

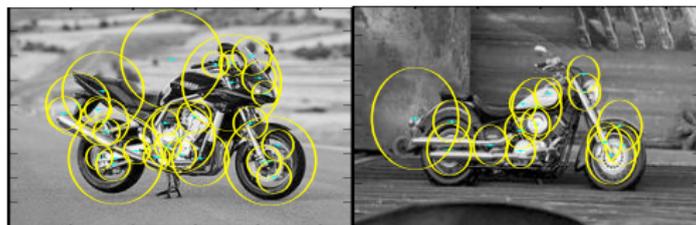
Latent Variable Models and Expectation Maximization

Oliver Schulte - CMPT 726

Bishop PRML Ch. 9

Learning Parameters to Probability Distributions

- We discussed probabilistic models at length
- Assignment 3: given fully observed training data, setting parameters θ_i for Bayes nets is straight-forward
- However, in many settings not all variables are observed (labelled) in the training data: $x_i = (x_i, h_i)$
 - e.g. Speech recognition: have speech signals, but not phoneme labels
 - e.g. Object recognition: have object labels (car, bicycle), but not part labels (wheel, door, seat)
 - Unobserved variables are called **latent variables**



figs from Fergus et al.

Latent Variable Models: Pros

- Statistically powerful, often good predictions. Many applications:
- Learning with **missing data**.
- **Clustering**: “missing” cluster label for data points.
- **Principal Component Analysis**: data points are generated in linear fashion from a small set of unobserved components. (more later)
- **Matrix Factorization, Recommender Systems**:
 - Assign users to unobserved “user types”, assign items to unobserved “item types”.
 - Use similarity between user type, item type to predict preference of user for item.
 - Winner of \$1M Netflix challenge.
- If latent variables have an intuitive interpretation (e.g., “action movies”, “factors”), discovers **new features**.

Latent Variable Models: Cons

- From a user's point of view, like a black box if latent variables don't have an intuitive interpretation.
- Statistically, hard to guarantee convergence to a correct model with more data (the **identifiability** problem).
- Harder computationally, usually no closed form for maximum likelihood estimates.
- However, the **Expectation-Maximization** algorithm provides a *general-purpose* local search algorithm for learning parameters in probabilistic models with latent variables.

Outline

K-Means

The Expectation Maximization Algorithm

EM Example: Gaussian Mixture Models

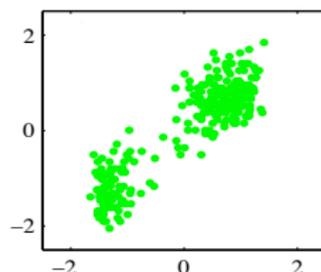
Outline

K-Means

The Expectation Maximization Algorithm

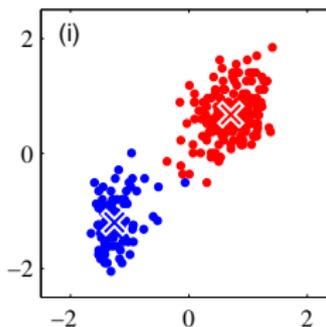
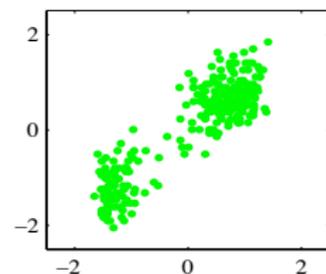
EM Example: Gaussian Mixture Models

Unsupervised Learning



- We will start with an unsupervised learning (clustering) problem:
- Given a dataset $\{\mathbf{x}_1, \dots, \mathbf{x}_N\}$, each $\mathbf{x}_i \in \mathbb{R}^D$, partition the dataset into K clusters
 - Intuitively, a **cluster** is a group of points, which are close together and far from others

Distortion Measure



- Formally, introduce **prototypes** (or **cluster centers**) $\boldsymbol{\mu}_k \in \mathbb{R}^D$
- Use binary r_{nk} , 1 if point n is in cluster k , 0 otherwise (1-of- K coding scheme again)
- Find $\{\boldsymbol{\mu}_k\}$, $\{r_{nk}\}$ to minimize **distortion measure**:

$$J = \sum_{n=1}^N \sum_{k=1}^K r_{nk} \|\mathbf{x}_n - \boldsymbol{\mu}_k\|^2$$

Minimizing Distortion Measure

- Minimizing J directly is hard

$$J = \sum_{n=1}^N \sum_{k=1}^K r_{nk} \|\mathbf{x}_n - \boldsymbol{\mu}_k\|^2$$

- However, two things are easy
 - If we know $\boldsymbol{\mu}_k$, minimizing J wrt r_{nk}
 - If we know r_{nk} , minimizing J wrt $\boldsymbol{\mu}_k$
- This suggests an iterative procedure
 - Start with initial guess for $\boldsymbol{\mu}_k$
 - Iteration of two steps:
 - Minimize J wrt r_{nk}
 - Minimize J wrt $\boldsymbol{\mu}_k$
 - Rinse and repeat until convergence

Minimizing Distortion Measure

- Minimizing J directly is hard

$$J = \sum_{n=1}^N \sum_{k=1}^K r_{nk} \|\mathbf{x}_n - \boldsymbol{\mu}_k\|^2$$

- However, two things are easy
 - If we know $\boldsymbol{\mu}_k$, minimizing J wrt r_{nk}
 - If we know r_{nk} , minimizing J wrt $\boldsymbol{\mu}_k$
- This suggests an iterative procedure
 - Start with initial guess for $\boldsymbol{\mu}_k$
 - Iteration of two steps:
 - Minimize J wrt r_{nk}
 - Minimize J wrt $\boldsymbol{\mu}_k$
 - Rinse and repeat until convergence

Minimizing Distortion Measure

- Minimizing J directly is hard

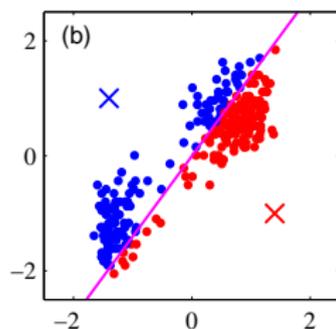
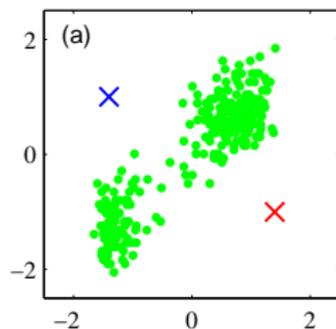
$$J = \sum_{n=1}^N \sum_{k=1}^K r_{nk} \|\mathbf{x}_n - \boldsymbol{\mu}_k\|^2$$

- However, two things are easy
 - If we know $\boldsymbol{\mu}_k$, minimizing J wrt r_{nk}
 - If we know r_{nk} , minimizing J wrt $\boldsymbol{\mu}_k$
- This suggests an iterative procedure
 - Start with initial guess for $\boldsymbol{\mu}_k$
 - Iteration of two steps:
 - Minimize J wrt r_{nk}
 - Minimize J wrt $\boldsymbol{\mu}_k$
 - Rinse and repeat until convergence

