

# CS559 Machine Learning

## Clustering, GMM, and EM

Tian Han

Department of Computer Science  
Stevens Institute of Technology

Week 9

# Outline

- Clustering
  - K-means Algorithm
- Gaussian Mixture Model (GMM)
- Expectation-Maximization (EM) algorithm for GMM

## Clustering

A horizontal row of 18 small circles, each containing a dot, used as a visual separator at the bottom of the page.

GMM, EM

A horizontal row of 18 small, uniform circles arranged in a single line.

## Clustering

## Supervised and unsupervised learning

Recall: previous lectures, we discussed **supervised learning** where label information is available.

- *Linear Model*: linear regression, linear classification (logistic regression, perceptron, Fisher Discriminant Analysis etc)
  - *Non-parametric*: K-nearest neighbour.
  - *Non-linear*: Neural Network, SVM, boosting machine etc

## Supervised and unsupervised learning

Recall: previous lectures, we discussed **supervised learning** where label information is available.

- *Linear Model*: linear regression, linear classification (logistic regression, perceptron, Fisher Discriminant Analysis etc)
  - *Non-parametric*: K-nearest neighbour.
  - *Non-linear*: Neural Network, SVM, boosting machine etc

We also discussed a bit about **unsupervised learning** where no label information is given.

- Density estimation: Parzen window, KNN and the general latent variable model.
  - Clustering: K means, GMM (this lecture).

## Clustering

- Goal: Grouping a collection of objects (data points) into subsets or “clusters”, such that objects in the same cluster are more similar (in some sense) to each other than to those in other clusters.

## Clustering

- Goal: **Grouping** a collection of objects (data points) into subsets or “**clusters**”, such that objects in the same cluster are more similar (in some sense) to each other than to those in other clusters.
  - Fundamental to all clustering techniques is the choice of **distance** or **dissimilarity** measure between two objects.

## What is similarity?



Figure: Y. Ning, CS559 S19

- The quality or state of being similar; likeness; resemblance; as, a similarity of features – Websters' dictionary.

## What is similarity?



Figure: Y. Ning, CS559 S19

- The quality or state of being similar; likeness; resemblance; as, a similarity of features – Websters' dictionary.
  - Similarity is hard to define, but *we know it when we see it*.

## What is similarity?



Figure: Y. Ning, CS559 S19

- The quality or state of being similar; likeness; resemblance; as, a similarity of features – Websters' dictionary.
  - Similarity is hard to define, but *we know it when we see it*.
  - The real meaning of similarity is a philosophical question. We will take a more pragmatic approach.

## Dissimilarities based on features

- Assuming  $\mathbf{x}_i = (x_i^{(1)}, x_i^{(2)}, \dots, x_i^{(D)})^T \in \mathbb{R}^D, i = 1, \dots, N$

## Dissimilarities based on features

- Assuming  $\mathbf{x}_i = (x_i^{(1)}, x_i^{(2)}, \dots, x_i^{(D)})^T \in \mathbb{R}^D, i = 1, \dots, N$
- Distance (dissimilarity):  $D(\mathbf{x}_i, \mathbf{x}_j) = \sum_{k=1}^D d_k(x_i^{(k)}, x_j^{(k)})$

## Dissimilarities based on features

- Assuming  $\mathbf{x}_i = (x_i^{(1)}, x_i^{(2)}, \dots, x_i^{(D)})^T \in \mathbb{R}^D, i = 1, \dots, N$
- Distance (dissimilarity):  $D(\mathbf{x}_i, \mathbf{x}_j) = \sum_{k=1}^D d_k(x_i^{(k)}, x_j^{(k)})$
- Squared Euclidean distance:  $d_k(x_i^{(k)}, x_j^{(k)}) = (x_i^{(k)} - x_j^{(k)})^2 \Rightarrow D(\mathbf{x}_i, \mathbf{x}_j) = \sum_{k=1}^D (x_i^{(k)} - x_j^{(k)})^2$

## Nonprobabilistic Algorithms

- These algorithms work **directly on the observed data**, without regard to a probability model describing the data.

## Nonprobabilistic Algorithms

- These algorithms work **directly on the observed data**, without regard to a probability model describing the data.
- Commonly used in data mining, since often no prior knowledge about the process that generated the data is available.

# Nonprobabilistic Algorithms

- Assuming  $\mathbf{x}_i \in \mathbb{R}^D, i = 1, \dots, N$

# Nonprobabilistic Algorithms

- Assuming  $\mathbf{x}_i \in \mathbb{R}^D, i = 1, \dots, N$
- Pre-specified number of cluster  $K, k \in \{1, \dots, K\}$

## Nonprobabilistic Algorithms

- Assuming  $\mathbf{x}_i \in \mathbb{R}^D, i = 1, \dots, N$
- Pre-specified number of cluster  $K, k \in \{1, \dots, K\}$
- Each data point  $\mathbf{x}_i$  is assigned to **one, and only one** cluster.

## Nonprobabilistic Algorithms

- Assuming  $\mathbf{x}_i \in \mathbb{R}^D, i = 1, \dots, N$
- Pre-specified number of cluster  $K, k \in \{1, \dots, K\}$
- Each data point  $\mathbf{x}_i$  is assigned to **one, and only one** cluster.
- Goal: Find a partition of the data into  $K$  clusters that achieves a required objective, defined in terms of a dissimilarity function  $D(\mathbf{x}_i, \mathbf{x}_k)$ .

## Nonprobabilistic Algorithms

- Assuming  $\mathbf{x}_i \in \mathbb{R}^D, i = 1, \dots, N$
- Pre-specified number of cluster  $K, k \in \{1, \dots, K\}$
- Each data point  $\mathbf{x}_i$  is assigned to **one, and only one** cluster.
- Goal: Find a partition of the data into  $K$  clusters that achieves a required objective, defined in terms of a dissimilarity function  $D(\mathbf{x}_i, \mathbf{x}_k)$ .
- Usually, the assignment of data to clusters is done so as to minimize a “loss” function that measures the degrees to which the clustering goal is not met.

