[80245013 Machine Learning, Fall, 2019]

# Unsupervised Learning
## Clustering

**Jun Zhu**
dcszj@mail.tsinghua.edu.cn
http://ml.cs.tsinghua.edu.cn/~jun
State Key Lab of Intelligent Technology & Systems
Tsinghua University

October 8, 2019

# Unsupervised Learning

- Task: learn an explanatory function $f(x),\ x \in \mathcal{X}$
- Aka "Learning without a teacher"

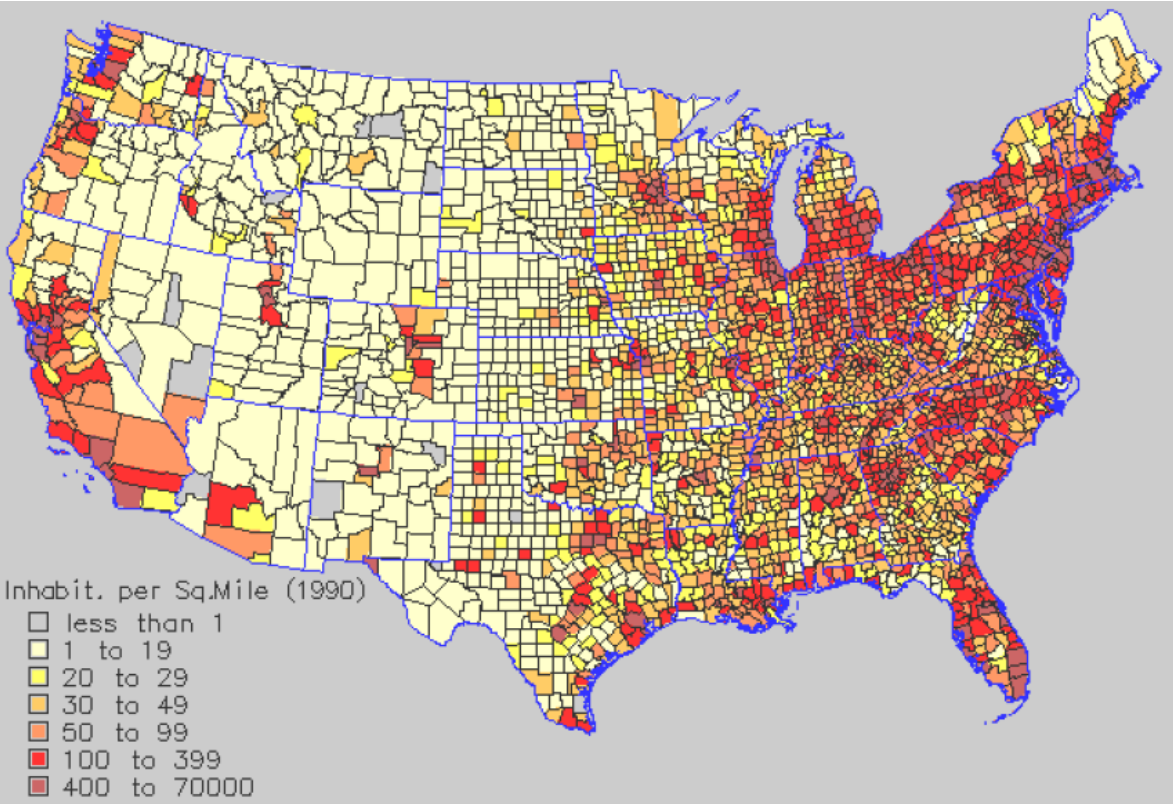
**Feature** space $\mathcal{X}$

Words in documents

$\Rightarrow$ Word distribution (probability of a word)

- No training/test split

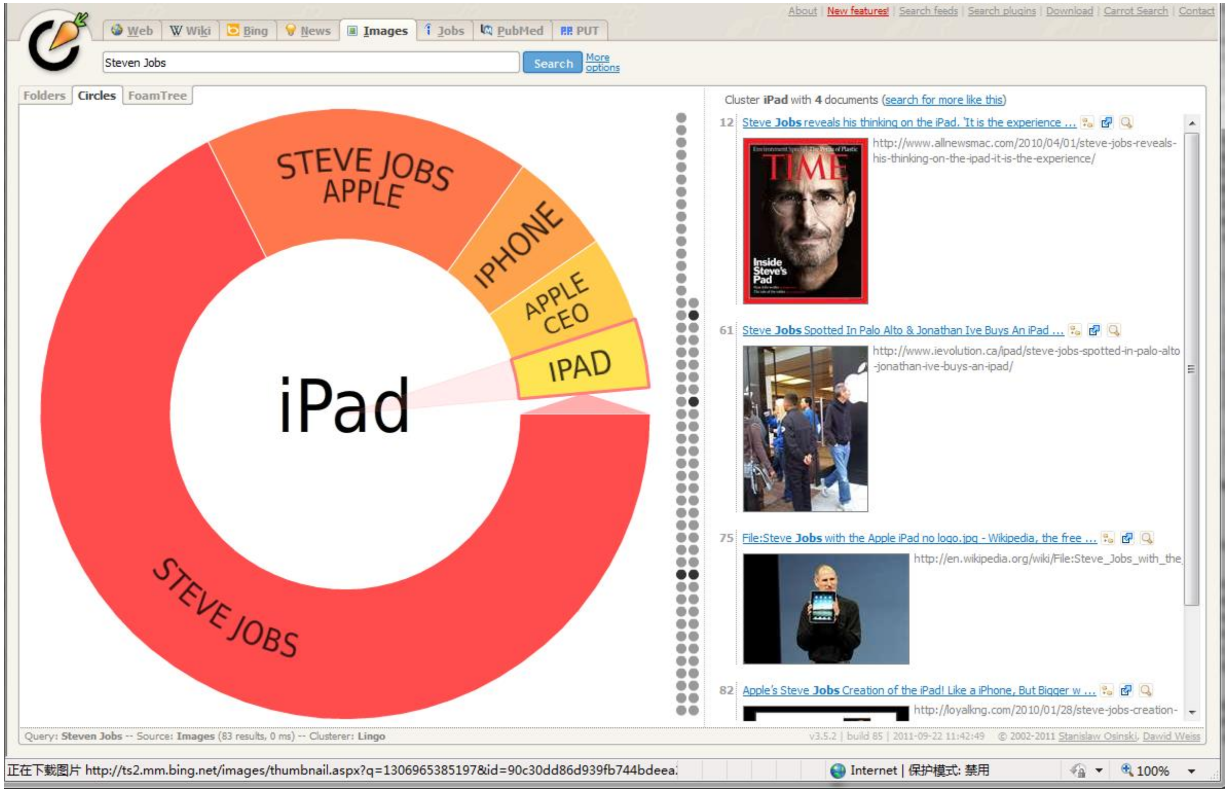
# Unsupervised Learning – density estimation




**Feature** space $\mathcal{X}$
geographical information of a location

Density function
$f(x), \; x \in \mathcal{X}$

# Unsupervised Learning – clustering




http://search.carrot2.org/stable/search

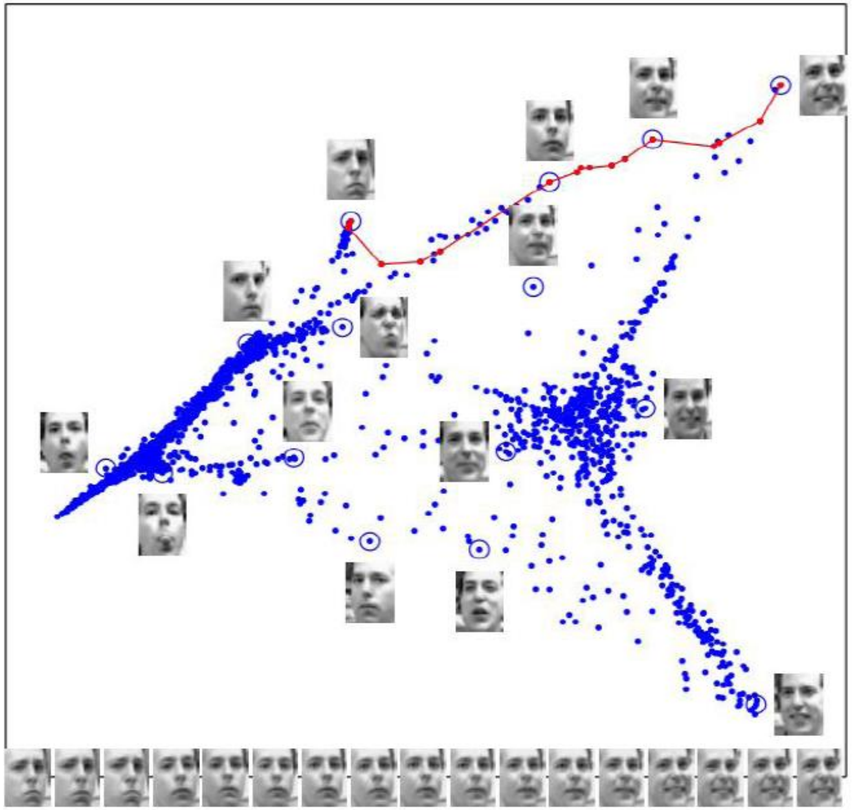
**Feature** space $\mathcal{X}$
Attributes (e.g., pixels & text) of images

Cluster assignment function
$$f(x),\ \ x \in \mathcal{X}$$

# Unsupervised Learning – dimensionality reduction

Images have thousands or millions of pixels

Can we give each image a coordinate, such that similar images are near each other ?



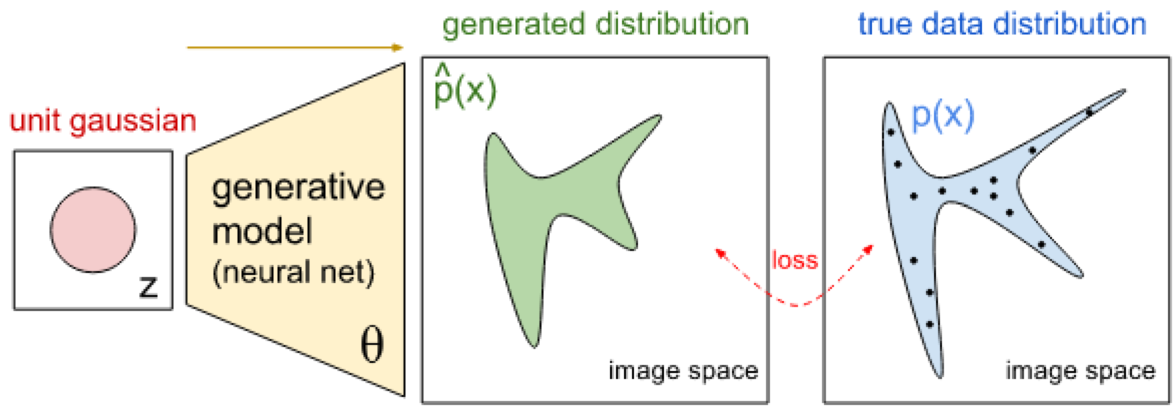
**Feature** space $\mathcal{X}$
pixels of images

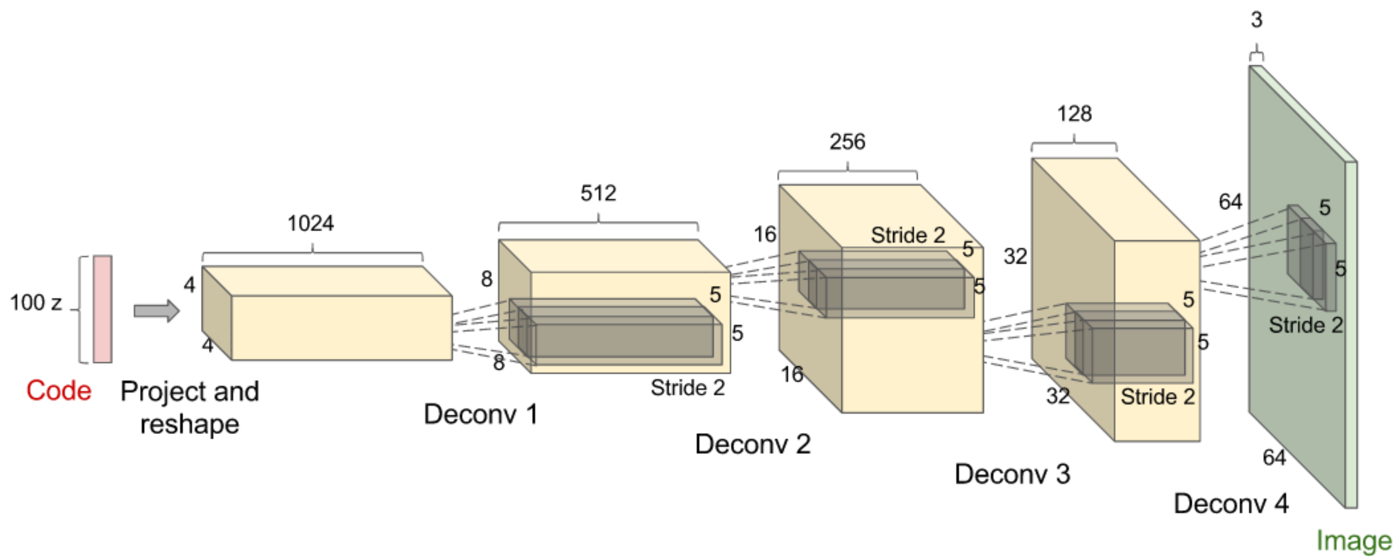
Coordinate function in 2D space

$$f(x), \;\; x \in \mathcal{X}$$

# Deep Generative Models

- Learn a generative model



# Clustering

# (K-Means, Gaussian Mixtures)

# What is clustering?

- Clustering: the process of grouping a set of objects into classes of similar objects
  - High intra-class similarity
  - Low inter-class similarity