Determining Membership Variables

- Step 1 in an iteration of K-means is to minimize distortion measure J wrt cluster membership variables r_{nk}

$$J = \sum_{n=1}^N \sum_{k=1}^K r_{nk} \|\mathbf{x}_n - \boldsymbol{\mu}_k\|^2$$



- Terms for different data points \mathbf{x}_n are independent, for each data point set r_{nk} to minimize

$$\sum_{k=1}^K r_{nk} \|\mathbf{x}_n - \boldsymbol{\mu}_k\|^2$$

- Simply set $r_{nk} = 1$ for the cluster center $\boldsymbol{\mu}_k$ with smallest distance

Determining Membership Variables

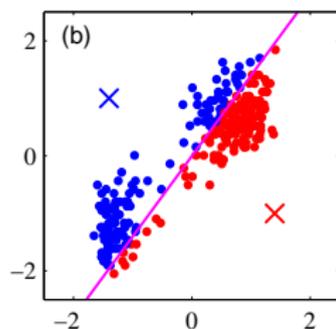
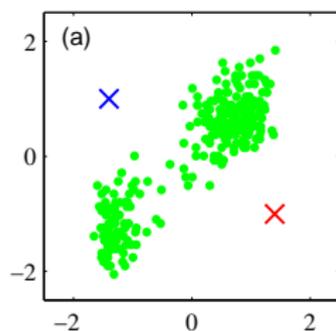
- Step 1 in an iteration of K-means is to minimize distortion measure J wrt cluster membership variables r_{nk}

$$J = \sum_{n=1}^N \sum_{k=1}^K r_{nk} \|\mathbf{x}_n - \boldsymbol{\mu}_k\|^2$$

- Terms for different data points \mathbf{x}_n are independent, for each data point set r_{nk} to minimize

$$\sum_{k=1}^K r_{nk} \|\mathbf{x}_n - \boldsymbol{\mu}_k\|^2$$

- Simply set $r_{nk} = 1$ for the cluster center $\boldsymbol{\mu}_k$ with smallest distance



Determining Membership Variables

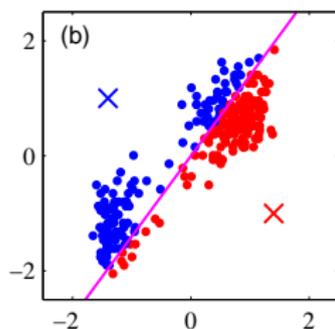
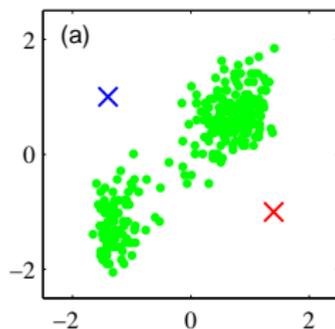
- Step 1 in an iteration of K-means is to minimize distortion measure J wrt cluster membership variables r_{nk}

$$J = \sum_{n=1}^N \sum_{k=1}^K r_{nk} \|\mathbf{x}_n - \boldsymbol{\mu}_k\|^2$$

- Terms for different data points \mathbf{x}_n are independent, for each data point set r_{nk} to minimize

$$\sum_{k=1}^K r_{nk} \|\mathbf{x}_n - \boldsymbol{\mu}_k\|^2$$

- Simply set $r_{nk} = 1$ for the cluster center $\boldsymbol{\mu}_k$ with smallest distance



Determining Cluster Centers

- Step 2: fix r_{nk} , minimize J wrt the cluster centers μ_k

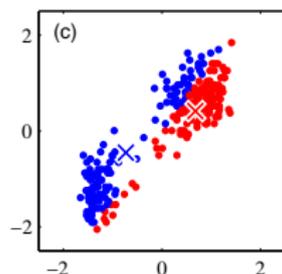
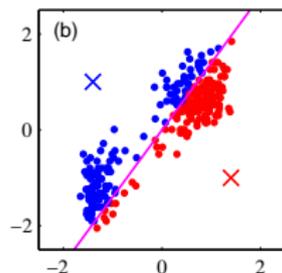
$$J = \sum_{k=1}^K \sum_{n=1}^N r_{nk} \|\mathbf{x}_n - \mu_k\|^2 \text{ switch order of sums}$$

- So we can minimize wrt each μ_k separately
- Take derivative, set to zero:

$$2 \sum_{n=1}^N r_{nk} (\mathbf{x}_n - \mu_k) = 0$$

$$\Leftrightarrow \mu_k = \frac{\sum_n r_{nk} \mathbf{x}_n}{\sum_n r_{nk}}$$

i.e. mean of datapoints \mathbf{x}_n assigned to cluster k



Determining Cluster Centers

- Step 2: fix r_{nk} , minimize J wrt the cluster centers μ_k

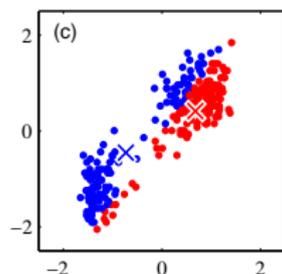
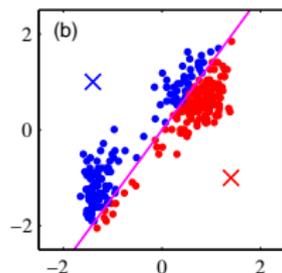
$$J = \sum_{k=1}^K \sum_{n=1}^N r_{nk} \|\mathbf{x}_n - \mu_k\|^2 \text{ switch order of sums}$$

- So we can minimize wrt each μ_k separately
- Take derivative, set to zero:

$$2 \sum_{n=1}^N r_{nk} (\mathbf{x}_n - \mu_k) = 0$$

$$\Leftrightarrow \mu_k = \frac{\sum_n r_{nk} \mathbf{x}_n}{\sum_n r_{nk}}$$

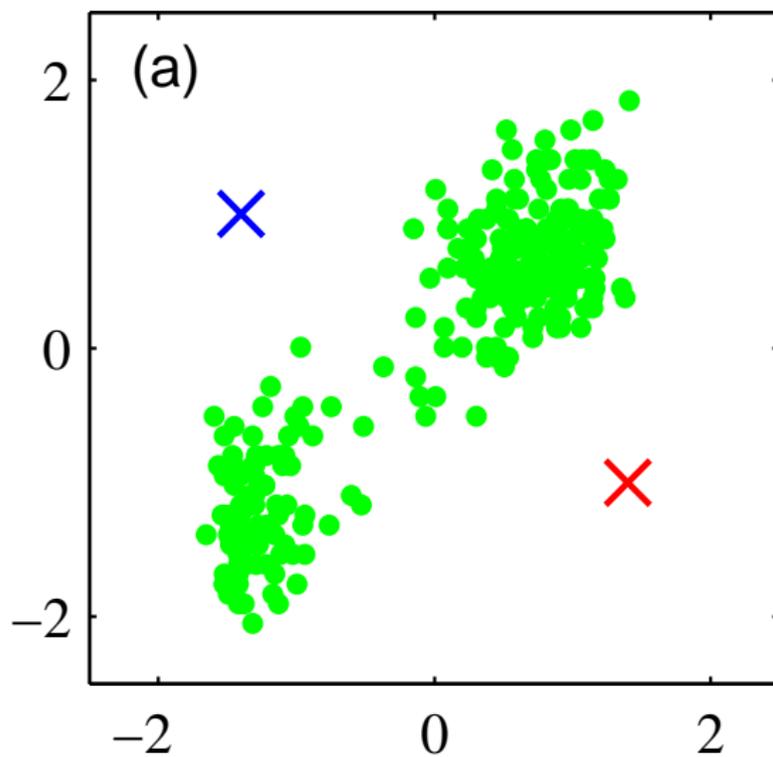
i.e. mean of datapoints \mathbf{x}_n assigned to cluster k



