Cluster Analysis

Lecture 3

Outline

1. Clustering overview
   – Why
   – Distance measures
   – Types
2. K-Means (in-depth)
   – Derivation
   – Algorithm, convergence
   – Assumptions/limitations
   – Complexity/scaling
3. Agglomerative Hierarchical Clustering
4. DBSAN
5. Gaussian Mixture Models
6. Evaluation
   – Internal: partitional/hierarchical
   – External: classification/similarity

Clustering

Goal: group data into similar classes s.t.

- objects within a group are similar/related
  - Maximize intra-cluster similarity
- objects in different groups are different/unrelated
  - Minimize inter-cluster similarity

Why Cluster?

Understanding

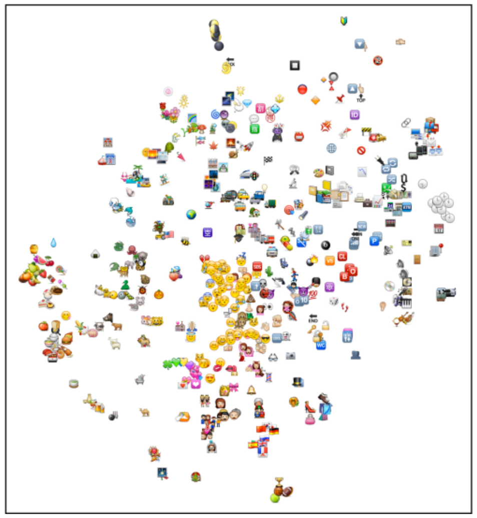
- Biological taxonomies
- Query understanding
  - Movie -> ratings, trailers…
- Diseases
  - Subtypes, progression
- Customer segmentation

Utility

- Summarize
  - Prototypes << N
- Compression
- NN acceleration

Similarity is Task-Specific

- Flags: map vs visual similarity
  - http://virostatiq.com/data/countries-by-flag-similarity/
- Emoji: category/search vs use
  - https://emojikeyboard.org
  - https://engineering.instagram.com/emojineering-part-1-machine-learning-for-emoji-trendsmachine-learning-for-emoji-trends-7f5f9cb979ad



Similarity vs Distance

Similarity

- No formal requirements/agree-upon definitions
- Generally: bigger=more similar
- Sometimes: normalized, inverse distance (e.g. 1-d_{norm})
- Proposal: https://doi.org/10.1016/j.tcs.2009.02.023

Distance

- D(A, B) = D(B, A)
  - Symmetric
- D(A, B) ≥ 0
  - Non-negative
- D(A, B) = 0 iff A=B
  - Positive
- D(A, B) ≤ D(A, B) + D(B, C)
  - Obeys Triangle Inequality

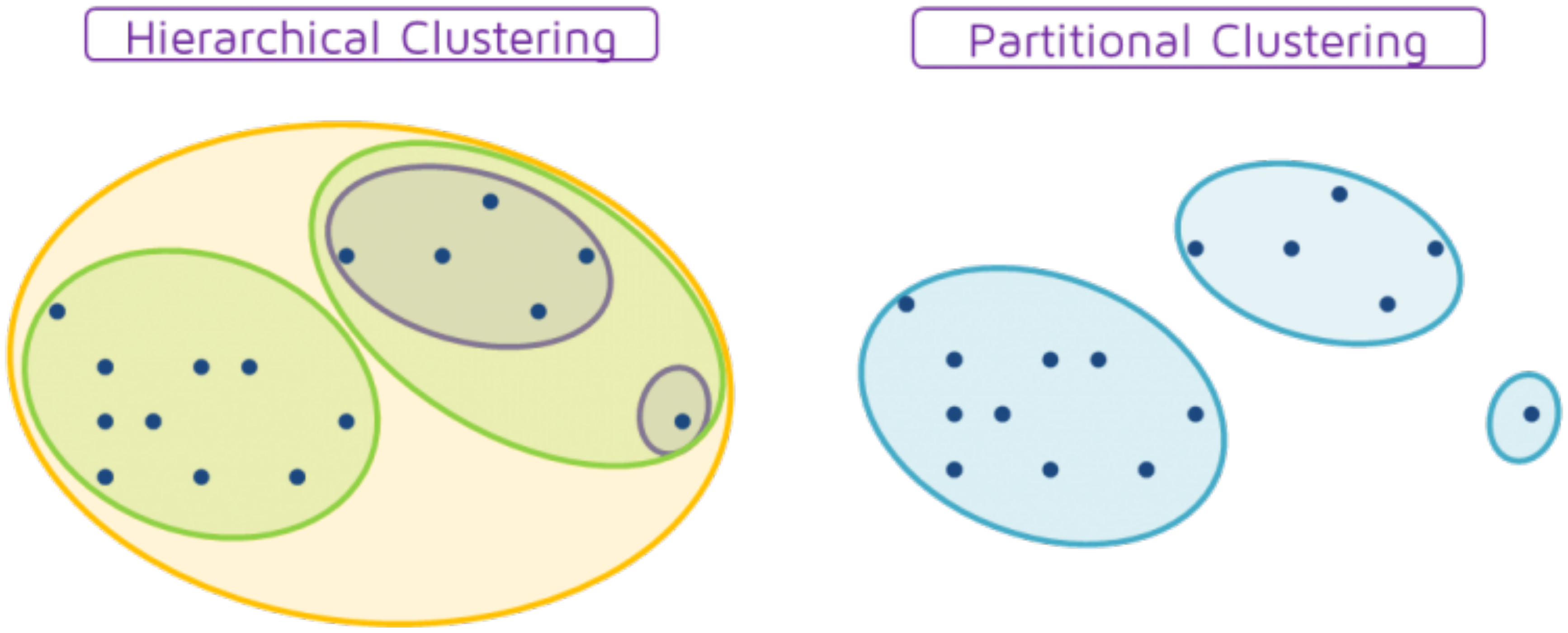
Common Distance Measures

Minkowski - 1=Manhattan - 2=Euclidean (usually Euclidean data)	$$(\sum_{i=1}^{n} \lvert x_i - y_i \rvert^p)^{\frac{1}{p}}$$
Cosine **Jaccard** (usually Documents)	$$\frac{A \cdot B}{\lvert\lvert A \rvert\rvert_2 \lvert\lvert B \rvert\rvert_2} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}}$$ $$\frac{\lvert A \cap B \rvert}{\lvert A \cup B \rvert}$$
Levenshtein (edit) **Hamming** (usually Strings)	# insert/remove/substitute operations # positions with different symbols

Clustering Characteristics

- Hierarchical/nested vs partitional
- Exclusive vs overlapping vs fuzzy
- Complete vs Partial

Hierarchical vs Partitional



Exclusive vs Overlapping vs Fuzzy

- Exclusive
  - An object belongs to one cluster
  - One-hot: [0,0,1,0,0]
- Overlapping
  - An object can belong to more than one cluster
  - Binary membership: [1,0,1,0,0]
- Fuzzy
  - An object has a membership of [0,1] with each cluster (typically sum to 1)
  - Proportional membership: [0.8, 0.0, 0.1, 0.1, 0.0]

Complete vs Partial

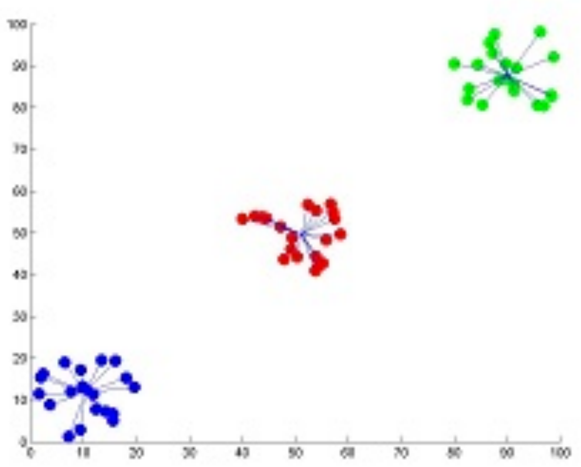
- Complete
  - All objects are assigned to (at least) one cluster
- Partial
  - Objects may not be assigned to any clusters
  - Examples: noise, outliers

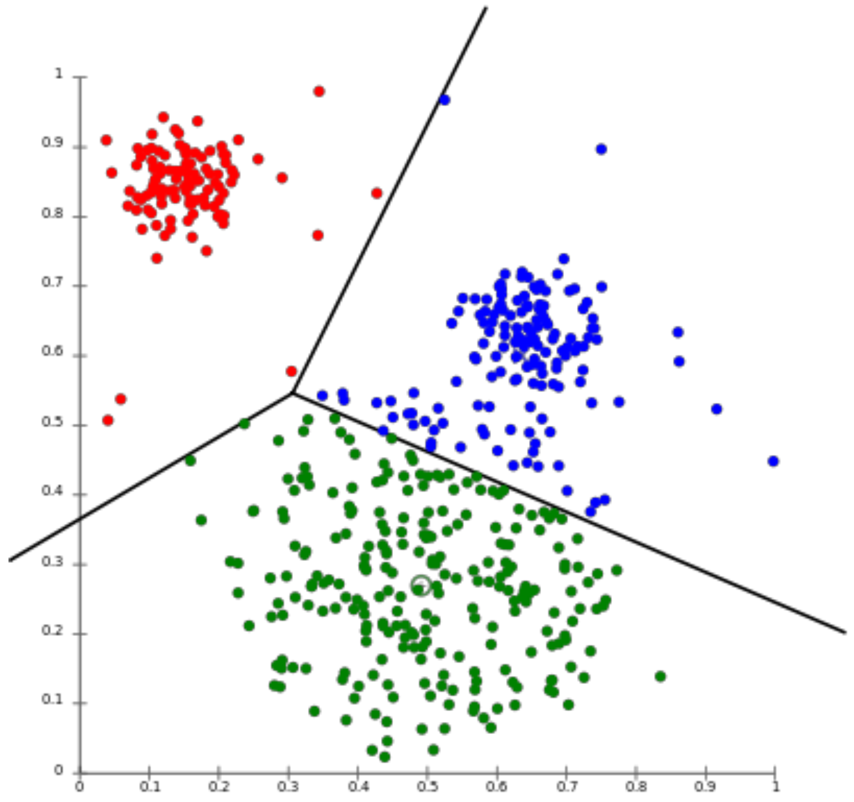
Clustering Algorithm Types

- Centroid/prototype-based
- Hierarchical/connectivity-based
- Density-based
- Distribution-based

Centroid/Prototype

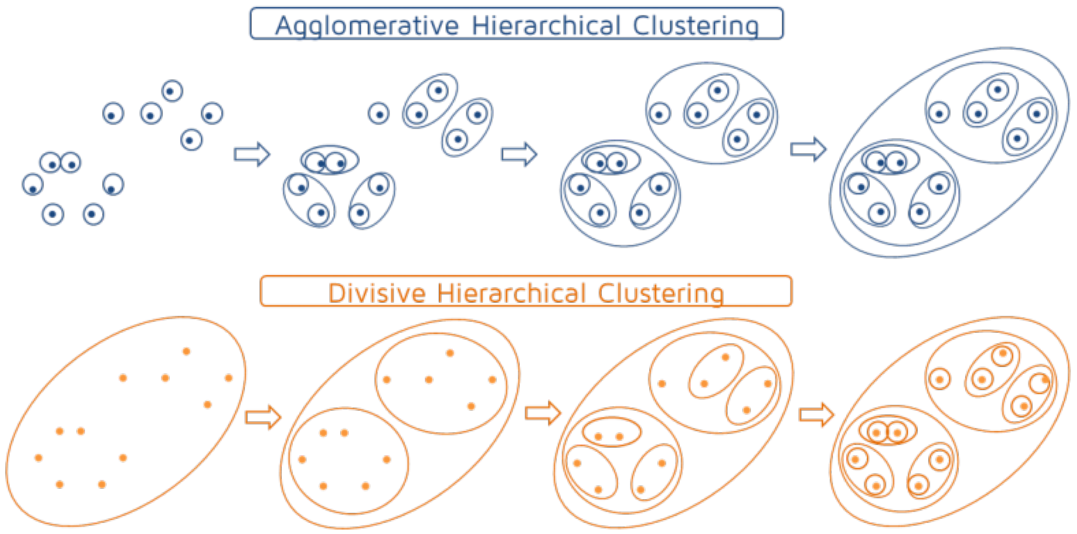
e.g. K-means



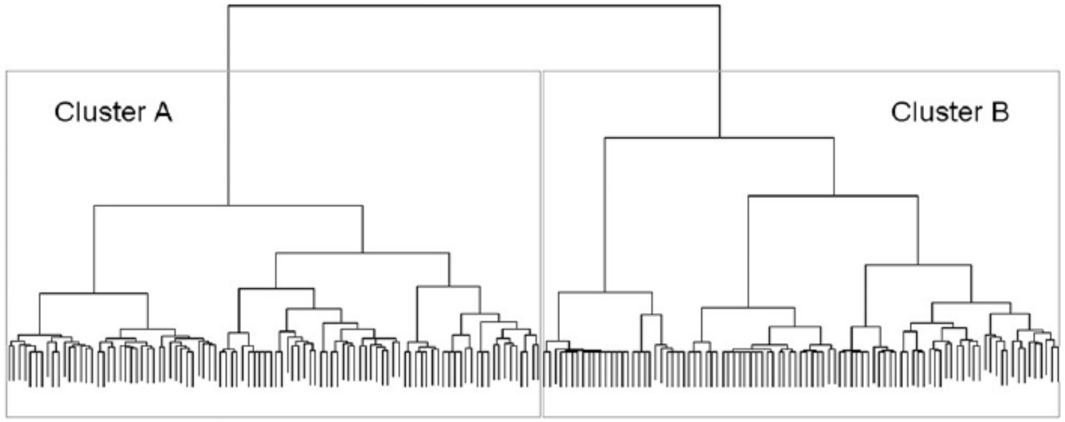


Induced Voronoi cells

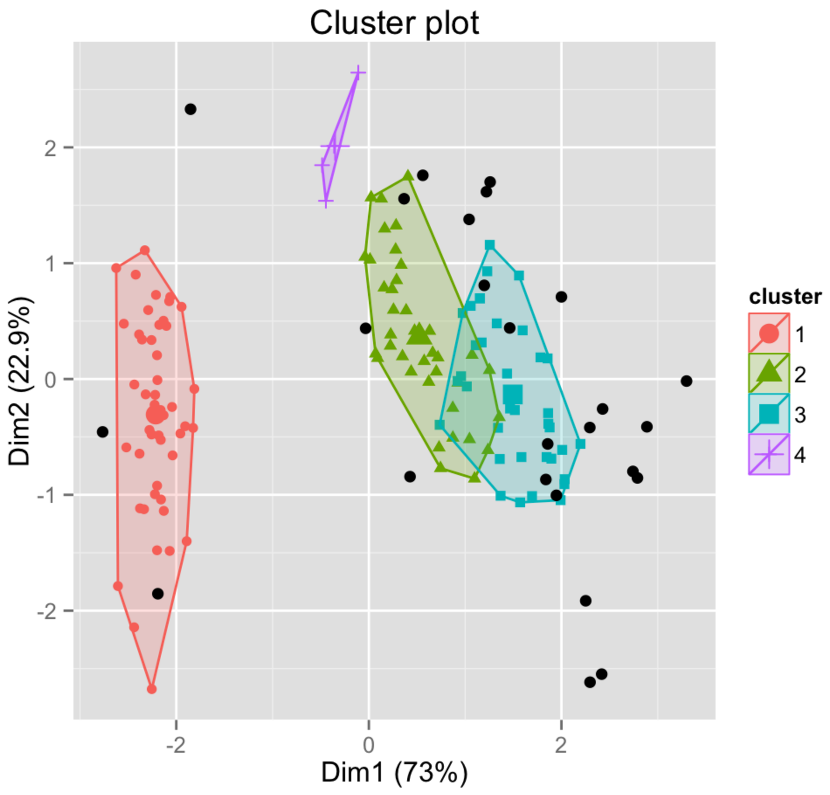
Hierarchical/Connectivity

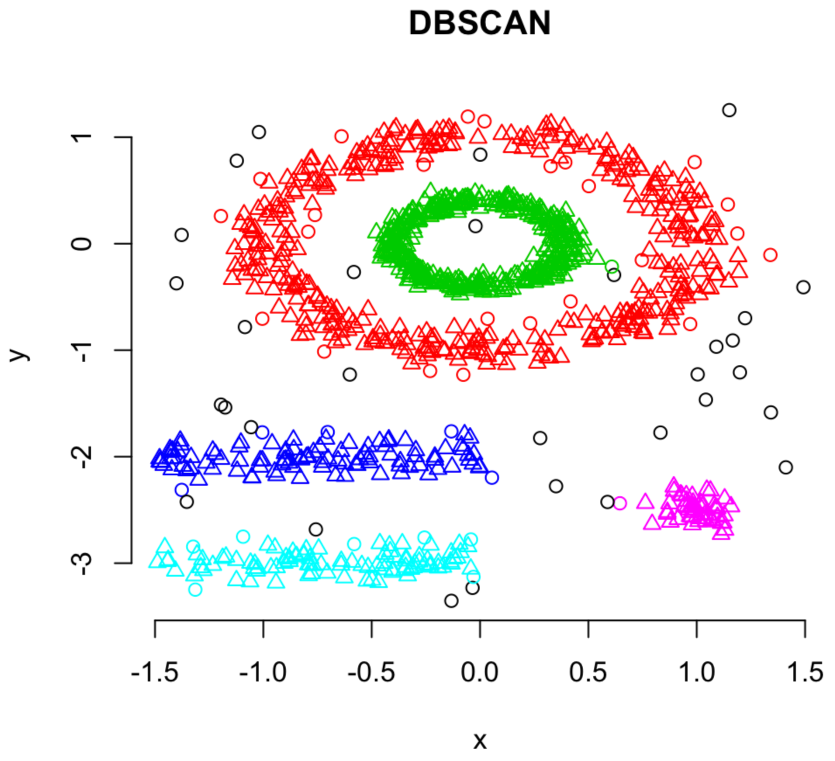

Agglomerative Hierarchical Clustering
Divisive Hierarchical Clustering

Dendogram cut off at some depth



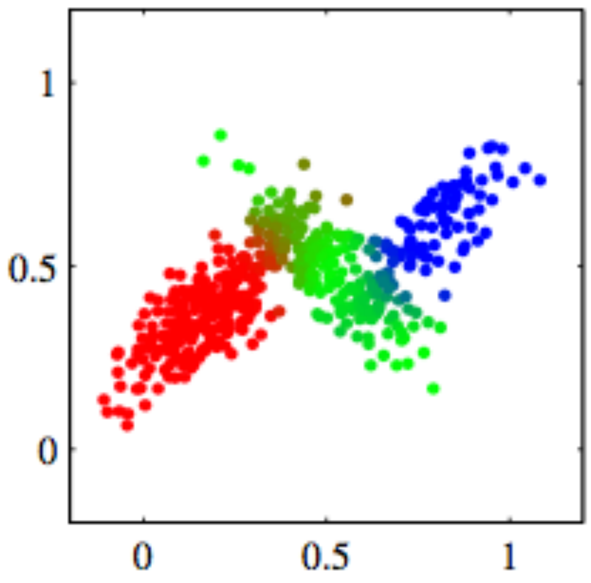
Density
e.g. DBSCAN

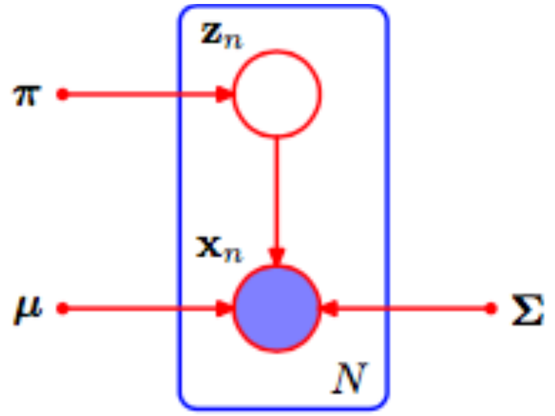


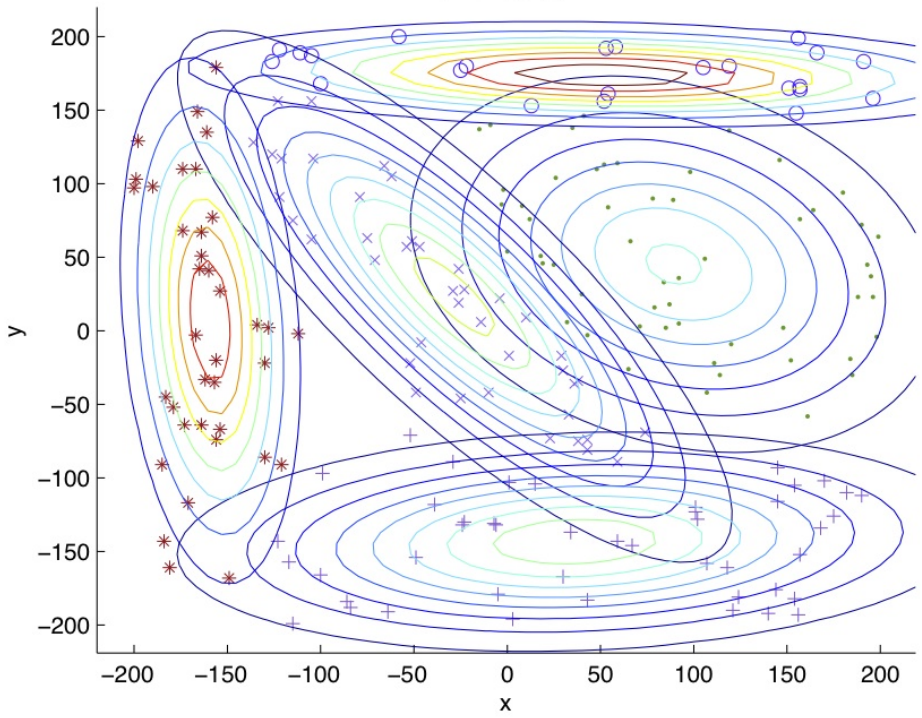


Distribution

e.g. Gaussian Mixture Models





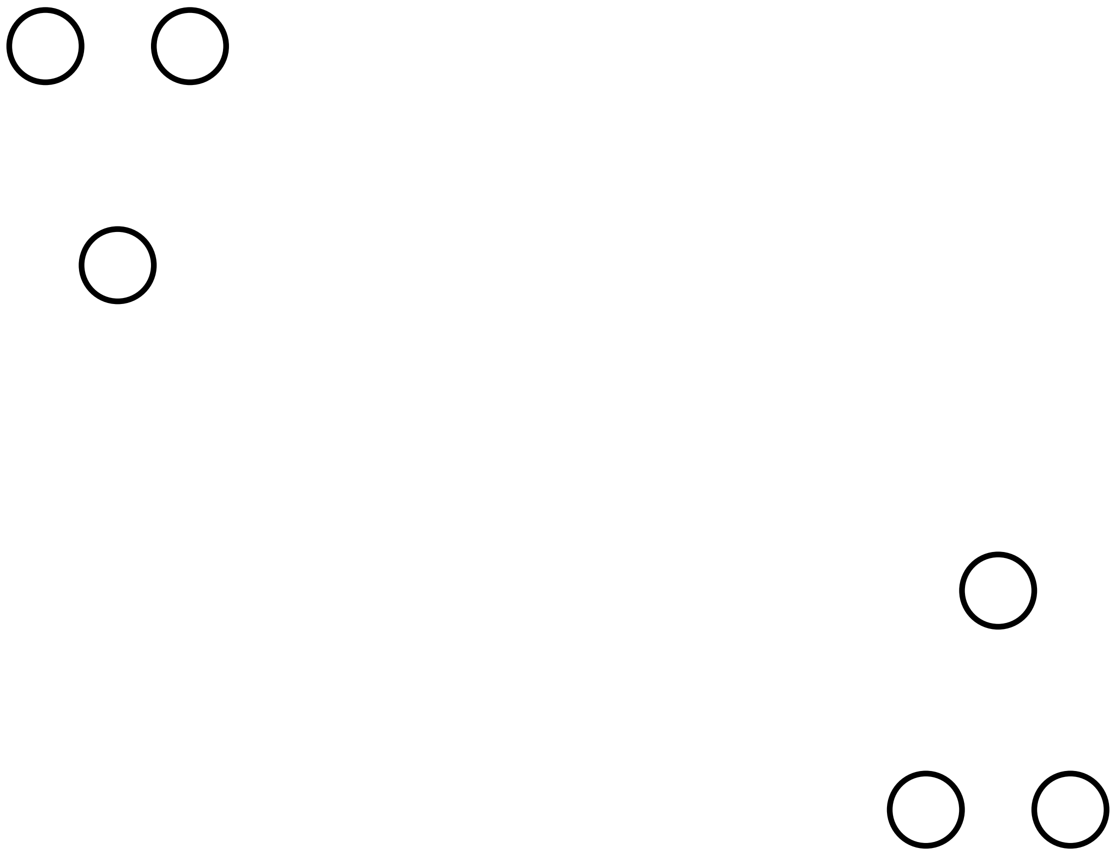


The K-Means Problem

- Given a dataset and a fixed parameter *K*…
- associate each data point with one of *K* clusters …
- such that the sum of the squares of the distances from each data point to its cluster's mean is minimized

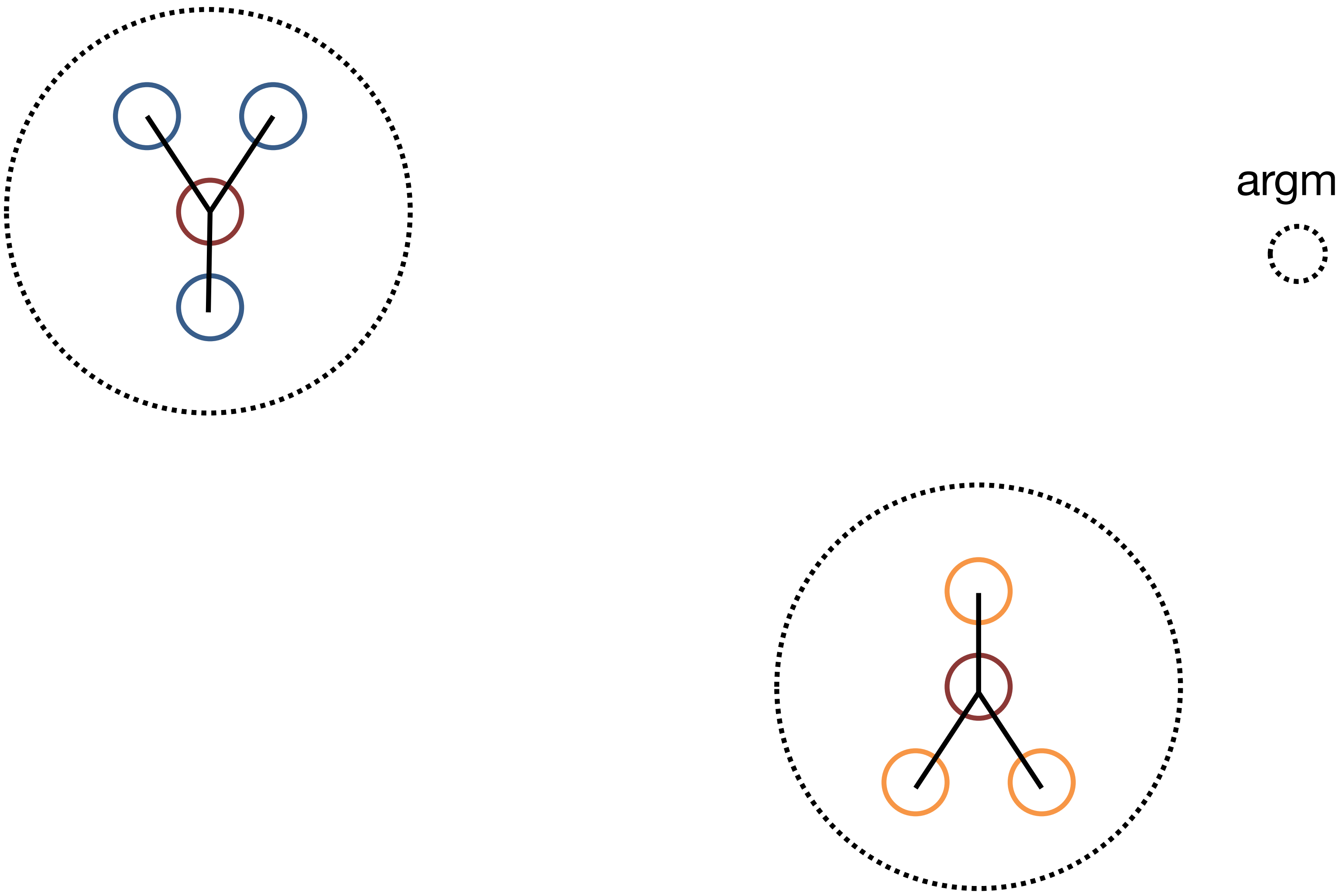
The K-Means Problem Visually

K=2



The K-Means Problem Visually

K=2



Quick Check

- Hierarchical or Partitional?
- Exclusive, Overlapping, Fuzzy?
- Complete or Partial?
- Centroid, Hierarchical, Density, Distribution?

Quick Check

- Hierarchical or **Partitional**?
- **Exclusive**, Overlapping, Fuzzy?
- **Complete** or Partial?
- **Centroid**, Hierarchical, Density, Distribution?

More Formally…

- $\{\mathbf{x_n}\}$: input data points, for n in 1…N
- $\{\boldsymbol{\mu}_\mathbf{k}\}$: center of the kth cluster, for k in 1…K
- $\{r_{nk}\}$: **binary indicator variable**
  - for each of {data point} x {cluster}
  - $r_{nk} \in \{0,1\}$
  - **One-Hot**: if data point x_n is assigned to cluster k, then $r_{nk}=1$ and $r_{nj}=0$ for $j \neq k$

$$\arg\min_{\mathbf{r_{nk}},\mu_\mathbf{k}} \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} ||\mathbf{x_n} - \mu_\mathbf{k}||^2$$

Quick Check

- How many partitions could there be?
  - N data points
  - K clusters

Quick Check

- How many partitions could there be?
  – N data points
  – K clusters

- Data point 1 can be in cluster {1…K}
- Data point 2 can be in cluster {1…K}

…

- Data point N can be in cluster {1…K}

Independent partitions: K^N 😭 (so **heuristic**!)

Iterative Parameter Estimates

For the K-Means algorithm, we'll iteratively move towards a local minimum:

1. Initialize: choose $\boldsymbol{\mu}_k$ (more later)
2. Loop till convergence (no change in $\mathbf{r}_{nk}$)
   a. Hold $\boldsymbol{\mu}_k$ fixed, minimize w.r.t. $\mathbf{r}_{nk}$
   b. Hold $\mathbf{r}_{nk}$ fixed, minimize w.r.t. $\boldsymbol{\mu}_k$

Note: this is a special case of a more general **Expectation Maximization** (**EM**) algorithm for parameter estimation

So 3 Questions

1. How to optimize $\mathbf{r_{nk}}$ (E-step)
2. How to optimize $\boldsymbol{\mu}_\mathbf{k}$ (M-step)
3. Will it converge?

K-Means: E-Step ($\mathbf{r_{nk}}$)

$$\arg\min_{\mathbf{r_{nk}},\mu_\mathbf{k}} \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} ||\mathbf{x_n} - \mu_\mathbf{k}||^2$$

- Observations
  - The objective is a linear sum of $\mathbf{r_{nk}}$
  - Each term involving a value of n is independent (i.e. each data point independent)
  - Partial w.r.t. $\mathbf{r_{nk}}$ is proportional to the distance from the point to a cluster center

K-Means: E-Step ($\mathbf{r_{nk}}$)

$$\arg\min_{\mathbf{r_{nk}},\mu_{\mathbf{k}}} \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} ||\mathbf{x_n} - \mu_{\mathbf{k}}||^2$$

- So… for each data point, choose the closest cluster center

$$r_{nk} = \begin{cases} 1 & \text{if } k = \arg\min_j ||\mathbf{x_n} - \mu_{\mathbf{j}}||^2 \\ 0 & \text{else} \end{cases}$$

K-Means: M-Step ($\boldsymbol{\mu}_\mathbf{k}$)

$$\arg\min_{\mathbf{r_{nk}},\mu_\mathbf{k}} \sum_{n=1}^{N}\sum_{k=1}^{K} r_{nk}||\mathbf{x_n} - \mu_\mathbf{k}||^2$$

- Observations
  - Distance is a quadratic function of $\boldsymbol{\mu}_\mathbf{k}$
  - Each term involving a value of k is independent (i.e. each cluster is independent)
  - Partial w.r.t. $\boldsymbol{\mu}_\mathbf{k}$ … $\sum_{n=1}^{N} r_{nk}(\mathbf{x_n} - \mu_\mathbf{k})^2$

K-Means: M-Step (**μ$_k$**)

$$\arg\min_{\mathbf{r_{nk}},\mu_\mathbf{k}} \sum_{n=1}^{N}\sum_{k=1}^{K} r_{nk}||\mathbf{x_n}-\mu_\mathbf{k}||^2$$

- SO … $2\sum_{n=1}^{N} r_{nk}(\mathbf{x_n}-\mu_\mathbf{k})=0$
- Solve for **μ$_k$**: $\mu_\mathbf{k}=\frac{\sum_n r_{nk}\mathbf{x_n}}{\sum_n r_{nk}}$
  - Den=?

K-Means: M-Step ($\boldsymbol{\mu}_\mathbf{k}$)

$$\arg\min_{\mathbf{r_{nk}},\mu_\mathbf{k}} \sum_{n=1}^{N}\sum_{k=1}^{K} r_{nk}||\mathbf{x_n} - \mu_\mathbf{k}||^2$$

- SO … $2\sum_{n=1}^{N} r_{nk}(\mathbf{x_n} - \mu_\mathbf{k}) = 0$
- Solve for $\boldsymbol{\mu}_\mathbf{k}$: $\mu_\mathbf{k} = \frac{\sum_n r_{nk}\mathbf{x_n}}{\sum_n r_{nk}}$ ***Avg of points in the cluster***
  - Den=# points

Iterative Parameter Estimates

For the K-Means algorithm, we'll iteratively move towards a local minimum:

1. Initialize: choose $\mathbf{\mu_k}$ (more later)
2. Loop till convergence (no change in $\mathbf{r_{nk}}$)
   a. Points -> closest cluster
   b. Cluster -> avg of associated points

Will it ~~blend~~ converge???

Argument for Convergence

- Observations:
  - Finite clusterings (K^N)
  - Each clustering based *only* upon the last
  - Objective always decreases
    - E: each point changes *only* to a better cluster
    - M: mean minimizes total distance given current clustering
  - Deterministic movement
    - if new clustering is same as old, will never change
    - if new clustering is different, lower cost

- SO…
  - Converges to a ***local*** minimum
  - Must happen eventually, usually quickly

K-Means Algorithm

Algorithm 8.1 Basic K-means algorithm.

```
1: Select $K$ points as initial centroids.
2: repeat
3:     Form $K$ clusters by assigning each point to its closest centroid.
4:     Recompute the centroid of each cluster.
5: until Centroids do not change.
```

Pending Questions

- Initial centroids?
- Value of *K*?
- Assumptions/limitations?
- Complexity/scaling?

Good Clustering

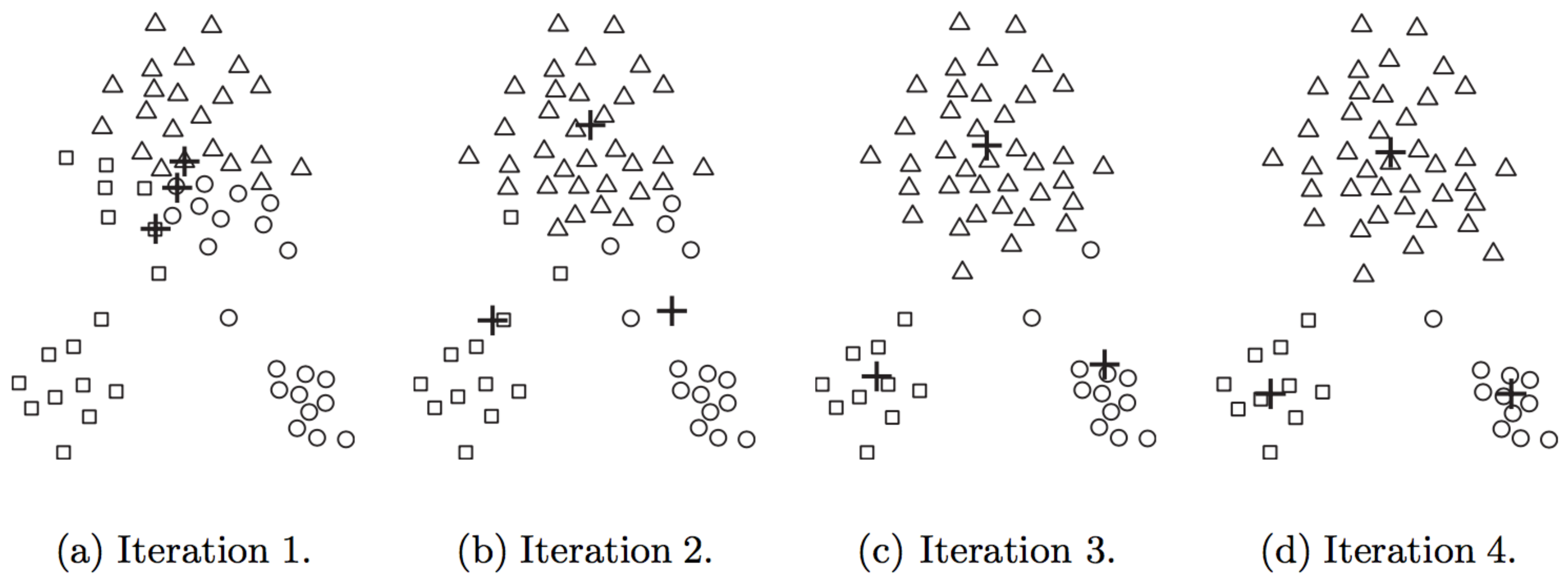


Figure 8.3. Using the K-means algorithm to find three clusters in sample data.

Not-So-Good Clustering

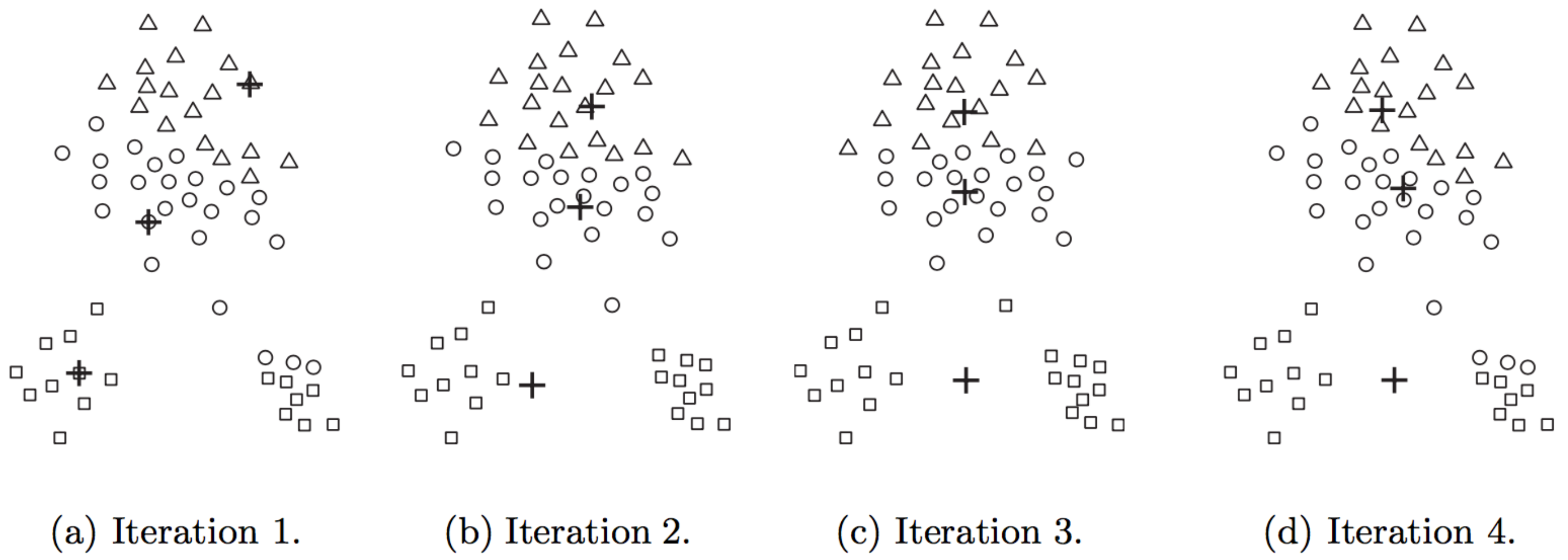


Figure 8.5. Poor starting centroids for K-means.