- A common and important task that finds many applications in science, engineering, information science, etc
  - Group genes that perform the same function
  - Group individuals that has similar political view
  - Categorize documents of similar topics
  - Identify similar objects from pictures
  - …

# The clustering problem

- **Input**: training data $D = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$, where $\mathbf{x} \in \mathbb{R}^d$, integer $K$ clusters
- **Output**: a set of clusters $C_1, \ldots, C_K$

## Machine learning

From Wikipedia, the free encyclopedia

*For the journal, see Machine Learning (journal).*

*See also: Pattern recognition*

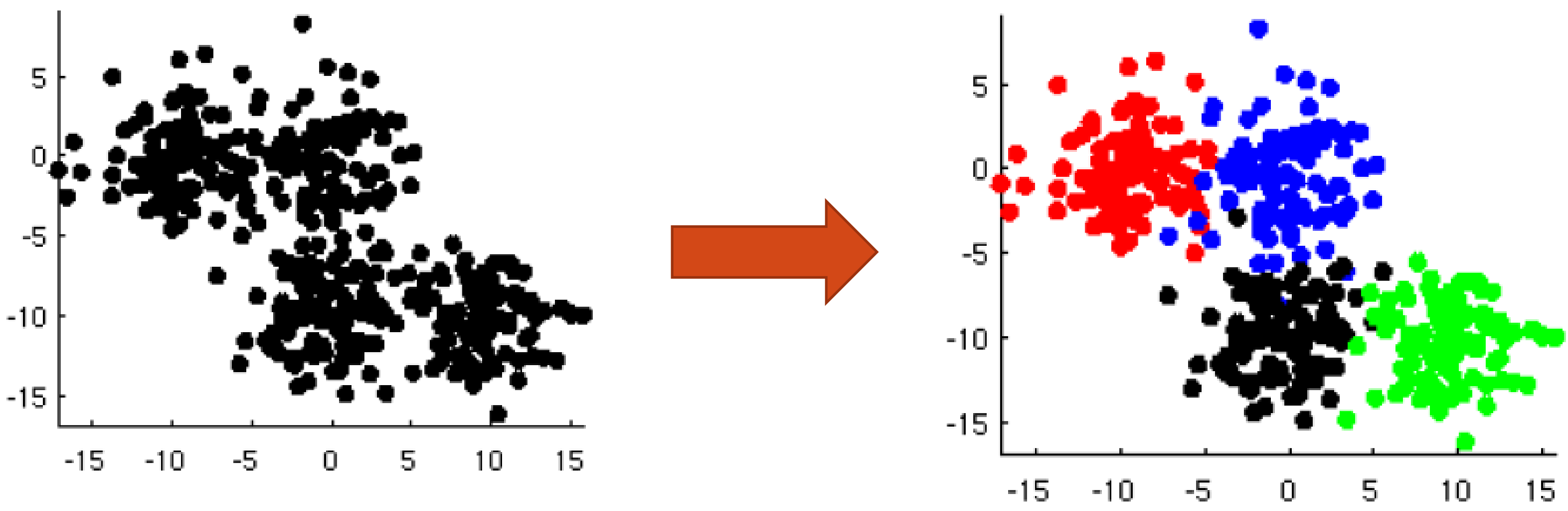
**Machine learning** is a scientific discipline that explores the construction and study of algorithms that can learn from data.[1] Such algorithms operate by building a model from example inputs and using that to make predictions or decisions,[2]:2 rather than following strictly static program instructions. Machine learning is closely related to and often overlaps with computational statistics; a discipline which also specializes in prediction-making.

$$\left\{\begin{matrix} 4 \\ 4 \\ \vdots \\ 0 \\ \vdots \\ 1 \end{matrix}\right\} \qquad \left\{\begin{matrix} \text{machine} \\ \text{learning} \\ \vdots \\ \text{JMLR} \\ \vdots \\ \text{prediction} \end{matrix}\right\}$$

**Word Vector Space** **Vocabulary**

# The clustering problem

- **Input**: training data $D = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ , where $\mathbf{x} \in \mathbb{R}^d$ , integer $K$ clusters
- **Output**: a set of clusters $C_1, \ldots, C_K$

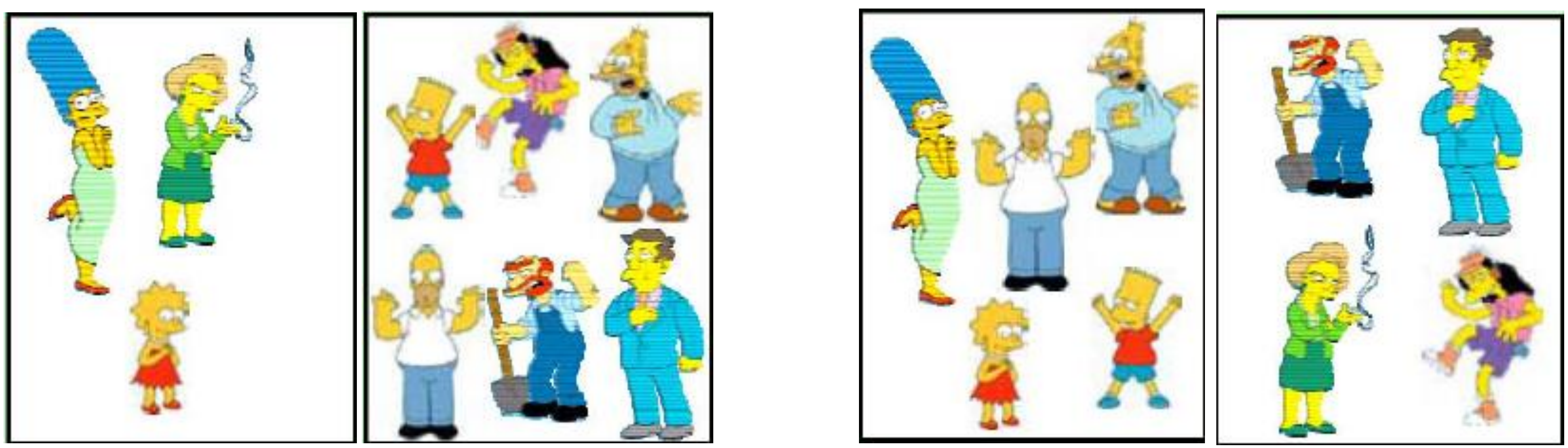


# Issues for clustering

- What is a natural grouping among these objects?
  - Definition of "groupness"
- What makes objects "related"?
  - Definition of "similarity/distance"
- Representation for objects
  - Vector space? Normalization?
- How many clusters?
  - Fixed a priori?
  - Completely data driven?
- Clustering algorithms
  - Partitional algorithms
  - Hierarchical algorithms
- Formal foundation and convergence

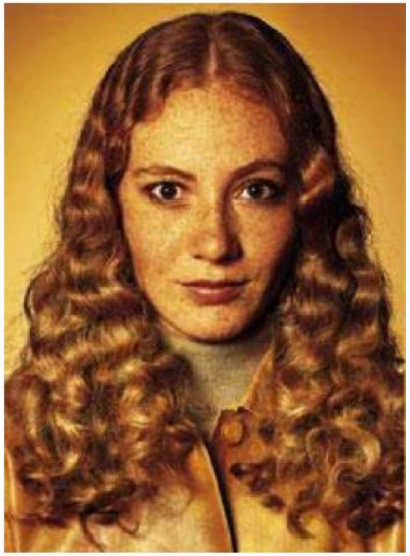
# What is a natural grouping among objects?

**Clustering is subjective**



Females | Males | Simpson's Family | School Employees

# What is similarity?

- The real meaning of similarity is a philosophical question.
- Depends on representation and algorithm. For many rep./alg., easier to think in terms of distance between vectors

# Desirable distance measure properties

- $d(A,B) = d(B,A)$ Symmetry
  - Otherwise you could claim "Alex looks like Bob, but Bob looks nothing like Alex"
- $d(A,A) = 0$ Constancy of Self-Similarity
  - Otherwise you could claim "Alex looks more like Bob, than Bob does"
- $d(A,B) = 0$ iff $A=B$ Positivity Separation
  - Otherwise there are objects that are different, but you can't tell apart
- $d(A,B) \leq d(A,C)+d(B,C)$ Triangular Inequality
  - Otherwise you could claim "Alex is very like Bob, and Alex is very like Carl, but Bob is very unlike Carl"

# Minkowski Distance

$$dist(\mathbf{x},\mathbf{y}) = \sqrt[r]{\sum_{i=1}^{d} |x_i - y_i|^r}$$

- Common Minkowski distances
  - Euclidean distance ($r$=2):

$$dist(\mathbf{x},\mathbf{y}) = \sqrt{\sum_{k=1}^{d} (x_k - y_k)^2} = \|\mathbf{x}-\mathbf{y}\|_2$$

  - Manhattan distance ($r$=1):

$$dist(\mathbf{x},\mathbf{y}) = \sum_{k=1}^{d} |x_k - y_k| = \|\mathbf{x}-\mathbf{y}\|_1$$

  - "Sup" distance ($r = \infty$):

$$dist(\mathbf{x},\mathbf{y}) = \sup_{k=1}^{d} |x_k - y_k| = \|\mathbf{x}-\mathbf{y}\|_\infty$$

# Hamming distance

- Manhattan distance is called Hamming distance when all features are binary

  - E.g., gene expression levels under 17 conditions (1-high; 0-low)

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| *GeneA* | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 1 |
| *GeneB* | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 |

  - Hamming distance: #(0 1) + #(1 0) = 4 + 1 = 5

# Correlation coefficient

- Pearson correlation coefficient

$$s(\mathbf{x}, \mathbf{y}) = \frac{(\mathbf{x} - \bar{x}\mathbf{1})^\top (\mathbf{y} - \bar{y}\mathbf{1})}{\|\mathbf{x} - \bar{x}\mathbf{1}\|_2 \|\mathbf{y} - \bar{y}\mathbf{1}\|_2}$$

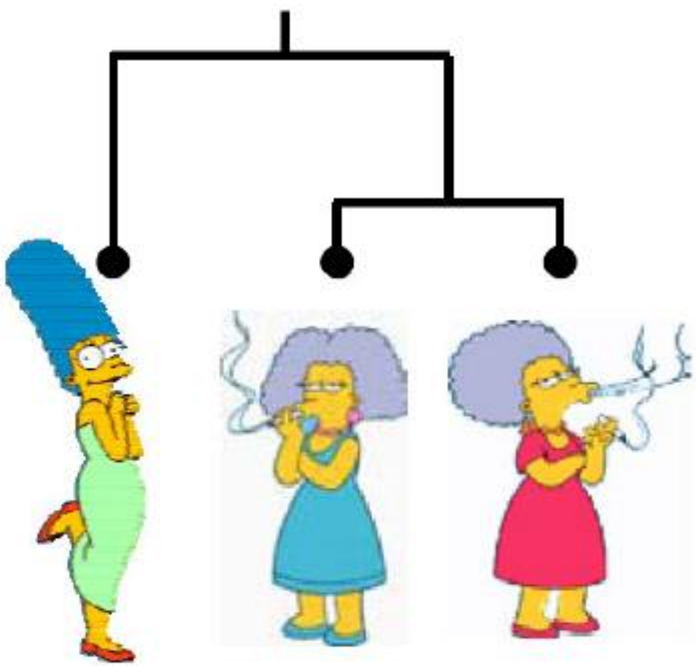
$$\text{where } \bar{x} = \frac{1}{d}\sum_i x_i, \;\; \bar{y} = \frac{1}{d}\sum_i y_i$$

- Cosine Similarity:

$$\cos(\mathbf{x}, \mathbf{y}) = \frac{\mathbf{x}^\top \mathbf{y}}{\|\mathbf{x}\|_2 \|\mathbf{y}\|_2}$$

# Edit Distance

- To measure the similarity between two objects, transform one into the other, and measure how much effort it took. The measure of effort becomes the distance measure



Marge Patty Selman

**The distance between Patty and Selma.**

Change dress color, 1 point
Change earring shape, 1 point
Change hair part, 1 point