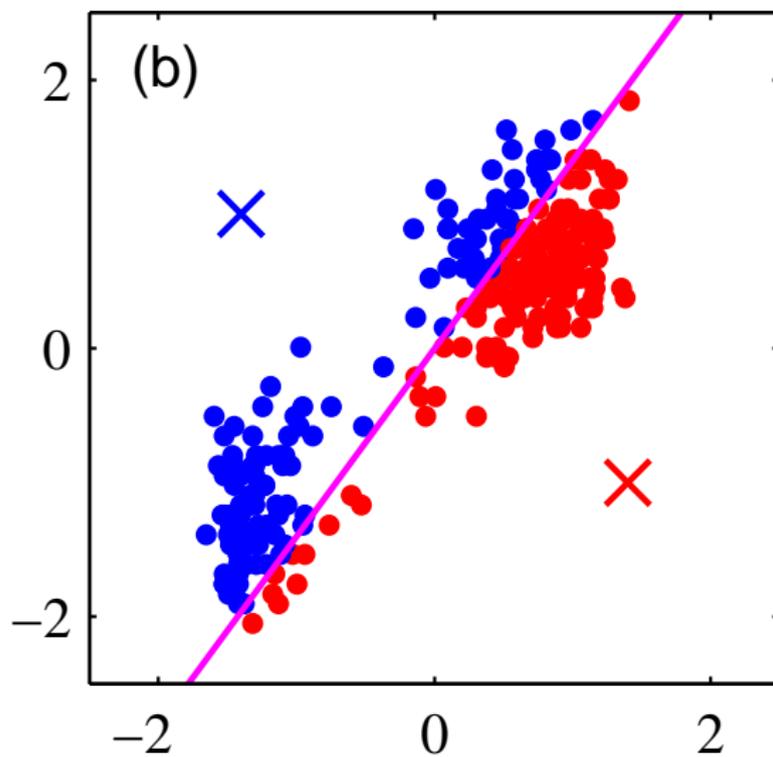
K-means Algorithm

- Start with initial guess for μ_k
- Iteration of two steps:
 - Minimize J wrt r_{nk}
 - Assign points to nearest cluster center
 - Minimize J wrt μ_k
 - Set cluster center as average of points in cluster
- Rinse and repeat until convergence