K-Means is Sensitive to Initialization

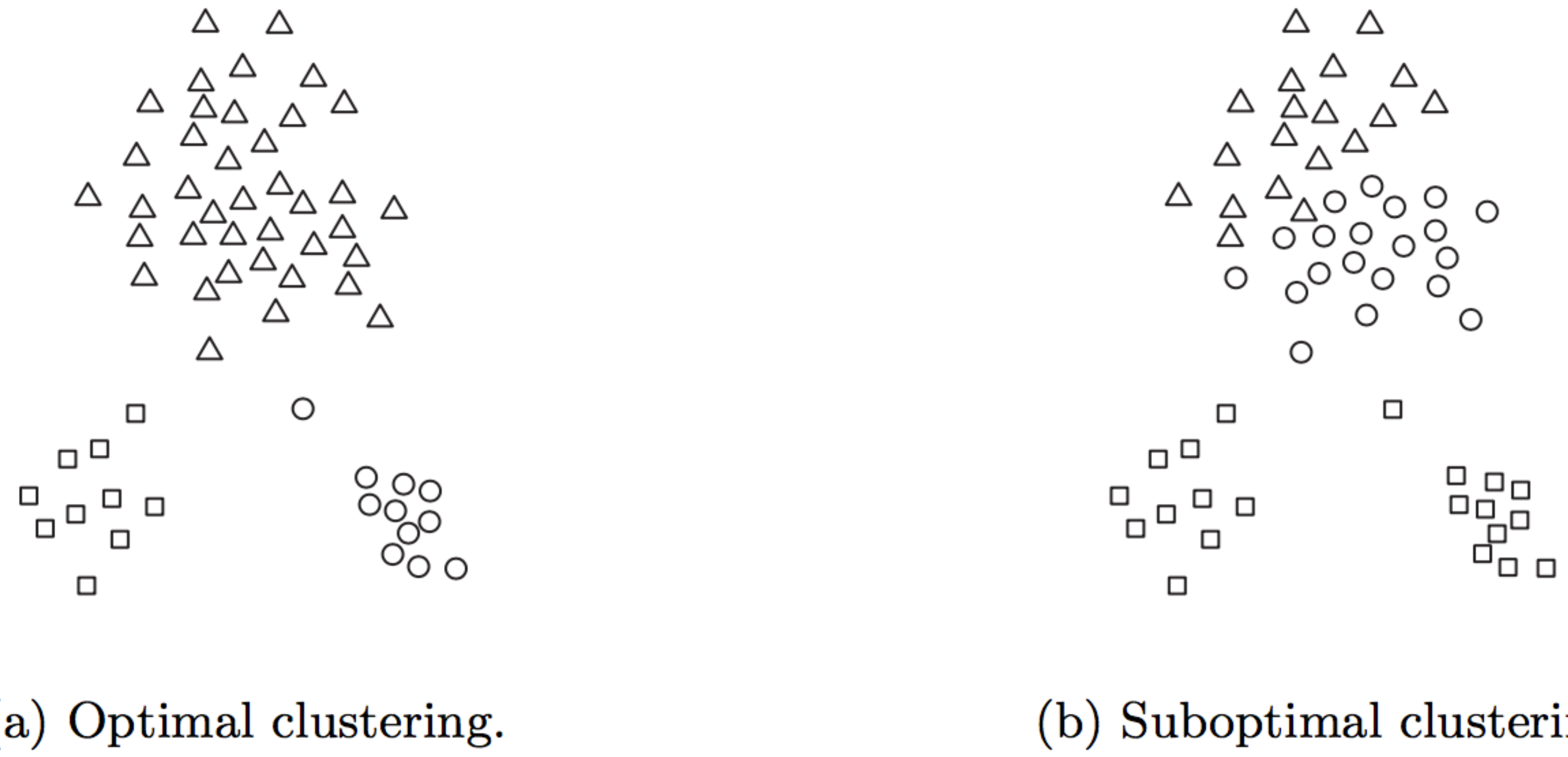


Figure 8.4. Three optimal and non-optimal clusters.

Common Approach

- Uniform random assignment
  - Could be data points (**Forgy**) or in $\mathbb{R}^d$
- Repeat k times and choose best SSE
- What could *possibly* go wrong!?

Good Clustering

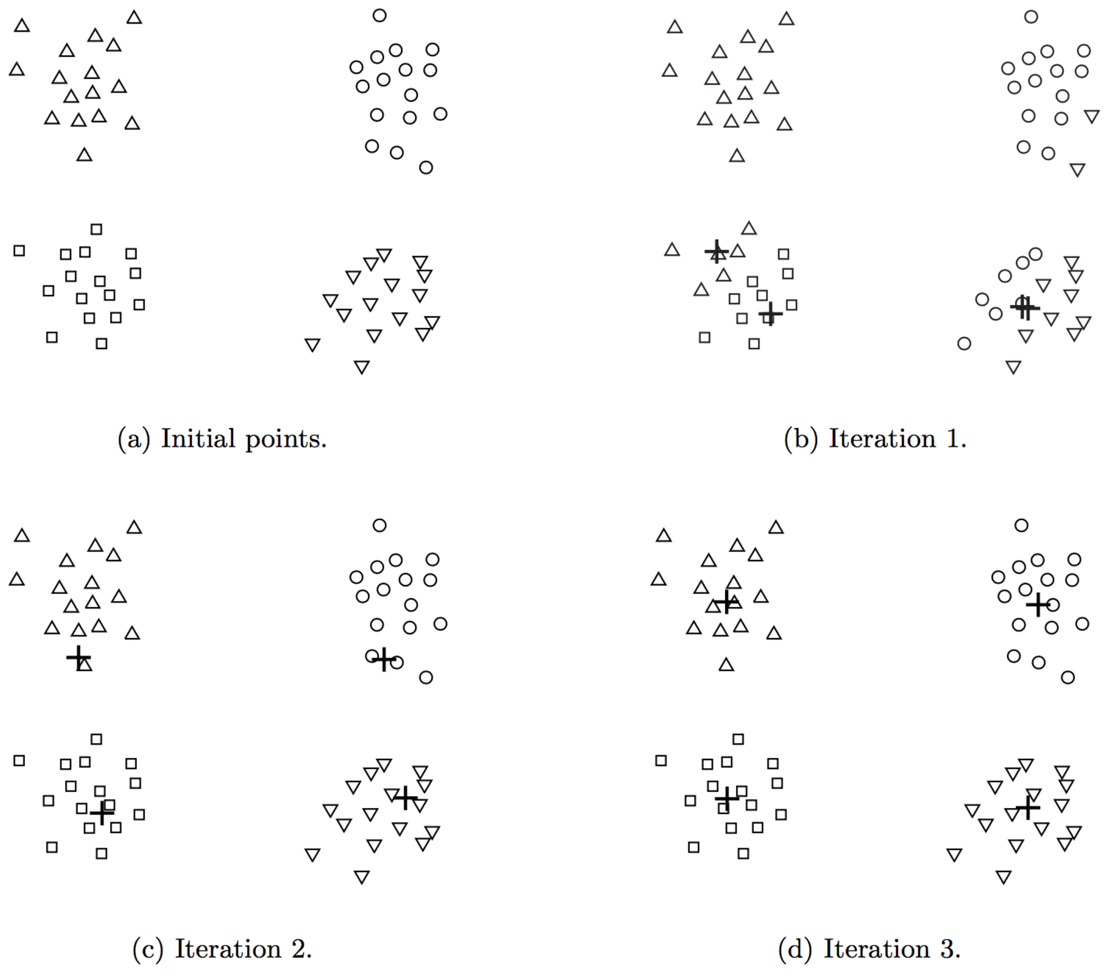


Figure 8.6. Two pairs of clusters with a pair of initial centroids within each pair of clusters.

Unequal Distribution w.r.t Clusters

Now Think Large k

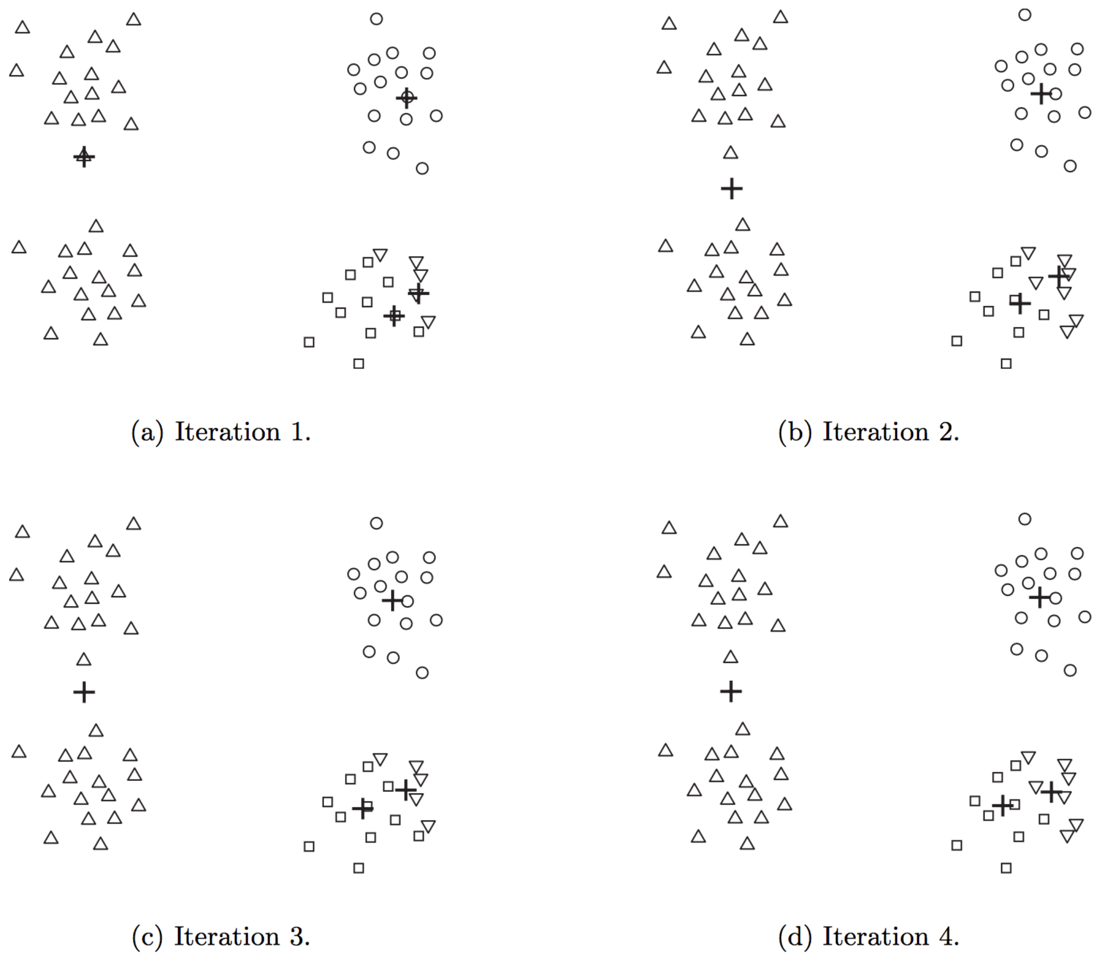


Figure 8.7. Two pairs of clusters with more or fewer than two initial centroids within a pair of clusters.

Initialization Approaches (1)

K-Means++

- Choose 1st at random from **x**
- For remaining, compute distance of each remaining point in **x** to closest centroid
- Select, weighting probabilistically towards farther
- Good: random, separated
- Bad: **expensive** (help: sampling and/or data structures)

Initialization Approaches (2)

Bisecting K-Means -> Initial Points

- Divisive hierarchical clustering, with K-Means local to each chosen sub-cluster
- Not locally minimal, so serves as initialization to global K-Means

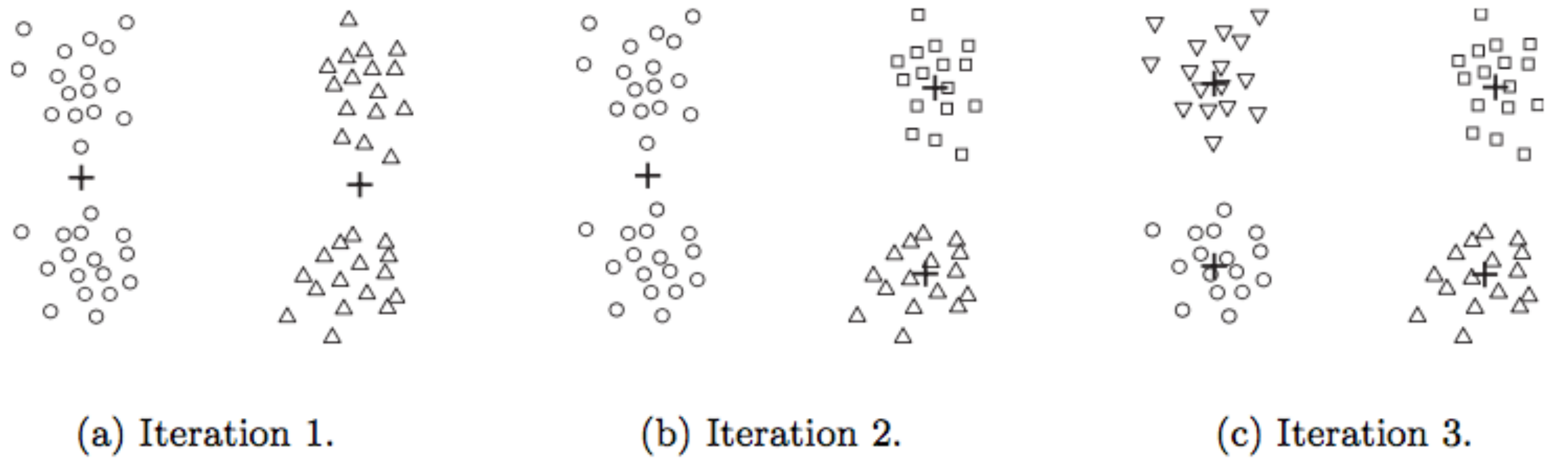
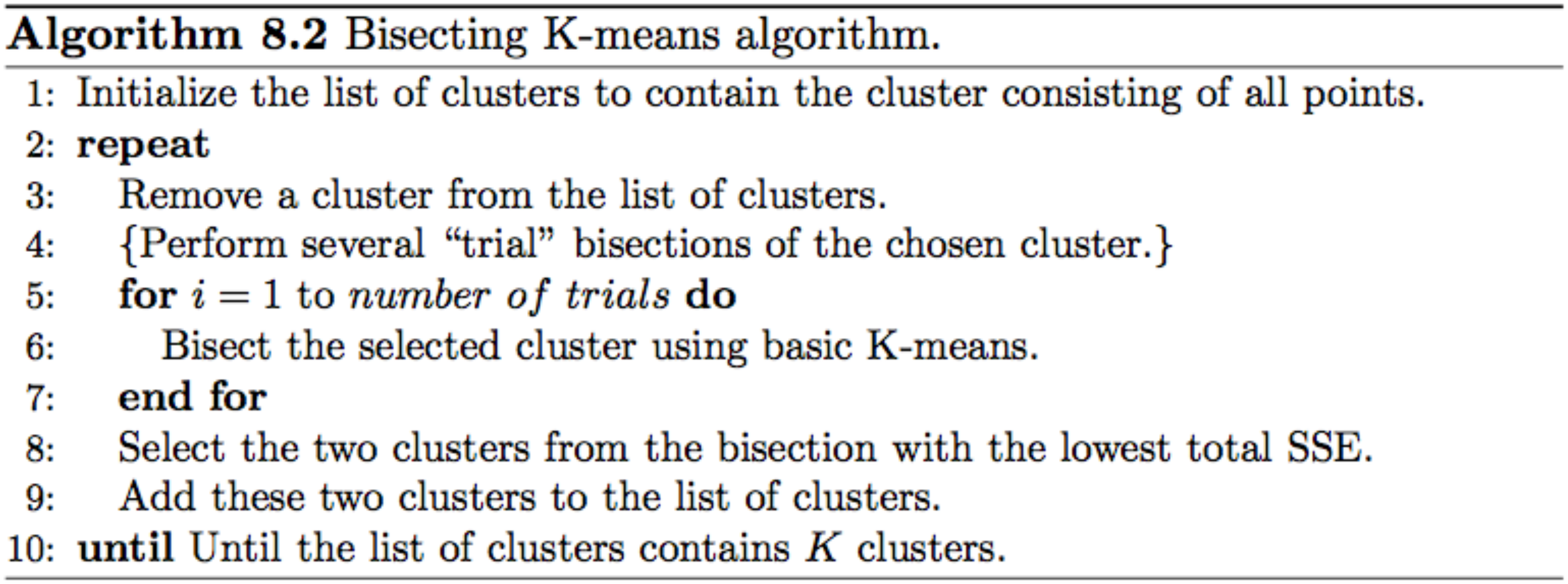


Figure 8.8. Bisecting K-means on the four clusters example.

Bisecting K-Means



Algorithm 8.2 Bisecting K-means algorithm.

```
1: Initialize the list of clusters to contain the cluster consisting of all points.
2: repeat
3:    Remove a cluster from the list of clusters.
4:    {Perform several "trial" bisections of the chosen cluster.}
5:    for $i = 1$ to number of trials do
6:       Bisect the selected cluster using basic K-means.
7:    end for
8:    Select the two clusters from the bisection with the lowest total SSE.
9:    Add these two clusters to the list of clusters.
10: until Until the list of clusters contains $K$ clusters.
```

Picking the Right Value of *K*

- Ideal: problem-specific context identifies a likely value
  - Post-processing may be required for fine-tuning
- But what if we aren't sure at the start as to a reasonable value of *K*?

Quick Check

- Describe how SSE changes as we increase the value of K?
  - What is the maximum value?

Quick Check

- Describe how SSE changes as we increase the value of K from 1 to N?

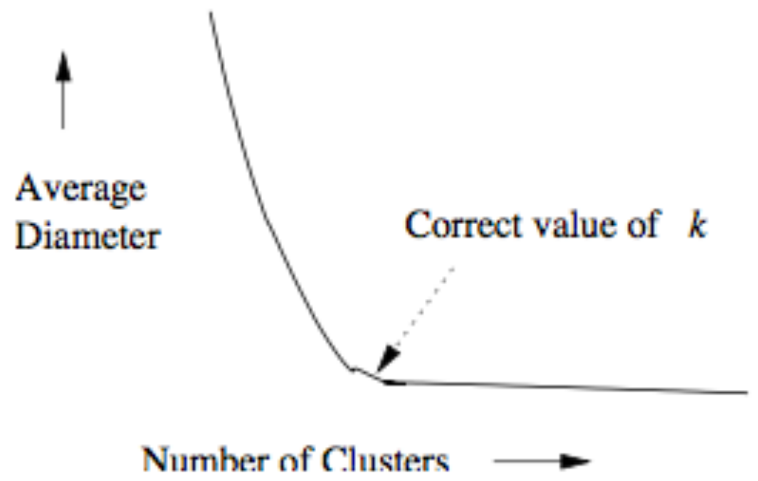


Figure 7.9: Average diameter or another measure of diffuseness rises quickly as soon as the number of clusters falls below the true number present in the data

The “Elbow” Method

- Identify a criterion w.r.t. SSE or variance
  - Harder than it sounds
- Binary parameter search to find range
  - 1, 2, 4, 8, 16, 32
- Binary search within to identify elbow
  - 24, 20, 22, 21

Others

- X-Means: add a **regularization** term to penalize large values of K, search!
  - Commonly Bayesian Information Criterion (BIC), others possible
- Information Theoretic: balance error with compression
- Internal cluster-quality evaluation criteria (e.g. Silhouette; more later)

Examples

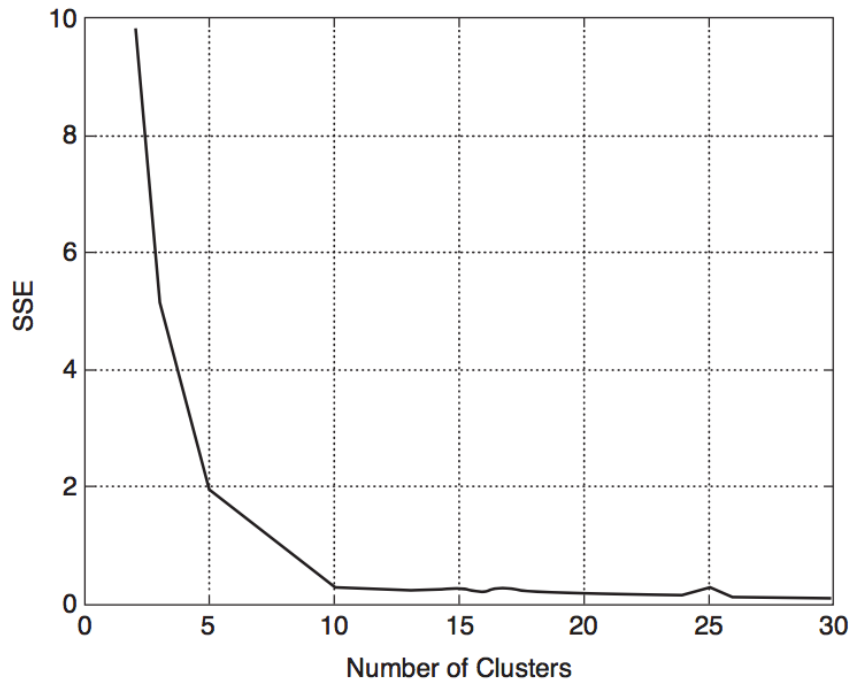


Figure 8.32. SSE versus number of clusters for the data of Figure 8.29.

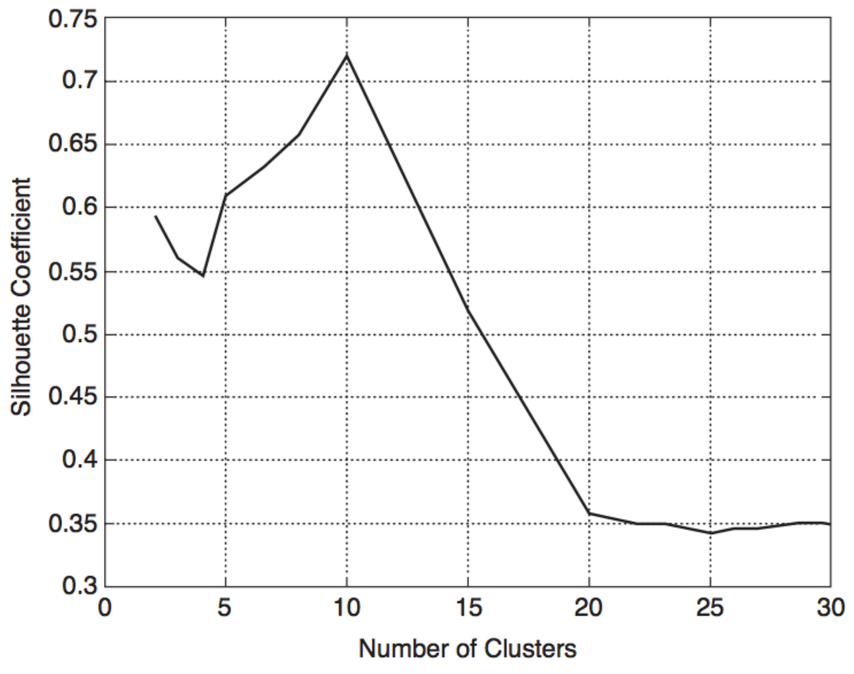


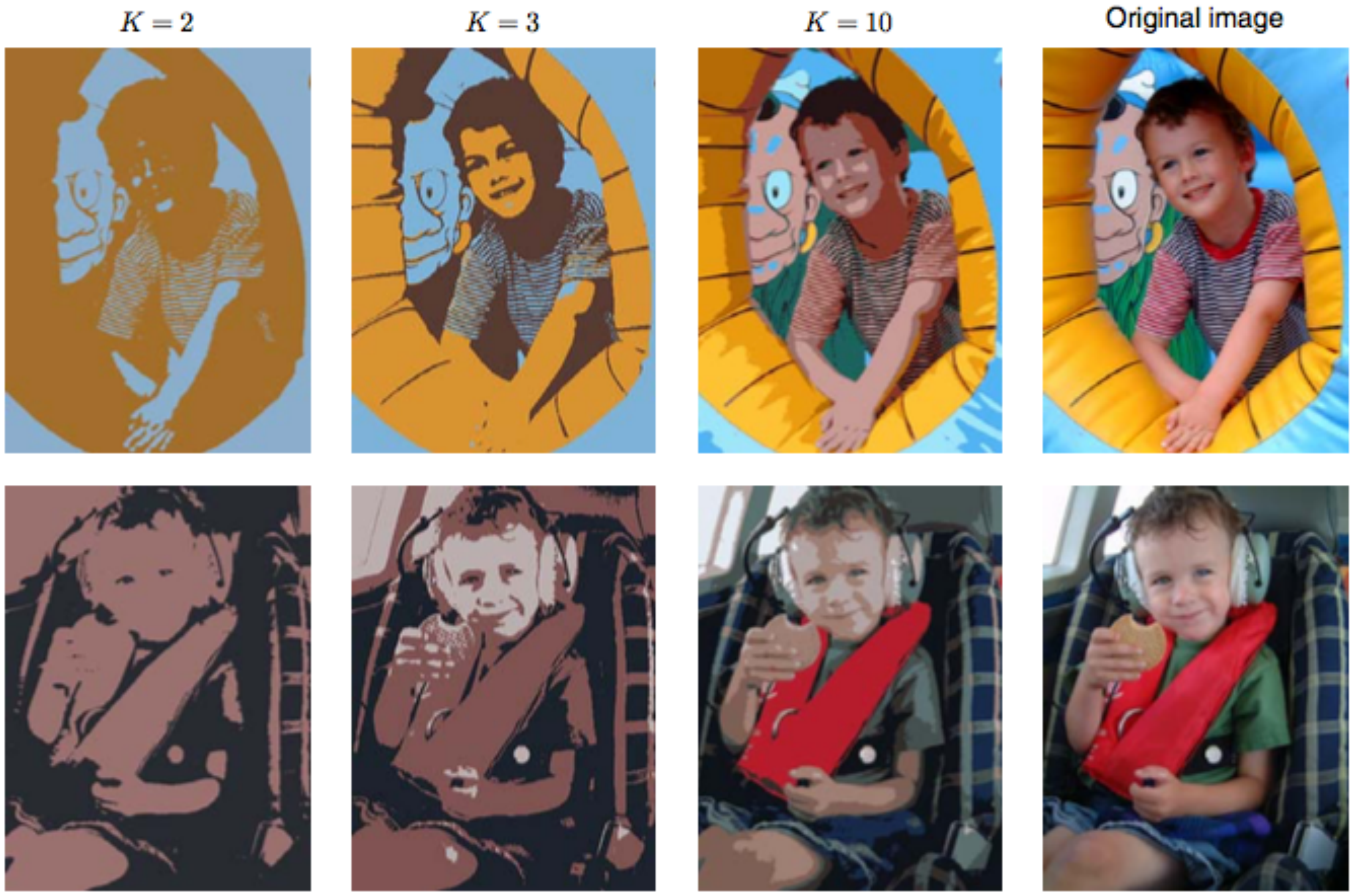
Figure 8.33. Average silhouette coefficient versus number of clusters for the data of Figure 8.29.

Example: Image Compression

- Consider the following (simplistic) method of image compression via K-Means
  - Cluster distinct colors
  - Represent the image pixels as “pointers” to *K* color means
    - **Vector Quantization**, where the K are **Code-Book Vectors**

- NOT a good image segmentation/ compression approach, but illustrates tradeoffs nicely

Change Values of K



Choosing *K*

- If each of N pixels requires 3 colors, each with 8 bits of precision, how many bits for the whole image?
  - 24N
- How many bits for a "pointer" pixel?
  - $\log_2 K$
- So total transmission: $24K + N\log_2 K$
  - 2~4%; 3~8%; 10~17%

Post Processing

- Given the result of K-Means on an initialization/K, it is common to alternate splitting/merging clusters to reduce SSE
- Common operations
  - **Add**. points with high SSE
  - **Split**. highest SSE, largest SD of an attribute
  - **Remove**. increases SSE least
  - **Merge**. close or increase SSE least

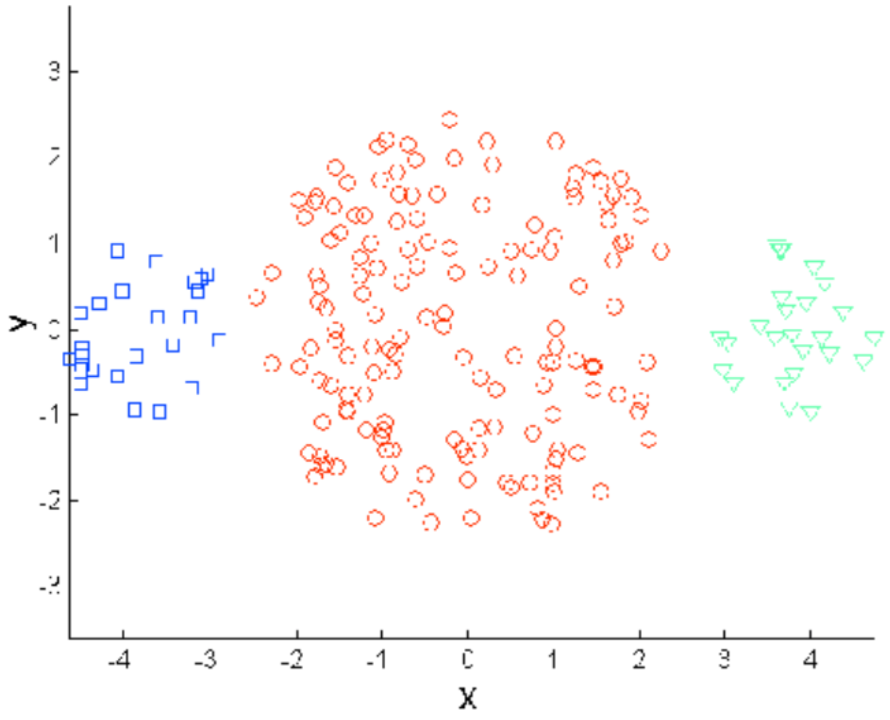
Core K-Means Assumption

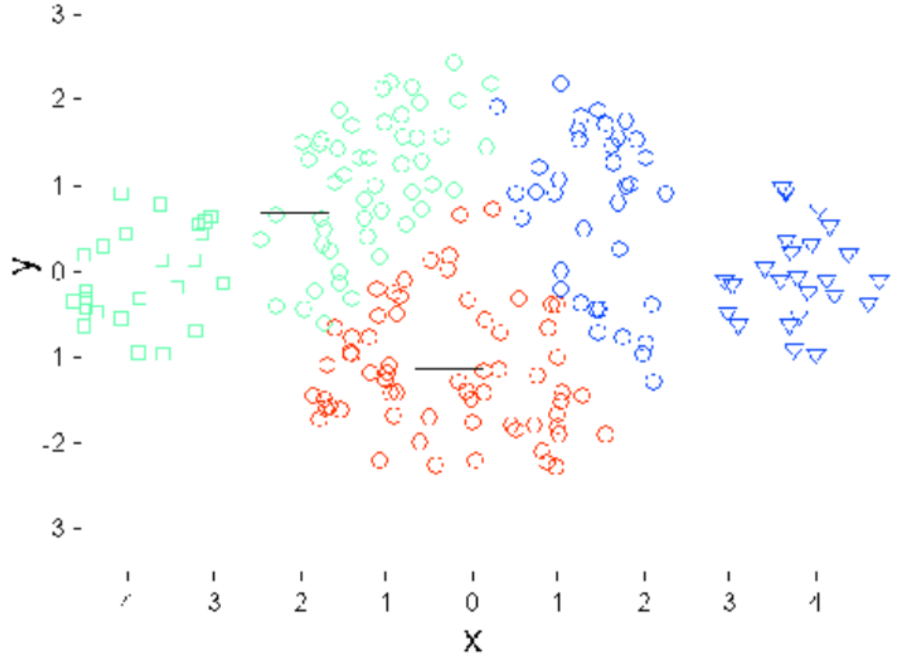
- Look to definition of SSE

$$\arg\min_{\mathbf{r_{nk}},\mu_\mathbf{k}} \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} ||\mathbf{x_n} - \mu_\mathbf{k}||^2$$

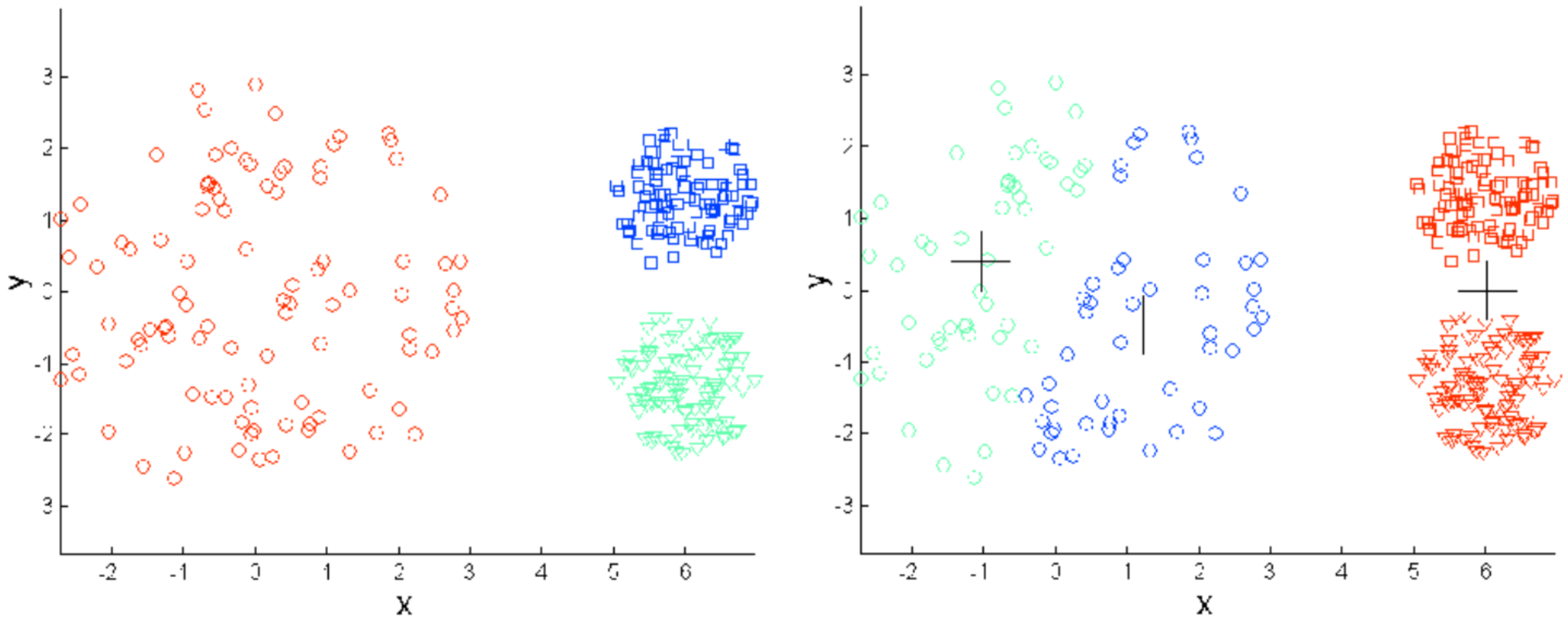
- Uniform "spherical" clusters
  - Same size/density
    - Points/clusters aren't weighted
  - Across dimensions
    - Dimensions aren't weighted