**D(Patty,Selma) = 3**

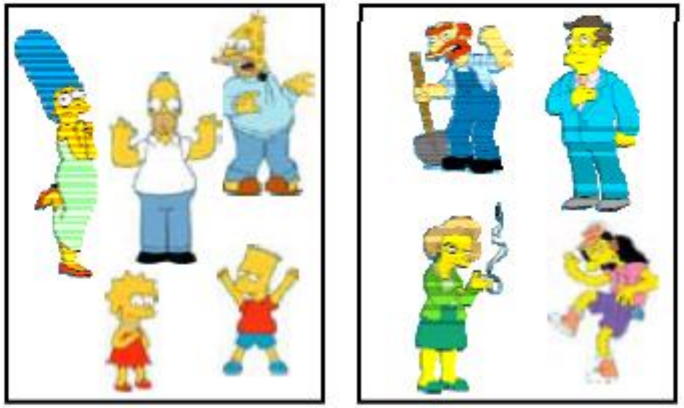
**The distance between Marge and Selma**

Change dress color, 1 point
Add earrings, 1 point
Decrease height, 1 point
Take up smoking, 1 point
Loss weight, 1 point

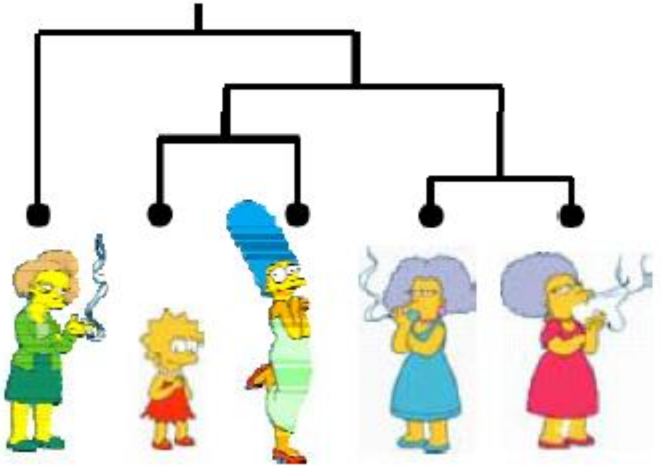
**D(Marge, Selma) = 5**

# Clustering algorithms

- Partitional algorithms
  - Usually start with a random (partial) partitioning
  - Refine it iteratively
    - K-means
    - Mixture-Model based clustering

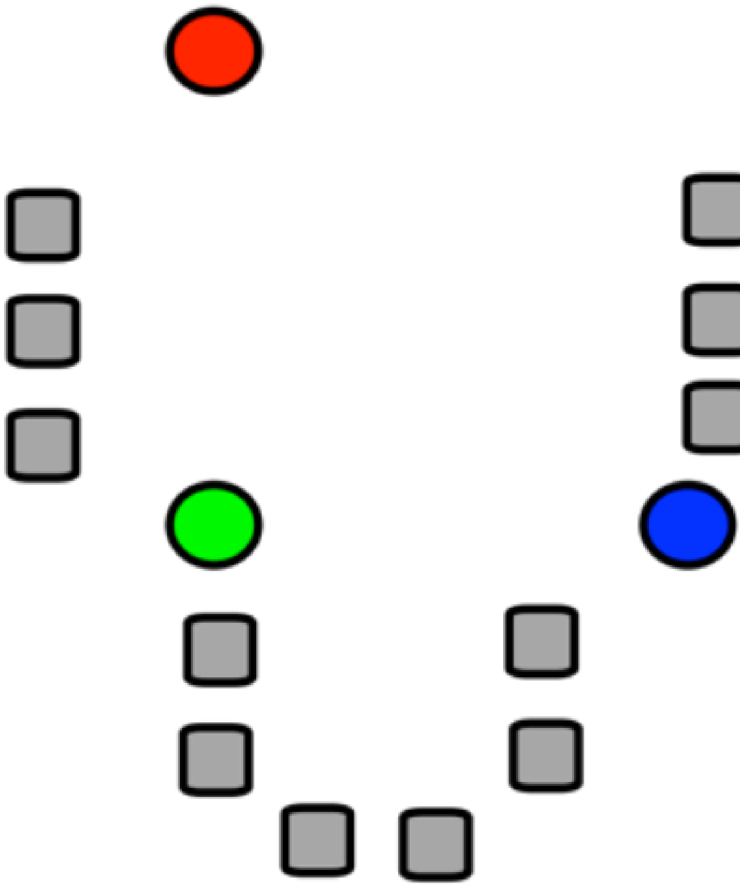


- Hierarchical algorithms
  - Bottom-up, agglomerative
  - Top-down, divisive



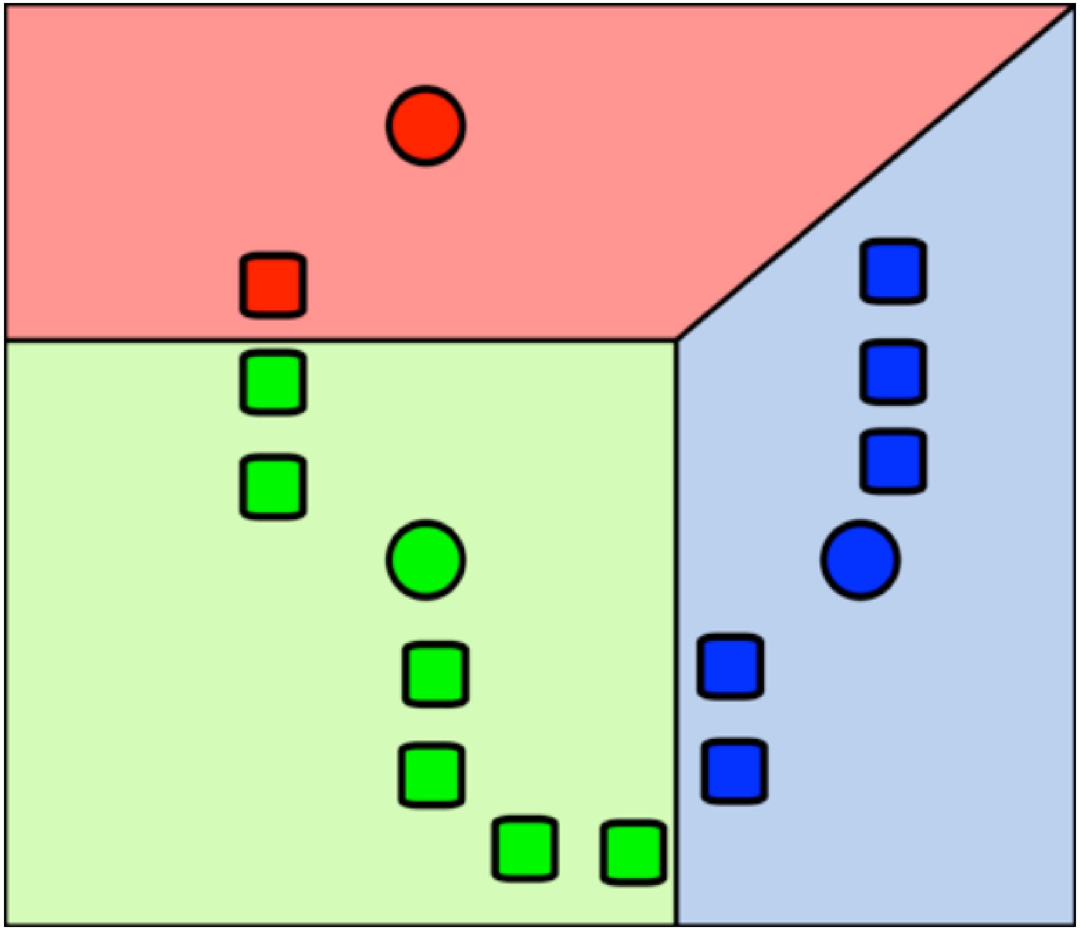
# K-means Algorithm

- 1. Initialize the centroids $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K$



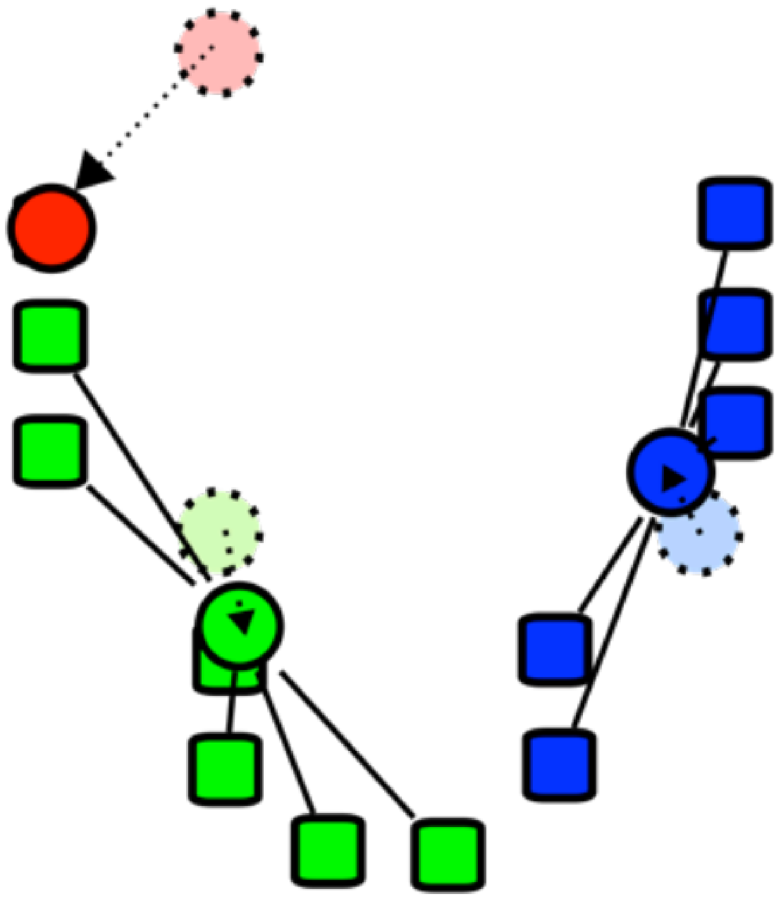
# K-means Algorithm

- 2. for each $k$, $C_k = \{i, \text{ s.t. } \mathbf{x}_i \text{ is closest to } \boldsymbol{\mu}_k\}$



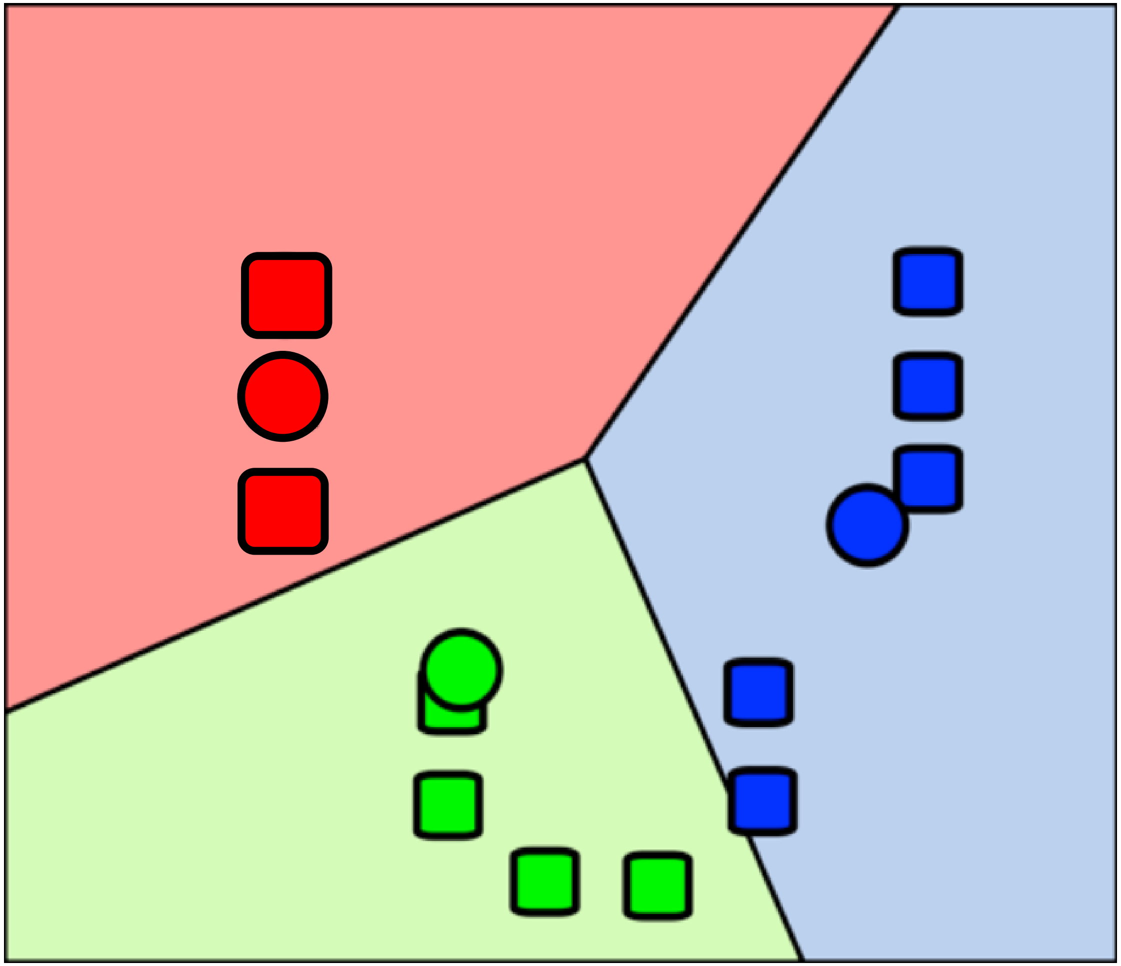
# K-means Algorithm

- 3. for each $k$, $\quad \boldsymbol{\mu}_k \leftarrow \frac{1}{|C_k|} \sum_{j \in C_k} \mathbf{x}_j$ (sample mean)



# K-means Algorithm

- Repeat until no further change in cluster assignment



# Summary of K-means Algorithm

- 1. Initialize centroids $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K$
- 2. Repeat until no change of cluster assignment
  - (1) for each $k$:

$$C_k = \{i, \text{ s.t. } \mathbf{x}_i \text{ is closest to } \boldsymbol{\mu}_k\}$$

  - (2) for each $k$:

$$\boldsymbol{\mu}_k \leftarrow \frac{1}{|C_k|} \sum_{j \in C_k} \mathbf{x}_j$$

- **Note**: each iteration requires $O(NK)$ operations

# K-means Questions

- What is it trying to optimize?
- Are we sure it will terminate?
- Are we sure it will find an optimal clustering?
- How should we start it?
- How could we automatically choose the number of centers?