## Nonprobabilistic Algorithms

- Since the goal is to assign close points to the same cluster, a natural objective function  $J$  would be **within cluster tightness**.  
Need to define following two notations:

## Nonprobabilistic Algorithms

- Since the goal is to assign close points to the same cluster, a natural objective function  $J$  would be **within cluster tightness**. Need to define following two notations:
- Notations:
  - binary indicator  $r_{nk}$ : (describe which of the  $K$  clusters the data point  $\mathbf{x}_n$  is assigned to)

$$r_{nk} = \begin{cases} 1, & \text{if } \mathbf{x}_n \text{ is assigned to cluster } K \\ 0, & \text{otherwise} \end{cases}$$

## Nonprobabilistic Algorithms

- Since the goal is to assign close points to the same cluster, a natural objective function  $J$  would be **within cluster tightness**. Need to define following two notations:
- Notations:
  - binary indicator  $r_{nk}$ : (describe which of the  $K$  clusters the data point  $\mathbf{x}_n$  is assigned to)

$$r_{nk} = \begin{cases} 1, & \text{if } \mathbf{x}_n \text{ is assigned to cluster } K \\ 0, & \text{otherwise} \end{cases}$$

E.g., if  $K = 3$ ,  $\mathbf{x}_n$  is assigned to cluster 2, then  
 $r_{n1} = 0, r_{n2} = 1, r_{n3} = 0$ . Known as *1-of-K coding scheme*.

## Nonprobabilistic Algorithms

- Since the goal is to assign close points to the same cluster, a natural objective function  $J$  would be **within cluster tightness**. Need to define following two notations:
- Notations:
  - binary indicator  $r_{nk}$ : (describe which of the  $K$  clusters the data point  $\mathbf{x}_n$  is assigned to)

$$r_{nk} = \begin{cases} 1, & \text{if } \mathbf{x}_n \text{ is assigned to cluster } K \\ 0, & \text{otherwise} \end{cases}$$

E.g., if  $K = 3$ ,  $\mathbf{x}_n$  is assigned to cluster 2, then

$r_{n1} = 0, r_{n2} = 1, r_{n3} = 0$ . Known as *1-of-K coding scheme*.

- Prototype  $\mu_k$  for each cluster.  $\mu_k \in \mathcal{R}^D$  and can be considered as centres of clusters.

# Objective function $J$

$$J = \sum_{n=1}^N \sum_{k=1}^K r_{nk} D(\mathbf{x}_n, \mu_k)$$

## Objective function $J$

$$J = \sum_{n=1}^N \sum_{k=1}^K r_{nk} D(\mathbf{x}_n, \mu_k)$$

- Also called *distortion measure*.

## Objective function $J$

$$J = \sum_{n=1}^N \sum_{k=1}^K r_{nk} D(\mathbf{x}_n, \mu_k)$$

- Also called *distortion measure*.
- Represents distances of each data point to its assigned vector  $\mu_k$ .

## Objective function $J$

$$J = \sum_{n=1}^N \sum_{k=1}^K r_{nk} D(\mathbf{x}_n, \mu_k)$$

- Also called *distortion measure*.
- Represents distances of each data point to its assigned vector  $\mu_k$ .
- Need to select  $D(.,.)$ , find  $r_{nk}$  and  $\mu_k$ .

## K-means: objective

$$J = \sum_{n=1}^N \sum_{k=1}^K r_{nk} D(\mathbf{x}_n, \mu_k)$$

## K-means: objective

$$J = \sum_{n=1}^N \sum_{k=1}^K r_{nk} D(\mathbf{x}_n, \mu_k)$$

Use squared Euclidean distance as  $D(., .)$ :

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

## K-means: objective

$$J = \sum_{n=1}^N \sum_{k=1}^K r_{nk} D(\mathbf{x}_n, \mu_k)$$

Use squared Euclidean distance as  $D(., .)$ :

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

- Represents the sum of square distance of each data point to its assigned vector  $\mu_k$ .

## K-means: objective

$$J = \sum_{n=1}^N \sum_{k=1}^K r_{nk} D(\mathbf{x}_n, \mu_k)$$

Use squared Euclidean distance as  $D(., .)$ :

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

- Represents the sum of square distance of each data point to its assigned vector  $\mu_k$ .
- Goal is to find values for the  $r_{nk}$  and the  $\mu_k$  so as to minimize  $J$ .

## K-means: objective

$$J = \sum_{n=1}^N \sum_{k=1}^K r_{nk} D(\mathbf{x}_n, \mu_k)$$

Use squared Euclidean distance as  $D(., .)$ :

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

- Represents the sum of square distance of each data point to its assigned vector  $\mu_k$ .
- Goal is to find values for the  $r_{nk}$  and the  $\mu_k$  so as to minimize  $J$ .
- Through iterative procedure...

## K-means: algorithm

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

Minimize  $J$  in iterative procedure to find out  $r_{nk}$  and  $\mu_k$ :

## K-means: algorithm

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

Minimize  $J$  in iterative procedure to find out  $r_{nk}$  and  $\mu_k$ :

- First phase: Choose some initial values for the  $\mu_k$ . Then minimize  $J$  with respect to the  $r_{nk}$ , keeping the  $\mu_k$  fixed.

## K-means: algorithm

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

Minimize  $J$  in iterative procedure to find out  $r_{nk}$  and  $\mu_k$ :

- First phase: Choose some initial values for the  $\mu_k$ . Then minimize  $J$  with respect to the  $r_{nk}$ , keeping the  $\mu_k$  fixed.
- Second phase: We minimize  $J$  with respect to the  $\mu_k$ , keeping  $r_{nk}$  fixed.

## K-means: algorithm

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

**First phase:** Minimize  $J$  with respect to the  $r_{nk}$ , keeping the  $\mu_k$  fixed.

## K-means: algorithm

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

**First phase:** Minimize  $J$  with respect to the  $r_{nk}$ , keeping the  $\mu_k$  fixed.

- When  $\mu_k$  fixed,  $J$  becomes linear function of  $r_{nk}$ .

## K-means: algorithm

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

**First phase:** Minimize  $J$  with respect to the  $r_{nk}$ , keeping the  $\mu_k$  fixed.

- When  $\mu_k$  fixed,  $J$  becomes linear function of  $r_{nk}$ .
- Terms involving different  $n$  are independent, optimize for each  $n$  separately.

## K-means: algorithm

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

**First phase:** Minimize  $J$  with respect to the  $r_{nk}$ , keeping the  $\mu_k$  fixed.

- When  $\mu_k$  fixed,  $J$  becomes linear function of  $r_{nk}$ .
- Terms involving different  $n$  are independent, optimize for each  $n$  separately.
- Choosing  $r_{nk}$  to be 1 for whichever value of  $k$  gives the minimum value of  $\|\mathbf{x}_n - \mu_k\|^2$ .

