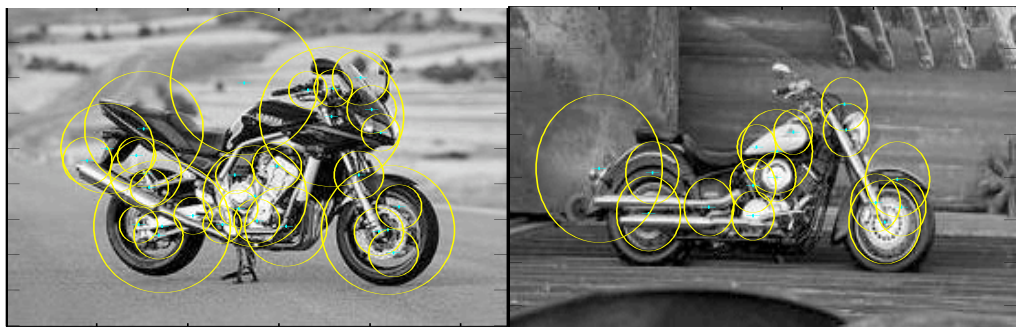
Latent Variable Models and Expectation Maximization

Oliver Schulte - CMPT 726

Bishop PRML Ch. 9

Learning Parameters to Probability Distributions

- We discussed probabilistic models at length
- Given fully observed training data, setting parameters $\boldsymbol{\theta}_i$ for Bayes nets is straight-forward
- However, in many settings not all variables are observed (labelled) in the training data: $\boldsymbol{x}_i = (\boldsymbol{x}_i, \boldsymbol{h}_i)$
  - 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 Variables and Simplicity

- Latent variables can explain observed correlations with a simple model.
  - Fewer parameters.
  - Common in science: The heart, genes, energy, gravity, ...

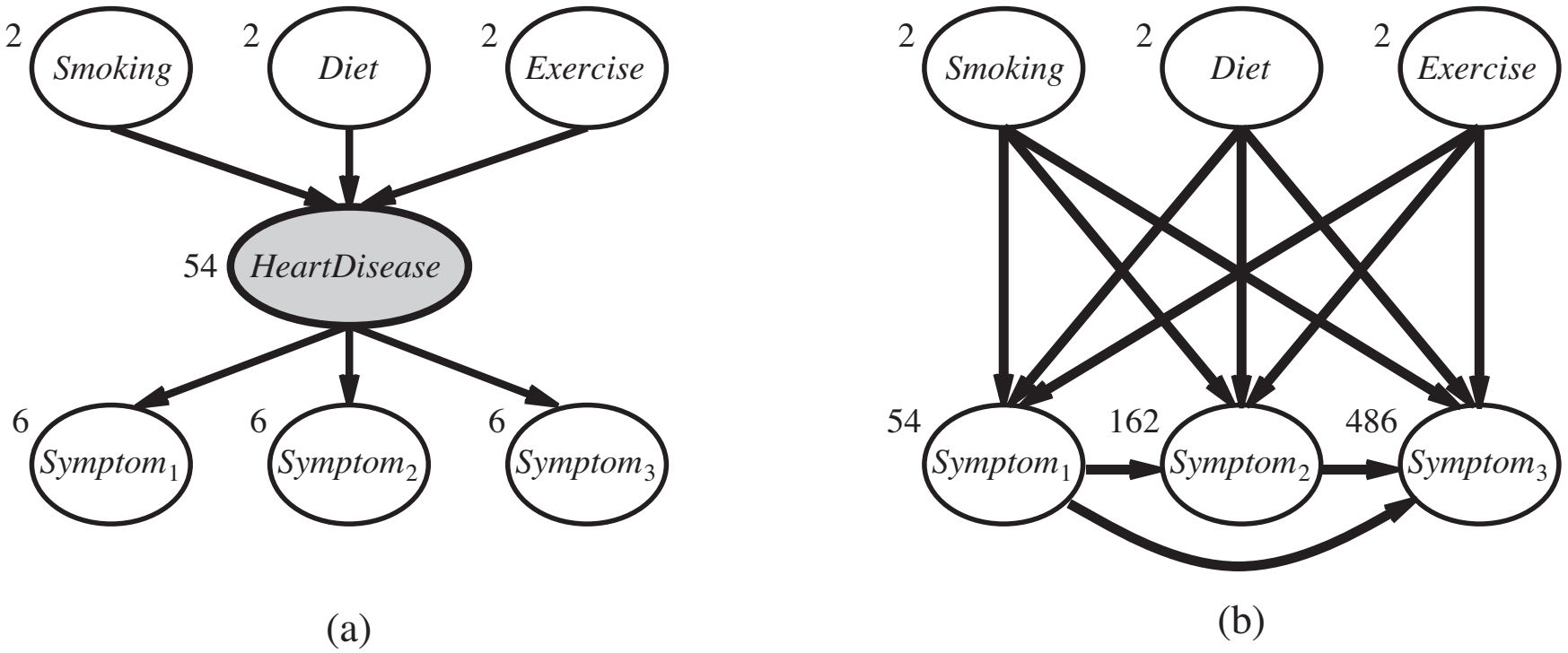


Fig. Russell and Norvig 20.10

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**.

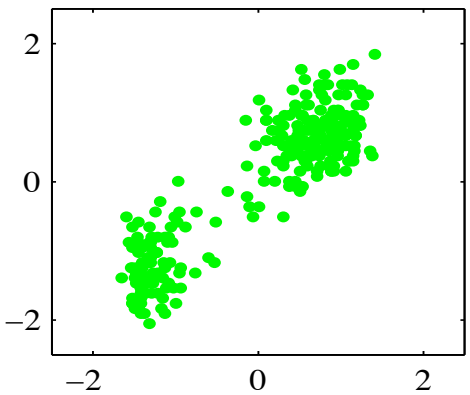
Latent Variable Models: Cons

- 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

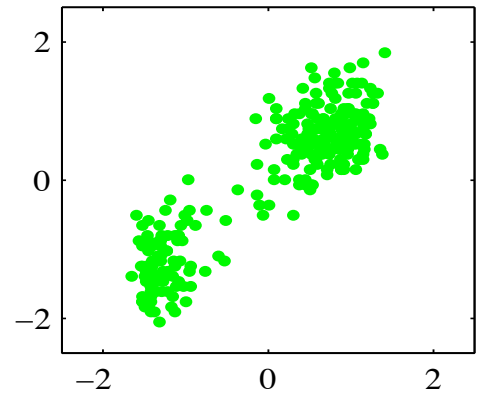
Outline

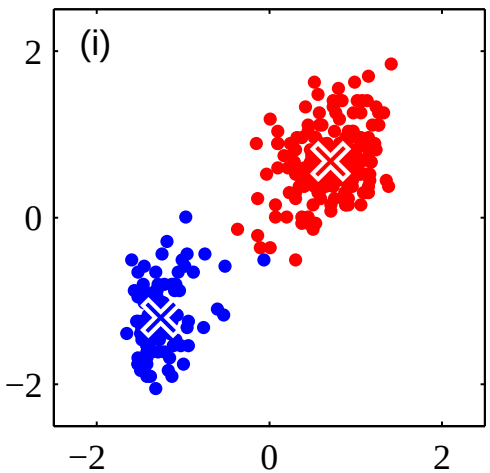
Unsupervised Learning



- We will start with an unsupervised learning (clustering) problem:
- Given a dataset $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$, each $\boldsymbol{x}_i \in \mathbb{R}^D$, 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:

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

Minimizing Distortion Measure

- Minimizing J directly is hard

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

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

Minimizing Distortion Measure

- Minimizing J directly is hard

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

- 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 chicken-and-egg scenario 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

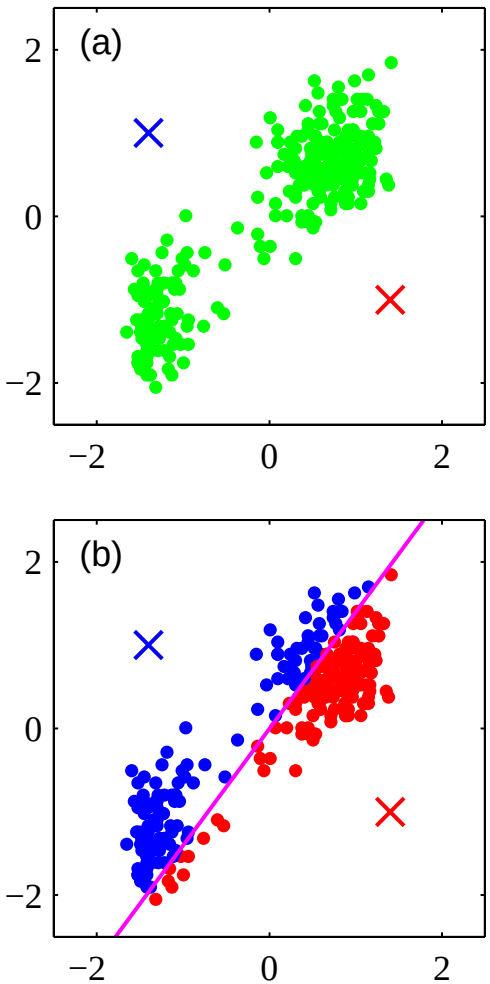
Minimizing Distortion Measure

- Minimizing J directly is hard

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

- 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 chicken-and-egg scenario 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}

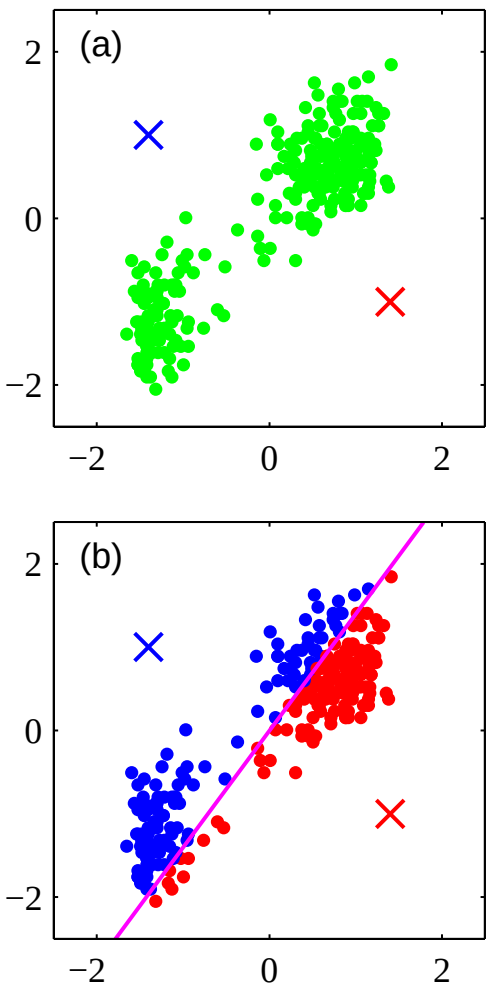
$$J = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} ||\boldsymbol{x}_n - \boldsymbol{\mu}_k||^2$$

