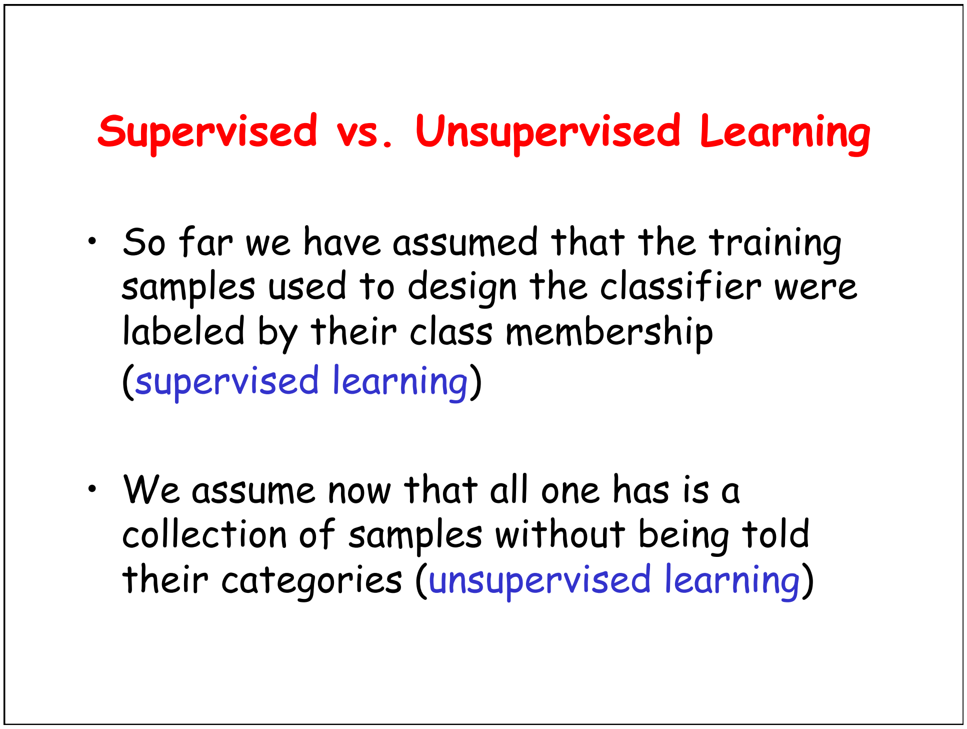
Clustering



Supervised vs. Unsupervised Learning

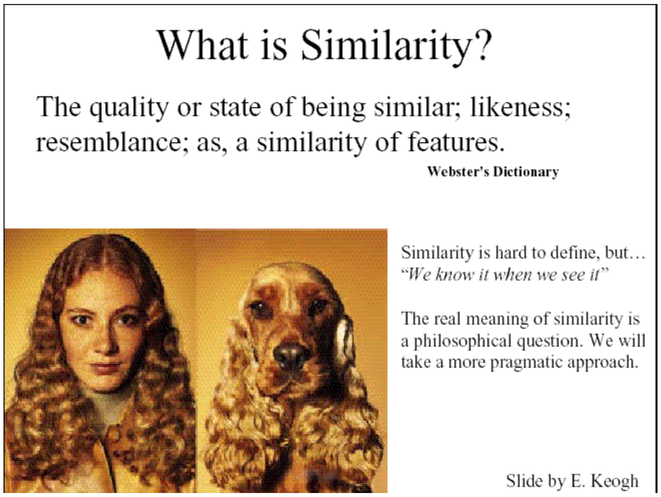
- So far we have assumed that the training samples used to design the classifier were labeled by their class membership (supervised learning)

- We assume now that all one has is a collection of samples without being told their categories (unsupervised learning)

Clustering

- **Goal**: Grouping a collection of objects (data points) into subsets or "clusters", such that those within each cluster are more closely related to one other than objects assigned to different clusters.

- Fundamental to all clustering techniques is the choice of ***distance or dissimilarity measure*** between two objects.



What is Similarity?

The quality or state of being similar; likeness; resemblance; as, a similarity of features.

Webster's 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.

Slide by E. Keogh

Dissimilarities based on Features

$$\boldsymbol{x}_i = \left(x_{i1}, x_{i2}, \cdots, x_{iq}\right)^T \in \Re^q, \;\; i = 1, \cdots, N$$

$$D\left(\boldsymbol{x}_i, \boldsymbol{x}_j\right) = \sum_{k=1}^{q} d_k\left(x_{ik}, x_{jk}\right)$$

$$d_k\left(x_{ik}, x_{jk}\right) = \left(x_{ik} - x_{jk}\right)^2$$

$$\Rightarrow D\left(\boldsymbol{x}_i, \boldsymbol{x}_j\right) = \sum_{k=1}^{q} \left(x_{ik} - x_{jk}\right)^2$$

Squared Euclidean distance

$$D_w\left(\boldsymbol{x}_i, \boldsymbol{x}_j\right) = \sum_{k=1}^{q} w_k\left(x_{ik} - x_{jk}\right)^2$$

Weighted squared Euclidean distance

Categorical Features

- E.g.: color, shape, etc.
- No natural ordering between variables exist;
- The degree-of-difference between pairs of values must be defined;
- If a variable assumes M distinct values:

$M \times M$ symmetric matrix with elements:

$m_{rs} = m_{sr}, \; m_{rr} = 0, \; m_{rs} \geq 0$

Common choice: $m_{rs} = 1 \;\; \forall r \neq s$

Clustering

- Discovering patterns (e.g., groups) in data without any guidance (labels) sounds like an "unpromising" problem.

- The question of whether or not it is possible in principle to learn anything from unlabeled data depends upon the assumptions one is willing to accept.

Clustering

- **Mixture modeling**: makes the assumption that data are samples of a population described by a probability density function

- **Combinatorial algorithms**: work directly on the observed data with no reference to an underlying probability model.

Clustering Algorithms: Mixture Modeling

- Data is a sample from a population described by a probability density function;
- The density function is modeled as a mixture of component density functions (e.g., mixture of Gaussians). Each component density describes one of the clusters;
- The parameters of the model (e.g., means and covariance matrices for mixture of Gaussians) are estimated as to best fit the data (*maximum likelihood estimation*).

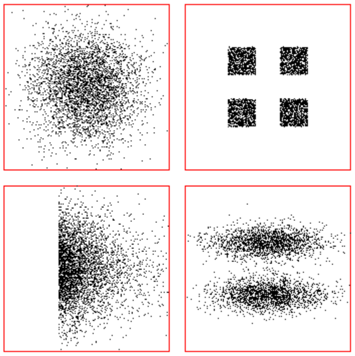
Clustering Algorithms: Mixture Modeling

- Suppose that we knew, somehow, that the given sample data come from a single normal distribution
- Then: the most we could learn from the data would be contained in the *sample mean* and in the *sample covariance matrix*
- These statistics constitute a compact description of the data.

