Unsupervised Learning: Clustering

Prof. Mingkui Tan

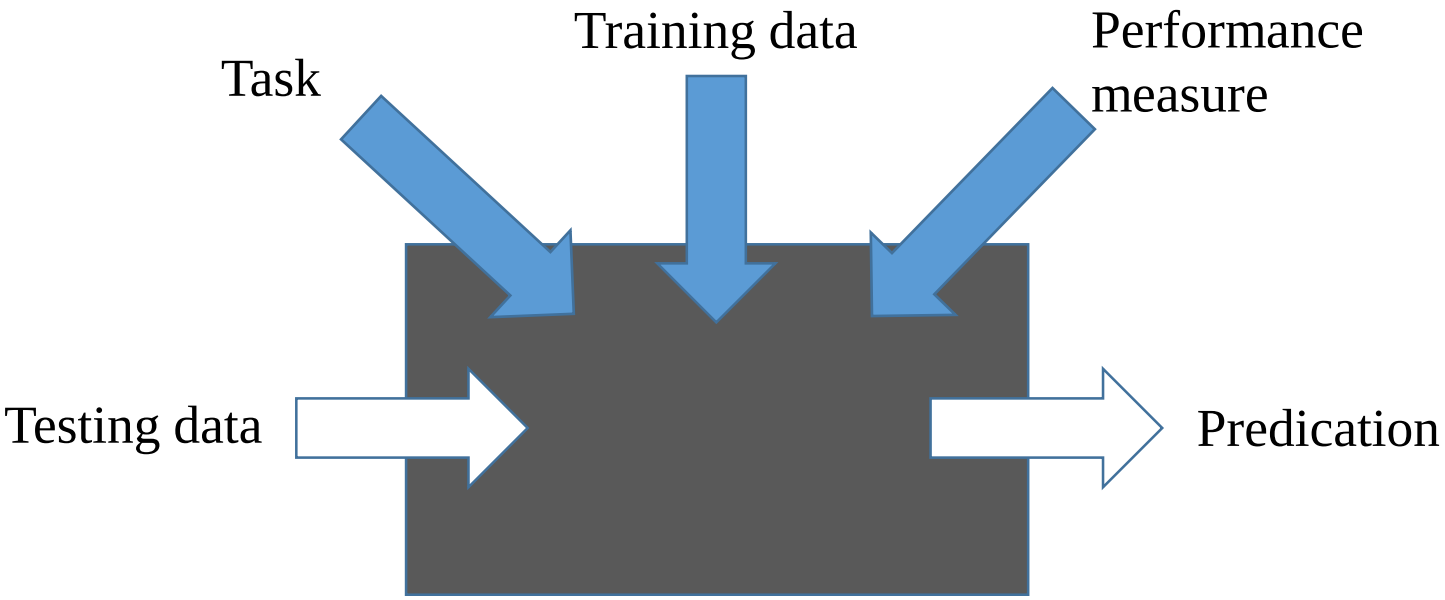
SCUT Machine Intelligence Laboratory (SMIL)