## K-means: algorithm

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

**First phase:** Minimize  $J$  with respect to the  $r_{nk}$ , keeping the  $\mu_k$  fixed.

- When  $\mu_k$  fixed,  $J$  becomes linear function of  $r_{nk}$ .
- Terms involving different  $n$  are independent, optimize for each  $n$  separately.
- Choosing  $r_{nk}$  to be 1 for whichever value of  $k$  gives the minimum value of  $\|\mathbf{x}_n - \mu_k\|^2$ .

$$r_{nk} = \begin{cases} 1, & \text{if } k = \arg \min_j \|\mathbf{x}_n - \mu_j\|^2 \\ 0, & \text{otherwise} \end{cases}$$

## K-means: algorithm

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

**Second phase:** We minimize  $J$  with respect to the  $\mu_k$ , keeping  $r_{nk}$  fixed.

## K-means: algorithm

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

**Second phase:** We minimize  $J$  with respect to the  $\mu_k$ , keeping  $r_{nk}$  fixed.

- $J$  is quadratic function of  $\mu_k$ .

## K-means: algorithm

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

**Second phase:** We minimize  $J$  with respect to the  $\mu_k$ , keeping  $r_{nk}$  fixed.

- $J$  is quadratic function of  $\mu_k$ .
- Let:  $\frac{\partial J}{\partial \mu_k} = 0$ :

## K-means: algorithm

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

**Second phase:** We minimize  $J$  with respect to the  $\mu_k$ , keeping  $r_{nk}$  fixed.

- $J$  is quadratic function of  $\mu_k$ .
- Let:  $\frac{\partial J}{\partial \mu_k} = 0$ :

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

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

## K-means: algorithm

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

**Second phase:** We minimize  $J$  with respect to the  $\mu_k$ , keeping  $r_{nk}$  fixed.

- $J$  is quadratic function of  $\mu_k$ .
- Let:  $\frac{\partial J}{\partial \mu_k} = 0$ :

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

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

- Mean of all of the data points  $\mathbf{x}_n$  assigned to cluster  $k$ .

## K-means: algorithm

Summary: initialize the  $\mu_k$ , then iterate the following two steps until convergence.

1. Minimize  $J$  with respect to the  $r_{nk}$ , keeping the  $\mu_k$  fixed.

$$r_{nk} = \begin{cases} 1, & \text{if } k = \arg \min_j \|\mathbf{x}_n - \mu_j\|^2 \\ 0, & \text{otherwise} \end{cases}$$

2. Minimize  $J$  with respect to the  $\mu_k$ , keeping  $r_{nk}$  fixed.

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

## K-means: example

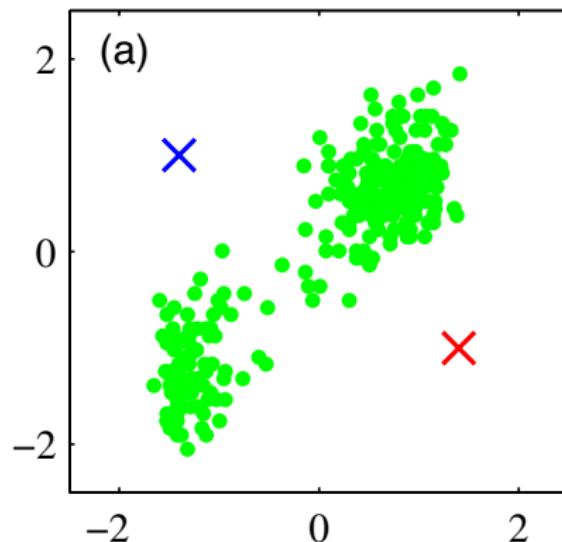


Figure: Initialization [C. Bishop, PRML]

## K-means: example

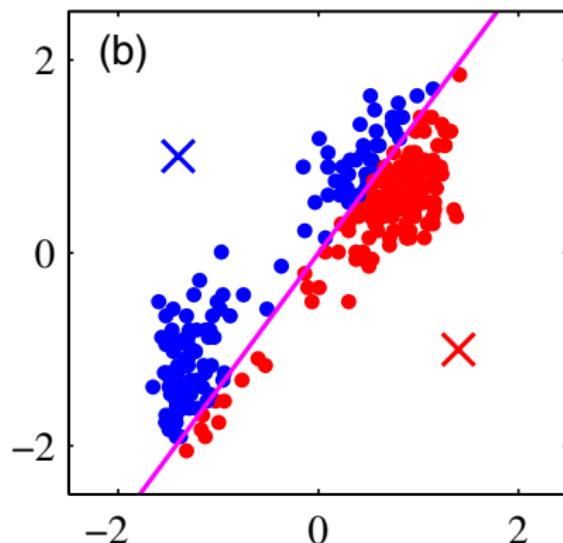


Figure: Step 1: assign clusters, determine  $r_{nk}$ . [C. Bishop, PRML]

## K-means: example

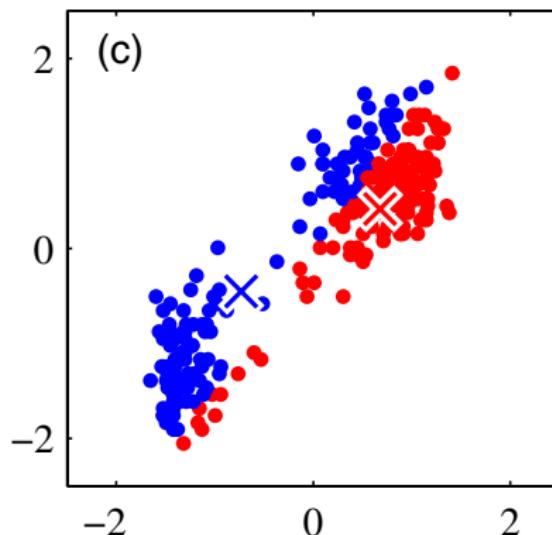


Figure: Step 1: recalculate means of clusters, determine  $\mu_k$ . [C. Bishop, PRML]

## K-means: example

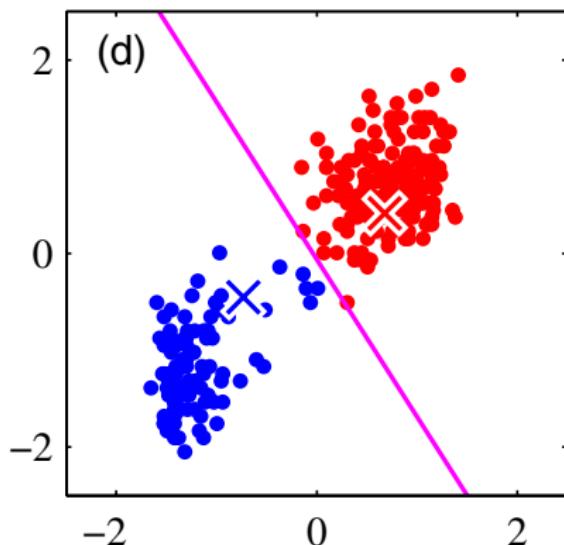


Figure: Step 2: assign clusters to new means, determine  $r_{nk}$ . [C. Bishop, PRML]