Clustering Algorithms: Mixture Modeling

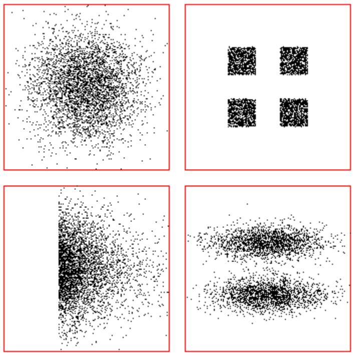
What happens if our knowledge is inaccurate, and the data are not actually normally distributed?

Example



All four data sets have identical mean and covariance matrix

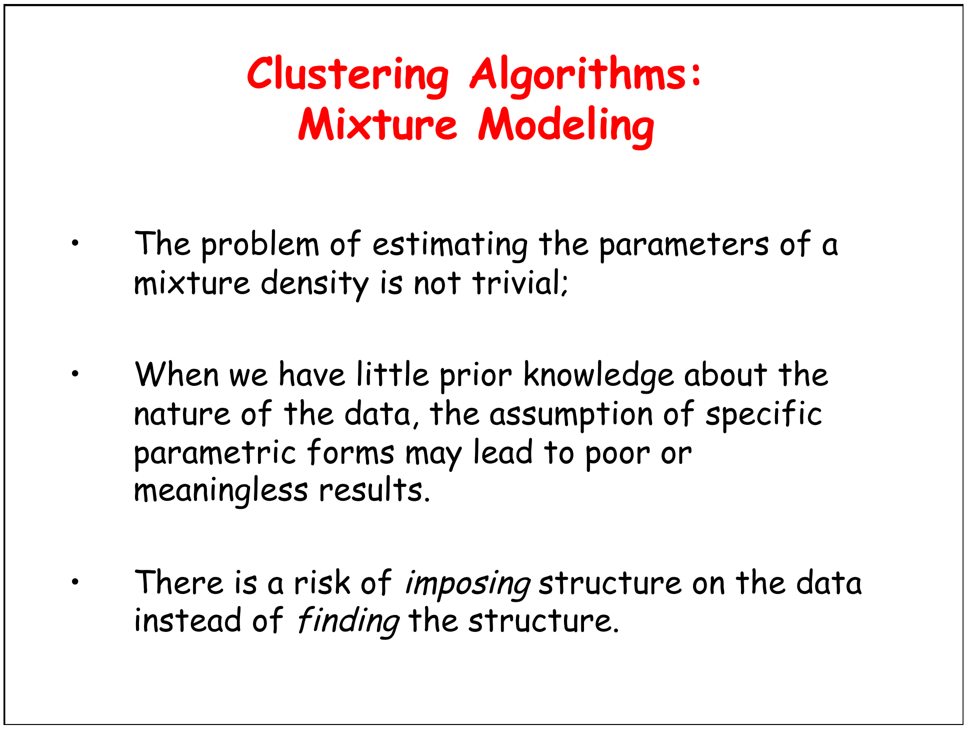
Example



Clearly second-order statistics are not capable to revealing all of the structure in an arbitrary set of data

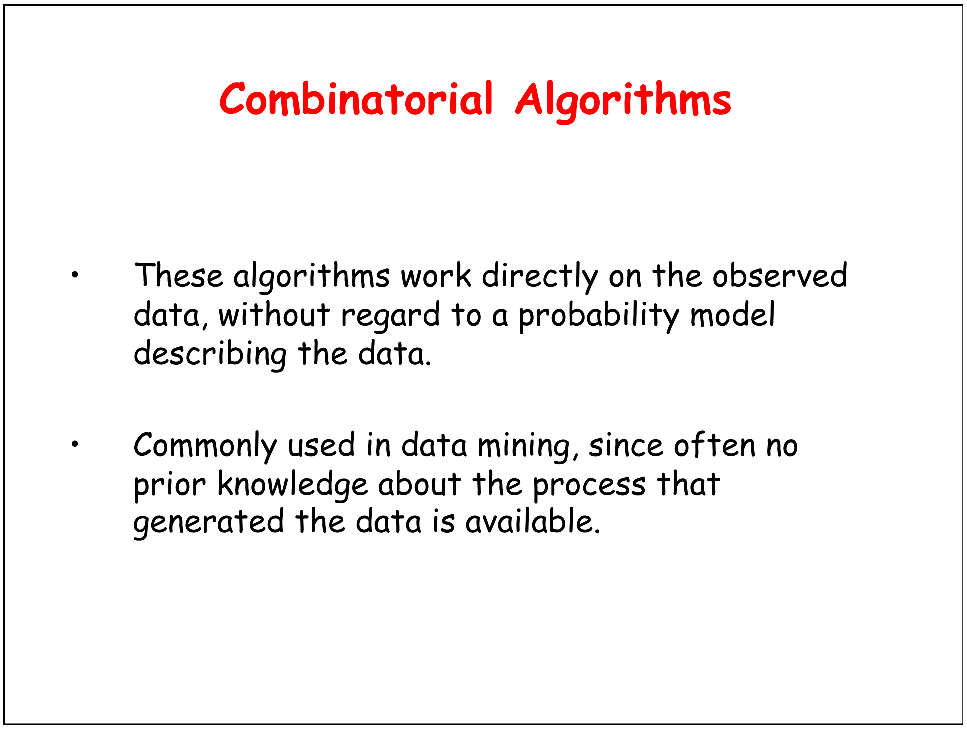
Clustering Algorithms: Mixture Modeling

- If we assume that the samples come from a mixture of c normal distributions, we can approximate a greater variety of situations;
- If the number of component densities is sufficiently high, we can approximate virtually any density function as a mixture model.
- However…



Clustering Algorithms: Mixture Modeling

- The problem of estimating the parameters of a mixture density is not trivial;
- When we have little prior knowledge about the nature of the data, the assumption of specific parametric forms may lead to poor or meaningless results.
- There is a risk of *imposing* structure on the data instead of *finding* the structure.



Combinatorial 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.