- Terms for different data points x_n are independent, for each data point set r_{nk} to minimize:

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

- Simply set $r_{nk} = 1$ for the cluster center μ_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}

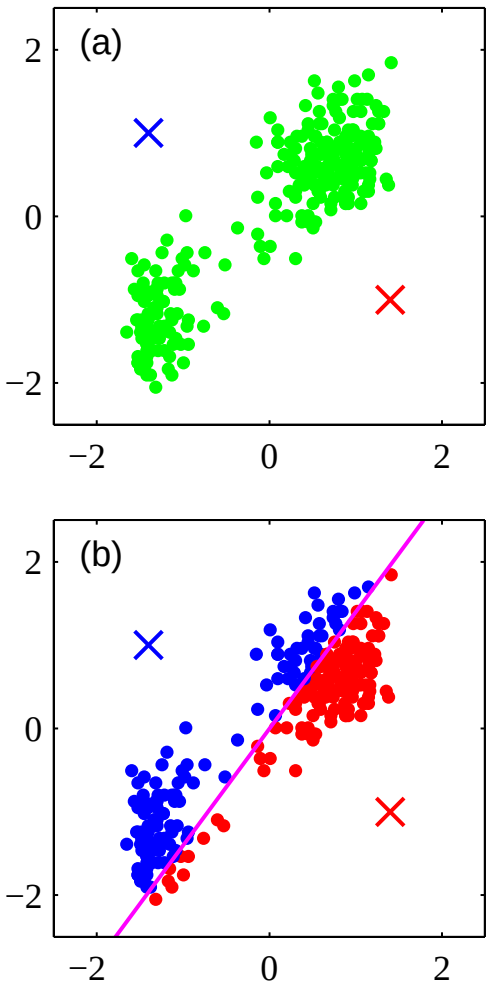
$$J = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} ||\boldsymbol{x}_n - \boldsymbol{\mu}_k||^2$$

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

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

- Simply set $r_{nk} = 1$ for the cluster center μ_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}

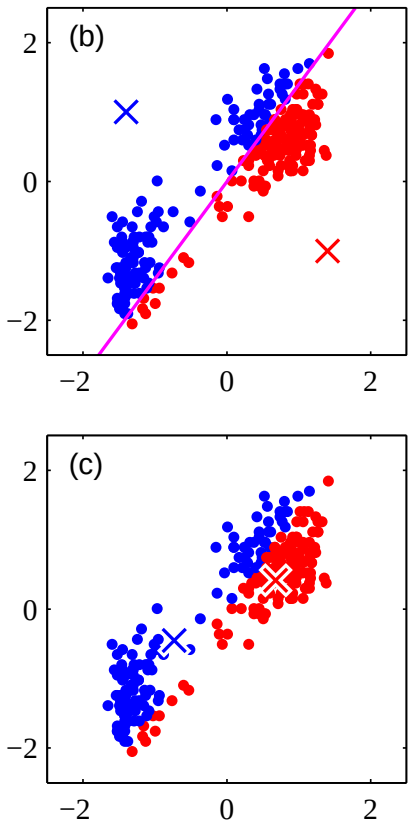
$$J = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} ||\boldsymbol{x}_n - \boldsymbol{\mu}_k||^2$$

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

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

- 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 $\boldsymbol{\mu}_k$

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

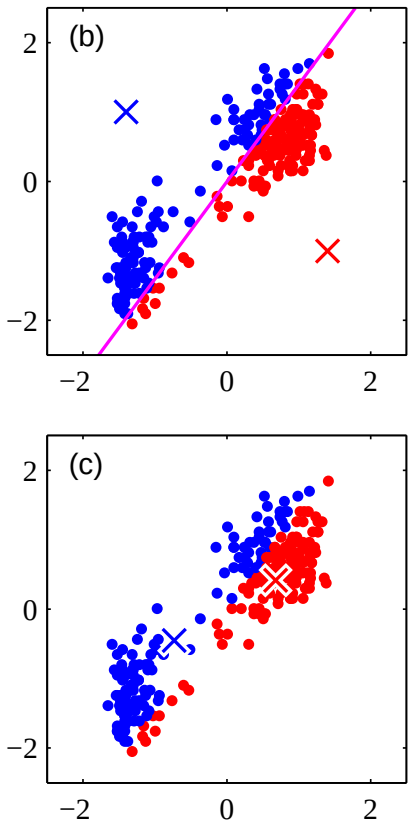
- Minimize wrt each $\boldsymbol{\mu}_k$ separately (how?)
- Take derivative, set to zero:

$$2 \sum_{n=1}^{N} r_{nk} (\boldsymbol{x}_n - \boldsymbol{\mu}_k) = 0$$

$$\Leftrightarrow \boldsymbol{\mu}_k = \frac{\sum_n r_{nk} \boldsymbol{x}_n}{\sum_n r_{nk}}$$

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

Determining Cluster Centers



- Step 2: fix r_{nk}, minimize J wrt the cluster centers $\boldsymbol{\mu}_k$

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

- Minimize wrt each $\boldsymbol{\mu}_k$ separately (how?)
- Take derivative, set to zero:

$$2 \sum_{n=1}^{N} r_{nk} (\boldsymbol{x}_n - \boldsymbol{\mu}_k) = 0$$

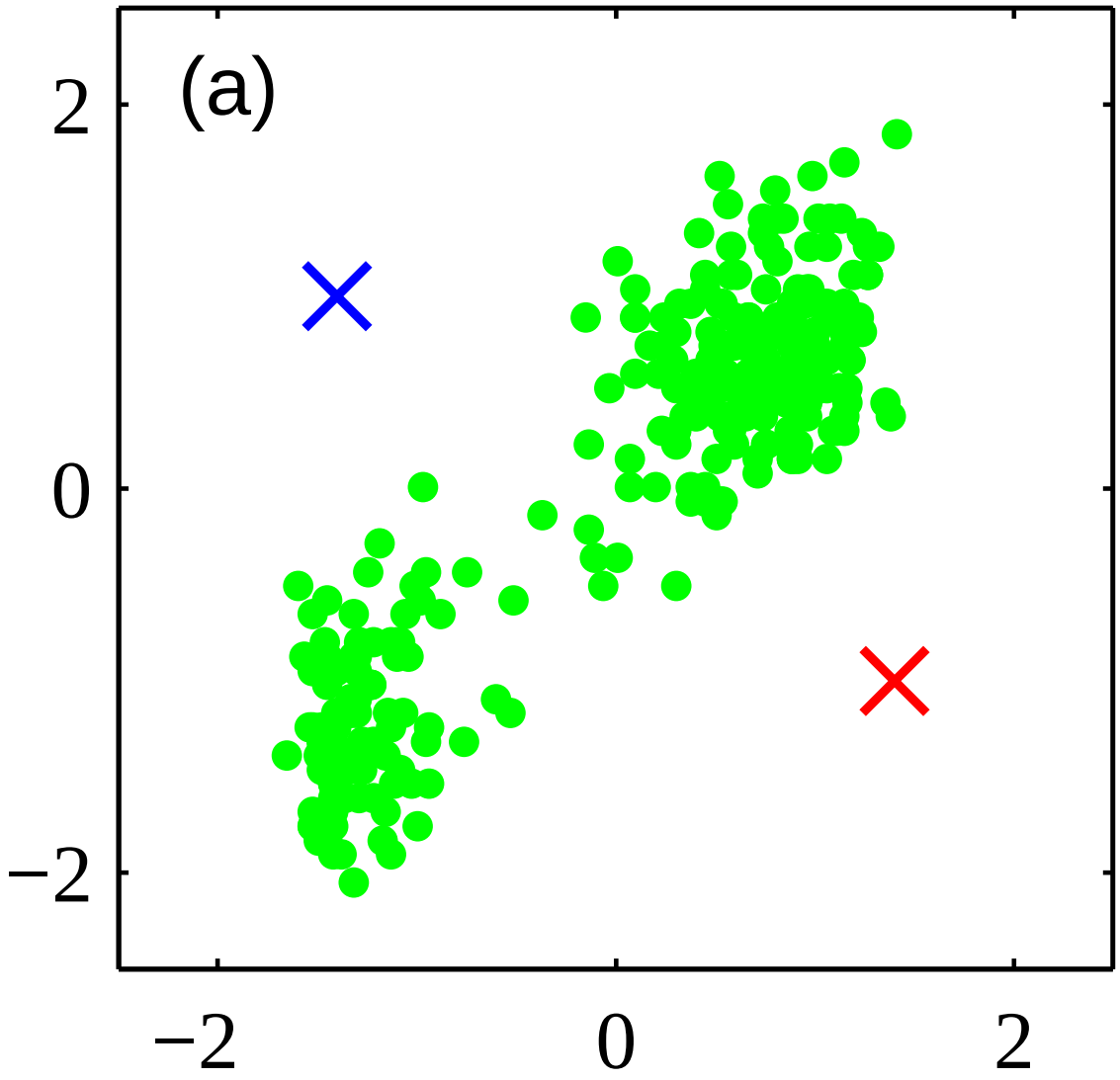
$$\Leftrightarrow \boldsymbol{\mu}_k = \frac{\sum_n r_{nk} \boldsymbol{x}_n}{\sum_n r_{nk}}$$