Contents

Contents

Supervised Learning

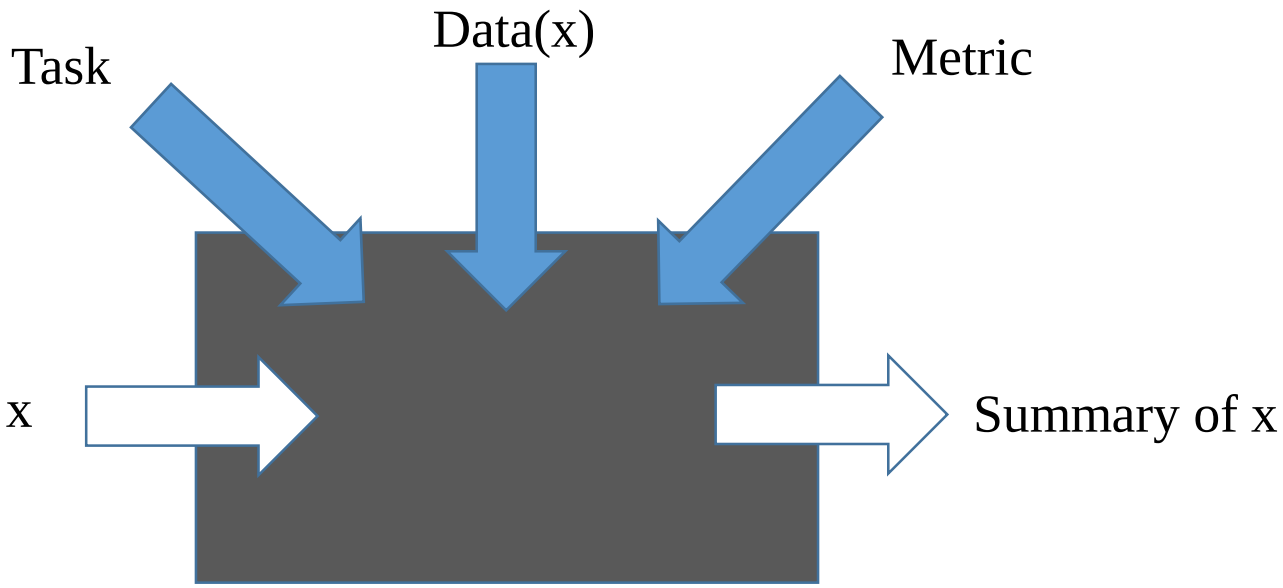
- Data $\mathcal{D} = \{(\mathbf{x}_1, \mathbf{y}_1), \cdots (\mathbf{x}_n, \mathbf{y}_n)\}$ has target values
- **Supervised Learning methods:** linear regression, logistic regression, Naive Bayes, Neural Nets, SVMs. etc.



Unsupervised Learning

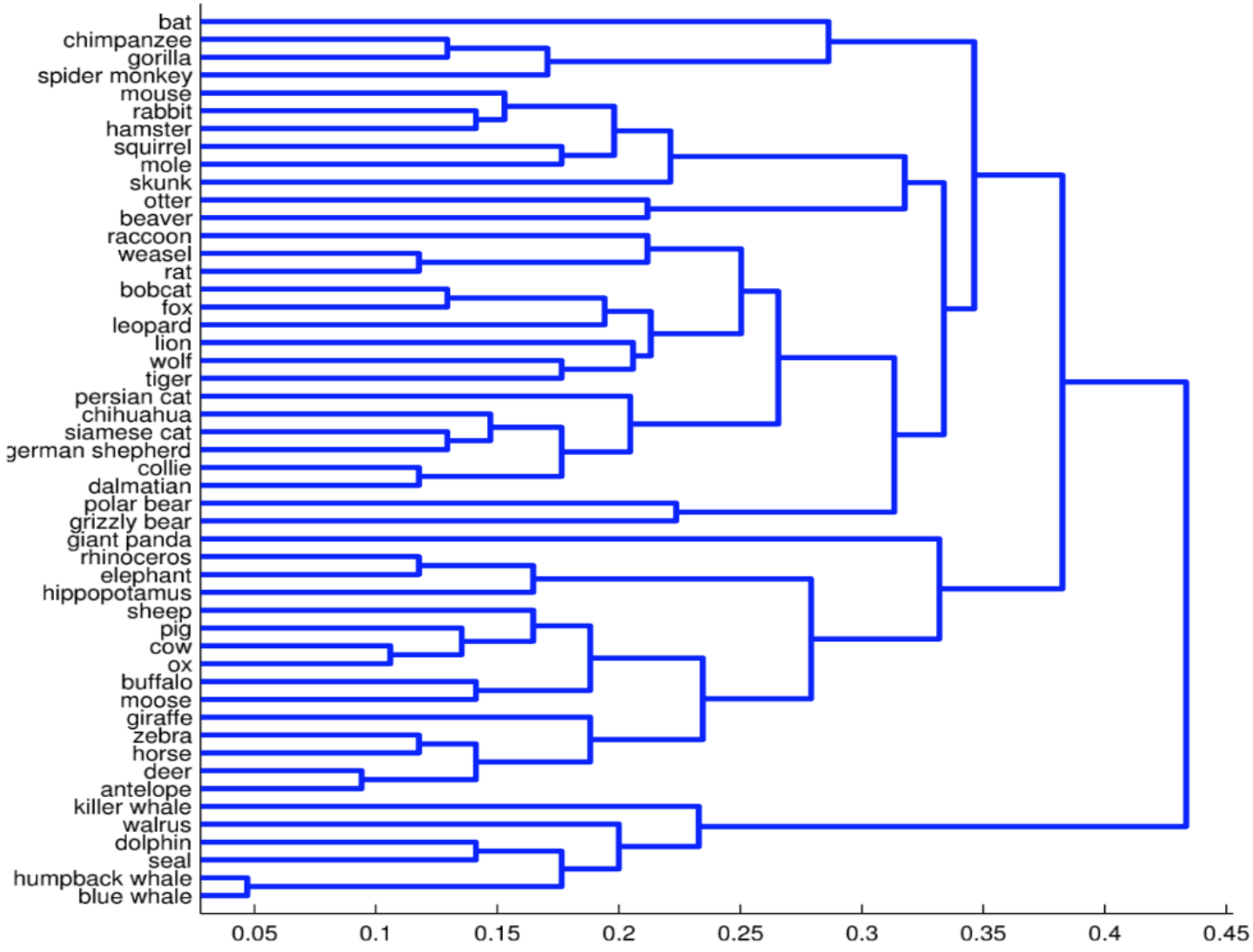
- Data $\mathcal{D} = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n\}$. No target values