Different Sizes

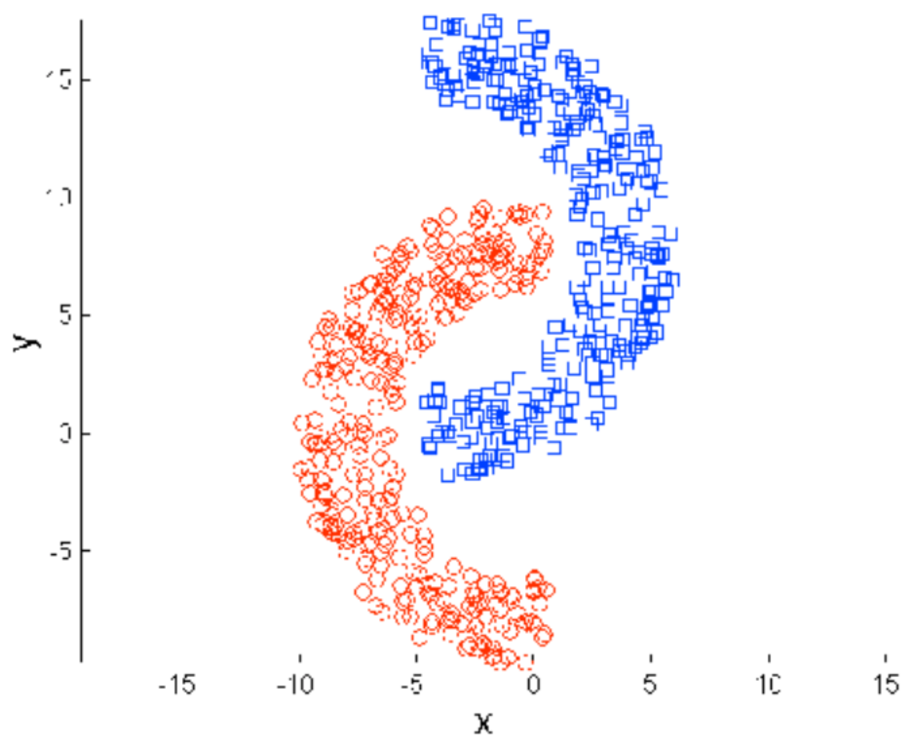


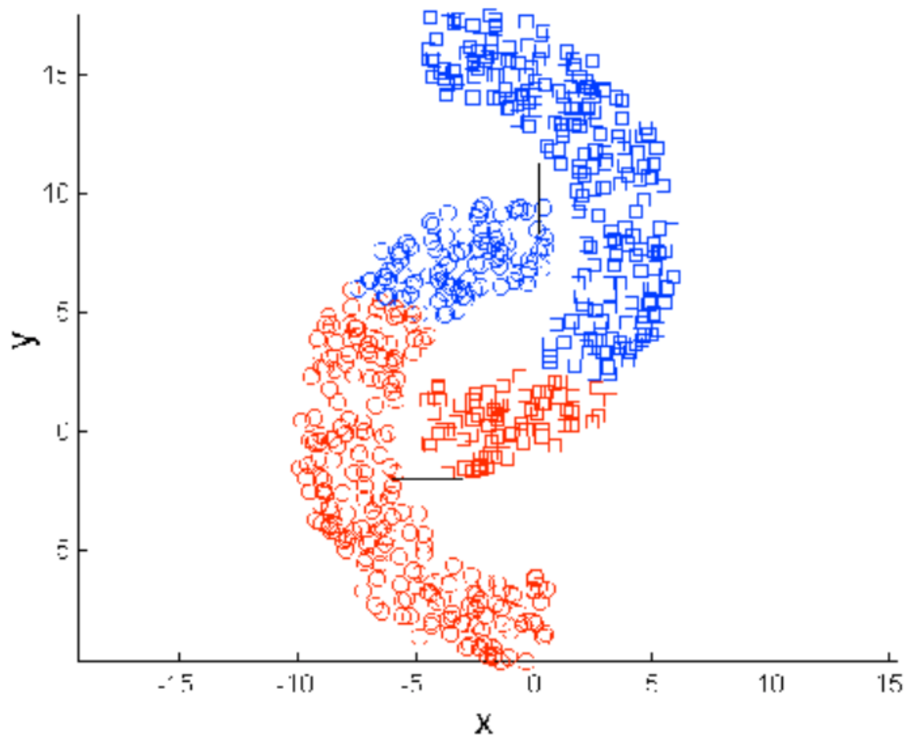


Different Densities

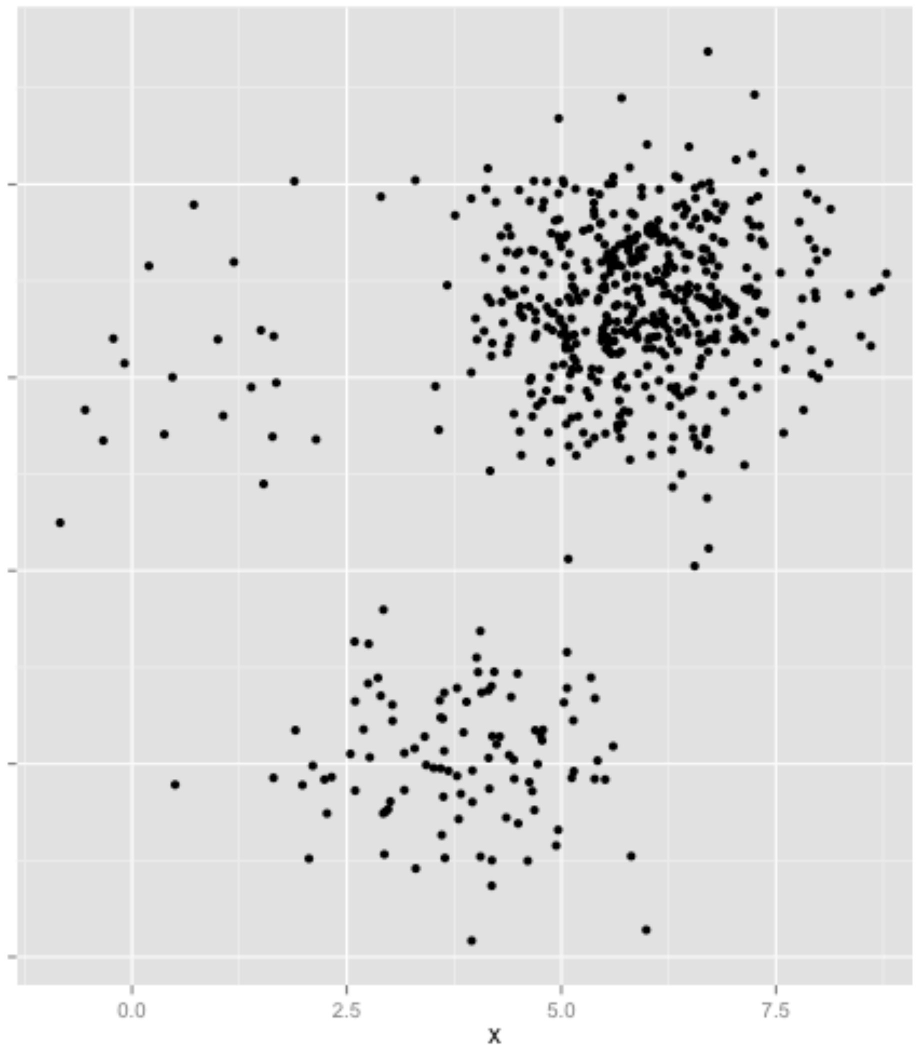


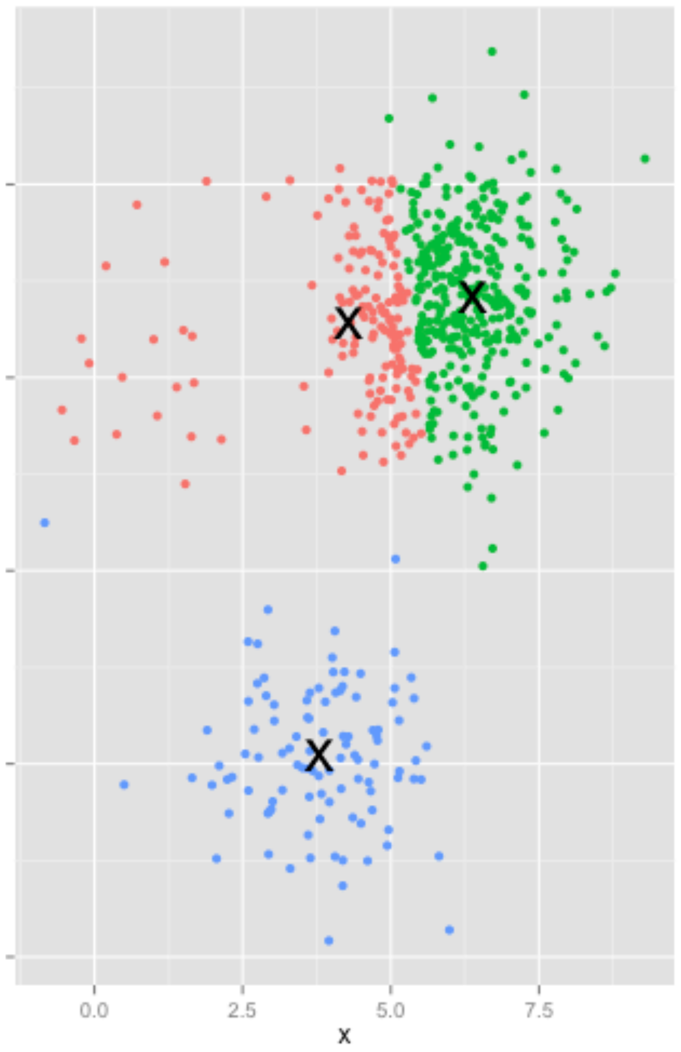
Non-Spherical Shapes



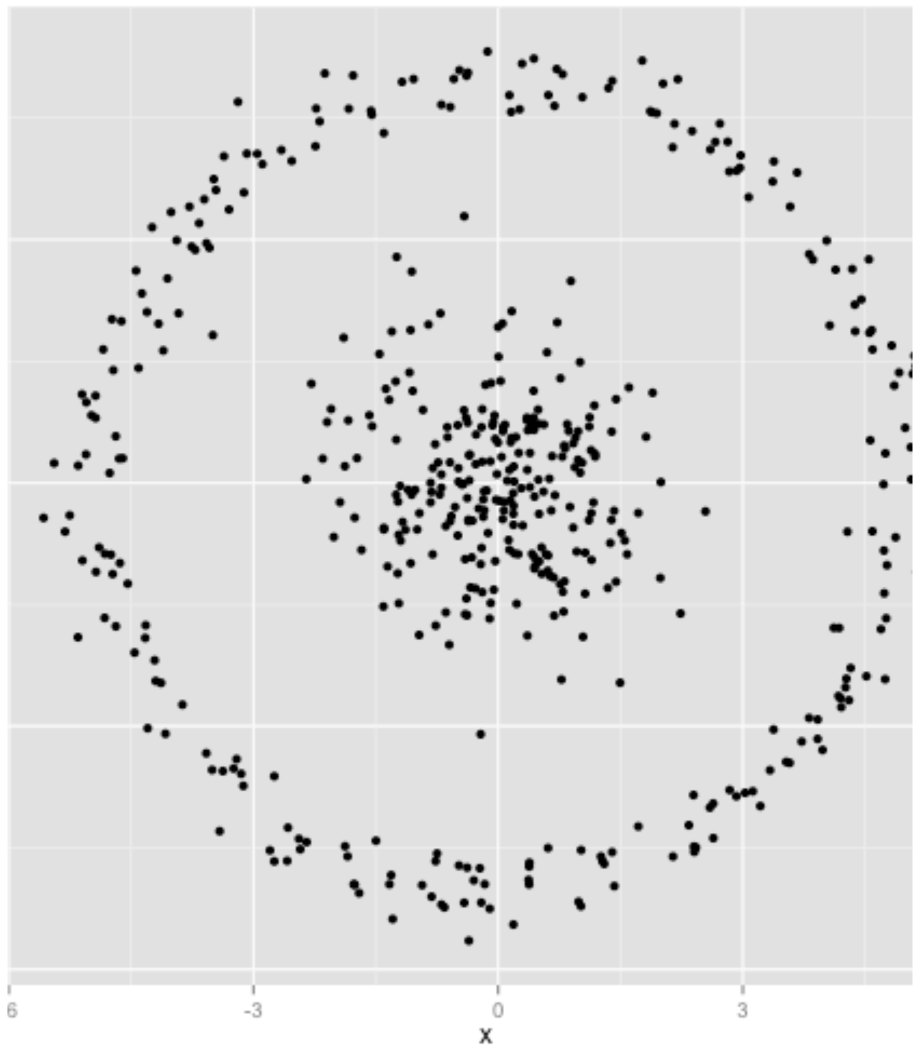


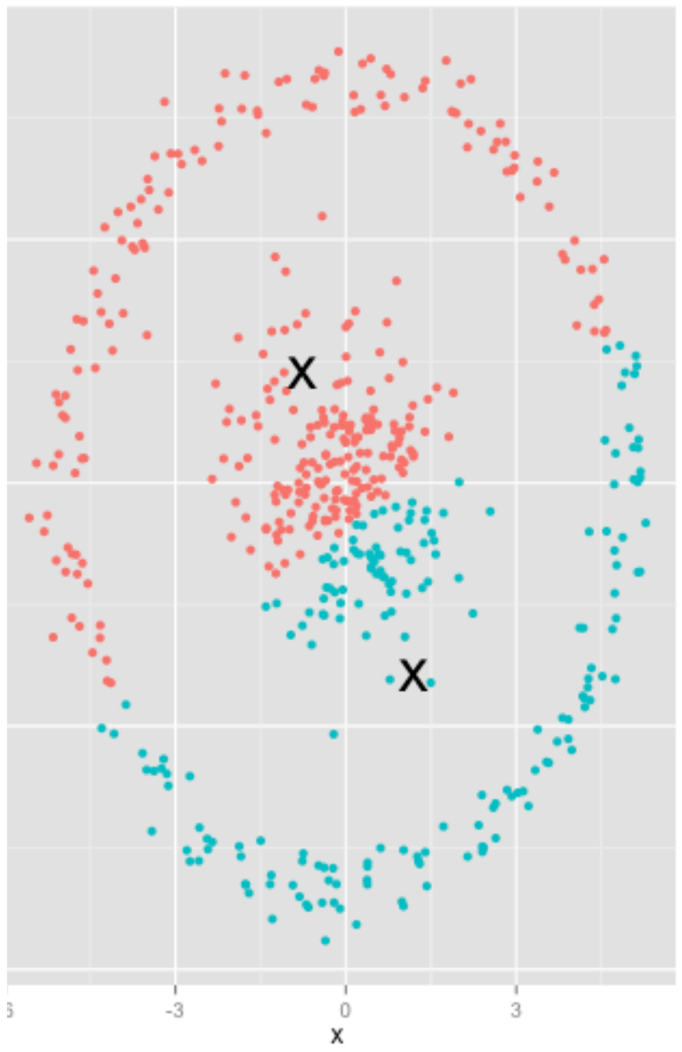
Quick Check: K=3





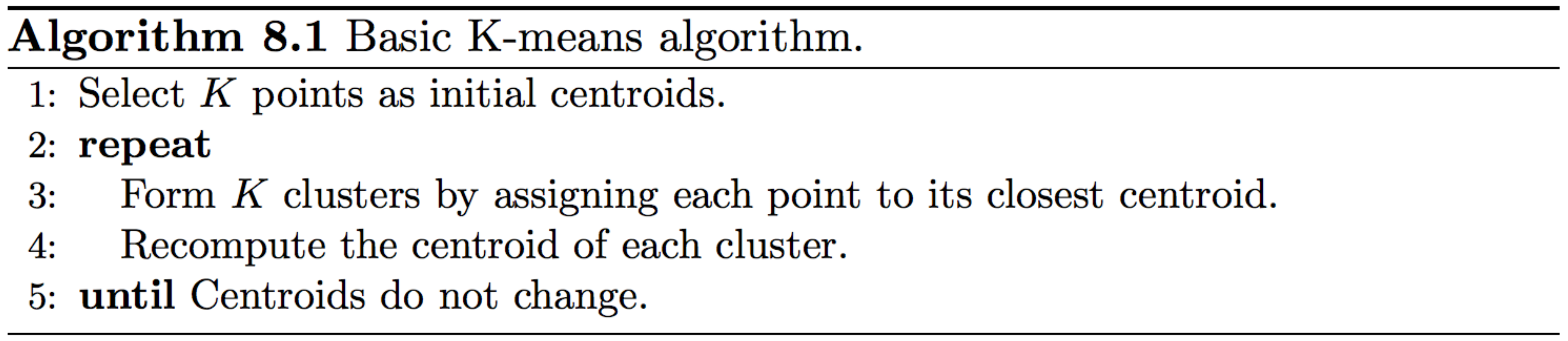
Quick Check: K=2





K-Means Complexity (1)

- What are the parameters of the base algorithm?

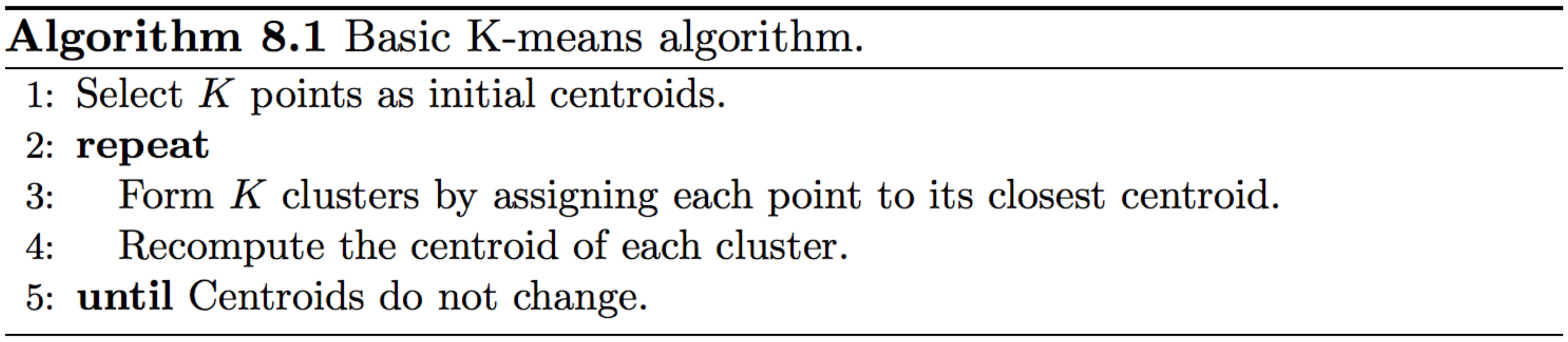


Algorithm 8.1 Basic K-means algorithm.

```
1: Select $K$ points as initial centroids.
2: repeat
3:     Form $K$ clusters by assigning each point to its closest centroid.
4:     Recompute the centroid of each cluster.
5: until Centroids do not change.
```

K-Means Complexity (2)

- What are the parameters of the base algorithm?
  - K = number of centroids
  - N = number of points
  - I = number of iterations
  - D = number of dimensions $|| x – \mu ||^2$

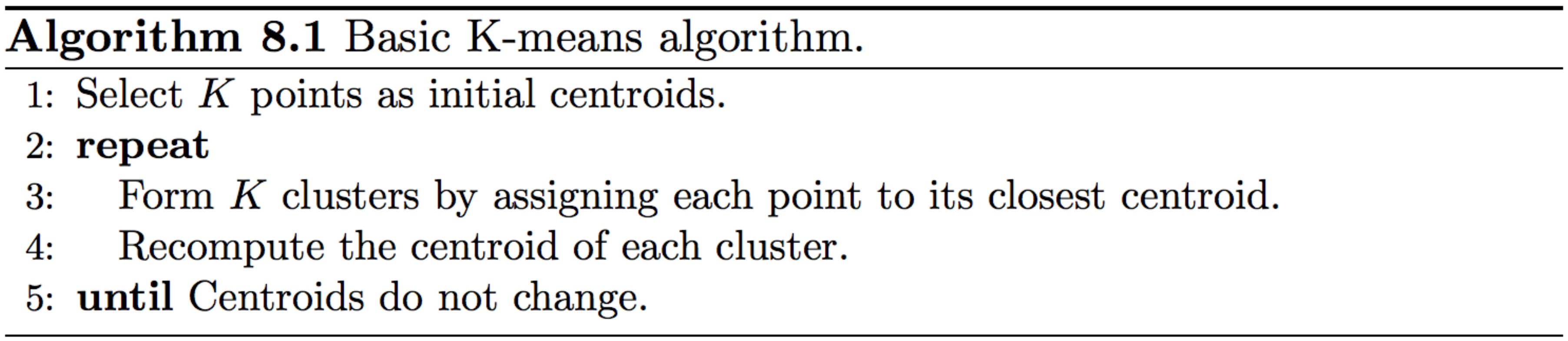


Algorithm 8.1 Basic K-means algorithm.

```
1: Select $K$ points as initial centroids.
2: repeat
3:     Form $K$ clusters by assigning each point to its closest centroid.
4:     Recompute the centroid of each cluster.
5: until Centroids do not change.
```

K-Means Complexity (3)

- Initialization
  - KD
- Each iteration
  - NKD
  - NKD

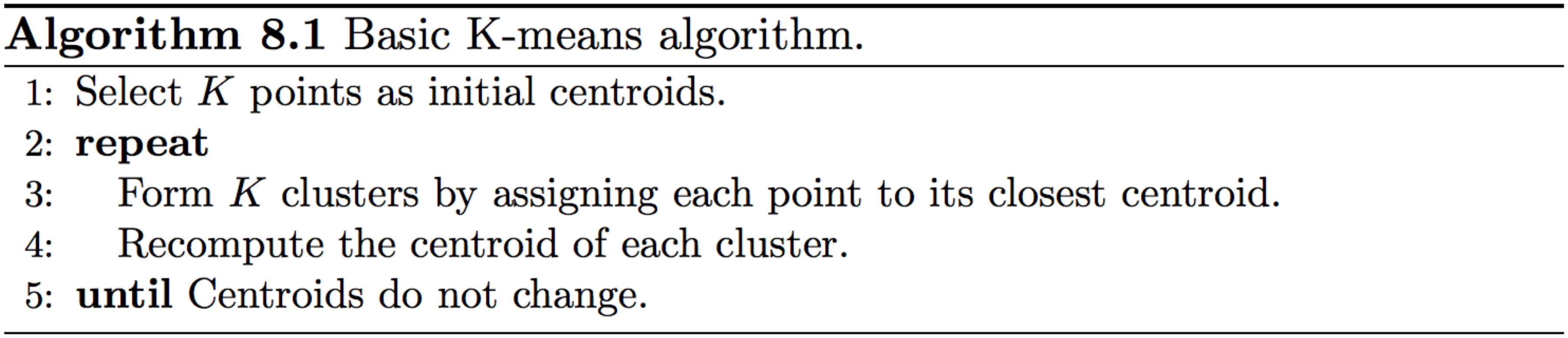


Algorithm 8.1 Basic K-means algorithm.

```
1: Select $K$ points as initial centroids.
2: repeat
3:     Form $K$ clusters by assigning each point to its closest centroid.
4:     Recompute the centroid of each cluster.
5: until Centroids do not change.
```

K-Means Complexity (4)

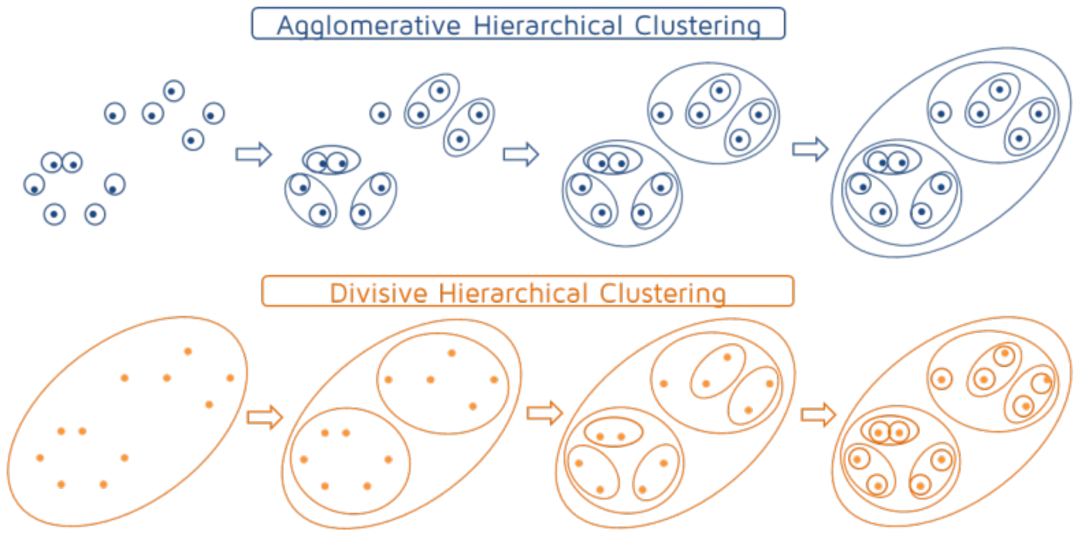
- Overall complexity: $\mathcal{O}$(NKDI)
  - Typically few iterations (10's)
  - Typically: K, D << N
- Variant: Mini-batch K-Means
  - Depends on mini-batch size (M), not (N)
  - Relatively good SSE



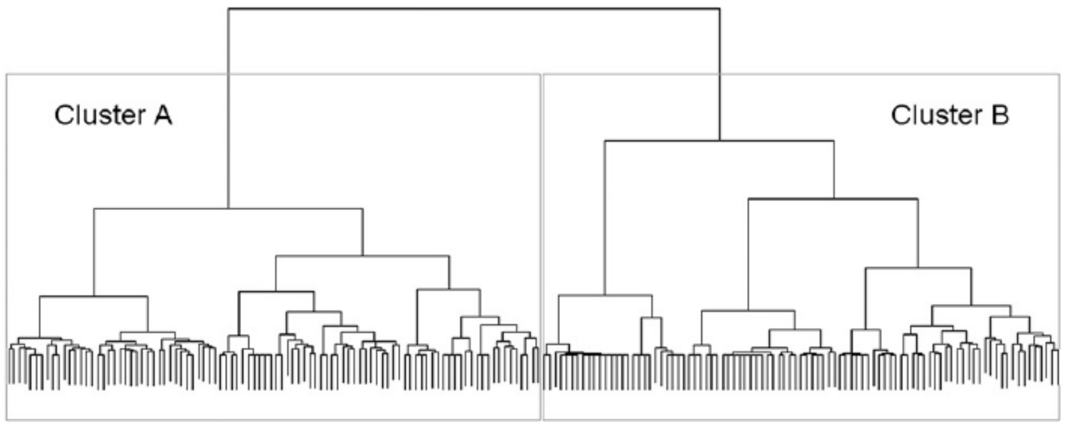
Algorithm 8.1 Basic K-means algorithm.

```
1: Select $K$ points as initial centroids.
2: repeat
3:     Form $K$ clusters by assigning each point to its closest centroid.
4:     Recompute the centroid of each cluster.
5: until Centroids do not change.
```

Hierarchical/Connectivity


Agglomerative Hierarchical Clustering
Divisive Hierarchical Clustering

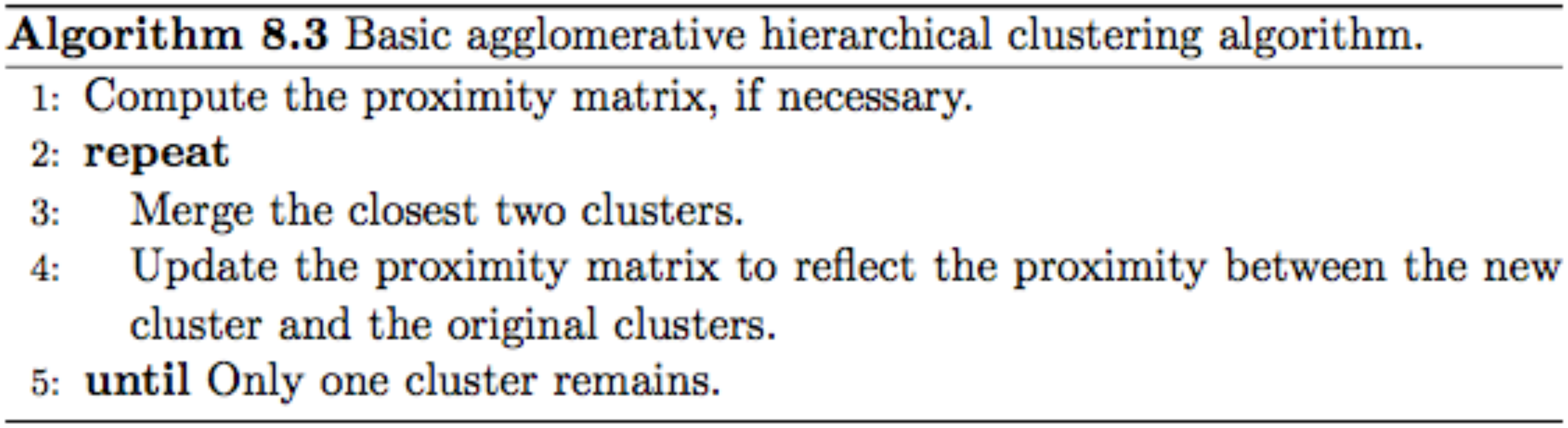
Dendogram cut off at some depth



Agglomerative Clustering

- Much more common than Divisive
- Basic idea
  - Start with all points as individual clusters
  - Loop
    - Merge two "closest" clusters
  - Until only one cluster remains

Algorithm



Algorithm 8.3 Basic agglomerative hierarchical clustering algorithm.

```
1: Compute the proximity matrix, if necessary.
2: repeat
3:     Merge the closest two clusters.
4:     Update the proximity matrix to reflect the proximity between the new
       cluster and the original clusters.
5: until Only one cluster remains.
```

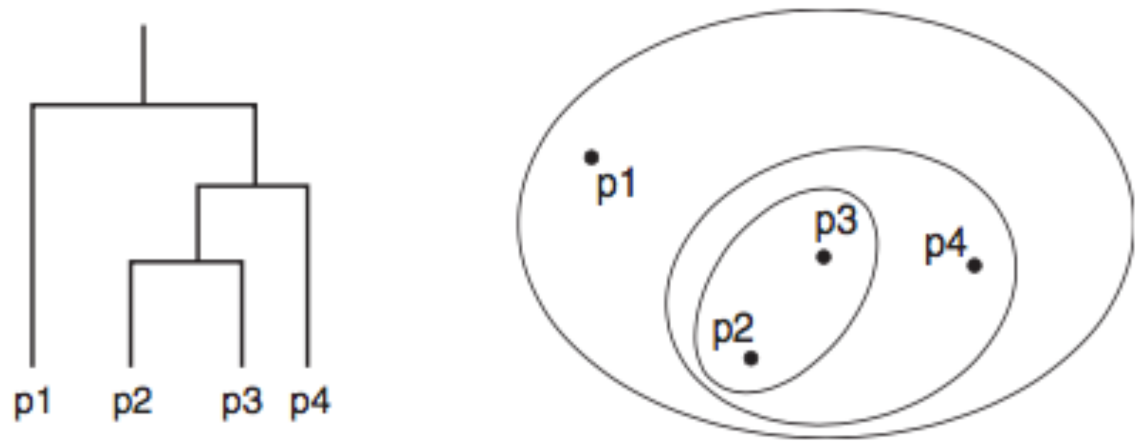
Quick Check

- Hierarchical or Partitional?
- Exclusive, Overlapping, Fuzzy?
- Complete or Partial?
- Centroid, Hierarchical, Density, Distribution?

Quick Check

- **Hierarchical** or Partitional?
- Exclusive, **Overlapping**, Fuzzy?
- **Complete** or Partial?
- Centroid, **Hierarchical**, Density, Distribution?

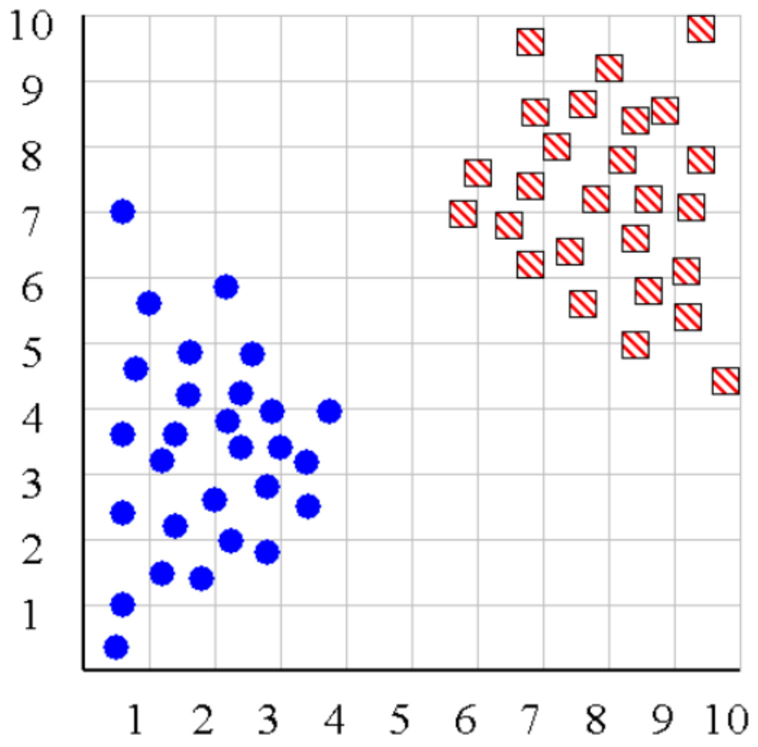
Example Output Representations

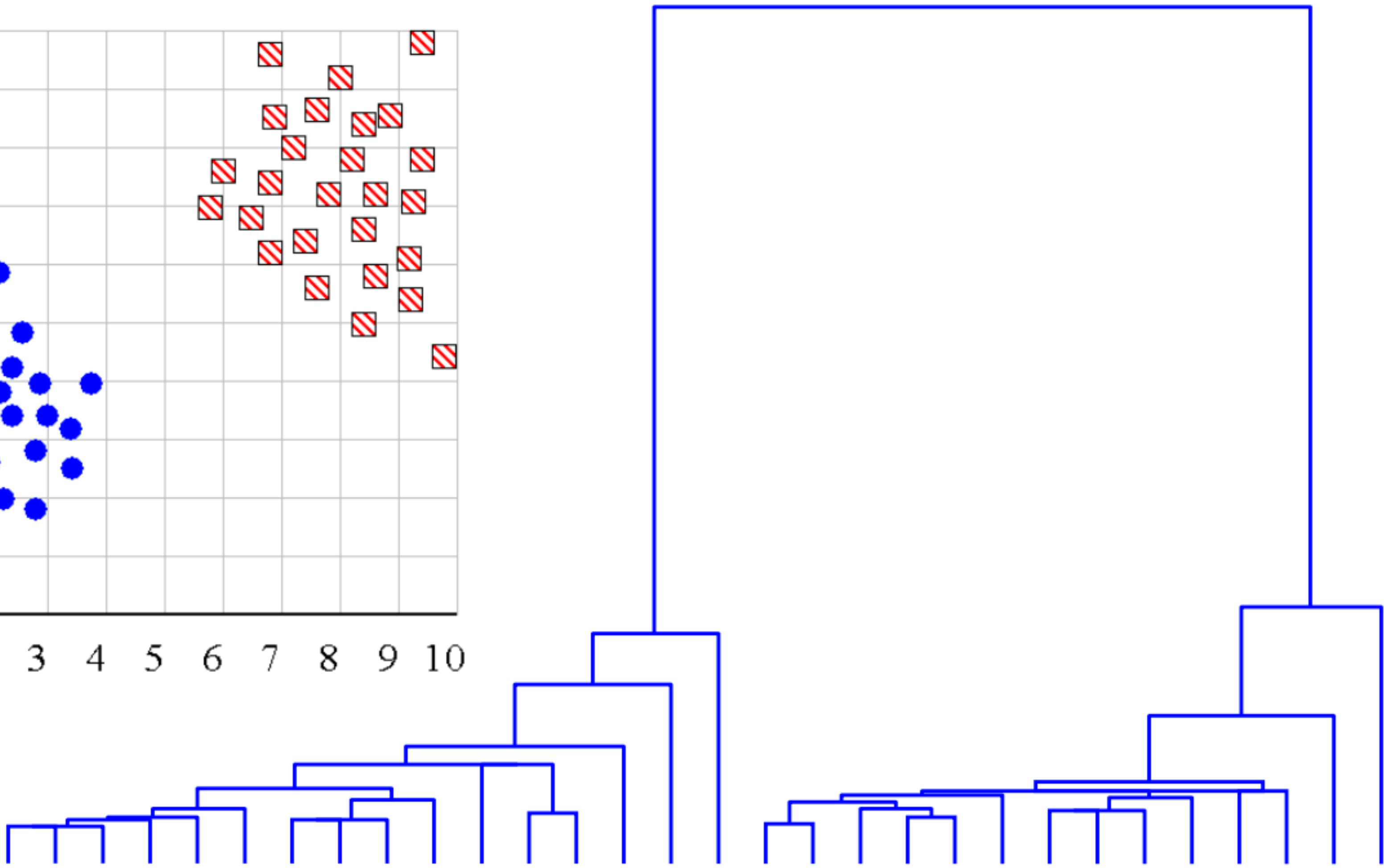


(a) Dendrogram. (b) Nested cluster diagram.

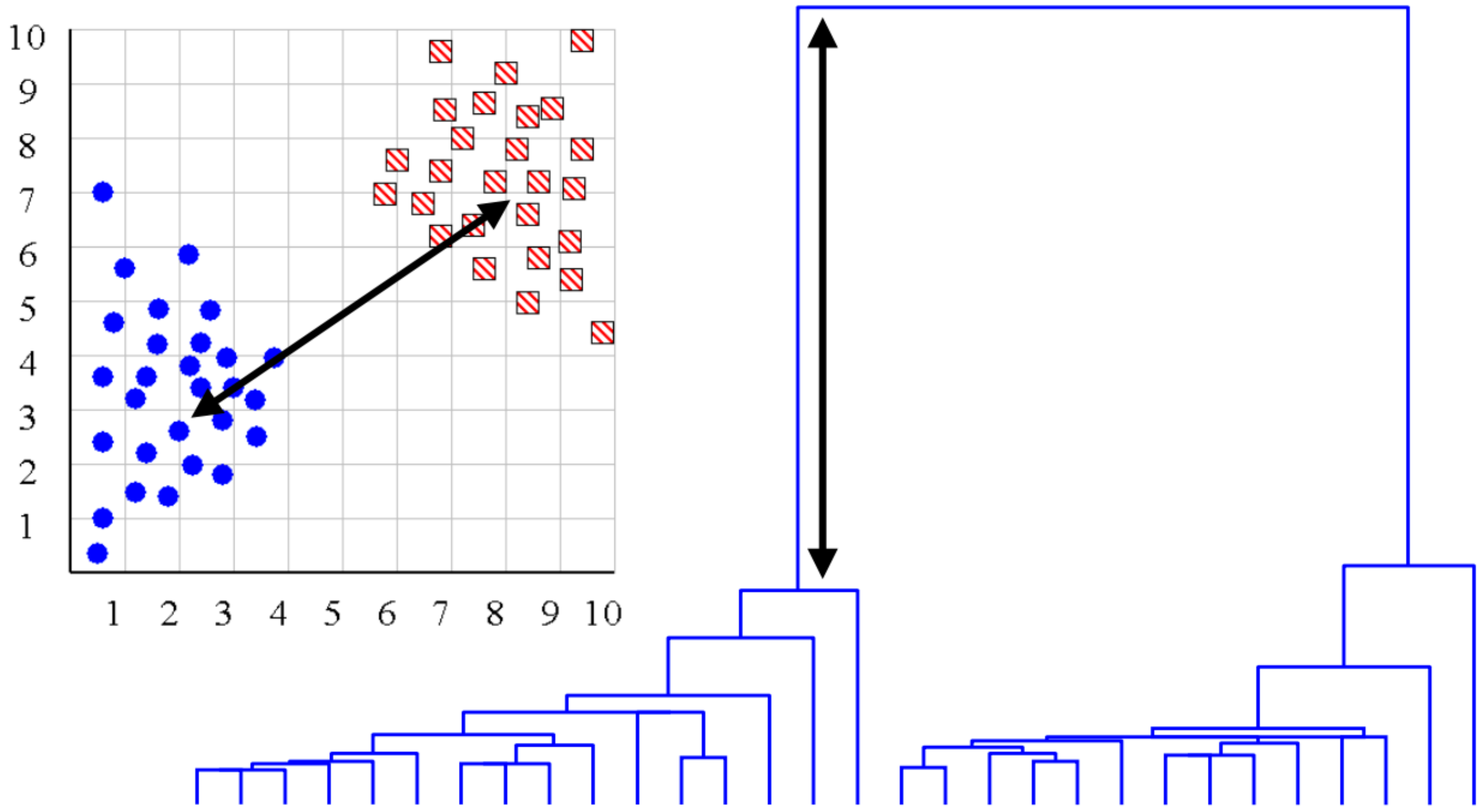
Figure 8.13. A hierarchical clustering of four points shown as a dendrogram and as nested clusters.

Number of Clusters?