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

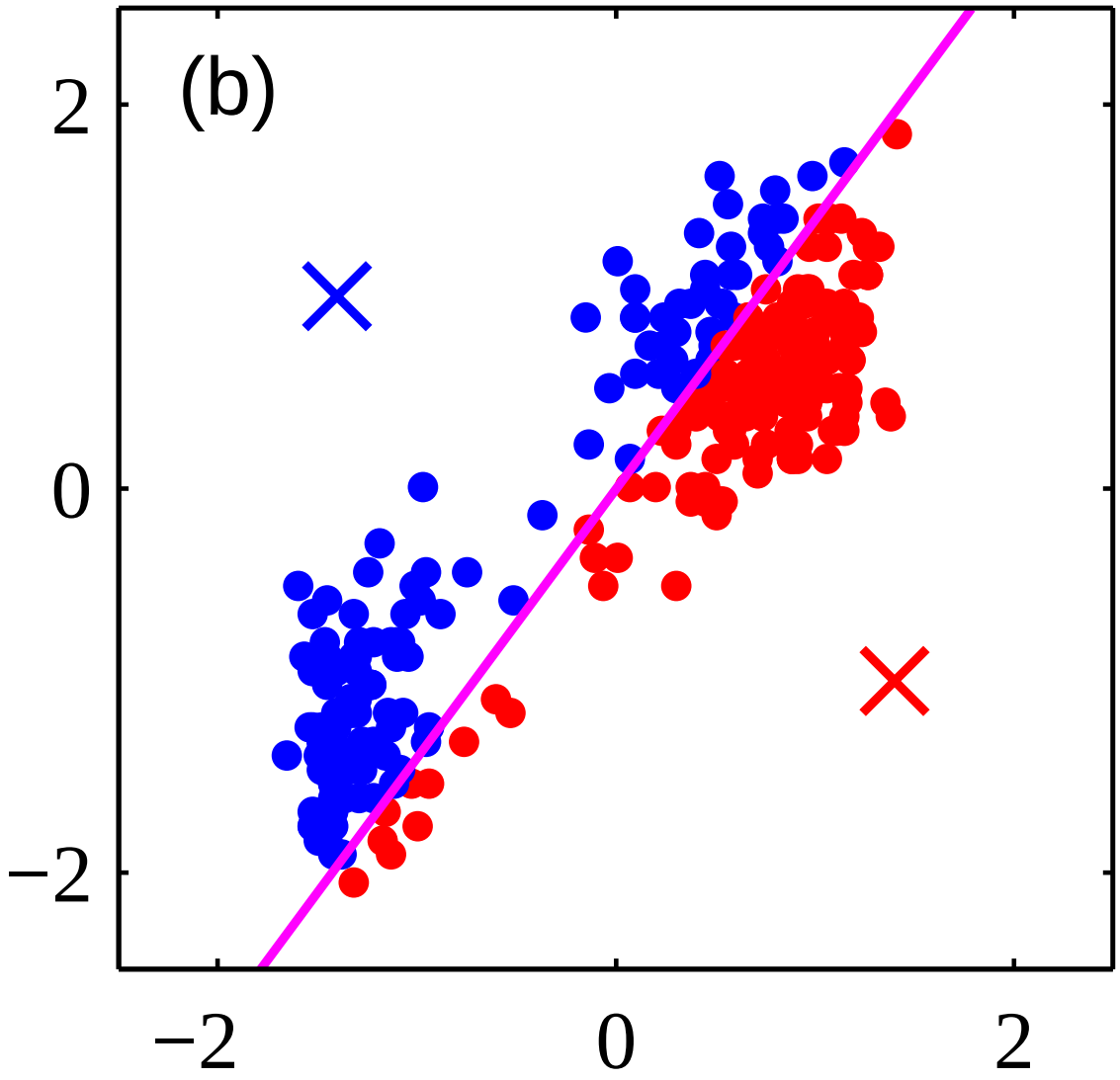
K-means Algorithm

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

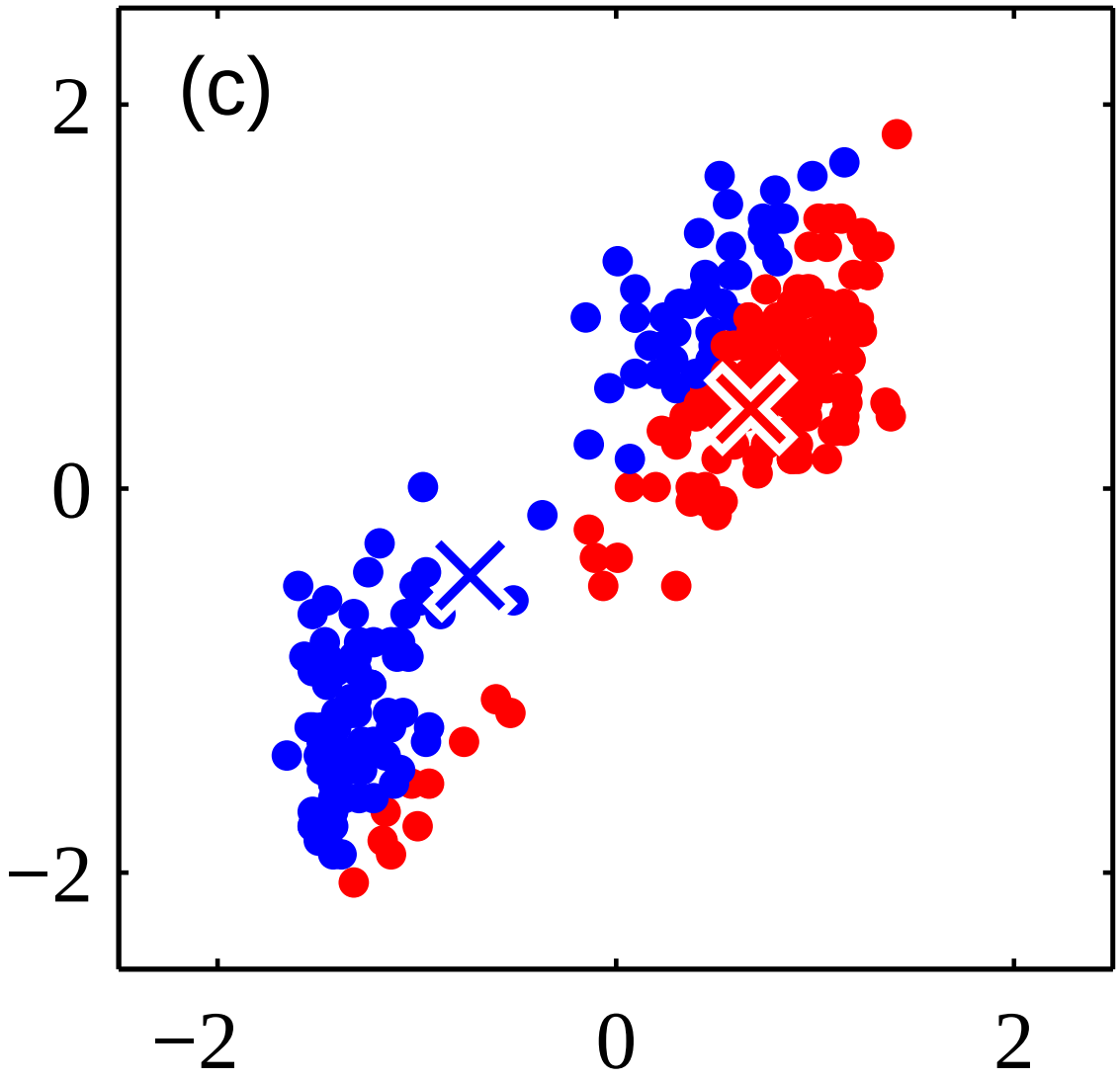
K-means example



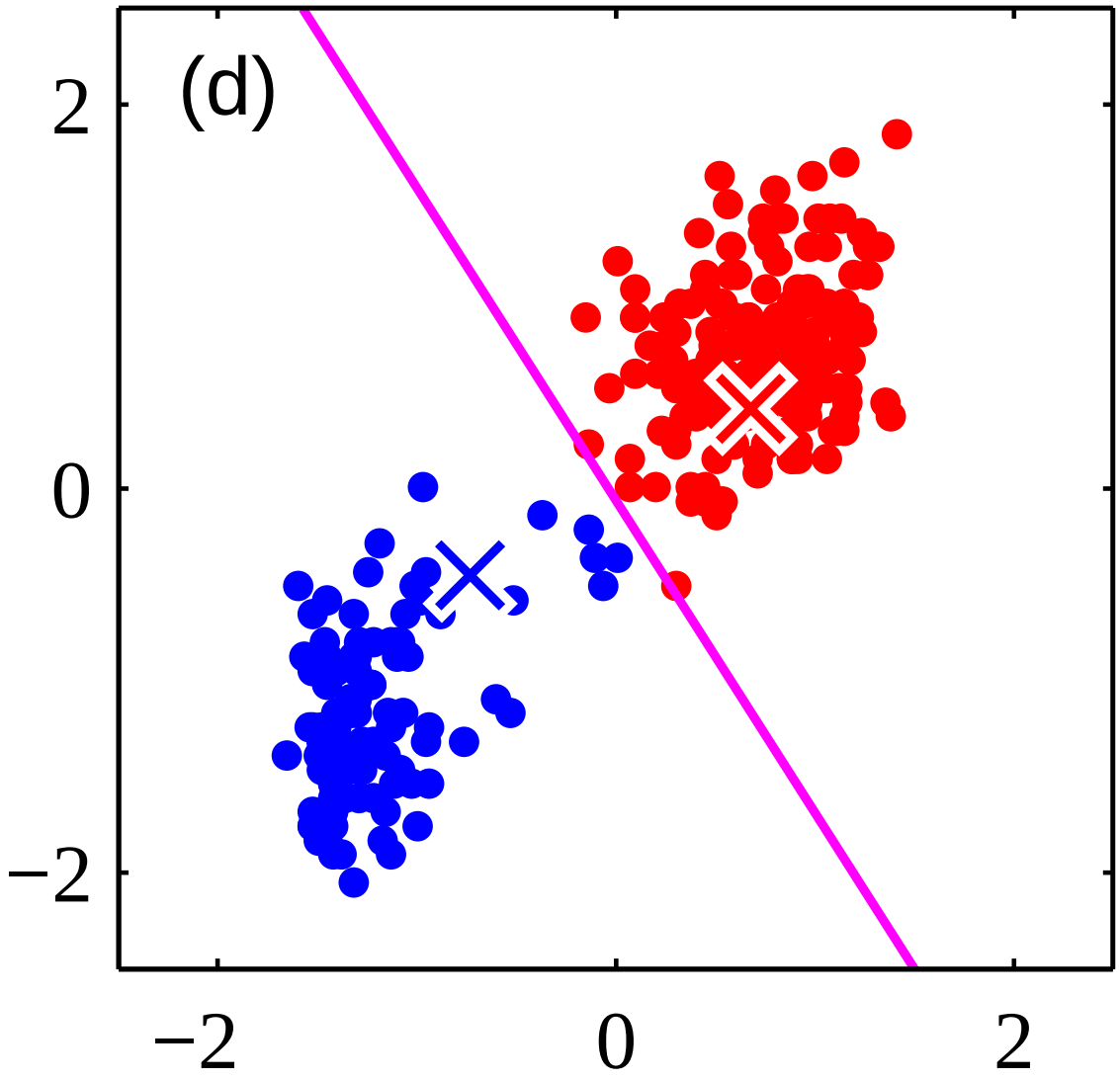
K-means example



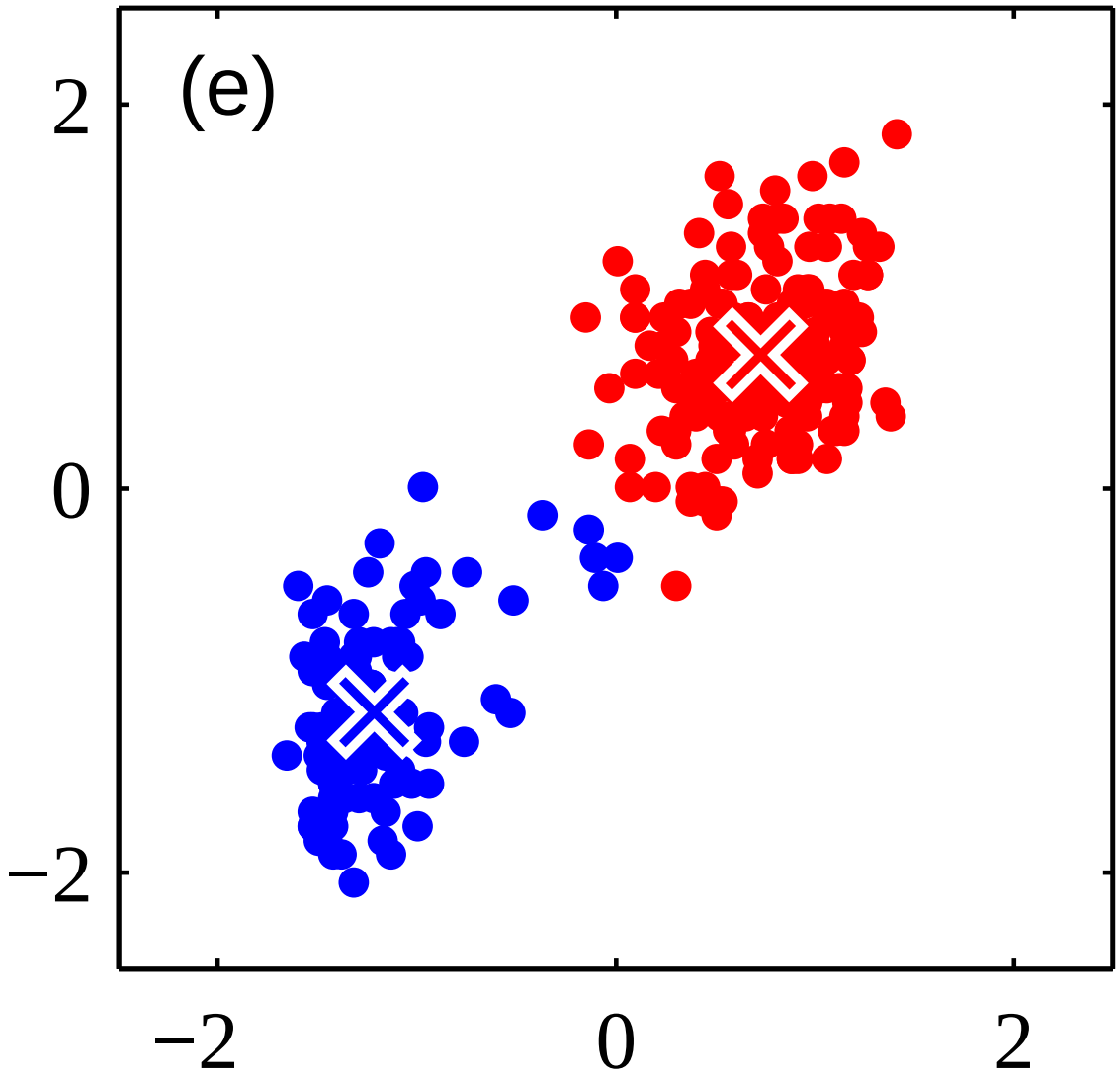
K-means example



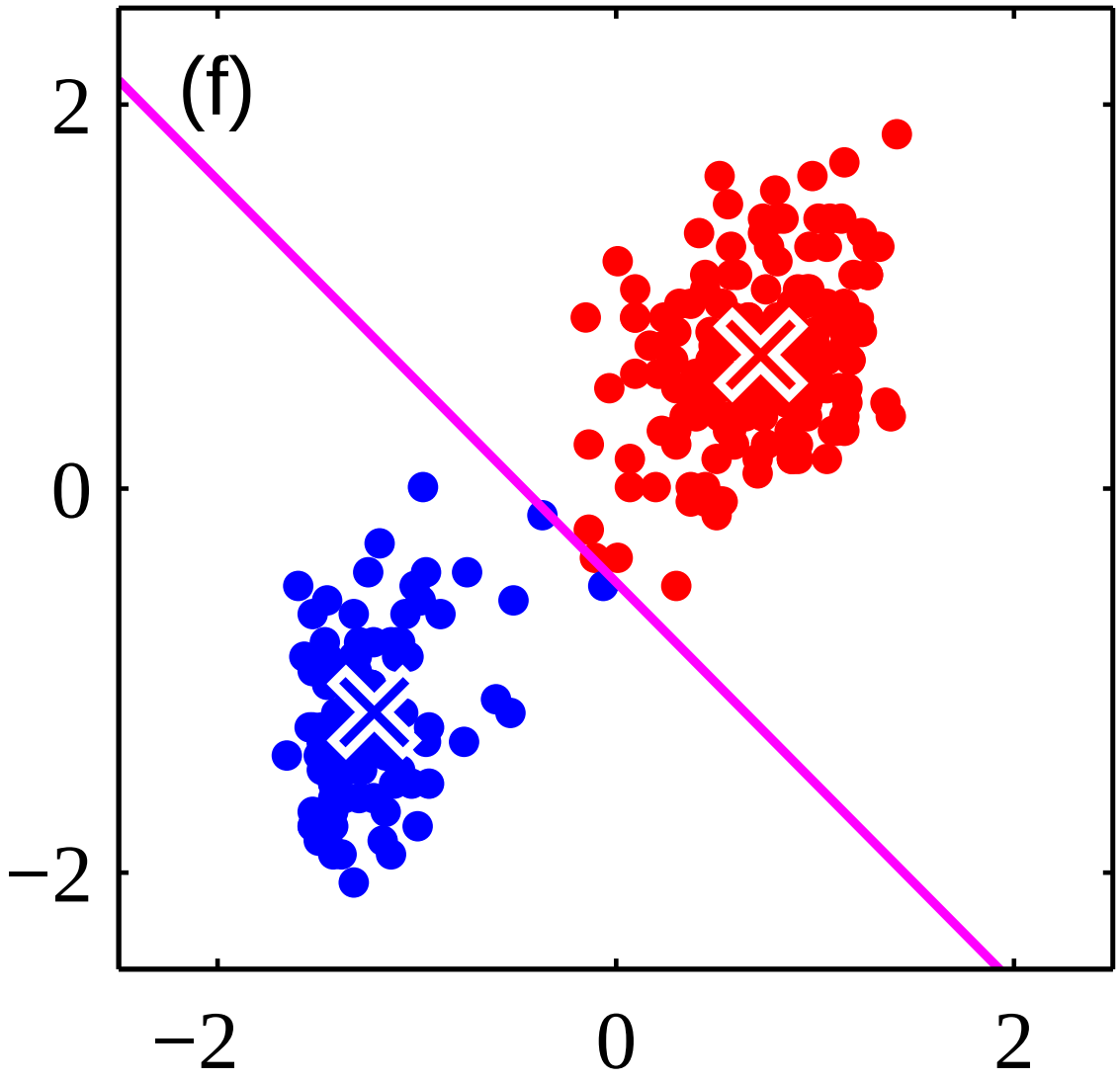
K-means example



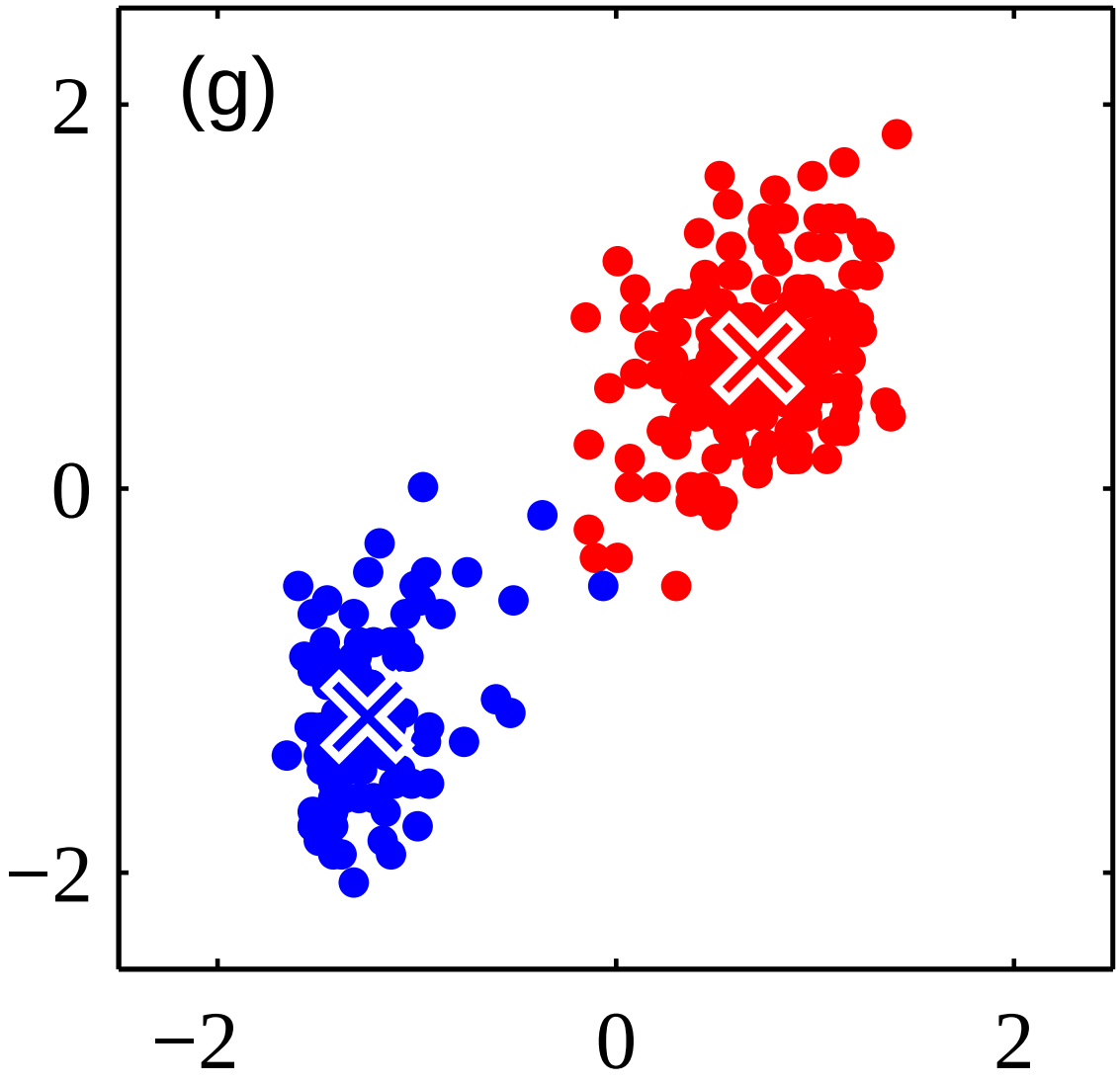
K-means example



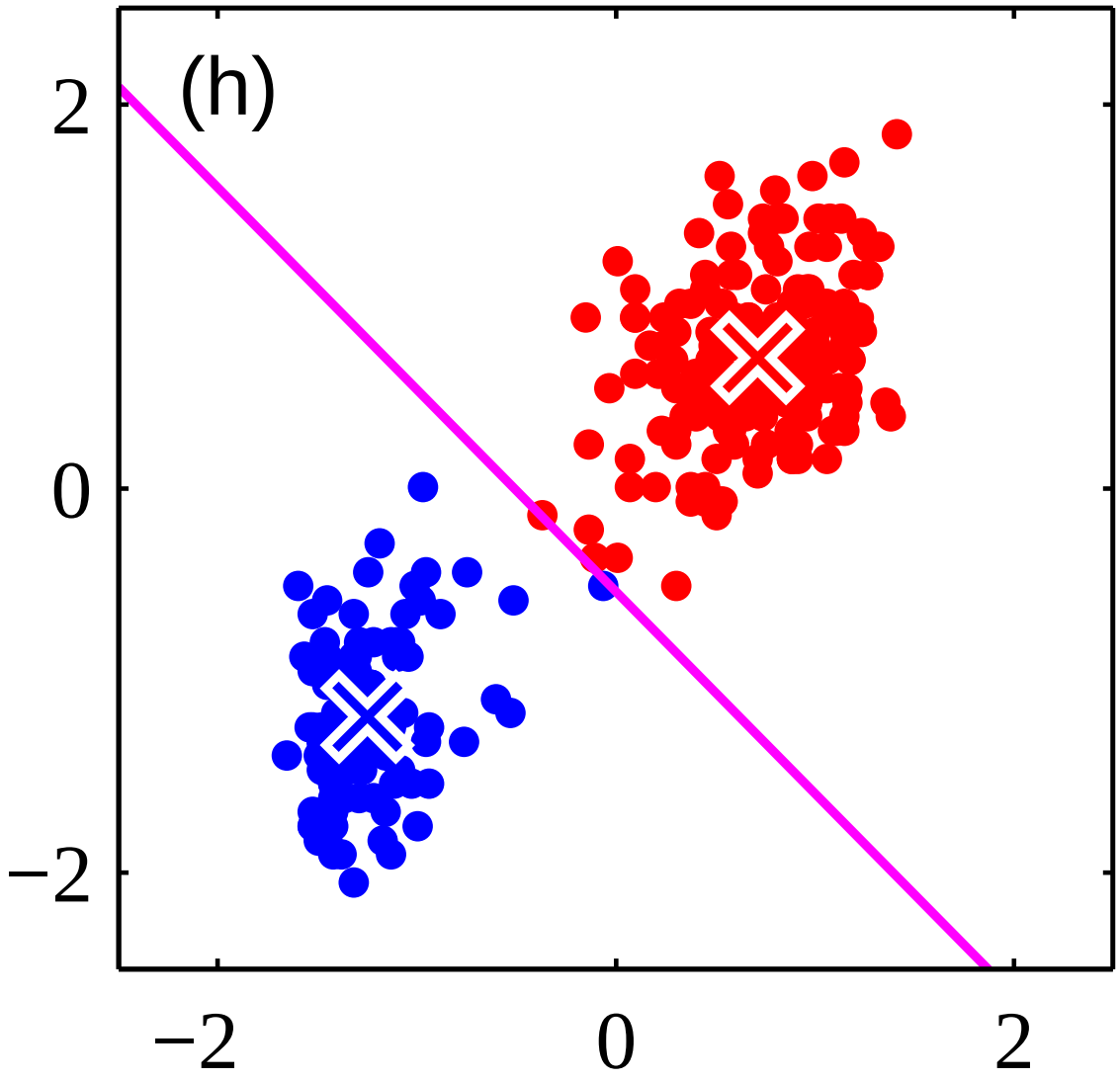
K-means example



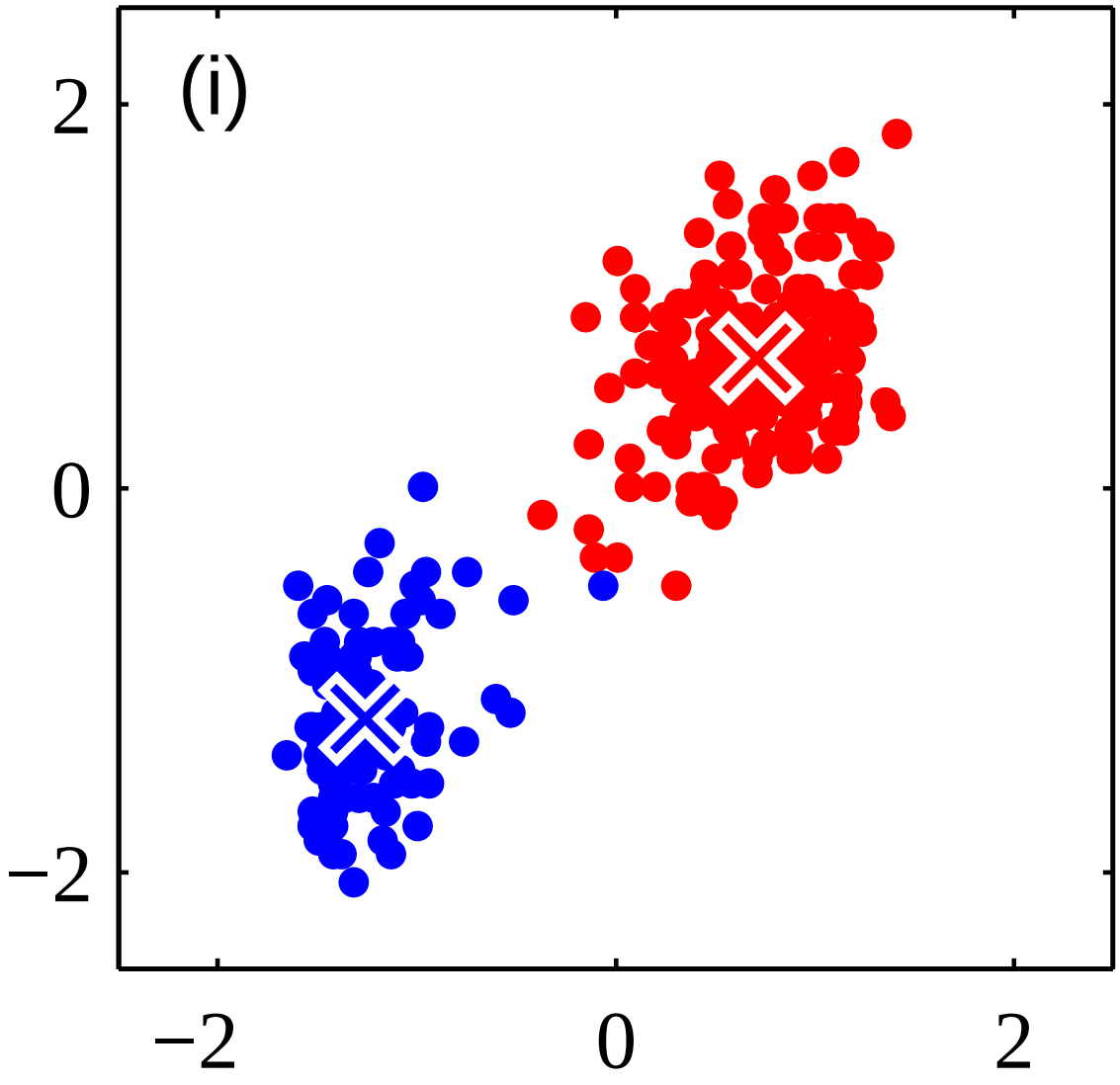
K-means example



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

$K = 2$

$K = 3$

$K = 10$

Original image

- 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

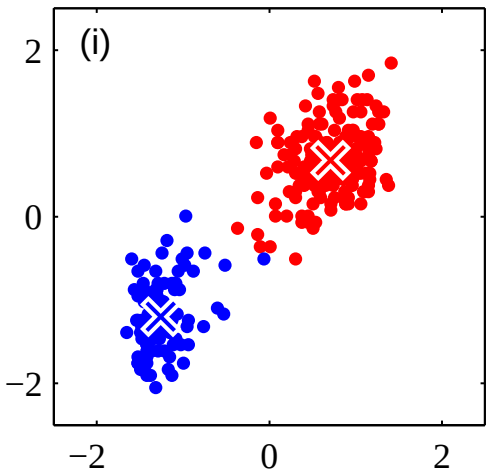
K-means Generalized: the set-up