- **Typical goals:** understand data, summarize data, identify concepts



Application: Clustering Animals by Features

- Data set of 50 animals,85 binary features (e.g. longneck, smelly)



Application: Clustering Image Data


(a) Cluster Centers


(b) Cluster 1


(c) Cluster 2


(d) Cluster 3


(e) Cluster 4

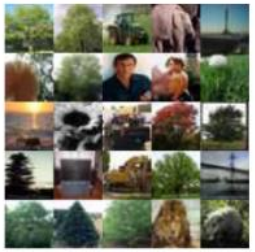

(f) Cluster 5


(g) Cluster 6


(h) Cluster 7


(i) Cluster 8


(j) Cluster 9


(k) Cluster 10


(l) Cluster 11


(m) Cluster 12

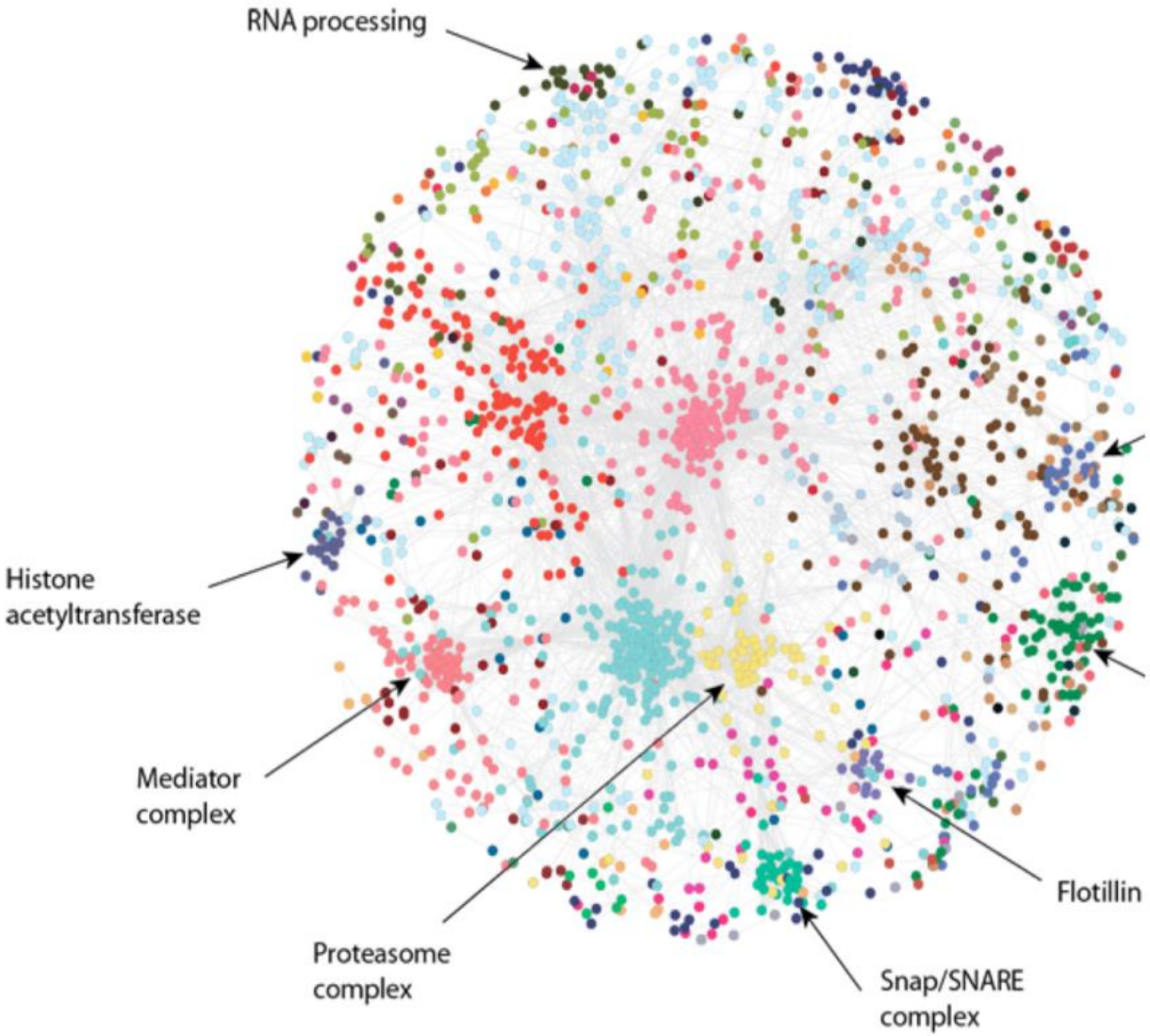