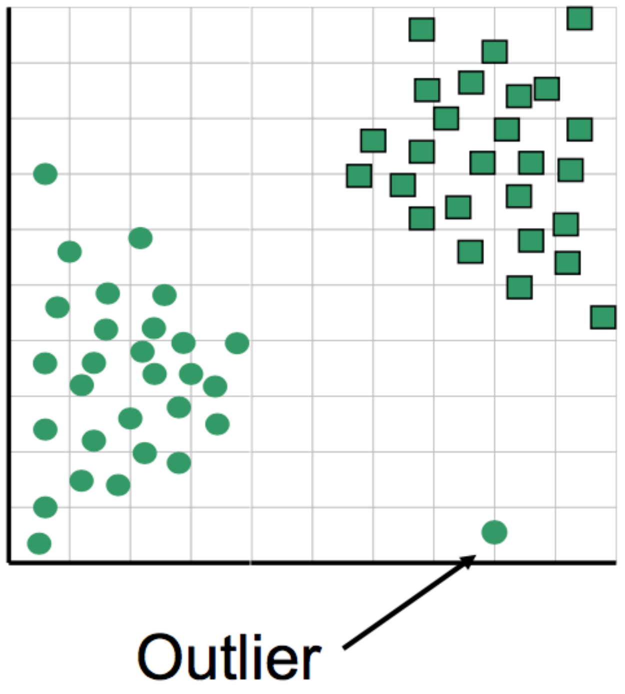


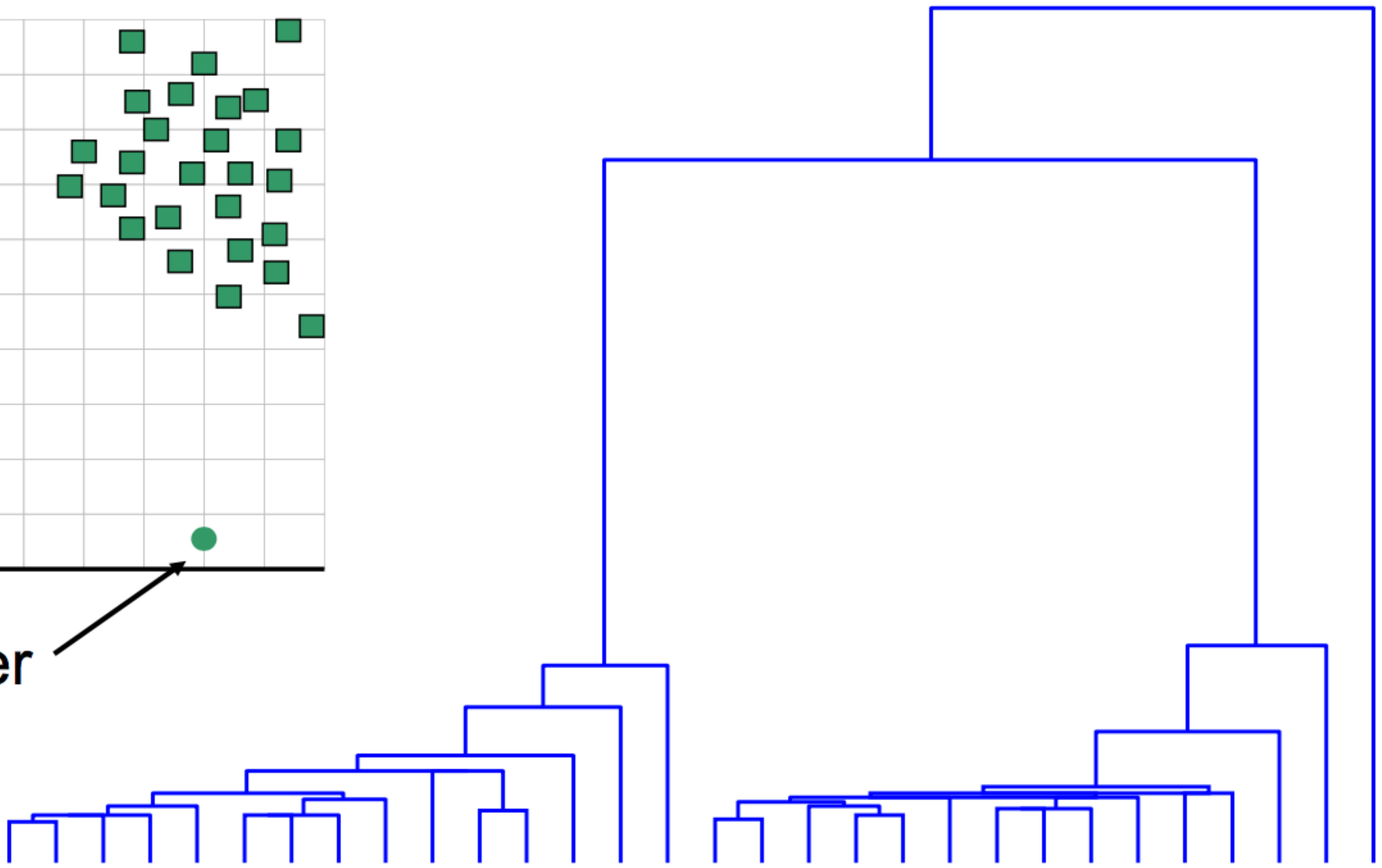


Look to Relative Distance Changes

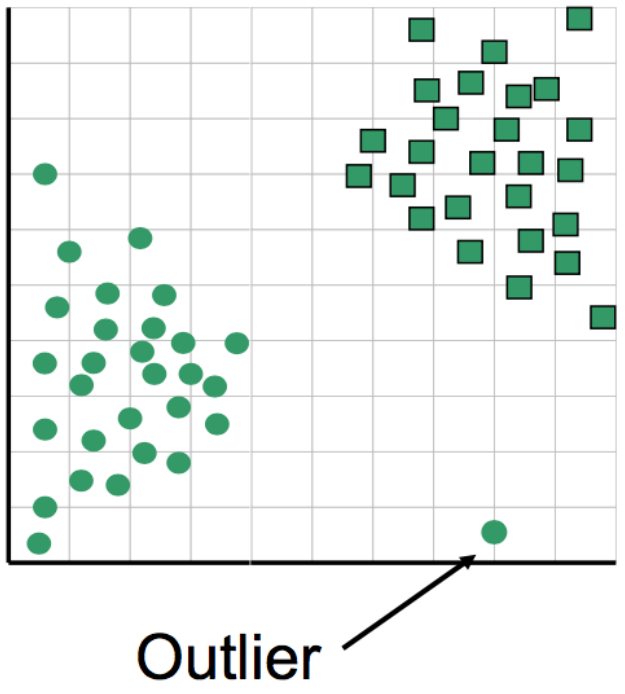


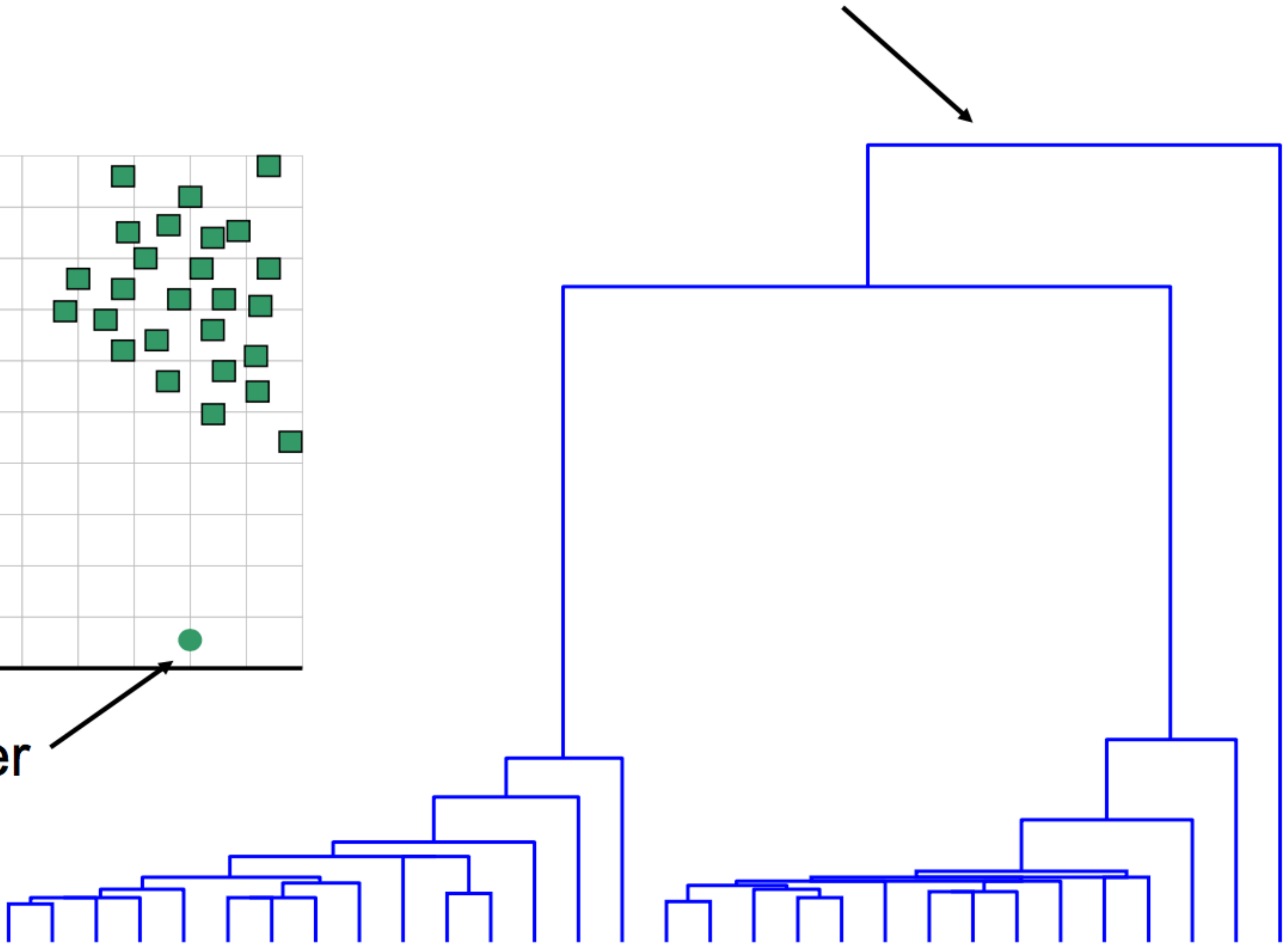
Outliers?





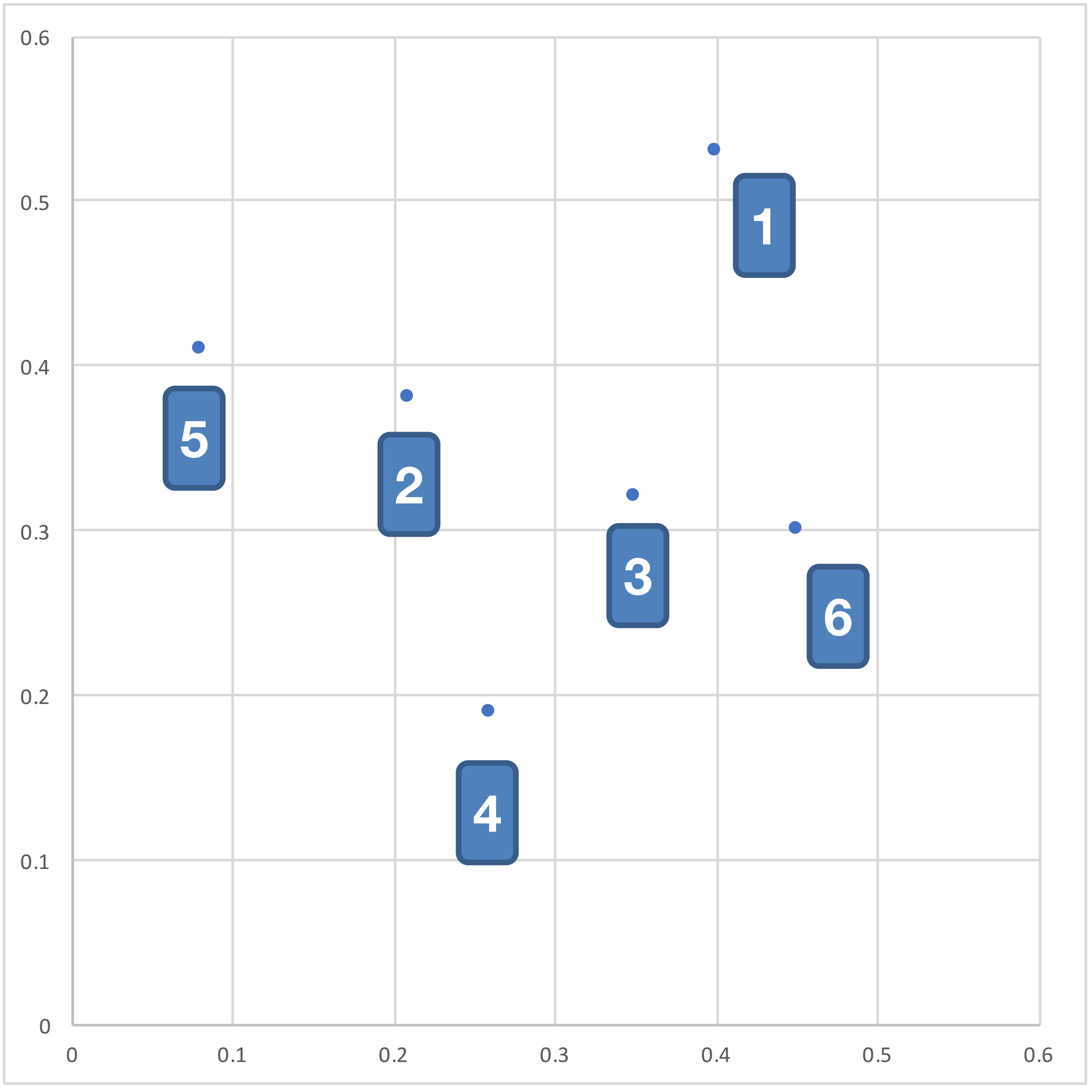
Look to Isolated Branches





Example

	x	y
p1	0.40	0.53
p2	0.21	0.38
p3	0.35	0.32
p4	0.26	0.19
p5	0.08	0.41
p6	0.45	0.30



Example (1)

Compute Proximity Matrix

	x	y
p1	0.40	0.53
p2	0.21	0.38
p3	0.35	0.32
p4	0.26	0.19
p5	0.08	0.41
p6	0.45	0.30

	p1	p2	p3	p4	p5	p6
p1						
p2						
p3						
p4						
p5						
p6						

Example (2)

Compute Proximity Matrix

	x	y
p1	0.40	0.53
p2	0.21	0.38
p3	0.35	0.32
p4	0.26	0.19
p5	0.08	0.41
p6	0.45	0.30

	p1	p2	p3	p4	p5	p6
p1						
p2						
p3						
p4						
p5						
p6						

Example (3)

Compute Proximity Matrix

	x	y
p1	0.40	0.53
p2	0.21	0.38
p3	0.35	0.32
p4	0.26	0.19
p5	0.08	0.41
p6	0.45	0.30

	p1	p2	p3	p4	p5	p6
p1						
p2	.24					
p3	.22	.15				
p4	.37	.20	.16			
p5	.34	.13	.28	.28		
p6	.24	.25	.10	.22	.39	

Example (4)

Minimize!

	x	y
p1	0.40	0.53
p2	0.21	0.38
p3	0.35	0.32
p4	0.26	0.19
p5	0.08	0.41
p6	0.45	0.30

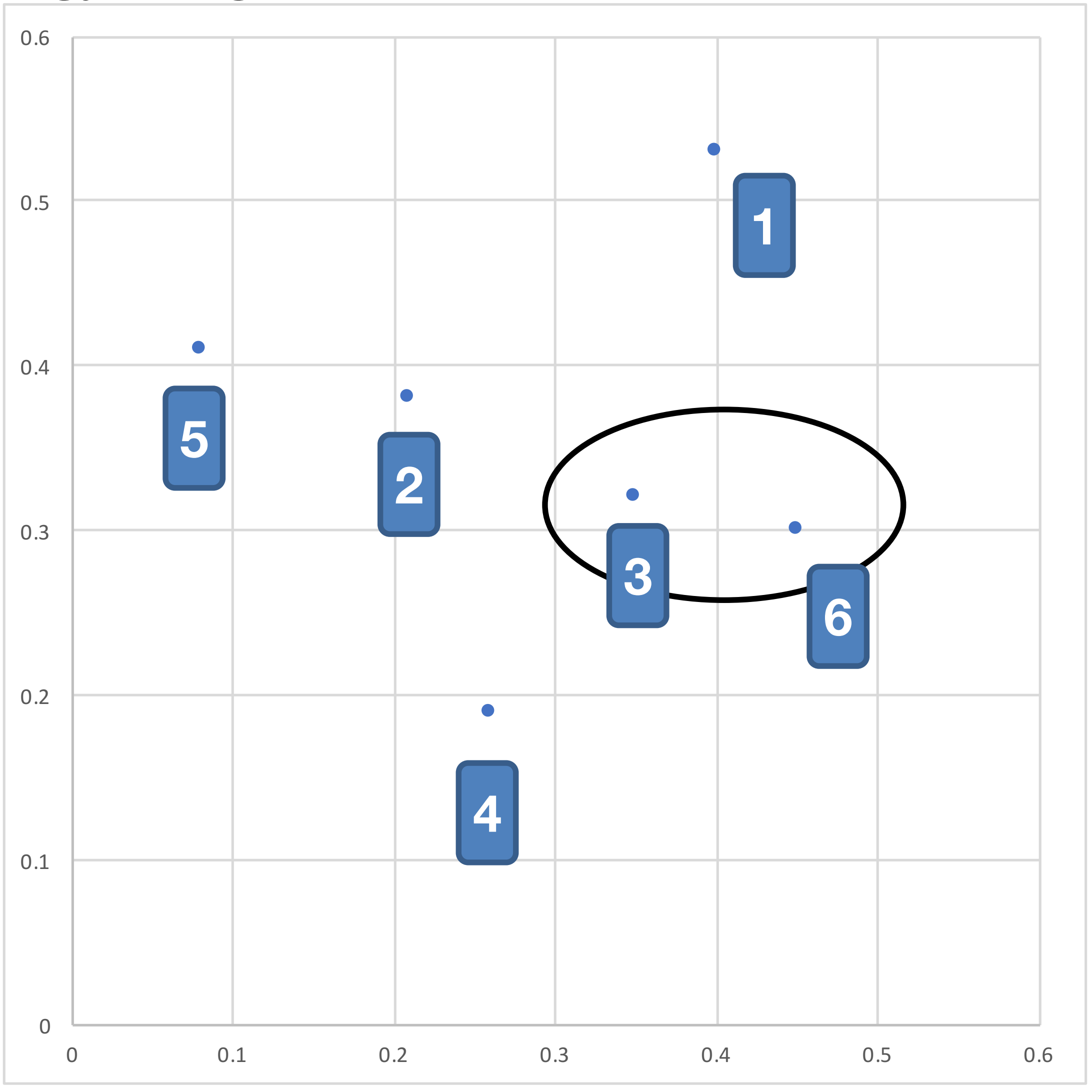
	p1	p2	p3	p4	p5	p6
p1						
p2	.24					
p3	.22	.15				
p4	.37	.20	.16			
p5	.34	.13	.28	.28		
p6	.24	.25	**.10**	.22	.39	

First Cluster: {3, 6}

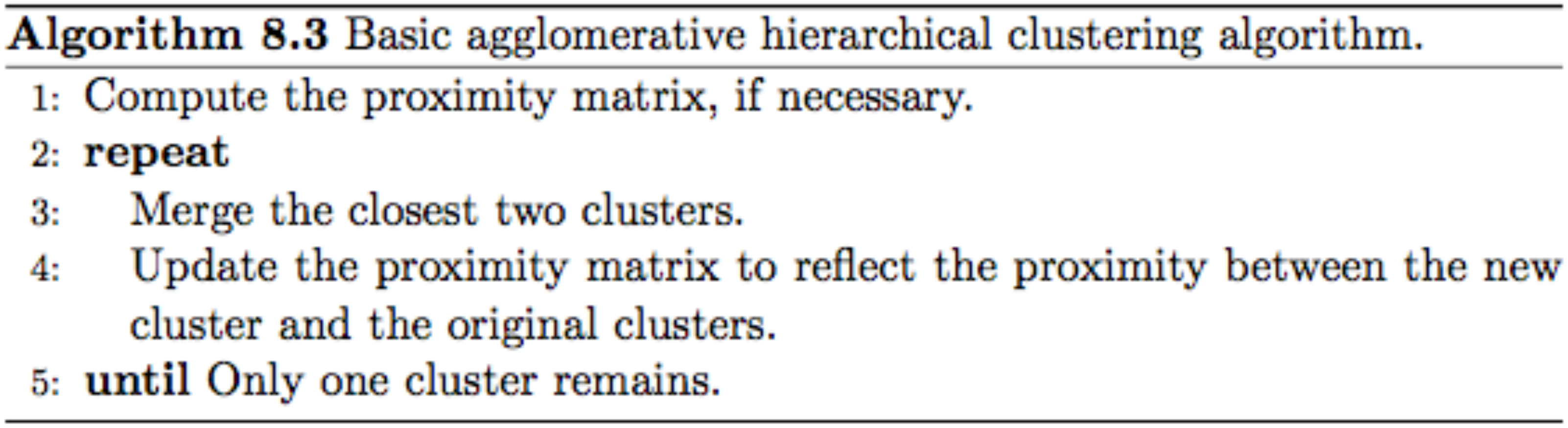
Example (5)

What Now?!

	x	y
p1	0.40	0.53
p2	0.21	0.38
p3	0.35	0.32
p4	0.26	0.19
p5	0.08	0.41
p6	0.45	0.30



Algorithm

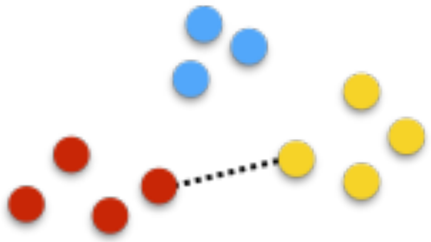


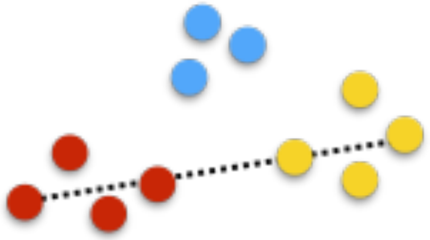
Algorithm 8.3 Basic agglomerative hierarchical clustering algorithm.

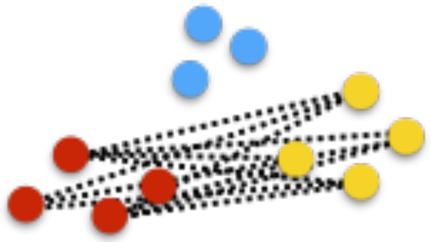
```
1: Compute the proximity matrix, if necessary.
2: repeat
3:     Merge the closest two clusters.
4:     Update the proximity matrix to reflect the proximity between the new
       cluster and the original clusters.
5: until Only one cluster remains.
```

Distances Between Clusters??

- Common criteria:
  - MIN/Single Link
    Closest Point
  - MAX/Complete Link
    Farthest Point
  - AVG/Group
    Average of all pairs



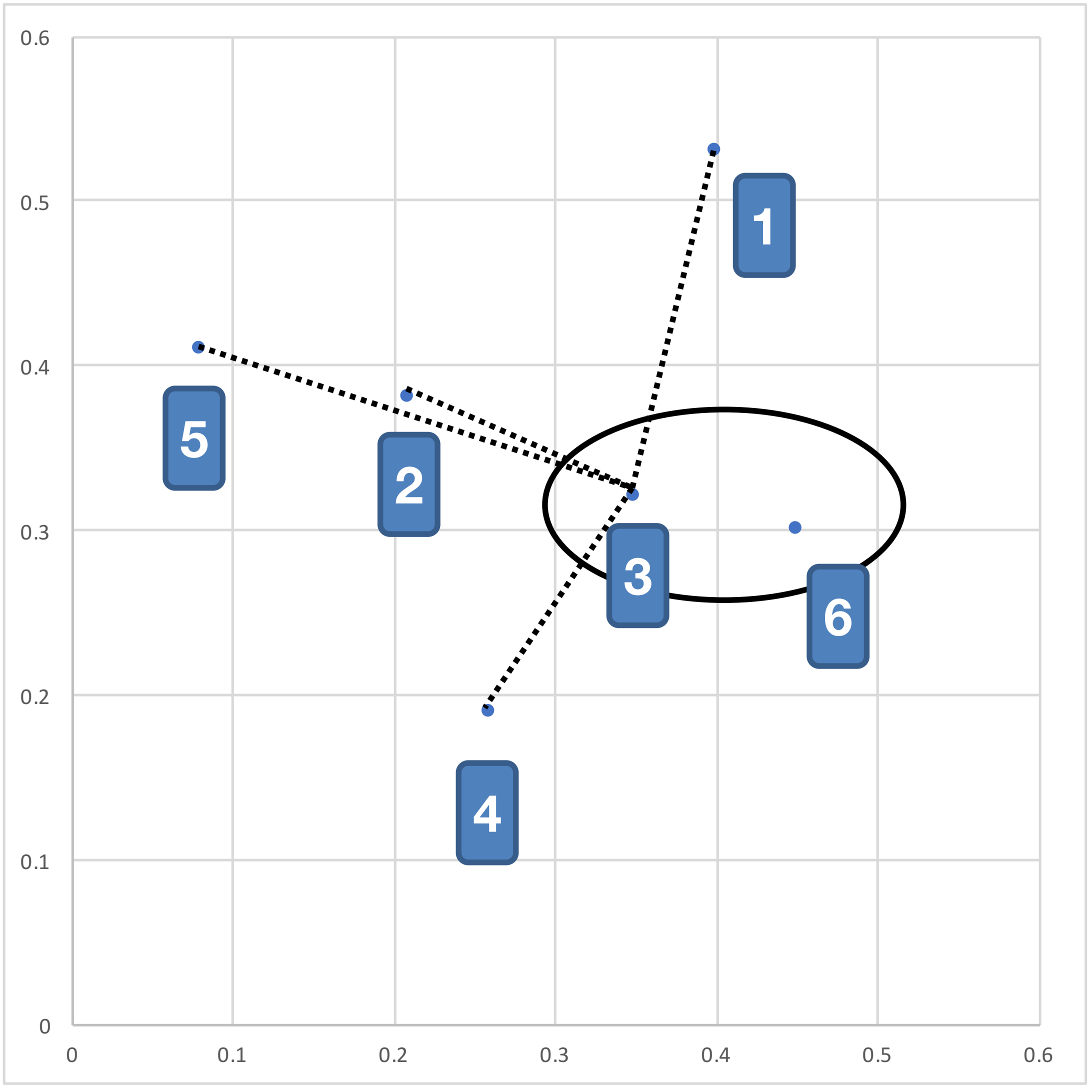




- It turns out these and more (e.g. Ward's) are special cases of the Lance William's Formula (see TSK)

Example (6-MIN)

	x	y
p1	0.40	0.53
p2	0.21	0.38
p3	0.35	0.32
p4	0.26	0.19
p5	0.08	0.41
p6	0.45	0.30



Example (6-MIN)

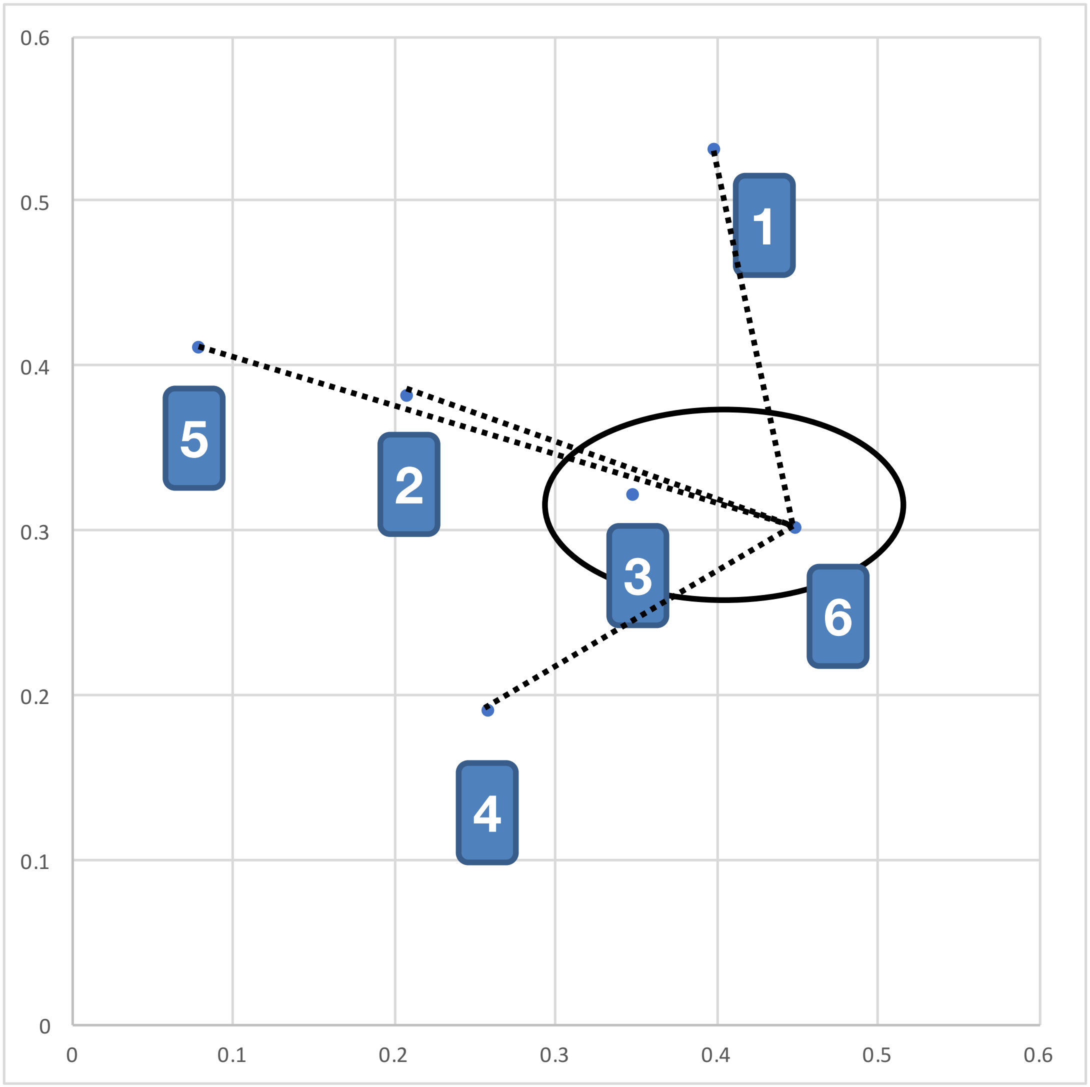
	x	y
p1	0.40	0.53
p2	0.21	0.38
p3	0.35	0.32
p4	0.26	0.19
p5	0.08	0.41
p6	0.45	0.30

	p1	p2	p3	p4	p5	p6
p1						
p2	.24					
p3	.22	.15				
p4	.37	.20	.16			
p5	.34	**.13**	.28	.28		
p6	~~.24~~	~~.25~~	~~.10~~	~~.22~~	~~.39~~	

- d({1},{3,6}) = min(d({1},{3}), d({1},{6})) = d({1},{3})
- d({2},{3,6}) = min(d({2},{3}), d({2},{6})) = d({2},{3})
- d({4},{3,6}) = min(d({4},{3}), d({4},{6})) = d({4},{3})
- d({5},{3,6}) = min(d({5},{3}), d({5},{6})) = d({5},{3})

Example (6-MAX)

	x	y
p1	0.40	0.53
p2	0.21	0.38
p3	0.35	0.32
p4	0.26	0.19
p5	0.08	0.41
p6	0.45	0.30



Example (6-MAX)

	x	y
p1	0.40	0.53
p2	0.21	0.38
p3	0.35	0.32
p4	0.26	0.19
p5	0.08	0.41
p6	0.45	0.30

	p1	p2	p3	p4	p5	p6
p1						
p2	.24					
p3	~~.22~~	~~.15~~				
p4	.37	.20	~~.16~~			
p5	.34	**.13**	~~.28~~	.28		
p6	.24	.25	~~.10~~	.22	.39	

- d({1},{3,6}) = max(d({1},{3}), d({1},{6})) = d({1},{6})
- d({2},{3,6}) = max(d({2},{3}), d({2},{6})) = d({2},{6})
- d({4},{3,6}) = max(d({4},{3}), d({4},{6})) = d({4},{6})
- d({5},{3,6}) = max(d({5},{3}), d({5},{6})) = d({5},{6})

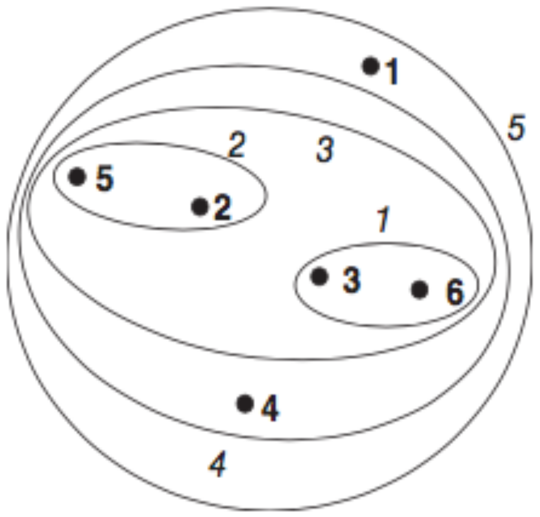
Example (6-AVG)

	x	y
p1	0.40	0.53
p2	0.21	0.38
p3	0.35	0.32
p4	0.26	0.19
p5	0.08	0.41
p6	0.45	0.30

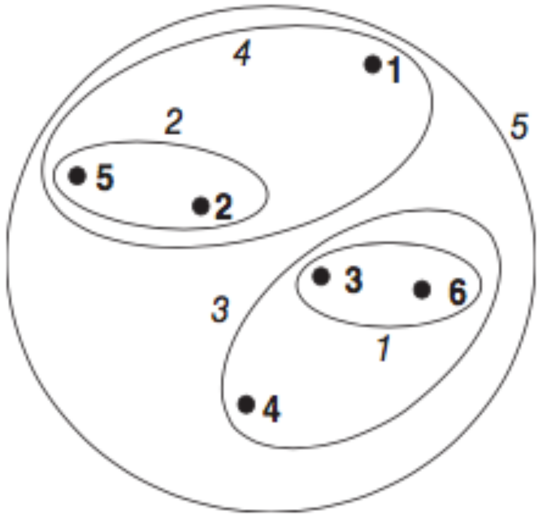
	p1	p2	p3	p4	p5	p6
p1						
p2	.24					
p3	~~.22~~	~~.15~~				
p4	.37	.20	~~.16~~			
p5	.34	**.13**	~~.28~~	.28		
p6	~~.24~~	~~.25~~	~~.10~~	~~.22~~	~~.39~~	

- d({1},{3,6}) = avg(d({1},{3}), d({1},{6})) ~ 0.23
- d({2},{3,6}) = avg(d({2},{3}), d({2},{6})) ~ 0.20
- d({4},{3,6}) = avg(d({4},{3}), d({4},{6})) ~ 0.19
- d({5},{3,6}) = avg(d({5},{3}), d({5},{6})) ~ 0.34

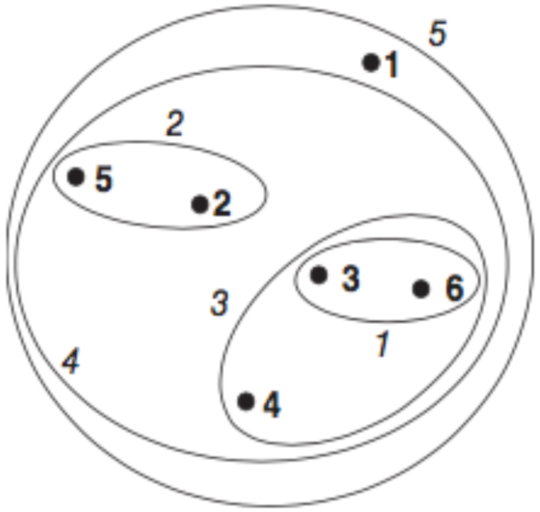
Clustering Comparison



(a) Single link clustering.



(a) Complete link clustering.



(a) Group average clustering.

Algorithm Evaluation

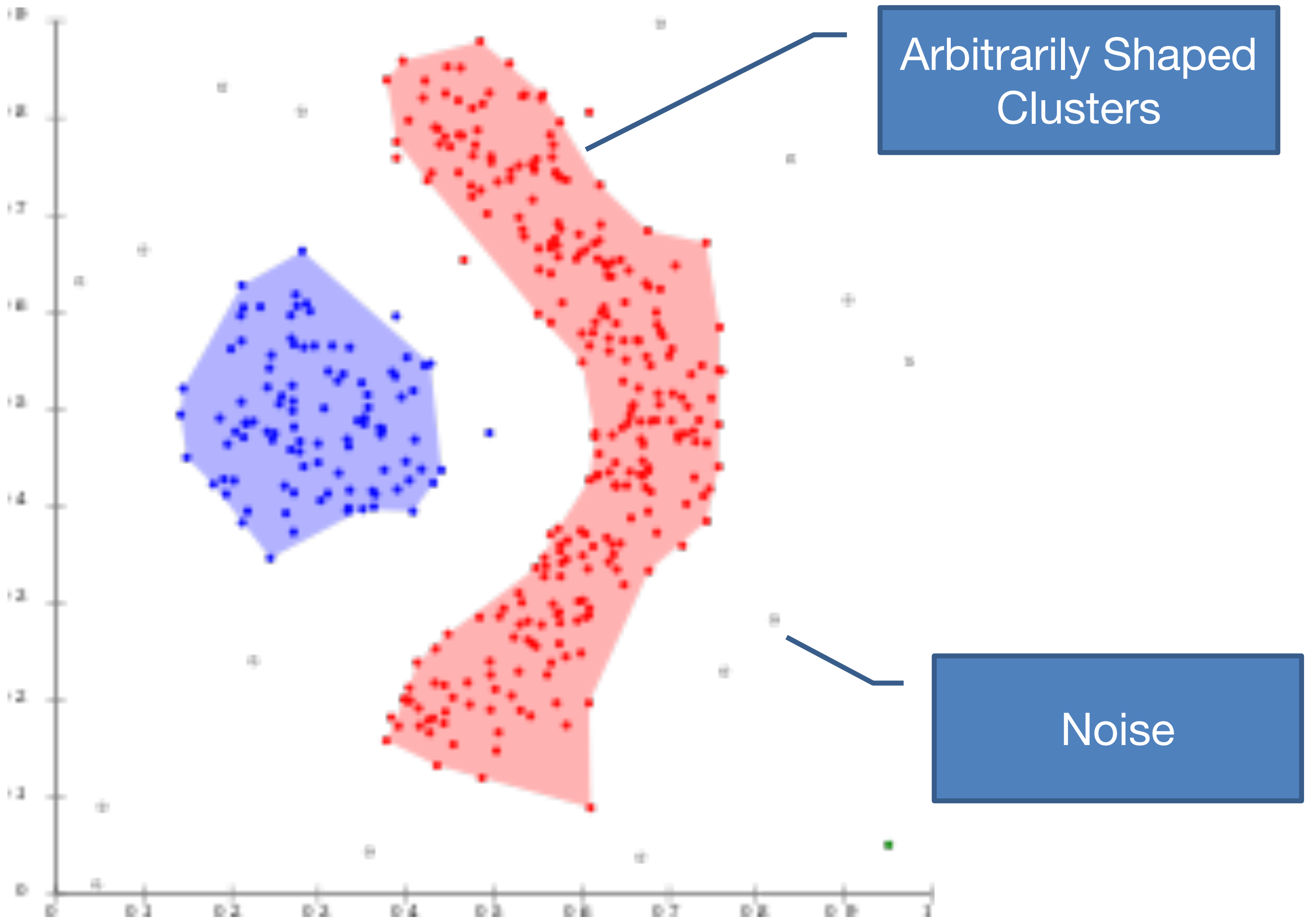
Pros

- No need to specify # clusters, initial points
- Hierarchical result *may* map onto intuition
- Local optimum
- Complexity
  - Space = $\mathcal{O}(n^2)$
  - Time = $\mathcal{O}(n^2 \log n)$
    - Being smart about storing/finding distances

Cons

- Still may want to decide height cutoff (~elbow)
- Interpreting results is subjective

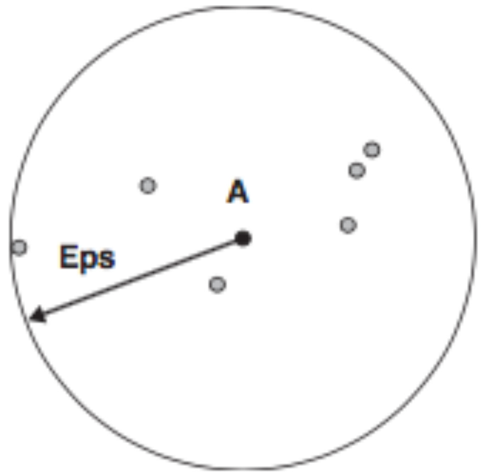
DBSCAN: The Promise



Density-Based Clustering

- We first need a concept of "density" by which we will cluster

- DBSCAN uses a center-based approach
  - How many points are within a small distance (ε, or eps) of a point (including itself)
  - Density of A?



