UVA CS 4774: Machine Learning

S5: Lecture 25:

Unsupervised Clustering (II): Partitional

Dr. Yanjun Qi

University of Virginia
Department of Computer Science

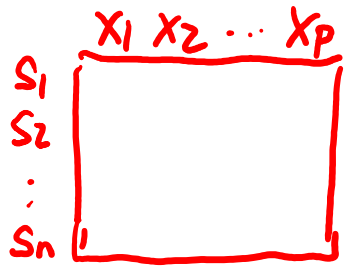
Module I

Course Content Plan → Regarding Tasks

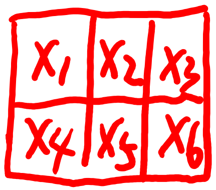
- ~~Regression (supervised)~~ — Y is a continuous
- ~~Learning theory~~ — About f()
- ~~Classification (supervised)~~ — Y is a discrete
- Unsupervised models — NO Y
- ~~Graphical models~~ — About interactions among Y,X1,. Xp
- Reinforcement Learning — Learn to Interact with environment

Course Content Plan ➔ Regarding Data

- ❑ Tabular / Matrix



- ❑ 2D Grid Structured: Imaging



- ❑ 1D Sequential Structured: Text
- ❑ Graph Structured (Relational)
- ❑ Set Structured / 3D /

	X_1	X_2	X_3
s_1			
s_2			
s_3			
s_4			
s_5			
s_6			

An unlabeled Dataset X

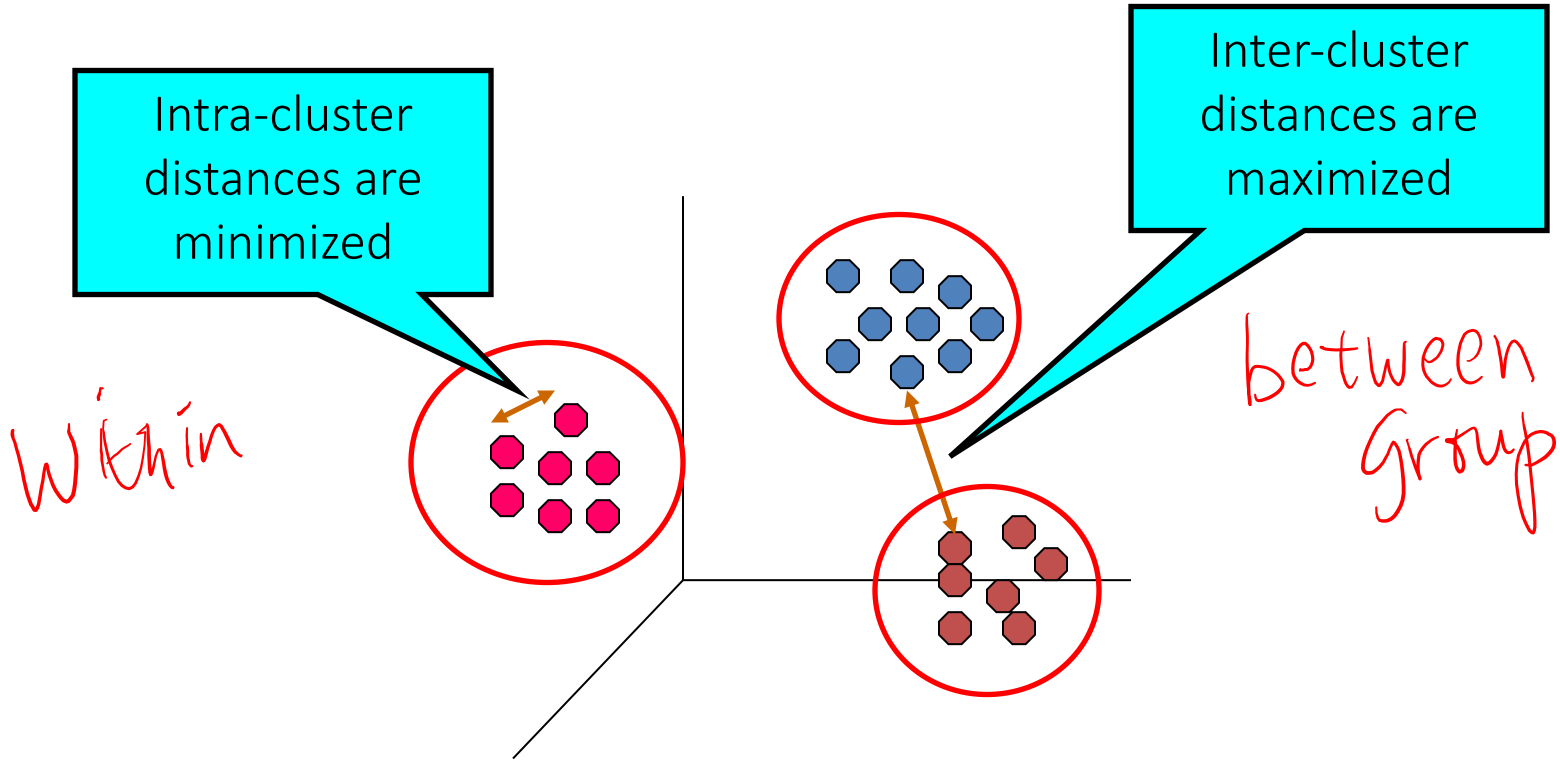
a data matrix of n observations on p variables $x_1, x_2, \ldots x_p$

Unsupervised learning = learning from raw (unlabeled, unannotated, etc) data, as opposed to supervised data where a classification label of examples is given

- **Data**/*points/instances/examples/samples/records*: [rows]
- **Features**/*attributes/dimensions/independent variables/covariates/predictors/regressors*: [columns]

What is clustering?

- Find groups (clusters) of data points such that data points in a group will be similar (or related) to one another and different from (or unrelated to) the data points in other groups

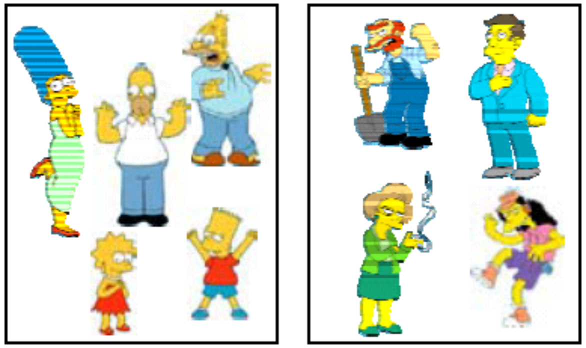


Roadmap: clustering

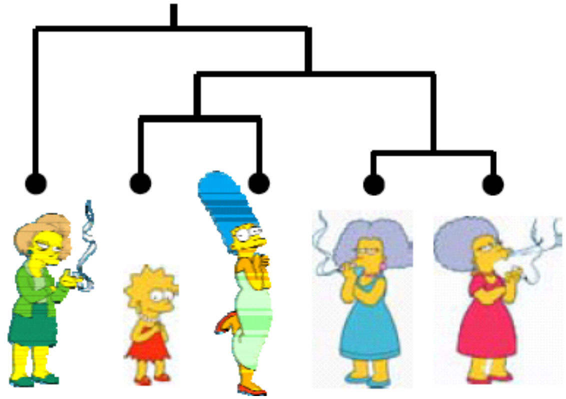
- Definition of "groupness”
- Definition of "similarity/distance"
- Representation for objects
- How many clusters?
- Clustering Algorithms
  - Partitional algorithms
  - Hierarchical algorithms
- Formal foundation and convergence

Clustering Algorithms

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



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



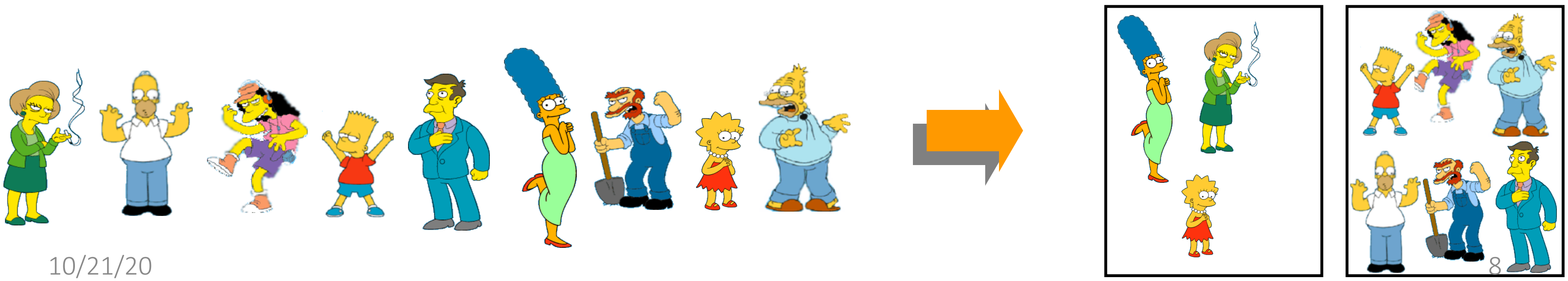
(2) Partitional Clustering

R^p

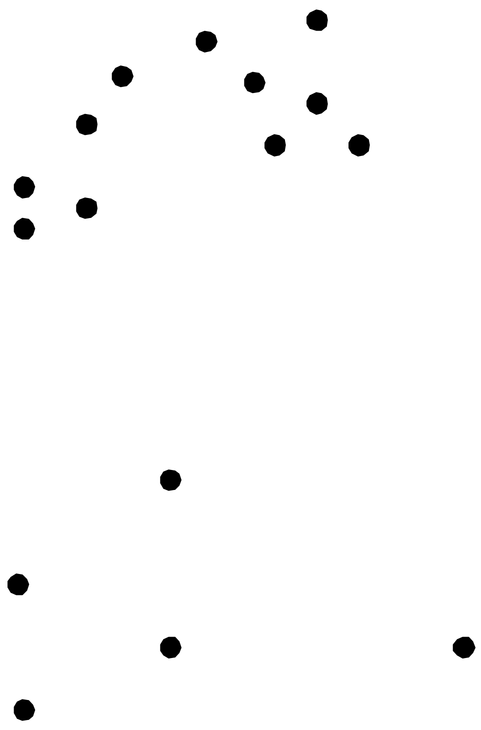
n inputs

k groups

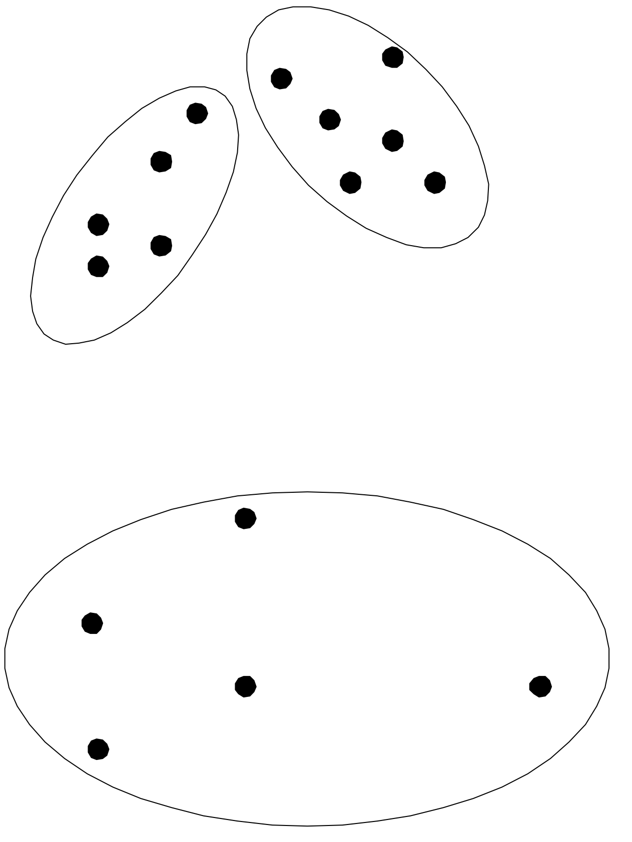
- Nonhierarchical
- Construct a partition of n objects into a set of K clusters
- User has to specify the desired number of clusters K.



Partitional clustering (e.g. K=3)



Original points



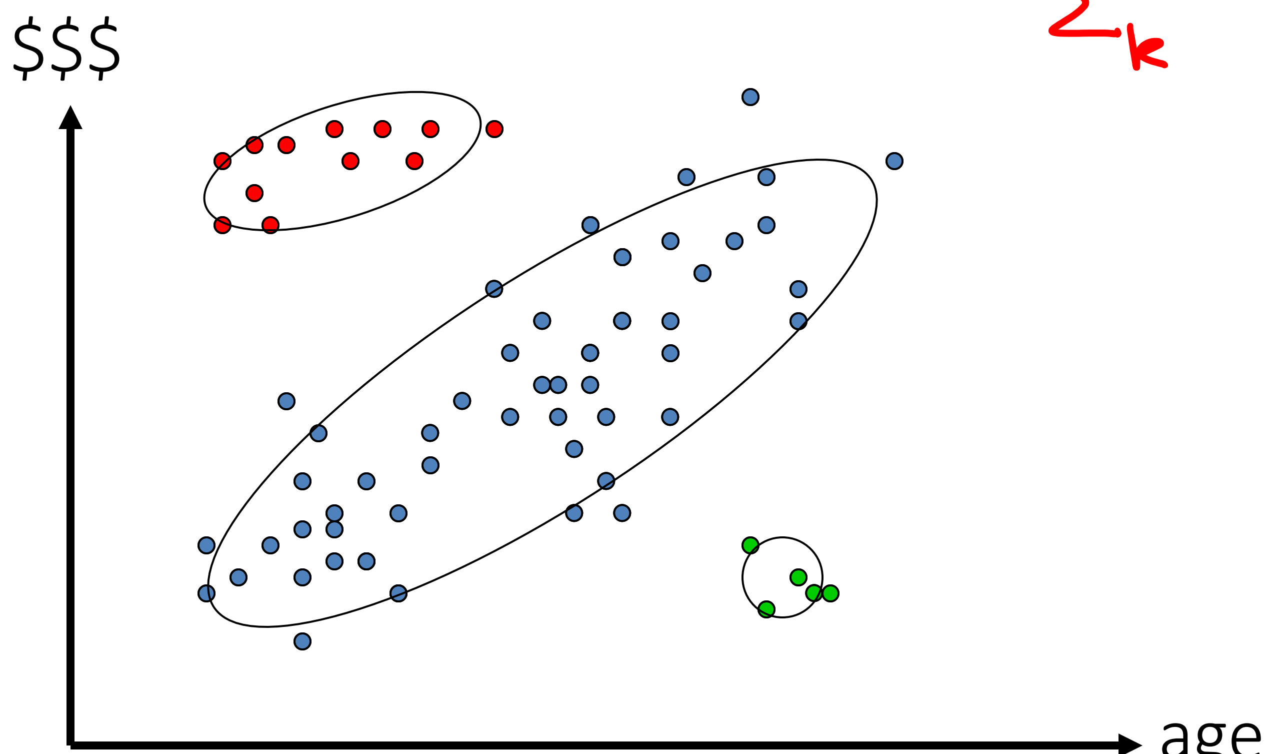
Partitional clustering

Partitional clustering (e.g. K=3)

$k=1,2,\ldots K$

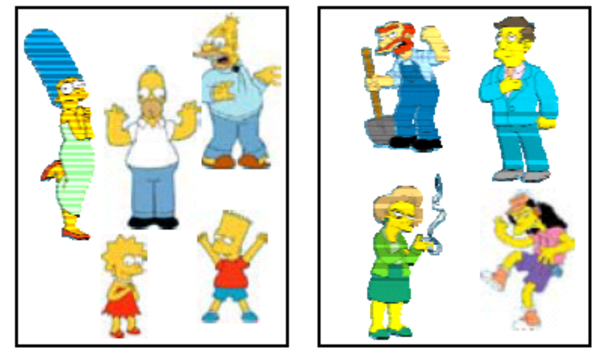
μ_k

Σ_k



Clustering Algorithms

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



Partitioning Algorithms

e.g. 2^{100} two groups

- Given: a set of objects and the number *K*
- Find: a partition of *K* clusters that optimizes a chosen partitioning criterion
  - Globally optimal: exhaustively enumerate all partitions
  - Effective heuristic methods: K-means and K-medoids algorithms

too expensive K^n

K-Means

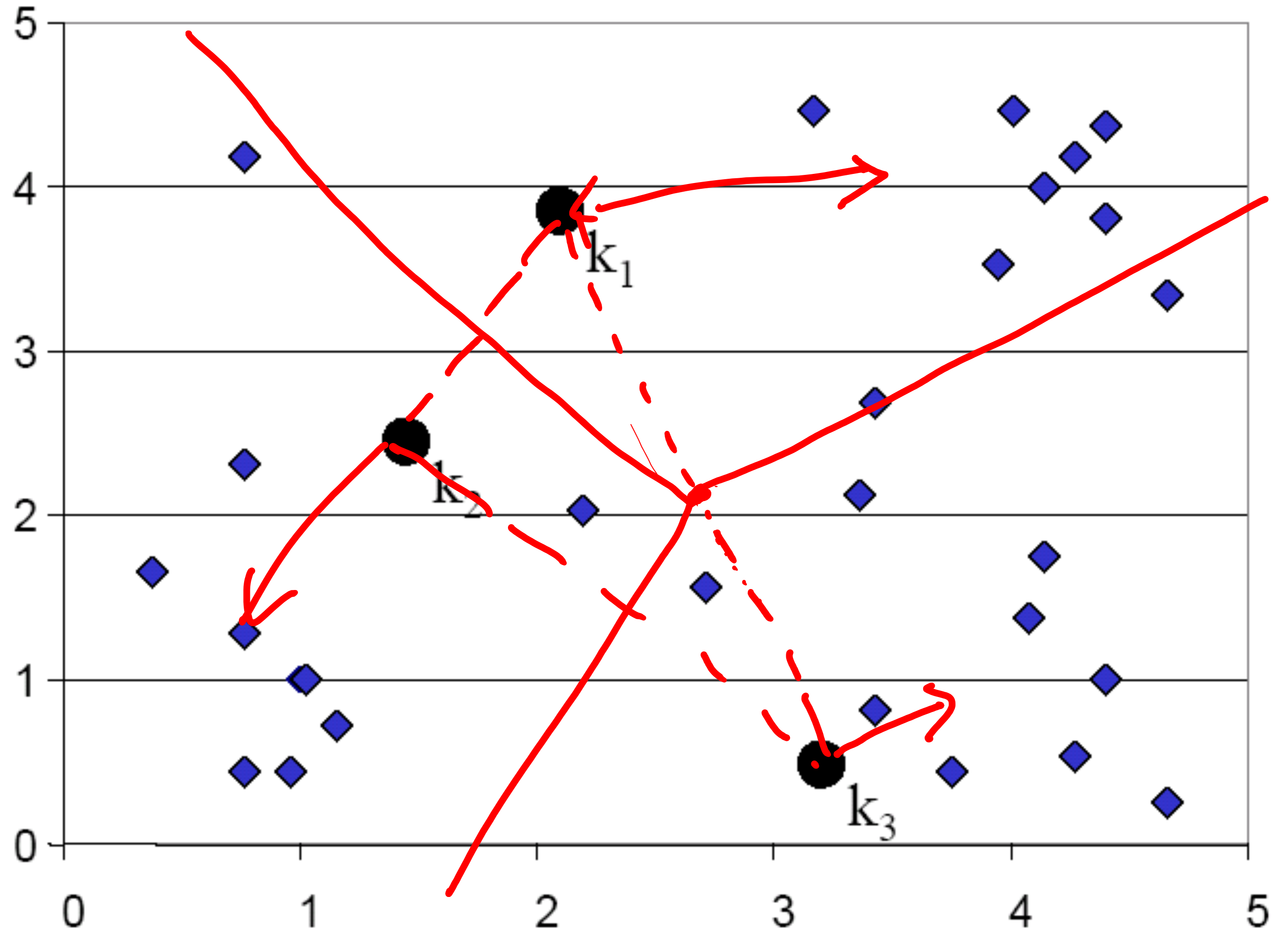
Algorithm

1. Decide on a value for *k*.
2. Initialize the *k* cluster centers randomly if necessary.
3. Decide the class memberships of the *N* objects by assigning them to the nearest cluster centroids (aka the center of gravity or mean)