Combinatorial Algorithms

$\boldsymbol{x}_i \in \Re^q, \ i = 1, \cdots, N$

Prespecified number of clusters $K, \ k \in \{1, \cdots, K\}$

Each data point $\boldsymbol{x}_i$ 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 $D(\boldsymbol{x}_i, \boldsymbol{x}_k)$

Usually, the assignment of data to clusters is done so as to **minimize** a "loss" function that measures the degree to which the clustering goal is **not** met

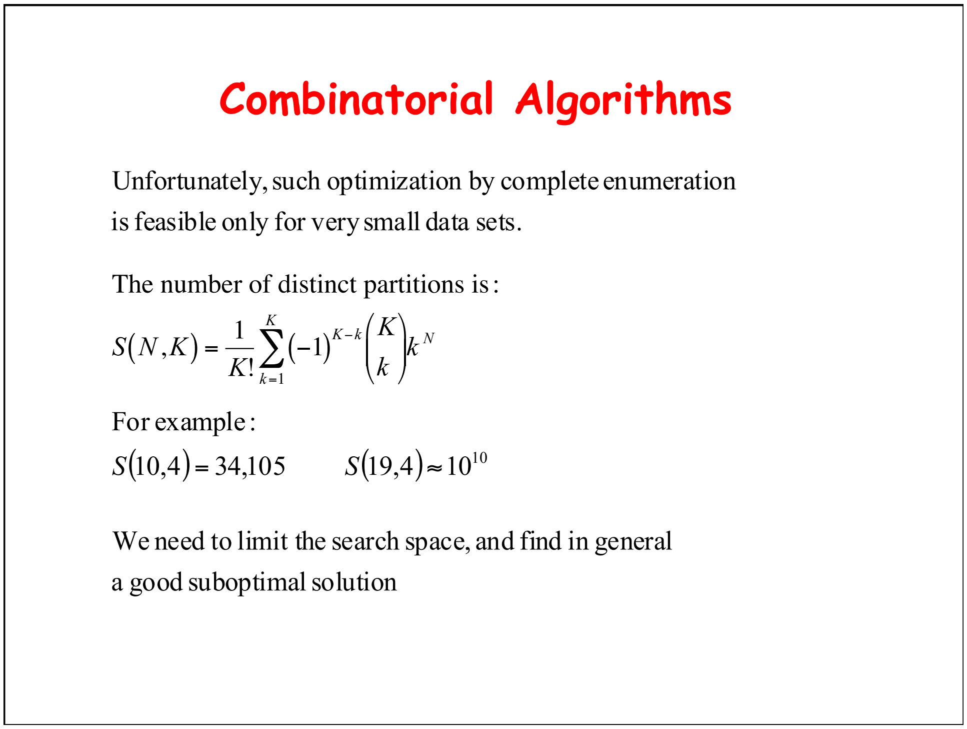
Combinatorial Algorithms

Since the goal is to assign close points to the same cluster, a natural loss function would be :

$$W(C) = \frac{1}{2}\sum_{k=1}^{K}\sum_{i \in C_k}\sum_{j \in C_k} D(\boldsymbol{x}_i, \boldsymbol{x}_j)$$

Within cluster scatter

Then, clustering becomes straightforward *in principle* : Minimize W over all possible assignments of the N data points to K clusters



Combinatorial Algorithms

Unfortunately, such optimization by complete enumeration is feasible only for very small data sets.

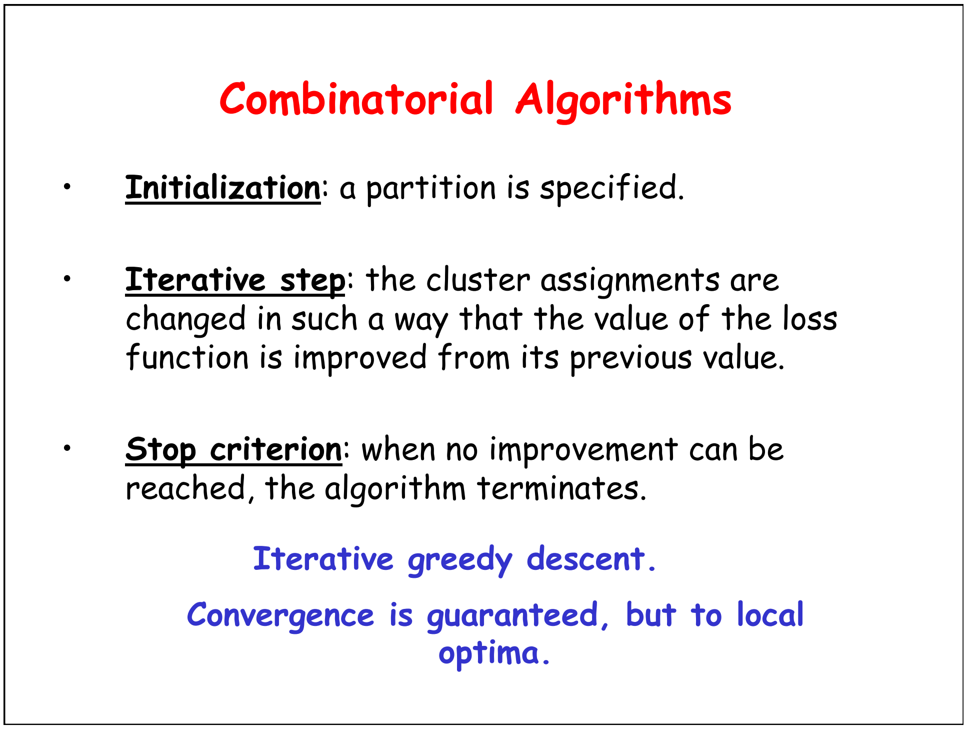
The number of distinct partitions is:

$$S(N,K) = \frac{1}{K!}\sum_{k=1}^{K}(-1)^{K-k}\binom{K}{k}k^{N}$$

For example:

$S(10,4) = 34{,}105 \qquad S(19,4) \approx 10^{10}$

We need to limit the search space, and find in general a good suboptimal solution