(n) Cluster 13


(o) Cluster 14


(p) Cluster 15


(q) Cluster 16

Application: Understanding Gene Regulation



Contents

Clustering

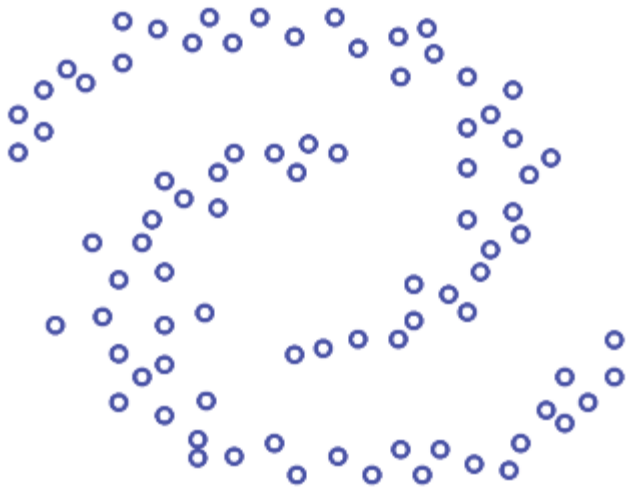
- **Description:** Simple idea for discovering structure
- Find groups of similar samples:
  1) To understand the data
  2) For dimensionality reduction
  3) To preprocess unlabeled data, find concepts to use for supervised learning

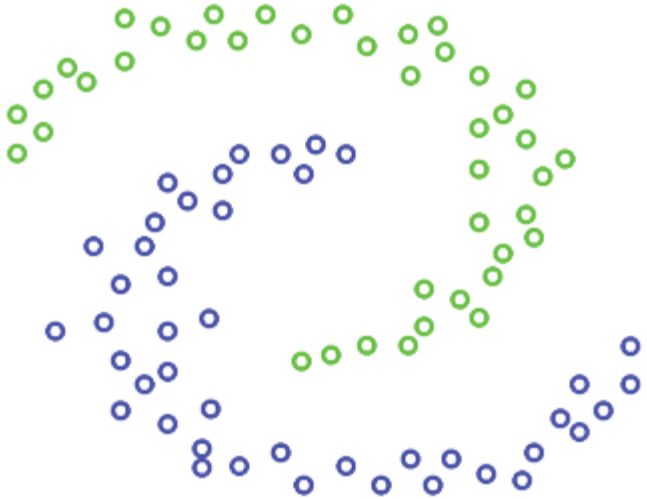
Today's lecture:

- K-means clustering
- Hierarchical Agglomerative Clustering(HAC)