## K-means: example

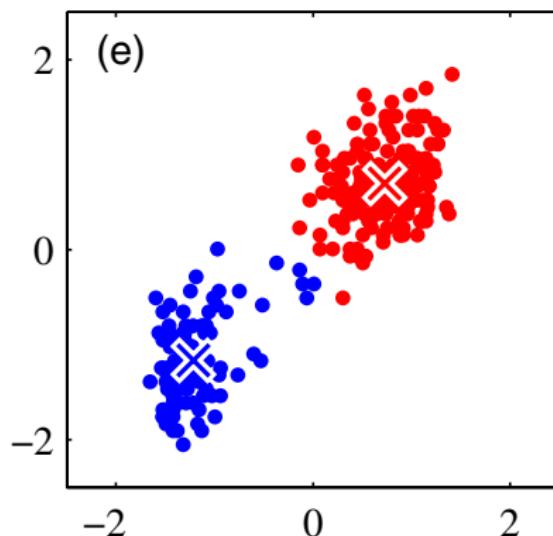


Figure: Step 2: recalculate means of clusters, determine  $\mu_k$ . [C. Bishop, PRML]

## K-means: example

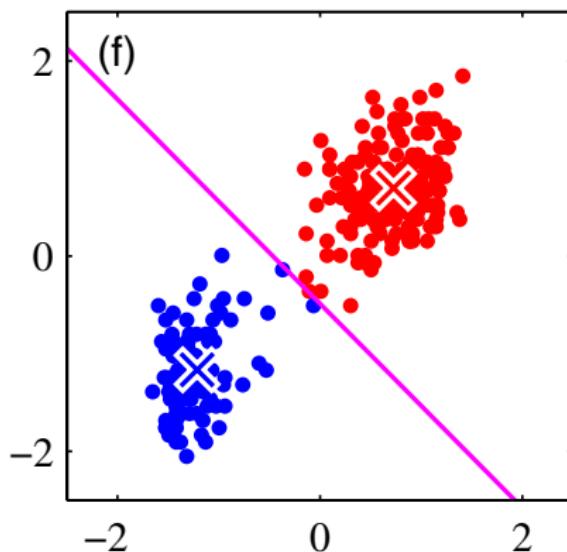


Figure: Step 3: assign clusters to new means, determine  $r_{nk}$ . [C. Bishop, PRML]

## K-means: example

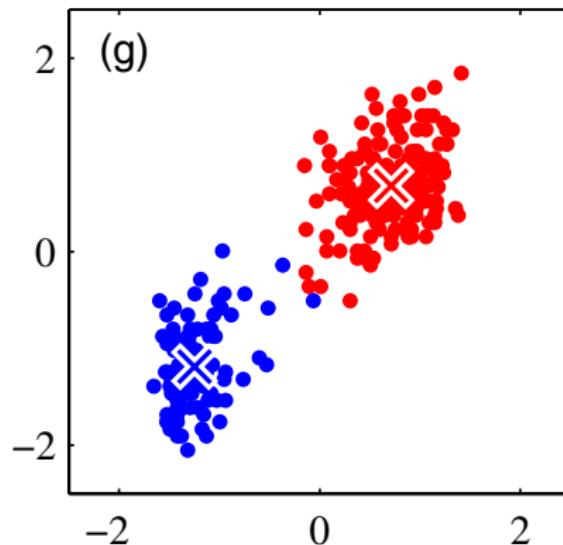


Figure: Step 3: recalculate means of clusters, determine  $\mu_k$ . [C. Bishop, PRML]

## K-means: example

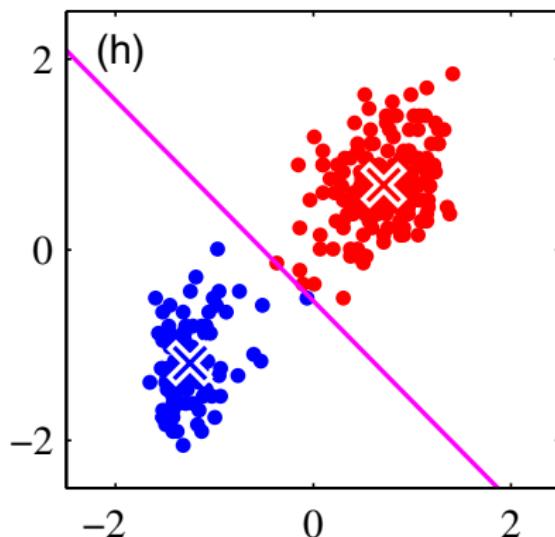
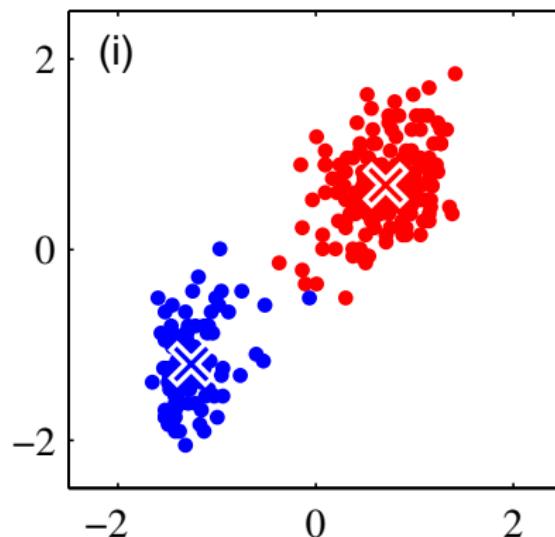


Figure: Step 4: assign clusters to new means, determine  $r_{nk}$ . [C. Bishop, PRML]

## K-means: example



**Figure:** Step 4: recalculate means of clusters, determine  $\mu_k$ . converged  
[C. Bishop, PRML]

## K-means: properties and limitations

- The algorithm converges to a local minimum

## K-means: properties and limitations

- The algorithm converges to a local minimum
- The solution depends on the initial values

## K-means: properties and limitations

- The algorithm converges to a local minimum
- The solution depends on the initial values
- One should start the algorithm with many different random choices for the initial means, and choose the solution having smallest value of the objective function

## K-means: properties and limitations

- The algorithm converges to a local minimum
- The solution depends on the initial values
- One should start the algorithm with many different random choices for the initial means, and choose the solution having smallest value of the objective function
- The algorithm is sensitive to outliers

## K-means: properties and limitations

- The algorithm converges to a local minimum
- The solution depends on the initial values
- One should start the algorithm with many different random choices for the initial means, and choose the solution having smallest value of the objective function
- The algorithm is sensitive to outliers
- The number of clusters  $K$  need to be pre-specified, but can be estimated from the data.