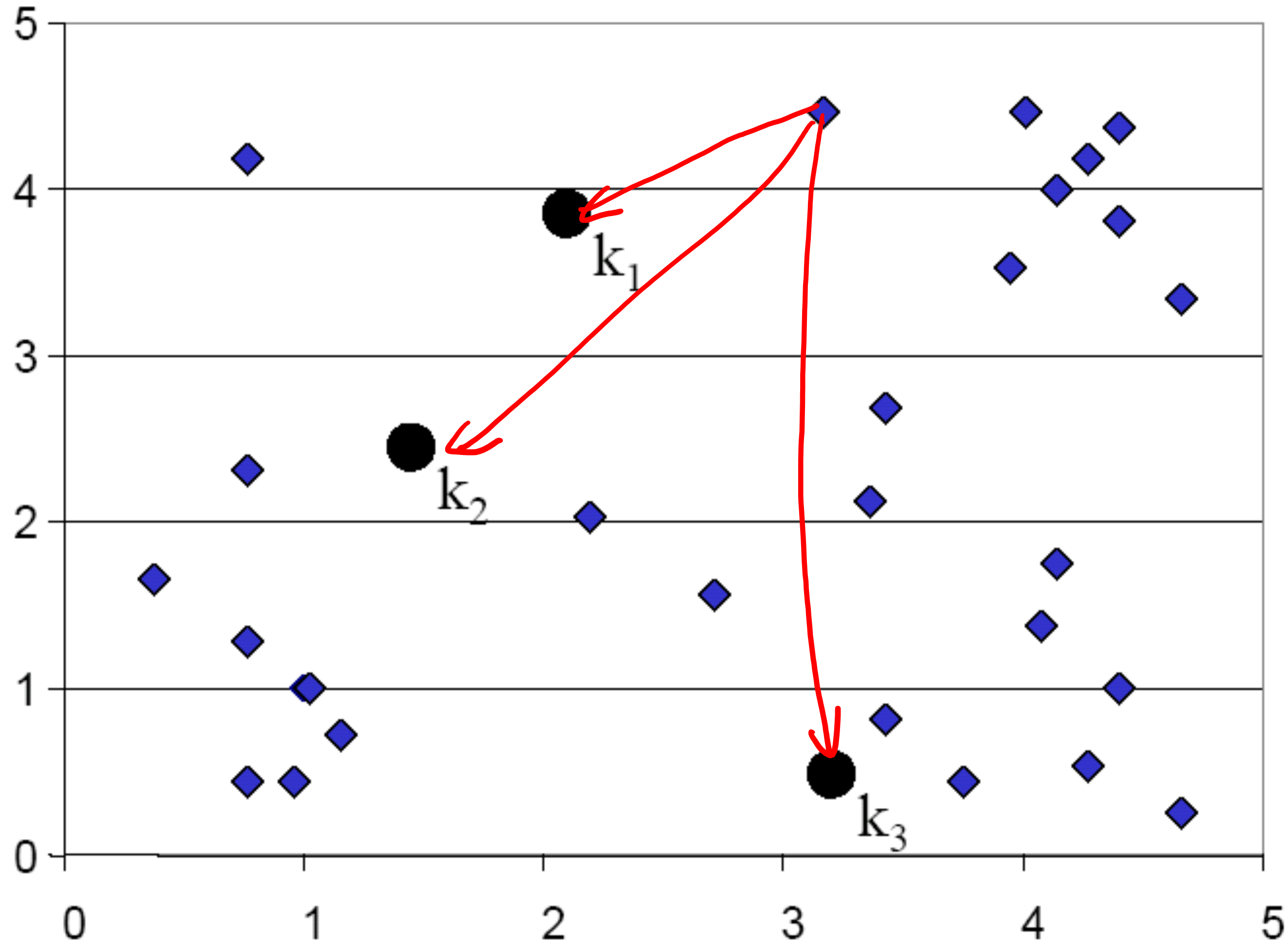
$$\vec{\mu}_k = \frac{1}{\mathcal{C}_k} \sum_{i \in \mathcal{C}_k} \vec{x}_i$$

4. Re-estimate the *k* cluster centers, by assuming the memberships found above are correct.
5. If none of the *N* objects changed membership in the last iteration, exit. Otherwise go to 3.

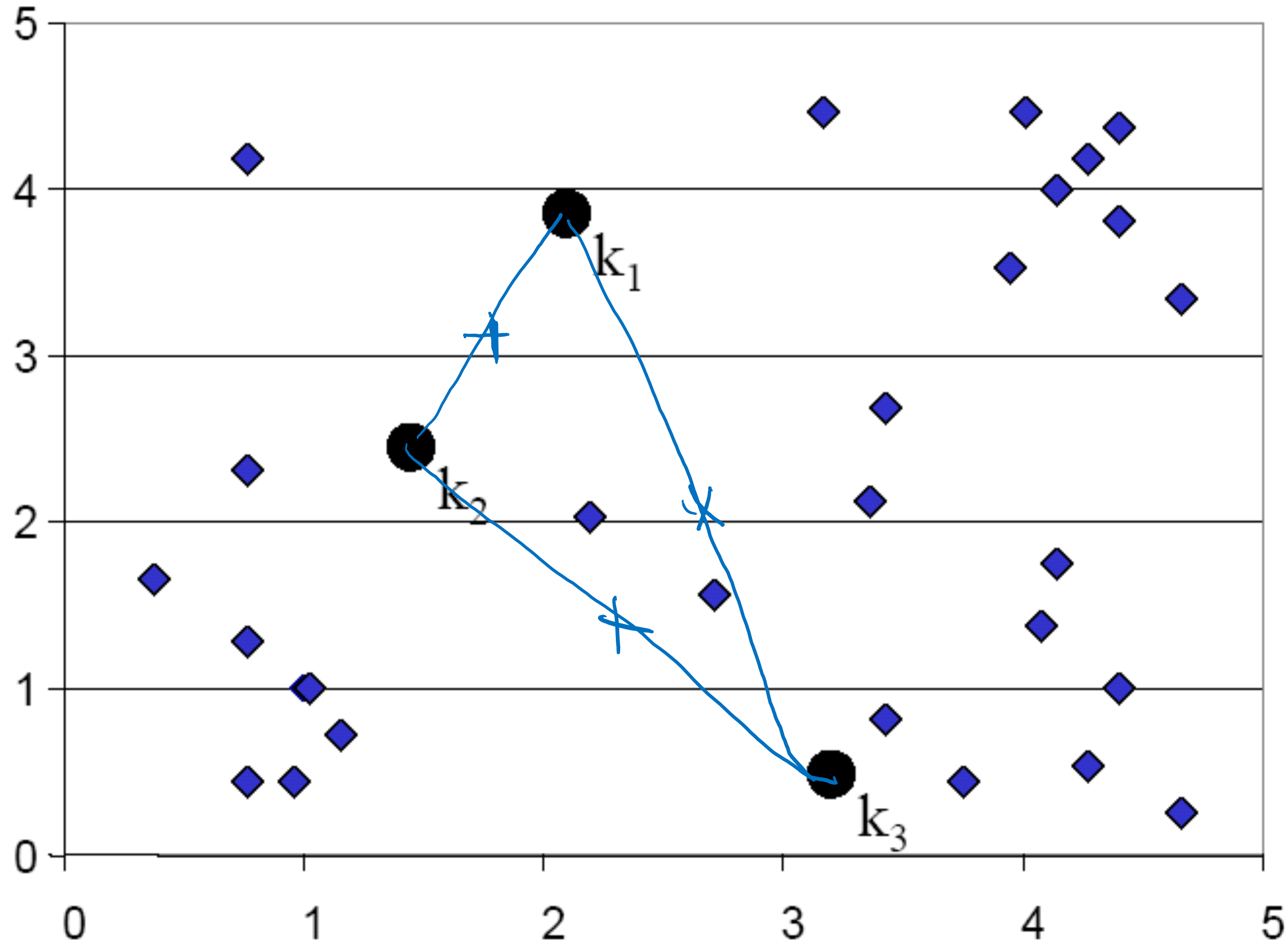
K-means Clustering: Step 1 - random guess of cluster centers



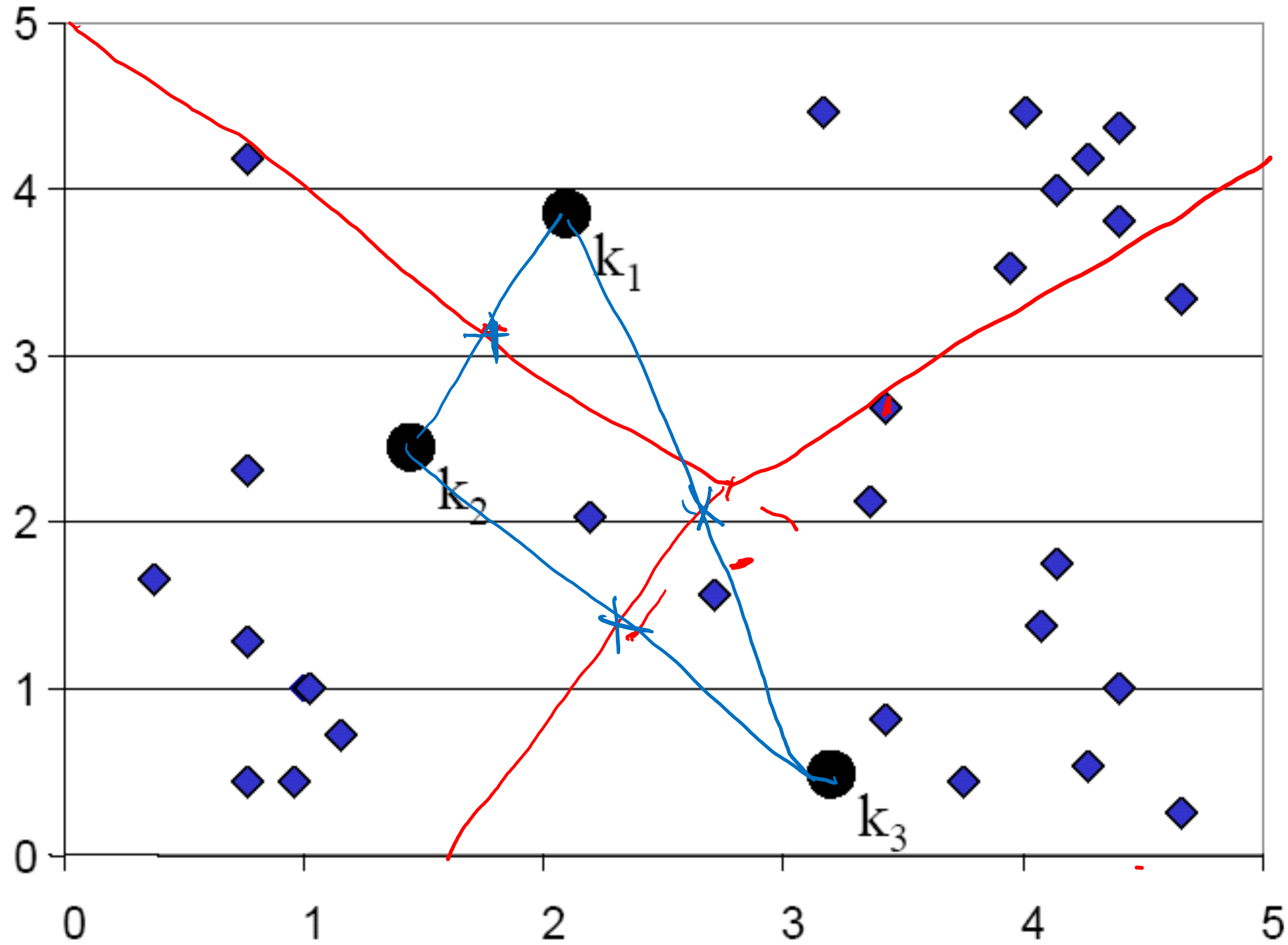
K-means Clustering: Step 1 - random guess of cluster centers



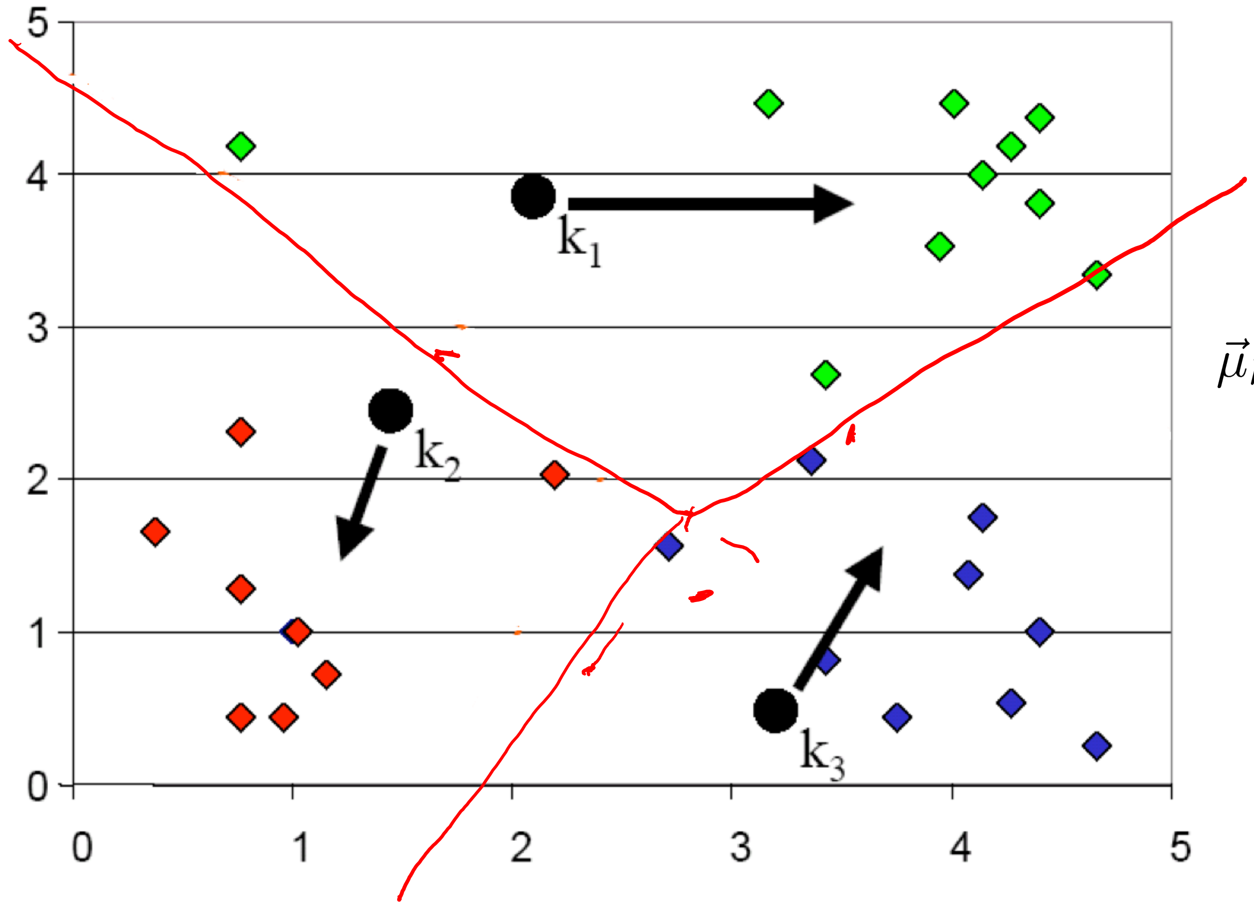
K-means Clustering: Step 1 - random guess of cluster centers