- The **eps-neighborhood (N_ε)** is the set of points within this radius

$$N_\epsilon(p) = \{q \in D | dist(p,q) \leq \epsilon\}$$

Classifying Points via Density

- **Core** (the “interior” of a cluster)

  $|N_\varepsilon(p)| \geq \text{MinPts}$

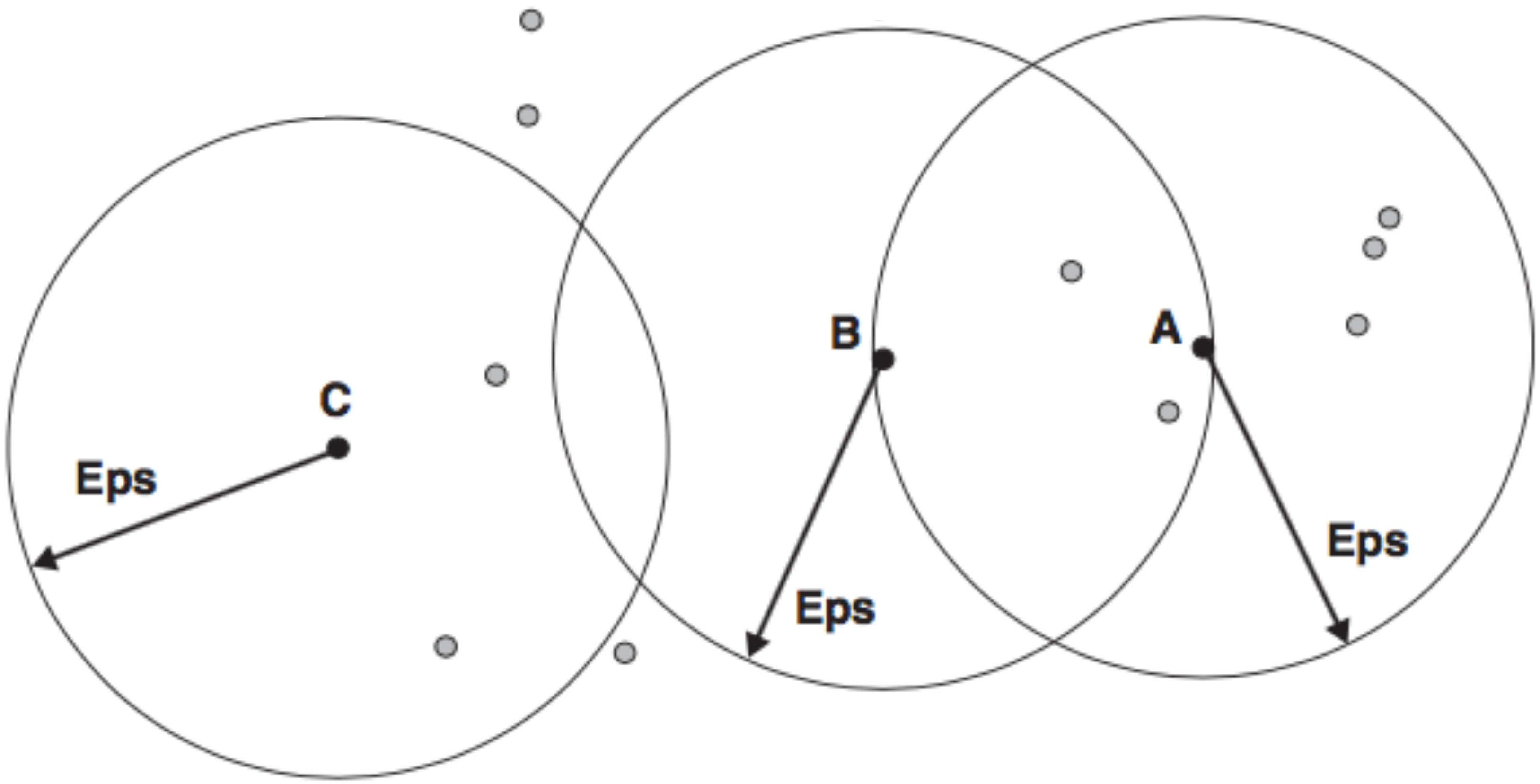
- **Border** (the “edge” of a cluster)

  $|N_\varepsilon(q)| < \text{MinPts}$

  $q \in N_\varepsilon(p)$, where p is a core point

- **Noise** (neither core nor border)

Example: MinPts=7



Direct Reachability

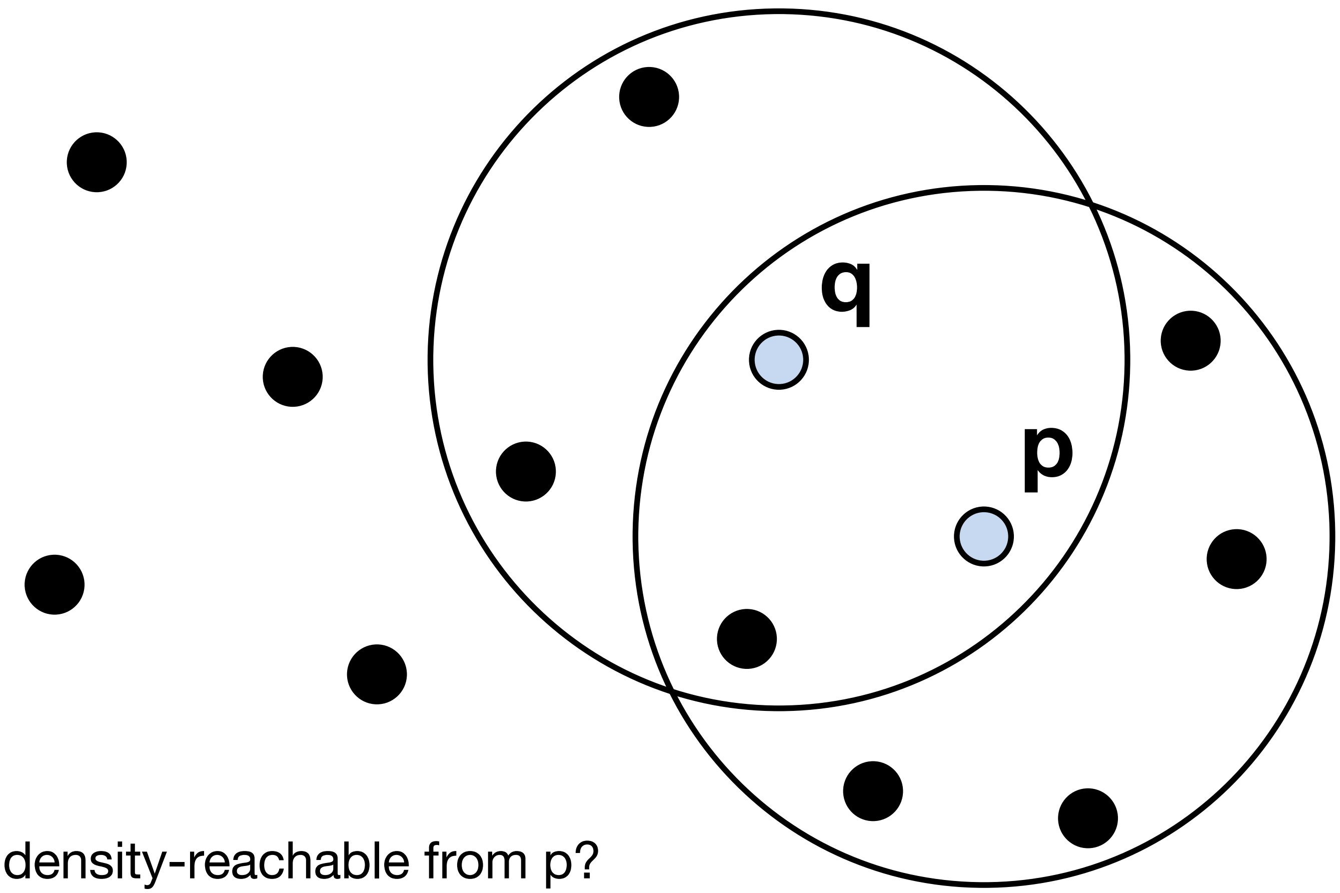
- A point q is **directly density-reachable** from point p w.r.t. eps and MinPts if…

  $q \in N_\varepsilon(p)$

  $|N_\varepsilon(p)| \geq \text{MinPts}$

- Thus, no points are directly reachable from a non-core point

Example: MinPts=6

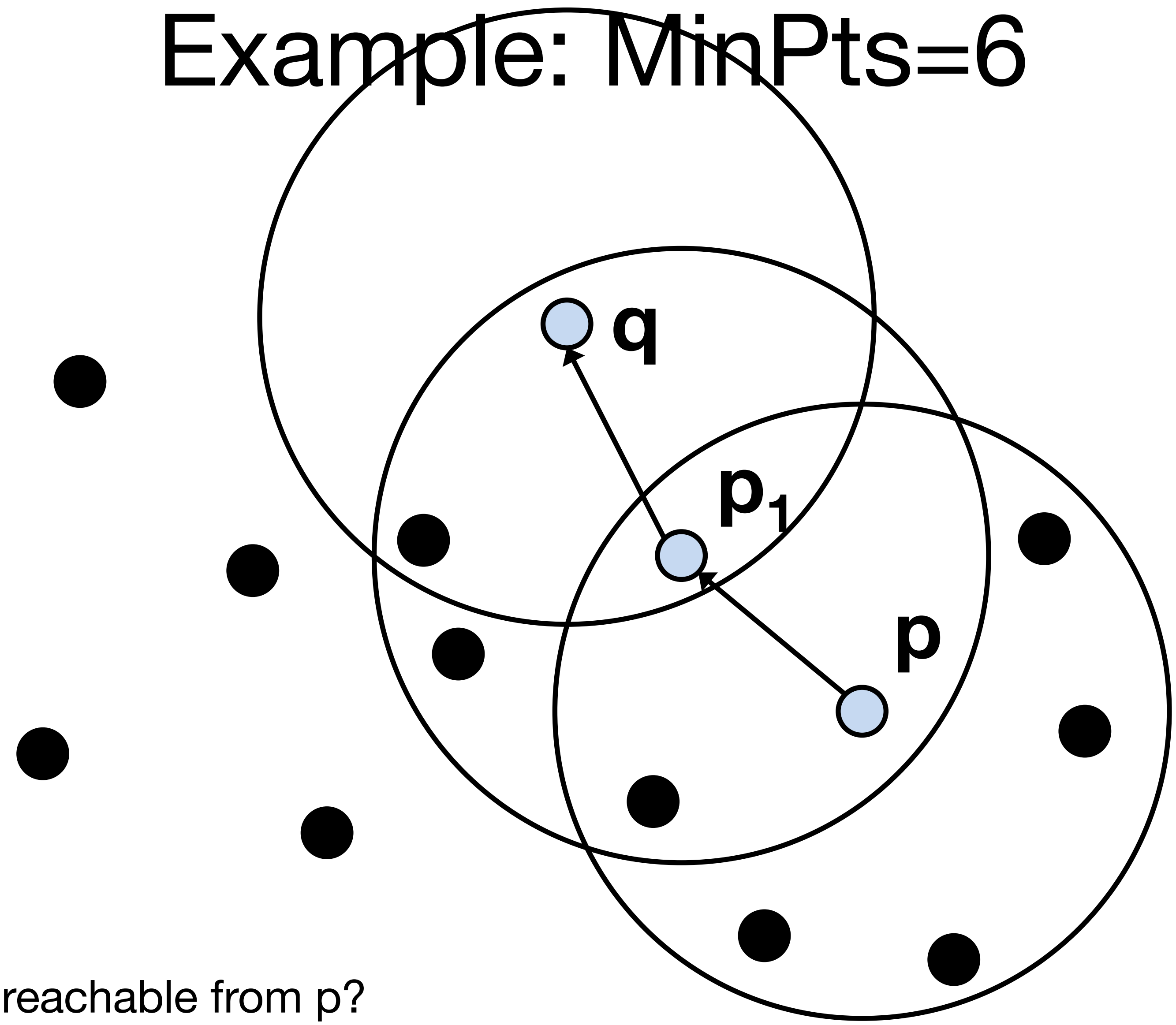


q directly density-reachable from p?
Reverse? Symmetric property?

Density Reachability

- A point q is **density-reachable** from a point p w.r.t. eps and MinPts if…
  - – There is a chain p_0 (=p), p_1, p_2, … p_n (=q)
  - – p_{i+1} is directly density reachable from p_i
    - i need not include n

Example: MinPts=6

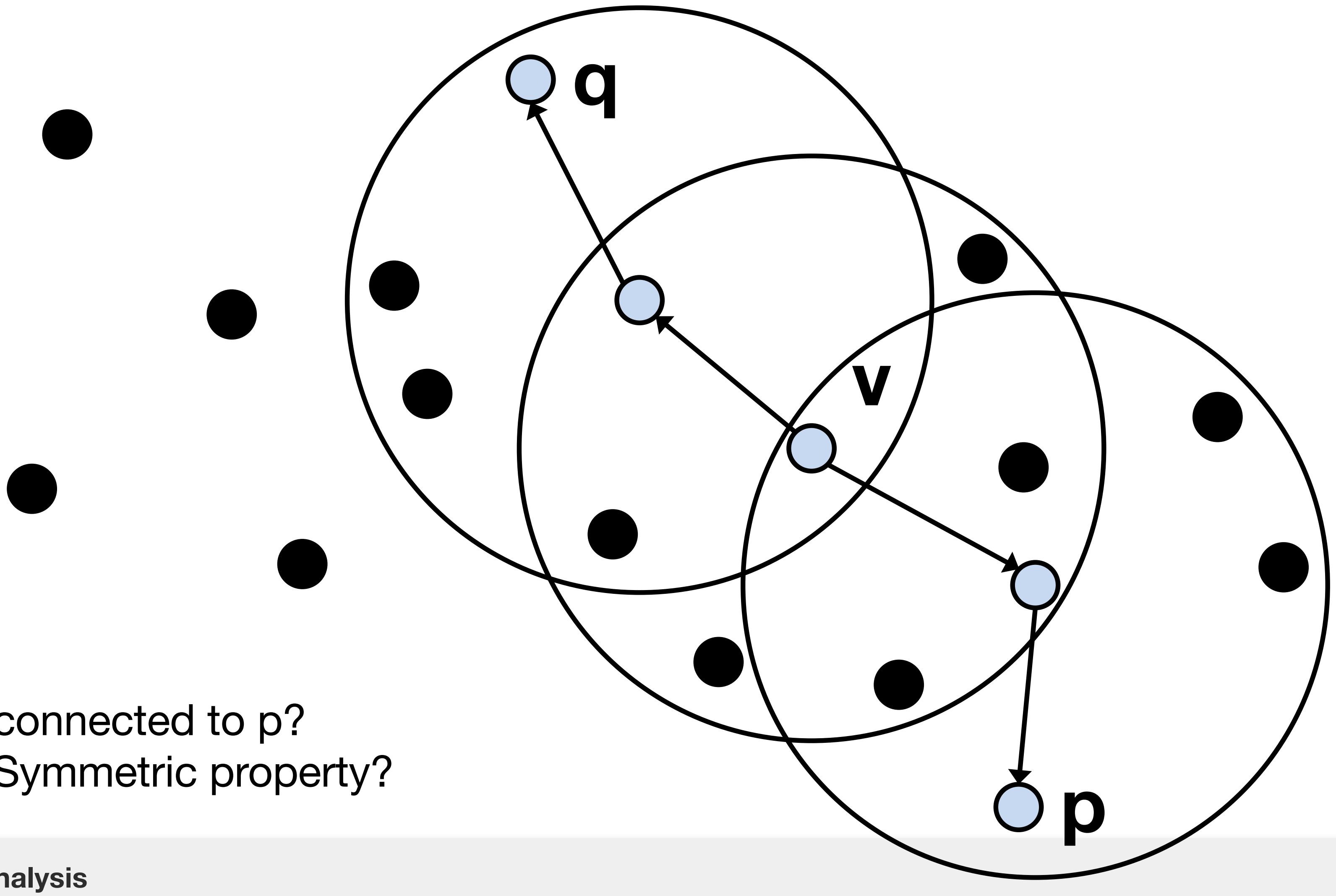


q density-reachable from p?
Reverse? Symmetric property?

Density Connectivity

- A point p is **density-connected** to a point q w.r.t. eps and MinPts if…
  - There is a point v such that p and q are density reachable from v

Example: MinPts=6

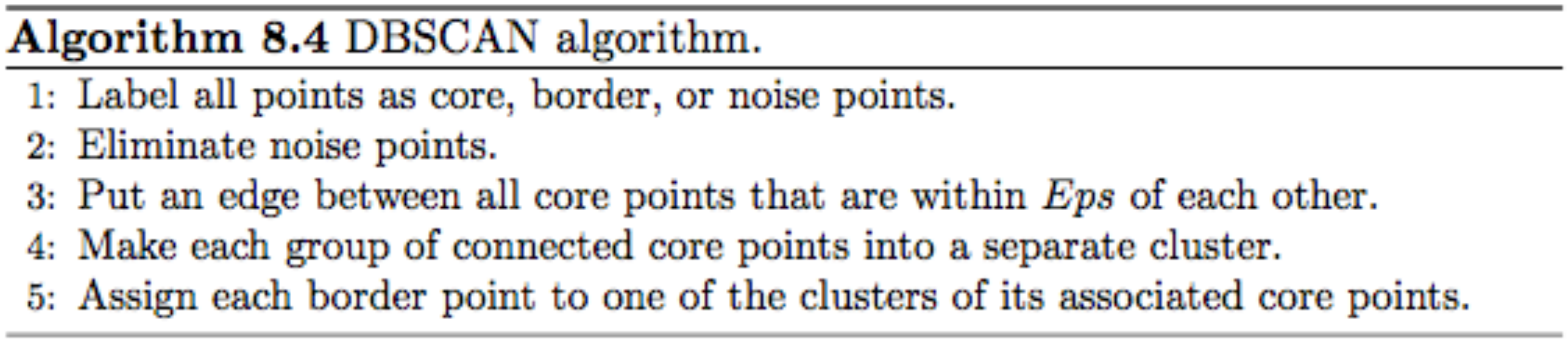


q density-connected to p?
Reverse? Symmetric property?

Cluster (w.r.t. eps/MinPts)

- All points within the cluster are density-connected
- If a point is density-reachable from any point of the cluster, it is part of the cluster (**maximality**)
- All points in a dataset not belonging to any cluster are considered **noise**.

DBSCAN



Algorithm 8.4 DBSCAN algorithm.

```
1: Label all points as core, border, or noise points.
2: Eliminate noise points.
3: Put an edge between all core points that are within *Eps* of each other.
4: Make each group of connected core points into a separate cluster.
5: Assign each border point to one of the clusters of its associated core points.
```

DBSCAN Pseudocode (Wikipedia)

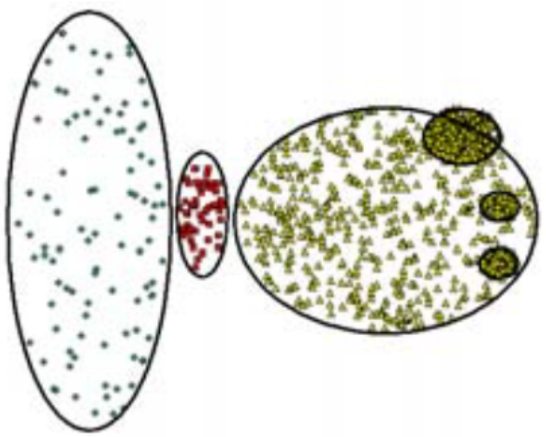
```
DBSCAN(DB, dist, eps, minPts) {
     C = 0 /* Cluster counter */
     for each point P in database DB {
          if label(P) ≠ undefined then continue /* Previously processed in inner loop */
          Neighbors N = RangeQuery(DB, dist, P, eps) /* Find neighbors */
          if |N| < minPts then { /* Density check */
               label(P) = Noise /* Label as Noise */
               continue
          }

          C = C + 1 /* next cluster label */
          label(P) = C /* Label initial point */
          Seed set S = N \ {P} /* Neighbors to expand */
          for each point Q in S { /* Process every seed point */
               if label(Q) = Noise then label(Q) = C /* Change Noise to border point */
               if label(Q) ≠ undefined then continue /* Previously processed */
               label(Q) = C /* Label neighbor */
               Neighbors N = RangeQuery(DB, dist, Q, eps) /* Find neighbors */
               if |N| ≥ minPts then { /* Density check */
                    S = S ∪ N /* Add new neighbors to seed set */
               }
          }
     }
}
```

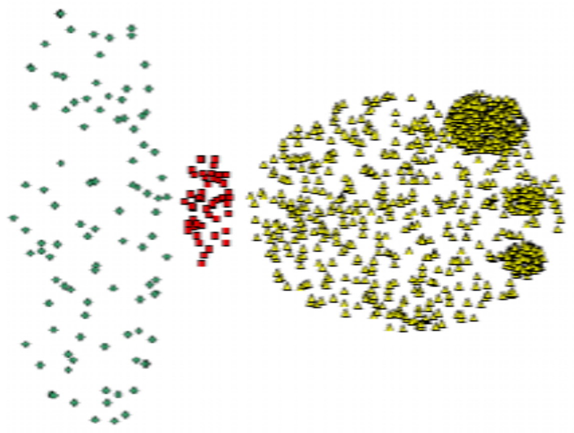
Computational Complexity

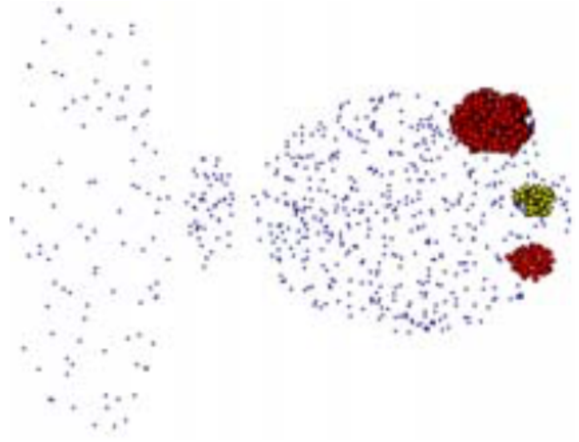
- Time: $\mathcal{O}(N^2)$ naïvely
  - O(NlogN) if using a spatial index for neighbor queries (works for low dimensions)
- Space: $\mathcal{O}(N)$

eps/MinPts

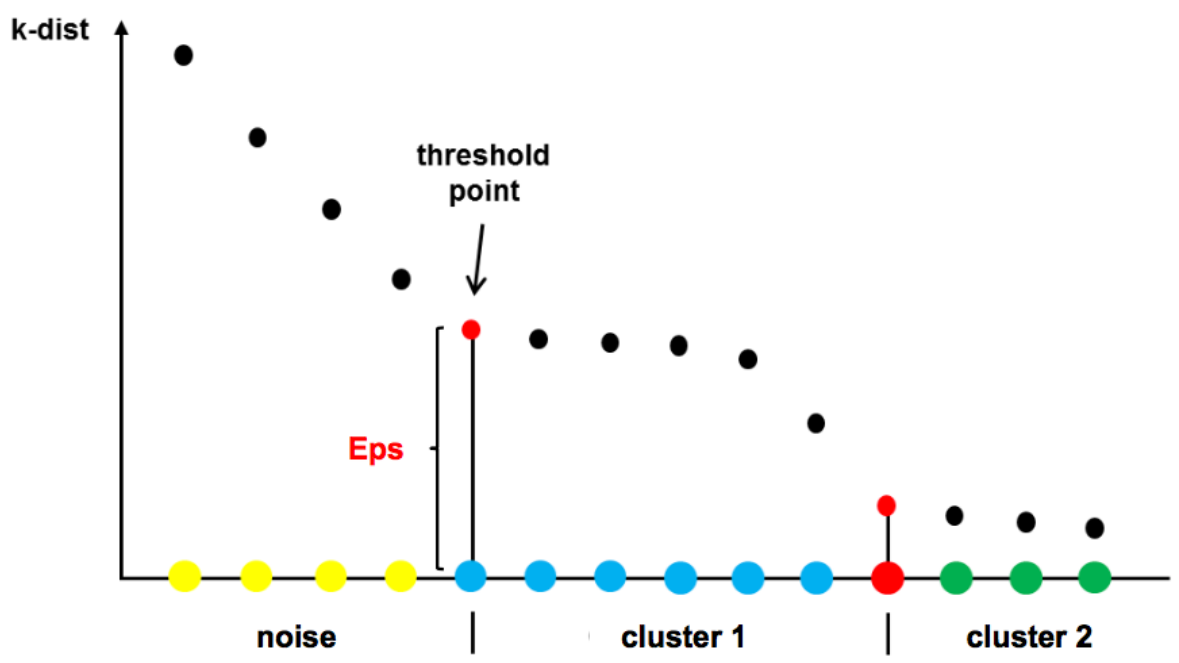
- Parameters must be chosen precisely
  - RoT: D < MinPts < 2D


Ground Truth


MinPts=4, Eps=9.92

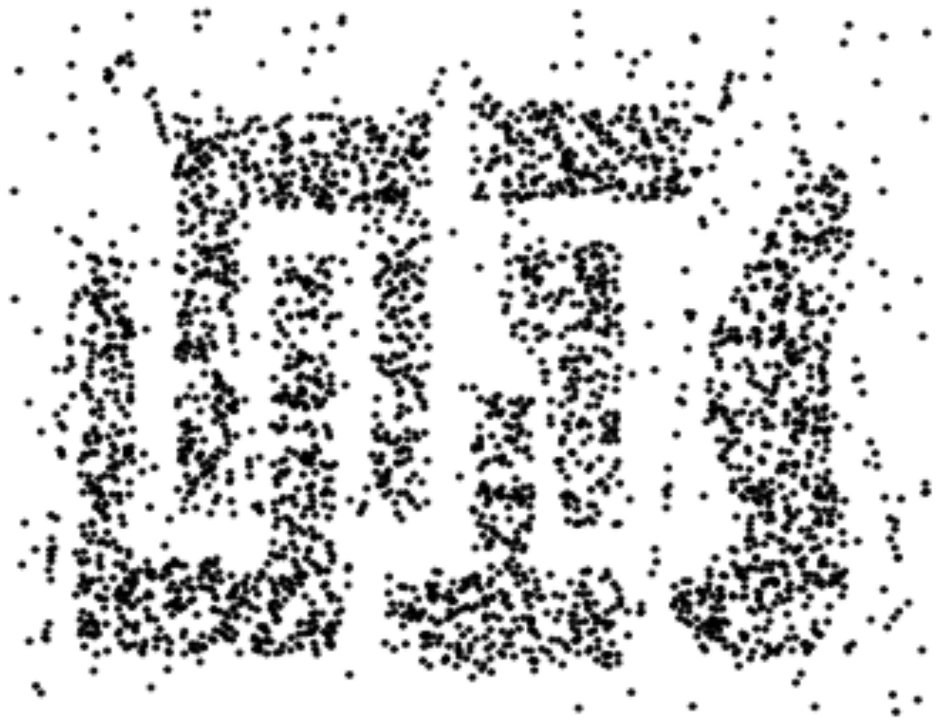

MinPts=4, Eps=9.75

Find the Knee

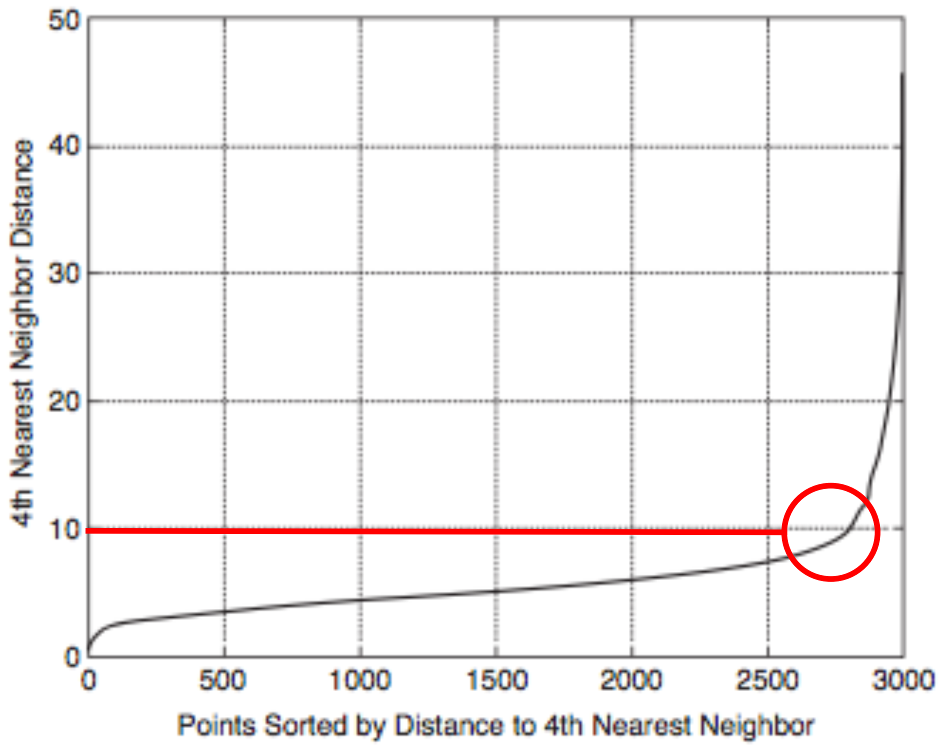


- Get distance from each point to kth nearest neighbor (MaxPts)
- Sort, plot distance vs points, find knee (eps)

Example: Input

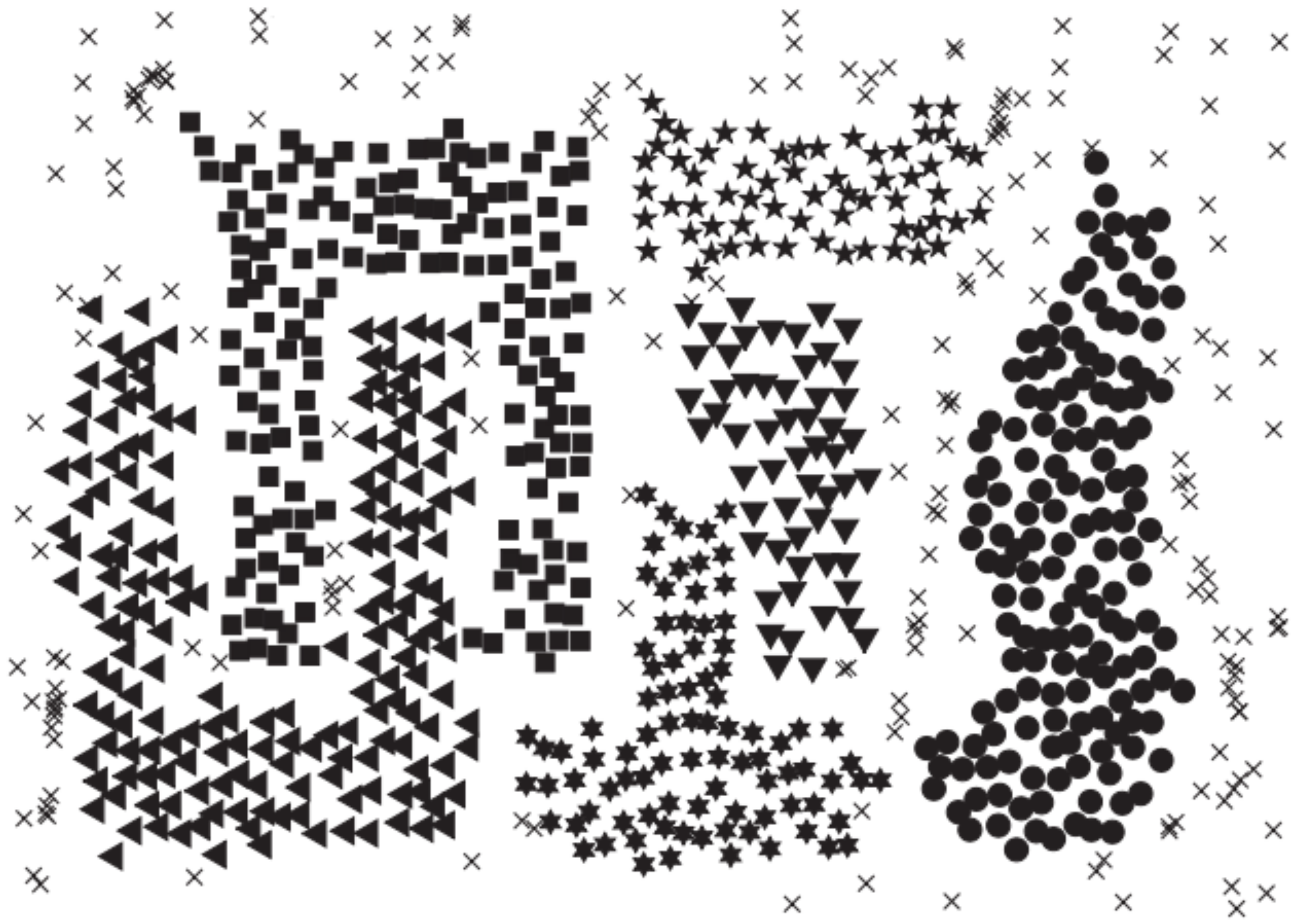


Example: 4-NN

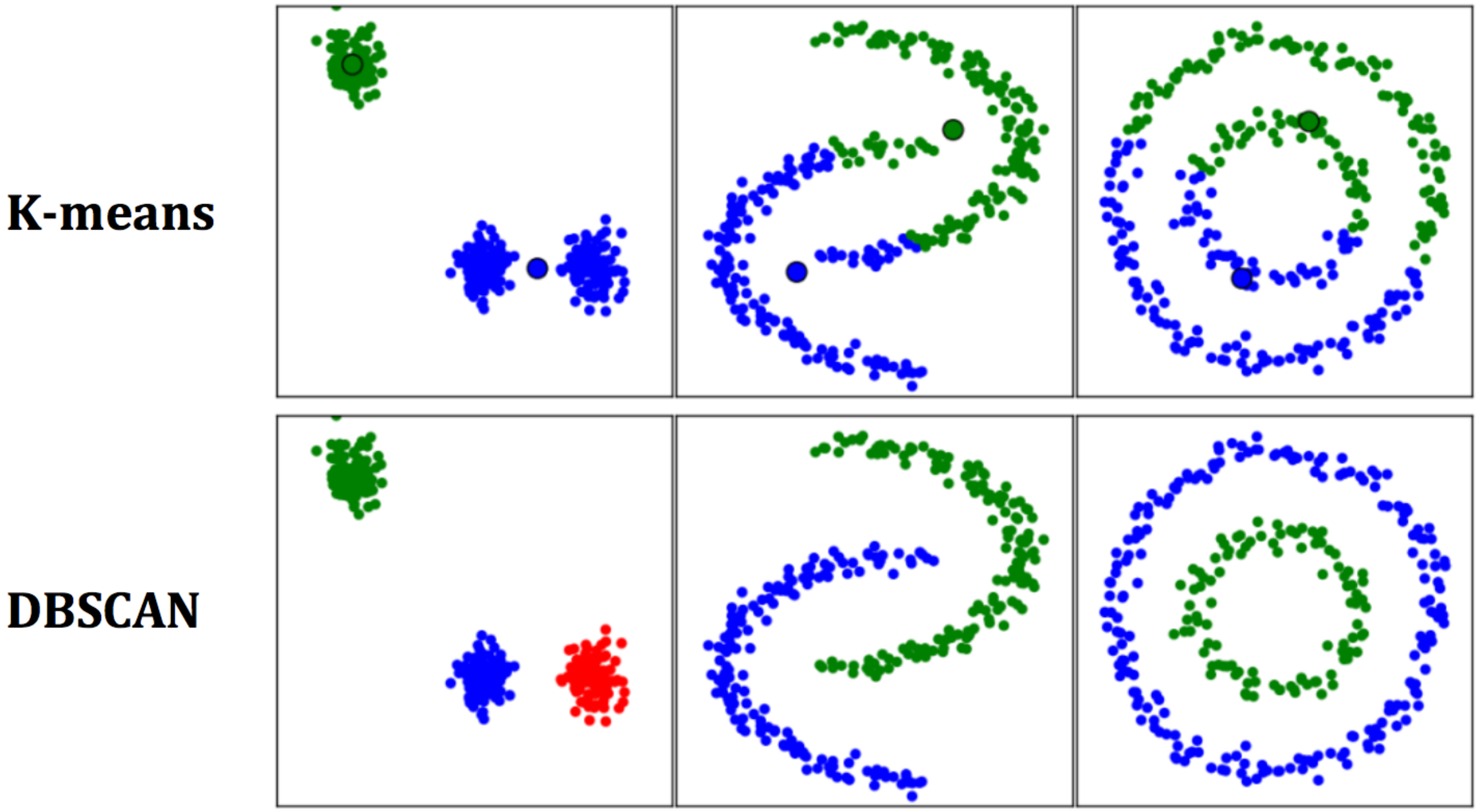


4th Nearest Neighbor Distance
0
10
20
30
40
50
0
500
1000
1500
2000
2500
3000
Points Sorted by Distance to 4th Nearest Neighbor

MaxPts=4, eps=10

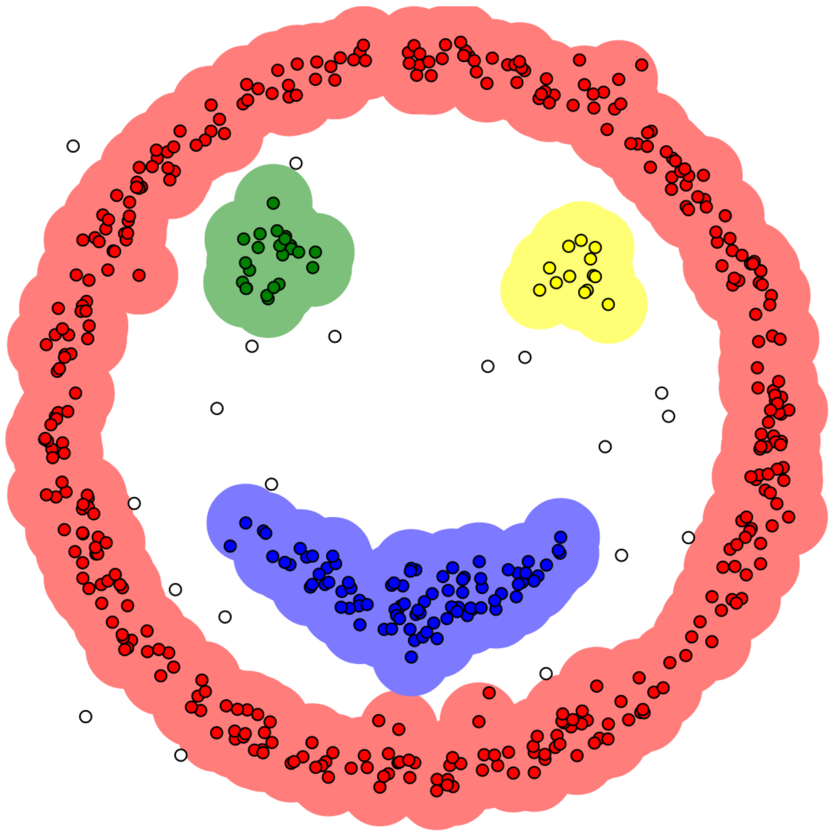


K-Means vs DBSCAN



Animation Time!

https://www.naftaliharris.com/blog/visualizing-dbscan-clustering/

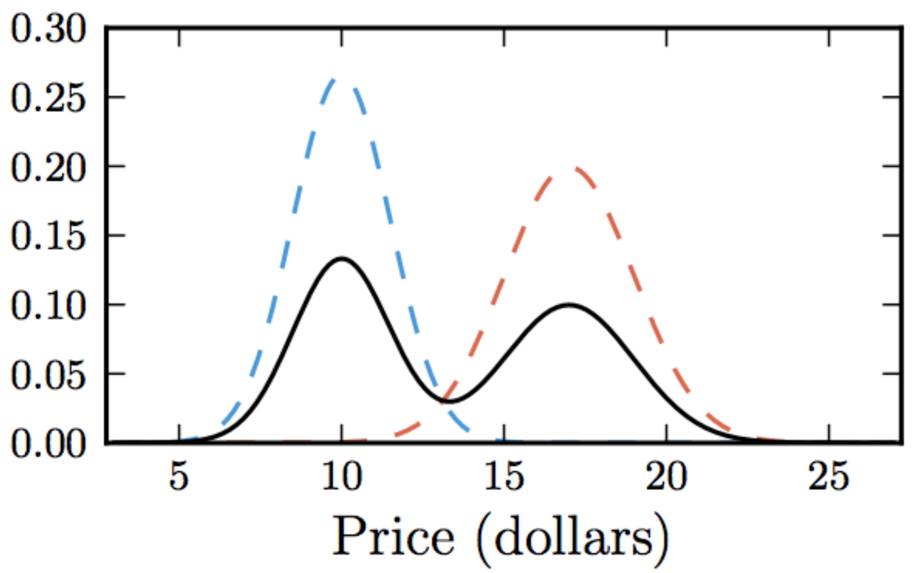


Mixture Models – Why!?