Example 1: How Would You Cluster These Points

Example 2: How Would You Cluster These Points



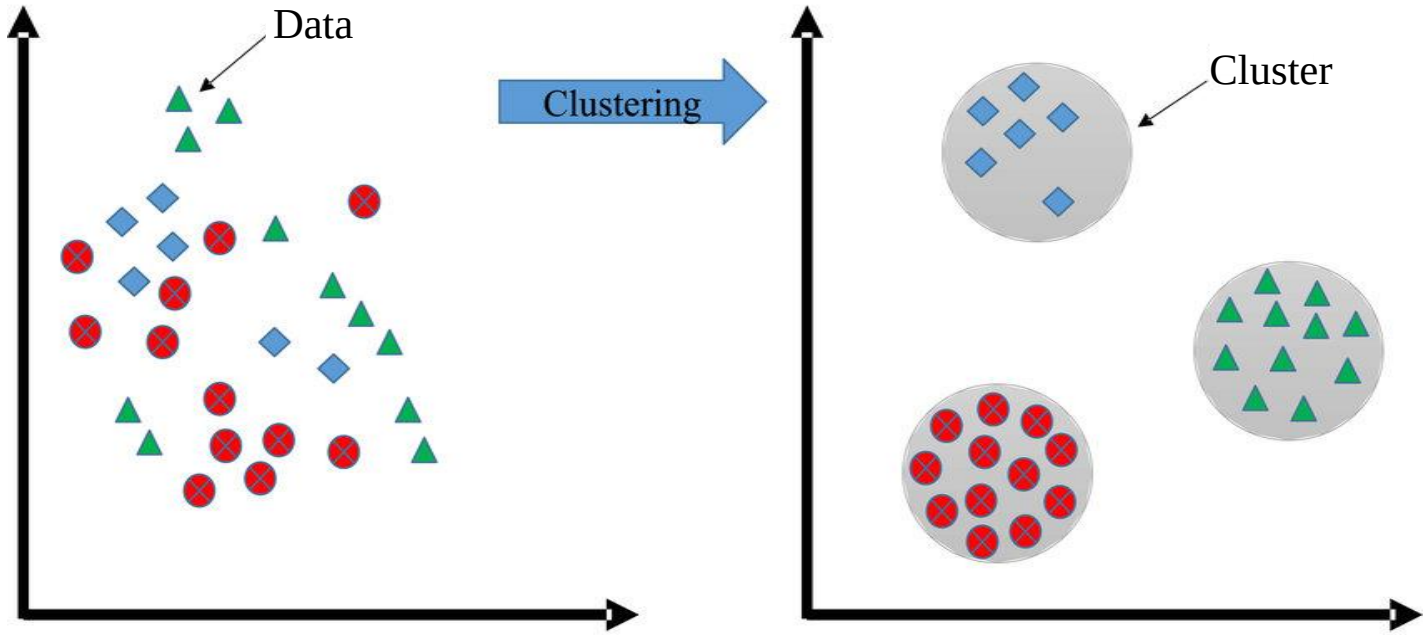


Clustering

- **Input**: Data $\mathcal{D} = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n\}$

  Number of clusters K (may not be given)

- **Output**: assignment of each sample to a cluster



What is Good Clustering?

- Samples are "more similar" to the samples in their cluster than to examples in other clusters.

- How to measure the similarity?
  - the similarity between one sample **x** to another $\hat{\mathbf{x}}$?
  - the similarity between one group of samples to another?

For data in $\mathbb{R}^m$, a typical approach is L_2 metric:

$$\boldsymbol{d}(\mathbf{x}, \hat{\mathbf{x}}) = \|\mathbf{x} - \hat{\mathbf{x}}\| = \sqrt{\sum_j (x_j - \hat{x}_j)^2}$$

Can also use specialized metrics, e.g., edit distance for strings or DNA sequences, the Hamming distance for bit vectors

Contents

K-means Objective

Defined data in $\mathbb{R}^m$

- Associate each cluster with a prototype $\boldsymbol{\mu}_k \in \mathbb{R}^m$, for $k \in \{1, \cdots, K\}$
- Make an assignment $\mathbf{r}_i$ of each sample to $\mathbf{x}_i$ a cluster

Objective: find prototypes and an assignment to minimize

$$\boldsymbol{L}(\{\boldsymbol{r}\}, \{\boldsymbol{\mu}\}) = \sum_{i=1}^{n} \sum_{k=1}^{K} r_{ik} \|\mathbf{x}_i - \boldsymbol{\mu}_k\|$$

Where $r_{ik} \in \{0, 1\}, r_{ik} = 1$ denotes $\mathbf{x}_i$ belongs to the cluster k, $r_{ik} = 0$ otherwise, n is the number of samples

This is highly non-convex, with lots of local minima

K-means Clustering Algorithm (Lloyd's Algorithm)

- Initialize prototypes $\boldsymbol{\mu}_1, \cdots, \boldsymbol{\mu}_K$

- Repeat until converged:

  Step 1: Assign each sample to the closest prototype

$$k^* = \underset{k \in \{1,\cdots,K\}}{\operatorname{argmin}} \|\mathbf{x}_i - \boldsymbol{\mu}_k\|, r_{ik} = \begin{cases} 1, & k = k^* \\ 0, & otherwise \end{cases}$$

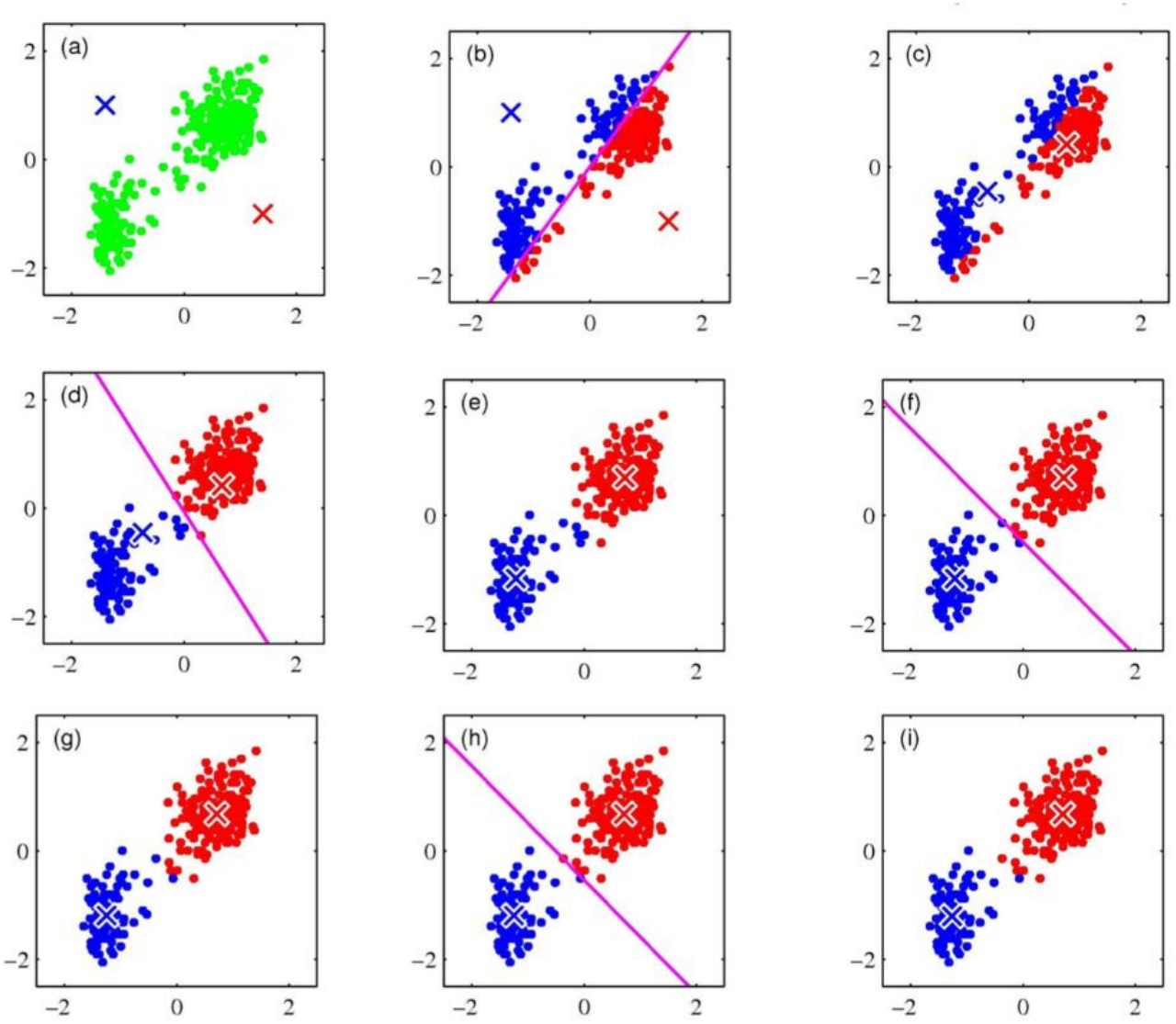
  Step 2: For each k, set $\boldsymbol{\mu}_k$ to the centroid of assigned samples