K-means Clustering: Step 1 - random guess of cluster centers

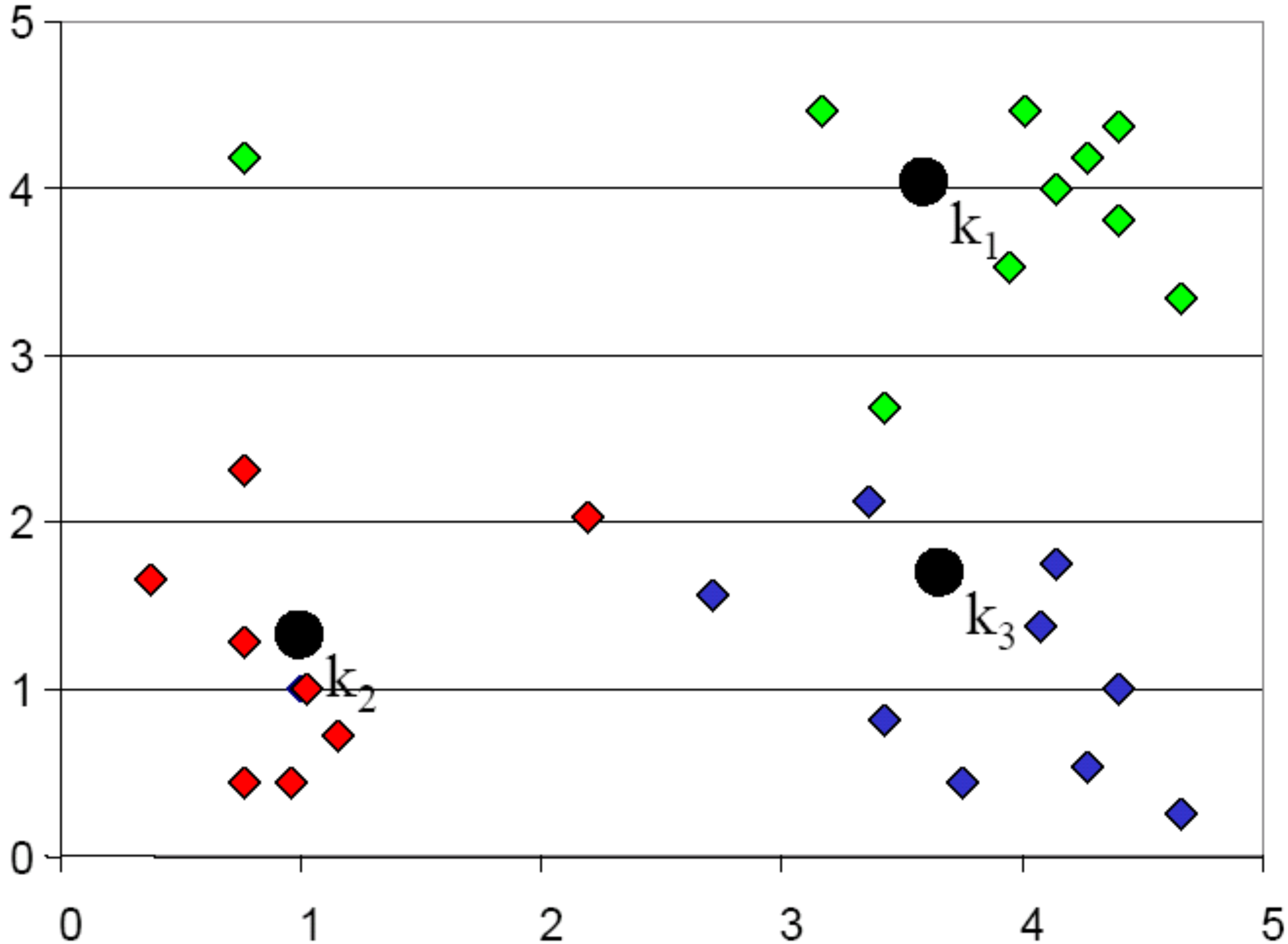


K-means Clustering: Step 2
- Determine the membership of each data points



$$\vec{\mu}_k = \frac{1}{\mathcal{C}_k} \sum_{i \in \mathcal{C}_k} \vec{x}_i$$

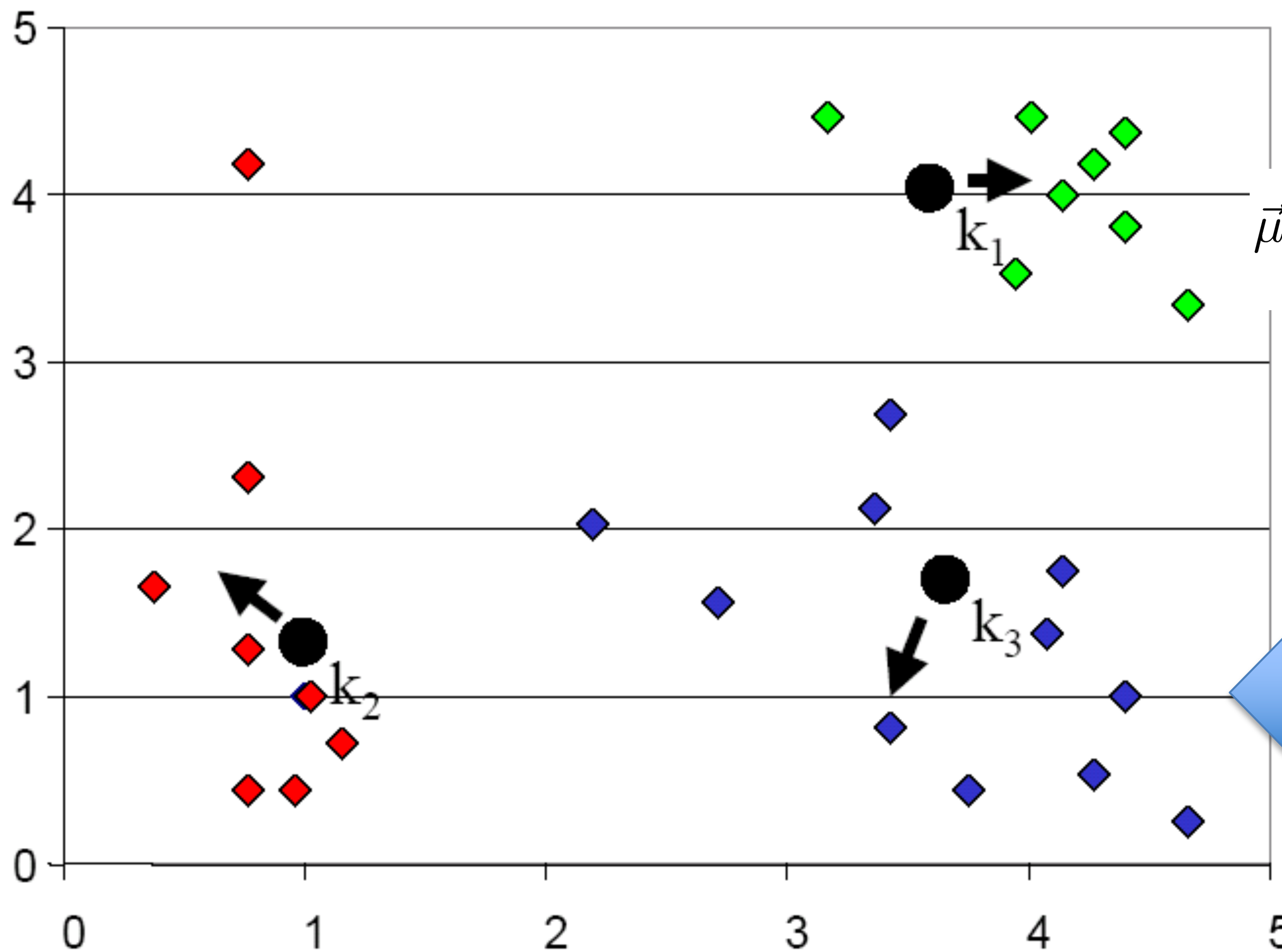
K-means Clustering: Step 3
- Adjust the cluster centers



K-means Clustering: Step 4 - redetermine membership

O(Np)

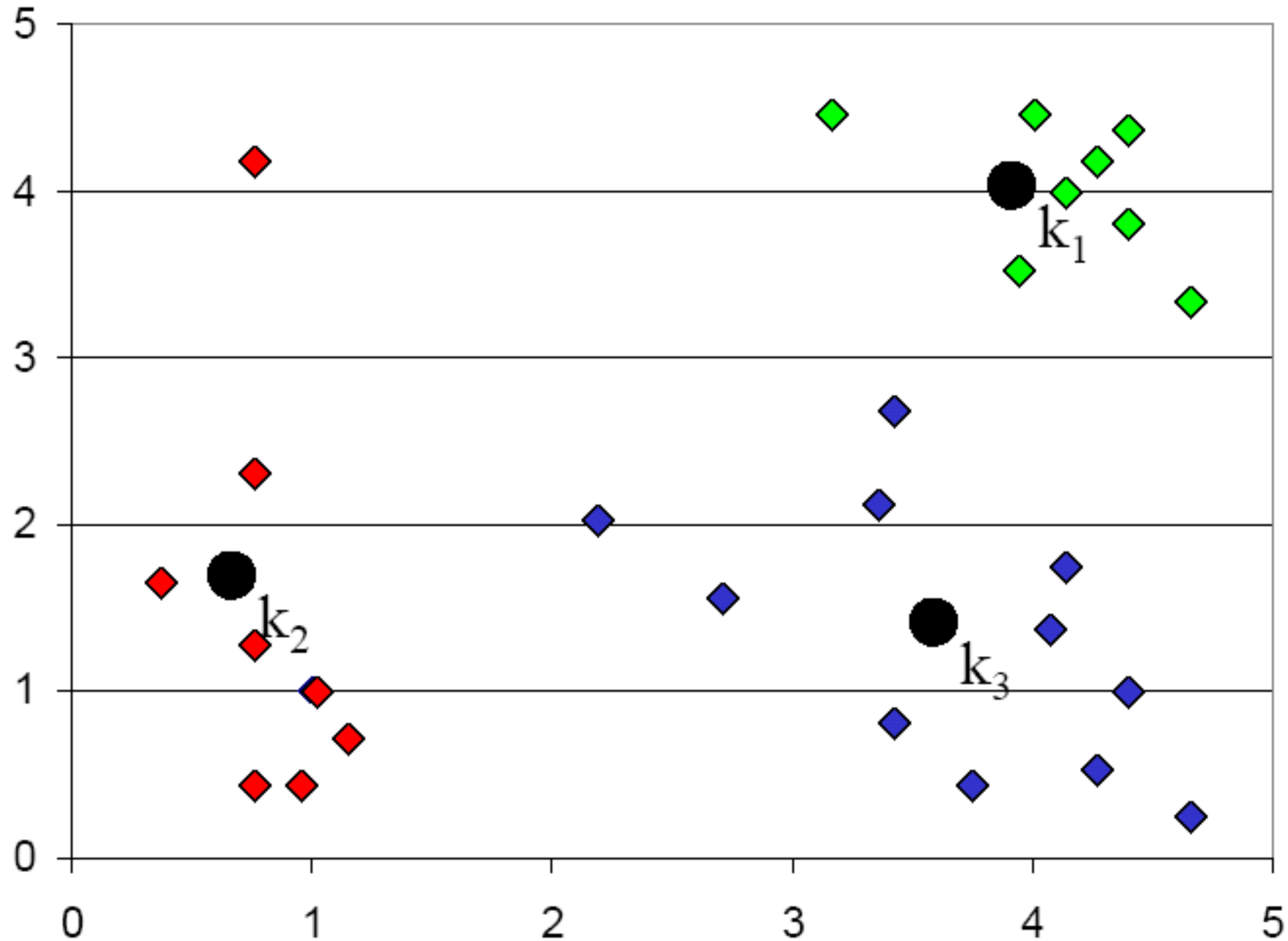
2



$$\vec{\mu}_k = \frac{1}{\mathcal{C}_k} \sum_{i \in \mathcal{C}_k} \vec{x}_i$$

Blue cluster gets more points

K-means Clustering: Step 5 - readjust cluster centers



K-means Clustering: Step 5 - readjust cluster centers

$O(NKP)$

$\Uparrow$

$x_1, x_2, \cdots, x_n$

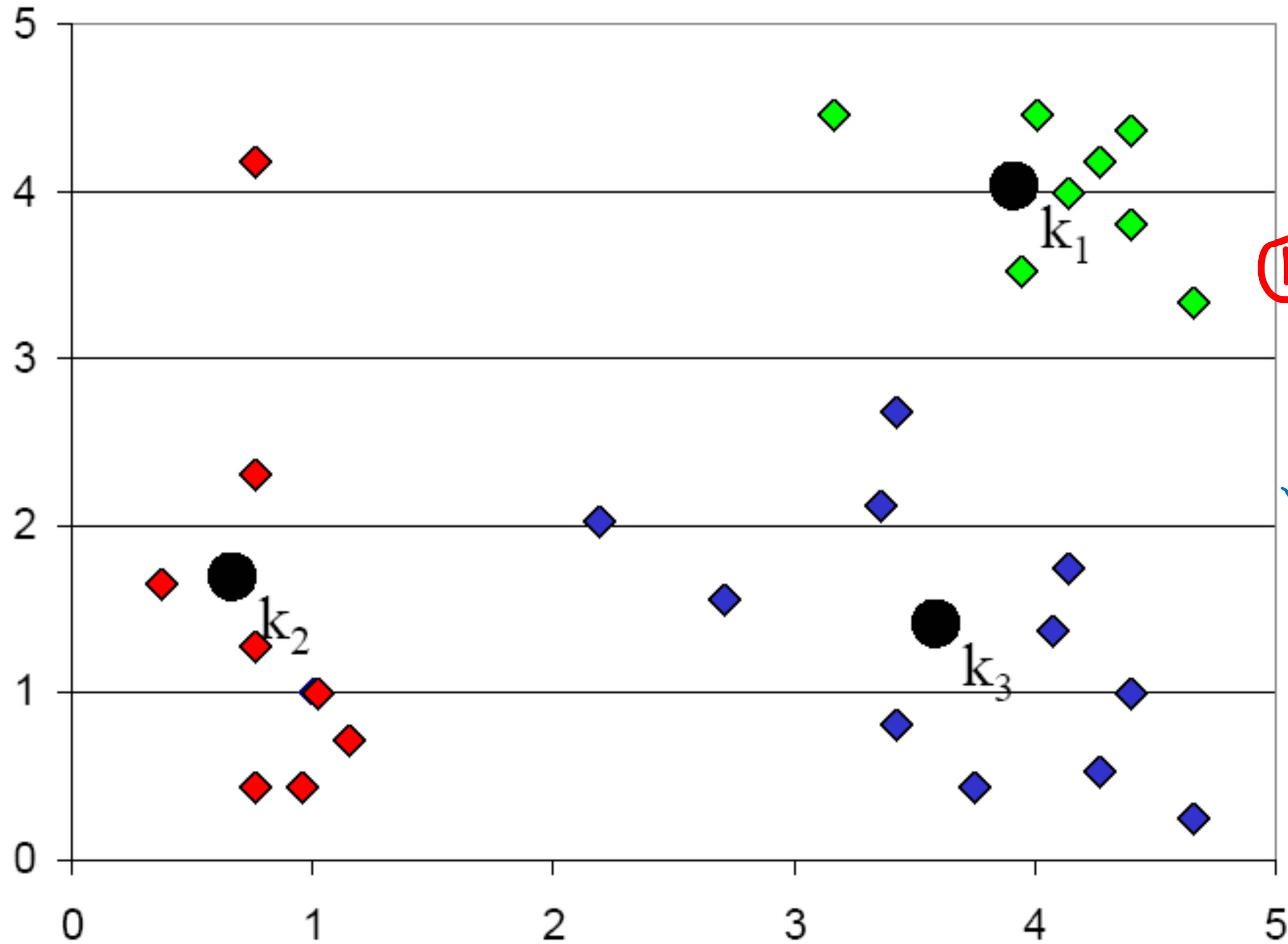
① $x_i, \forall i$

$\{k_1, k_2, k_3\}$

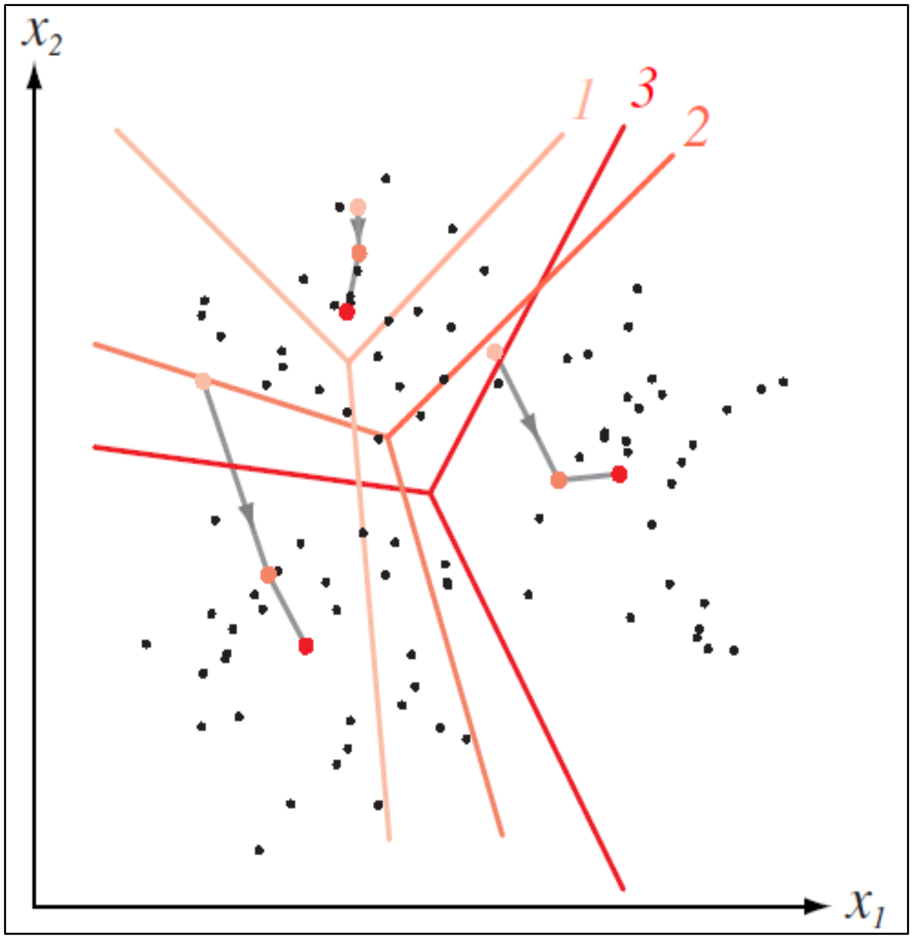
?? $O(KP)$

$d(k_j, x_i)$

$\operatorname{argmin}_{j=1,2,\cdots,K}$



How K-means partitions?



For each set of K centroids (when fixed),

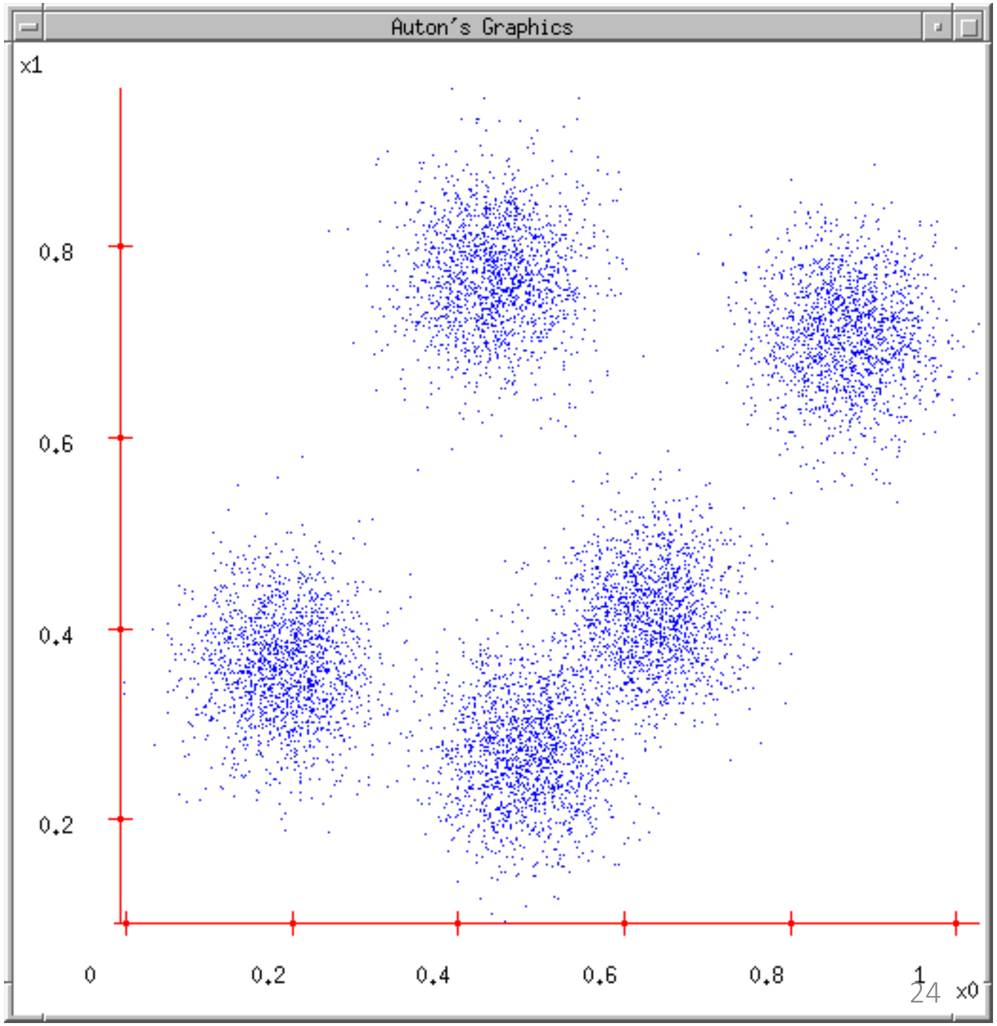
they partition the whole data space into K mutually exclusive subspaces to form a partition.

Changing positions of K centroids leads to a new partitioning.