## K-means: properties and limitations

- The algorithm converges to a local minimum
- The solution depends on the initial values
- One should start the algorithm with many different random choices for the initial means, and choose the solution having smallest value of the objective function
- The algorithm is sensitive to outliers
- The number of clusters  $K$  need to be pre-specified, but can be estimated from the data.
- Every data point is assigned to one, and only one of the clusters. (hard assignment, no probability involved to reflect the level of uncertainty)

## K-means: properties and limitations

- The algorithm converges to a local minimum
- The solution depends on the initial values
- One should start the algorithm with many different random choices for the initial means, and choose the solution having smallest value of the objective function
- The algorithm is sensitive to outliers
- The number of clusters  $K$  need to be pre-specified, but can be estimated from the data.
- Every data point is assigned to one, and only one of the clusters. (hard assignment, no probability involved to reflect the level of uncertainty)
- How about probabilistic approach that obtain the soft assignment of data points?

Clustering

oooooooooooooooooooo

GMM, EM

●oooooooooooooooooooo

# Gaussian Mixture Model and EM

## Mixture of Gaussians

Single Gaussian may not be enough for modelling:

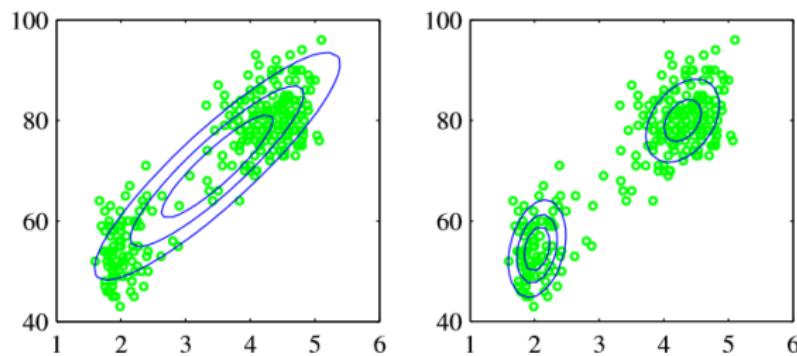


Figure: [C. Bishop, PRML]

## Mixture of Gaussians

Mixture of Gaussian distributions (Linear combination of Gaussians) can represent very complex densities.

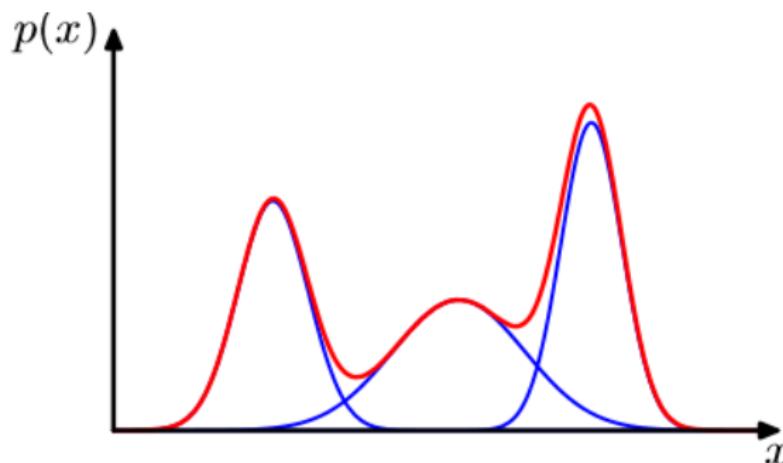


Figure: Three Gaussians (blue) and their sum (red). [C. Bishop, PRML]

## Mixtures of Gaussians

- We therefore consider a superposition of K Gaussian densities of the form

$$p(\mathbf{x}) = \sum_{k=1}^K \pi_k \mathcal{N}(\mathbf{x} | \mu_k, \Sigma_k)$$

- Each Gaussian density  $\mathcal{N}(\mathbf{x} | \mu_k, \Sigma_k)$  is called a *component* of the mixture and has its own mean  $\mu_k$  and covariance  $\Sigma_k$
- Mixing coefficients:**  $0 \leq \pi_k \leq 1$ ,  $\sum_{k=1}^K \pi_k = 1$  (integrate both sides of above definition)

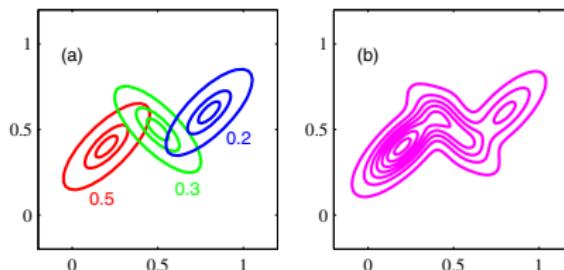


Figure: [C. Bishop, PRML]

## Mixtures of Gaussians

- From the sum and product rules, the **marginal density** is given by

$$p(\mathbf{x}) = \sum_{k=1}^K p(k)p(\mathbf{x}|k)$$

## Mixtures of Gaussians

- From the sum and product rules, the **marginal density** is given by

$$p(\mathbf{x}) = \sum_{k=1}^K p(k)p(\mathbf{x}|k)$$

- We can view  $\pi_k = p(k)$  as the **prior probability** of picking the  $k^{th}$  component.

## Mixtures of Gaussians

- From the sum and product rules, the **marginal density** is given by

$$p(\mathbf{x}) = \sum_{k=1}^K p(k)p(\mathbf{x}|k)$$

- We can view  $\pi_k = p(k)$  as the **prior probability** of picking the  $k^{th}$  component.
- The density  $\mathcal{N}(\mathbf{x}|\mu_k, \Sigma_k) = p(\mathbf{x}|k)$  as **the probability of  $\mathbf{x}$  conditioned on  $k$** .