- If we have a dataset, and can reasonably assume its distribution (e.g. Gaussian), easy to perform many useful operations…
  - Make statements about the data source
  - Learn parameters, e.g. mean/(co-)variance
  - Generate new points
  - Make statements about common/uncommon points (possibly part of pipeline, e.g. classification)
- But often we aren't so lucky…

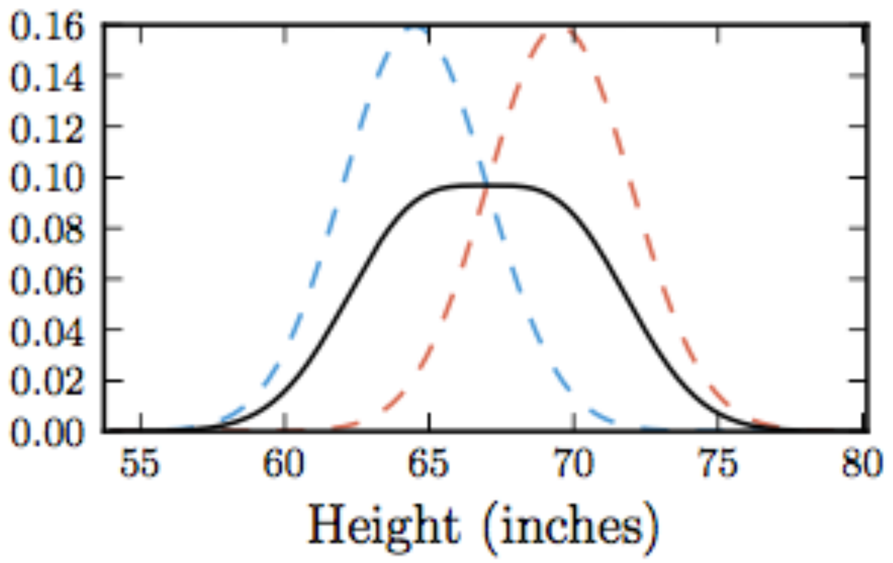
Gaussian Mixture Models (e.g. 1)

- The price of a paperback book is normally distributed with mean \$10, std \$1
- The price of a hardcover has mean \$17, std \$1.50
- Price of a book?



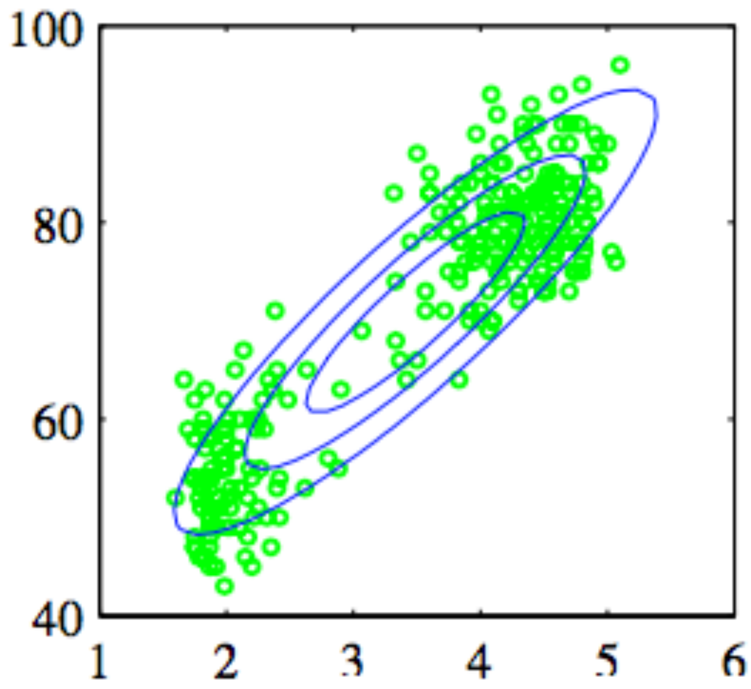
Gaussian Mixture Models (e.g. 2)

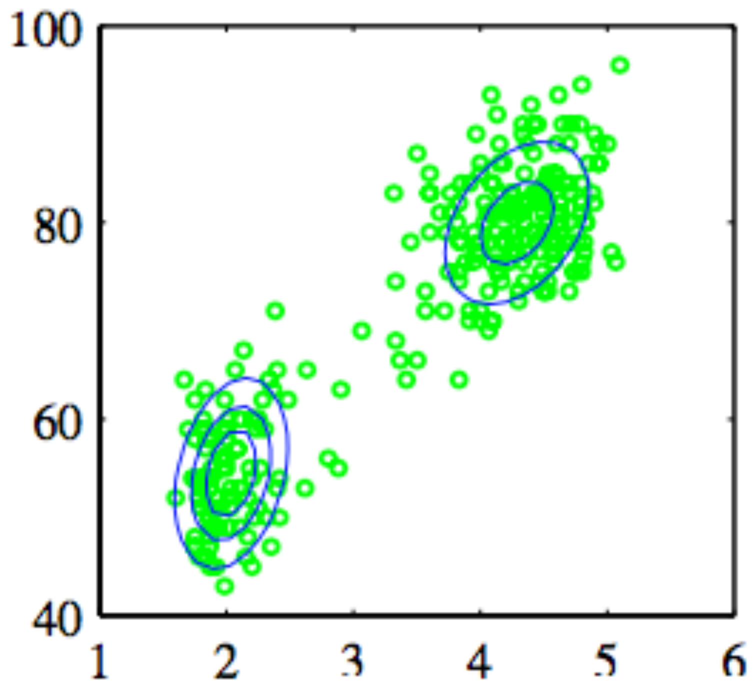
- The height of a randomly chosen man is normally distributed with mean 69.5”, std 2.5”
- The height of a woman is mean 64.5”, std 2.5”
- Height of a person?



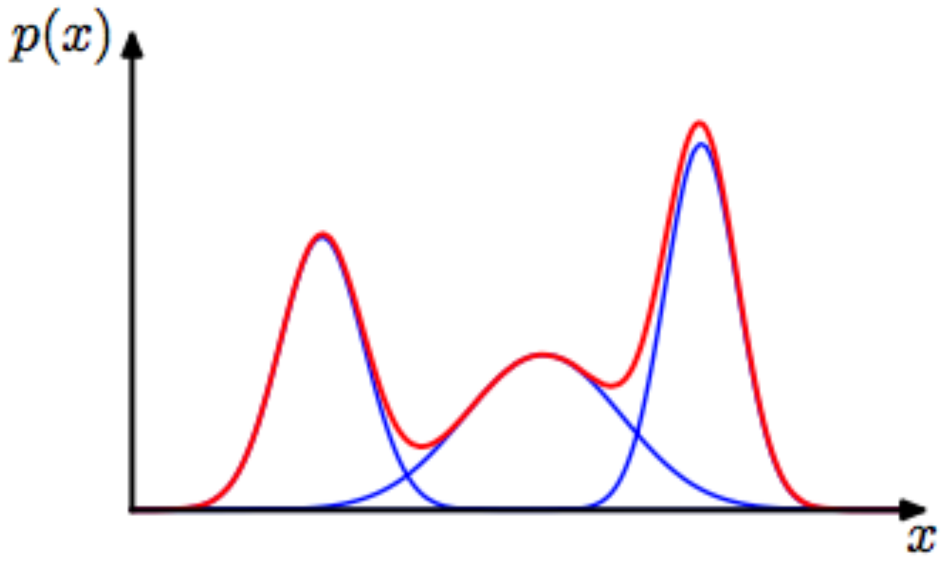
Gaussian Mixture Models (e.g. 3)

- Old Faithful: time till next eruption vs eruption time

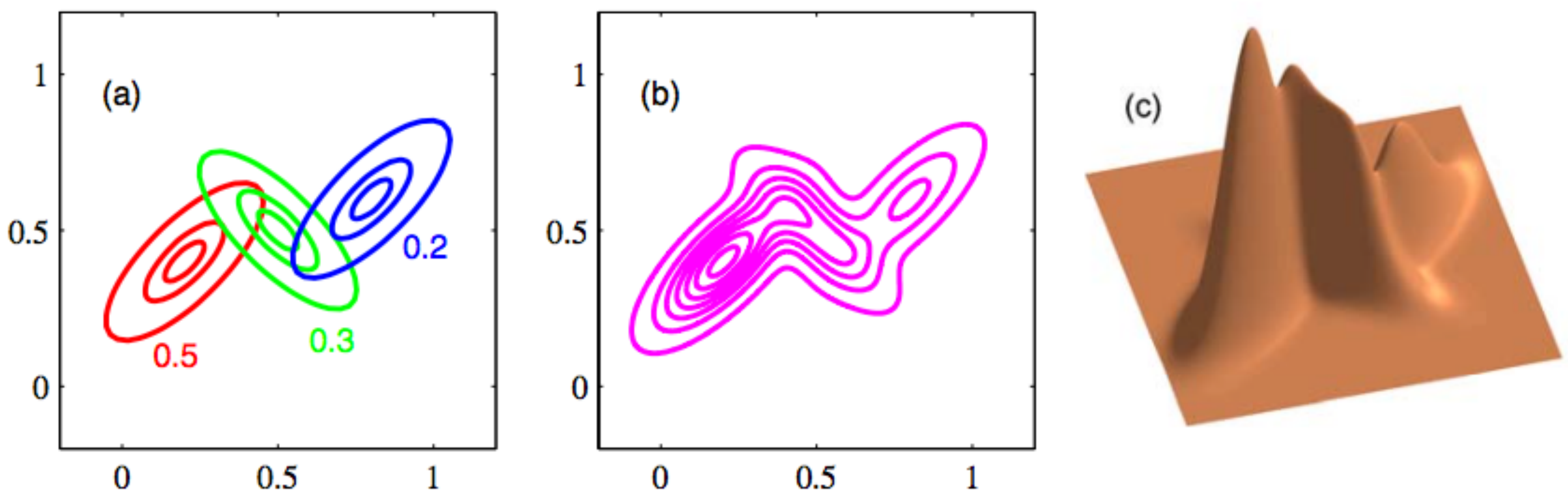




Not Limited to 2 Distributions (1)



Not Limited to 2 Distributions (2)



So How Do We Model?

- Basic idea: we assume a "mix" of a finite number of known distributions (Gaussians for now)
- Each distribution has its own parameters: for GMMs, mean (μ) & (co)variance (Σ) as usual
- We ALSO add a "mixing" parameter (π_k), per distribution, that accounts for the probability of drawing from that distribution
  – Example: World Bank 2016
    - p(Female) = 49.558%
    - p(Female|USA) = 50.5%

Gaussian Mixture Model

- So now we can express the probability of a point in the superposition of the individual distributions

$$p(\boldsymbol{x}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{x} | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

$$\sum_{k=1}^{K} \pi_k = 1 \qquad 0 \leq \pi_k \leq 1$$

Quick Check

- If you knew π, μ, Σ: could you sample from this distribution?

$$p(\boldsymbol{x}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

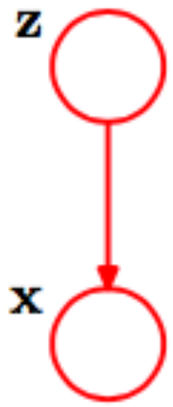
Quick Check

- If you knew π, μ, Σ: could you sample from this distribution?
- Yes – it's **generative** (vs **discriminative**)
- How? HW2 part 3 :)

$$p(\boldsymbol{x}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

Some Observations

- Given the constraints on π, it can be thought of as the **prior** probability of selecting a Gaussian
- And the normal is simply the **likelihood** of drawing the point given a Gaussian has been chosen



z_k is one-hot

$p(z_k=1)= \pi_k$

z is a **latent** variable

Posterior = Responsibility

$$\gamma(z_k) \equiv p(z_k = 1|\boldsymbol{x})$$

Posterior = Responsibility

$$\gamma(z_k) \equiv p(z_k = 1|\boldsymbol{x}) = \frac{p(z_k = 1)p(\boldsymbol{x}|z_k = 1)}{\sum_{j=1}^{K} p(z_j = 1)p(\boldsymbol{x}|z_j = 1)}$$

Posterior = Responsibility

$$\gamma(z_k) \equiv p(z_k = 1|\boldsymbol{x}) = \frac{p(z_k = 1)p(\boldsymbol{x}|z_k = 1)}{\sum_{j=1}^{K} p(z_j = 1)p(\boldsymbol{x}|z_j = 1)}$$

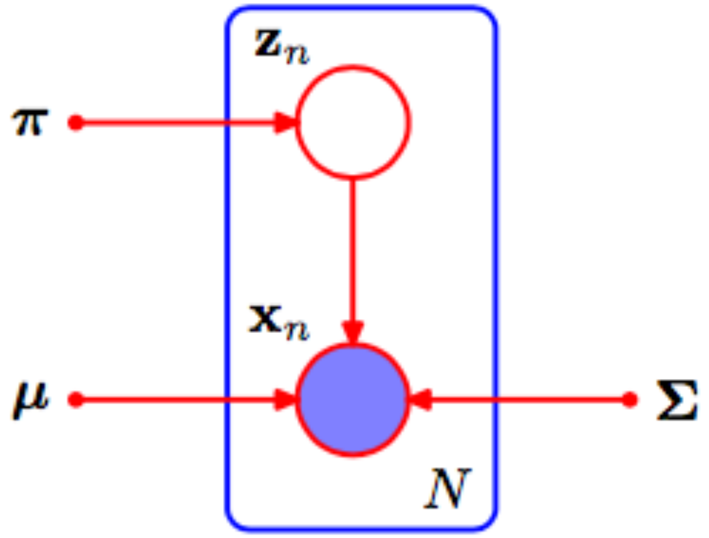
$$= \frac{\pi_k \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}$$

Now the Core Clustering Question

- Given a set of N observations

  $\{\mathbf{x}_1, \ldots \mathbf{x}_N\}$

- What parameter values (π, μ, Σ) best explain the data?



Objective

- Maximize the following function – the likelihood of seeing the dataset given the selected model parameters

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

Aside: Why Log-Likelihood?

Often used for practical reasons…

- Within useful ranges of values, relative ordering maintained
  - $p(a) > p(b) \Rightarrow \ln(p(a)) > \ln(p(b))$
- Easier math
  - Easy derivative
  - Combines well with exponential (e.g. $\ln(e^x)=x$)
  - Products become sums
- Avoids underflow

Quick Check

- Hierarchical or Partitional?
- Exclusive, Overlapping, Fuzzy?
- Complete or Partial?
- Centroid, Hierarchical, Density, Distribution?

Quick Check

- Hierarchical or **Partitional**?
- Exclusive, Overlapping, **Fuzzy**?
- **Complete** or Partial?
- Centroid, Hierarchical, Density, **Distribution**?

Game Plan

- Take the partial w.r.t. each parameter, set equal to 0, solve?
  - Not going to happen…
  - Possibility: gradient ascent
  - For now: EM
- EM for Gaussian Mixture Modeling
  - Initialize parameters (π, μ, Σ)
  - Loop till convergence (??)
    - E-Step: fix parameters, evaluate responsibility
    - M-Step: fix responsibility, optimize parameters

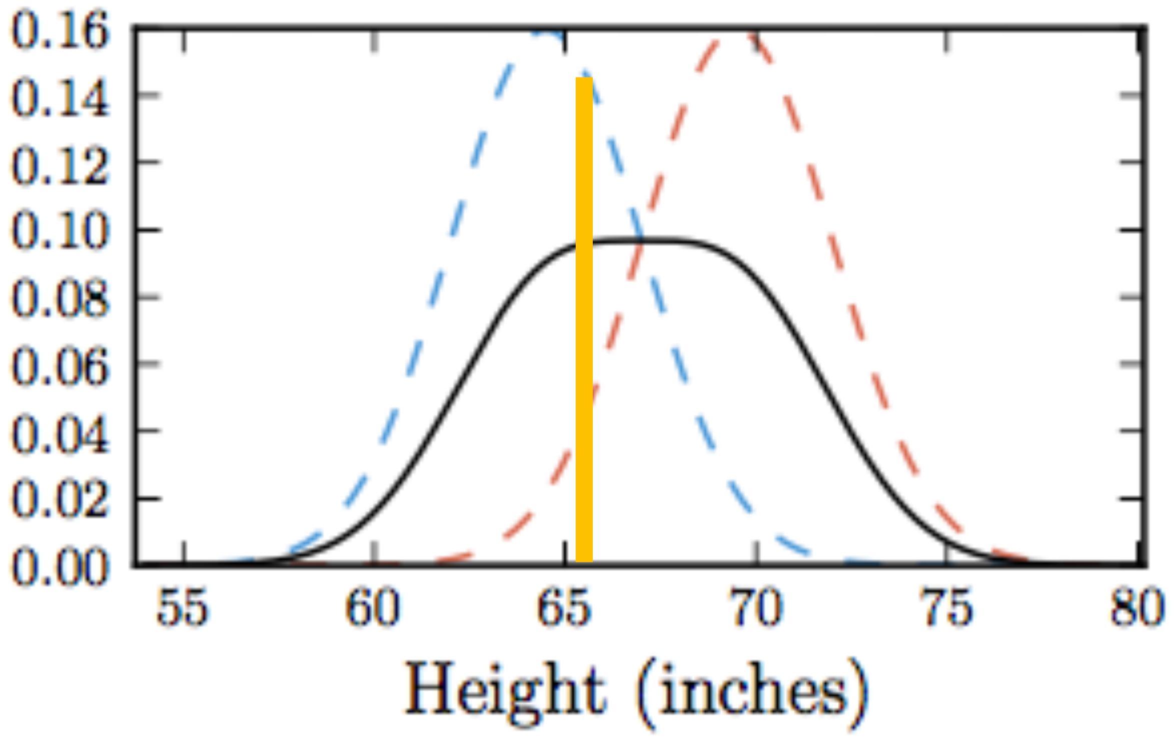
Parameter Initialization

- π_k = 1/K
- μ = Forgy
- Σ = global variance

Other possibilities exist (e.g. splitting), might attempt multiple and use lowest initial log-likelihood

E-Step (1)

- For each point, evaluate responsibility with respect to each Gaussian; normalize
- For example, with $\pi = (0.5, 0.5)$ …



$N(\mathbf{x}; \boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1) \sim 0.58$

$N(\mathbf{x}; \boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2) \sim 0.07$

$\gamma(\mathbf{z}_1) \sim 0.89$

$\gamma(\mathbf{z}_2) \sim 0.11$

E-Step (2)

Evaluate for all (n) data points x (k) models…

$$\gamma(z_{nk}) = \frac{\pi_k \mathcal{N}(\boldsymbol{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\boldsymbol{x}_n | \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}$$

M-Step (1)

- Holding the responsibilities fixed, now maximize each of the parameters
  - Derivation is quite similar to K-Means, but π requires introduction of a Langrange multiplier to enforce summation to 1
- All terms reference Nk, which can be thought of as the effective number of points assigned to a cluster

$$N_k = \sum_{n=1}^{N} \gamma(z_{nk})$$

M-Step (2)

$$\pi_k^{\text{new}} = \frac{N_k}{N}$$

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

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

Convergence Criterion

- Unlike K-Means, we don't have crisp membership variables that we can monitor for discrete changes

- Instead, commonly…
  - Compute log-likelihood after each iteration, stop when change drops below ε
  - Could also have a hold-out set, monitor change in log-likelihood

Example Run: Setup

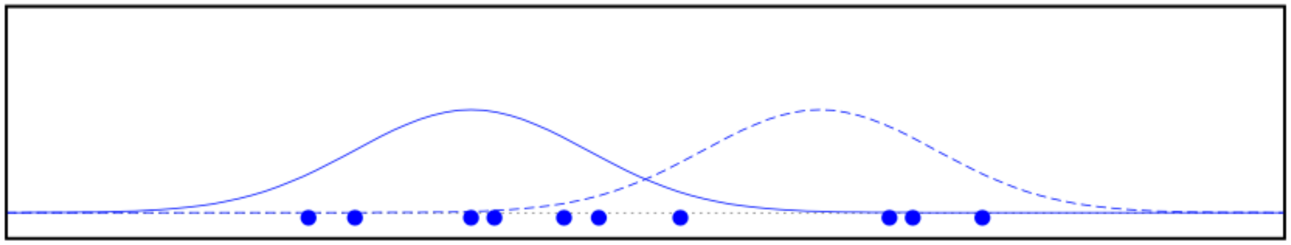
- Two-component univariate GMM; 10 data points.
- The data: $x_1, \ldots, x_{10}$

$$8.4, 7.6, 4.2, 2.6, 5.1, 4.0, 7.8, 3.0, 4.8, 5.8$$

- Initial parameter values:

p_1	μ_1	σ_1^2	p_2	μ_2	σ_2^2
0.5	4	1	0.5	7	1

- Training data; densities of initial Gaussians.



Credit: Michael Picheny et al., Watson Group @ IBM

Example Run: E-Step

x_i	$p_1 \cdot \mathcal{N}_1$	$p_2 \cdot \mathcal{N}_2$	$P(x_i)$	$\tilde{P}(1\|x_i)$	$\tilde{P}(2\|x_i)$
8.4	0.0000	0.0749	0.0749	0.000	1.000
7.6	0.0003	0.1666	0.1669	0.002	0.998
4.2	0.1955	0.0040	0.1995	0.980	0.020
2.6	0.0749	0.0000	0.0749	1.000	0.000
5.1	0.1089	0.0328	0.1417	0.769	0.231
4.0	0.1995	0.0022	0.2017	0.989	0.011
7.8	0.0001	0.1448	0.1450	0.001	0.999
3.0	0.1210	0.0001	0.1211	0.999	0.001
4.8	0.1448	0.0177	0.1626	0.891	0.109
5.8	0.0395	0.0971	0.1366	0.289	0.711

$$\tilde{P}(h|x_i) = \frac{P(h, x_i)}{\sum_h P(h, x_i)} = \frac{p_h \cdot \mathcal{N}_h}{P(x_i)} \qquad h \in \{1, 2\}$$

Credit: Michael Picheny et al., Watson Group @ IBM

Example Run: M-Step

$$\mu = \frac{1}{N}\sum_{i=1}^{N} x_i \quad \Rightarrow \quad \mu_h = \frac{1}{\sum_i \tilde{P}(h|x_i)} \sum_{i=1}^{N} \tilde{P}(h|x_i) x_i$$

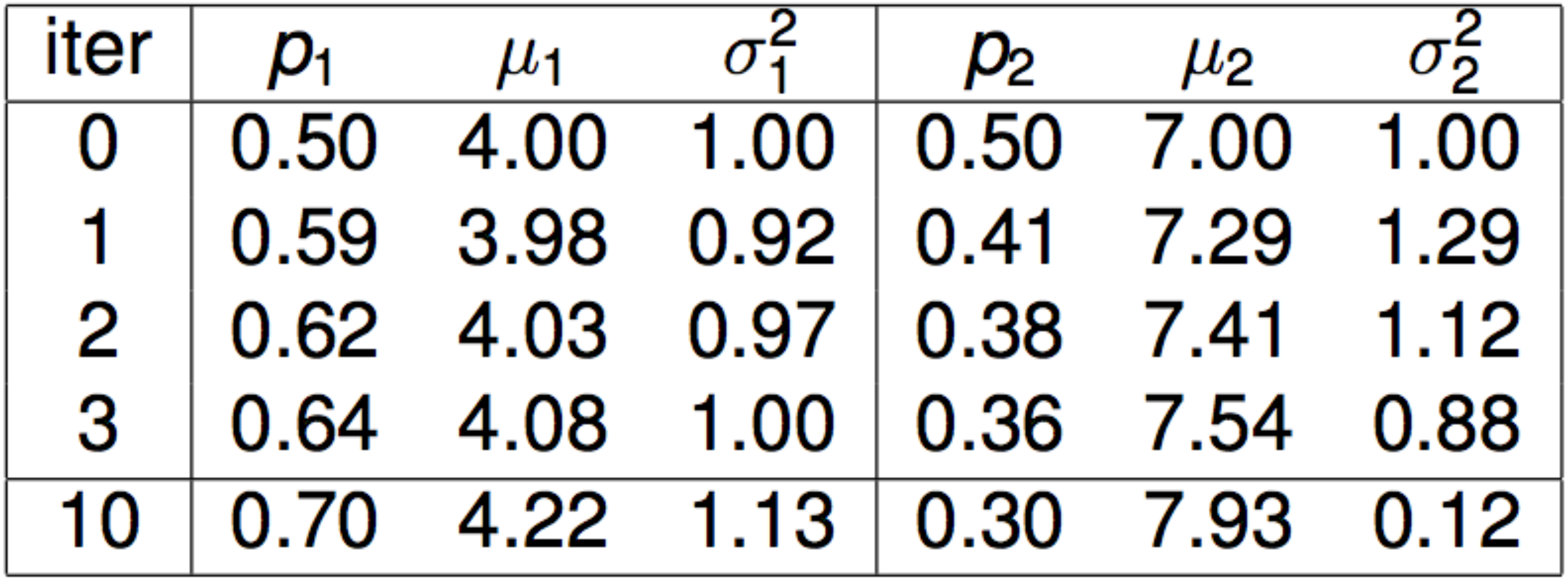
$$\mu_1 = \frac{1}{0.000 + 0.002 + 0.980 + \cdots} \times (0.000 \times 8.4 + 0.002 \times 7.6 + 0.980 \times 4.2 + \cdots)$$

$$= 3.98$$

$$p_1 = \frac{0.000 + 0.002 + 0.980 + \cdots}{10} = 0.59$$

Credit: Michael Picheny et al., Watson Group @ IBM

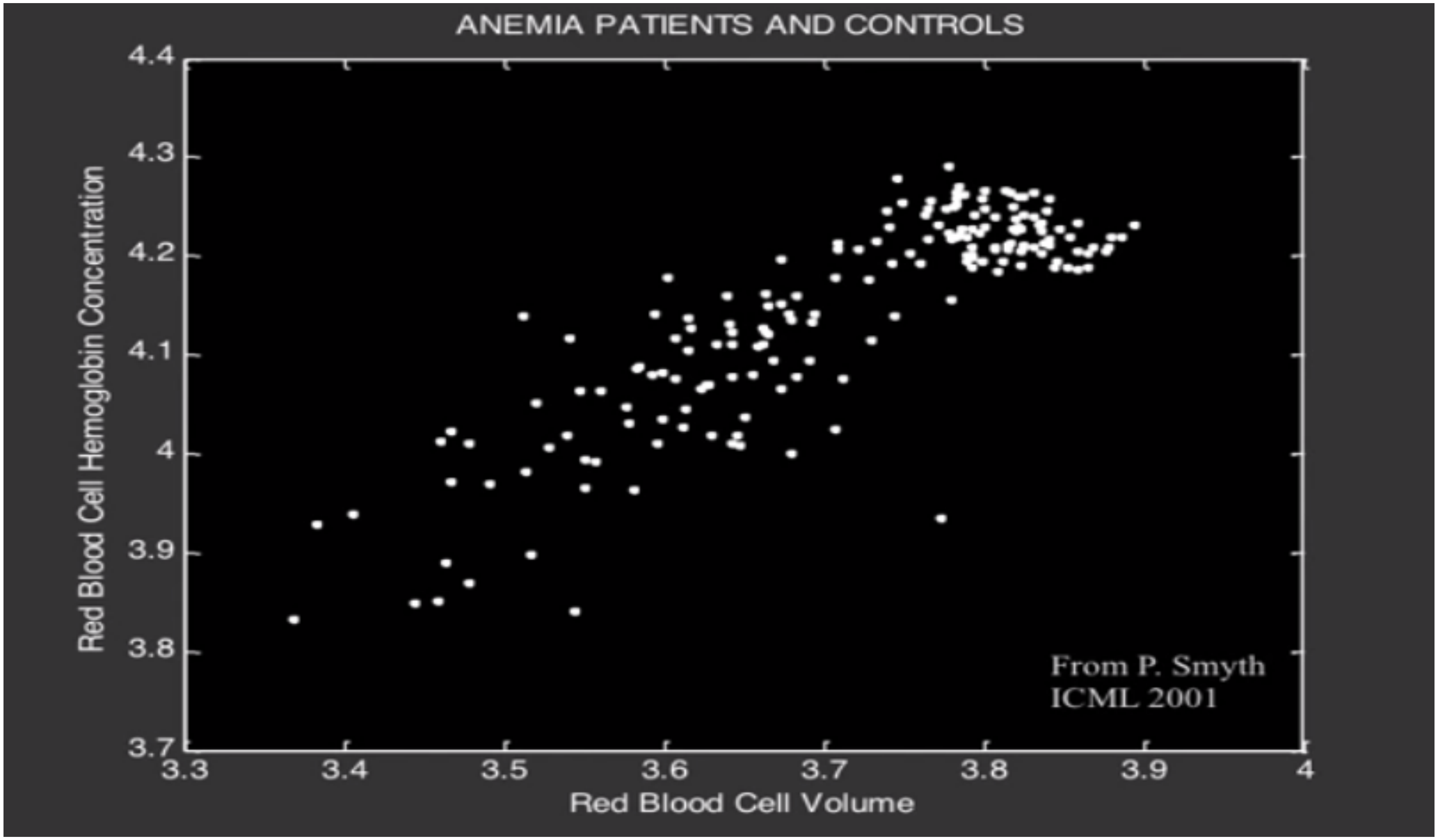
Example Run: Results



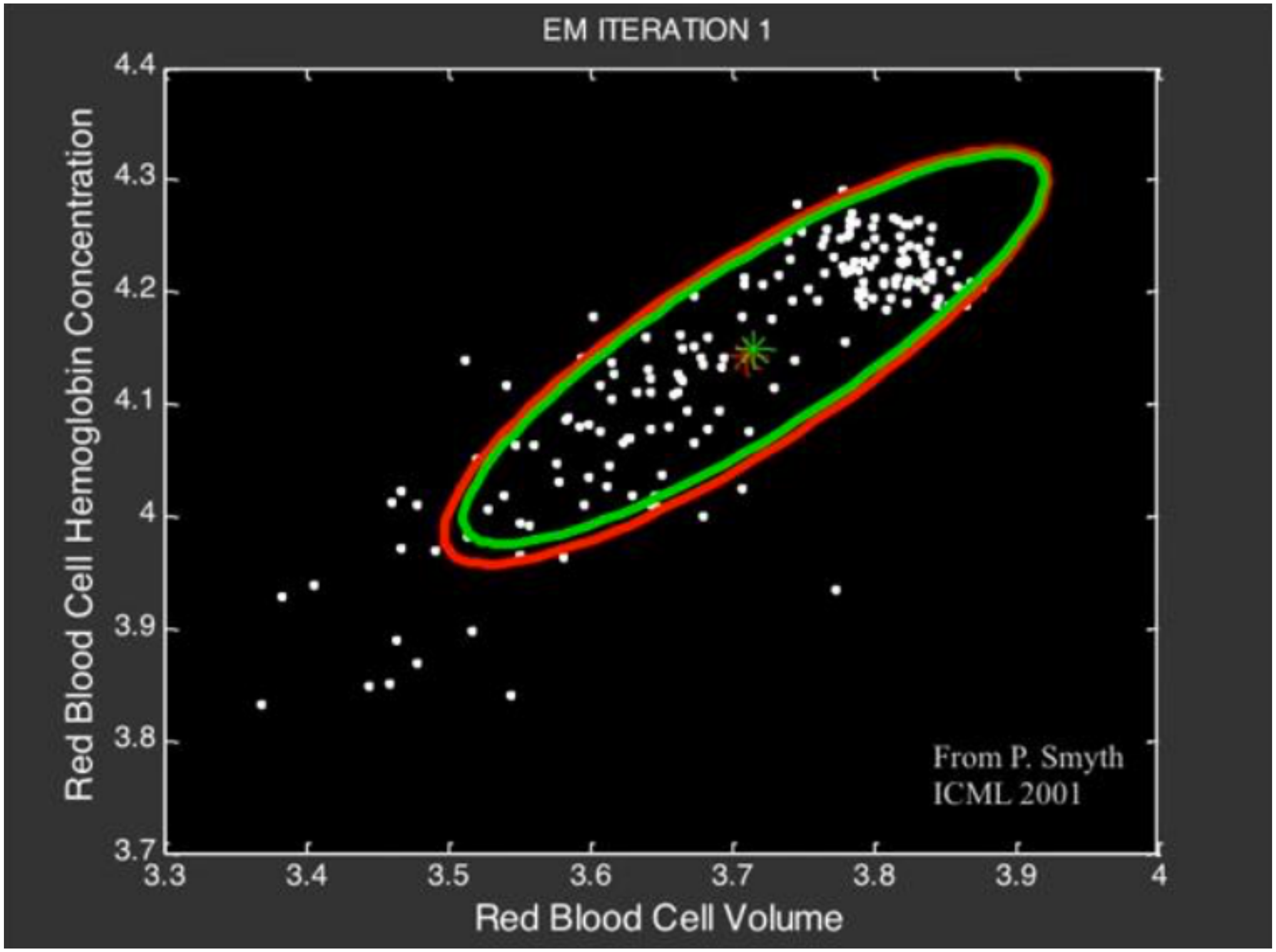
iter	p_1	μ_1	σ_1^2	p_2	μ_2	σ_2^2
0	0.50	4.00	1.00	0.50	7.00	1.00
1	0.59	3.98	0.92	0.41	7.29	1.29
2	0.62	4.03	0.97	0.38	7.41	1.12
3	0.64	4.08	1.00	0.36	7.54	0.88
10	0.70	4.22	1.13	0.30	7.93	0.12

Credit: Michael Picheny et al., Watson Group @ IBM

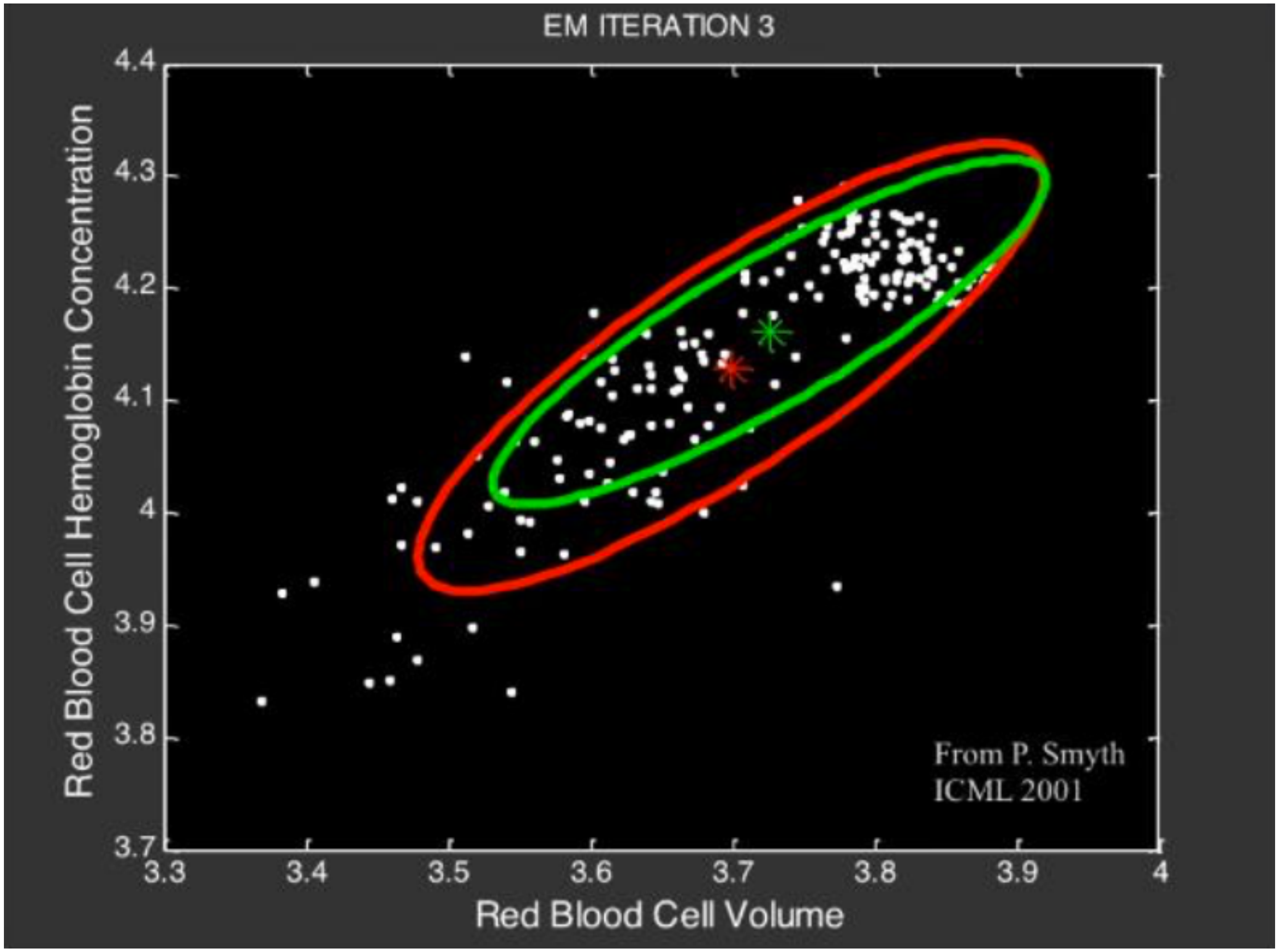
Example Data



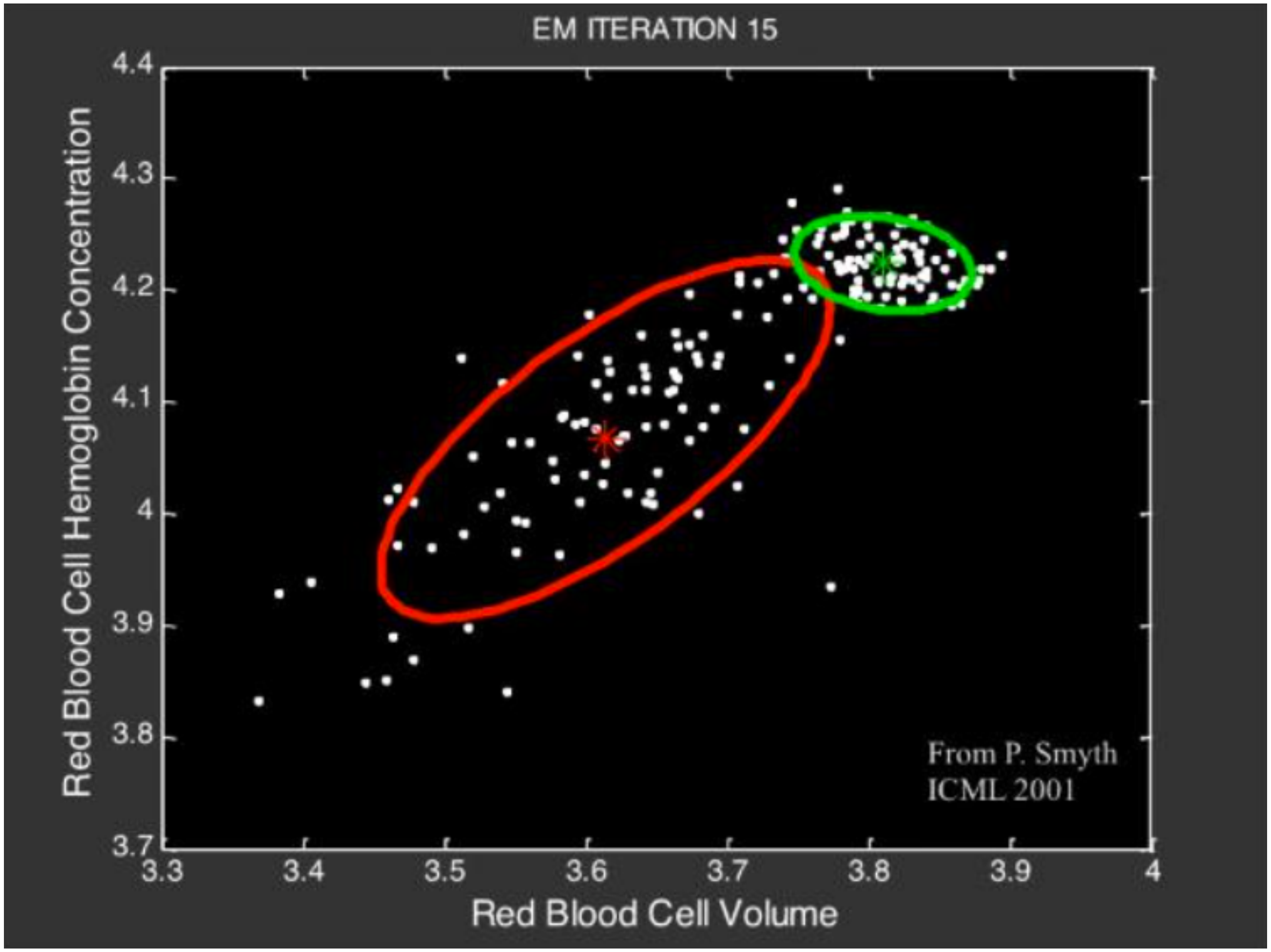
Iteration 1


EM ITERATION 1
Red Blood Cell Hemoglobin Concentration
4.4
4.3
4.2
4.1
4
3.9
3.8
3.7
3.3
3.4
3.5
3.6
3.7
3.8
3.9
4
Red Blood Cell Volume
From P. Smyth
ICML 2001