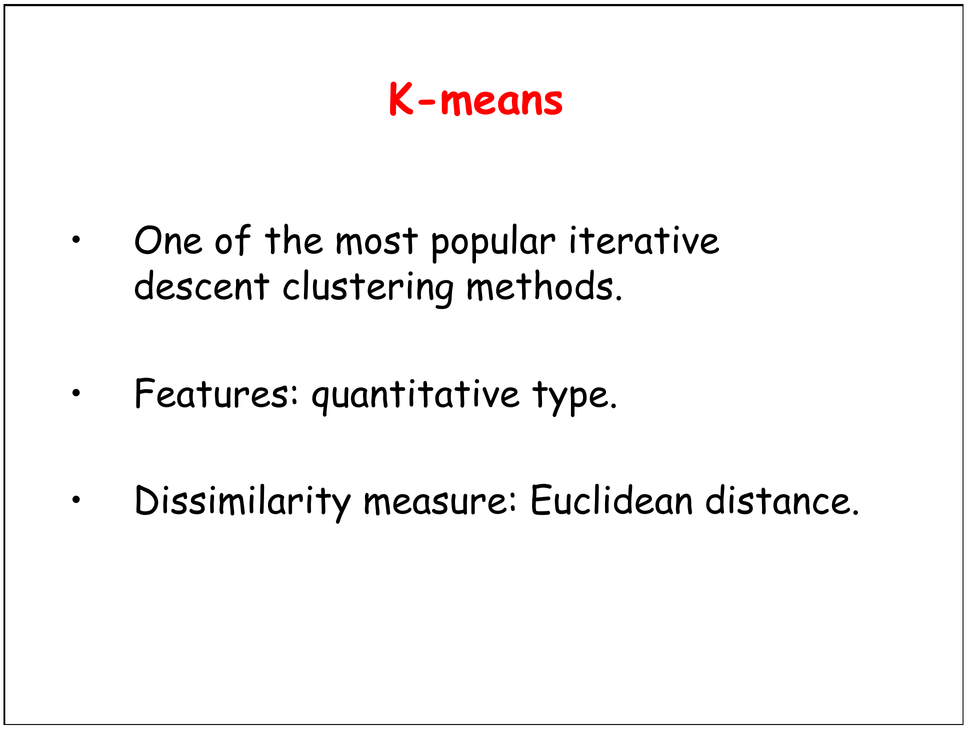


Combinatorial Algorithms

- **Initialization**: a partition is specified.
- **Iterative step**: the cluster assignments are changed in such a way that the value of the loss function is improved from its previous value.
- **Stop criterion**: when no improvement can be reached, the algorithm terminates.

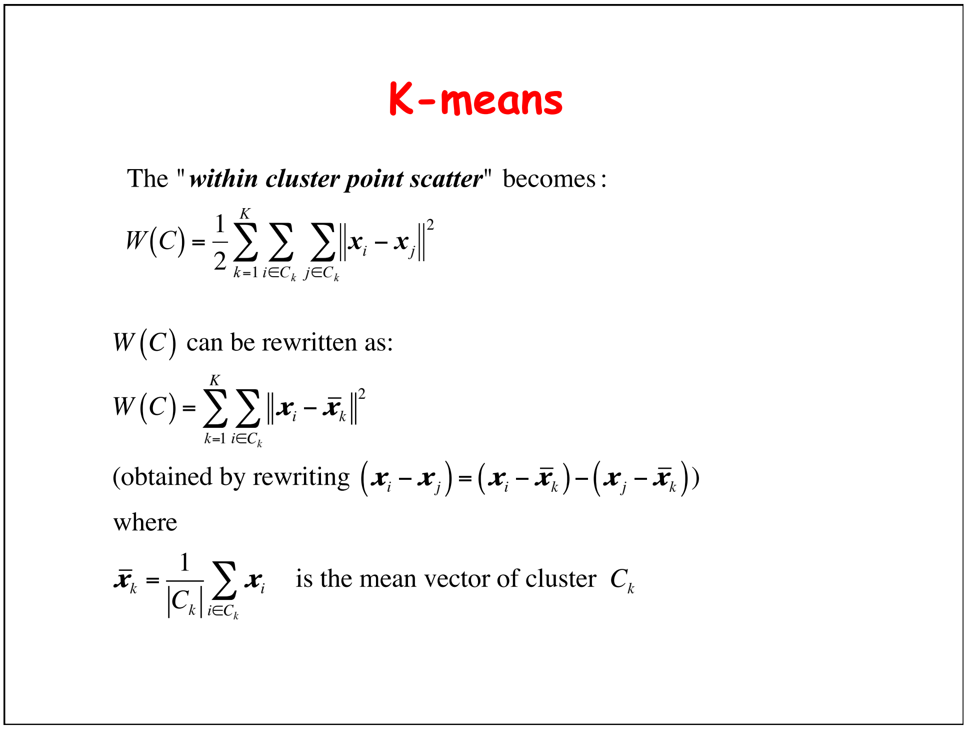
Iterative greedy descent.

Convergence is guaranteed, but to local optima.



K-means

- One of the most popular iterative descent clustering methods.
- Features: quantitative type.
- Dissimilarity measure: Euclidean distance.



K-means

The "***within cluster point scatter***" becomes:

$$W(C) = \frac{1}{2}\sum_{k=1}^{K}\sum_{i \in C_k}\sum_{j \in C_k}\left\|x_i - x_j\right\|^2$$

$W(C)$ can be rewritten as:

$$W(C) = \sum_{k=1}^{K}\sum_{i \in C_k}\left\|\boldsymbol{x}_i - \bar{\boldsymbol{x}}_k\right\|^2$$

(obtained by rewriting $(\boldsymbol{x}_i - \boldsymbol{x}_j) = (\boldsymbol{x}_i - \bar{\boldsymbol{x}}_k) - (\boldsymbol{x}_j - \bar{\boldsymbol{x}}_k)$)

where

$\bar{\boldsymbol{x}}_k = \frac{1}{|C_k|}\sum_{i \in C_k}\boldsymbol{x}_i$ is the mean vector of cluster C_k

K-means

The objective is:

$$\min_{C} \sum_{k=1}^{K} \sum_{i \in C_k} \|\boldsymbol{x}_i - \bar{\boldsymbol{x}}_k\|^2$$

We can solve this problem by noticing:

for any set of data S

$$\bar{\boldsymbol{x}}_S = \arg\min_{\boldsymbol{m}} \sum_{i \in S} \|\boldsymbol{x}_i - \boldsymbol{m}\|^2$$

(this is obtained by setting $\dfrac{\partial \sum_{i \in S} \|\boldsymbol{x}_i - \boldsymbol{m}\|^2}{\partial \boldsymbol{m}} = 0$)

So we can solve the enlarged optimization problem:

$$\min_{C, \boldsymbol{m}_k} \sum_{k=1}^{K} \sum_{i \in C_k} \|\boldsymbol{x}_i - \boldsymbol{m}_k\|^2$$

K-means: The Algorithm

1. Given a cluster assignment C, the total within cluster scatter

$\sum_{k=1}^{K} \sum_{i \in C_k} \|\boldsymbol{x}_i - \boldsymbol{m}_k\|^2$ is minimized with respect to the $\{\boldsymbol{m}_1, \cdots, \boldsymbol{m}_K\}$