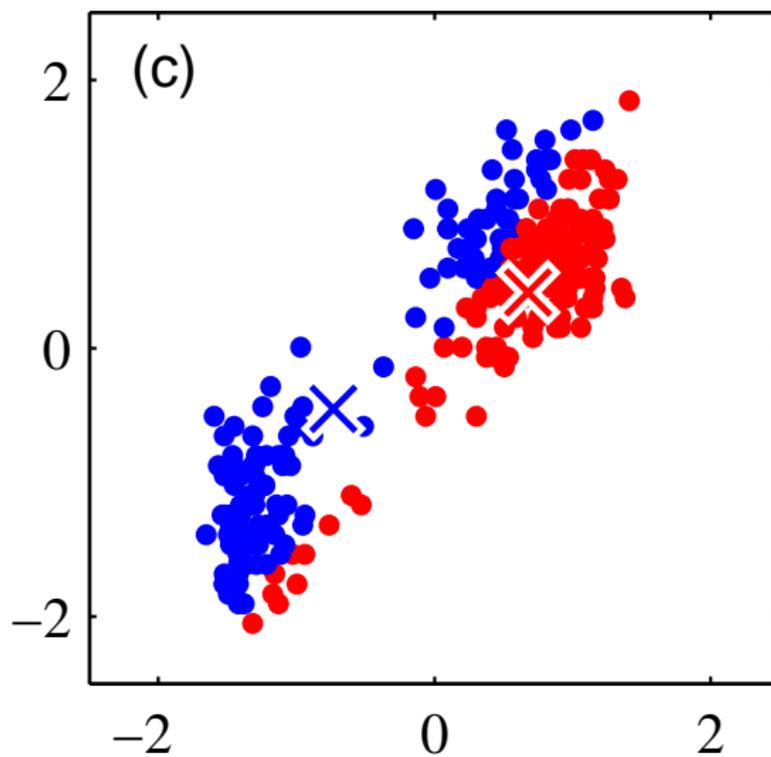
K-means example



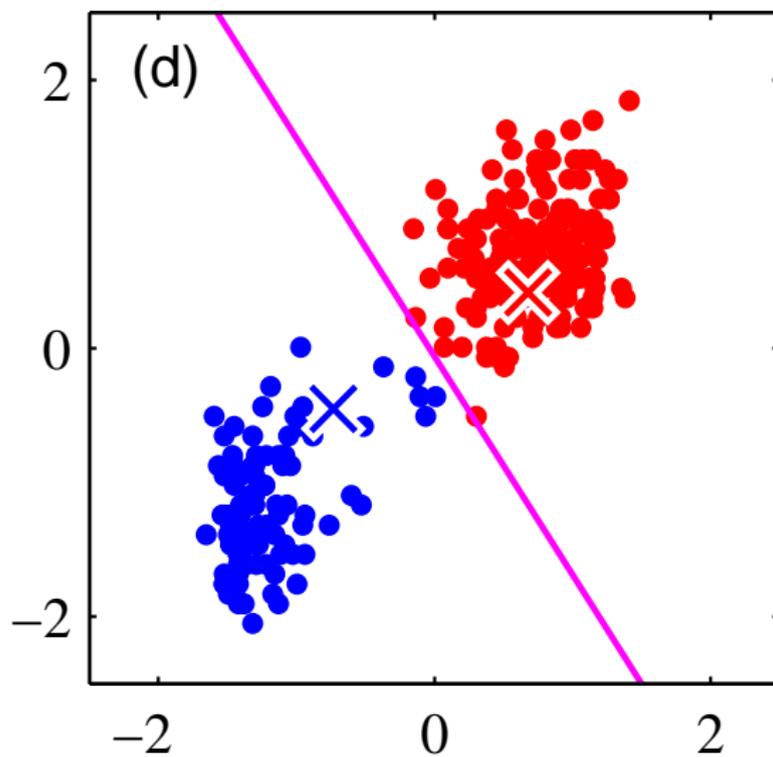
K-means example



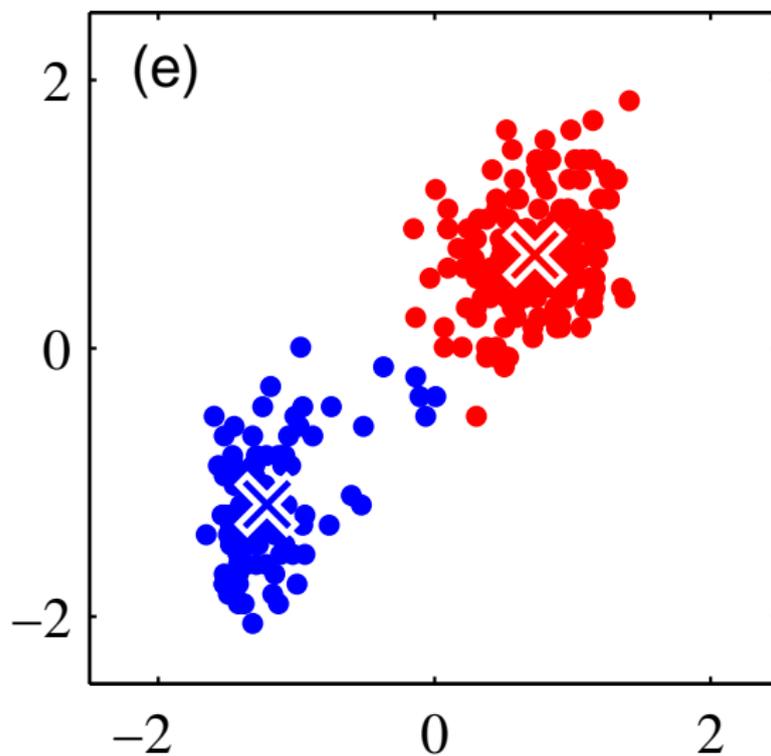
K-means example



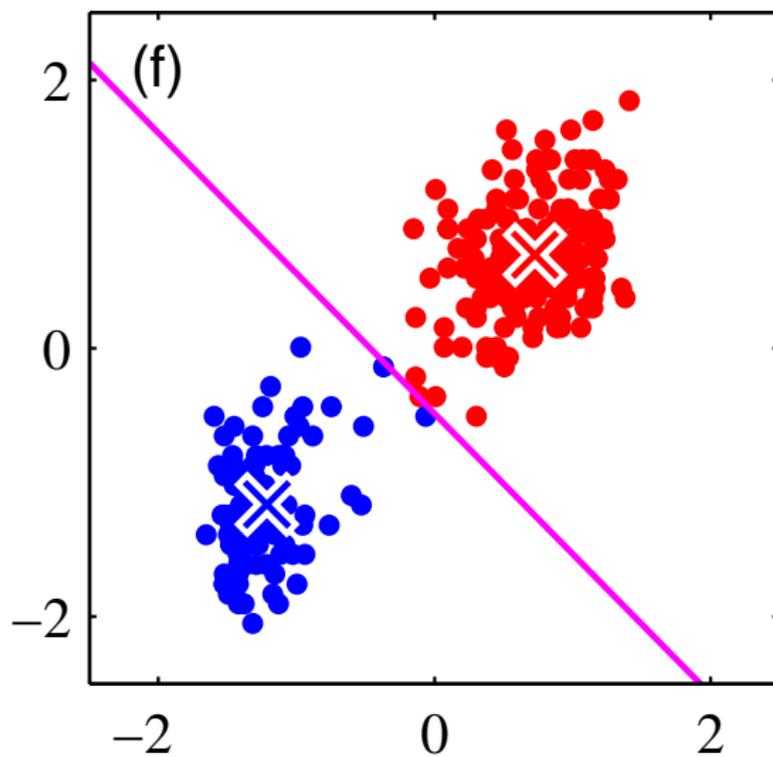
K-means example



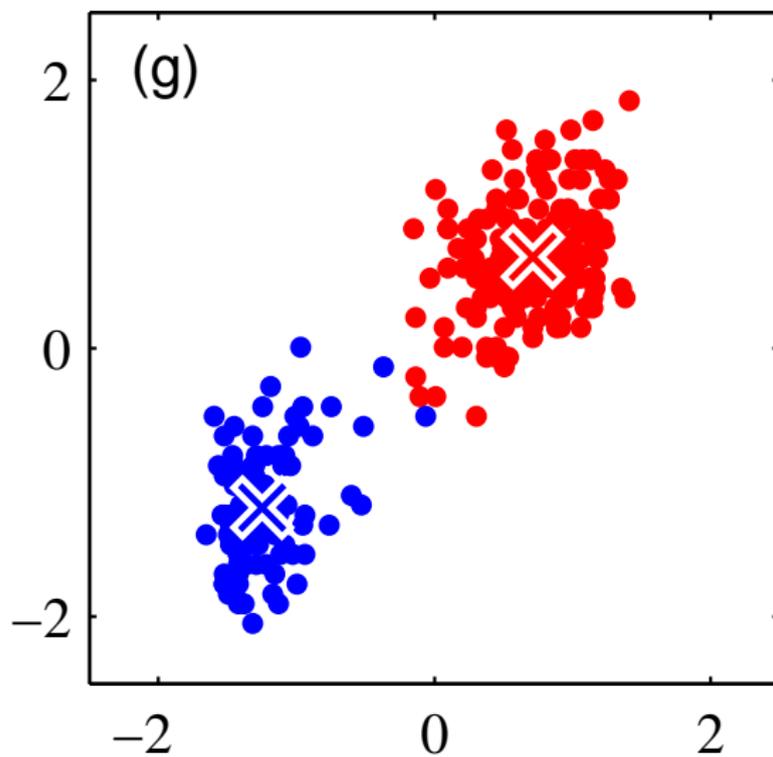
K-means example



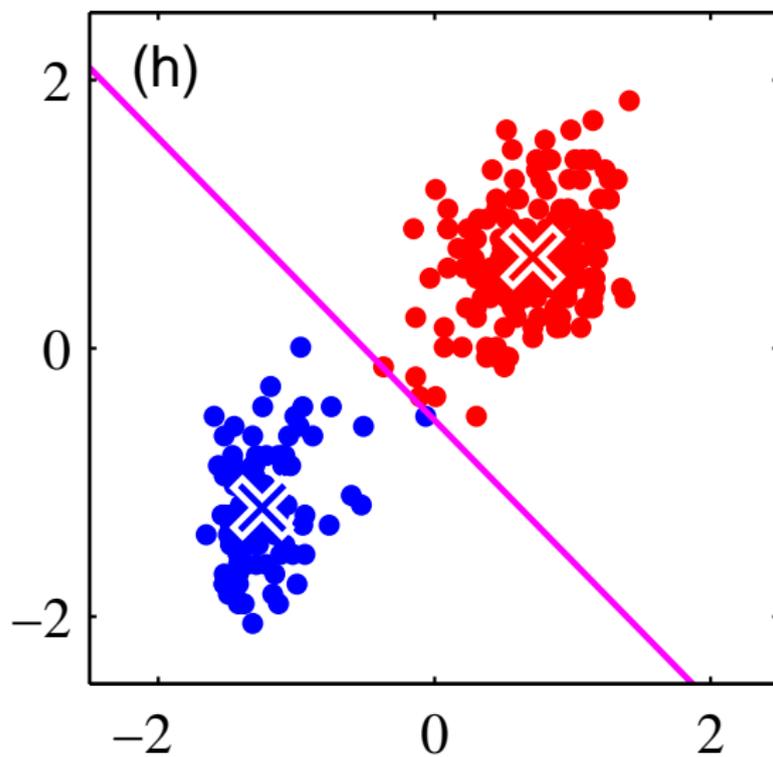
K-means example



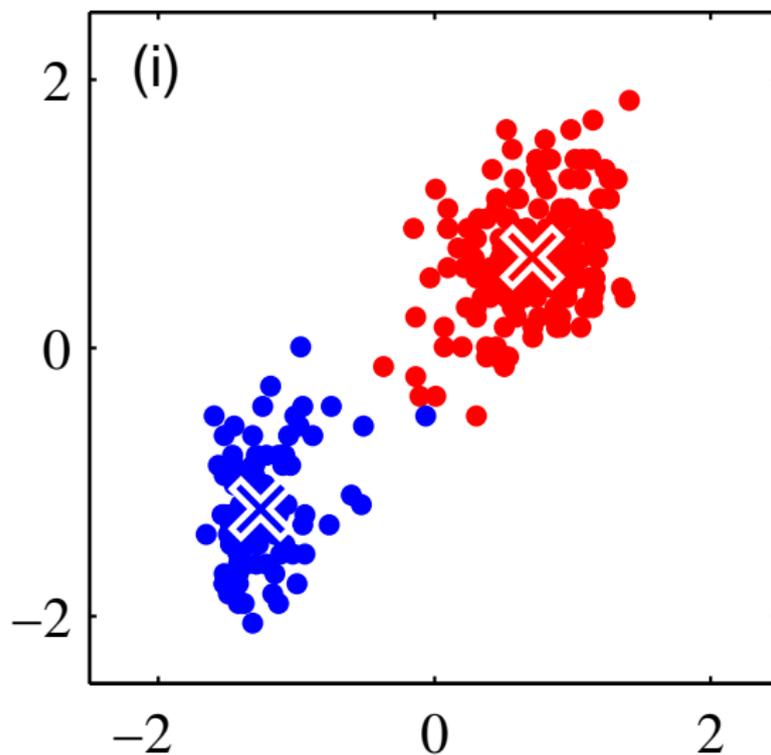
K-means example



K-means example



K-means example



Next step doesn't change membership – stop

K-means Convergence

- Repeat steps until no change in cluster assignments
- For each step, value of J either goes down, or we stop
- Finite number of possible assignments of data points to clusters, so we are guaranteed to converge eventually
- Note it may be a **local maximum** rather than a **global maximum** to which we converge

K-means Example - Image Segmentation



- K-means clustering on pixel colour values
- Pixels in a cluster are coloured by cluster mean
- Represent each pixel (e.g. 24-bit colour value) by a cluster number (e.g. 4 bits for $K = 10$), compressed version
- This technique known as **vector quantization**
 - Represent vector (in this case from RGB, \mathbb{R}^3) as a single discrete value

K-means Generalized: the set-up

Let's generalize the idea. Suppose we have the following set-up.

- X denotes all observed variables (e.g., data points).
- Z denotes all latent (hidden, unobserved) variables (e.g., cluster means).
- $J(X, Z|\theta)$ where J measures the “goodness” of an assignment of latent variable models given the data points and parameters θ .
 - e.g., $J =$ -dispersion measure.
 - parameters = assignment of points to clusters.
- It's easy to maximize $J(X, Z|\theta)$ wrt θ for **fixed** Z .
- It's easy to maximize $J(X, Z|\theta)$ wrt Z for **fixed** θ .

K-means Generalized: The Algorithm

The fact that conditional maximization is simple suggests an **iterative algorithm**.

1. Guess an initial value for latent variables Z .
2. Repeat until convergence:
 - 2.1 Find best parameter values θ given the current guess for the latent variables. Update the parameter values.