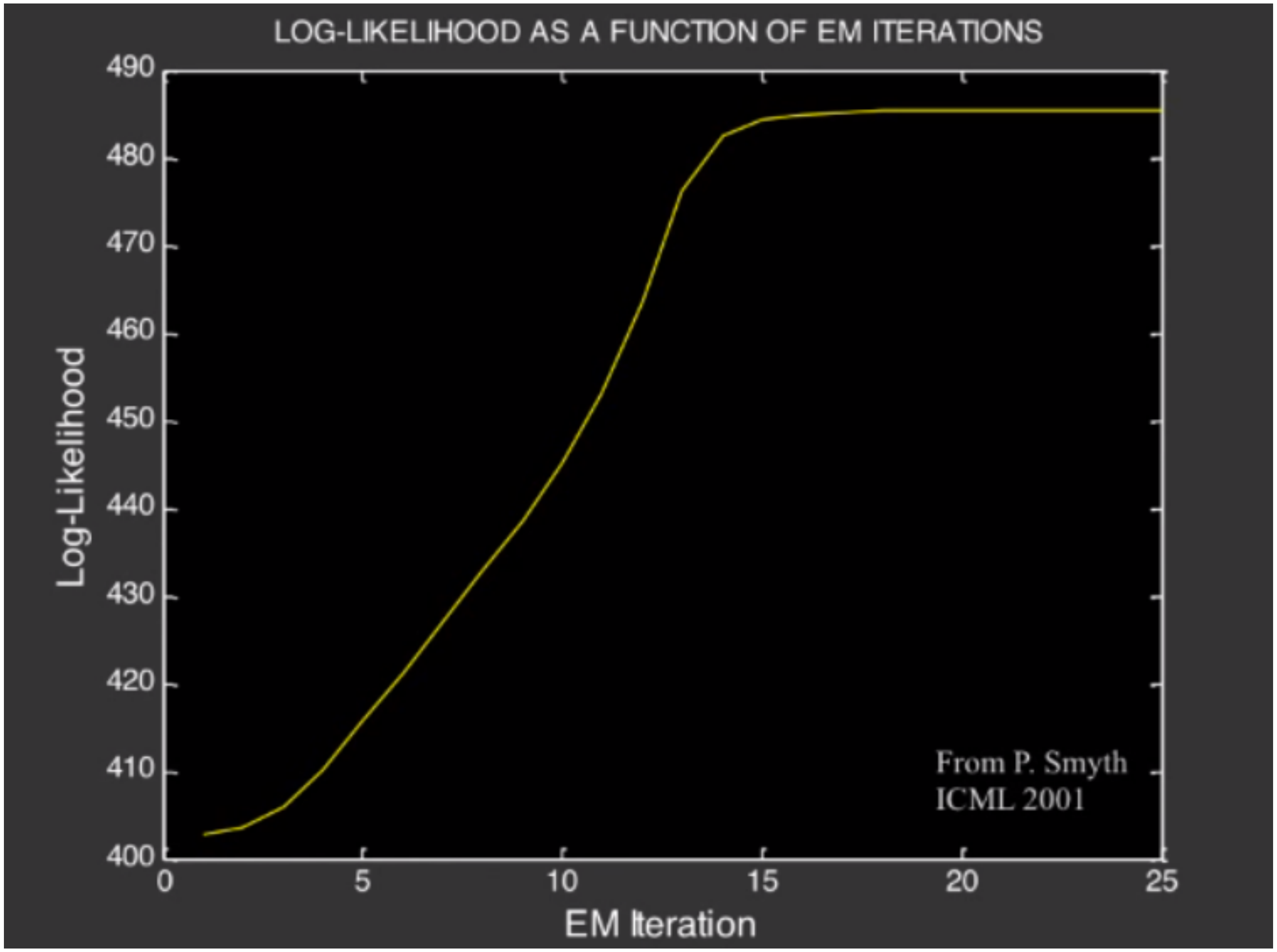
Iteration 3


EM ITERATION 3
Red Blood Cell Hemoglobin Concentration
Red Blood Cell Volume
4.4
4.3
4.2
4.1
4
3.9
3.8
3.7
3.3
3.4
3.5
3.6
3.7
3.8
3.9
4
From P. Smyth
ICML 2001

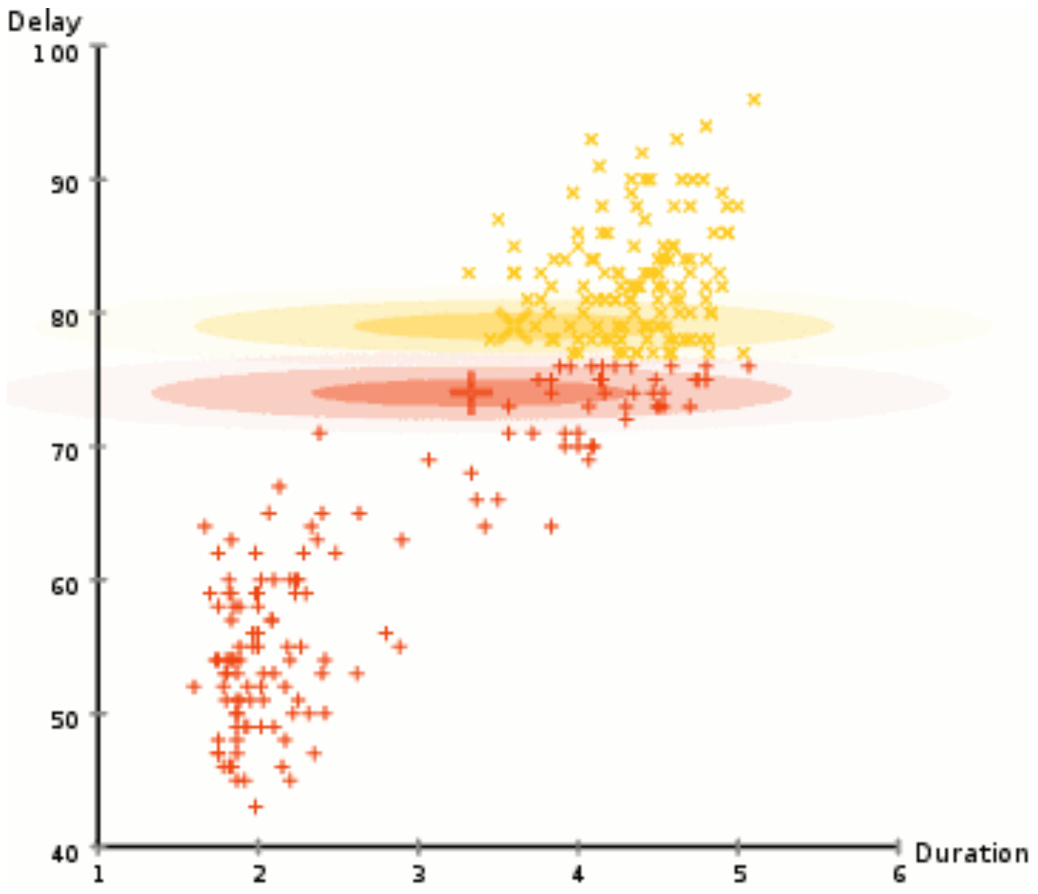
Iteration 15



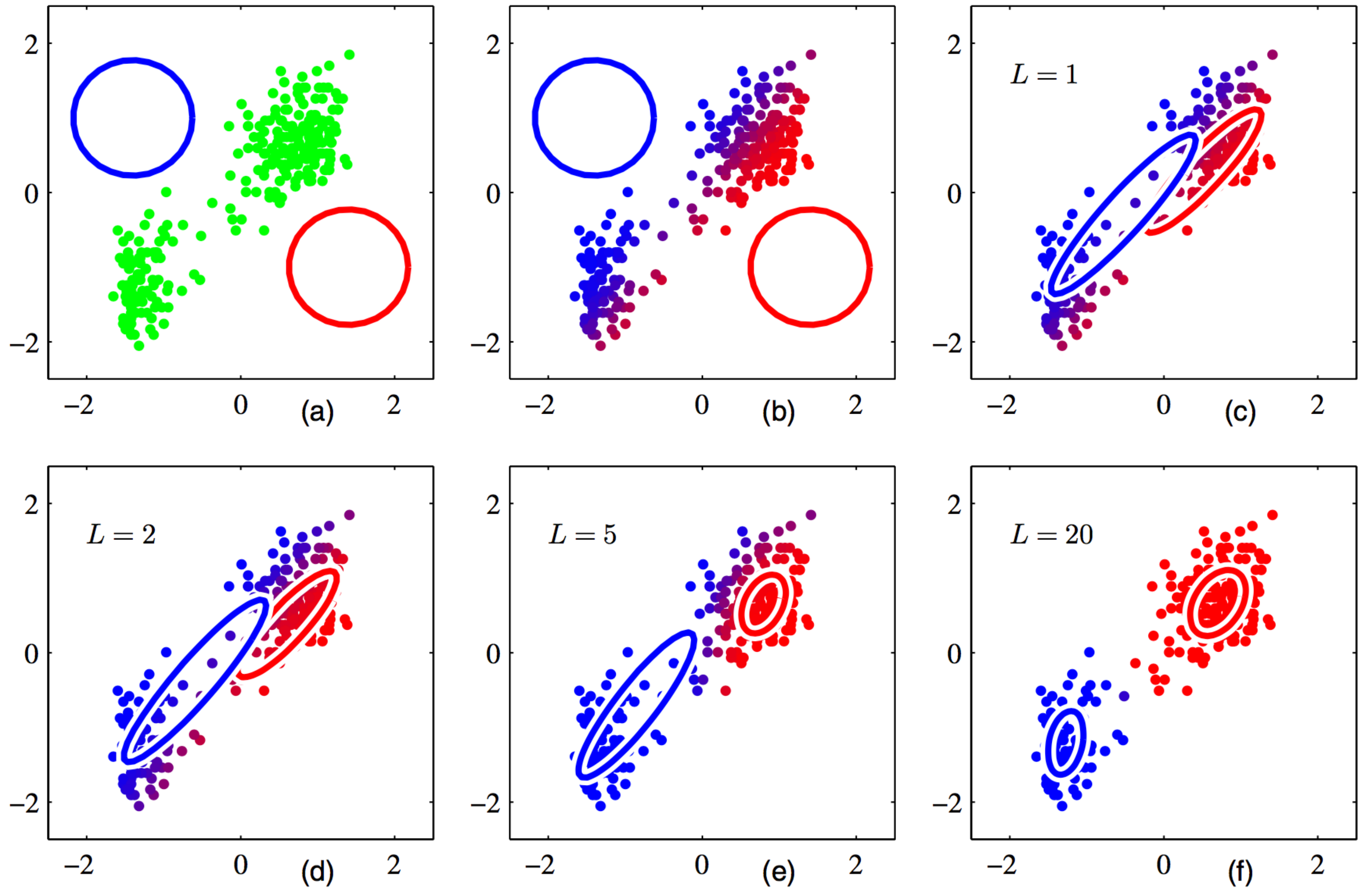
Convergence via Log-Likelihood



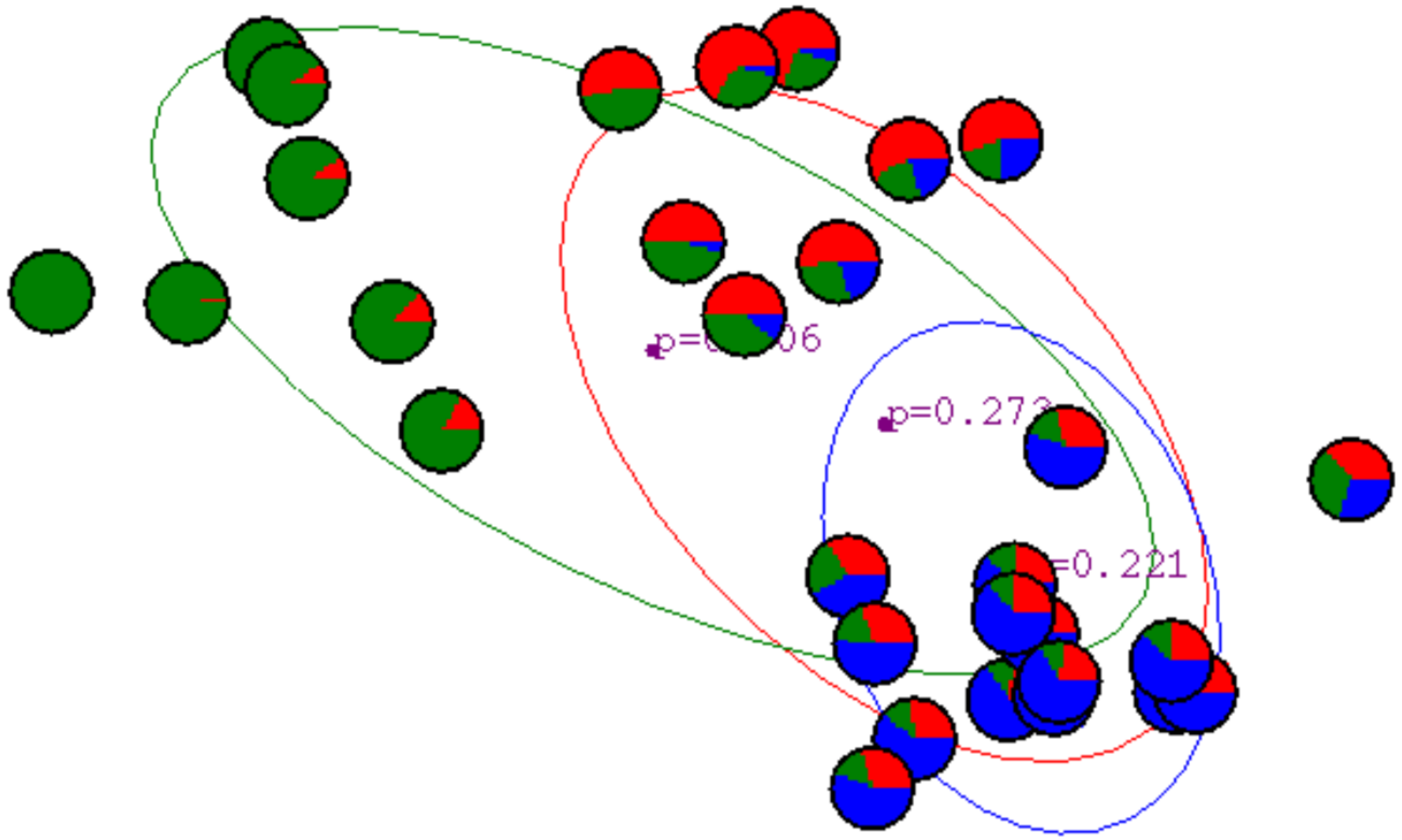
Old Faithful (1)



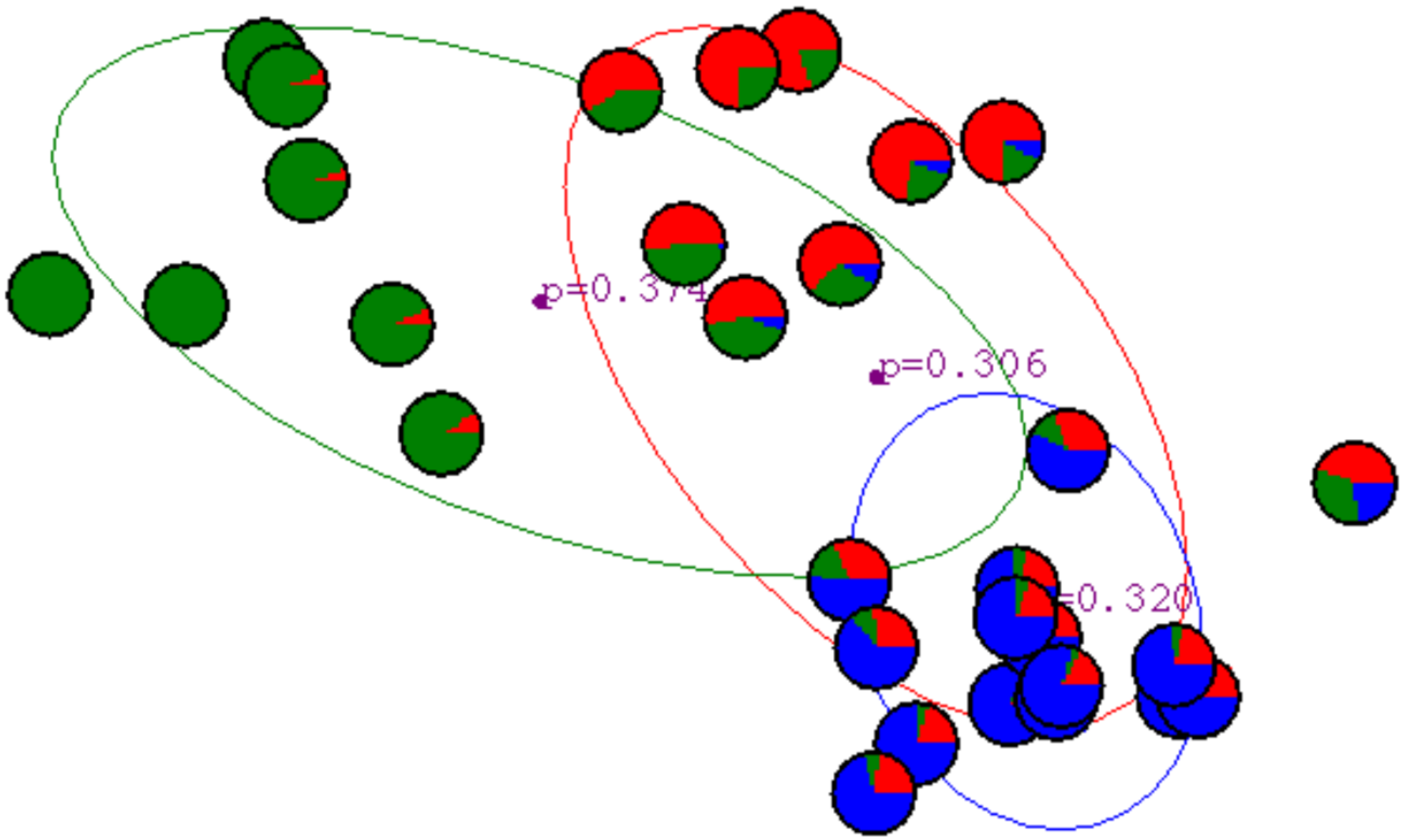
Old Faithful (2)



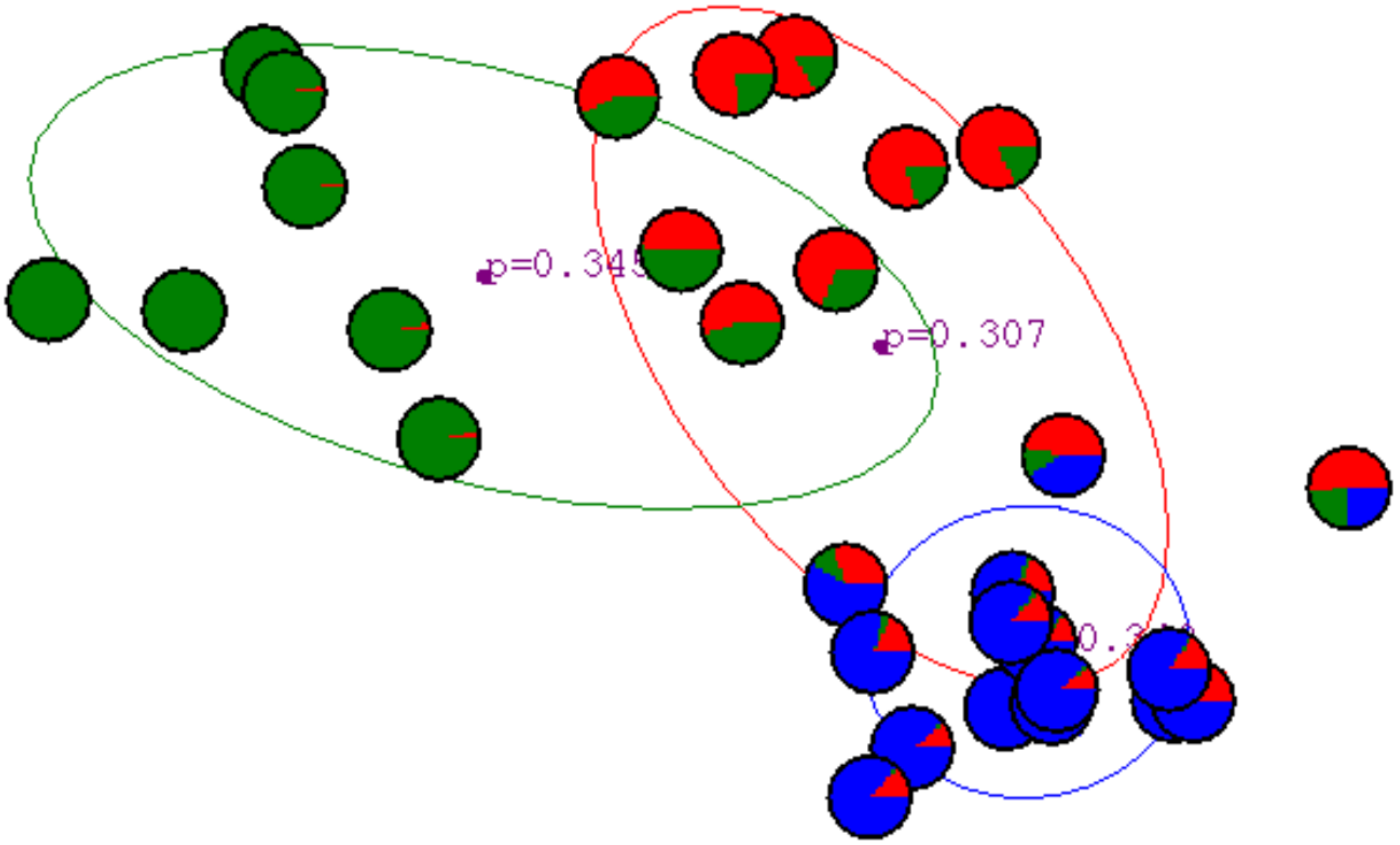
Example (Andrew Moore; 1)



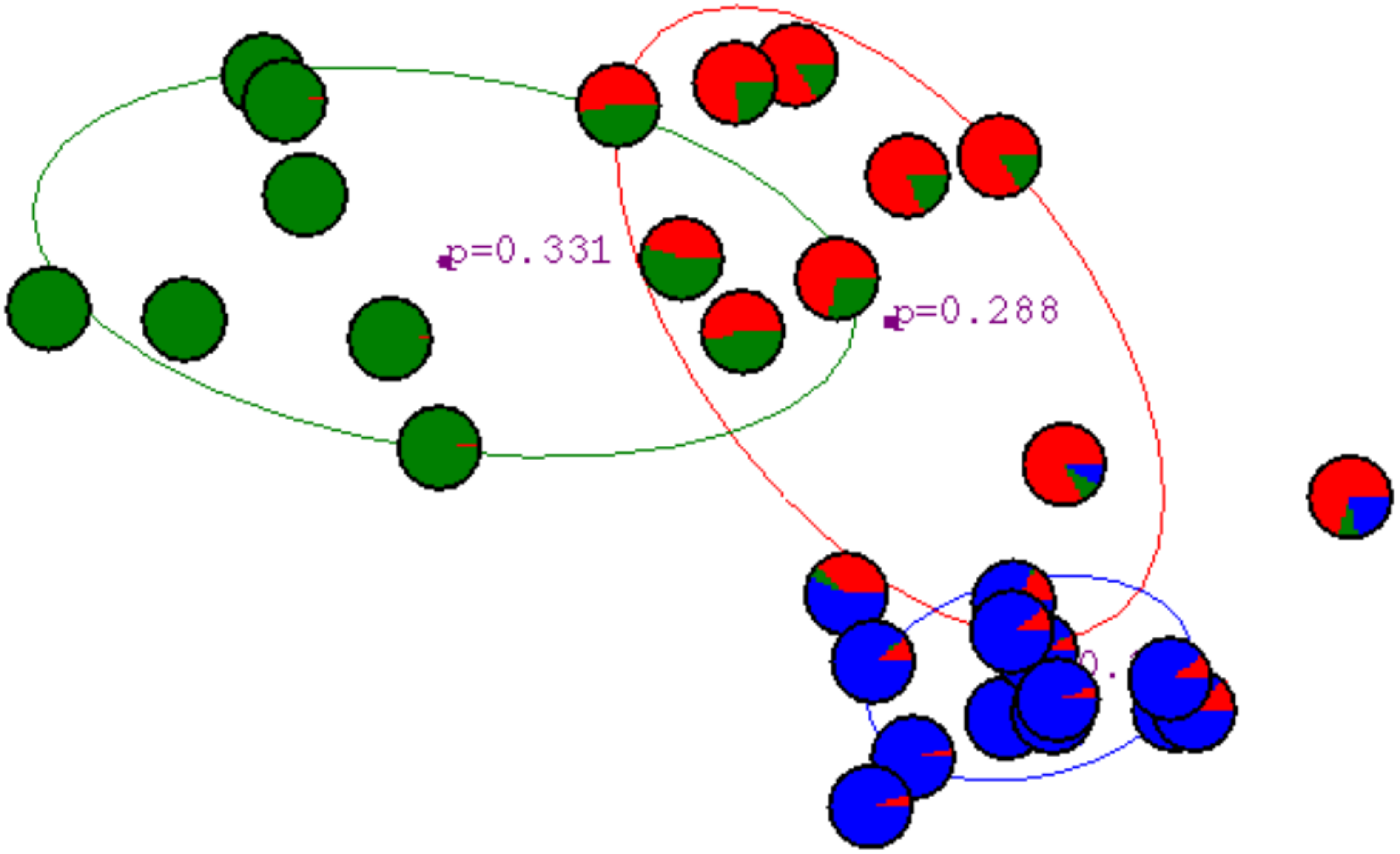
Example (Andrew Moore; 2)



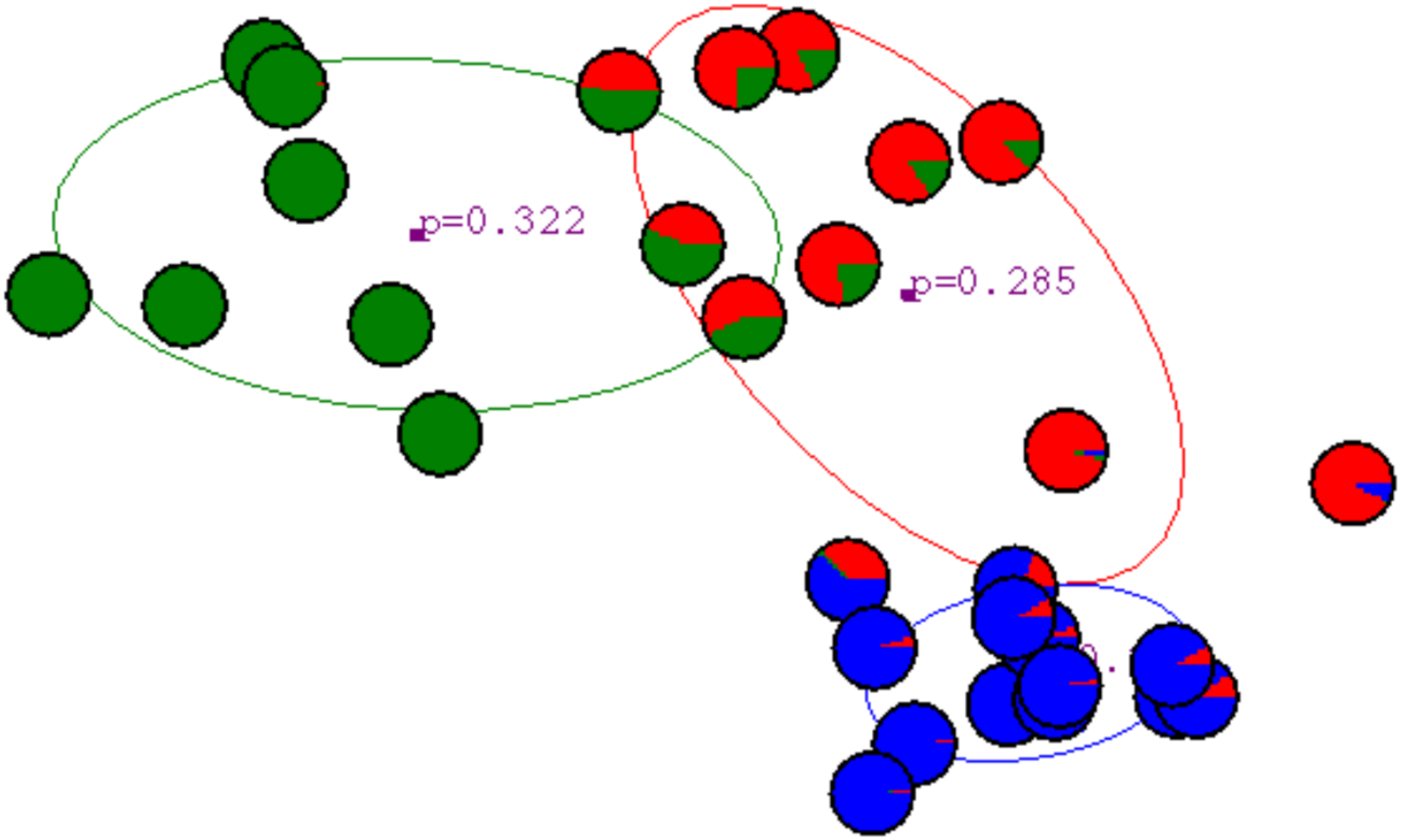
Example (Andrew Moore; 3)



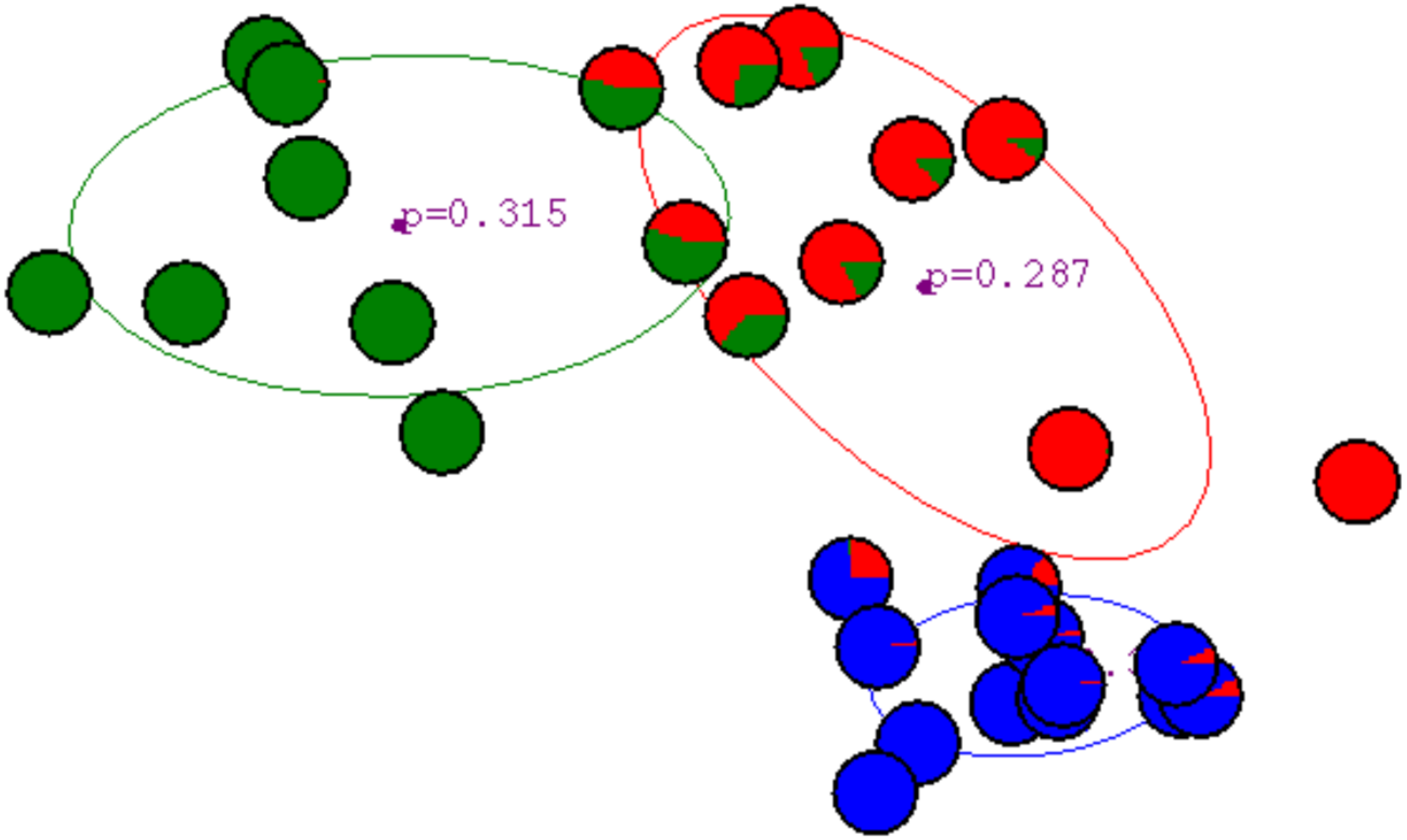
Example (Andrew Moore; 4)



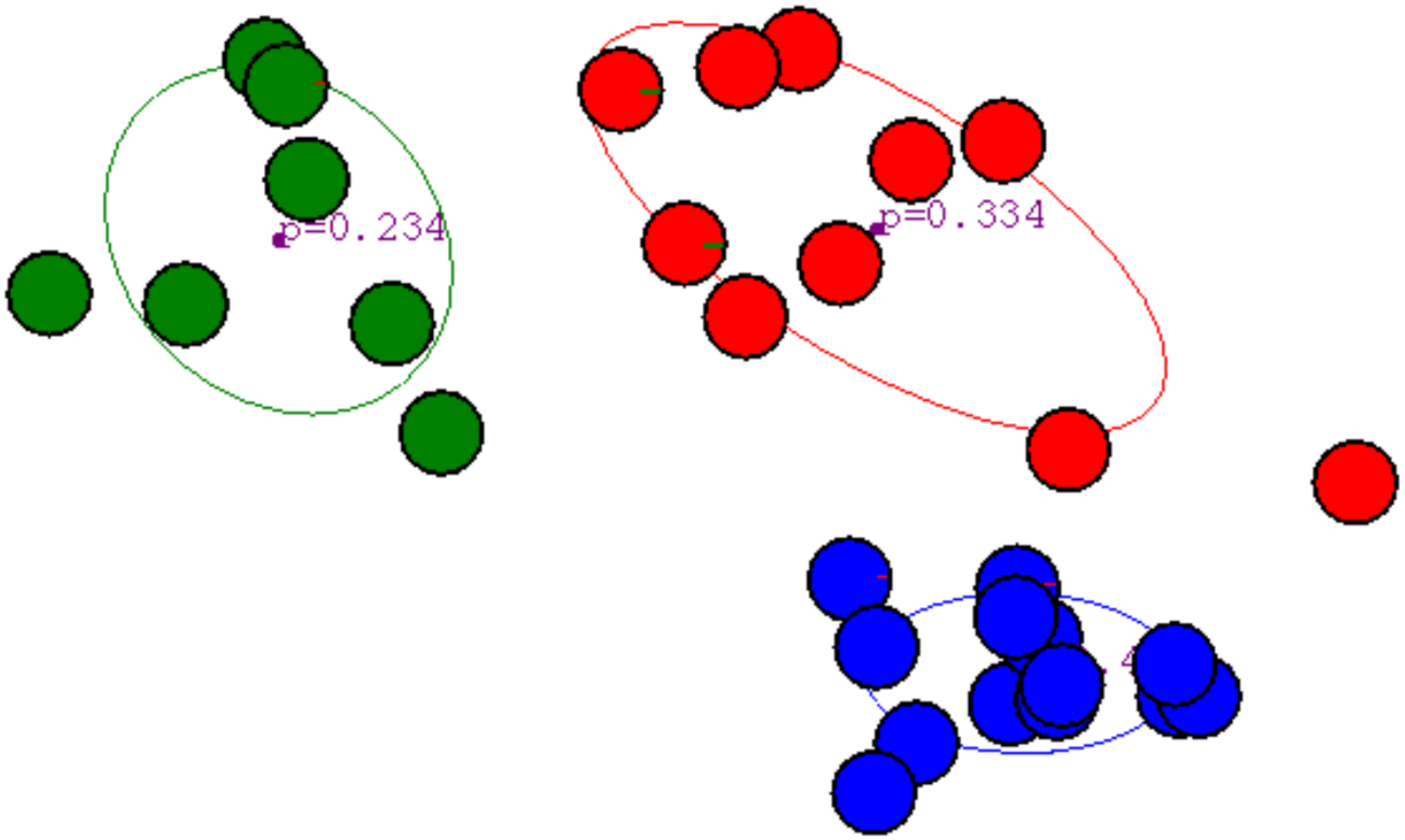
Example (Andrew Moore; 5)