# Theory: K-Means as an Opt. Problem

- The opt. problem

$$\min_{\{C_k\}_{k=1}^K} \quad \sum_{k=1}^{K} \sum_{\mathbf{x} \in C_k} \|\mathbf{x} - \boldsymbol{\mu}_k\|_2^2$$

$$\text{s.t}: \quad \boldsymbol{\mu}_k = \frac{1}{|C_k|} \sum_{\mathbf{x} \in C_k} \mathbf{x}$$

- **Theorem**: *K-means iteratively leads to a non-increasing of the objective, until local minimum is achieved*
  - *Proof ideas:*
    - *Each operation leads to non-increasing of the objective*
    - *The objective is bounded and the number of clusters is finite*

# K-means as gradient descent

- Find $K$ prototypes to minimize the ***quantization error*** (i.e., the average distance between a data to its closest prototype):

$$\min_{\{\boldsymbol{\mu}_c\}_{c=1}^{K}} \quad \sum_{i=1}^{N} \min_k \|\mathbf{x}_i - \boldsymbol{\mu}_k\|_2^2$$

  - First-order gradient descent applies
  - Newton method leads to the same update rule:

$$\boldsymbol{\mu}_k = \frac{1}{|C_k|} \sum_{\mathbf{x} \in C_k} \mathbf{x}$$

- *See [Bottou & Bengio, NIPS'95] for more details*

# Trying to find a good optimum

- **Idea 1**: Be careful about where you start
- **Idea 2**: Do many runs of k-means, each from a different random start configuration
- Many other ideas floating around.

- **Note**: *K*-means is often used to initialize other clustering methods

# Mixture of Gaussians and EM algorithm

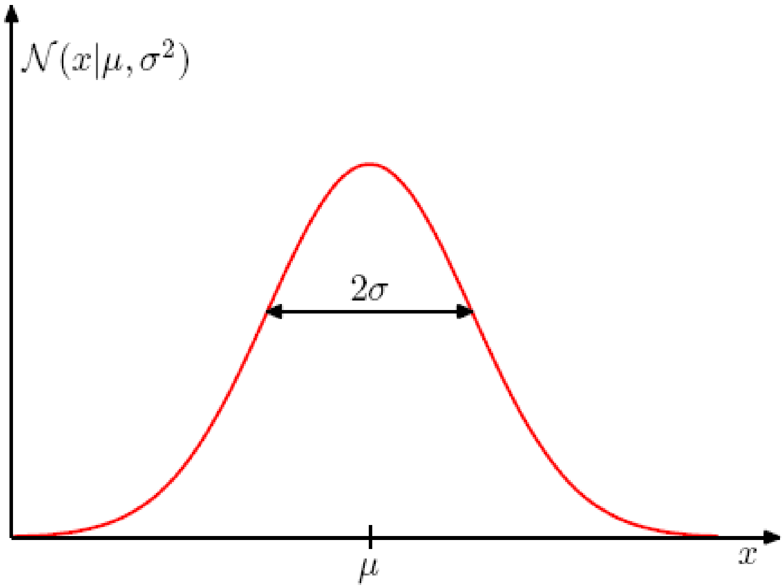
# Gaussian Distributions

Carl F. Gauss (1777 – 1855)

- Univariate Gaussian distribution

$$p(x|\mu,\sigma^2) = \frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

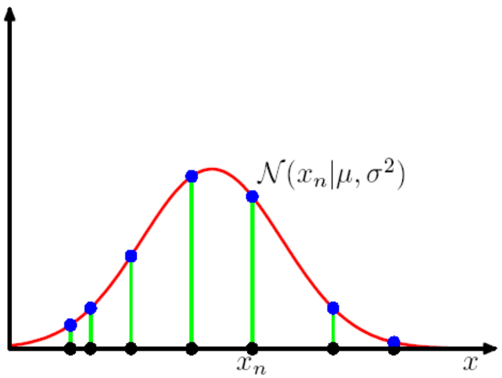
- Given parameters, we can draw samples and plot distributions



# Maximum Likelihood Estimation

- Given a data set $\mathcal{D} = \{x_1, \ldots, x_N\}$, the likelihood is

$$p(\mathcal{D}|\mu, \sigma^2) = \prod_{n=1}^{N} p(x_n|\mu, \sigma^2)$$




- MLE estimates the parameters as

$$(\mu_{\mathrm{ML}}, \sigma^2_{\mathrm{ML}}) = \underset{\mu, \sigma^2}{\operatorname{argmax}} \log p(\mathcal{D}|\mu, \sigma^2)$$

$$\mu_{\mathrm{ML}} = \frac{1}{N} \sum_{n=1}^{N} x_n$$ sample mean

$$\sigma^2_{\mathrm{ML}} = \frac{1}{N} \sum_{n=1}^{N} (x_n - \mu_{\mathrm{ML}})^2$$ sample variance

Note: MLE for the variance of a Gaussian is biased

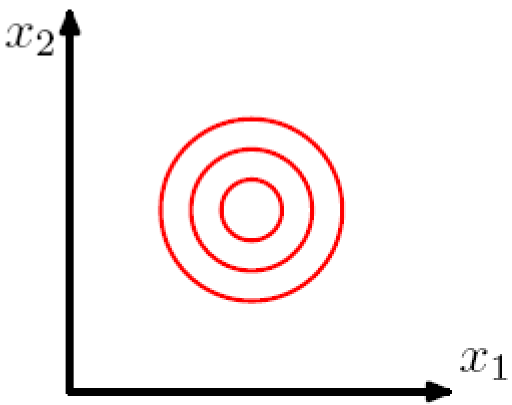
# Gaussian Distributions

Carl F. Gauss (1777 – 1855)

- $d$-dimensional multivariate Gaussian

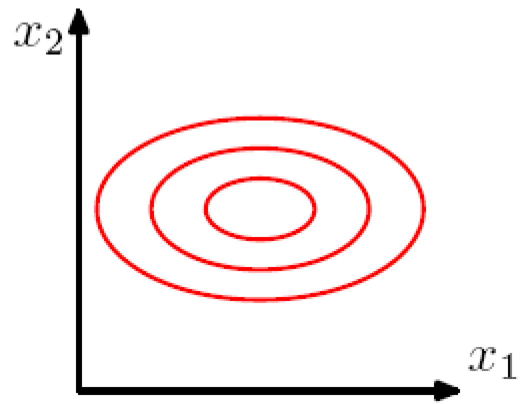
$$p(\mathbf{x}|\mu,\Sigma) = \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} \exp\Big( -\frac{1}{2}(\mathbf{x}-\mu)\Sigma^{-1}(\mathbf{x}-\boldsymbol{\mu})\Big)$$

- Given parameters, we can draw samples and plot distributions



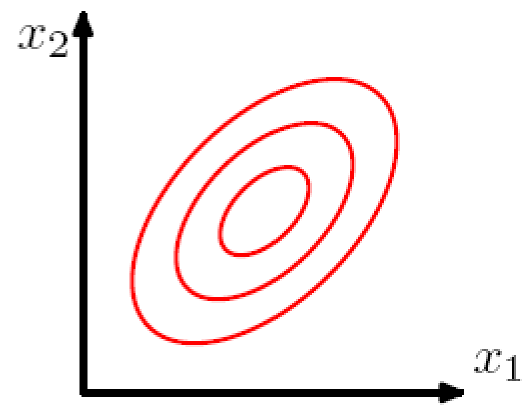

Isotropic

$$\Sigma = \sigma^2 \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$




Diagonal

$$\Sigma = \begin{bmatrix} \sigma_1^2 & 0 \\ 0 & \sigma_2^2 \end{bmatrix}$$




General

$$\Sigma = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$$

# Maximum Likelihood Estimation

- Given a data set $\mathcal{D} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$, the likelihood is

$$p(\mathcal{D}|\mu, \Sigma) = \prod_{n=1}^{N} p(\mathbf{x}_n|\mu, \Sigma)$$

- MLE estimates the parameters as

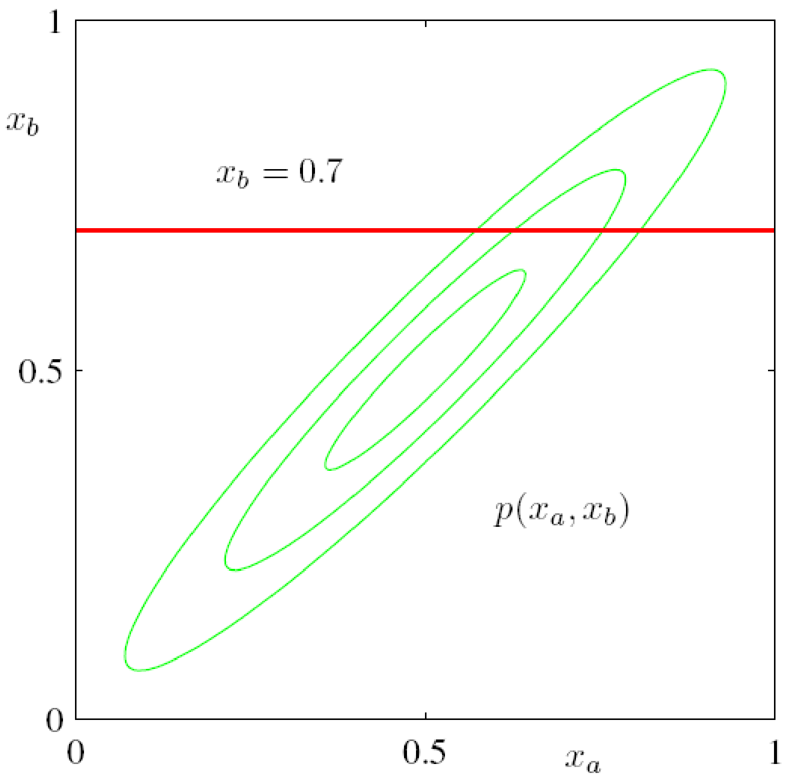
$$(\mu_{\mathrm{ML}}, \Sigma_{\mathrm{ML}}) = \underset{\mu, \Sigma}{\operatorname{argmax}} \log p(\mathcal{D}|\mu, \Sigma)$$