 - 2.2 Find best value for latent variables Z given the current parameter values. Update the latent variable values.

Outline

K-Means

The Expectation Maximization Algorithm

EM Example: Gaussian Mixture Models

EM Algorithm: The set-up

- We assume a probabilistic model, specifically the **complete-data likelihood function** $p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\theta})$.
- “Goodness” of the model is the log-likelihood $\ln p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\theta})$.
- Key difference: instead of guessing a single best value for latent variables *given current parameter settings*, we use the conditional *distribution* $p(\mathbf{Z}|\mathbf{X}, \boldsymbol{\theta}^{old})$ over latent variables.
- *Given a latent variable distribution*, parameter values $\boldsymbol{\theta}$ are evaluated by taking the expected “goodness” $\ln p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\theta})$ over all possible latent variable settings.

EM Algorithm: The procedure

1. Guess an initial parameter setting θ^{old} .
2. Repeat until convergence:
3. The **E-step**: Evaluate $p(\mathbf{Z}|\mathbf{X}, \theta^{old})$. (Ideally, find a closed form as a function of \mathbf{Z}).
4. The **M-step**:
 - 4.1 Evaluate the function
$$Q(\theta, \theta^{old}) = \sum_{\mathbf{Z}} p(\mathbf{Z}|\mathbf{X}, \theta^{old}) \ln p(\mathbf{X}, \mathbf{Z}|\theta).$$
 - 4.2 Maximize $Q(\theta, \theta^{old})$ wrt θ . Update θ^{old} .
5. This procedure is guaranteed to increase at each step. the data log-likelihood $\ln p(\mathbf{X}|\theta) = \sum_{\mathbf{Z}} \ln p(\mathbf{X}, \mathbf{Z}|\theta)$.
6. Therefore converges to local log-likelihood maximum.
More theoretical analysis in text.

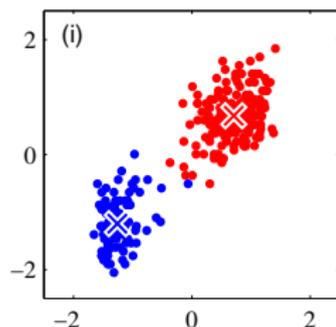
Outline

K-Means

The Expectation Maximization Algorithm

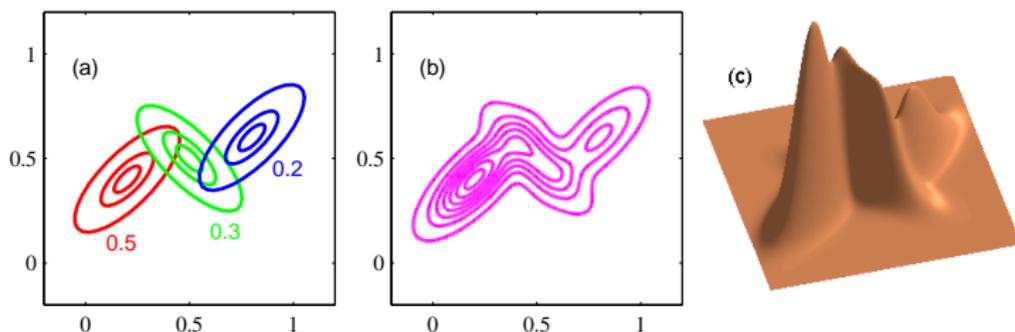
EM Example: Gaussian Mixture Models

Hard Assignment vs. Soft Assignment



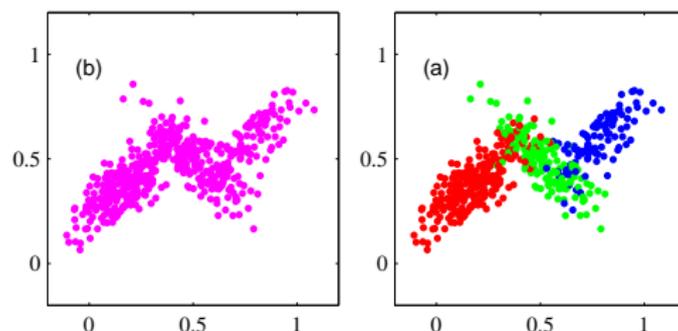
- In the K-means algorithm, a **hard assignment** of points to clusters is made
- However, for points near the decision boundary, this may not be such a good idea
- Instead, we could think about making a **soft assignment** of points to clusters