$$\boldsymbol{\mu}_k = \frac{1}{n_k} \sum_{i=1}^{n} r_{ik} \mathbf{x}_i$$

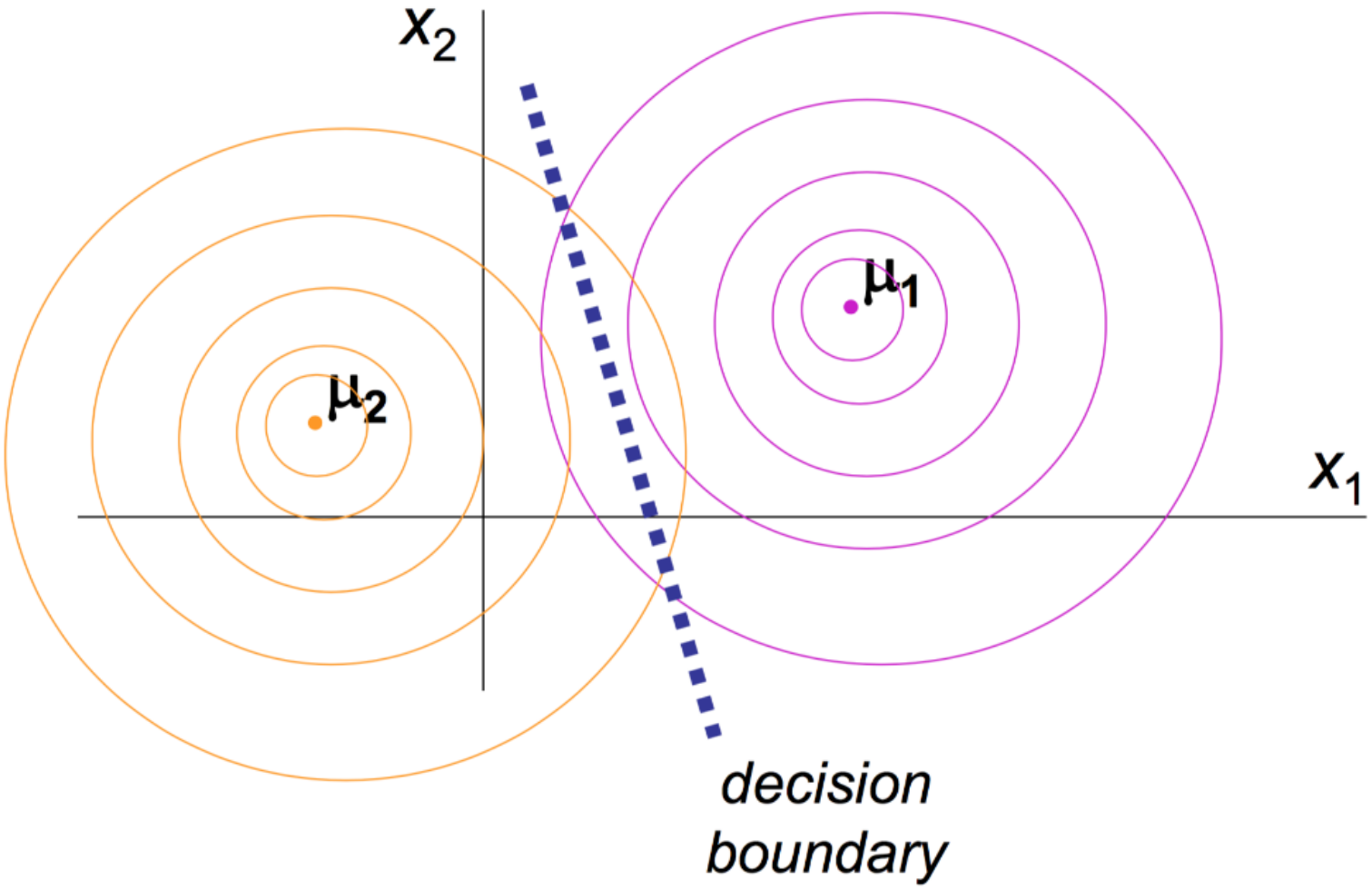
  where $n_k = \sum_i r_{ik}$

Typical to run this multiple times, with different initial conditions