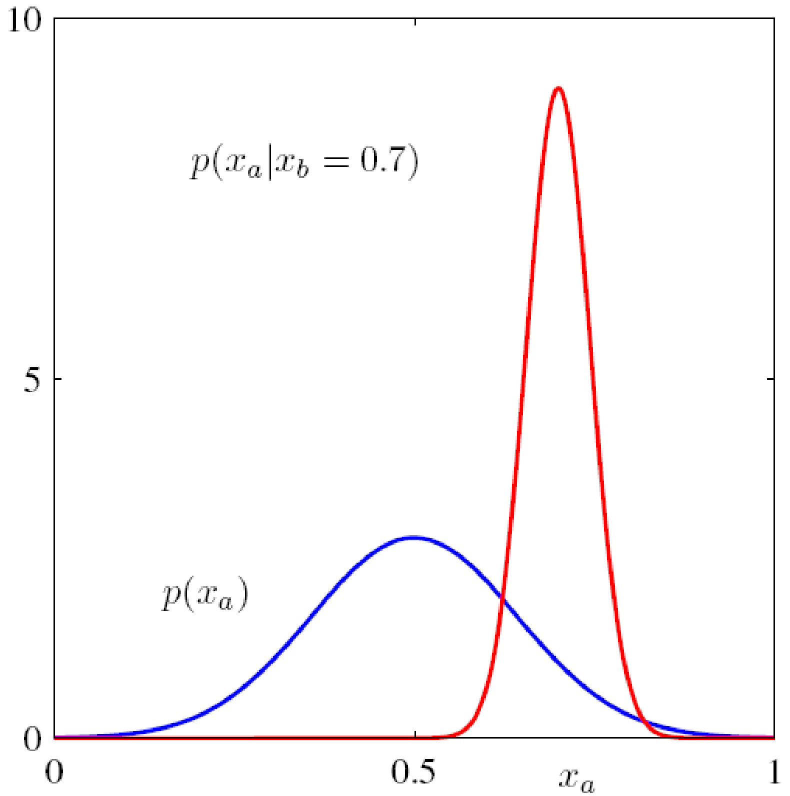
$$\mu_{\mathrm{ML}} = \frac{1}{N} \sum_{n=1}^{N} \mathbf{x}_n$$ sample mean

$$\Sigma_{\mathrm{ML}}^2 = \frac{1}{N} \sum_{n=1}^{N} (x_n - \mu_{\mathrm{ML}})(x_n - \mu_{\mathrm{ML}})^\top$$ sample covariance

# Other Nice Analytic Properties

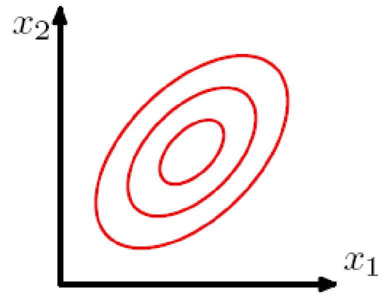
- Marginal is Gaussian
- Conditional is Gaussian



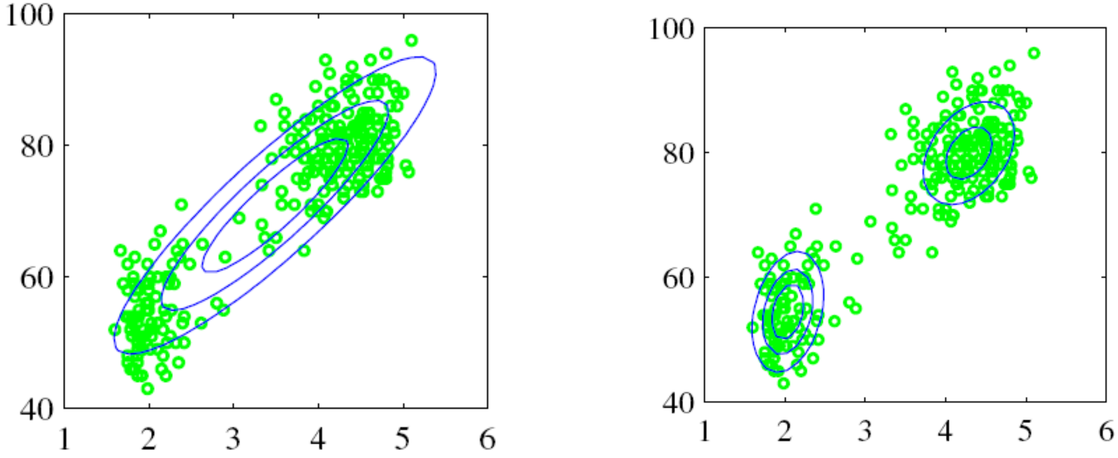



# Limitations of Single Gaussians

- Single Gaussian is unimodal



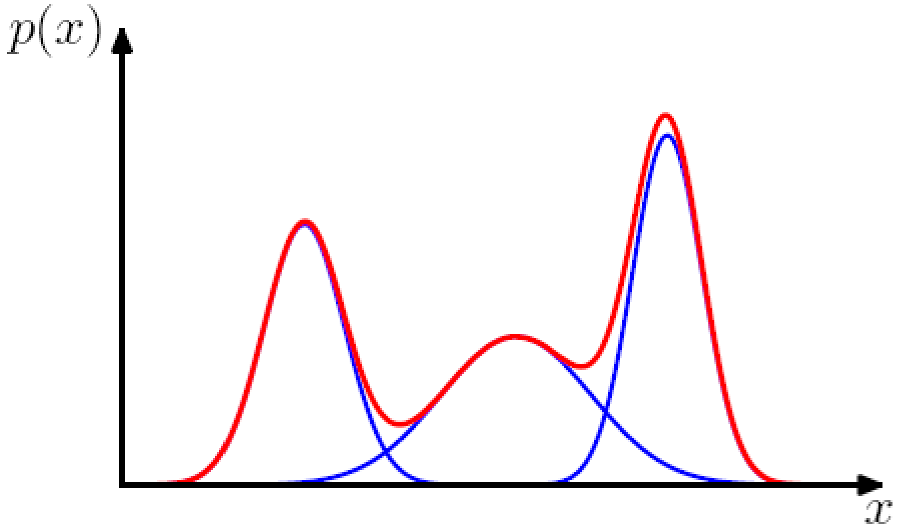

- … can’t fit well multimodal data, which is more realistic!



# Mixture of Gaussians

- A simple family of multi-modal distributions
  - treat unimodal Gaussians as basis (or component) distributions
  - superpose multiple Gaussians via linear combination

$$p(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x|\mu_k, \sigma_k^2)$$

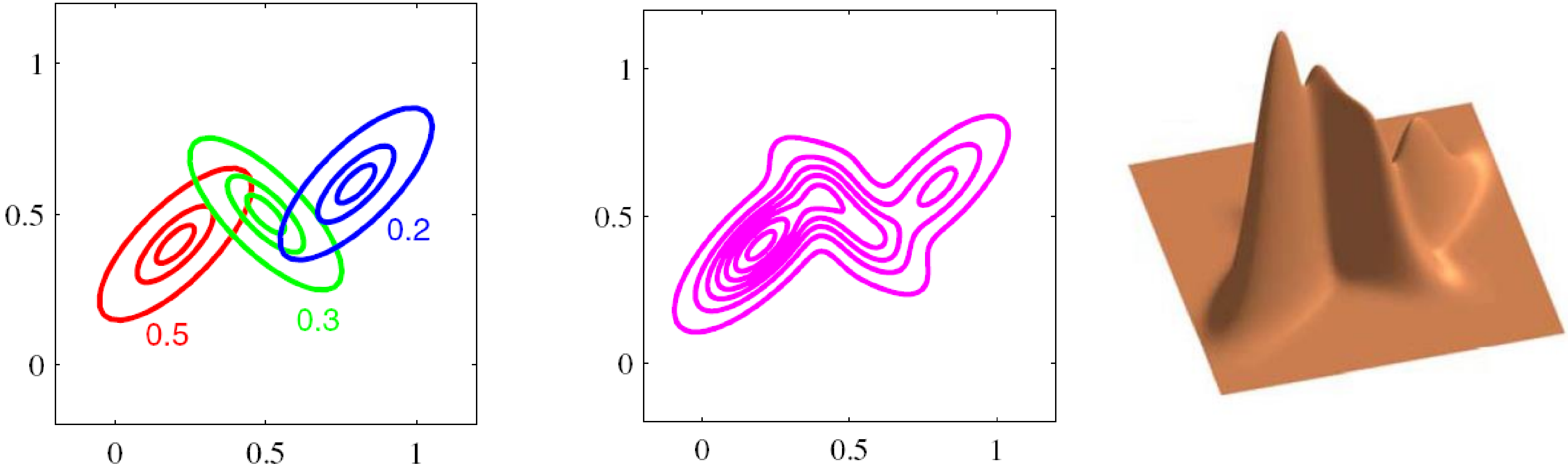


# Mixture of Gaussians

- A simple family of multi-modal distributions
  - treat unimodal Gaussians as basis (or component) distributions
  - superpose multiple Gaussians via linear combination

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

*What conditions should the mixing coefficients satisfy ?*



# MLE for Mixture of Gaussians

- Log-likelihood

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

  - this is complicated … ☹
  - … but, we know the MLE for single Gaussians are easy

- A heuristic procedure (can we iterate?)
  - allocate data into different components
  - estimate each component Gaussian analytically

# Optimal Conditions

- Some math

$$\mathcal{L}(\boldsymbol{\mu}, \Sigma) = \log p(\mathcal{D}|\boldsymbol{\mu}, \Sigma) = \sum_{n=1}^{N} \log \left( \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \Sigma_k) \right)$$

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{\mu}_k} = 0 \quad \Rightarrow \quad \sum_{n=1}^{N} \underbrace{\frac{\pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \Sigma_k)}{\sum_j \pi_j \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_j, \Sigma_j)}}_{\gamma(z_{nk})} \Sigma_k^{-1}(\mathbf{x}_n - \boldsymbol{\mu}_k) = 0$$

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

***A weighted sample mean!***

# Optimal Conditions

- Some math

$$\mathcal{L}(\boldsymbol{\mu}, \Sigma) = \log p(\mathcal{D}|\boldsymbol{\mu}, \Sigma) = \sum_{n=1}^{N} \log \left( \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \Sigma_k) \right)$$

$$\frac{\partial \mathcal{L}}{\partial \Sigma_k} = 0 \quad \Rightarrow \quad \Sigma_k = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk})(\mathbf{x}_n - \boldsymbol{\mu}_k)(\mathbf{x}_n - \boldsymbol{\mu}_k)^\top$$

***A weighted sample variance!***

# Optimal Conditions

- Some math

$$\mathcal{L}(\boldsymbol{\mu}, \Sigma) = \log p(\mathcal{D}|\boldsymbol{\mu}, \Sigma) = \sum_{n=1}^{N} \log\Big(\sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \Sigma_k)\Big)$$

**Note: constraints exist for mixing coefficients!**

$$L = \mathcal{L}(\boldsymbol{\mu}, \Sigma) + \lambda\Big(\sum_{k=1}^{K} \pi_k - 1\Big)$$

$$\frac{\partial L}{\partial \pi_k} = 0 \quad \Rightarrow \quad \sum_{n=1}^{N} \frac{\mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \Sigma_k)}{\sum_j \pi_j \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_j, \Sigma_j)} + \lambda = 0$$

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

*The ratio of data assigned to component k!*

# Optimal Conditions – summary

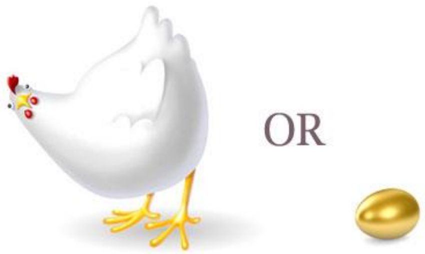
- The set of couple conditions

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

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

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

- The key factor to get them coupled




$$\gamma(z_{nk}) = \frac{\pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \Sigma_k)}{\sum_j \pi_j \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_j, \Sigma_j)}$$

- If we know $\gamma(z_{nk})$ , each component Gaussian is easy to estimate!