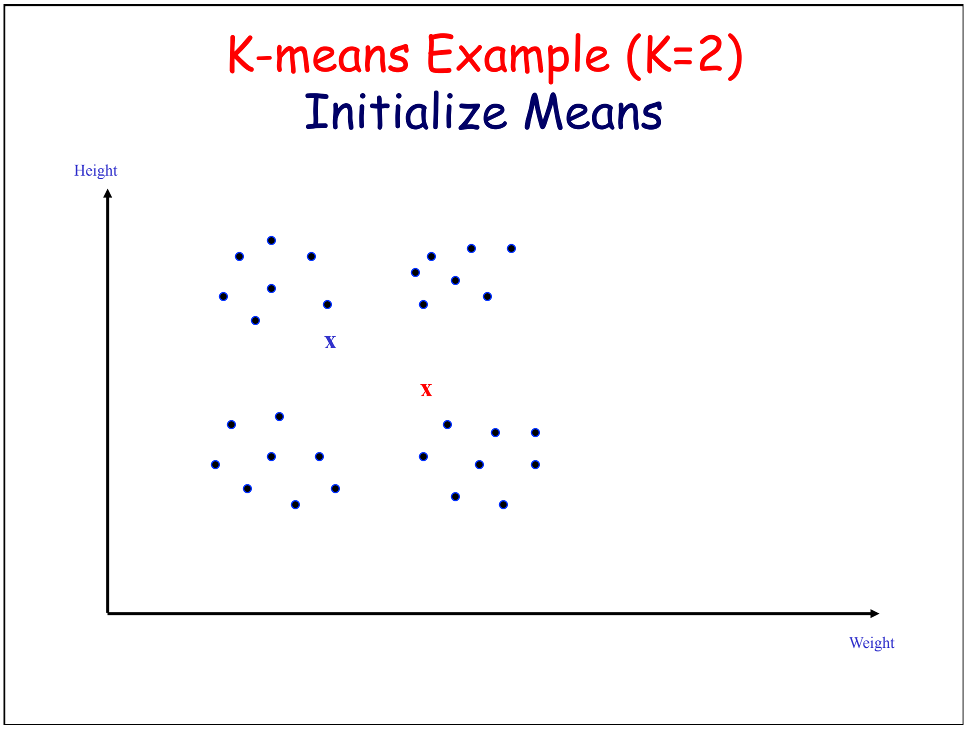
giving the means of the currently assigned clusters;

2. Given a current set of means $\{\boldsymbol{m}_1, \cdots, \boldsymbol{m}_K\}$,

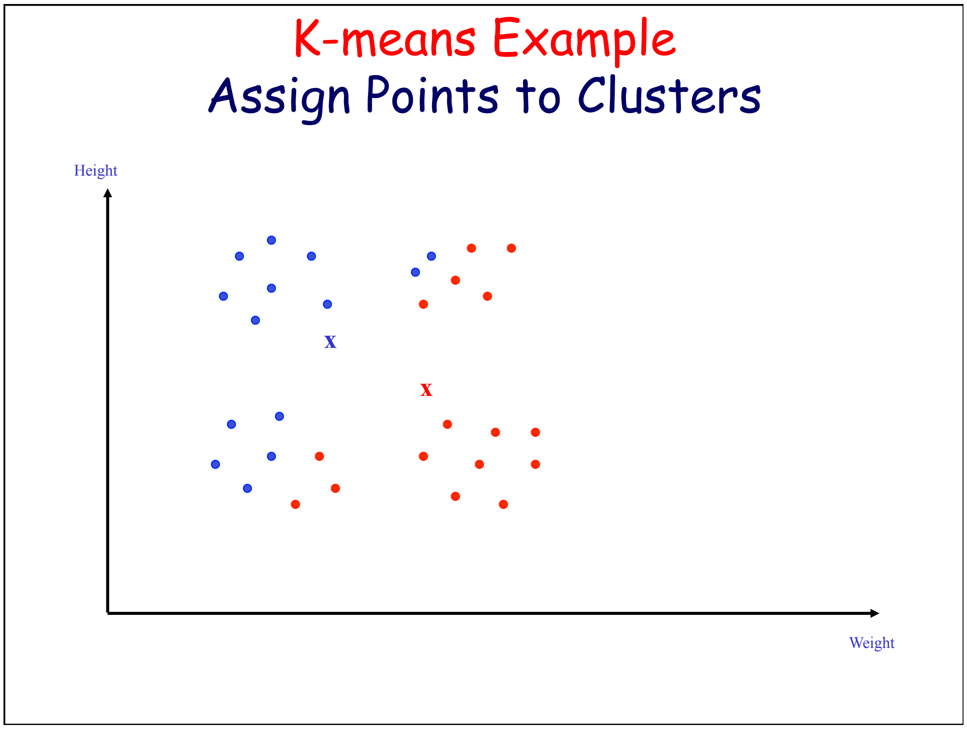
$\sum_{k=1}^{K} \sum_{i \in C_k} \|\boldsymbol{x}_i - \boldsymbol{m}_k\|^2$ is minimized with respect to C

by assigning each point to the closest current cluster mean;

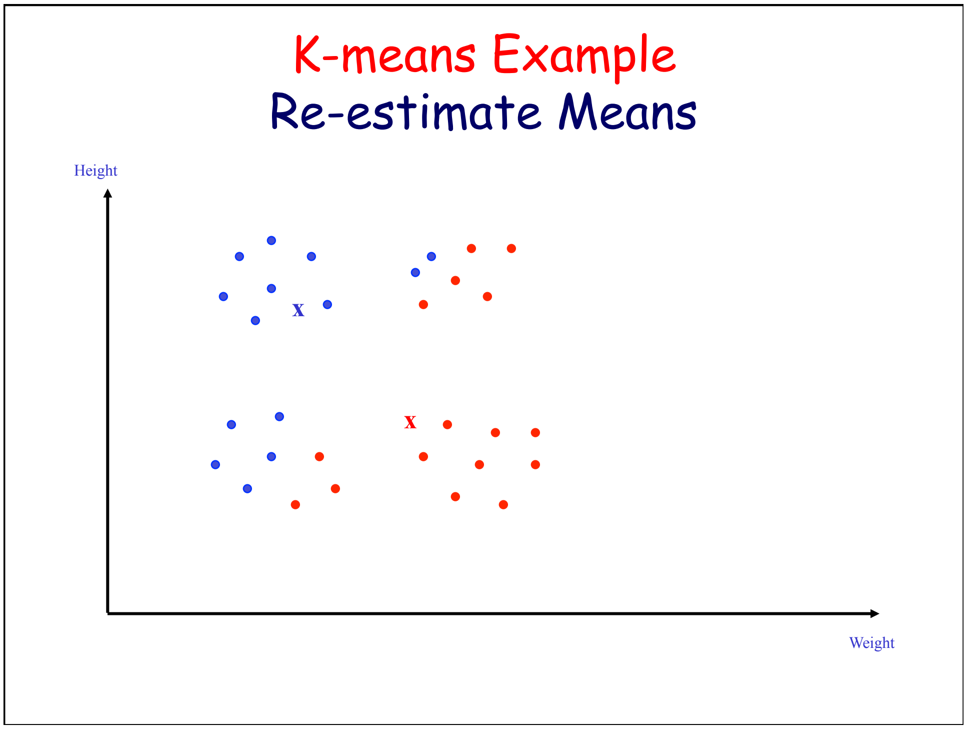
3. Steps 1 and 2 are iterated until the assignments do not change.