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

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

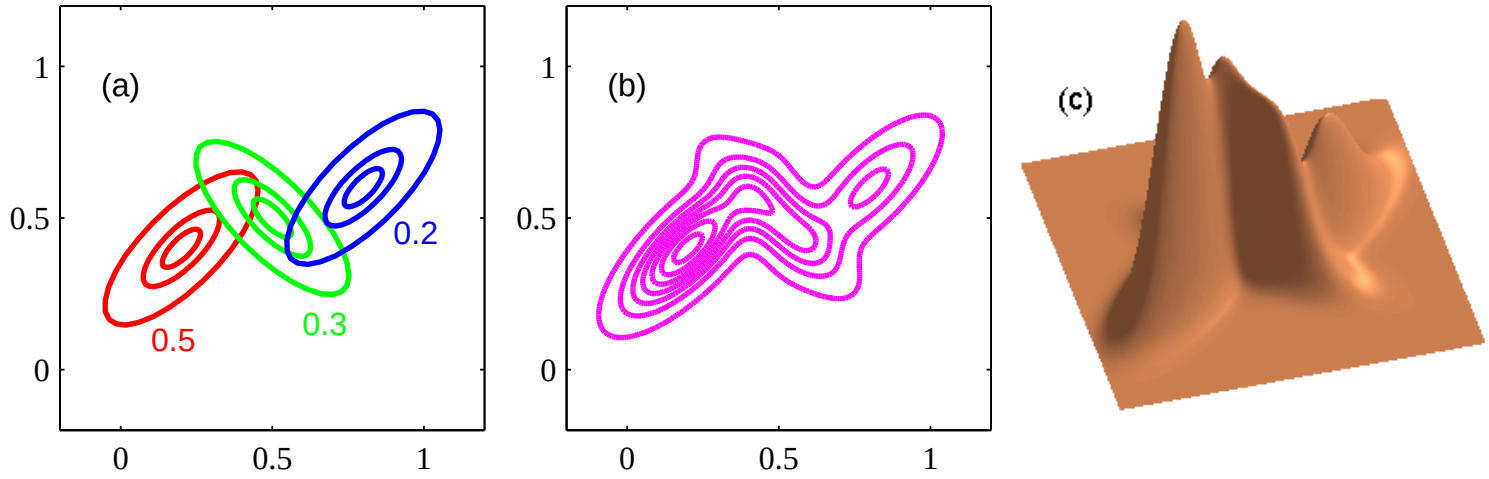
Outline

Hard Assignment vs. Soft Assignment



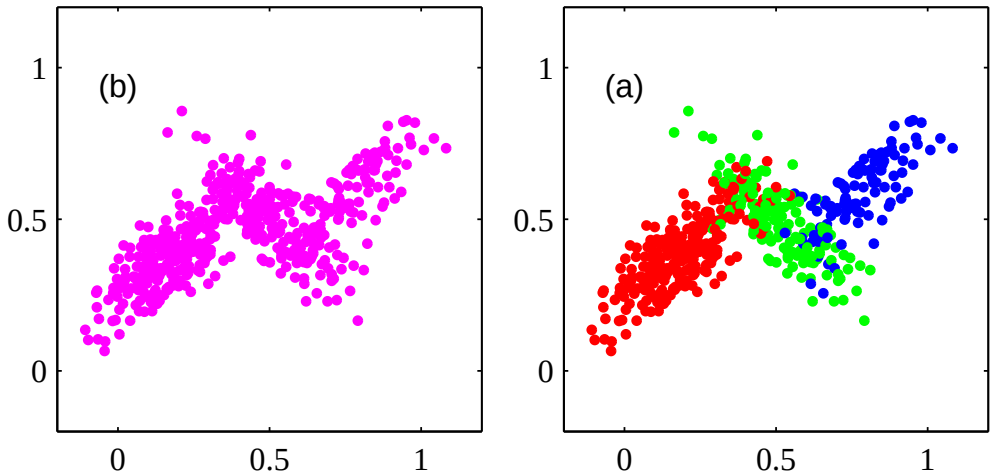
- In the K-means algorithm, a hard assignment of points to clusters is made
- However, for points near the decision boundary, this may not be a good idea
- Instead, we could think about making a soft assignment of points to clusters

Gaussian Mixture Model



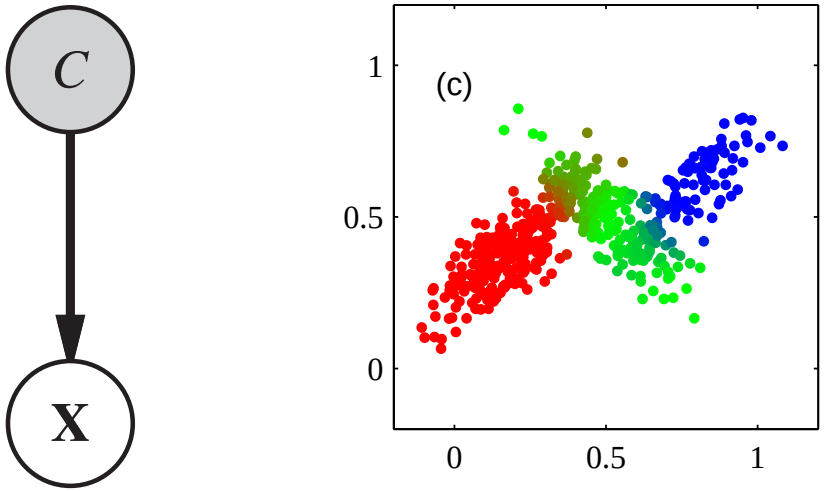
- The Gaussian mixture model (or mixture of Gaussians MoG) models the data as a combination of Gaussians.
- a: constant density contours. b: marginal probability $p(\boldsymbol{x})$. c: surface plot.
- Widely used general approximation for multi-modal distributions.
- See text Sec. 20.2.6 for kernel version