# The EM Algorithm

- **E-step**: estimate the responsibilities

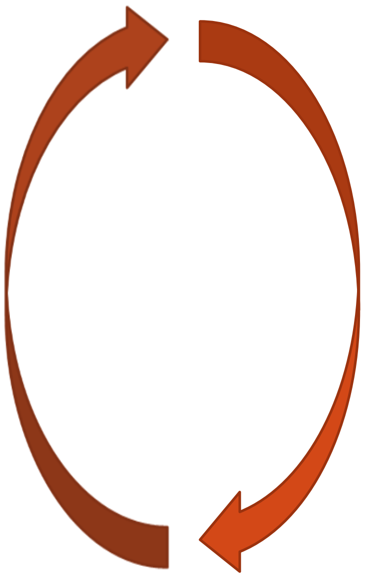
$$\gamma(z_{nk}) = \frac{\pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \Sigma_k)}{\sum_j \pi_j \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_j, \Sigma_j)}$$

- **M-step**: re-estimate the parameters

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

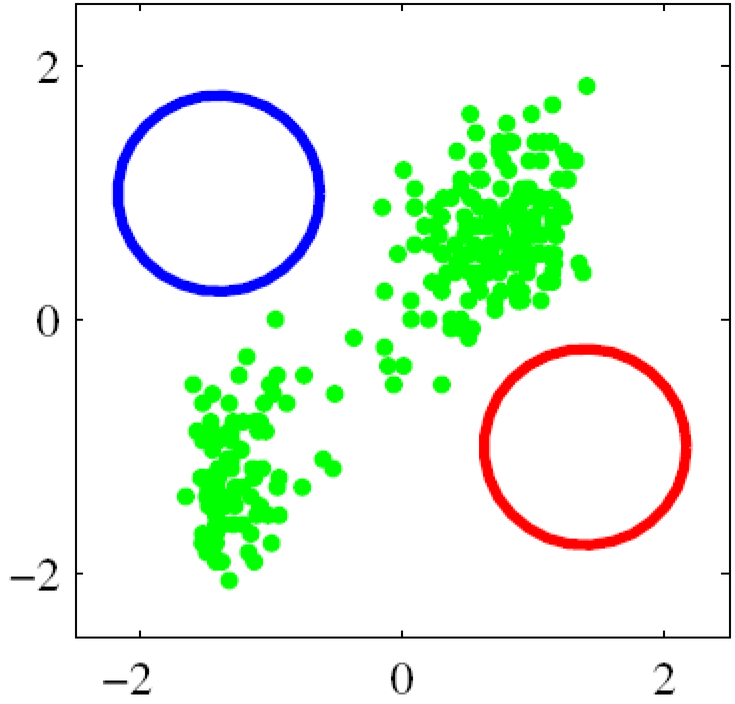
$$\Sigma_k = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk}) (\mathbf{x}_n - \boldsymbol{\mu}_k)(\mathbf{x}_n - \boldsymbol{\mu}_k)^\top$$

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



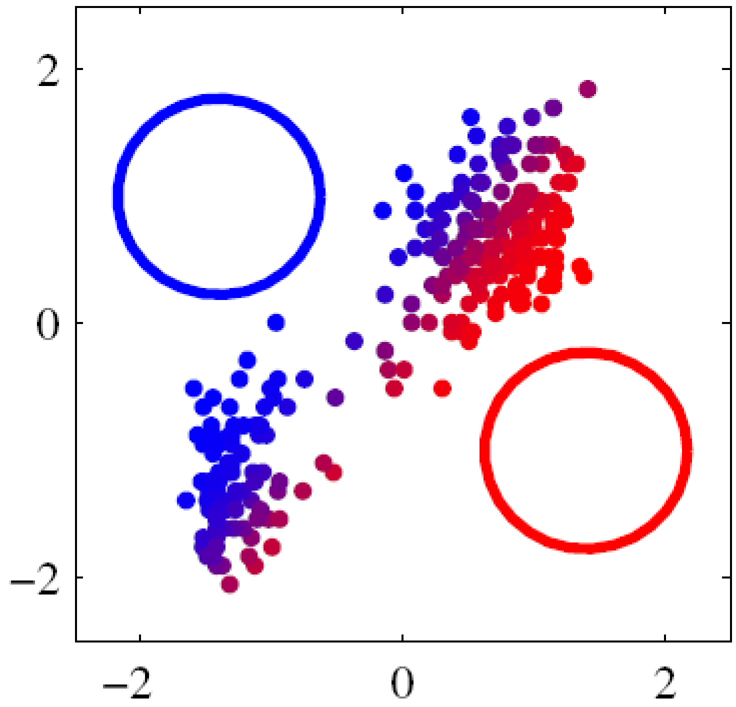
**Initialization plays a key role to succeed!**

# A Running Example



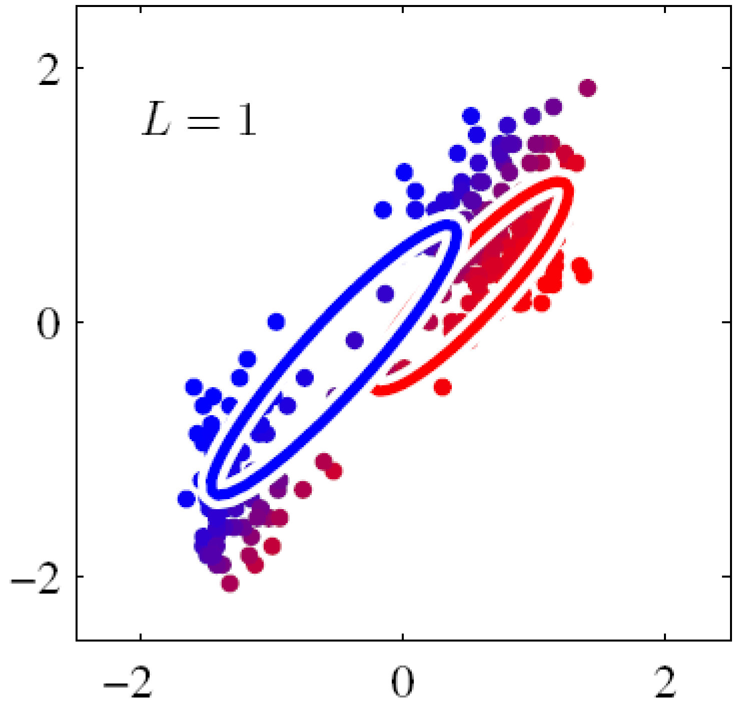

- The data and a mixture of two isotropic Gaussians

# A Running Example



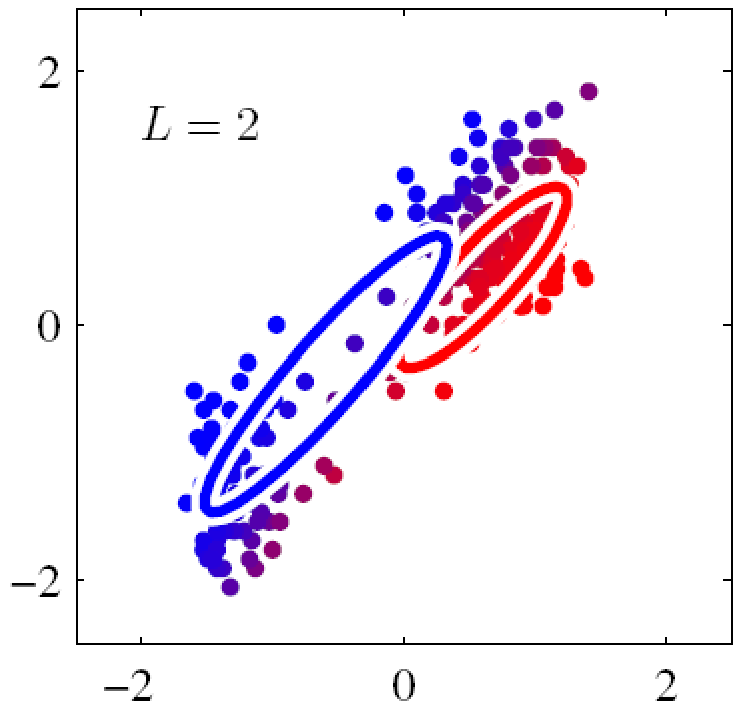

- Initial E-step

# A Running Example



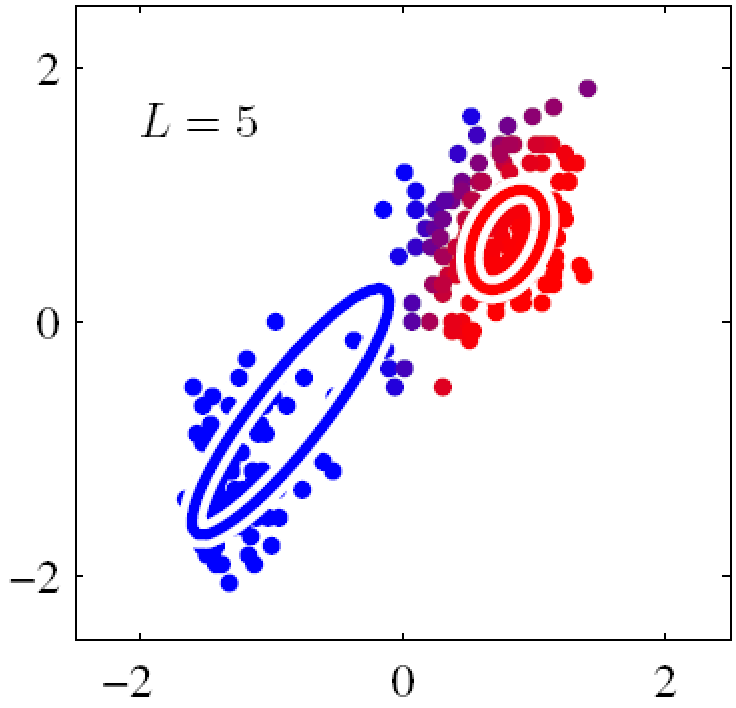

- Initial M-step

# A Running Example



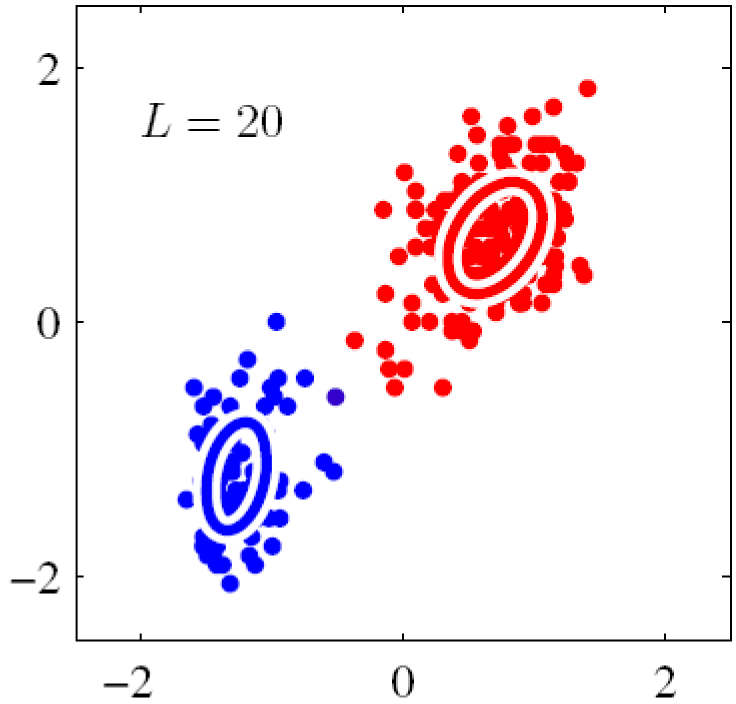

- The 2nd M-step

# A Running Example




- The 5th M-step

# A Running Example

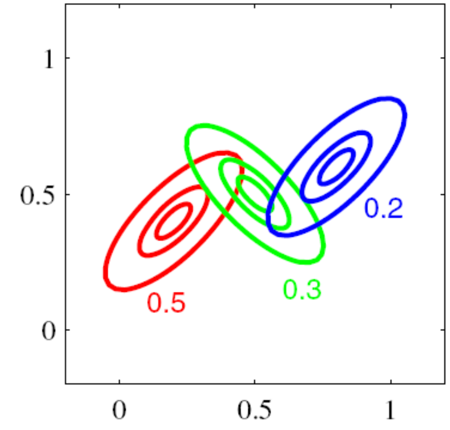



- The 20th M-step

# Theory

- Let's take the latent variable view of mixture of Gaussians

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

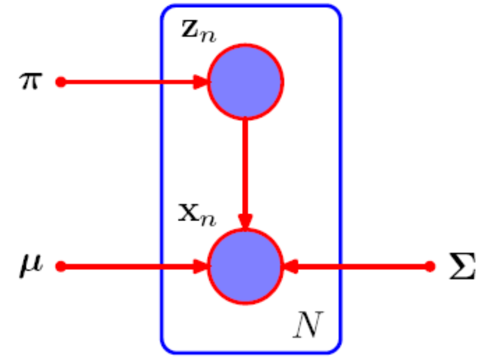



  - Indicator (selecting) variable

$$\mathbf{z} = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}$$

$$p(\mathbf{x}, \mathbf{z}) = \prod_{k=1}^{K} \pi_k^{z_k} \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \Sigma_k)^{z_k}$$

$$p(\mathbf{x}) \stackrel{?}{=} \sum_{\mathbf{z}} p(\mathbf{x}, \mathbf{z})$$



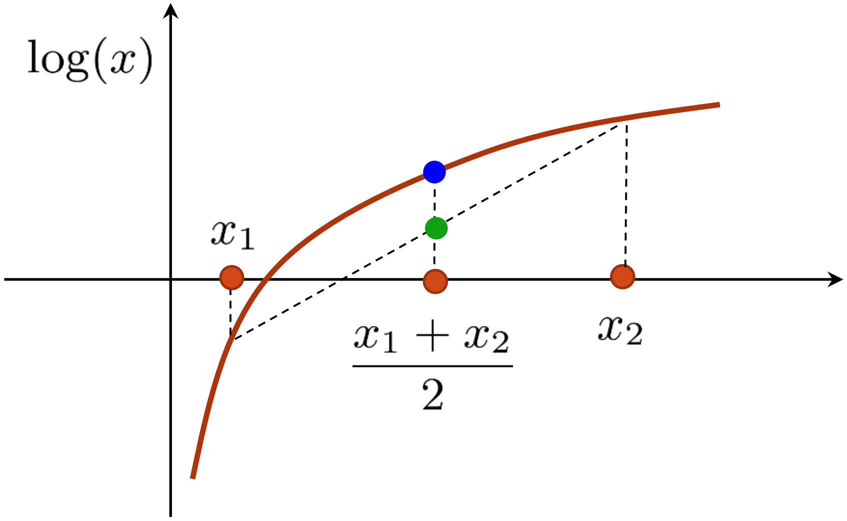

**Note: the idea of data augmentation is influential in statistics and machine learning!**

# Theory

- Re-visit the log-likelihood

$$\log p(\mathcal{D}|\Theta) = \sum_{n=1}^{N} \log \left( \sum_{\mathbf{z}_n} p(\mathbf{x}_n, \mathbf{z}_n) \right)$$