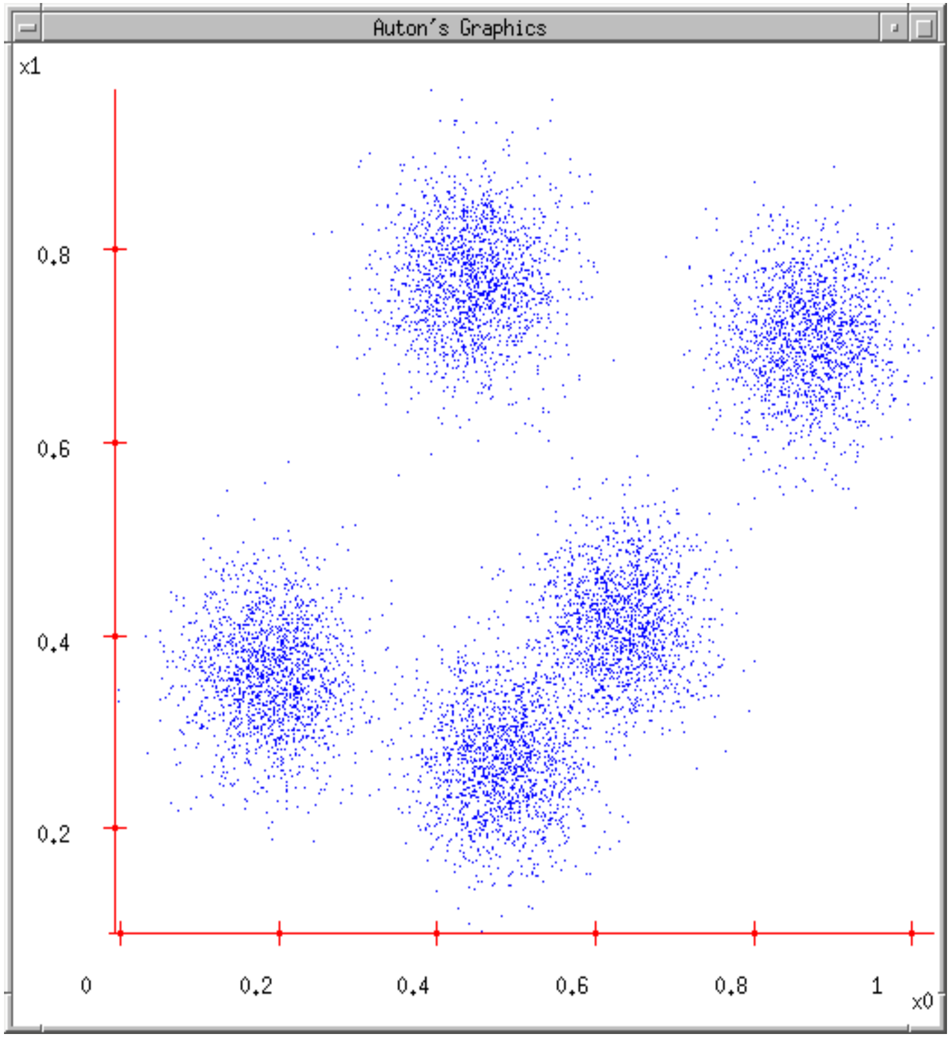
K-means: another Demo

- K-means
  - Start with a random guess of cluster centers
  - Determine the membership of each data points
  - Adjust the cluster centers

$$\vec{\mu}_k = \frac{1}{\mathcal{C}_k} \sum_{i \in \mathcal{C}_k} \vec{x}_i$$

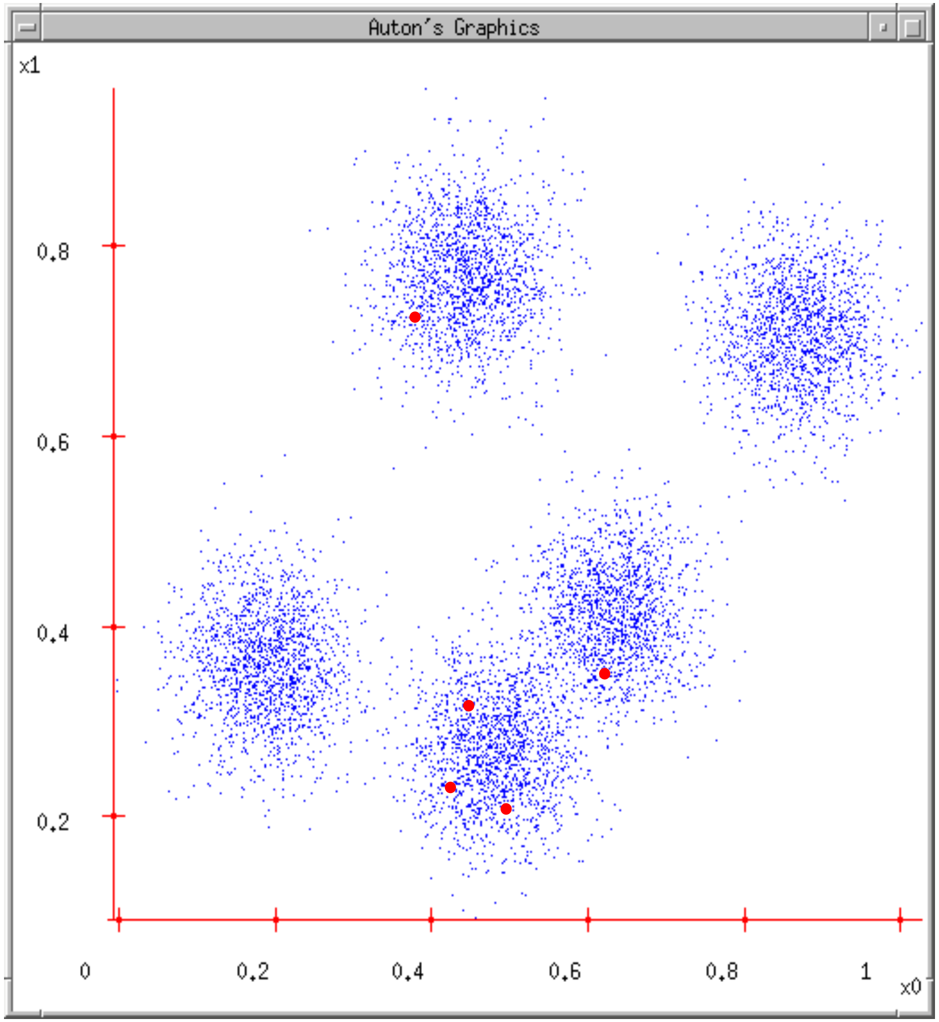


K-means: another Demo



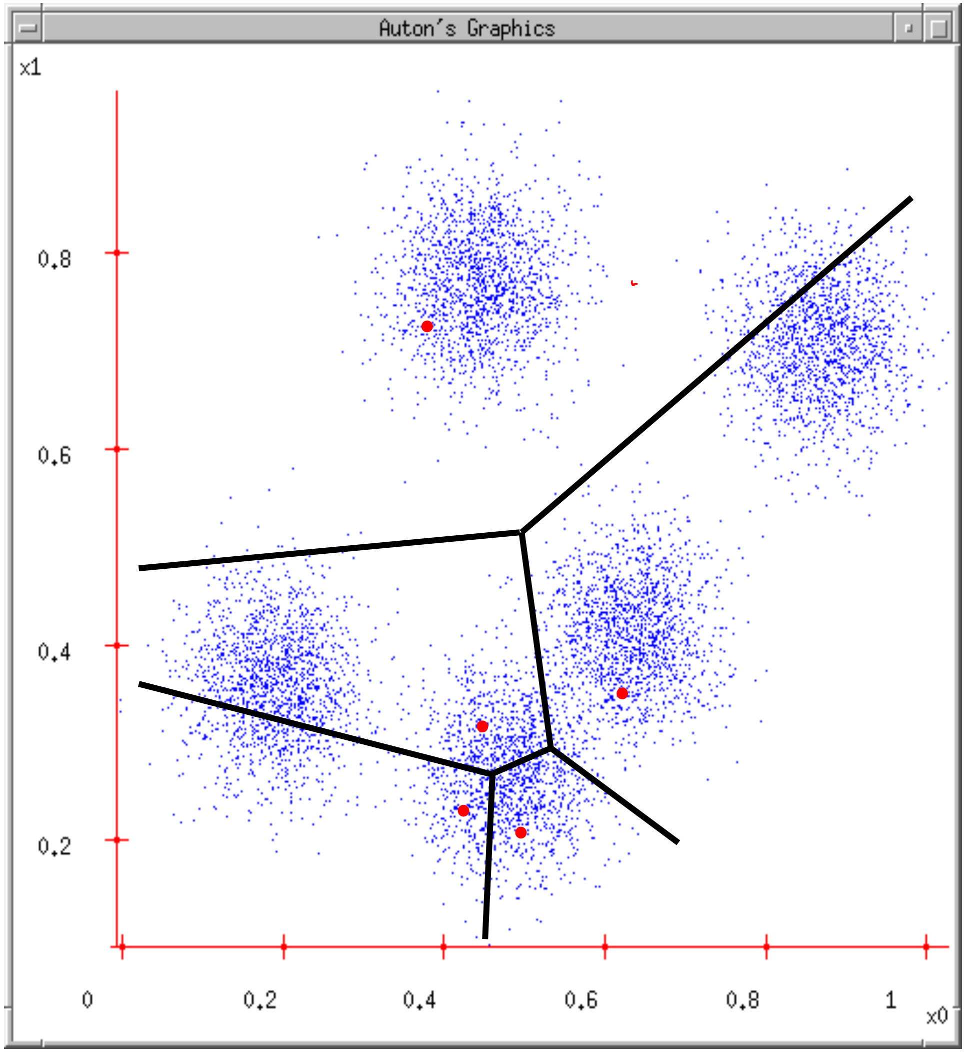
1. User set up the number of clusters they'd like. (e.g. k=5)

K-means: another Demo



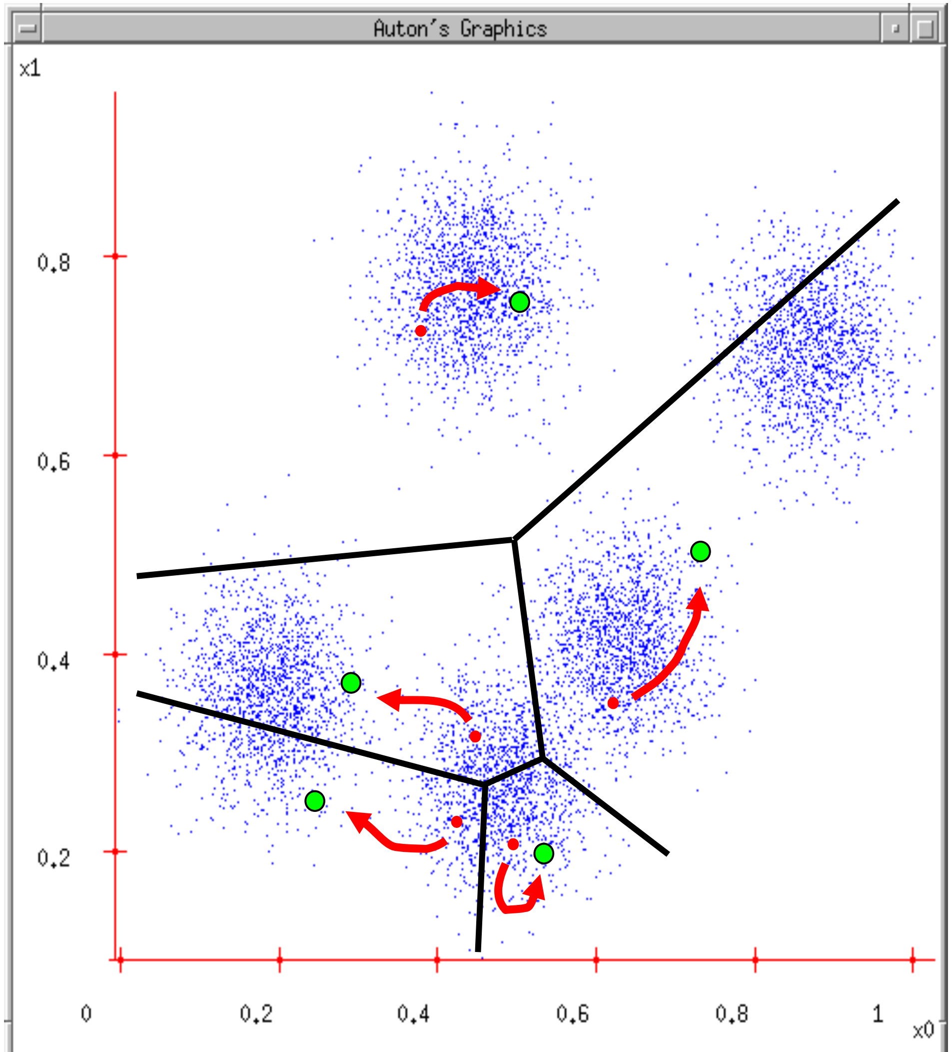
1. User set up the number of clusters they'd like. (e.g. K=5)
2. Randomly guess K cluster Center locations

K-means: another Demo



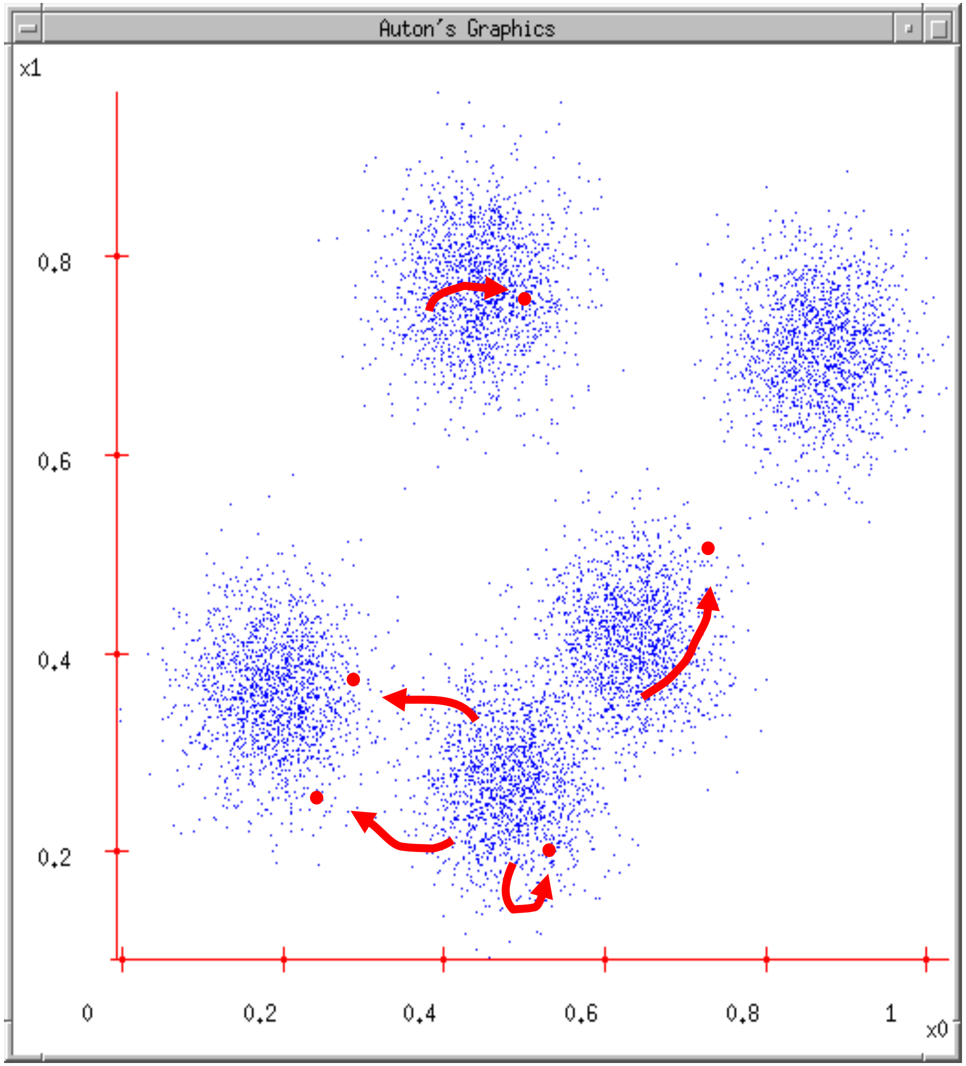
1. User set up the number of clusters they'd like. (e.g. K=5)
2. Randomly guess K cluster Center locations
3. Each data point finds out which Center it's closest to. (Thus each Center "owns" a set of data points)

K-means: another Demo



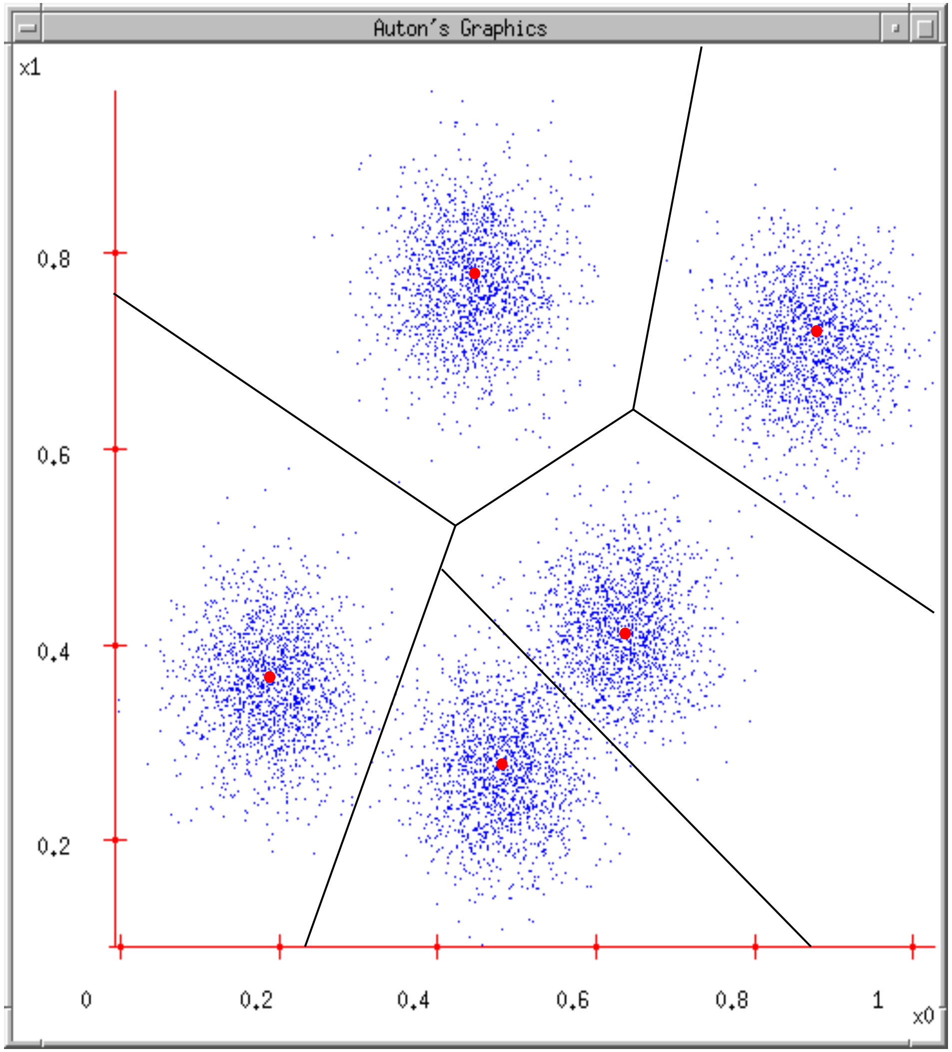
1. User set up the number of clusters they'd like. (e.g. K=5)
2. Randomly guess K cluster centre locations
3. Each data point finds out which centre it's closest to. (Thus each Center "owns" a set of data points)
4. Each centre finds the centroid of the points it owns

K-means: another Demo



1. User set up the number of clusters they'd like. (e.g. K=5)
2. Randomly guess K cluster centre locations
3. Each data point finds out which centre it's closest to. (Thus each centre "owns" a set of data points)
4. Each centre finds the centroid of the points it owns
5. …and jumps there