Gaussian Mixture Model



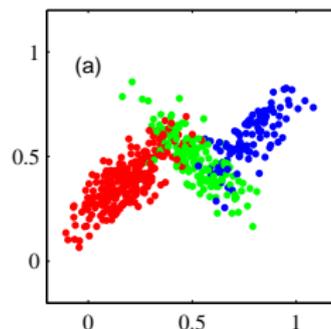
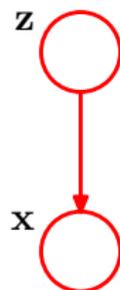
- The **Gaussian mixture model** (or **mixture of Gaussians** MoG) models the data as a combination of Gaussians.
- a: constant density contours. b: marginal probability $p(x)$. c: surface plot.
- Widely used general approximation for multi-modal distributions.

Gaussian Mixture Model



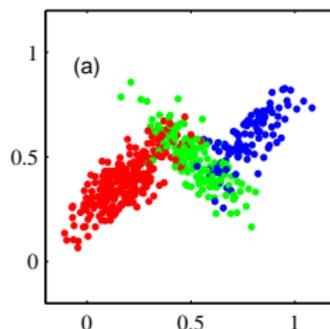
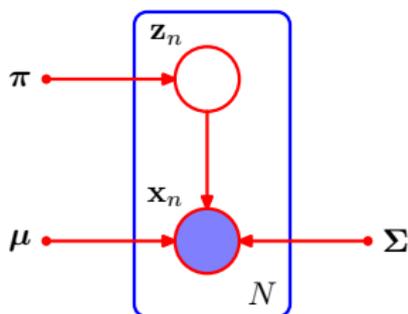
- Above shows a dataset generated by drawing samples from three different Gaussians

Generative Model



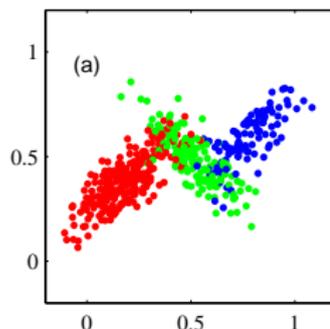
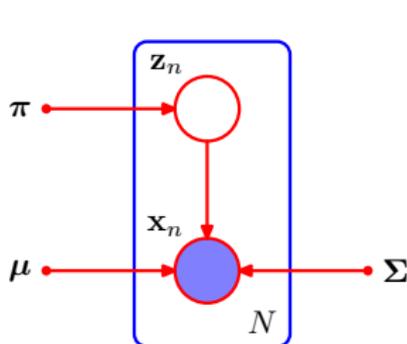
- The **mixture of Gaussians** is a **generative model**
- To generate a datapoint x_n , we first generate a value for a discrete variable $z_n \in \{1, \dots, K\}$
- We then generate a value $x_n \sim \mathcal{N}(x | \mu_k, \Sigma_k)$ for the corresponding Gaussian

Graphical Model



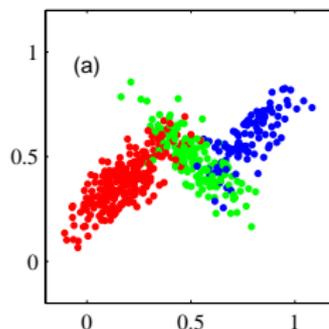
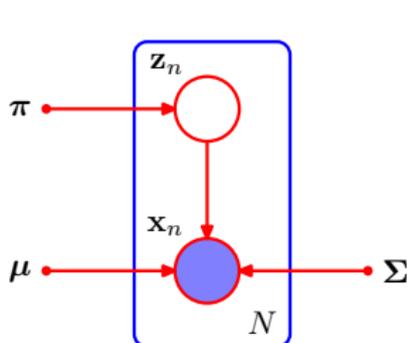
- Full graphical model using plate notation
 - Note z_n is a **latent variable**, unobserved
- Need to give conditional distributions $p(z_n)$ and $p(x_n|z_n)$
- The one-of- K representation is helpful here: $z_{nk} \in \{0, 1\}$,
 $z_n = (z_{n1}, \dots, z_{nK})$

Graphical Model - Latent Component Variable



- Use a **Bernoulli distribution** for $p(z_n)$
 - i.e. $p(z_{nk} = 1) = \pi_k$
 - Parameters to this distribution $\{\pi_k\}$
 - Must have $0 \leq \pi_k \leq 1$ and $\sum_{k=1}^K \pi_k = 1$
- $p(z_n) = \prod_{k=1}^K \pi_k^{z_{nk}}$

Graphical Model - Observed Variable

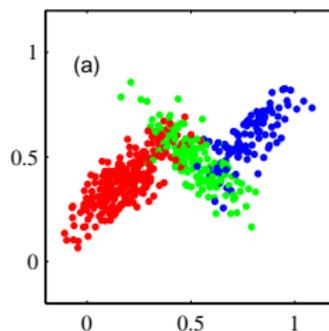
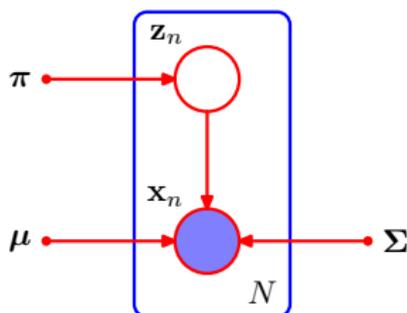


- Use a **Gaussian distribution** for $p(\mathbf{x}_n | z_n)$
 - Parameters to this distribution $\{\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}$

$$p(\mathbf{x}_n | z_{nk} = 1) = \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

$$p(\mathbf{x}_n | z_n) = \prod_{k=1}^K \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)^{z_{nk}}$$

Graphical Model - Joint distribution



- The full joint distribution is given by:

$$\begin{aligned}
 p(\mathbf{x}, \mathbf{z}) &= \prod_{n=1}^N p(z_n) p(\mathbf{x}_n | z_n) \\
 &= \prod_{n=1}^N \prod_{k=1}^K \pi_k^{z_{nk}} \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)^{z_{nk}}
 \end{aligned}$$