## Mixtures of Gaussians

- From the sum and product rules, the **marginal density** is given by

$$p(\mathbf{x}) = \sum_{k=1}^K p(k)p(\mathbf{x}|k)$$

- We can view  $\pi_k = p(k)$  as the **prior probability** of picking the  $k^{th}$  component.
- The density  $\mathcal{N}(\mathbf{x}|\mu_k, \Sigma_k) = p(\mathbf{x}|k)$  as **the probability of  $\mathbf{x}$  conditioned on  $k$** .
- Using **Bayes' theorem**, the **posterior probabilities**  $p(k|\mathbf{x})$  (**responsibilities**):

$$\gamma_k(\mathbf{x}) = p(k|\mathbf{x}) = \frac{p(k)p(\mathbf{x}|k)}{\sum_l p(l)p(\mathbf{x}|l)} = \frac{\pi_k \mathcal{N}(\mathbf{x}|\mu_k, \Sigma_k)}{\sum_l \pi_l \mathcal{N}(\mathbf{x}|\mu_l, \Sigma_l)}$$

# Maximum Likelihood

Gaussian Mixture Distribution:

$$p(\mathbf{x}) = \sum_{k=1}^K \pi_k \mathcal{N}(\mathbf{x} | \mu_k, \Sigma_k)$$

# Maximum Likelihood

Gaussian Mixture Distribution:

$$p(\mathbf{x}) = \sum_{k=1}^K \pi_k \mathcal{N}(\mathbf{x} | \mu_k, \Sigma_k)$$

- The parameters:  $\pi, \mu, \Sigma$

# Maximum Likelihood

Gaussian Mixture Distribution:

$$p(\mathbf{x}) = \sum_{k=1}^K \pi_k \mathcal{N}(\mathbf{x} | \mu_k, \Sigma_k)$$

- The parameters:  $\pi, \mu, \Sigma$
- Suppose we have a data set of observations  $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\} \in \mathcal{R}^{N \times D}$ , model the data using a mixture of Gaussians. **Maximize the log of the likelihood function:**

$$\ln p(\mathbf{X} | \pi, \mu, \Sigma) = \sum_{n=1}^N \ln \left\{ \sum_{k=1}^K \pi_k \mathcal{N}(\mathbf{x}_n | \mu_k, \Sigma_k) \right\}$$

## EM for Mixtures of Gaussians

- An elegant and powerful method for finding maximum likelihood solutions for Gaussian Mixture Model is called the *expectation-maximization* algorithm, or EM algorithm.
- Find  $\pi_k$ ,  $\mu_k$  and  $\Sigma_k$ .
- Consider the gradients of log-likelihood function w.r.t these parameters.

$$\ln p(\mathbf{X}|\pi, \mu, \Sigma) = \sum_{n=1}^N \ln \left\{ \sum_{k=1}^K \pi_k \mathcal{N}(\mathbf{x}_n | \mu_k, \Sigma_k) \right\}$$

$$\mathcal{N}(\mathbf{x}|\mu, \Sigma) = \frac{1}{(2\pi)^{D/2}} \frac{1}{|\Sigma|^{1/2}} \exp \left( -\frac{1}{2} (\mathbf{x} - \mu)^T \Sigma^{-1} (\mathbf{x} - \mu) \right)$$

## EM for Mixtures of Gaussians: $\mu_k$

- Setting the derivatives of  $\ln p(\mathbf{X}|\pi, \mu, \Sigma)$  w.r.t  $\mu_k$  to 0:

$$0 = - \sum_{n=1}^N \underbrace{\frac{\pi_k \mathcal{N}(\mathbf{x}_n | \mu_k, \Sigma_k)}{\sum_j \pi_j \mathcal{N}(\mathbf{x} | \mu_j, \Sigma_j)}}_{\gamma_k(\mathbf{x}_n)} \Sigma_k (\mathbf{x}_n - \mu_k)$$

- We get:

$$\mu_k = \frac{1}{N_k} \sum_{n=1}^N \gamma_k(\mathbf{x}_n) \mathbf{x}_n$$

where  $N_k = \sum_{n=1}^N \gamma_k(\mathbf{x}_n)$

## EM for Mixtures of Gaussians: $\mu_k$

We have:

$$\begin{aligned}\mu_k &= \frac{1}{N_k} \sum_{n=1}^N \gamma_k(\mathbf{x}_n) \mathbf{x}_n \\ N_k &= \sum_{n=1}^N \gamma_k(\mathbf{x}_n)\end{aligned}$$

- $\gamma_k(\mathbf{x}_n)$ : posterior probability  $p(k|\mathbf{x}_n)$ , *responsibility* that component  $k$  takes for “explaining” the observation  $\mathbf{x}_n$ .
- $N_k$  as the effective number of points assigned to cluster  $k$ .

## EM for Mixtures of Gaussians: $\mu_k$

We have:

$$\mu_k = \frac{1}{N_k} \sum_{n=1}^N \gamma_k(\mathbf{x}_n) \mathbf{x}_n$$

$$N_k = \sum_{n=1}^N \gamma_k(\mathbf{x}_n)$$

- $\gamma_k(\mathbf{x}_n)$ : posterior probability  $p(k|\mathbf{x}_n)$ , *responsibility* that component  $k$  takes for “explaining” the observation  $\mathbf{x}_n$ .
- $N_k$  as the effective number of points assigned to cluster  $k$ .
- $\mu_k$  for the  $k$ -th Gaussian component is obtained by taking a weighted mean of **ALL** of the points in the data set, in which the weighting factor for data point  $\mathbf{x}_n$  is given by the posterior probability  $\gamma_k(\mathbf{x}_n)$  that component  $k$  was responsible for generating  $\mathbf{x}_n$ .

## EM for Mixtures of Gaussians: $\Sigma_k$

- Setting the derivatives of  $\ln p(\mathbf{X}|\pi, \mu, \Sigma)$  w.r.t  $\Sigma_k$  to 0
- We get:

$$\Sigma_k = \frac{1}{N_k} \sum_{n=1}^N \gamma_k(\mathbf{x}_n)(\mathbf{x}_n - \mu_k)(\mathbf{x}_n - \mu_k)^T$$

$$N_k = \sum_{n=1}^N \gamma_k(\mathbf{x}_n)$$

## EM for Mixtures of Gaussians: $\Sigma_k$

- Setting the derivatives of  $\ln p(\mathbf{X}|\pi, \mu, \Sigma)$  w.r.t  $\Sigma_k$  to 0
- We get:

$$\Sigma_k = \frac{1}{N_k} \sum_{n=1}^N \gamma_k(\mathbf{x}_n)(\mathbf{x}_n - \mu_k)(\mathbf{x}_n - \mu_k)^T$$

$$N_k = \sum_{n=1}^N \gamma_k(\mathbf{x}_n)$$

- Each data point weighted by the corresponding posterior probability. Denominator given by the effective number of points associated with the corresponding component.