K-means Example (K=2)
Initialize Means

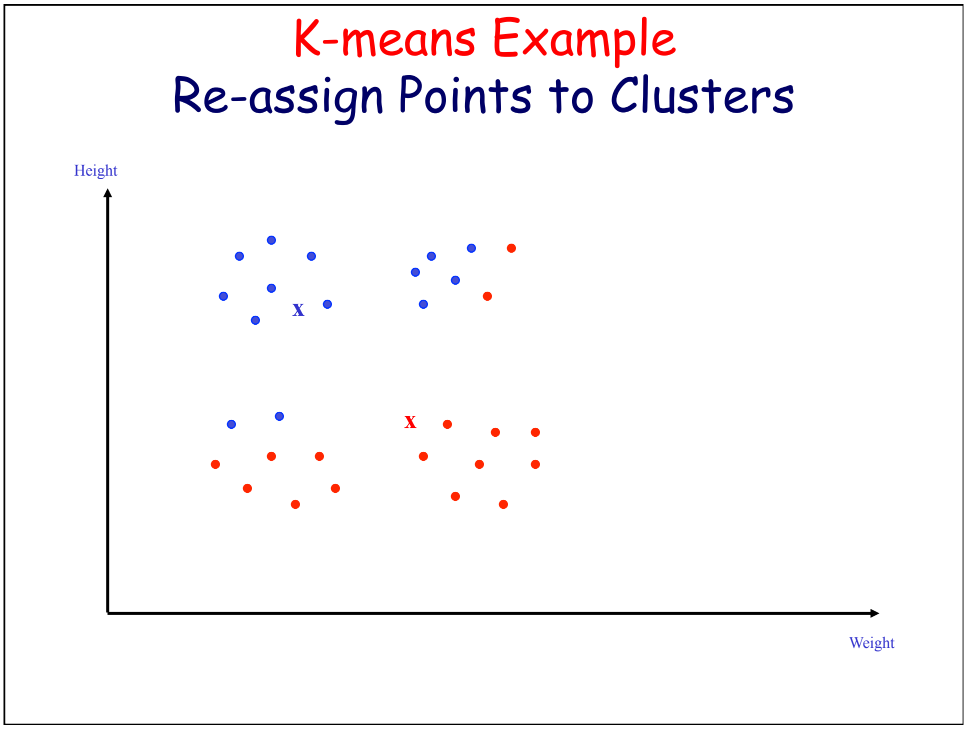


K-means Example
Assign Points to Clusters



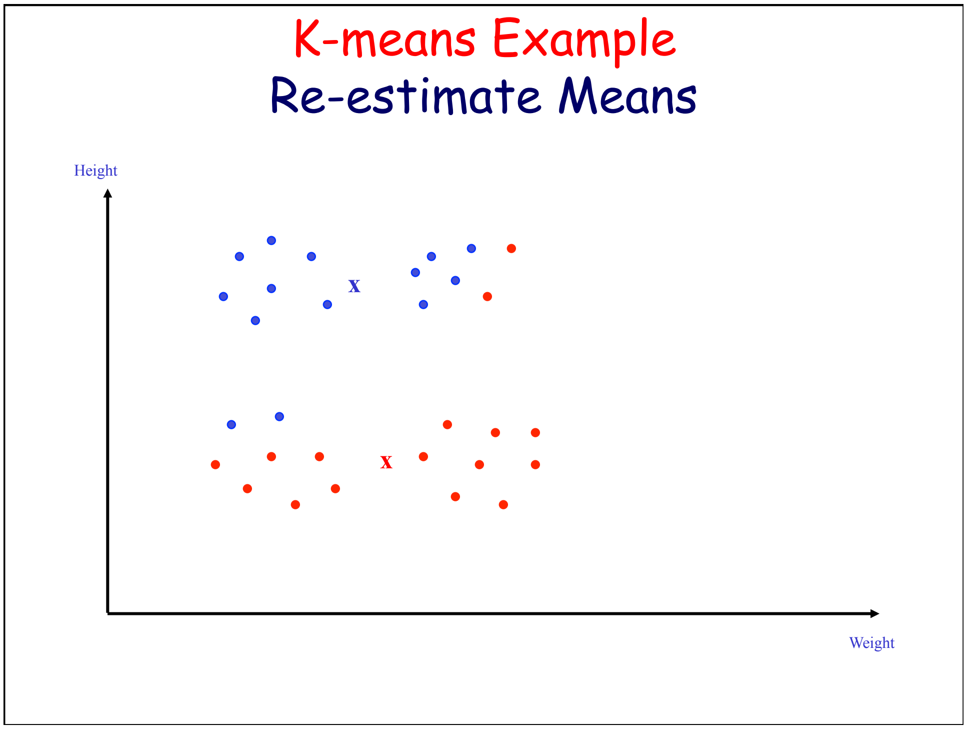
K-means Example

Re-estimate Means

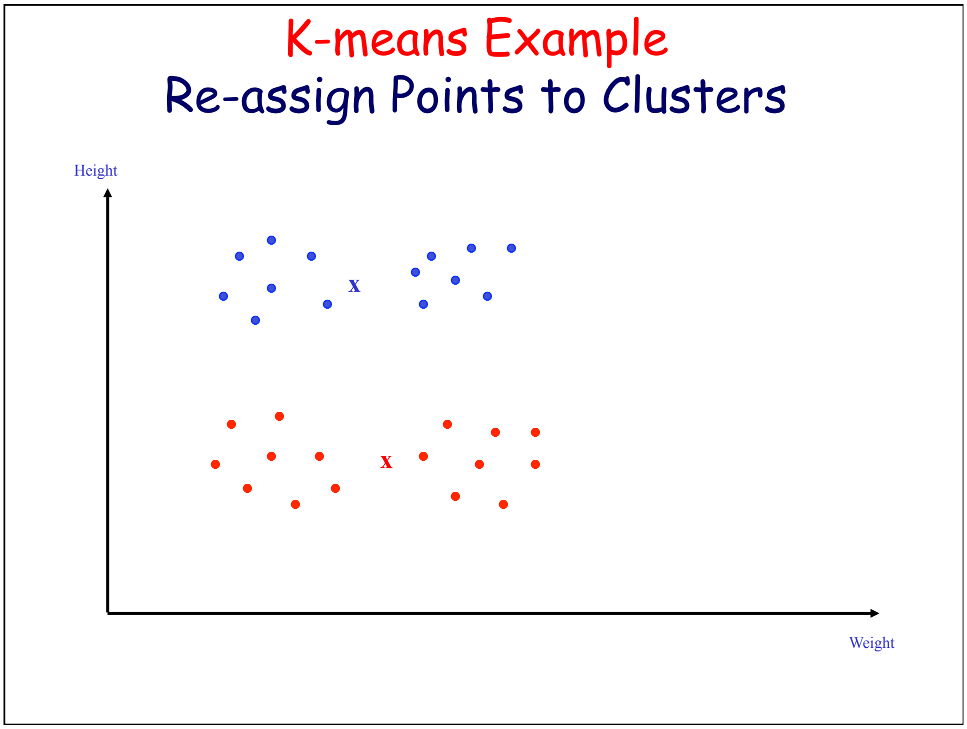


K-means Example

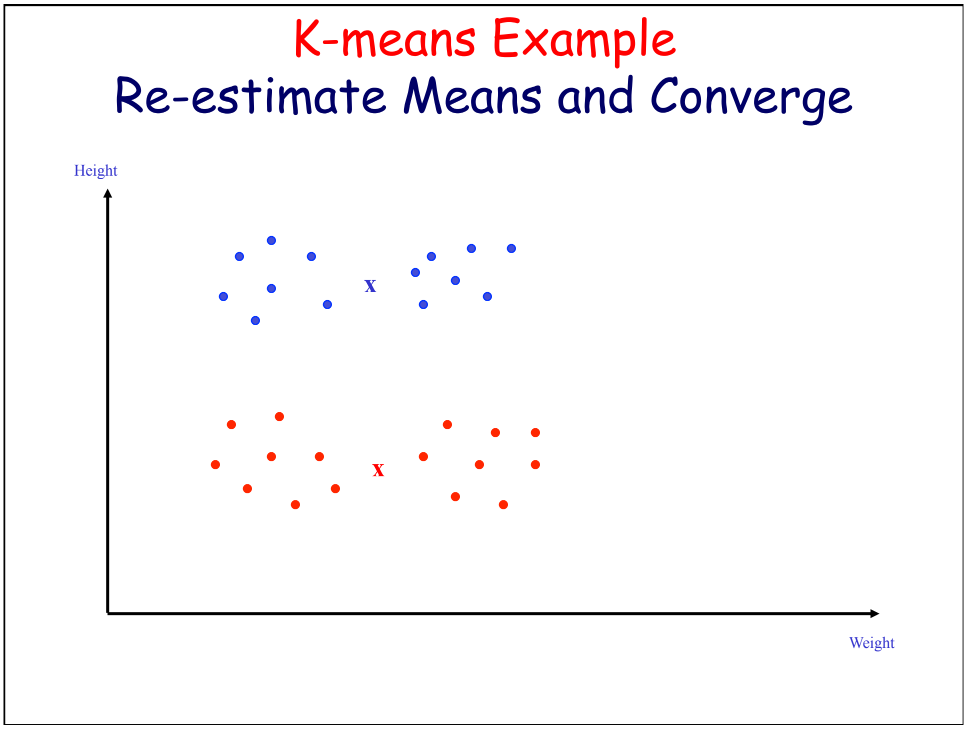
Re-assign Points to Clusters



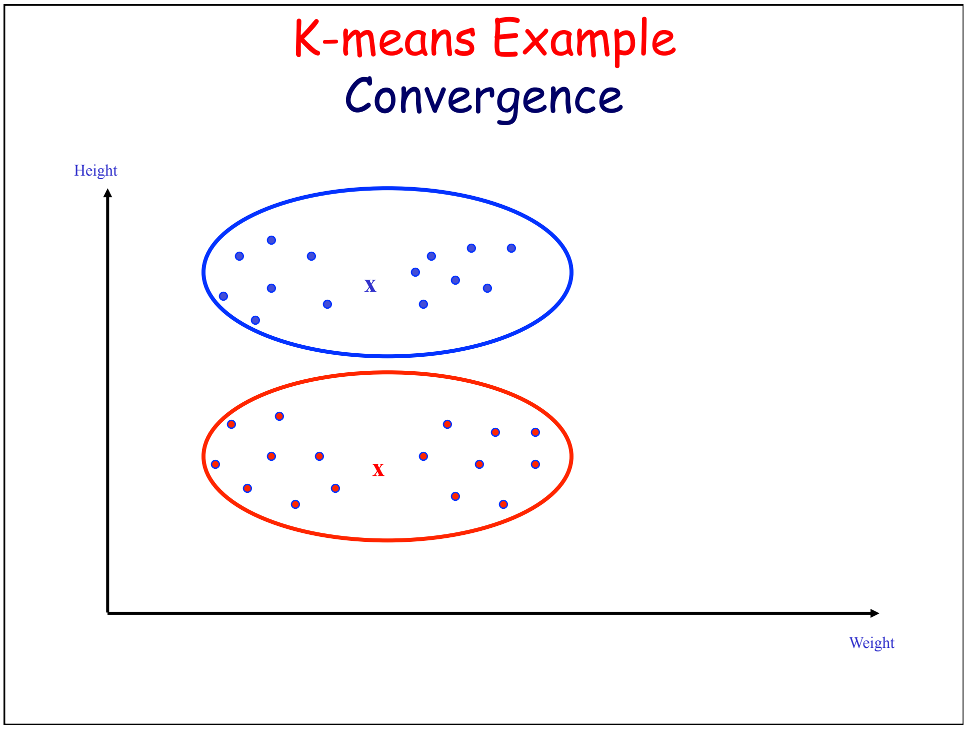
K-means Example
Re-estimate Means



K-means Example
Re-assign Points to Clusters



K-means Example
Re-estimate Means and Converge



K-means Example
Convergence

K-means: Properties and Limitations

- The algorithm converges to a local minimum

- The solution depends on the initial partition

- 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 is sensitive to outliers
- A variation of K-means improves upon robustness (K-medoids):
  - Centers for each cluster are restricted to be one of the points assigned to the cluster;
  - The center (*medoid*) is set to be the point that minimizes the total distance to other points in the cluster;
  - K-medoids is more computationally intensive than K-means.