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 $\boldsymbol{x}$, we first generate a value $C = i$ for the component/cluster $C \in \{1, 2, 3\}$
- We then generate a vector $\boldsymbol{x}$ using $P(\boldsymbol{x}|C = i) \sim \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$ using the Gaussian parameters for component i

MoG Marginal over Observed Variables

- The marginal distribution $p(\boldsymbol{x}|\boldsymbol{\theta})$ for this model with parameter values $\boldsymbol{\theta}$ is:

$$\begin{aligned} p(\boldsymbol{x}) &= \sum_{i=1}^{3} p(\boldsymbol{x}, C=i) = \sum_{i=1}^{3} p(C=i)p(\boldsymbol{x}|C=i) \\ &= \sum_{i=1}^{3} w_i \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) \end{aligned}$$

- A mixture of Gaussians

Learning Gaussian Mixture Models

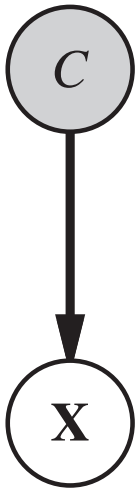
The EM algorithm can be used to learn Bayes net parameters with unobserved variables

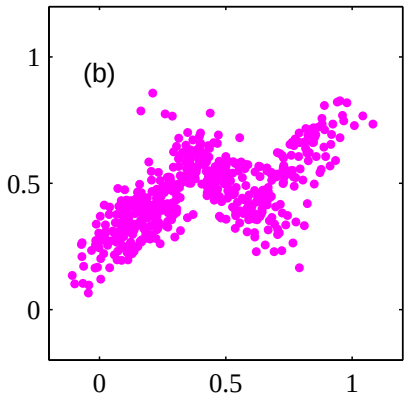
1. Initialize parameters randomly (or using k-means)
2. Repeat the following

   Expectation E-step Find probability of cluster assignments

   Maximization M-step Update parameter values

E-step: Probability of Cluster Assignments



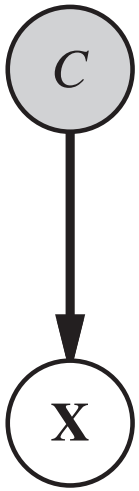


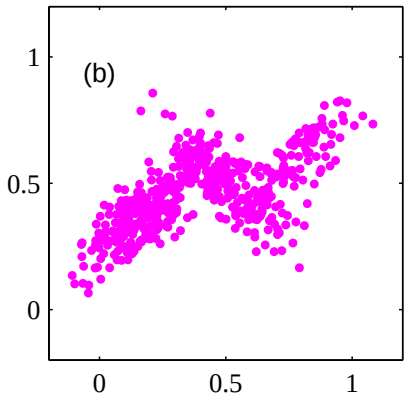
- How to find $p_{ij} = P(C = i|\boldsymbol{x}_j)$?
- Use Bayes' Theorem

$$p_{ij} \propto P(\boldsymbol{x}_j|C = i)P(C = i)$$

- Bayes net inference!

E-step: Probability of Cluster Assignments





- How to find $p_{ij} = P(C = i|\boldsymbol{x}_j)$?
- Use Bayes' Theorem

$$p_{ij} \propto P(\boldsymbol{x}_j|C = i)P(C = i)$$

- Bayes net inference!

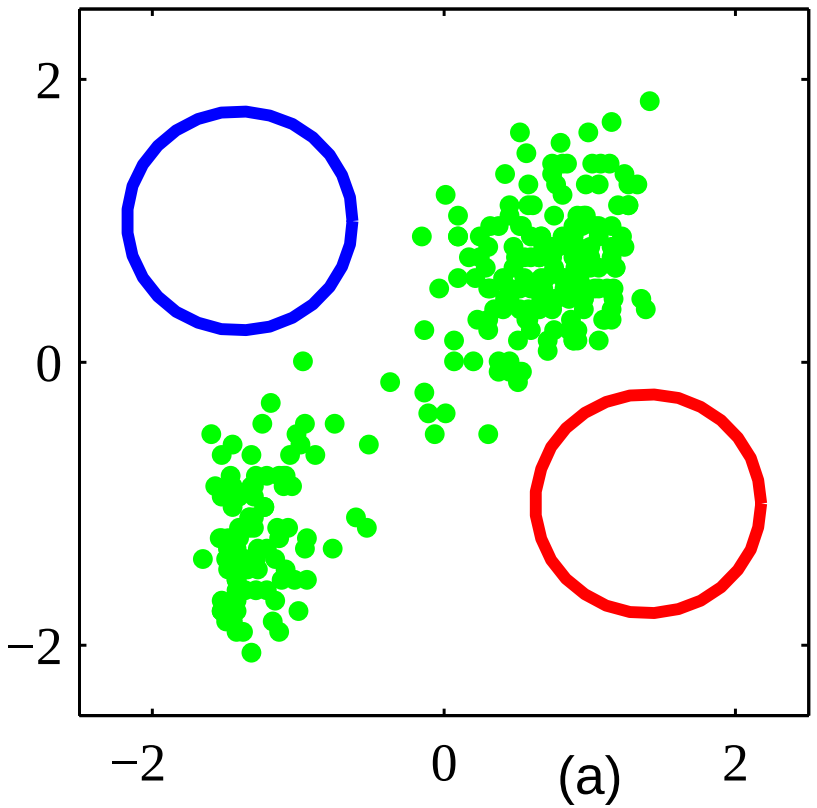
EM for Gaussian Mixtures: Update Parameters

