**
    - e.g., X_j that results in lowest cross-validation error learner when learning with $F_t - \{X_j\}$
  - $F_{t+1} \leftarrow F_t - \{X_i\}$
  - Recurse

©2005-2007 Carlos Guestrin

Impact of feature selection on classification of fMRI data [Pereira et al. '05]

Accuracy classifying category of word read by subject

#voxels	mean	subjects 233B	329B	332B	424B	474B	496B	77B	86B
50	0.735	0.783	0.817	0.55	0.783	0.75	0.8	0.65	0.75
100	0.742	0.767	0.8	0.533	0.817	0.85	0.783	0.6	0.783
200	0.737	0.783	0.783	0.517	0.817	0.883	0.75	0.583	0.783
300	**0.75**	**0.8**	**0.817**	**0.567**	**0.833**	**0.883**	**0.75**	**0.583**	**0.767**
400	0.742	0.8	0.783	0.583	0.85	0.833	0.75	0.583	0.75
800	0.735	0.833	0.817	0.567	0.833	0.833	0.7	0.55	0.75
1600	0.698	0.8	0.817	0.45	0.783	0.833	0.633	0.5	0.75
all (∼2500)	0.638	0.767	0.767	0.25	0.75	0.833	0.567	0.433	0.733

Table 1: **Average accuracy across all pairs of categories, restricting the procedure to use a certain number of voxels for each subject.** The highlighted line corresponds to the best mean accuracy, obtained using 300 voxels.

Voxels scored by p-value of regression to predict voxel value from the task

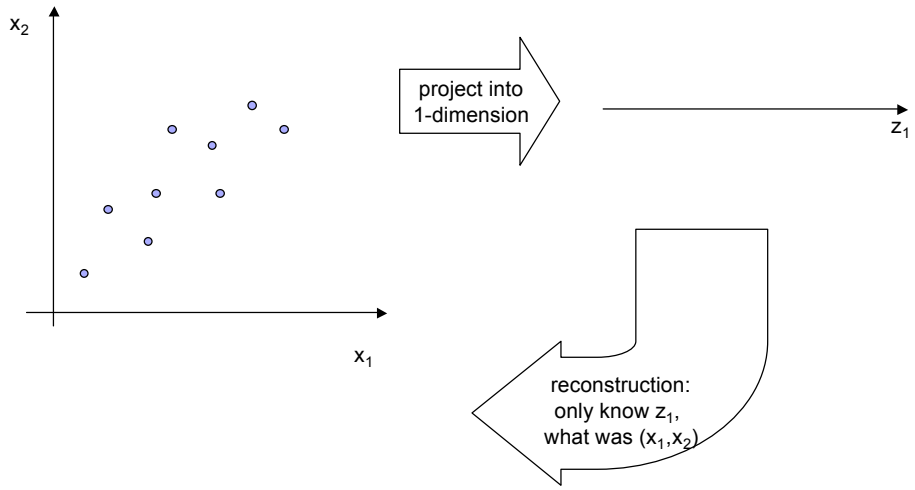
©2005-2007 Carlos Guestrin

Lower dimensional projections

- Rather than picking a subset of the features, we can new features that are combinations of existing features
- Let's see this in the unsupervised setting
  - just **X**, but no Y

©2005-2007 Carlos Guestrin

Linear projection and reconstruction



Principal component analysis – basic idea

- Project n-dimensional data into k-dimensional space while preserving information:
  - e.g., project space of 10000 words into 3-dimensions
  - e.g., project 3-d into 2-d
- Choose projection with minimum reconstruction error

Linear projections, a review

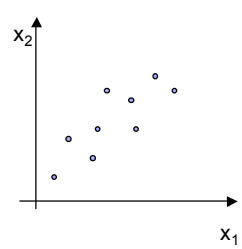
- Project a point into a (lower dimensional) space:
  - **point**: $\mathbf{x} = (x_1,\ldots,x_n)$
  - **select a basis** – set of basis vectors – $(\mathbf{u}_1,\ldots,\mathbf{u}_k)$
    - we consider orthonormal basis:
      - $\mathbf{u}_i \bullet \mathbf{u}_i = 1$, and $\mathbf{u}_i \bullet \mathbf{u}_j = 0$ for $i \neq j$
  - **select a center** – $\bar{\mathbf{x}}$, defines offset of space
  - **best coordinates** in lower dimensional space defined by dot-products: $(z_1,\ldots,z_k)$, $z_i = (\mathbf{x}-\bar{\mathbf{x}}) \bullet \mathbf{u}_i$
    - minimum squared error