K-means: Properties and Limitations

- The algorithm requires the number of clusters K;

- Often K is unknown, and must be estimated from the data:

We can test $K \in \{1,2,\cdots,K_{\max}\}$

Compute $\{W_1,W_2,\cdots,W_{\max}\}$

In general: $W_1 > W_2 > \cdots > W_{\max}$

K^* = actual number of clusters in the data,

when $K < K^*$, we can expect $W_K >> W_{K+1}$

when $K > K^*$, further splits provide smaller decrease of W

Set $\hat{K}^*$ by identifying an "elbow shape" in the plot of W_k

Gap Statistics:

Estimating the number of clusters in a data set via the gap statistic

Tibshirani, Walther, & Hastie, 2001

Plot $\log W_K$

Plot the curve $\log W_K$ obtained from data uniformely distributed

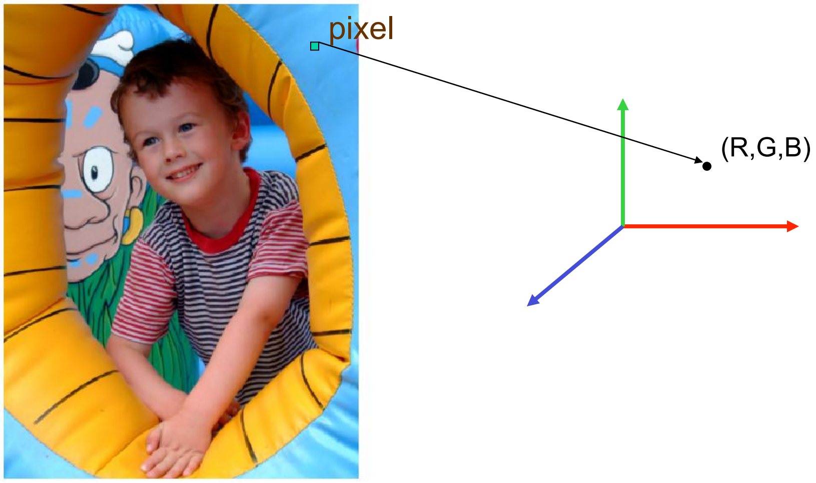
Estimate $\hat{K}^*$ to be the point where the gap between the two curves is largest

An Application of K-means: Image segmentation

- **Goal of segmentation**: partition an image into regions with homogeneous visual appearance (which could correspond to objects or parts of objects)
- **Image representation**: each pixel is represented as a three dimensional point in RGB space, where
  - R = intensity of red
  - G = intensity of green
  - B = intensity of blue

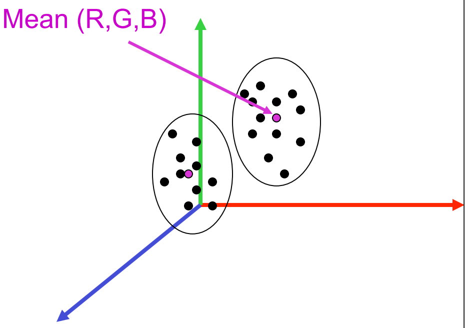
An Application of K-means: Image segmentation

Original image



An Application of K-means: Image segmentation

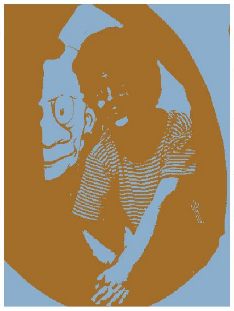
Original image



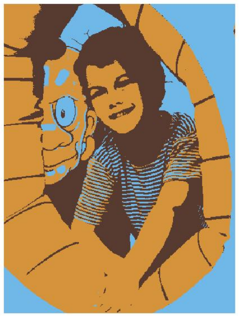
An Application of K-means: Image segmentation

Original image

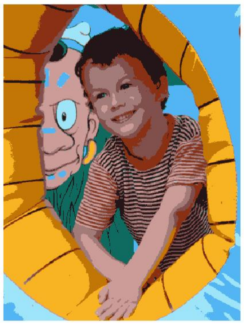
$K = 2$



$K = 3$



$K = 10$



An Application of K-means: (Lossy) Data compression

- Original image has N pixels
- Each pixel → (R,G,B) values
- Each value is stored with 8 bits of precision
- Transmitting the whole image costs 24N bits

Compression achieved by K-means:

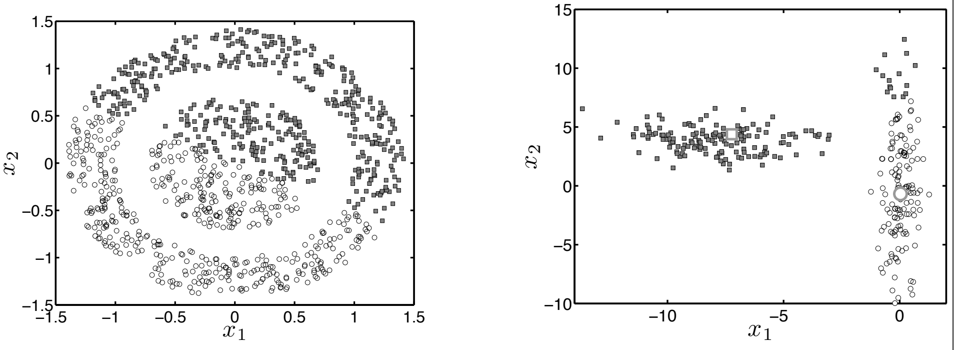
- Identify each pixel with the corresponding centroid
- We have K such centroids → we need $\log_2 K$ bits per pixel
- For each centroid we need 24 bits
- Transmitting the whole image costs $24K + N \log_2 K$ bits

Original image = 240x180=43,200 pixels → 43,200x24=1,036,800 bits

Compressed images:

K=2: 43,248 bits K=3: 86,472 K=10: 173,040 bits

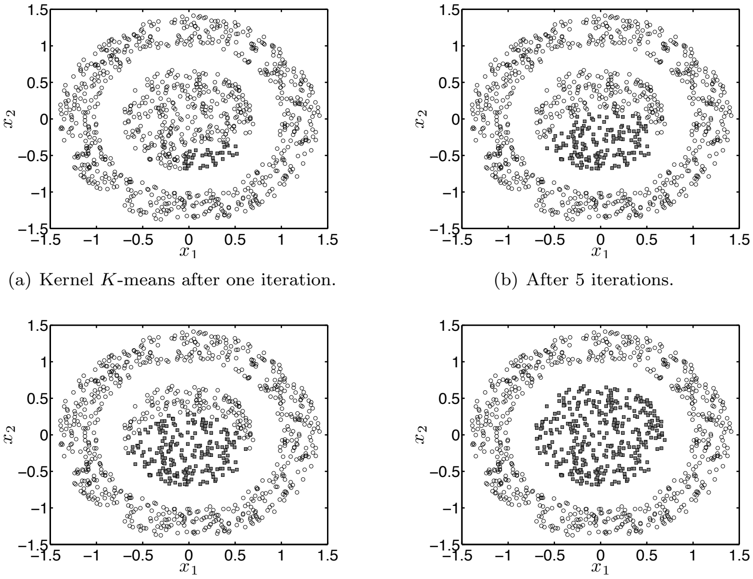
Where K-means fails



(a)

(b)

Kernel K-means



(a) Kernel K-means after one iteration.

(b) After 5 iterations.

(c) After 10 iterations.

(d) At convergence (30 iterations).