- Update parameters wrt soft cluster assignments p_{ij}

$$
\begin{aligned}
n_i &\equiv \sum_j p_{ij} \\
w_i \equiv P(C = i) &:= n_i / N \\
\boldsymbol{\mu}_i &:= \sum_j p_{ij} \boldsymbol{x}_j / n_i \\
\boldsymbol{\Sigma}_i &= \sum_j p_{ij} (\boldsymbol{x}_j - \boldsymbol{\mu}_i)(\boldsymbol{x}_j - \boldsymbol{\mu}_i)^T / n_i
\end{aligned}
$$

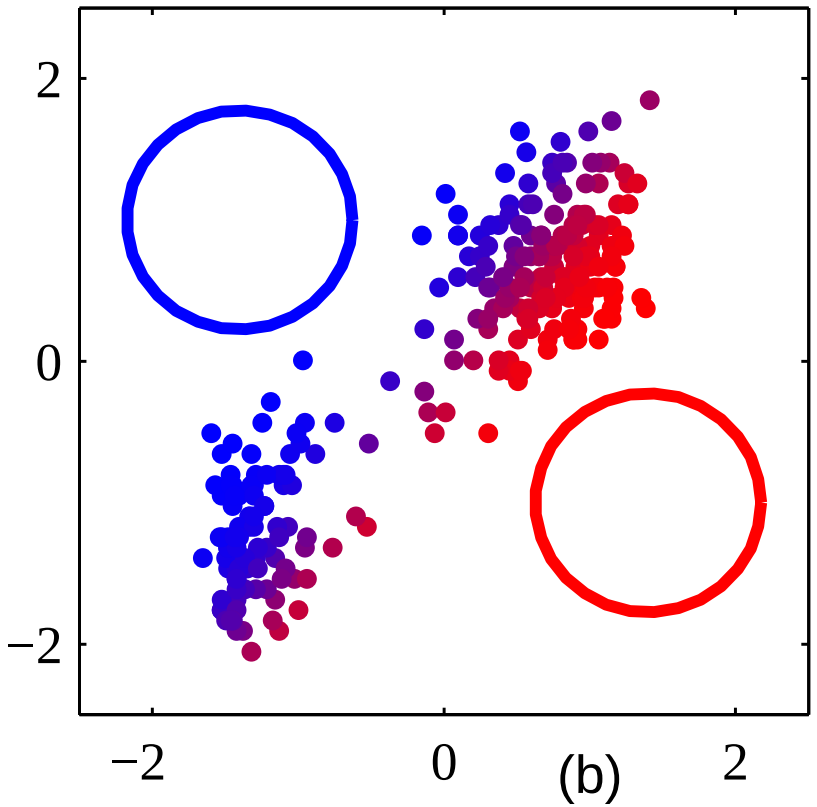
- n_i is the effective (estimated) number of data points from component i

MoG EM - Example



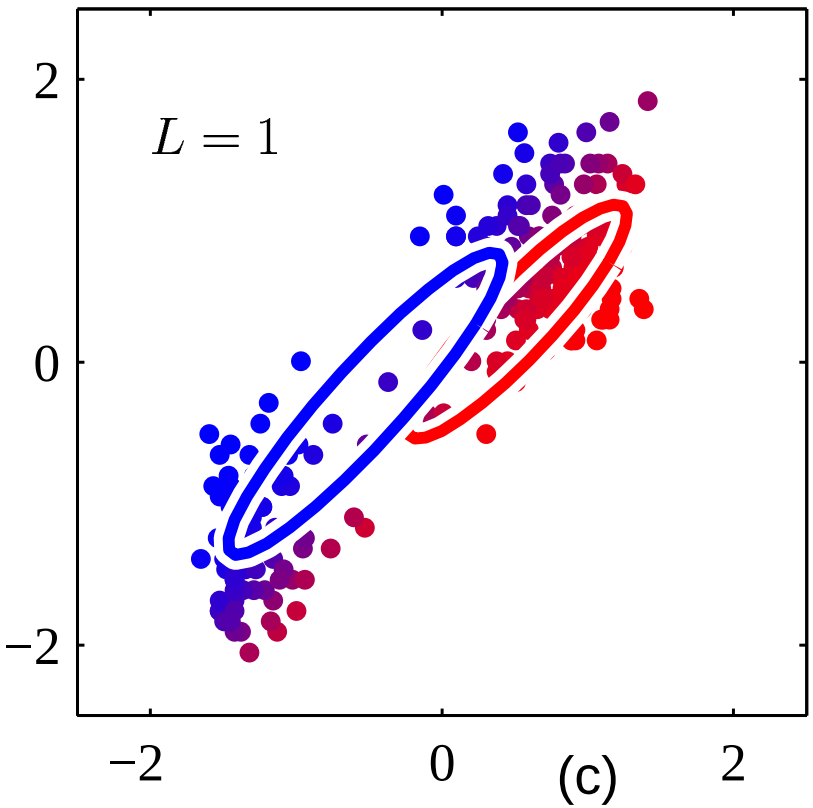
- Same initialization as with K-means before
  - Often, K-means is actually used to initialize EM

MoG EM - Example



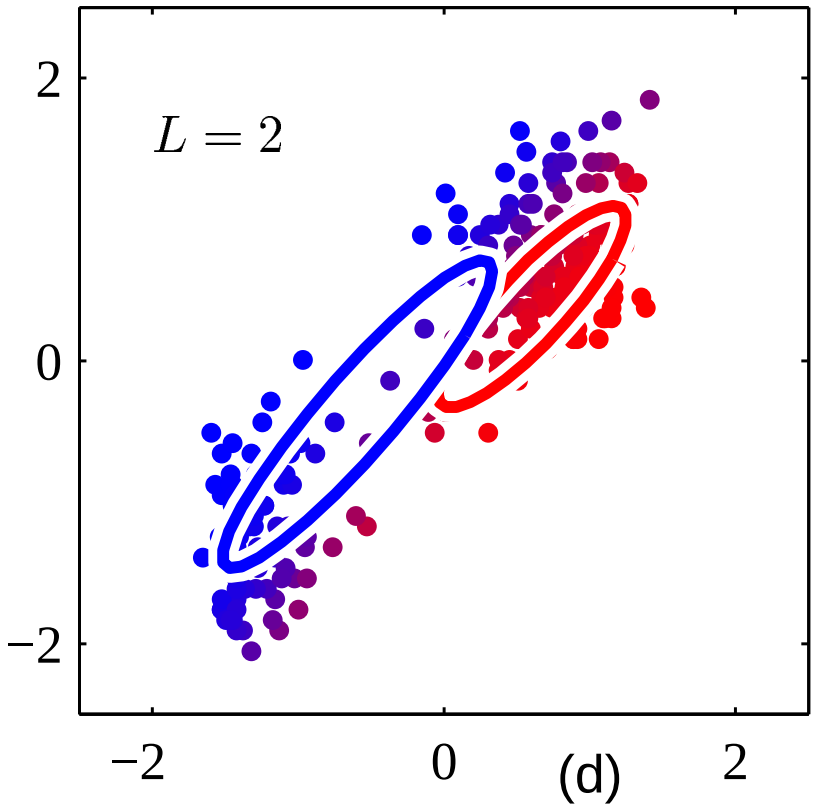
- Calculate cluster probabilities p_{ij}
- Shown using mixtures of colours.

MoG EM - Example



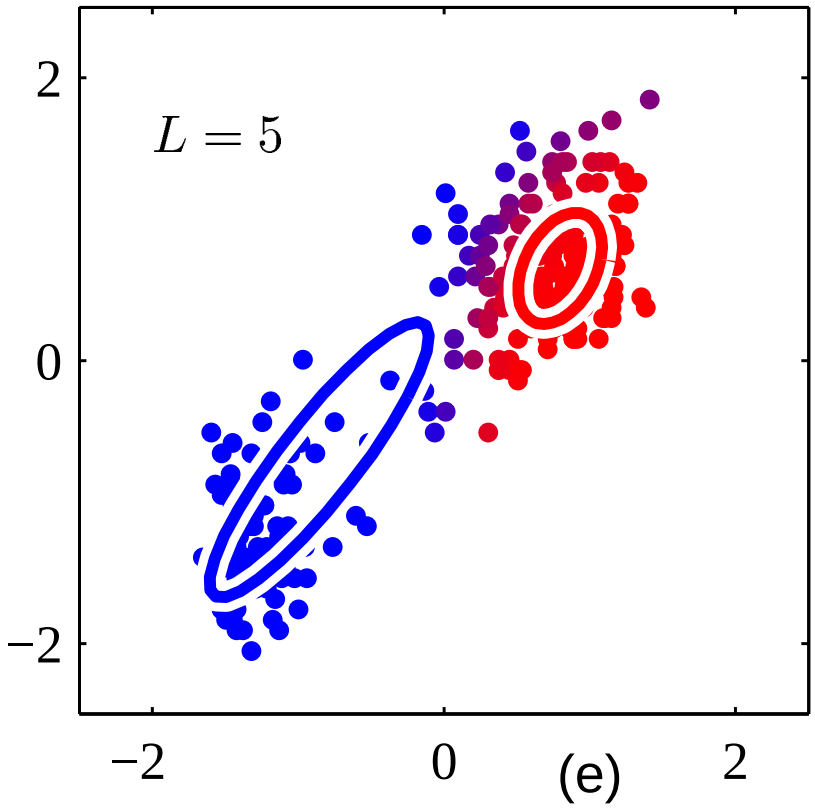
- Calculate model parameters $\{P(C = i), \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\}$ using the cluster probabilities