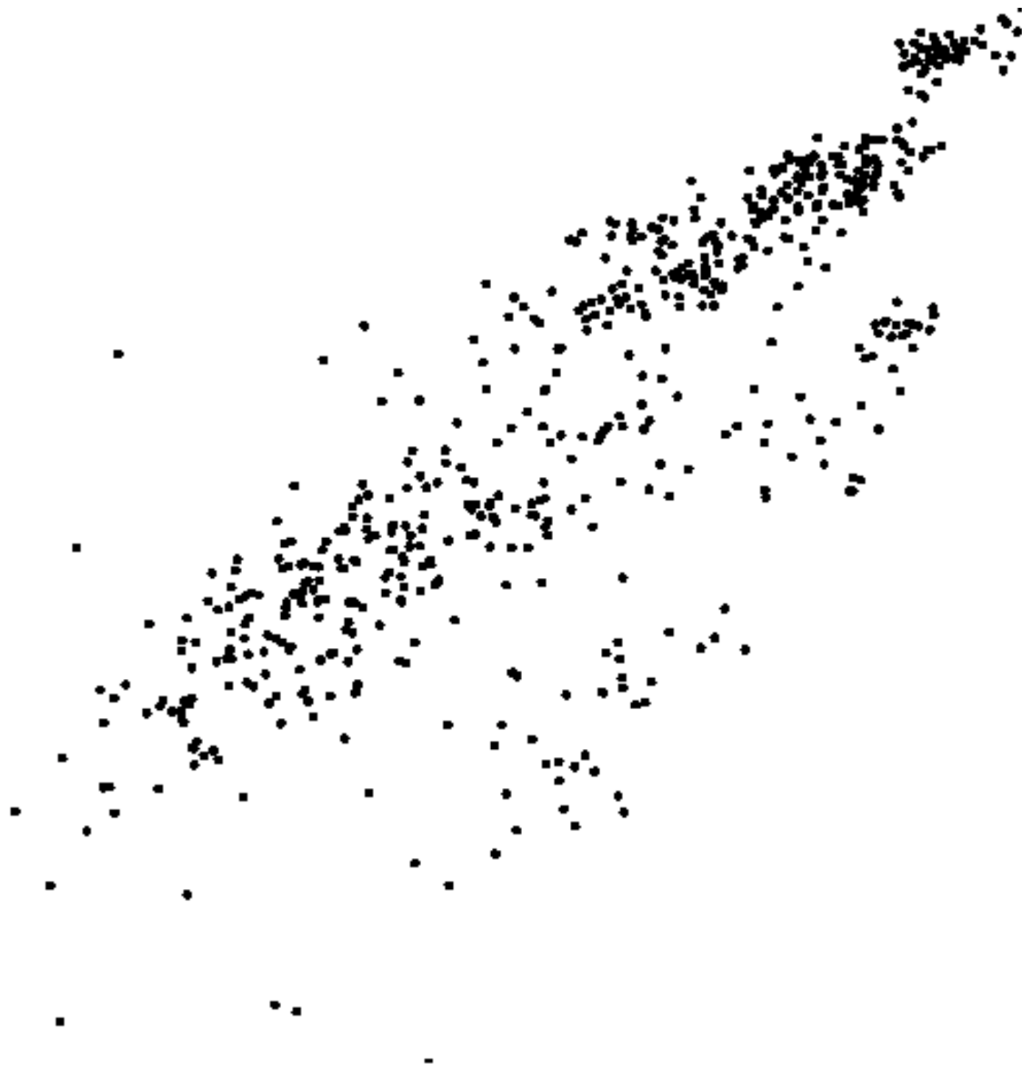
Example (Andrew Moore; 6)



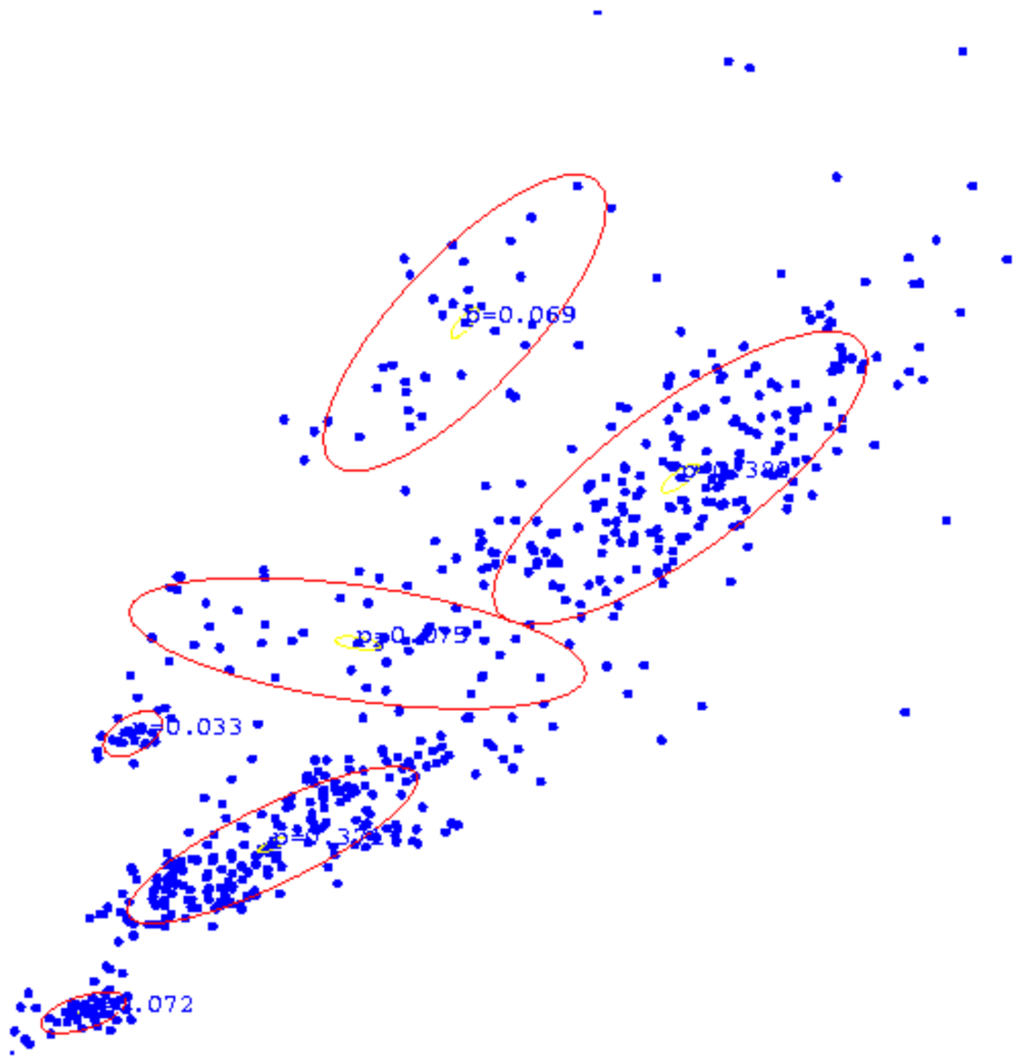
Example (Andrew Moore; 20)



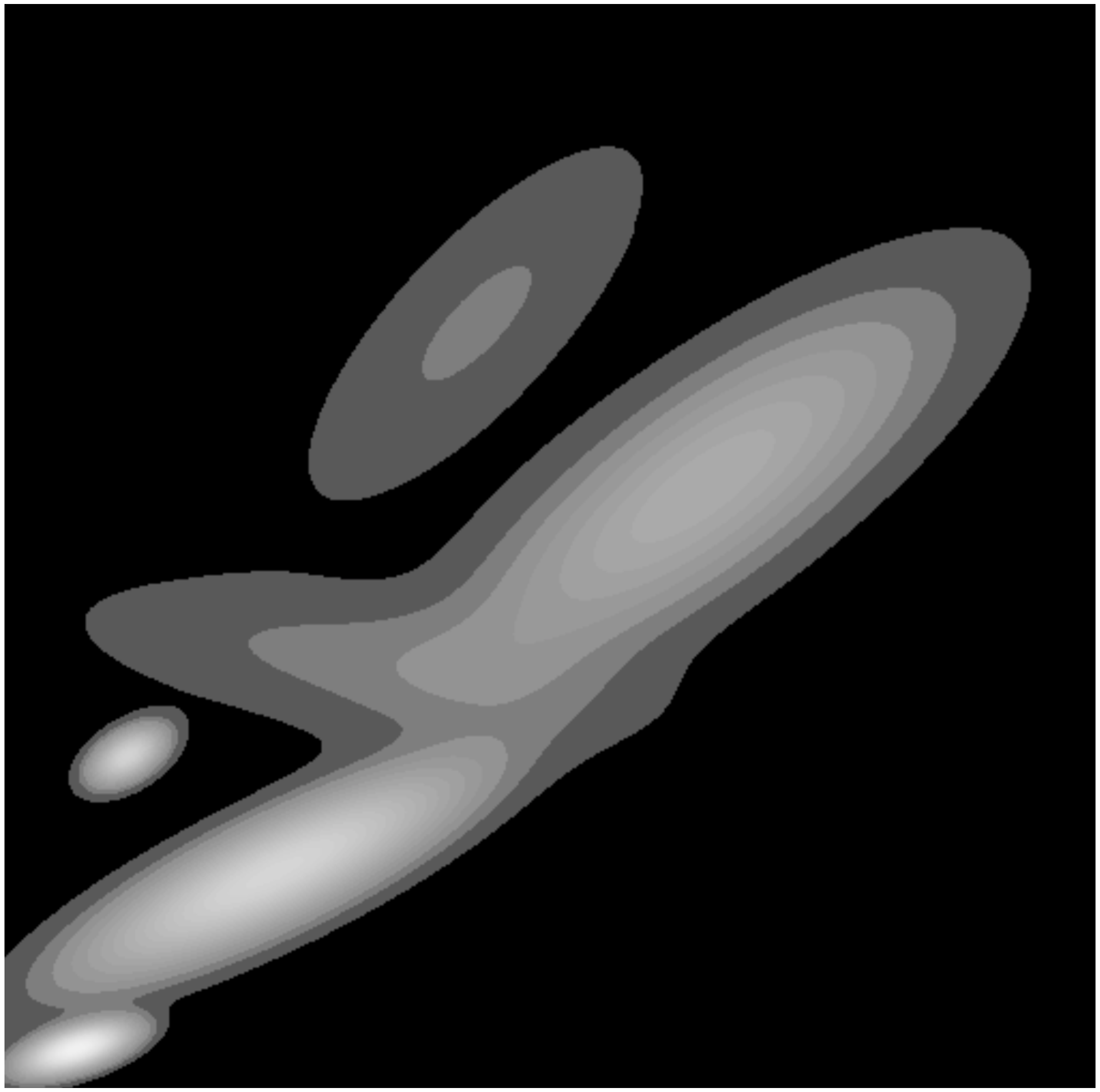
Assay Data (Andrew Moore)



Assay Clustering (Andrew Moore)



Assay Density (Andrew Moore)



Relationship to K-Means

K-Means is a special case of Gaussian Mixture Models in which…

$\Sigma = \sigma^2 \, \mathrm{I}$ (i.e. spherical, same for all)

$\pi = 1/K$ (i.e. equal probability of all)

Assignments are…

“hard” (r_n=one-hot) vs “soft” $p(z|x)$

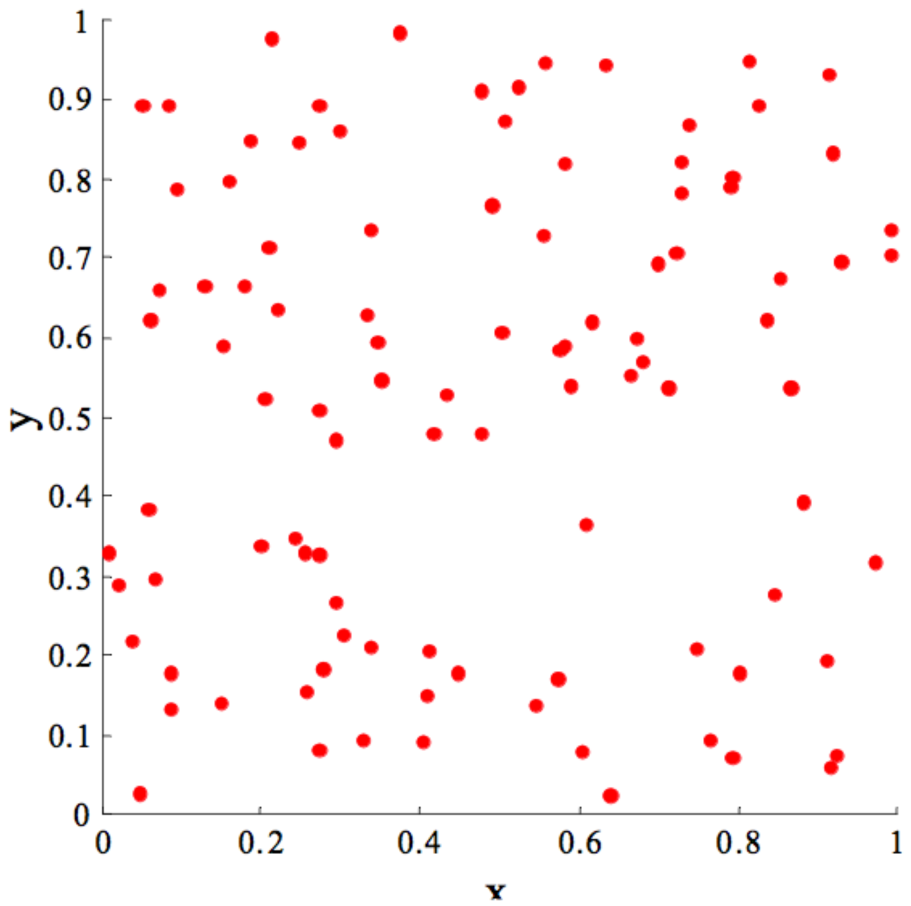
EM Notes

- Generally useful technique for finding maximum likelihood (MLE) or maximum a posteriori (MAP) estimates of parameters in statistical models
- Typically used where the model depends on unobserved latent variables
- Converges to a local maximum (may need random-restarts)

<u>Algorithm</u>

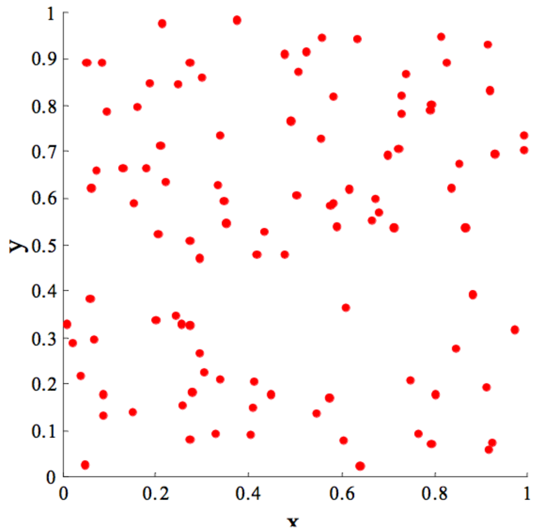
1. Initialize
2. Loop till convergence
   1. Maximize observed variables, fixing latent parameters
   2. Maximize parameters, fixing variables

What Makes for a "Good" Clustering?

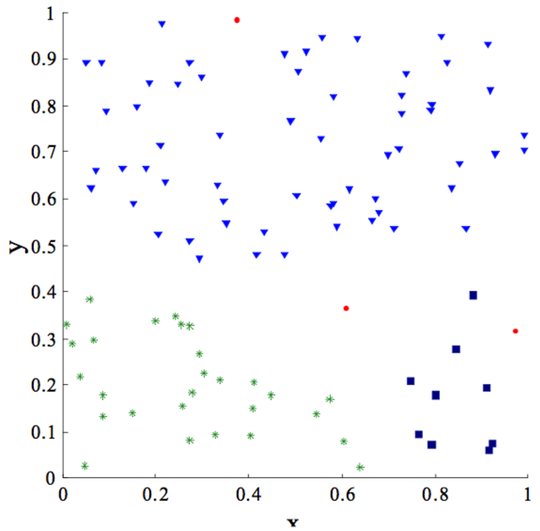


Did I Cluster *Well*?

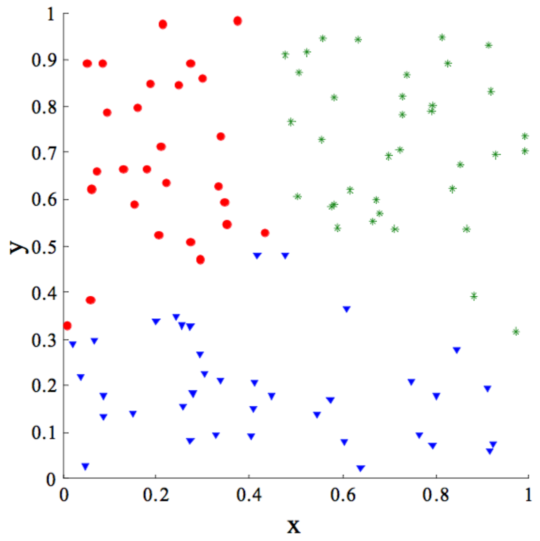
Random Points



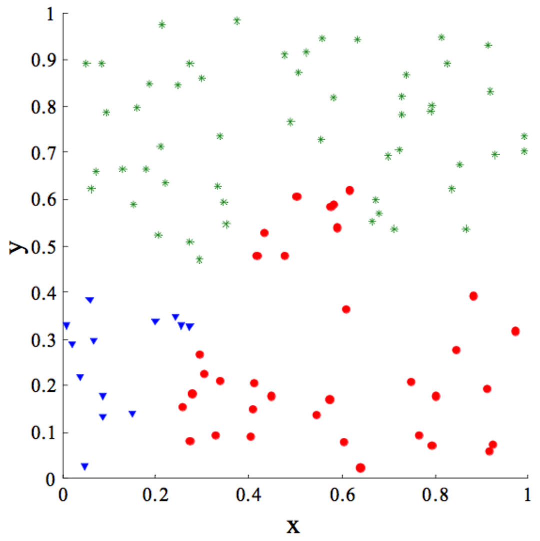
DBSCAN



K-means



Complete Link



Key Questions

1. Does non-random structure actually exist in the data?

2. What is the correct number of clusters?

3. How well do the results of a cluster analysis fit the data?

4. How well do the results of a cluster analysis adhere to externally known results?

5. Given two clusterings – which is better?

One of these is not like the other…

Key Distinction

- **Internal**/Unsupervised
  - No information, aside from the input data, is used during evaluation
  - Either you don't have ground truth, or are using this as a method of meta-optimization (e.g. how many clusters?)
- **External**/Supervised
  - Supplied ground truth not used during clustering, but is used during evaluation

Key Questions

1. Does non-random structure actually exist in the data?
2. What is the correct number of clusters?
3. **How well do the results of a cluster analysis fit the data?**
4. **How well do the results of a cluster analysis adhere to <u>externally known results</u>?**
5. Given two clusterings – which is better?

Evaluation Criteria*

Internal

Partitional

- Silhouette
- Proximity Matrix Analysis

Hierarchical

- Cophenetic Correlation

External

Classification

- Purity

Similarity

- Precision/Recall
- F-Measure

*There are many measures, we will examine a representative subset

Silhouette

- Combined measure of…
  - **Cohesion.** How similar an object is to other objects in its own cluster
  - **Separation.** How similar an object is to objects in other clusters

- Range: [-1, 1]

  Larger values = better clustering

Computing Silhouette

$$s(n) = \frac{b(n) - a(n)}{\max(a(n), b(n))}$$

- *a(n)*: the mean distance between an object and all objects in the same cluster (i.e. distance to the cluster mean)
- *b(n)*: the mean distance between an object and all other objects in the next nearest cluster (i.e. minimum distance to other cluster means)
- For a clustering, average for all objects: $\mathrm{SC} = \frac{1}{N}\sum_{n=1}^{N} s(n)$

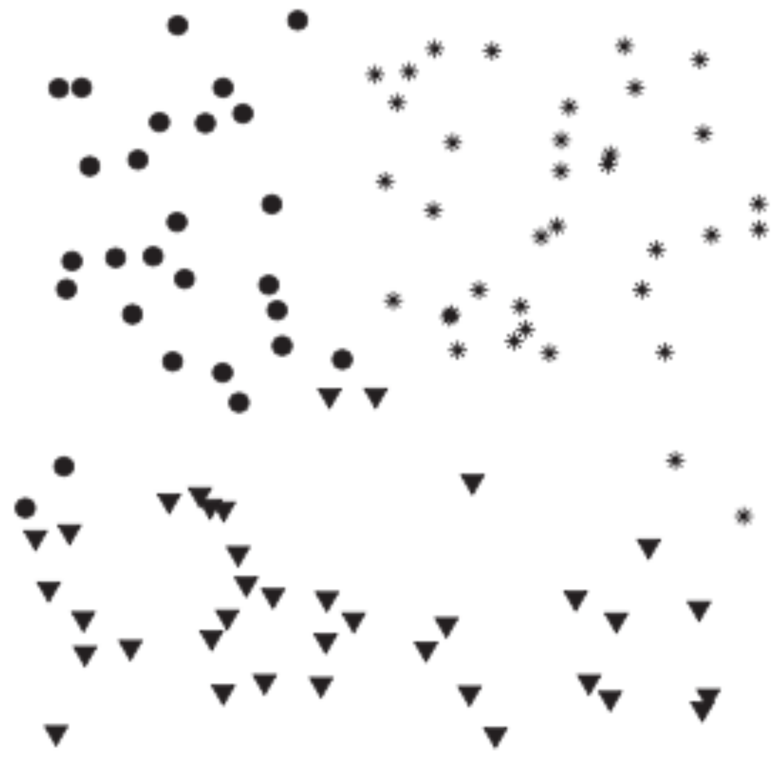
Proximity Matrix Analysis

- Given: the proximity matrix for a dataset, and an associated clustering

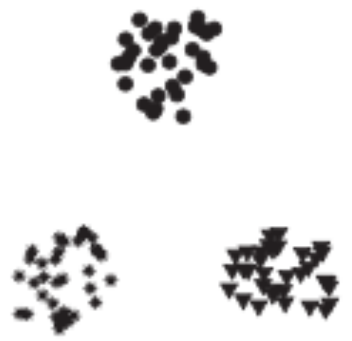
- If we sort the rows by cluster (i.e. rows that are in the cluster are nearby), we can then evaluate "goodness" in two ways
  - Correlation: compare to "ideal" matrix (similarity of 1=same, 0=different)
  - Visual: look for block diagonal structure

Proximity Correlation Comparison

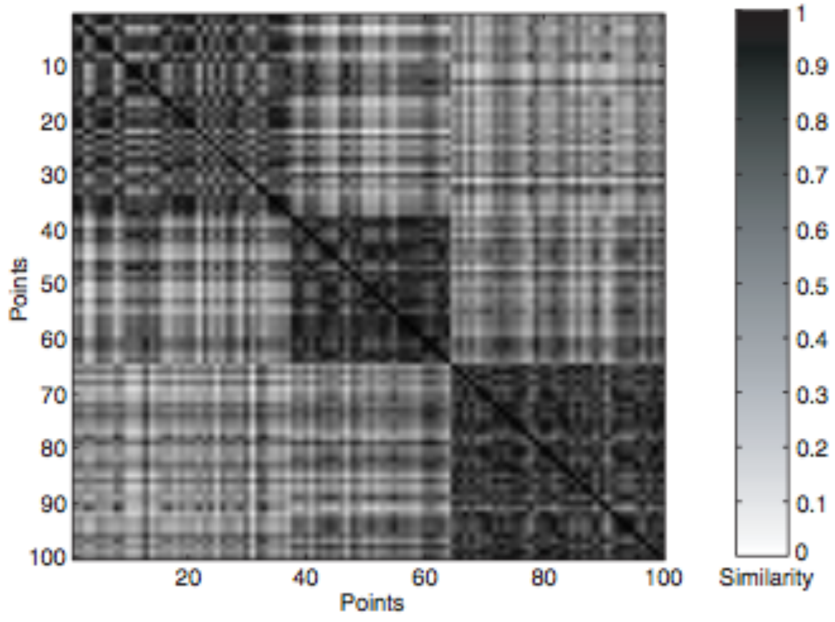
0.5810

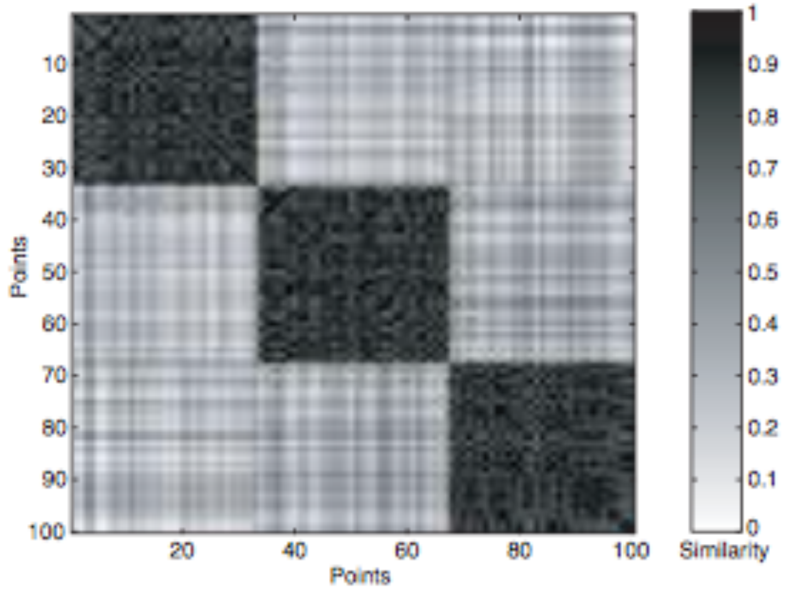


0.9235



Proximity Visual Comparison

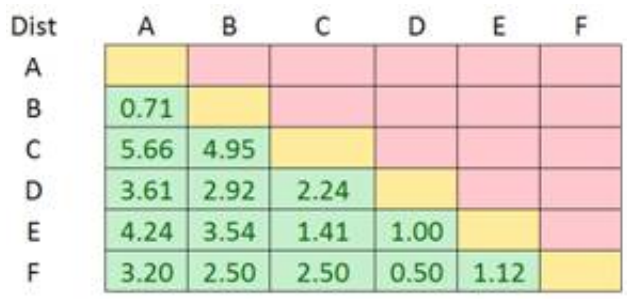




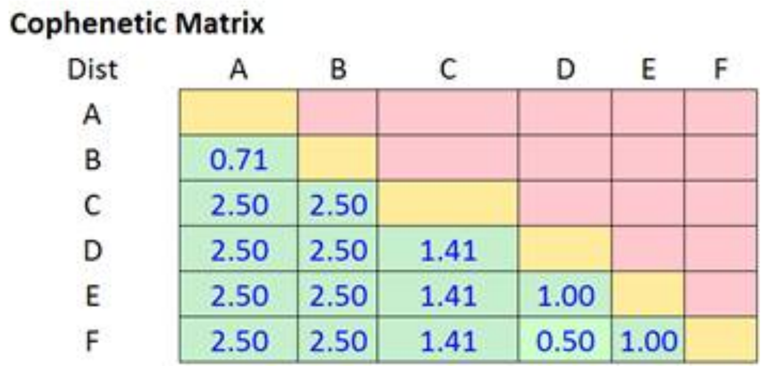
Cophenetic Correlation

- A popular evaluation tool for agglomerative hierarchical clustering
- Inputs: for each pair of points…
  - Distance
  - Cophonetic distance: distance when they were first put in the same cluster (also known as dendogrammatic distance)
- Output: closer to 1 is a better clustering

Example (1; Teknomo)



Dist	A	B	C	D	E	F
A						
B	0.71					
C	5.66	4.95				
D	3.61	2.92	2.24			
E	4.24	3.54	1.41	1.00		
F	3.20	2.50	2.50	0.50	1.12	



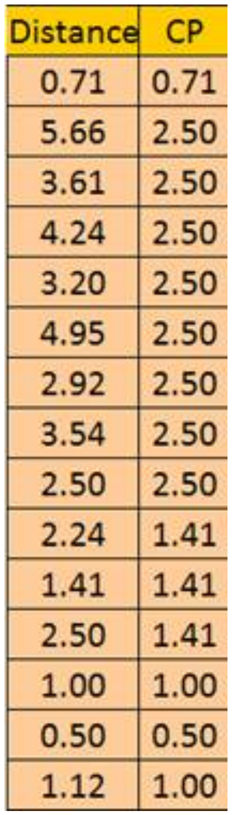
Cophenetic Matrix

Dist	A	B	C	D	E	F
A						
B	0.71					
C	2.50	2.50				
D	2.50	2.50	1.41			
E	2.50	2.50	1.41	1.00		
F	2.50	2.50	1.41	0.50	1.00	

Agglomerative Order

- {D,F} @ 0.5
- {A,B} @ 0.71
- {{D,F},E} @ 1.00
- {{{D,F},E},C} @ 1.41
- {{{{D,F},E},C},{A,B}} @ 2.5

Example (2; Teknomo)



Distance	CP
0.71	0.71
5.66	2.50
3.61	2.50
4.24	2.50
3.20	2.50
4.95	2.50
2.92	2.50
3.54	2.50
2.50	2.50
2.24	1.41
1.41	1.41
2.50	1.41
1.00	1.00
0.50	0.50
1.12	1.00

c = 0.8639

- For each pair, associate **d**istance with **c**ophenetic distance

- Compute…

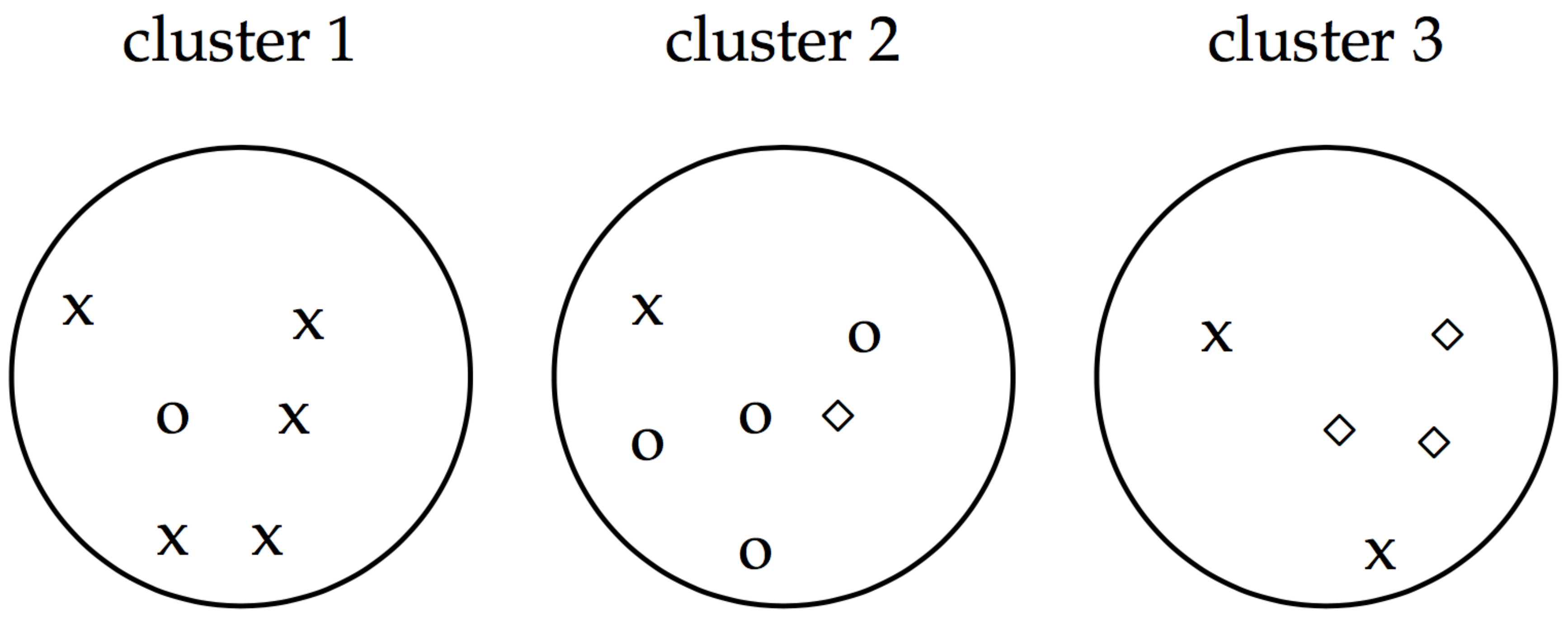
$$c = \frac{\sum_{i<j}(d_{ij} - \bar{d})(c_{ij} - \bar{c})}{\sqrt{[\sum_{i<j}(d_{ij} - \bar{d})^2][\sum_{i<j}(c_{ij} - \bar{c})^2]}}$$

Purity

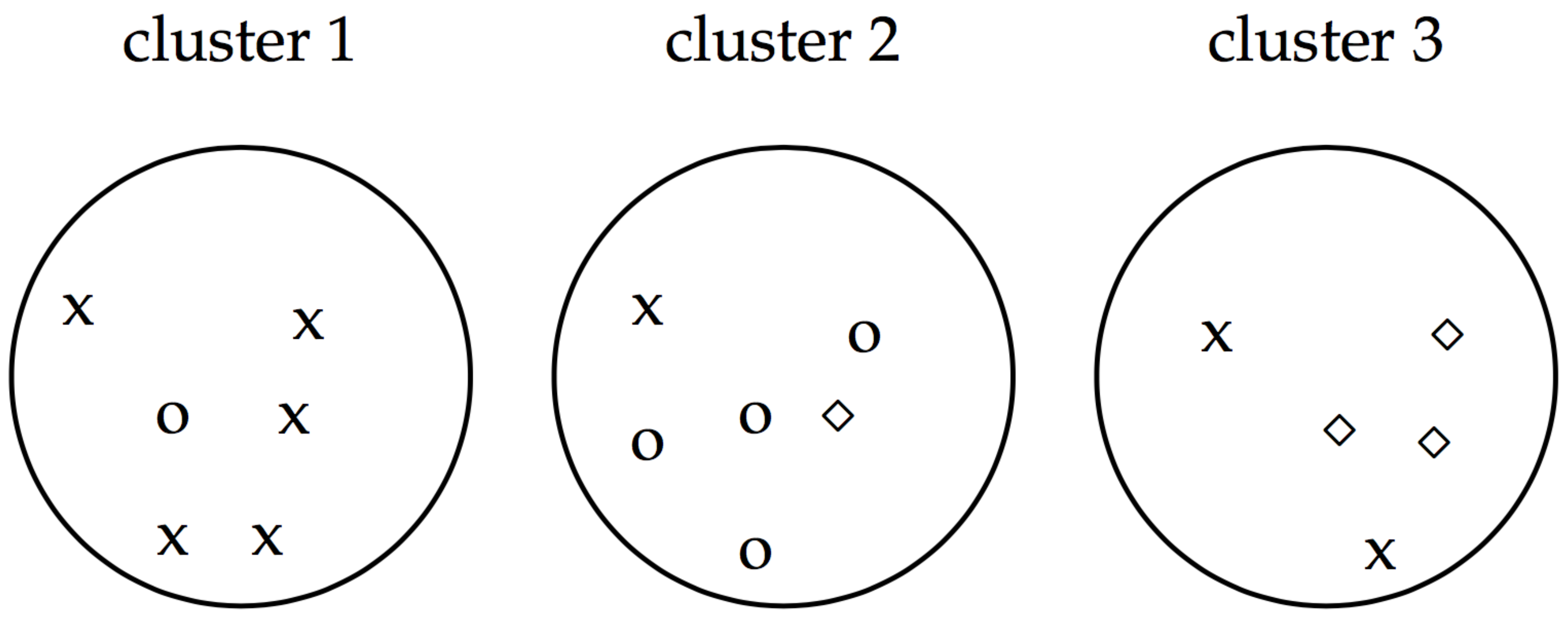
- A measure of the extent to which clusters contain a single class
- Calculation…
  - For each cluster, count the number of data points from the most common class in the cluster
  - Sum over all clusters and divide by the total number of data points
- Range: [0,1]

  Perfect = 1

Example (Manning et al.)



Example (Manning et al.)



5 + 4 + 3

= 12 / 17

= 0.706

Quick Check

- What happens to the purity measure as the number of clusters increases?

Quick Check

- What happens to the purity measure as the number of clusters increases?

- Easy to achieve 1 :)
  – Normalized Mutual Information (NMI) is a measure that allows you examine the tradeoff between number of clusters and cluster quality (related to KL divergence)

Decision Quality

Compute the following four quantities for all pairs (N[N-1]/2) of objects in the dataset

		True Class	
		Same	Different
Clustering	Same	**T**rue **P**ositive	**F**alse **P**ositive
	Different	**F**alse **N**egative	**T**rue **N**egative

Decision Quality

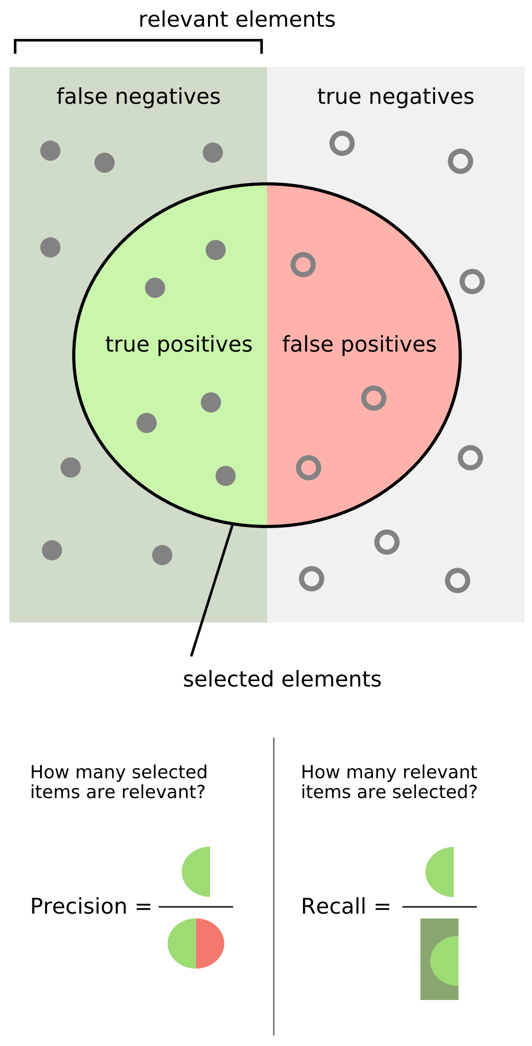
- Rand Index: accuracy of correct decisions

  RI = (TP + TN) / (TP + FP + FN + TN)

  – FP/FN weighted equally, not always ideal

- To build up to weighting between FP/FN…
  – **P**recision = TP / (TP + FP)
  – **R**ecall = TP / (TP + FN)

Precision/Recall Visualized



F-Measure

- Allows for weighting between FP/FN error types via parameter β

$$F_\beta = \frac{(\beta^2 + 1) \cdot P \cdot R}{\beta^2 \cdot P + R}$$

- Common F1 score (β=1) gives equal weighting

Where We’ve Been

1. Clustering overview
   - Why
   - Distance measures
   - Types
2. K-Means (in-depth)
   - Derivation
   - Algorithm, convergence
   - Assumptions/limitations
   - Complexity/scaling
3. Agglomerative Hierarchical Clustering
4. DBSAN
5. Gaussian Mixture Models
6. Evaluation
   - Internal: partitional/hierarchical
   - External: classification/similarity