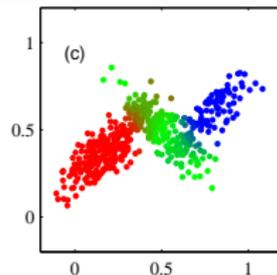
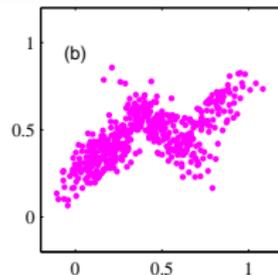
MoG Marginal over Observed Variables

- The marginal distribution $p(\mathbf{x}_n)$ for this model is:

$$\begin{aligned} p(\mathbf{x}_n) &= \sum_{\mathbf{z}_n} p(\mathbf{x}_n, \mathbf{z}_n) = \sum_{\mathbf{z}_n} p(\mathbf{z}_n) p(\mathbf{x}_n | \mathbf{z}_n) \\ &= \sum_{k=1}^K \pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \end{aligned}$$

- A **mixture** of Gaussians

MoG Conditional over Latent Variable

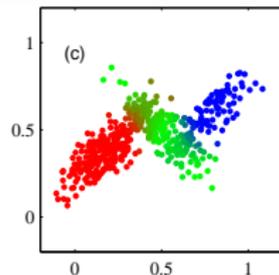
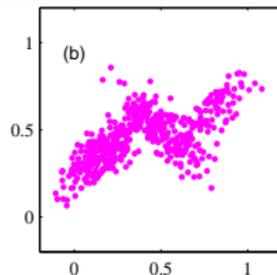


- To apply EM, need the conditional distribution $p(z_{nk} = 1 | \mathbf{x}_n, \theta)$ where θ are the model parameters.
- It is denoted by $\gamma(z_{nk})$ can be computed as:

$$\begin{aligned} \gamma(z_{nk}) \equiv p(z_{nk} = 1 | \mathbf{x}_n) &= \frac{p(z_{nk} = 1)p(\mathbf{x}_n | z_{nk} = 1)}{\sum_{j=1}^K p(z_{nj} = 1)p(\mathbf{x}_n | z_{nj} = 1)} \\ &= \frac{\pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^K \pi_j \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} \end{aligned}$$

- $\gamma(z_{nk})$ is the **responsibility** of component k for datapoint n

MoG Conditional over Latent Variable

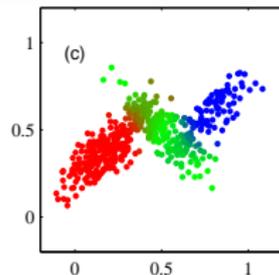
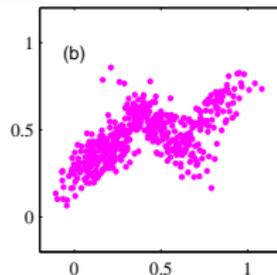


- To apply EM, need the conditional distribution $p(z_{nk} = 1 | \mathbf{x}_n, \theta)$ where θ are the model parameters.
- It is denoted by $\gamma(z_{nk})$ can be computed as:

$$\begin{aligned} \gamma(z_{nk}) \equiv p(z_{nk} = 1 | \mathbf{x}_n) &= \frac{p(z_{nk} = 1)p(\mathbf{x}_n | z_{nk} = 1)}{\sum_{j=1}^K p(z_{nj} = 1)p(\mathbf{x}_n | z_{nj} = 1)} \\ &= \frac{\pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^K \pi_j \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} \end{aligned}$$

- $\gamma(z_{nk})$ is the **responsibility** of component k for datapoint n

MoG Conditional over Latent Variable



- To apply EM, need the conditional distribution $p(z_{nk} = 1 | \mathbf{x}_n, \theta)$ where θ are the model parameters.
- It is denoted by $\gamma(z_{nk})$ can be computed as:

$$\begin{aligned} \gamma(z_{nk}) \equiv p(z_{nk} = 1 | \mathbf{x}_n) &= \frac{p(z_{nk} = 1)p(\mathbf{x}_n | z_{nk} = 1)}{\sum_{j=1}^K p(z_{nj} = 1)p(\mathbf{x}_n | z_{nj} = 1)} \\ &= \frac{\pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^K \pi_j \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} \end{aligned}$$

- $\gamma(z_{nk})$ is the **responsibility** of component k for datapoint n

EM for Gaussian Mixtures: E-step

- The complete-data log-likelihood is

$$\ln p(\mathbf{X}, \mathbf{Z} | \boldsymbol{\theta}) = \sum_{n=1}^N \sum_{k=1}^K z_{nk} [\ln \pi_k + \ln \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)].$$

- E step:** Calculate responsibilities using current parameters $\boldsymbol{\theta}^{old}$:

$$\gamma(z_{nk}) = \frac{\pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^K \pi_j \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}$$

- The z_n vectors assigning data point n to components are independent of each other.
- Therefore under the posterior distribution $p(z_{nk} = 1 | \mathbf{x}_n, \boldsymbol{\theta})$ the expected value of z_{nk} is $\gamma(z_{nk})$.

EM for Gaussian Mixtures: M-step

- **M step:** Because of the independence of the component assignments, we can calculate the expectation wrt the component assignments by using the expectations of the component assignments.
- So $Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{old}) = \sum_{n=1}^N \sum_{k=1}^K \gamma(z_{nk}) [\ln \pi_k + \ln \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)]$.
- Maximizing $Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{old})$ with respect to the model parameters is more or less straightforward.

EM for Gaussian Mixtures II

- Initialize parameters, then iterate:
 - E step:** Calculate responsibilities using current parameters

$$\gamma(z_{nk}) = \frac{\pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^K \pi_j \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}$$

- M step:** Re-estimate parameters using these $\gamma(z_{nk})$

$$N_k = \sum_{n=1}^N \gamma(z_{nk})$$

$$\boldsymbol{\mu}_k = \frac{1}{N_k} \sum_{n=1}^N \gamma(z_{nk}) \mathbf{x}_n$$

$$\boldsymbol{\Sigma}_k = \frac{1}{N_k} \sum_{n=1}^N \gamma(z_{nk}) (\mathbf{x}_n - \boldsymbol{\mu}_k)(\mathbf{x}_n - \boldsymbol{\mu}_k)^T$$

$$\pi_k = \frac{N_k}{N}$$

- Think of N_k as effective number of points in component k .