©2005-2007 Carlos Guestrin

PCA finds projection that minimizes reconstruction error

- Given m data points: $\mathbf{x}^i = (x_1^i,\ldots,x_n^i)$, i=1…m
- Will represent each point as a projection:
  - $\hat{\mathbf{x}}^i = \bar{\mathbf{x}} + \sum_{j=1}^{k} z_j^i \mathbf{u}_j$ where: $\bar{\mathbf{x}} = \frac{1}{m}\sum_{i=1}^{m} \mathbf{x}^i$ and $z_j^i = (\mathbf{x}^i - \bar{\mathbf{x}}) \cdot \mathbf{u}_j$
- PCA:
  - Given $k \leq n$, find $(\mathbf{u}_1,\ldots,\mathbf{u}_k)$ minimizing reconstruction error:

$$error_k = \sum_{i=1}^{m} (\mathbf{x}^i - \hat{\mathbf{x}}^i)^2$$



©2005-2007 Carlos Guestrin

Understanding the reconstruction error

- Note that $\mathbf{x}^i$ can be represented exactly by n-dimensional projection:

$$\mathbf{x}^i = \bar{\mathbf{x}} + \sum_{j=1}^{n} z_j^i \mathbf{u}_j$$

- Rewriting error:

$$\hat{\mathbf{x}}^i = \bar{\mathbf{x}} + \sum_{j=1}^{k} z_j^i \mathbf{u}_j$$

$$z_j^i = (\mathbf{x}^i - \bar{\mathbf{x}}) \cdot \mathbf{u}_j$$

- Given $k \leq n$, find $(\mathbf{u}_1, \ldots, \mathbf{u}_k)$ minimizing reconstruction error:

$$error_k = \sum_{i=1}^{m} (\mathbf{x}^i - \hat{\mathbf{x}}^i)^2$$

©2005-2007 Carlos Guestrin

Reconstruction error and covariance matrix

$$error_k = \sum_{i=1}^{m} \sum_{j=k+1}^{n} [\mathbf{u}_j \cdot (\mathbf{x}^i - \bar{\mathbf{x}})]^2$$

$$\Sigma = \frac{1}{m} \sum_{i=1}^{m} (\mathbf{x}^i - \bar{\mathbf{x}})(\mathbf{x}^i - \bar{\mathbf{x}})^T$$

©2005-2007 Carlos Guestrin

Minimizing reconstruction error and eigen vectors

- Minimizing reconstruction error equivalent to picking orthonormal basis ($\mathbf{u}_1,\ldots,\mathbf{u}_n$) minimizing:

$$error_k = \sum_{j=k+1}^{n} \mathbf{u}_j^T \Sigma \mathbf{u}_j$$

- Eigen vector:

- Minimizing reconstruction error equivalent to picking ($\mathbf{u}_{k+1},\ldots,\mathbf{u}_n$) to be eigen vectors with smallest eigen values

©2005-2007 Carlos Guestrin

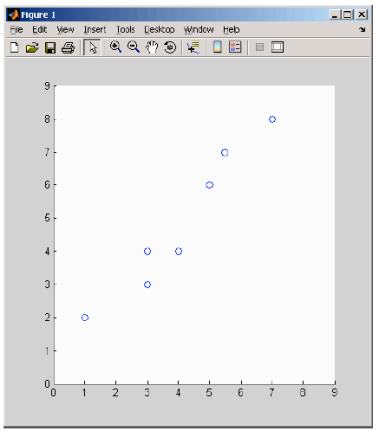
Basic PCA algoritm

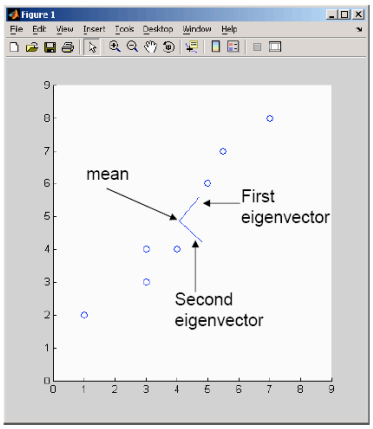
- Start from m by n data matrix **X**
- **Recenter**: subtract mean from each row of **X**
  - $\mathbf{X_c} \leftarrow \mathbf{X} - \overline{\mathbf{X}}$
- **Compute covariance matrix**:
  - $\Sigma \leftarrow 1/m\ \mathbf{X_c}^T \mathbf{X_c}$
- Find **eigen vectors and values** of Σ
- **Principal components:** k eigen vectors with highest eigen values