K-means: another Demo

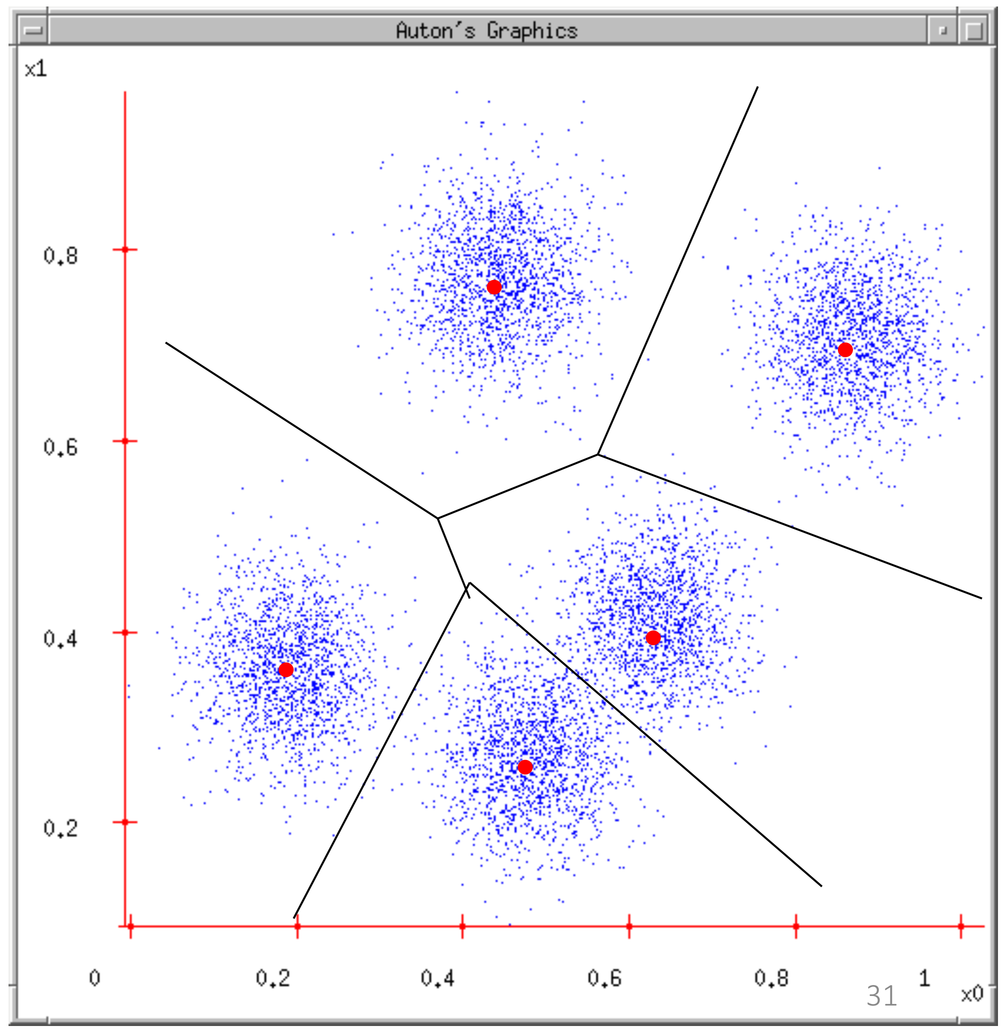


1. User set up the number of clusters they'd like. (e.g. K=5)
2. Randomly guess K cluster centre locations
3. Each data point finds out which centre it's closest to. (Thus each centre "owns" a set of data points)
4. Each centre finds the centroid of the points it owns
5. …and jumps there
6. …Repeat until terminated!

K-means

1. Ask user how many clusters they' d like. (e.g. k=5)
2. Randomly guess k cluster Center locations
3. Each datapoint finds out which Center it' s closest to.
4. Each Center finds the centroid of the points it owns

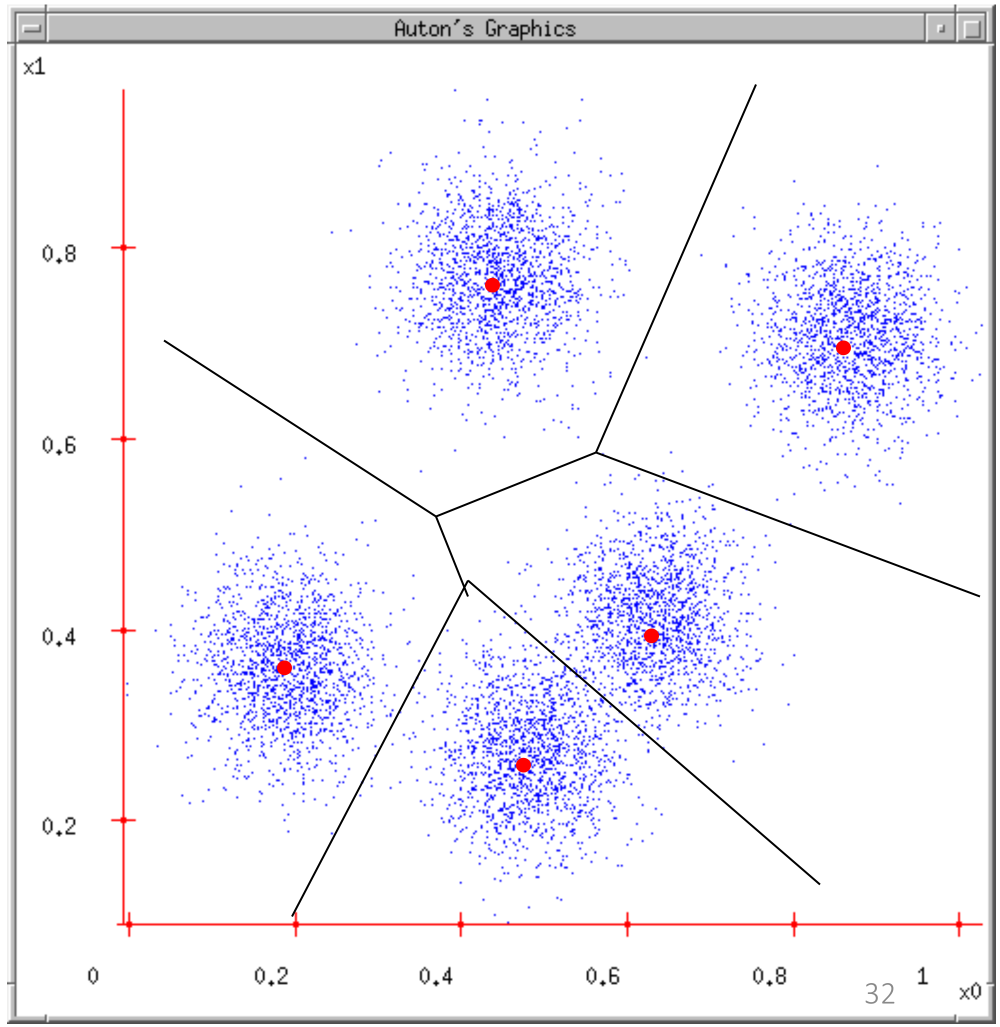
Any Computational Problem?



K-means

1. Ask user how many clusters they'd like. (e.g. k=5)
2. Randomly guess k cluster Center locations
3. Each datapoint finds out which Center it's closest to.
4. Each Center finds the centroid of the points it owns

Any Computational Problem?



Thank You

UVA CS 4774: Machine Learning

S5: Lecture 25: Unsupervised Clustering (II): Partitional

Module II Extra

Dr. Yanjun Qi

University of Virginia
Department of Computer Science

Time Complexity

- Computing distance between two objs is O(*p*) where *p* is the dimensionality of the vectors.
- Reassigning clusters: O(*Knp*) distance computations,
- Computing centroids: Each obj gets added once to some centroid: O(*np*).

$$\vec{\mu}_k = \frac{1}{\mathcal{C}_k} \sum_{i \in \mathcal{C}_k} \vec{x}_i$$

- Assume these two steps are each done once for l iterations: O(lKnp).

vs. Hierarchical

Time Complexity

→ n^2
→ $l\cancel{k}$

- Computing distance between two objs is O(*p*) where *p* is the dimensionality of the vectors.

Step 3

- Reassigning clusters: O(*Knp*) distance computations,

Step 2

- Computing centroids: Each obj gets added once to some centroid: $\underbrace{O(np)}$.

$$\vec{\mu}_k = \frac{1}{\mathcal{C}_k} \sum_{i \in \mathcal{C}_k} \vec{x}_i$$

- Assume these two steps are each done once for l iterations: O(lKnp).

vs. $O(n^3p)$ Hierarchical

Vs. Hierarchical Clustering Time Complexity

- Computing distance between two objs is O(*p*) where *p* is the dimensionality of the vectors.

- (Re-) calculating pairwise dist matrix: O($n^2 p$) distance computations,

- Computing current best cluster : $O(n^2)$

A total of n−1 merging iterations

$O(n^3 p)$

Vs. Hierarchical Clustering Cost analysis

$\vec{x}_i \in R^p$, $(\vec{x}_1, \vec{x}_2, \dots, \vec{x}_N)$

time: $O(\text{dist}(\vec{x}_i, \vec{x}_j)) \sim O(p)$

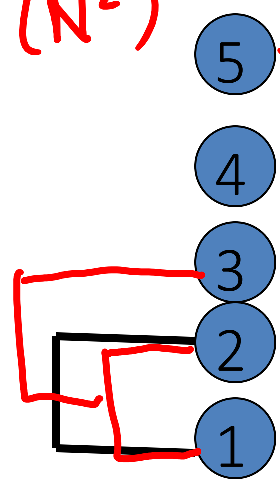
$O(\text{pairse Matrix}) \sim O(pN^2)$

$\text{BestCluster} \sim O(N^2)$

$$
\begin{array}{c|ccccc}
 & 1 & 2 & 3 & 4 & 5 \\
\hline
1 & 0 & & & & \\
2 & 2 & 0 & & & \\
3 & 6 & 3 & 0 & & \\
4 & 10 & 9 & 7 & 0 & \\
5 & 9 & 8 & 5 & 4 & 0
\end{array}
$$

$N \times N$

$$
\begin{array}{c|cccc}
 & (1,2) & 3 & 4 & 5 \\
\hline
(1,2) & 0 & & & \\
3 & 3 & 0 & & \\
4 & 9 & 7 & 0 & \\
5 & 8 & 5 & 4 & 0
\end{array}
\Rightarrow O(N)
$$



$O(N)$

$O(N)$

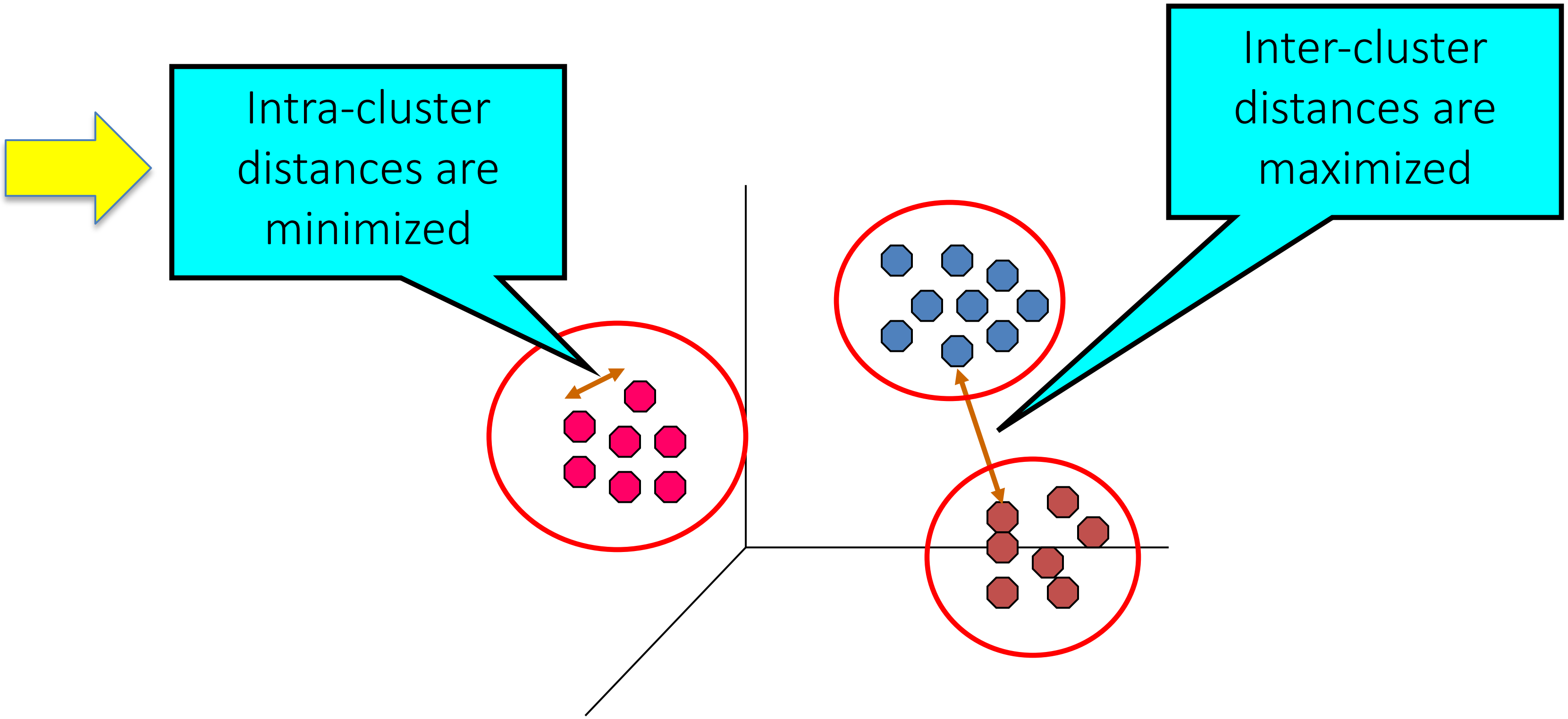
A total of n−1 merging iterations

Roadmap: clustering

- Definition of "groupness”
- Definition of "similarity/distance"
- Representation for objects
- How many clusters?
- Clustering Algorithms
  - Partitional algorithms
  - Hierarchical algorithms
- Formal foundation and convergence

How to Find good Clustering?

- Find groups (clusters) of data points such that data points in a group will be similar (or related) to one another and different from (or unrelated to) the data points in other groups



How to Find good Clustering? E.g.

- Minimize the sum of distance within clusters

$j=1,2,\cdots,K$

$$\underset{\{\vec{C}_j, m_{i,j}\}}{\arg\min} \sum_{j=1}^{K} \sum_{i=1}^{n} m_{i,j} \left(\vec{x}_i - \vec{C}_j\right)^2$$

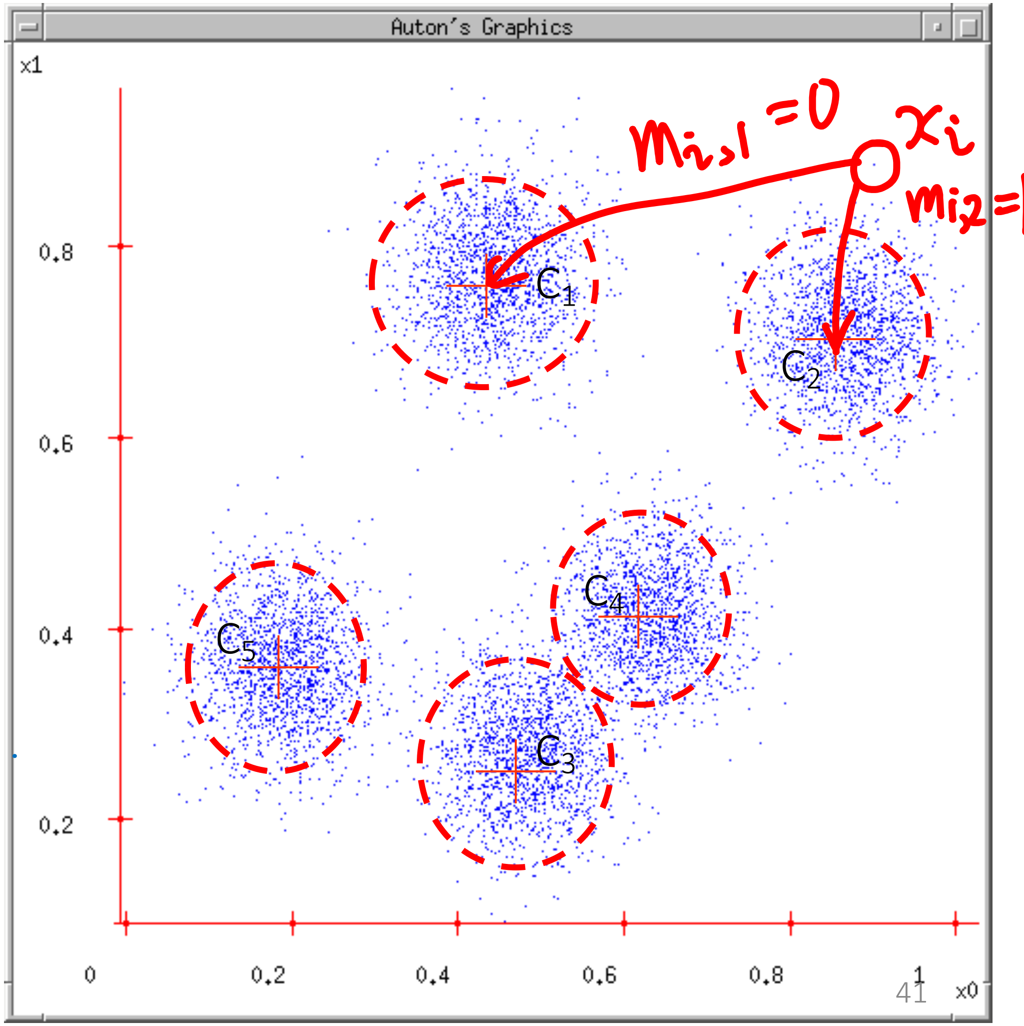
$$m_{i,j} = \begin{cases} 1 & \vec{x}_i \in \text{ the j-th cluster} \\ 0 & \vec{x}_i \notin \text{ the j-th cluster} \end{cases}$$

$i=1,\cdots,n$

$$\sum_{j=1}^{K=5} m_{i,j} = 1$$

$\underline{M_{N\times K}}$

$\rightarrow$ any $\vec{x}_i \in$ a single cluster



k=5

$$\underset{\{\vec{C}_j, m_{i,j}\}}{\arg\min} \sum_{j=1}^{K=5} \sum_{i=1}^{n} m_{i,j} \left(\vec{x}_i - \vec{C}_j\right)^2$$

$\rightarrow$ when given $\{m_{ij}\}$, $loss(\vec{C_j}) = \sum_{j=1}^{k} \sum_{i=1}^{n} m_{ij} (\vec{x_i} - \vec{C_j})^2$

$$\frac{\partial\, loss(\vec{C_j})}{\partial \vec{C_j}} = 0 \Rightarrow \vec{C}_j = \frac{\sum_{i=1}^{n} m_{i,j} \vec{x}_i}{\sum_{i=1}^{n} m_{i,j}}$$

$\rightarrow$ when given $\{\vec{C_j}\}$, $\frac{\partial\, loss(m_{ij})}{\partial m_{ij}} = 0 \Rightarrow$

$$\Rightarrow m_{i,j} = \begin{cases} 1 & j = \underset{k}{\arg\min} (\vec{x}_i - \vec{C}_j)^2 \\ 0 & \text{otherwise} \end{cases}$$

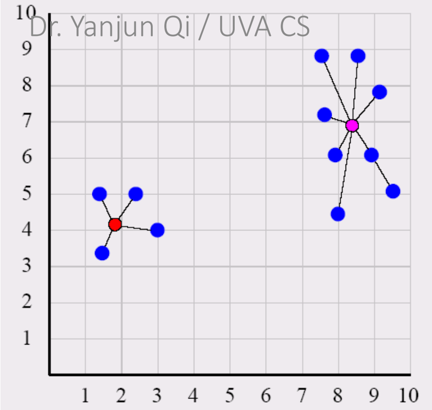
Iterative Optimization

$$\underset{\{\vec{C}_j, m_{i,j}\}}{\arg\min} \sum_{j=1}^{K} \sum_{i=1}^{n} m_{i,j} \left(\vec{x}_i - \vec{C}_j \right)^2$$

Memberships $\{m_{i,j}\}$ and centers $\{C_j\}$ are correlated.

$$\text{Given centers } \{\vec{C}_j\},\ m_{i,j} = \begin{cases} 1 & j = \underset{k}{\arg\min}(\vec{x}_i - \vec{C}_j)^2 \\ 0 & \text{otherwise} \end{cases}$$

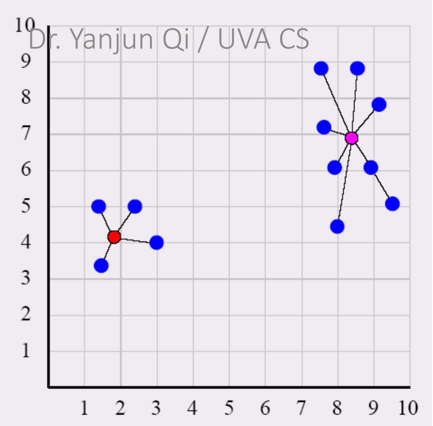
$$\text{Given memberships } \{m_{i,j}\},\ \vec{C}_j = \frac{\sum_{i=1}^{n} m_{i,j} \vec{x}_i}{\sum_{i=1}^{n} m_{i,j}} = \frac{\text{sum of points in cluster } j}{\text{\# points in cluster } j}$$



Convergence

- Why should the K-means algorithm ever reach a fixed point?
  - A state in which clusters don't change.
- K-means is a special case of a general procedure known as the Expectation Maximization (EM) algorithm.
  - EM is known to converge.
  - Number of iterations could be large.