EM for Gaussian Mixtures II

- Initialize parameters, then iterate:
 - E step:** Calculate responsibilities using current parameters

$$\gamma(z_{nk}) = \frac{\pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^K \pi_j \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}$$

- M step:** Re-estimate parameters using these $\gamma(z_{nk})$

$$N_k = \sum_{n=1}^N \gamma(z_{nk})$$

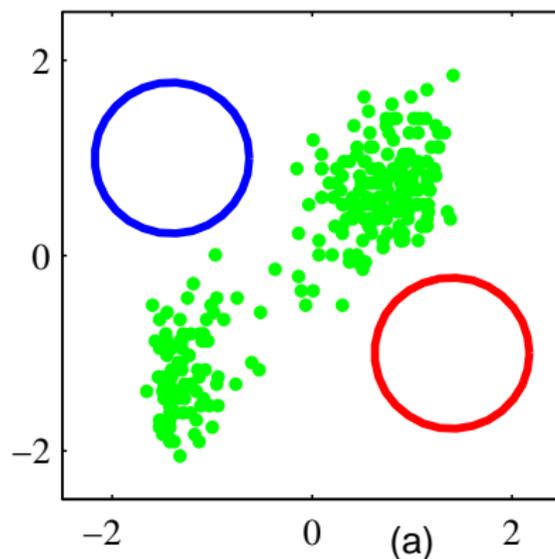
$$\boldsymbol{\mu}_k = \frac{1}{N_k} \sum_{n=1}^N \gamma(z_{nk}) \mathbf{x}_n$$

$$\boldsymbol{\Sigma}_k = \frac{1}{N_k} \sum_{n=1}^N \gamma(z_{nk}) (\mathbf{x}_n - \boldsymbol{\mu}_k)(\mathbf{x}_n - \boldsymbol{\mu}_k)^T$$

$$\pi_k = \frac{N_k}{N}$$

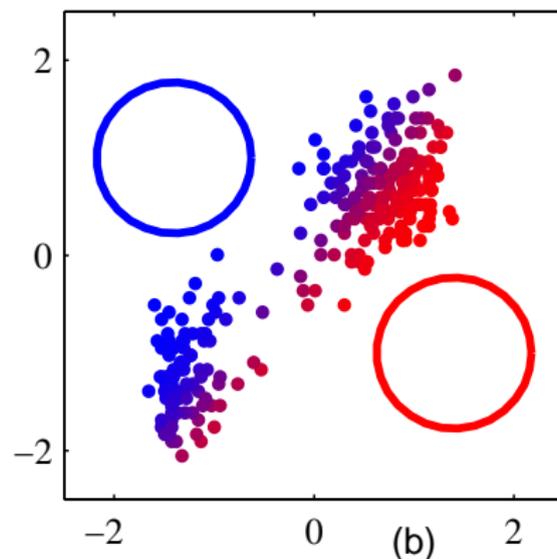
- Think of N_k as effective number of points in component k .

MoG EM - Example



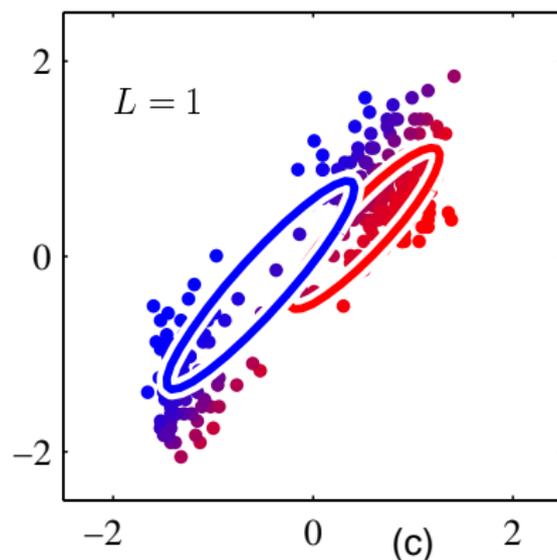
- Same initialization as with K-means before
 - Often, K-means is actually used to initialize EM

MoG EM - Example



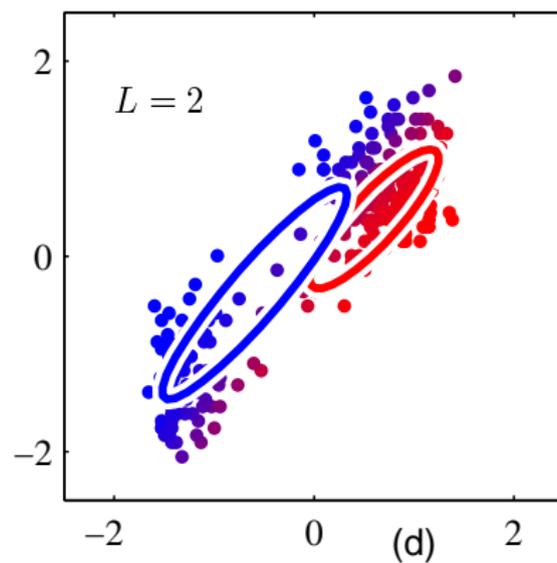
- Calculate responsibilities $\gamma(z_{nk})$

MoG EM - Example



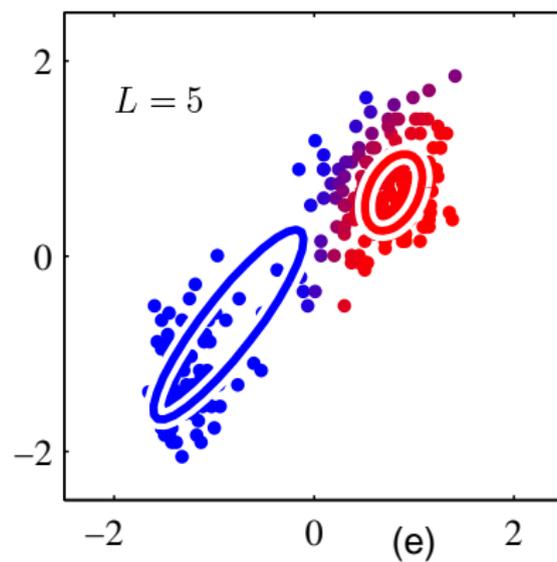
- Calculate model parameters $\{\pi_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}$ using these responsibilities

MoG EM - Example



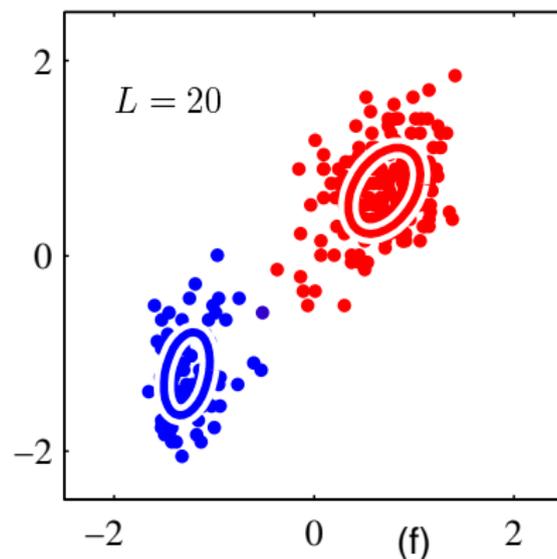
- Iteration 2

MoG EM - Example



- Iteration 5

MoG EM - Example



- Iteration 20 - converged

EM - Summary

- EM finds local maximum to likelihood

$$p(\mathbf{X}|\boldsymbol{\theta}) = \sum_{\mathbf{Z}} p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\theta})$$

- Iterates two steps:
 - **E step** “fills in” the missing variables \mathbf{Z} (calculates their distribution)
 - **M step** maximizes expected complete log likelihood (expectation wrt **E step** distribution)

Conclusion

- Readings: Ch. 9.1, 9.2, 9.4
- K-means clustering
- Gaussian mixture model
- What about K?
 - Model selection: either cross-validation or Bayesian version (average over all values for K)
- Expectation-maximization, a general method for learning parameters of models when not all variables are observed