## EM for Mixture of Gaussians: $\pi_k$

- Finally, we maximize  $\ln p(\mathbf{X}|\pi, \mu, \Sigma)$  w.r.t  $\pi_k$
-

## EM for Mixture of Gaussians: $\pi_k$

- Finally, we maximize  $\ln p(\mathbf{X}|\pi, \mu, \Sigma)$  w.r.t  $\pi_k$
- We know that  $\sum_k \pi_k = 1$

## EM for Mixture of Gaussians: $\pi_k$

- Finally, we maximize  $\ln p(\mathbf{X}|\pi, \mu, \Sigma)$  w.r.t  $\pi_k$
- We know that  $\sum_k \pi_k = 1$
- Using a Lagrange multiplier and maximizing the following quantity:

$$\ln p(X|\pi, \mu, \Sigma) + \lambda \left( \sum_{k=1}^K \pi_k - 1 \right)$$

## EM for Mixture of Gaussians: $\pi_k$

- Finally, we maximize  $\ln p(\mathbf{X}|\pi, \mu, \Sigma)$  w.r.t  $\pi_k$
- We know that  $\sum_k \pi_k = 1$
- Using a Lagrange multiplier and maximizing the following quantity:

$$\ln p(X|\pi, \mu, \Sigma) + \lambda \left( \sum_{k=1}^K \pi_k - 1 \right)$$

- Solve for  $\lambda$  and  $\pi_k$ , we get:

$$\pi_k = \frac{N_k}{N} = \frac{\sum_{n=1}^N \gamma_k(\mathbf{x}_n)}{N}$$

- The average responsibility which that component takes for explaining the data points.

## EM algorithm: initialization

- Initialize the means  $\mu_k$ , covariances  $\Sigma_k$  and mixing coefficient  $\pi_k$ , and evaluate the initial value of the log likelihood.

## EM algorithm: E step

- **E step.** Evaluate the responsibilities using the current parameter values:

$$\gamma_k(\mathbf{x}) = p(k|\mathbf{x}) = \frac{\pi_k \mathcal{N}(\mathbf{x}|\mu_k, \Sigma_k)}{\sum_{j=1}^K \pi_j \mathcal{N}(\mathbf{x}|\mu_j, \Sigma_j)}$$

## EM algorithm: M step

- **M step.** Re-estimate the parameters using the current responsibilities (where  $N_k = \sum_{n=1}^N \gamma_k(\mathbf{x}_n)$ )

$$\mu_k = \frac{1}{N_k} \sum_{n=1}^N \gamma_k(\mathbf{x}_n) \mathbf{x}_n$$

$$\Sigma_k = \frac{1}{N_k} \sum_{n=1}^N \gamma_k(\mathbf{x}_n) (\mathbf{x}_n - \mu_k) (\mathbf{x}_n - \mu_k)^T$$

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

## EM algorithm: Check convergence

- Evaluate the log likelihood

$$\ln p(\mathbf{X}|\pi, \mu, \Sigma) = \sum_{n=1}^N \ln \left\{ \sum_{k=1}^K \pi_k \mathcal{N}(\mathbf{x}_n | \mu_k, \Sigma_k) \right\}$$

check for convergence of either the parameters or the log likelihood.

## EM for GMM: example

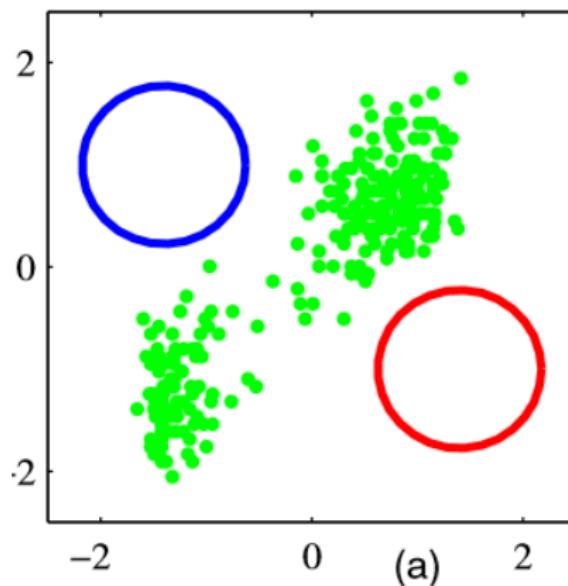


Figure: Initialization: initialize the means, covariance and mixing coefficient [C. Bishop, PRML]

## EM for GMM: example

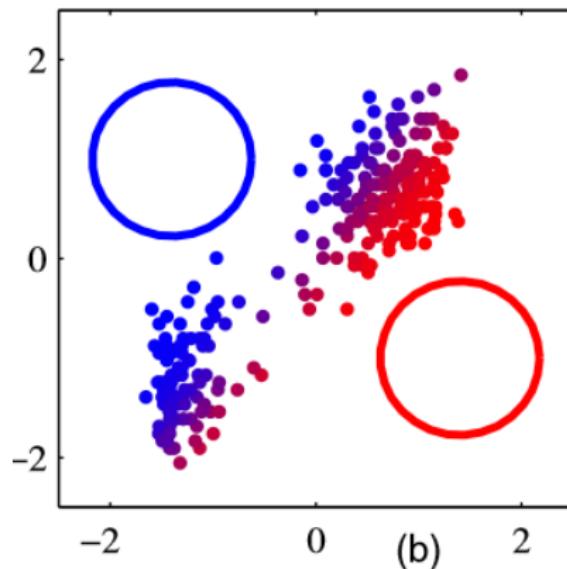
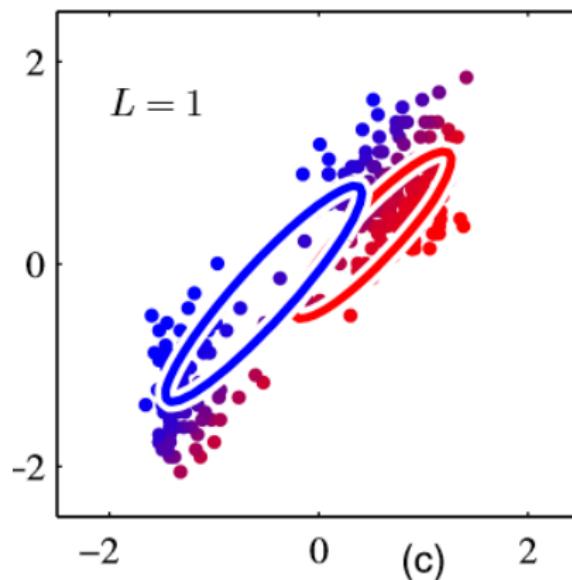