Convergence



- Why should the K-means algorithm ever reach a fixed point?
  - A state in which clusters don't change.
- K-means is a special case of a general procedure known as the Expectation Maximization (EM) algorithm.
  - EM is known to converge.
  - Number of iterations could be large.
- Optimize the goodness measure (i.e., minimize the Loss function)
  - sum of squared distances from cluster centroid:
- Reassignment monotonically decreases the goodness measure since each vector is assigned to the closest centroid.

Convergence Property of EM/Kmeans (EXTRA)

https://stats.stackexchange.com/questions/303448/rate-of-convergence-of-em-algorithm

In the general case you need to verify that your problem setup satisfies certain properties for the EM algorithm to converge to a stationary point that is a local maximum. Further requirements are needed for global maximum. Assuming these criteria are met you can then quantify the rate of the convergence toward the global optimum.

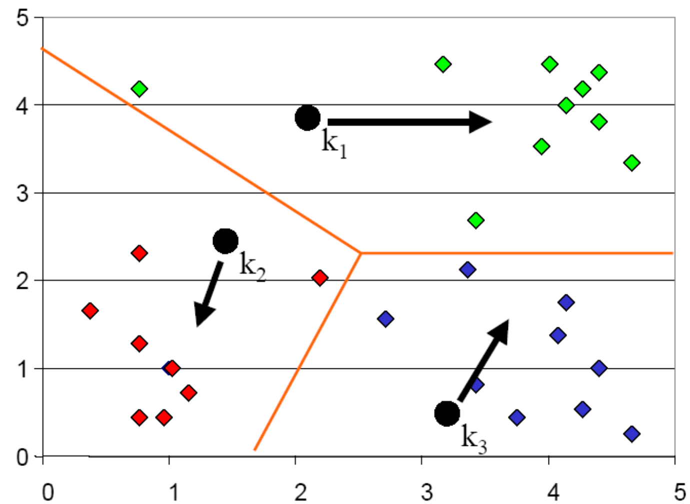
In the general case (and assuming you have global optimum) the most you can usually say is that the EM algorithm is a first order algorithm. First order algorithms are algorithms such that:

$$|\theta^{k+1} - \theta^*| \leq \gamma |\theta^k - \theta^*|.$$

If $\gamma = 1$ then convergence is linear and if $1 < \gamma < 2$ then the algorithm is said to have super-linear convergence and if $\gamma = 2$ is quadratic convergence. The convergence rate really depends on the specifics of the problem. Many examples and an a pedagogic introduction to convergence rates of the EM algorithm are given in the book by McLachlin and Krishnan if you want more details. Xu and Jordan provide an in-depth study for the mixture of Gaussians case.

Seed Choice

- Results can vary based on random seed selection.



K

$C_1, C_2, \ldots, C_K$

- Some seeds can result in poor convergence rate, or convergence to sub-optimal clustering.
  - Select good seeds using a heuristic (e.g., sample least similar to any existing mean)
  - Try out multiple starting points (very important!!!)
  - Initialize with the results of another method.

K-means Clustering

Data	Tabular
↓	↓
Task	Clustering
↓	↓
Representation	Mixture of spherical shapes
↓	↓
Score Function	Sum-of-square distance to centroid
↓	↓
Search/Optimization	K-means algorithm
↓	↓
Models, Parameters	Cluster membership & centroid

$i=1,\dots,n$

$j=1,\dots,k$

$\{\mu_j\}$

$\{C_j\}$

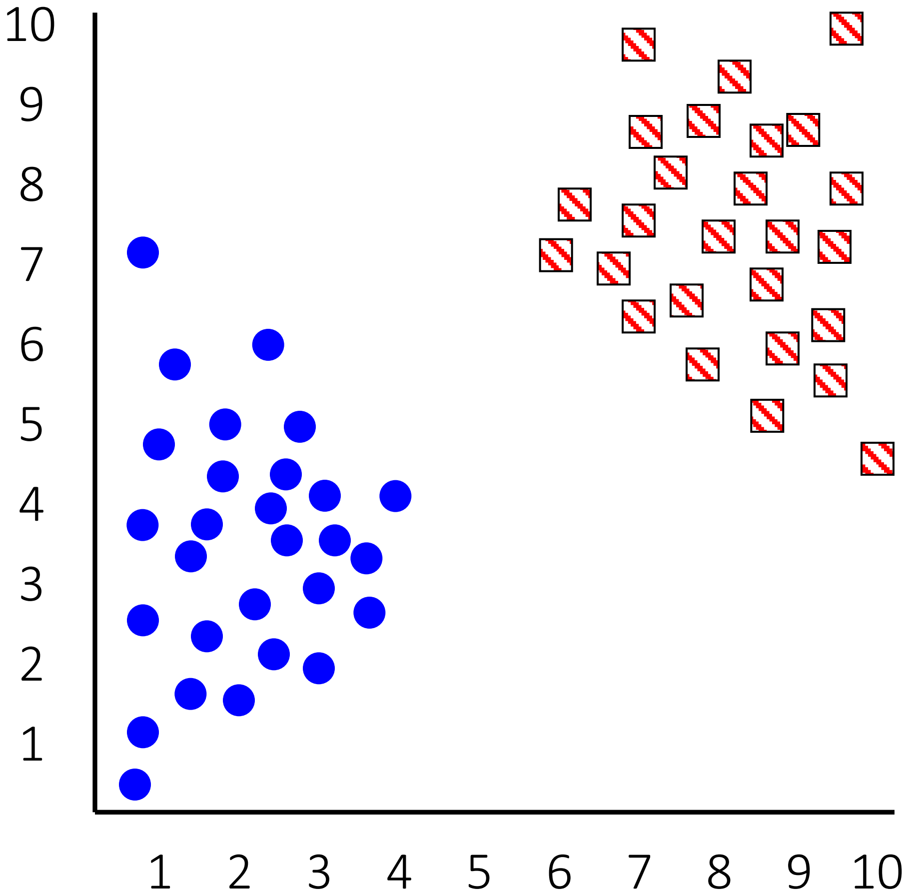
ERP

Roadmap: clustering

- Definition of "groupness”
- Definition of "similarity/distance"
- Representation for objects
- How many clusters?
- Clustering Algorithms
  - Partitional algorithms
  - Hierarchical algorithms
- Formal foundation and convergence

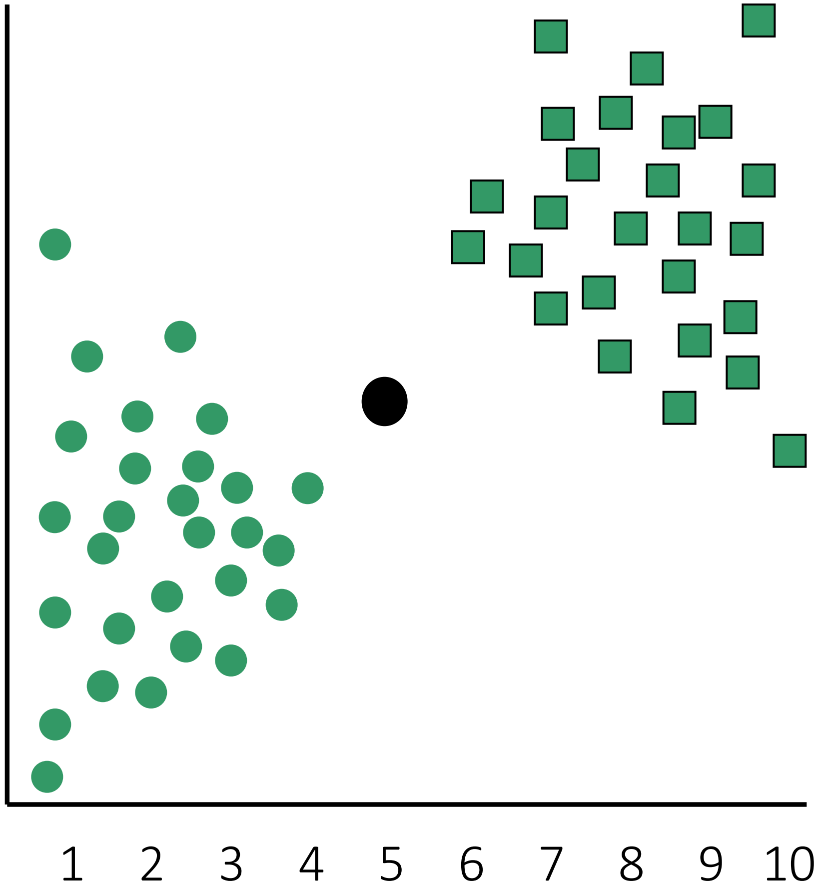
How can we tell the right number of clusters?

In general, this is a unsolved problem. However there exist many approximate methods.



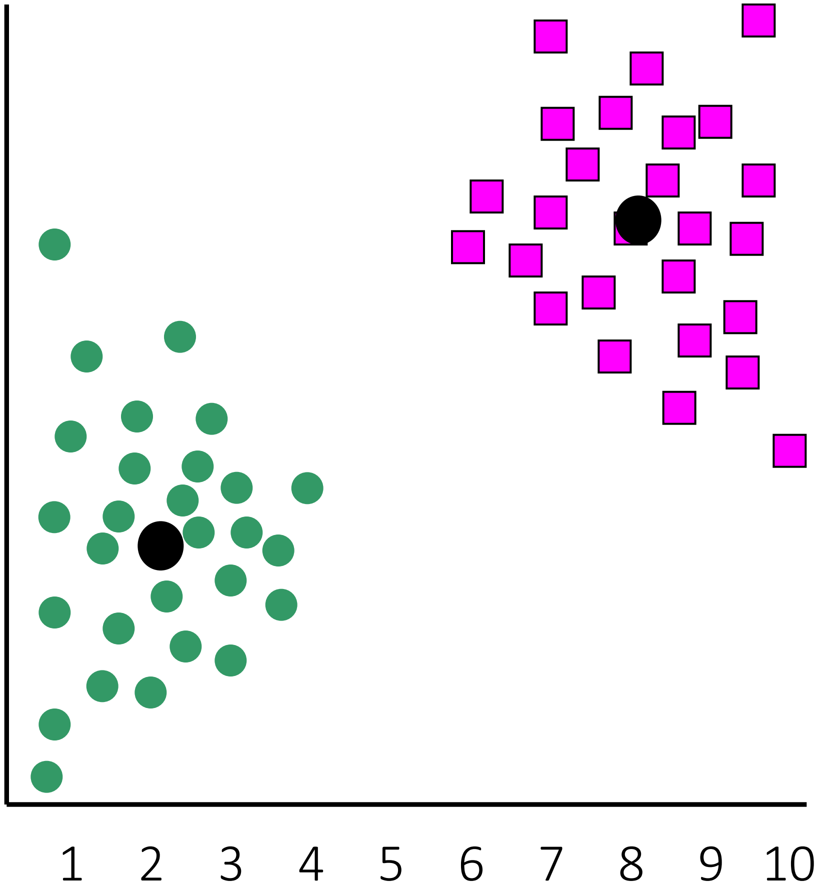
$$\underset{\{\vec{C}_j, m_{i,j}\}}{\arg\min} \sum_{j=1}^{K} \sum_{i=1}^{n} m_{i,j} \left(\vec{x}_i - \vec{C}_j\right)^2$$

When k = 1, the objective function is 873.0



$$\arg\min_{\{\vec{C}_j, m_{i,j}\}} \sum_{j=1}^{K} \sum_{i=1}^{n} m_{i,j} \left(\vec{x}_i - \vec{C}_j \right)^2$$

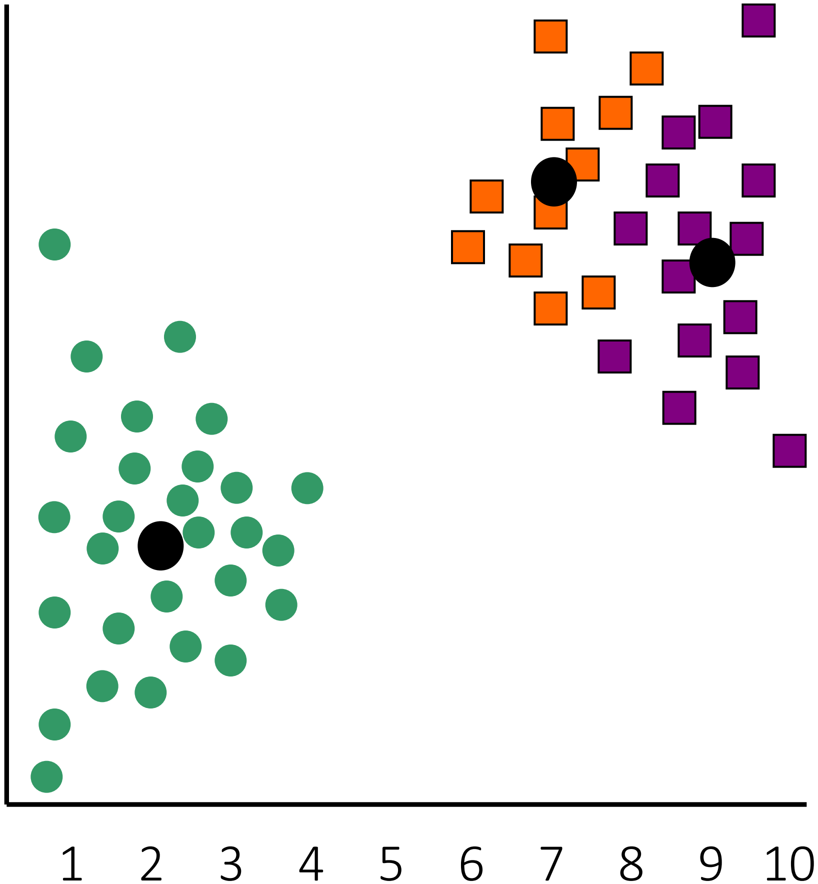
When k = 2, the objective function is 173.1



$$\underset{\{\vec{C}_j, m_{i,j}\}}{\arg\min} \sum_{j=1}^{K} \sum_{i=1}^{n} m_{i,j} \left(\vec{x}_i - \vec{C}_j\right)^2$$

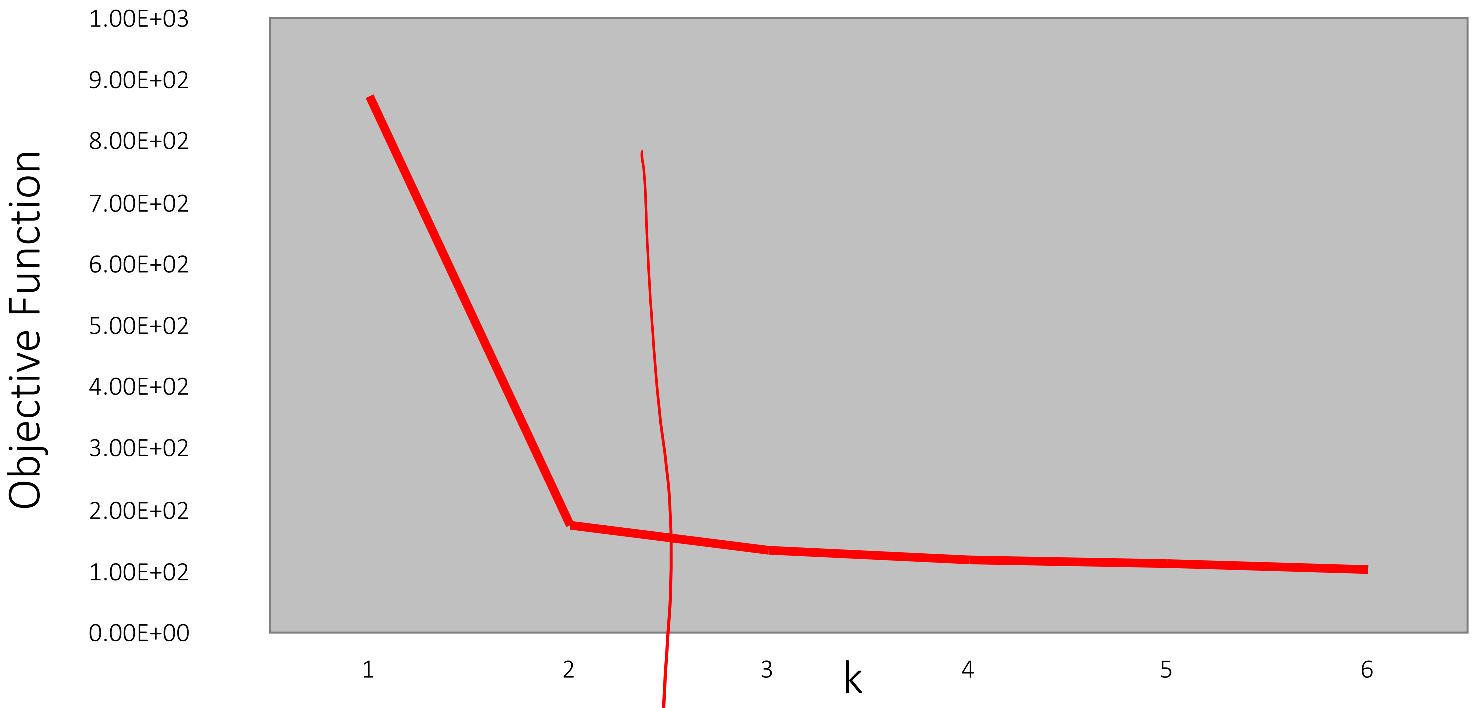
When k = 3, the objective function is 133.6

K=n , obj = 0



We can plot the objective function values for k equals 1 to 6…

The abrupt change at k = 2, is highly suggestive of two clusters in the data. This technique for determining the number of clusters is known as "knee finding" or "elbow finding".