- Jensen's inequality



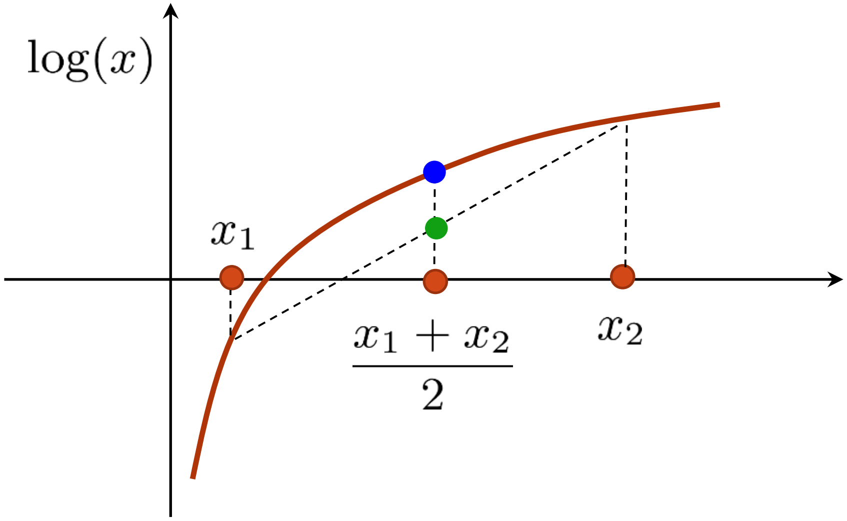

$$\boxed{\log \frac{x_1 + x_2}{2} \geq \frac{\log x_1 + \log x_2}{2}}$$

# Theory

- Re-visit the log-likelihood

$$\log p(\mathcal{D}|\Theta) = \sum_{n=1}^{N} \log \left( \sum_{\mathbf{z}_n} p(\mathbf{x}_n, \mathbf{z}_n) \right)$$

- Jensen's inequality




$$\boxed{\log \mathbb{E}_{p(x)}[x] \geq \mathbb{E}_{p(x)}[\log x]}$$

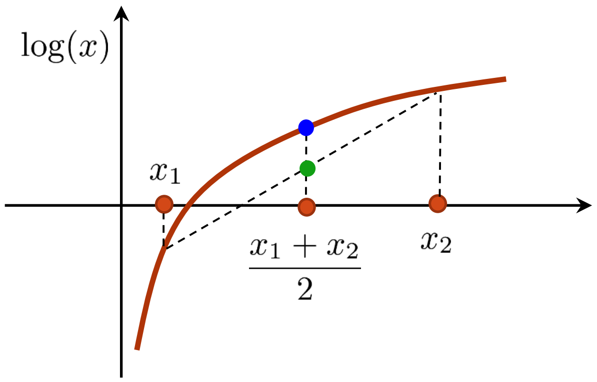
# Theory

- Re-visit the log-likelihood

$$\log p(\mathcal{D}|\Theta) = \sum_{n=1}^{N} \log \left( \sum_{\mathbf{z}_n} p(\mathbf{x}_n, \mathbf{z}_n) \right)$$

- Jensen's inequality

$$\boxed{\log \mathbb{E}_{p(x)}[x] \geq \mathbb{E}_{p(x)}[\log x]}$$




- How to apply?

$$\log p(\mathcal{D}|\Theta) = \sum_{n=1}^{N} \log \left( \sum_{\mathbf{z}_n} q(\mathbf{z}_n) \frac{p(\mathbf{x}_n, \mathbf{z}_n)}{q(\mathbf{z}_n)} \right)$$

$$\geq \sum_{n=1}^{N} \sum_{\mathbf{z}_n} q(\mathbf{z}_n) \log \left( \frac{p(\mathbf{x}_n, \mathbf{z}_n)}{q(\mathbf{z}_n)} \right)$$

# Theory

- What we have is a lower bound

$$\log p(\mathcal{D}|\Theta) \geq \sum_{n=1}^{N} \sum_{\mathbf{z}_n} q(\mathbf{z}_n) \log \left( \frac{p(\mathbf{x}_n, \mathbf{z}_n)}{q(\mathbf{z}_n)} \right) \triangleq \mathcal{L}(\Theta, q(\mathbf{Z}))$$

- What's the GAP?

$$\begin{aligned}
\mathcal{L}(\Theta, q(\mathbf{Z})) &= \sum_{n=1}^{N} \left\{ \sum_{\mathbf{z}_n} q(\mathbf{z}_n) \log p(\mathbf{x}_n, \mathbf{z}_n) - \sum_{\mathbf{z}_n} q(\mathbf{z}_n) \log q(\mathbf{z}_n) \right\} \\
&= \sum_{n=1}^{N} \left\{ \sum_{\mathbf{z}_n} q(\mathbf{z}_n) \log \left( \frac{p(\mathbf{x}_n, \mathbf{z}_n)}{p(\mathbf{x}_n)} \right) + \log p(\mathbf{x}_n) - \sum_{\mathbf{z}_n} q(\mathbf{z}_n) \log q(\mathbf{z}_n) \right\} \\
&= \log p(\mathcal{D}|\Theta) + \sum_{n=1}^{N} \left\{ \sum_{\mathbf{z}_n} q(\mathbf{z}_n) \log p(\mathbf{z}_n|\mathbf{x}_n) - \sum_{\mathbf{z}_n} q(\mathbf{z}_n) \log q(\mathbf{z}_n) \right\} \\
&= \log p(\mathcal{D}|\Theta) - \mathrm{KL}(q(\mathbf{Z}) \| p(\mathbf{Z}|\mathcal{D}))
\end{aligned}$$

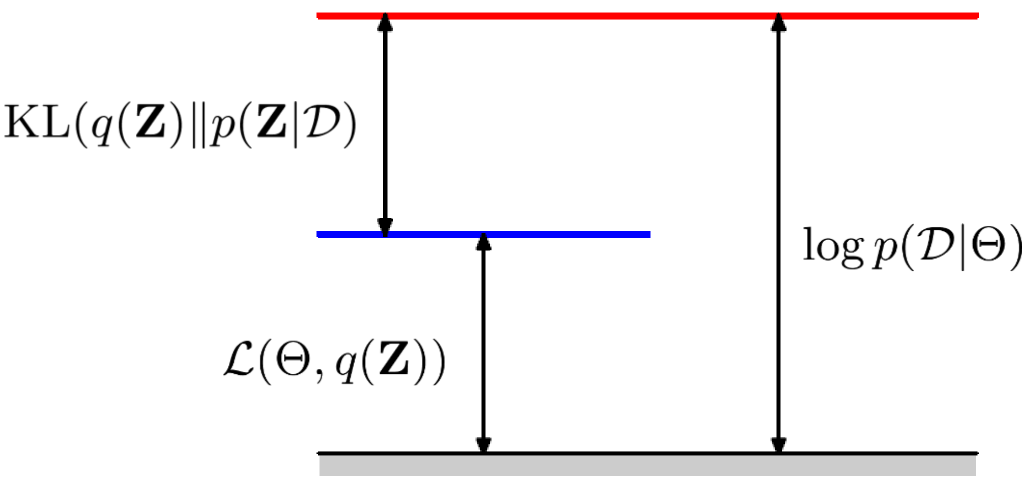
# Theory

- What we have is a lower bound

$$\log p(\mathcal{D}|\Theta) \geq \sum_{n=1}^{N} \sum_{\mathbf{z}_n} q(\mathbf{z}_n) \log \left( \frac{p(\mathbf{x}_n, \mathbf{z}_n)}{q(\mathbf{z}_n)} \right) \triangleq \mathcal{L}(\Theta, q(\mathbf{Z}))$$

- What's the GAP?

$$\log p(\mathcal{D}|\Theta) - \mathcal{L}(\Theta, q(\mathbf{Z})) = \mathrm{KL}(q(\mathbf{Z}) \| p(\mathbf{Z}|\mathcal{D})$$

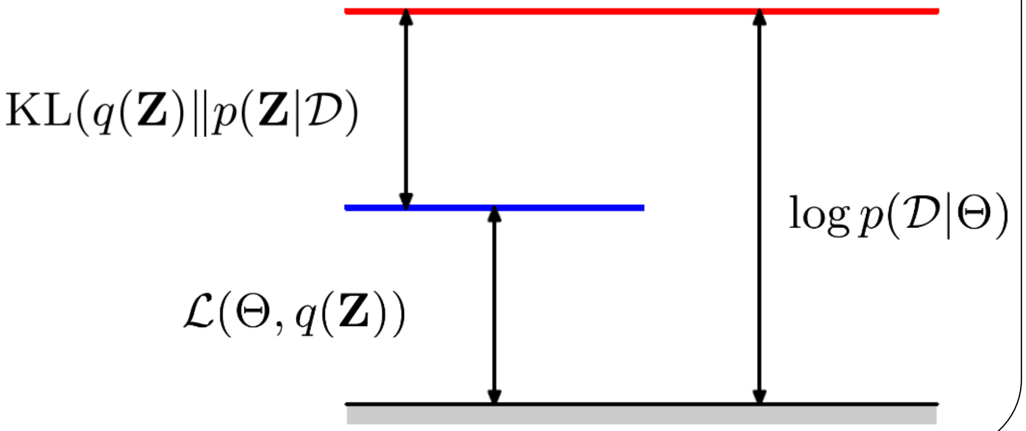


# EM-algorithm

- Maximize the lower bound or minimize the gap:

$$\log p(\mathcal{D}|\Theta) \geq \sum_{n=1}^{N} \sum_{\mathbf{z}_n} q(\mathbf{z}_n) \log \left( \frac{p(\mathbf{x}_n, \mathbf{z}_n)}{q(\mathbf{z}_n)} \right) \triangleq \mathcal{L}(\Theta, q(\mathbf{Z}))$$

  - Maximize over $q(\mathrm{Z})$ => E-step
  - Maximize over $\Theta$ => M-step



# Convergence of EM

- Local optimum is guaranteed under mild conditions (Depster et al., 1977)
  - alternating minimization for a bi-convex problem

$$\mathcal{L}(\Theta_{t+1}) \geq \mathcal{L}(\Theta_t)$$

- Some special cases with global optimum (Wu, 1983)

- First-order gradient descent for log-likelihood
  - for comparison with other gradient ascent methods, see (Xu & Jordan, 1995)

# Relation between GMM and K-Means

- Small variance asymptotics:
  - The EM algorithm for GMM reduces to K-Means under certain conditions:

E-step:

$$\gamma(z_{nk}) = \frac{\pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \Sigma_k)}{\sum_j \pi_j \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_j, \Sigma_j)}$$

M-step:

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

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

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

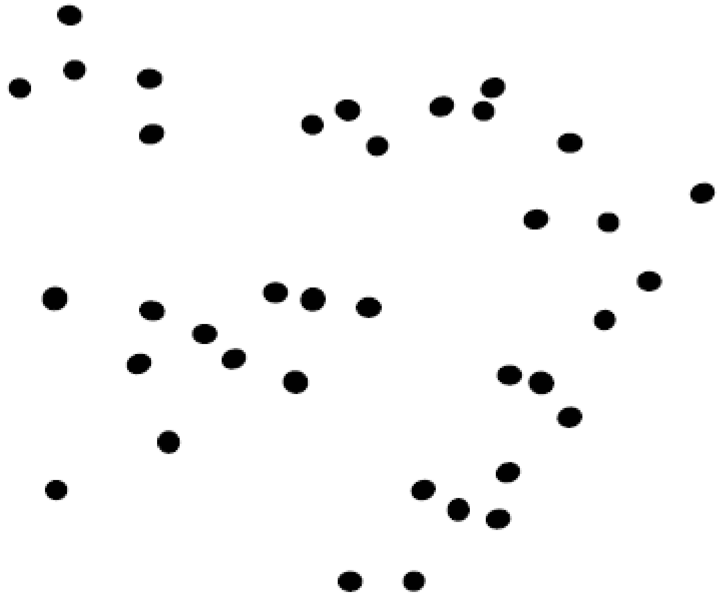
$$\text{let } \Sigma_k = \sigma I \text{ and } \sigma \to 0$$

$$\gamma(z_{nk}) = \frac{\pi_k \exp(-\frac{1}{2\sigma} \|\mathbf{x}_n - \boldsymbol{\mu}_k\|_2^2)}{\sum_j \pi_j \exp(-\frac{1}{2\sigma} \|\mathbf{x}_n - \boldsymbol{\mu}_j\|_2^2)}$$

$$= k^*, \text{ where } k^* = \mathrm{argmin}_k \|\mathbf{x}_n - \boldsymbol{\mu}_k\|_2^2$$