©2005-2007 Carlos Guestrin

PCA example

$$\hat{\mathbf{x}}^i = \bar{\mathbf{x}} + \sum_{j=1}^{k} z_j^i \mathbf{u}_j$$

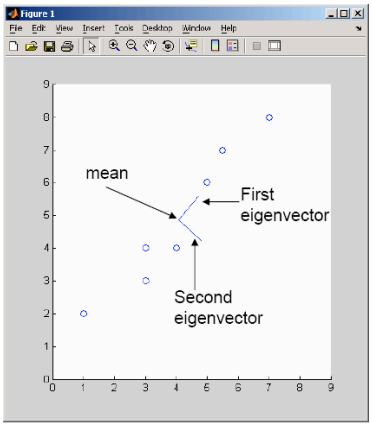


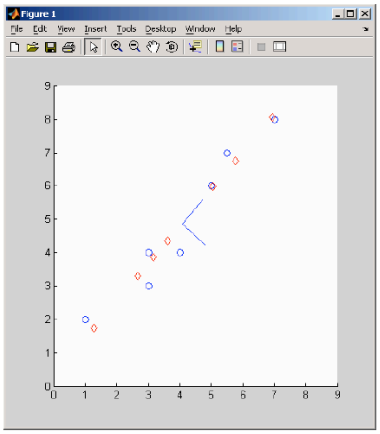


PCA example – reconstruction

$$\hat{\mathbf{x}}^i = \bar{\mathbf{x}} + \sum_{j=1}^{k} z_j^i \mathbf{u}_j$$

only used first principal component

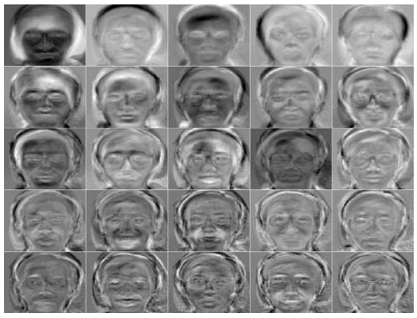




Eigenfaces [Turk, Pentland '91]

- Input images:

- Principal components:



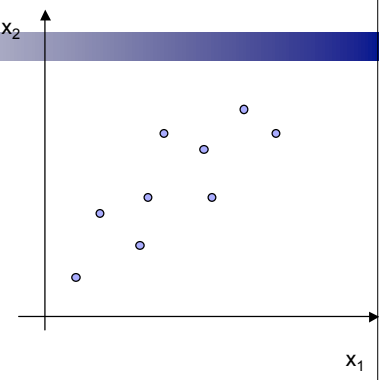
©2005-2007 Carlos Guestrin

Eigenfaces reconstruction

- Each image corresponds to adding 8 principal components:

©2005-2007 Carlos Guestrin

Relationship to Gaussians



- PCA assumes data is Gaussian
  - $\mathbf{x} \sim N(\bar{\mathbf{x}};\Sigma)$
- Equivalent to weighted sum of simple Gaussians:

$$\mathbf{x} = \bar{\mathbf{x}} + \sum_{j=1}^{n} z_j \mathbf{u}_j; \quad z_j \sim N(0;\sigma_j^2)$$

- Selecting top k principal components equivalent to lower dimensional Gaussian approximation:

$$\mathbf{x} \approx \bar{\mathbf{x}} + \sum_{j=1}^{k} z_j \mathbf{u}_j + \varepsilon; \quad z_j \sim N(0;\sigma_j^2)$$

  - $\varepsilon \sim N(0;\sigma^2)$, where σ^2 is defined by error_k

©2005-2007 Carlos Guestrin

Scaling up

- Covariance matrix can be really big!
  - Σ is n by n
  - 10000 features $\rightarrow$ $|\Sigma|$
  - finding eigenvectors is very slow…
- Use singular value decomposition (SVD)
  - finds to k eigenvectors
  - great implementations available, e.g., Matlab svd

©2005-2007 Carlos Guestrin

SVD

- Write $\mathbf{X} = \mathbf{W}\,\mathbf{S}\,\mathbf{V}^T$
  - $\mathbf{X} \leftarrow$ data matrix, one row per datapoint
  - $\mathbf{W} \leftarrow$ weight matrix, one row per datapoint – coordinate of $\mathbf{x}^i$ in eigenspace
  - $\mathbf{S} \leftarrow$ singular value matrix, diagonal matrix
    - in our setting each entry is eigenvalue λ_j
  - $\mathbf{V}^T \leftarrow$ singular vector matrix
    - in our setting each row is eigenvector $\mathbf{v}_j$

©2005-2007 Carlos Guestrin

PCA using SVD algoritm

- Start from m by n data matrix $\mathbf{X}$
- **Recenter**: subtract mean from each row of $\mathbf{X}$
  - $\mathbf{X_c} \leftarrow \mathbf{X} - \overline{\mathbf{X}}$
- Call SVD algorithm on $\mathbf{X_c}$ – ask for k singular vectors
- **Principal components:** k singular vectors with highest singular values (rows of $\mathbf{V}^T$)
  - **Coefficients** become:

©2005-2007 Carlos Guestrin

Using PCA for dimensionality reduction in classification

- Want to learn f:**X**→Y
  - **X**=<X_1,…,X_n>
  - but some features are more important than others
- **Approach**: Use PCA on **X** to select a few important features

©2005-2007 Carlos Guestrin

PCA for classification can lead to problems…

- Direction of maximum variation may be unrelated to "discriminative" directions:
- PCA often works very well, but sometimes must use more advanced methods
  - e.g., Fisher linear discriminant

©2005-2007 Carlos Guestrin

What you need to know

- Dimensionality reduction
  - why and when it's important
- Simple feature selection
- Principal component analysis
  - minimizing reconstruction error
  - relationship to covariance matrix and eigenvectors
  - using SVD
  - problems with PCA

©2005-2007 Carlos Guestrin