Note that the results are not always as clear cut as in this toy example

What Is A Good Clustering?

- Internal criterion: A good clustering will produce high quality clusters in which:
  - the intra-cluster similarity is high
  - the inter-cluster similarity is low
  - The measured quality of a clustering depends on both the data representation and the similarity measure used

- External criteria for clustering quality
  - Quality measured by its ability to discover some or all of the hidden patterns or latent classes in gold standard data
  - Assesses a clustering with respect to ground truth
  - Example:
    - Purity
    - entropy of classes in clusters (or mutual information between classes and clusters)

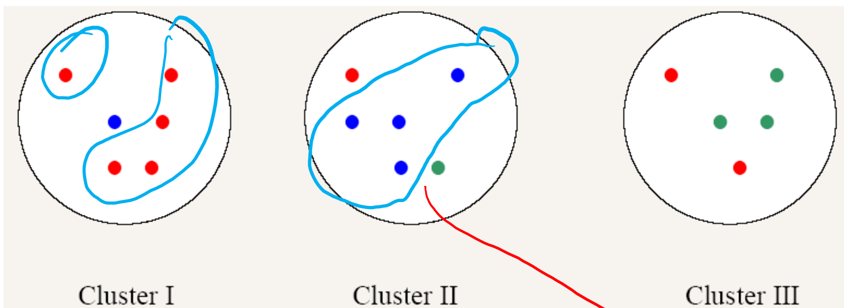
External Evaluation of Cluster Quality, e.g. using purity

- Simple measure: purity, the ratio between the dominant class in the cluster and the size of cluster
  - Assume data samples with C gold standard classes/groups, while the clustering algorithms produce K clusters, $\omega_1, \omega_2, \ldots, \omega_K$ with n_i members.

$$Purity(w_i) = \frac{1}{n_i} \max_j (n_{ij}) \quad j \in C$$

3 classes

  - Example



Cluster I: Purity = 1/6 (max(5, 1, 0)) = 5/6

Cluster II: Purity = 1/6 (max(1, 4, 1)) = 4/6

Cluster III: Purity = 1/5 (max(2, 0, 3)) = 3/5

K-means Clustering

Data → Task → Representation → Score Function → Search/Optimization → Models, Parameters

Tabular → Clustering → Mixture of spherical shapes → Sum-of-square distance to centroid → K-means algorithm → Cluster membership & centroid

$i = 1, \ldots, n$
$j = 1, \ldots, k$

$\{m_j\}$

$\{C_j\}$

ERP

Thank You

Thank you

More See L25 Extra GMM slides

EXTRA

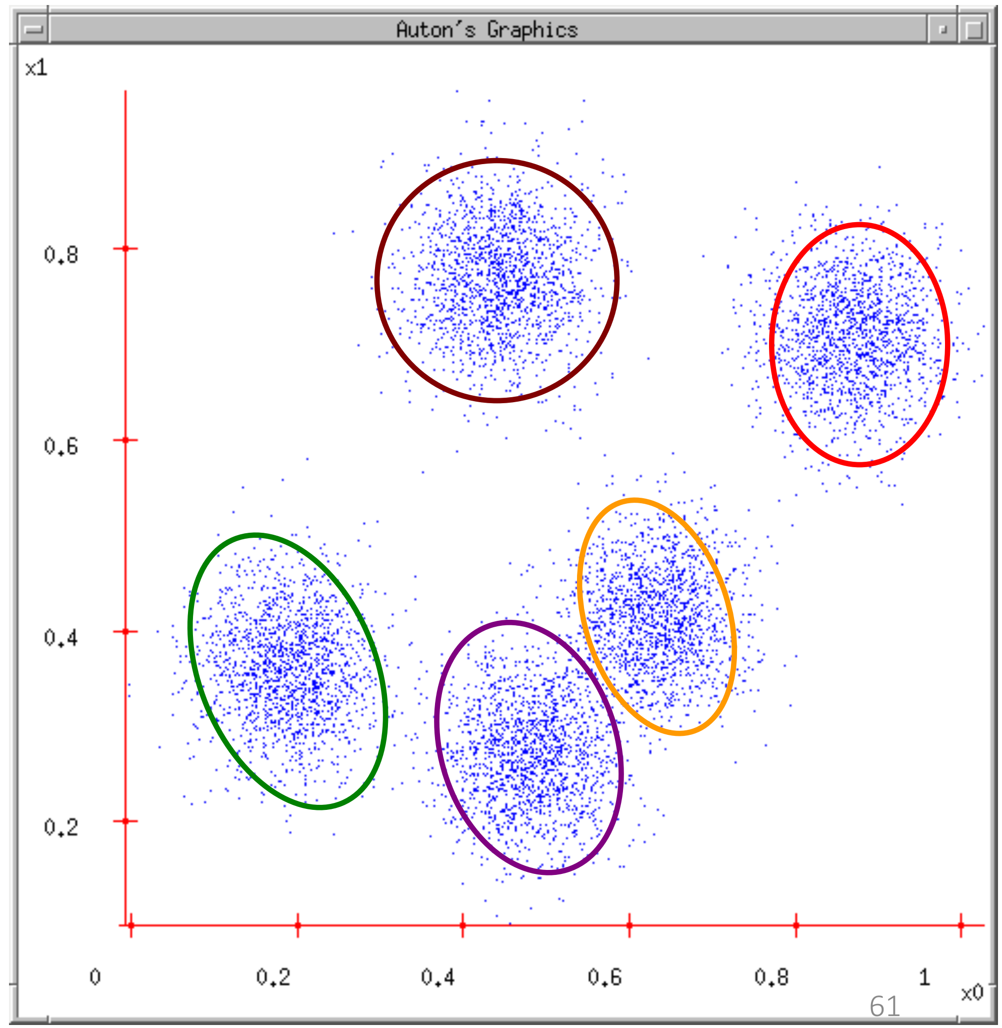
More Partitional : Gaussian Mixture Model

- 1. Review of Gaussian Distribution
- 2. GMM for clustering : basic algorithm
- 3. GMM connecting to K-means
- 4. Problems of GMM and K-means

A Gaussian Mixture Model for Clustering

- Assume that data are generated from a mixture of Gaussian distributions
- For each Gaussian distribution
  - Center: μ_j $j \in \{1, \cdots, k\}$
  - covariance: Σ_j
- For each data point
  - Determine membership

z_{ij} : if x_i belongs to j-th cluster



Expectation-Maximization for training GMM

- Start:
  - "Guess" the centroid and covariance for each of the K clusters
  - "Guess" the proportion of clusters, e.g., uniform prob 1/K

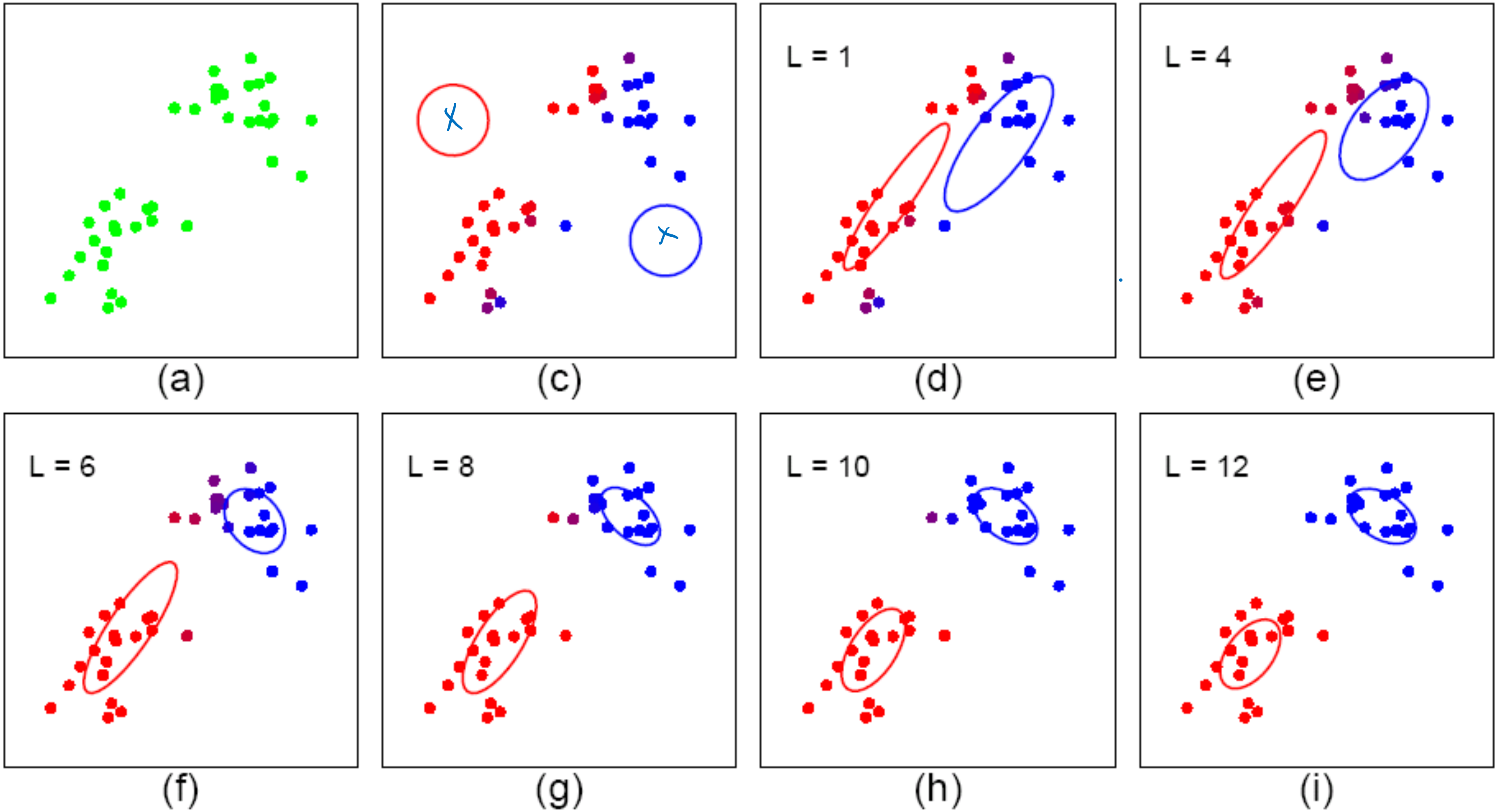
$$M_{ij} = \begin{cases} 1 & \text{if } X_i \text{ belong } C_j \\ 0 & \text{otherwise} \end{cases}$$

Hard

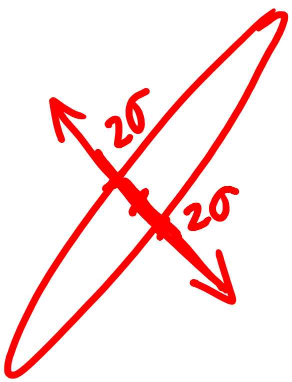
- Loop
  - For each point, revising its proportions belonging to each of the K clusters

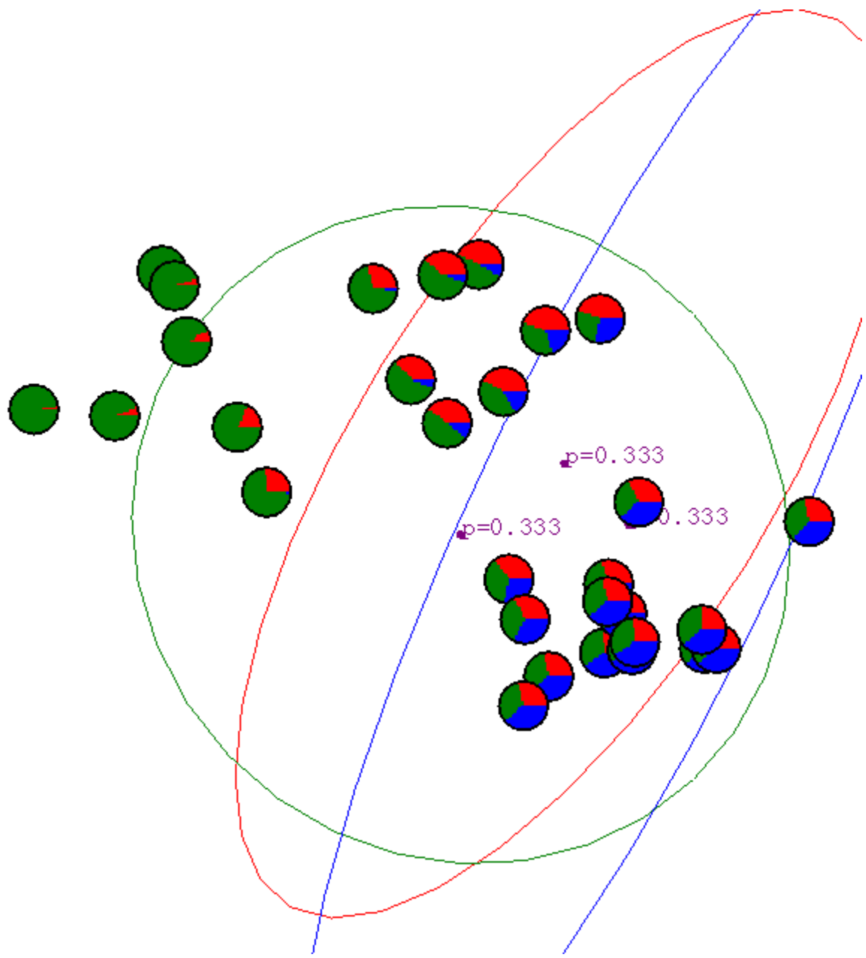
Soft

  - For each cluster, revising both the mean (centroid position) and covariance (shape) each cluster, revising both the mean (centroid position) and covariance (shape)



Another
Gaussian Mixture Example: Start

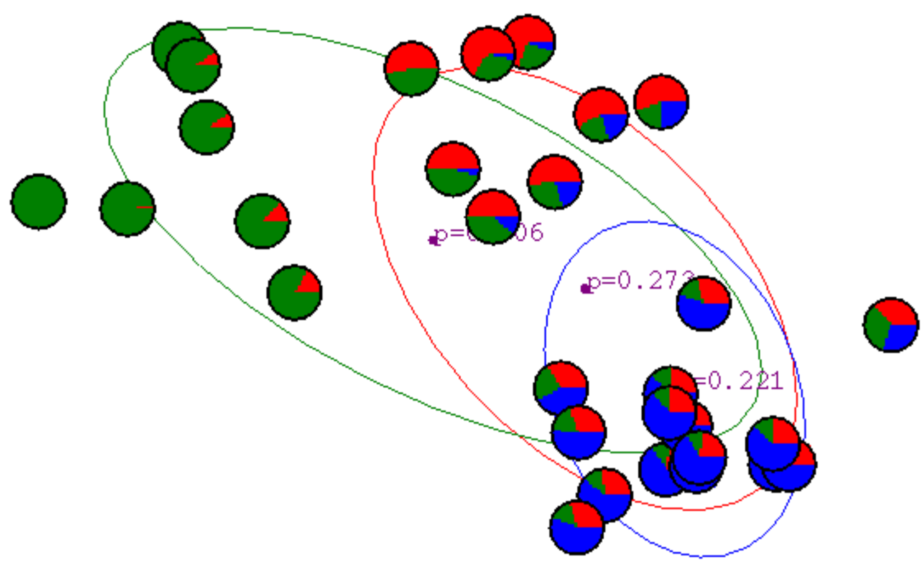




$p(\mu_j | x_i)$

Another GMM Example: After First Iteration

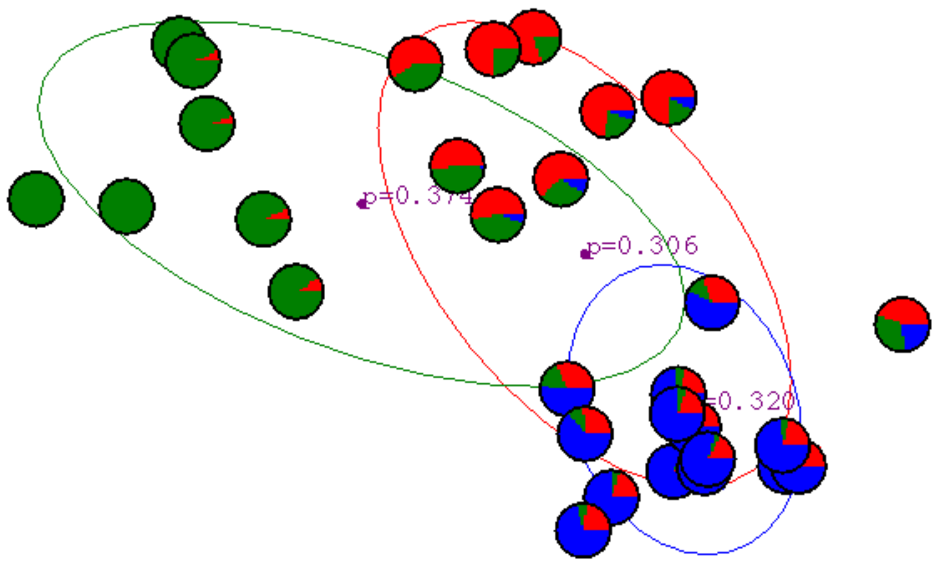
For each point, revising its proportions belonging to each of the K clusters



For each cluster, revising its mean (centroid position), covariance (shape) and proportion in the mixture

Another GMM Example: After 2nd Iteration

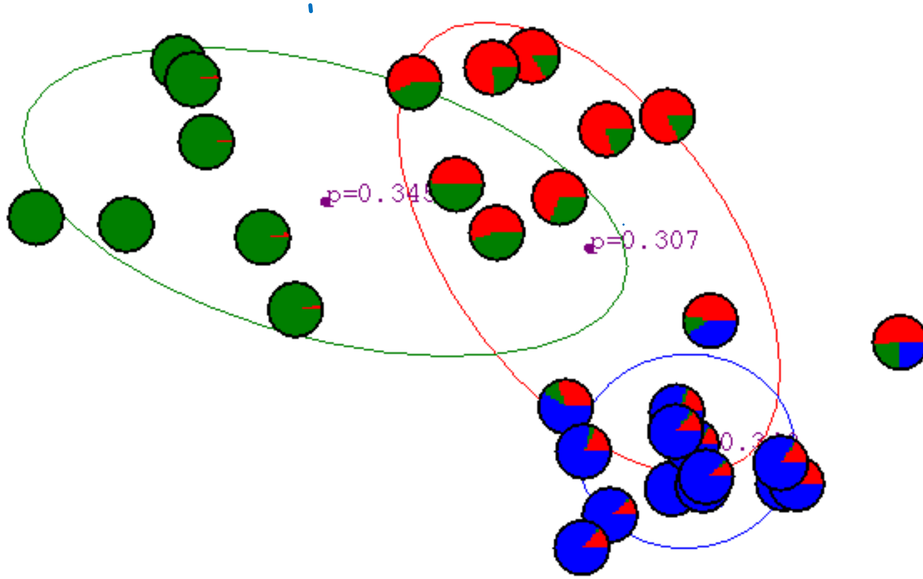
For each point, revising its proportions belonging to each of the K clusters