Figure: **E step:** use the current values for the parameters to evaluate the posterior probabilities, or responsibilities  $\gamma_k(\mathbf{x}_n)$ . [C. Bishop, PRML]

## EM for GMM: example



**Figure: M step:** use the posterior  $\gamma_k(\mathbf{x}_n)$  to re-estimate the means, covariances, and mixing coefficients [C. Bishop, PRML]

## EM for GMM: example

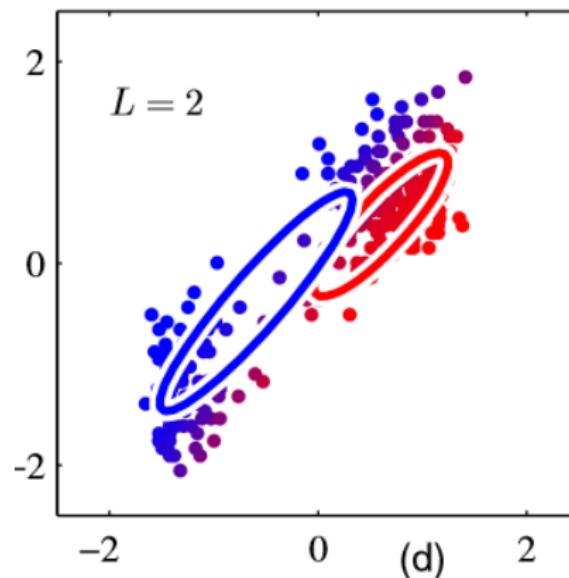


Figure: After 2 complete cycle of EM. [C. Bishop, PRML]

## EM for GMM: example

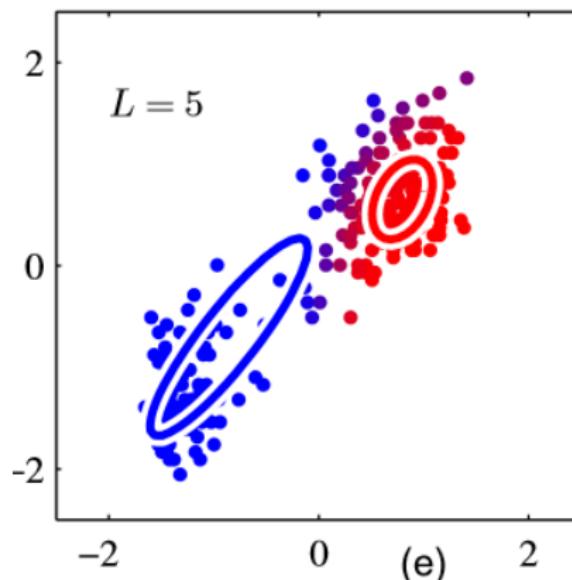


Figure: After 5 complete cycle of EM. [C. Bishop, PRML]

## EM for GMM: example

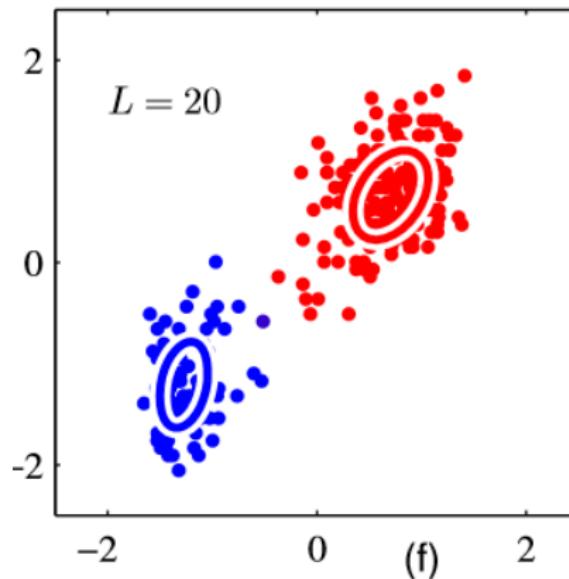


Figure: After 20 complete cycle of EM. [C. Bishop, PRML]

## K means VS GMM

- K means: **hard assignment**, each data point is associated uniquely with one cluster.

## K means VS GMM

- K means: **hard assignment**, each data point is associated uniquely with one cluster.
- EM for GMM: **soft assignment**, based on the posterior probabilities.
  - K-means as special case: consider **same and infinitely small** variance for each Gaussian component.
  - K-means results can be used as the initialization for EM algorithm.

## K means VS GMM

- K means: **hard assignment**, each data point is associated uniquely with one cluster.
- EM for GMM: **soft assignment**, based on the posterior probabilities.
  - K-means as special case: consider **same and infinitely small** variance for each Gaussian component.
  - K-means results can be used as the initialization for EM algorithm.

Different cluster analysis results on "mouse" data set:

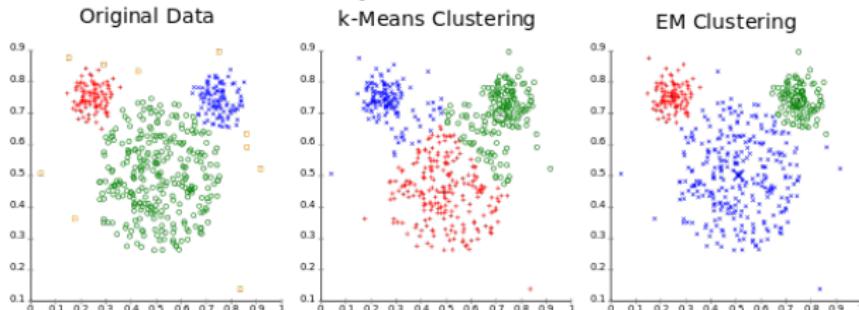


Figure: From Wikipedia.

## Acknowledgement and Further Reading

part of slides are taken from Dr. Y. Ning's Spring 19 offering of CS-559.

Further Reading:

Chapter 2.3.9, 9.1, 9.2 of *Pattern Recognition and Machine Learning* by C. Bishop.