MoG EM - Example



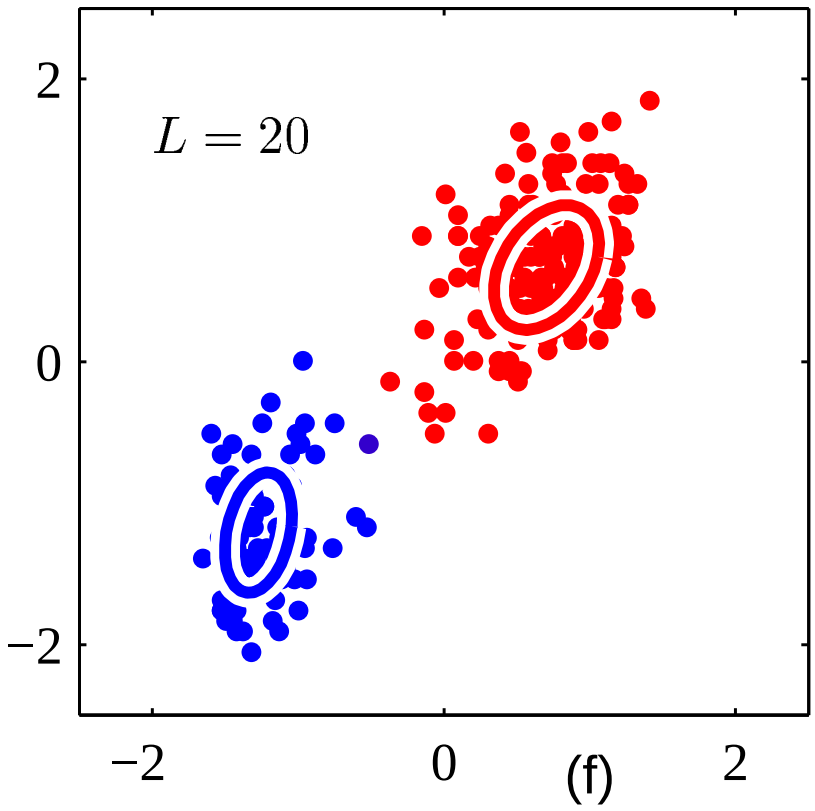
- Iteration 2

MoG EM - Example



- Iteration 5

MoG EM - Example



- Iteration 20 - converged

EM and Maximum Likelihood

1. The EM procedure is guaranteed to increase at each step, the data log-likelihood $\ln p(\boldsymbol{d}|\boldsymbol{\theta})$.
2. Therefore converges to *local log-likelihood maximum.*
3. Data log-likelihood of true model is around 650 in figure

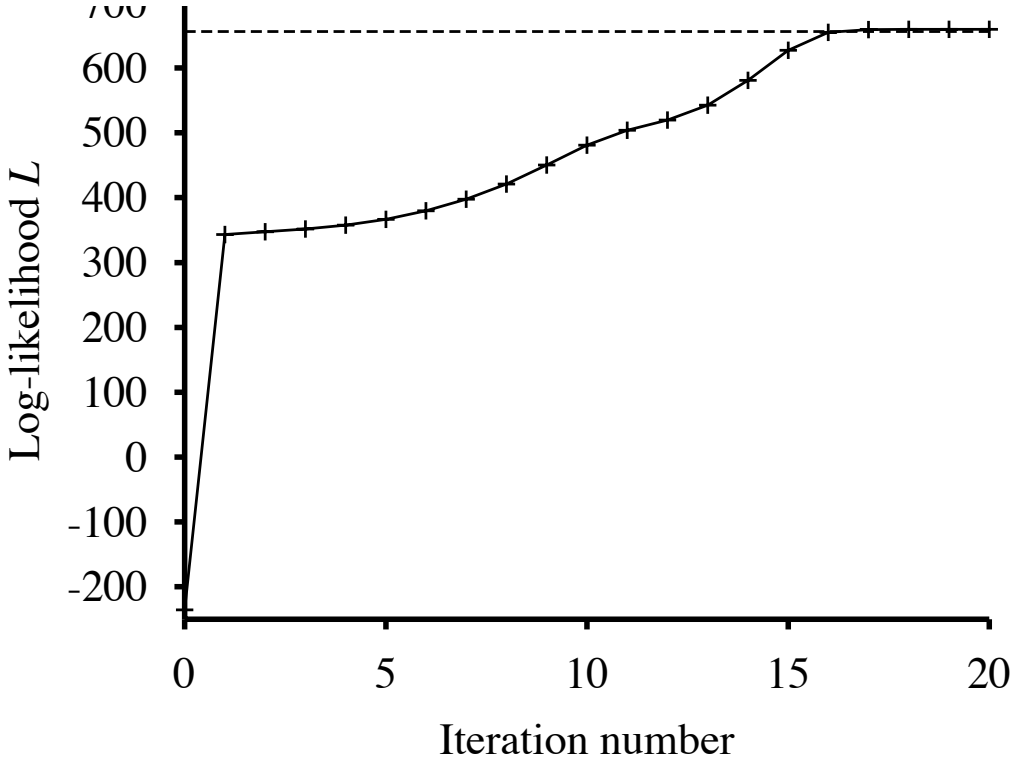


Fig. Russell and Norvig 20.12

Outline

Hard EM Algorithm

- We assume a probabilistic model, specifically the complete-data log likelihood function $P(\boldsymbol{x}, \boldsymbol{Z} = \boldsymbol{z}|\boldsymbol{\theta})$.
- "Goodness" of the model is the log-likelihood $L(\boldsymbol{x}, \boldsymbol{Z} = \boldsymbol{z}|\boldsymbol{\theta}) \equiv \ln P(\boldsymbol{x}, \boldsymbol{Z} = \boldsymbol{z}|\boldsymbol{\theta})$.
- Guess the value for latent variables that is the *expected value given current parameter settings*: $E[\boldsymbol{Z}]$ where the distribution over latent variables is given by $P(\boldsymbol{Z}|\boldsymbol{x}, \boldsymbol{\theta}^{(i)})$.
- *Given latent variable values*, the *next* parameter values $\boldsymbol{\theta}^{i+1}$ are evaluated by maximizing the likelihood $L(\boldsymbol{x}|\boldsymbol{Z} = E[\boldsymbol{Z}], \boldsymbol{\theta})$.
- Example: fill in missing values, single variable, Gaussian model.

Generalized EM Algorithm

- In Bayesian fashion, do *not* guess a best value for the latent variables $\boldsymbol{Z}$.
- Instead, average over the distribution $P(\boldsymbol{Z}|\boldsymbol{x}, \boldsymbol{\theta}^{(i)})$ given the current hypothesis.
- *Given a latent variable distribution*, the next parameter values $\boldsymbol{\theta}^{(i+1)}$ are evaluated by taking the *expected* likelihood $L(\boldsymbol{x}, \boldsymbol{Z} = \boldsymbol{z}|\boldsymbol{\theta}^{(i+1)})$ over all possible latent variable settings.
- In one equation:

$$\boldsymbol{\theta}^{(i+1)} = \text{argmax}_{\boldsymbol{\theta}} \sum_{z} P(\boldsymbol{Z}|\boldsymbol{x}, \boldsymbol{\theta}^{(i)}) L(\boldsymbol{x}, \boldsymbol{Z} = \boldsymbol{z}|\boldsymbol{\theta})$$

EM Algorithm: The procedure

1. Guess an initial parameter setting $\boldsymbol{\theta}^{(i)}$.
2. Repeat until convergence:
   - 2.1 The E-step: Evaluate $P(\boldsymbol{Z} = \boldsymbol{z}|\boldsymbol{x}, \boldsymbol{\theta}^{(i)})$.
   - 2.2 The M-step:
     - Evaluate the function $\mathcal{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}^{(i)}) \equiv \sum_{\boldsymbol{z}} P(\boldsymbol{Z}|\boldsymbol{x}, \boldsymbol{\theta}^{(i)}) L(\boldsymbol{x}, \boldsymbol{Z} = \boldsymbol{z}|\boldsymbol{\theta})$
     - Maximize $\mathcal{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}^{(i)})$ wrt $\boldsymbol{\theta}$. Update $\boldsymbol{\theta}^{(i+1)}$ to be the maximum.

EM for Gaussian Mixtures: E-step (I)

- Want to evaluate $\mathcal{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}^{(i)}) \equiv \sum_{\boldsymbol{z}} P(\boldsymbol{Z}|\boldsymbol{x}, \boldsymbol{\theta}^{(i)}) L(\boldsymbol{x}, \boldsymbol{Z} = \boldsymbol{z}|\boldsymbol{\theta})$
- Let $Z_{ij} = 1$ denote the event that data point j is assigned to cluster i.
- Then

$$\begin{aligned} L(\boldsymbol{x}, \boldsymbol{Z} = \boldsymbol{z}|\boldsymbol{\theta}) &= \ln \prod_i \prod_j P(\boldsymbol{x}_j|C = i, \boldsymbol{\theta})^{z_{ij}} \times P(C = i|\boldsymbol{\theta})^{z_{ij}} \\ &= \sum_i \sum_j z_{ij} [\ln P(\boldsymbol{x}_j|C = i, \boldsymbol{\theta})^{z_{ij}} + \ln P(C = i|\boldsymbol{\theta})] \\ &\equiv \sum_i \sum_j z_{ij} \ell_{ij} \end{aligned}$$

EM for Gaussian Mixtures: E-step (II)

- Want to evaluate $\mathcal{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}^{(i)}) \equiv \sum_{\boldsymbol{z}} P(\boldsymbol{Z}|\boldsymbol{x}, \boldsymbol{\theta}^{(i)}) \sum_i \sum_j z_{ij}\ell_{ij}$
- This is the expectation of a sum = sum of expectations (see assignment).
- Therefore

$$\sum_{\boldsymbol{z}} P(\boldsymbol{Z}|\boldsymbol{x}, \boldsymbol{\theta}^{(i)}) \sum_i \sum_j z_{ij}\ell_{ij} \quad = \quad \sum_i \sum_j E_{Z_{ij}|\boldsymbol{x}, \boldsymbol{\theta}^{(i)}}[z_{ij}\ell_{ij}]$$

$$= \sum_i \sum_j P(Z_{ij} = 1|\boldsymbol{x}, \boldsymbol{\theta}^{(i)})\ell_{ij} \quad = \sum_i \sum_j p_{ij}\ell_{ij}$$

EM for Gaussian Mixtures: E-step (III)

- Want to evaluate $\mathcal{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}^{(i)}) \equiv \sum_{\boldsymbol{z}} P(\boldsymbol{Z}|\boldsymbol{x}, \boldsymbol{\theta}^{(i)}) L(\boldsymbol{x}, \boldsymbol{Z} = \boldsymbol{z}|\boldsymbol{\theta})$
- We showed that this is equal to $\sum_i \sum_j p_{ij} L(x_j, z_{ij}|\boldsymbol{\theta})$
- Deriving the maximization solution is more or less straightforward and leads to the updates shown before.
- Variations on this theme work for many models where the likelihood function factors (think Bayes nets).

EM - Summary

- EM finds local maximum to (incomplete) data likelihood

$$P(\boldsymbol{x}|\boldsymbol{\theta}) = \sum_{\boldsymbol{Z}} p(\boldsymbol{x}, \boldsymbol{Z}|\boldsymbol{\theta})$$

- Iterates two steps:
  - **E step** calculates the distribution of the missing variables $\boldsymbol{Z}$
  - (Hard EM "fills in" the variables).
  - **M step** maximizes expected complete log likelihood (expectation wrt **E step** distribution)
  - Often the updates can be obtained in closed form
  - E.g., for discrete Bayesian networks (see Ch.20.3.2)

Conclusion

- 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