For each cluster, revising its mean (centroid position), covariance (shape) and proportion in the mixture

After 3rd Iteration

For each point, revising its proportions belonging to each of the K clusters

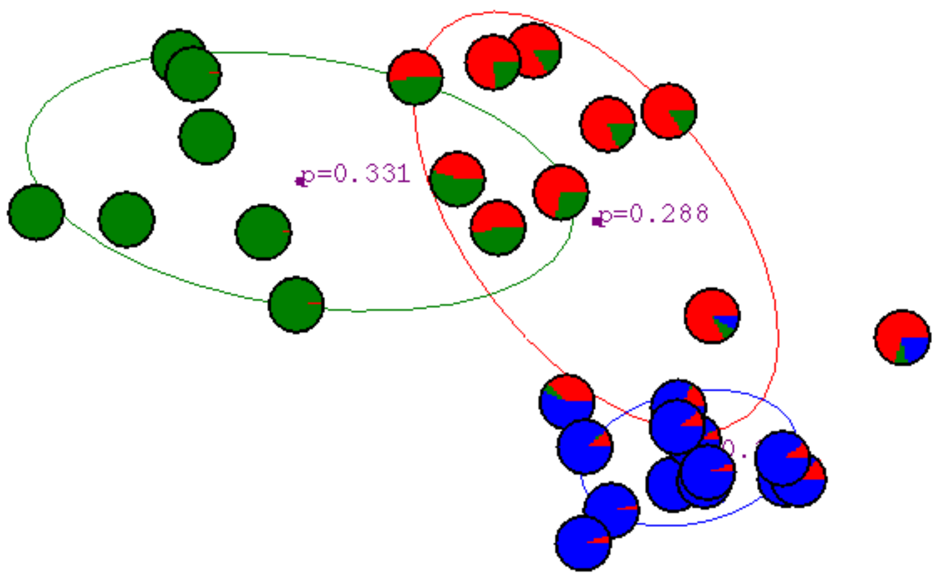


$p(\mu_j)$

For each cluster, revising its mean (centroid position), covariance (shape) and proportion in the mixture

After 4th Iteration

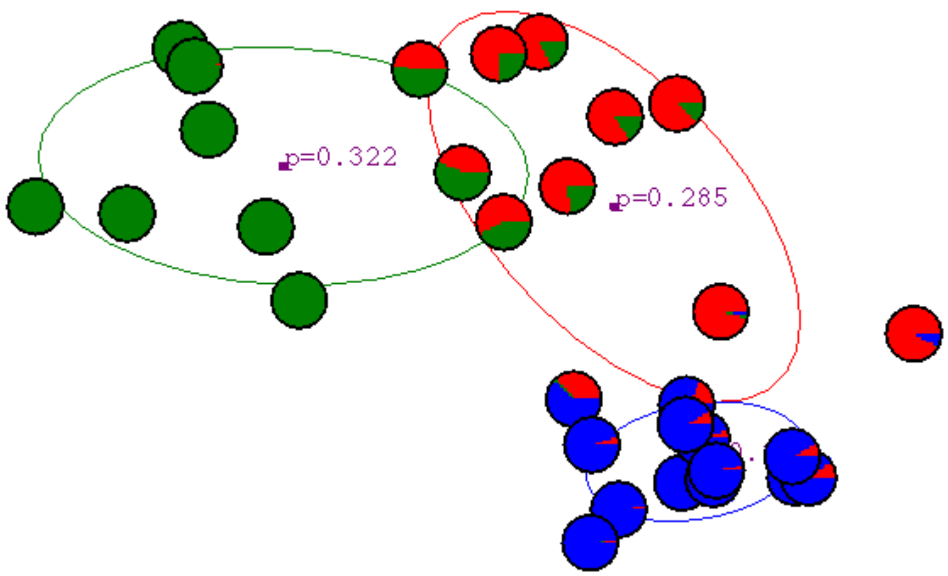
For each point, revising its proportions belonging to each of the K clusters



For each cluster, revising its mean (centroid position), covariance (shape) and proportion in the mixture

After 5th Iteration

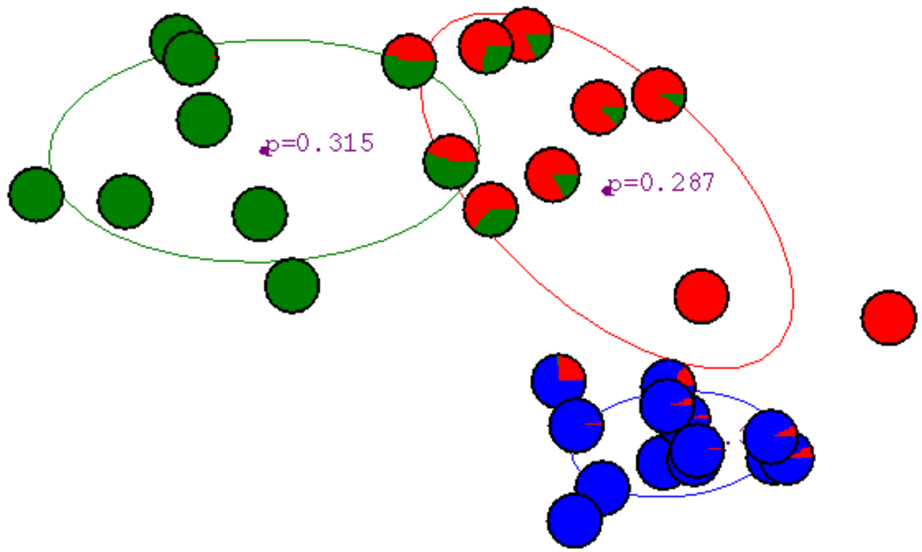
For each point, revising its proportions belonging to each of the K clusters



For each cluster, revising its mean (centroid position), covariance (shape) and proportion in the mixture

After 6th Iteration

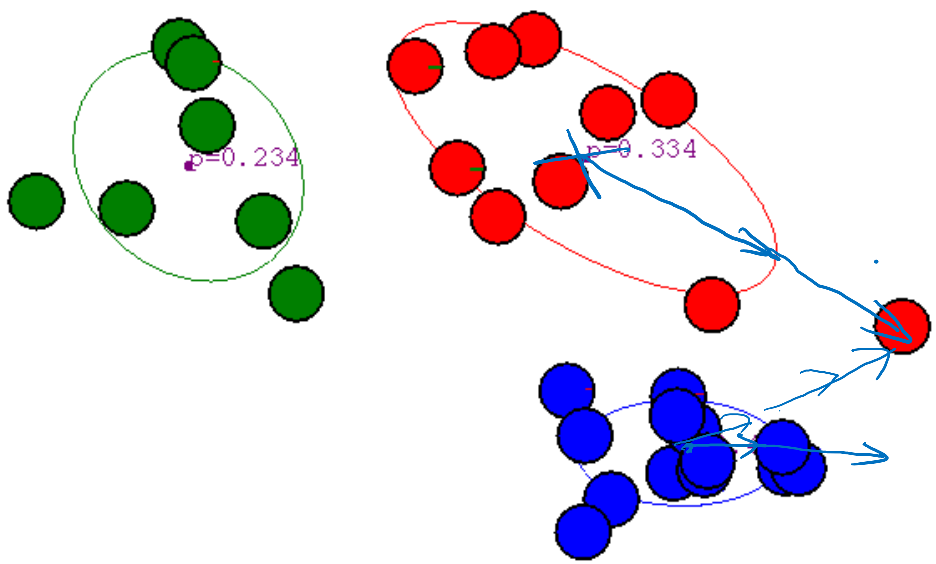
For each point, revising its proportions belonging to each of the K clusters



For each cluster, revising its mean (centroid position), covariance (shape) and proportion in the mixture

Another GMM Example: After 20th Iteration

For each point, revising its proportions belonging to each of the K clusters



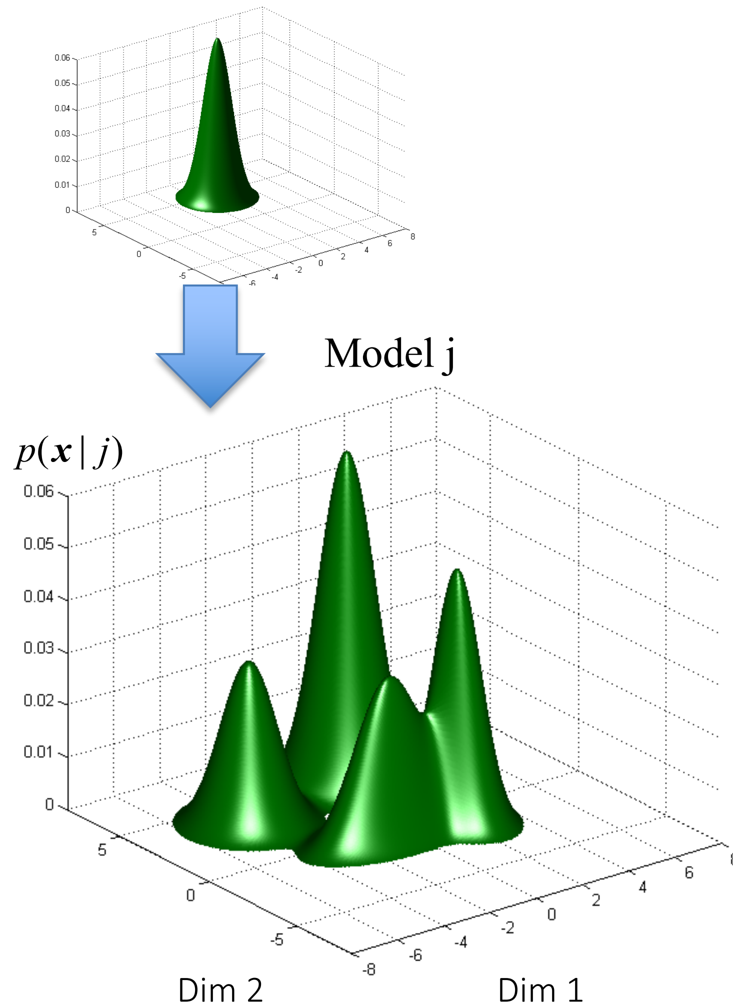
For each cluster, revising its mean (centroid position), covariance (shape) and proportion in the mixture

Application : GMMs for speaker recognition

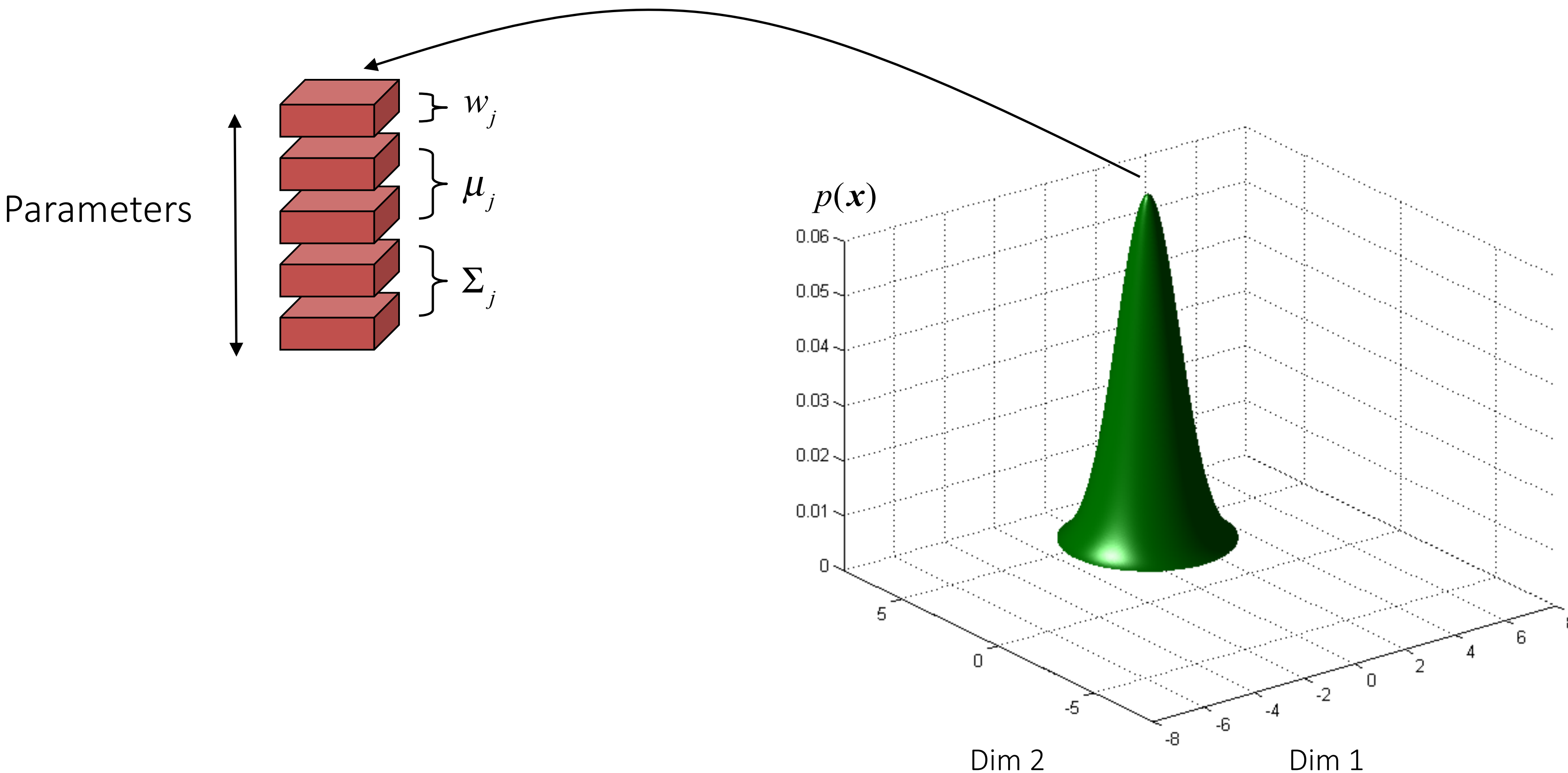
- A Gaussian mixture model (GMM) represents as the weighted sum of multiple Gaussian distributions

- Each Gaussian state i has a
  - Mean μ_j
  - Covariance Σ_j
  - Weight

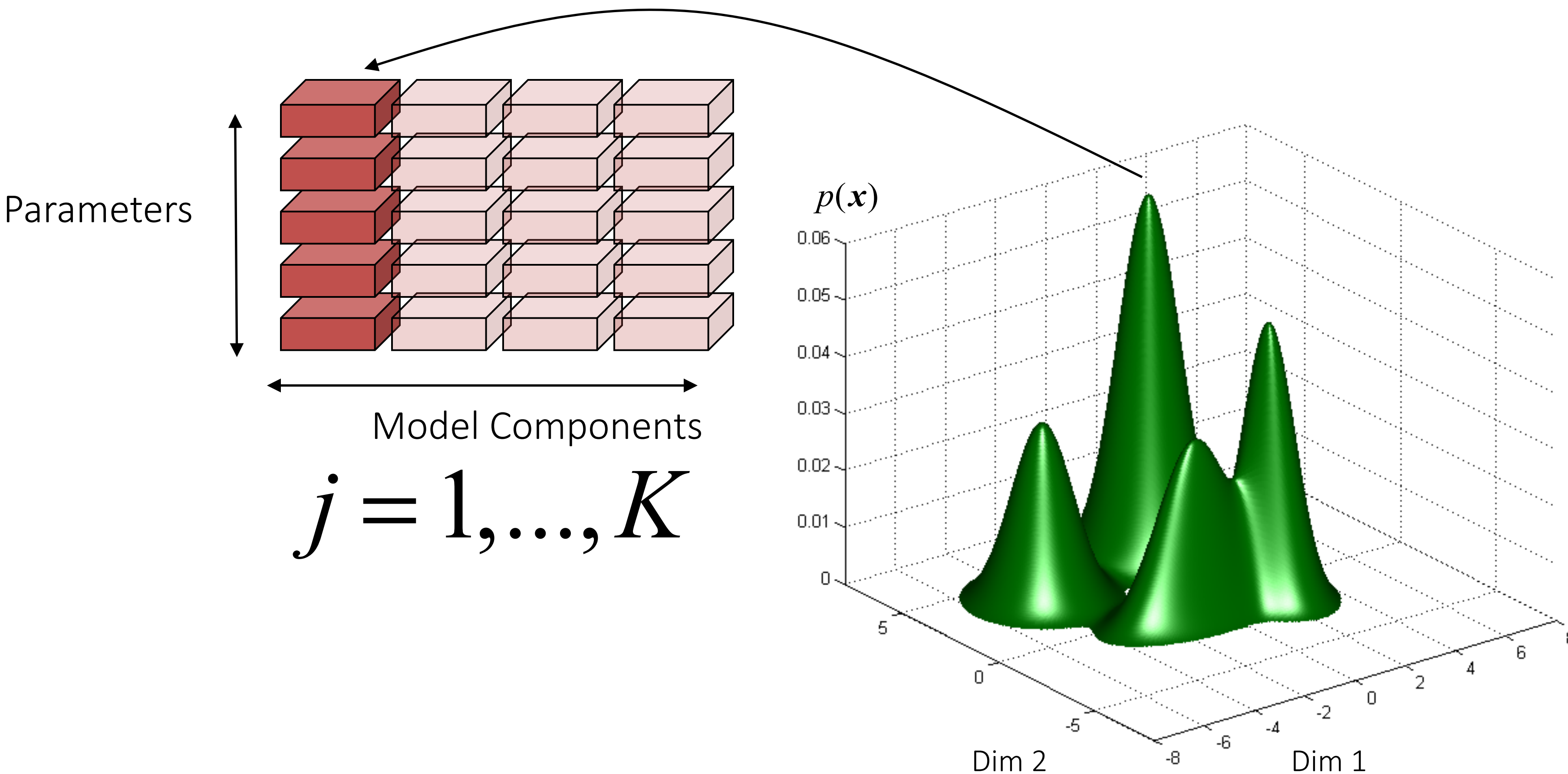
$$w_j \equiv p(\mu = \mu_j)$$



Recognition Systems
Gaussian Mixture Models



Recognition Systems
Gaussian Mixture Models



Recap: Expectation-Maximization for training GMM

More details See L19c GMM and L19d EM

- Start:
  - "Guess" the centroid and covariance for each of the K clusters
  - “Guess” the proportion of clusters, e.g., uniform prob 1/K

- Loop
  - For each point, revising its proportions belonging to each of the K clusters
  - For each cluster, revising both the mean (centroid position) and covariance (shape)

(3) GMM Clustering

Data
↓
Task
↓
Representation
↓
Score Function
↓
Search/Optimization
↓
Models, Parameters

Tabular
↓
Clustering
↓
Mixture of Gaussian
↓
Likelihood
↓
EM algorithm
↓
Each point's soft membership & mean / covariance per cluster

$$\sum_i \log \prod_{i=1}^{n} p(x = x_i) = \sum_i \log \left[\sum_{\mu_j} p(\mu = \mu_j) \frac{1}{(2\pi)\left|\Sigma_j\right|^{1/2}} e^{-\frac{1}{2}\left(\vec{x}-\vec{\mu}_j\right)^T \Sigma_j^{-1} \left(\vec{x}-\vec{\mu}_j\right)} \right]$$

References

- Hastie, Trevor, et al. *The elements of statistical learning*. Vol. 2. No. 1. New York: Springer, 2009.
- Big thanks to Prof. Eric Xing @ CMU for allowing me to reuse some of his slides
- Big thanks to Prof. Ziv Bar-Joseph @ CMU for allowing me to reuse some of his slides
- clustering slides from Prof. Rong Jin @ MSU