# Single Linkage Hierarchical Clustering

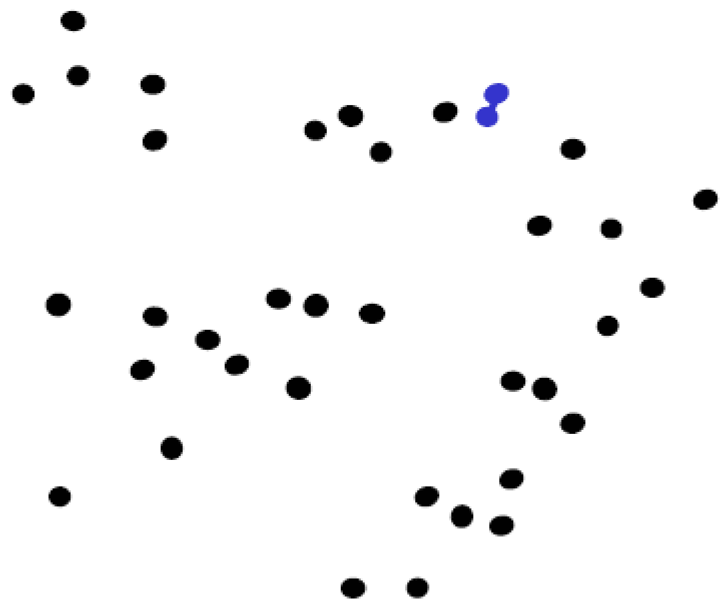
- Start with "every point is its own cluster"



[Slide courtesy: Andrew Moore]

# Single Linkage Hierarchical Clustering

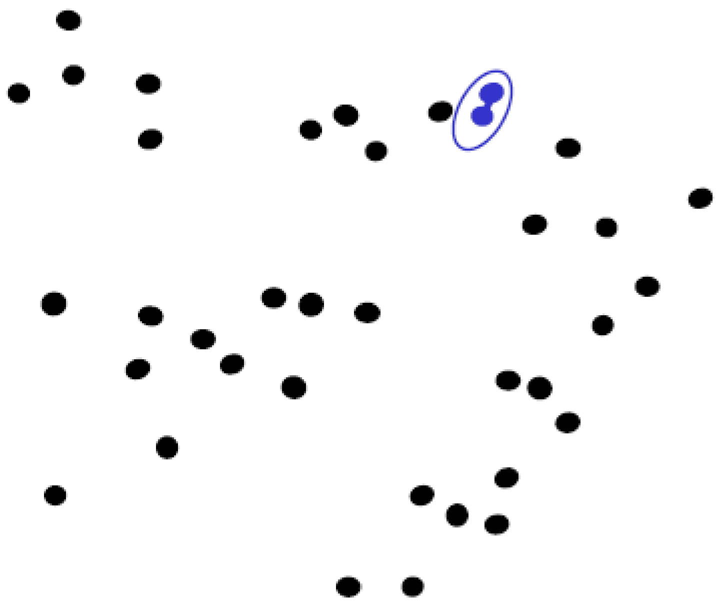
- Start with "every point is its own cluster"
- Find "most similar" pairs of clusters



[Slide courtesy: Andrew Moore]

# Single Linkage Hierarchical Clustering

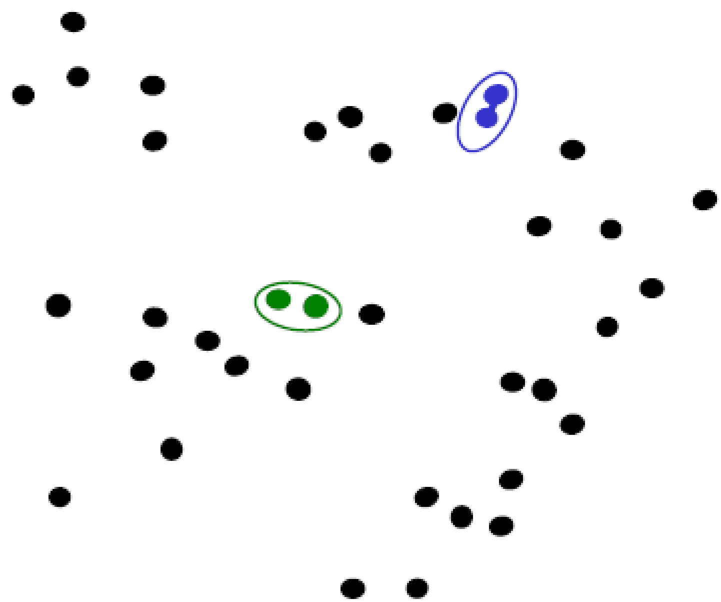
- Start with "every point is its own cluster"
- Find "most similar" pairs of clusters
- Merge it into a parent cluster



[Slide courtesy: Andrew Moore]

# Single Linkage Hierarchical Clustering

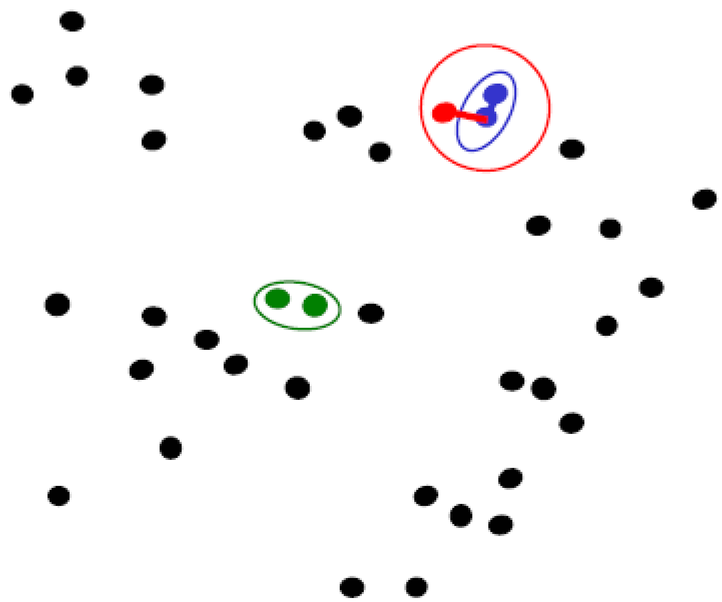
- Start with "every point is its own cluster"
- Find "most similar" pairs of clusters
- Merge it into a parent cluster
- Repeat

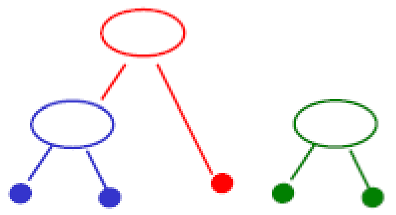


[Slide courtesy: Andrew Moore]

# Single Linkage Hierarchical Clustering

- Start with "every point is its own cluster"
- Find "most similar" pairs of clusters
- Merge it into a parent cluster
- Repeat

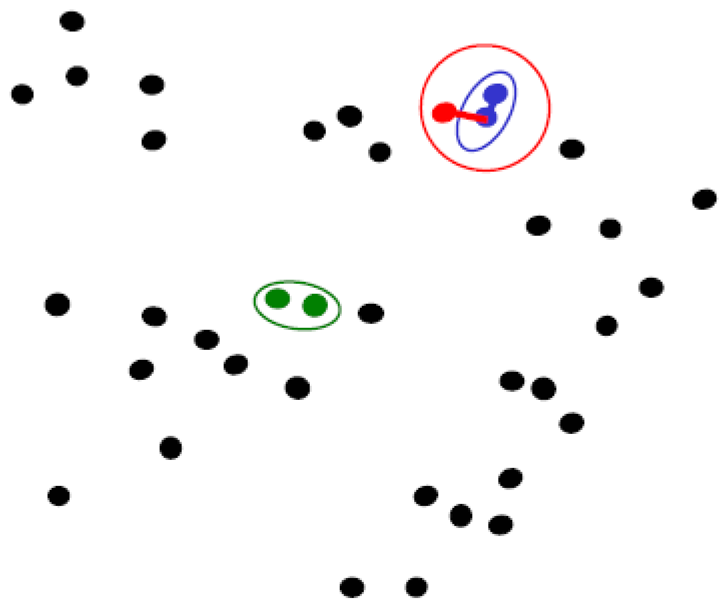




[Slide courtesy: Andrew Moore]

# Single Linkage Hierarchical Clustering

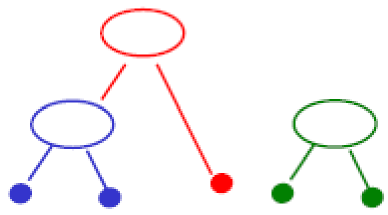
- Start with "every point is its own cluster"
- Find "most similar" pairs of clusters
- Merge it into a parent cluster
- Repeat



**Key Question:**

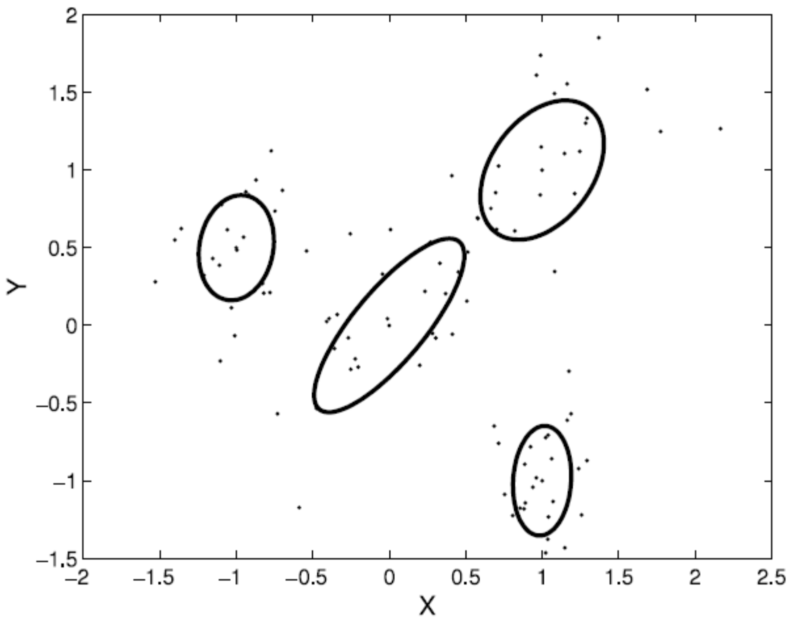
**How do we define similarity between clusters?**

**=> minimum, maximum, or average distance between points in clusters**



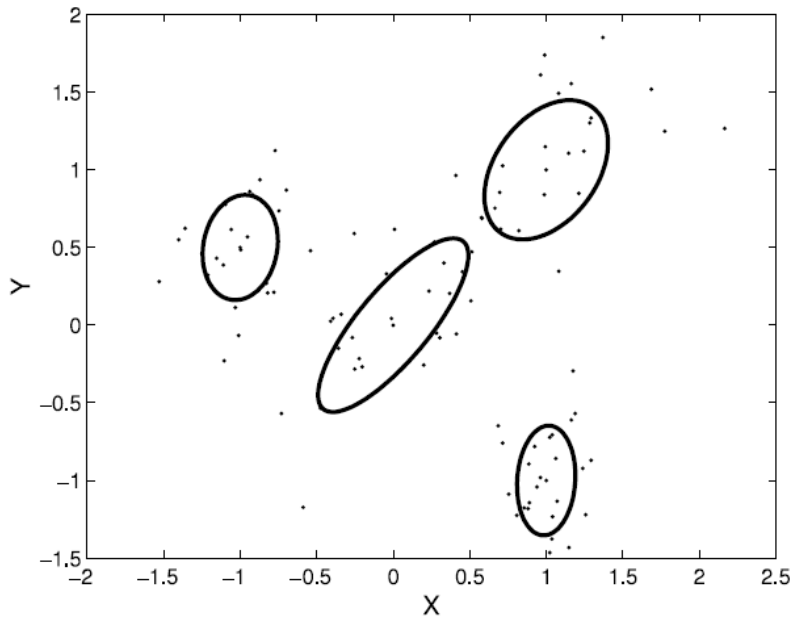
[Slide courtesy: Andrew Moore]

# How many components are good?




- Can we let the data speak for themselves?
  - let data determine model complexity (e.g., the number of components in mixture models)
  - allow model complexity to grow as more data observed

# How many components are good?




- Can we let the data speak for themselves?
  - Dirichlet Process (DP) Mixtures
  - and nonparametric Bayesian models

# Summary

- Gaussian Mixtures and K-means are effective tools to discover clustering structures
- EM algorithms can be applied to do MLE for GMMs
- Relationships between GMMs and K-means are discussed

- Unresolved issues
  - How to determine the number of components for mixture models?
  - How to determine the number of components for K-means?

# Materials to Read

- Chap. 9 of Bishop's PRML book
- Bottou, L. & Bengio, Y. Convergence Properties of the K-means Algorithms, NIPS 1995.