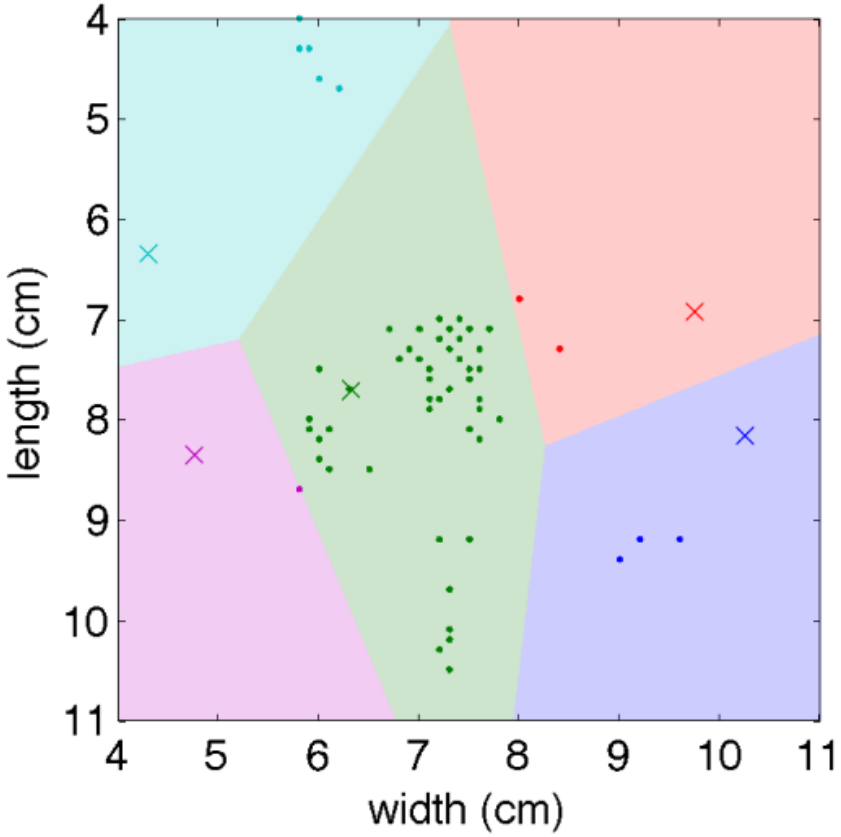
Example: K-means on Old Faithful Eruptions



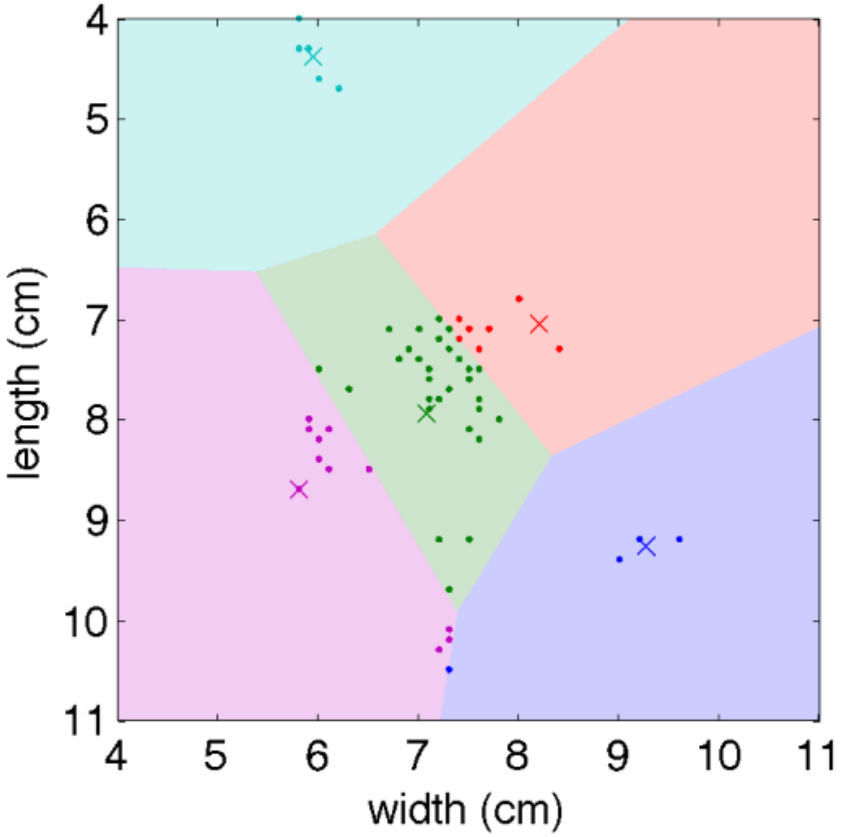
Note: Linear Decision Boundaries



Example: K-means on Oranges and Lemons(1 of 4)



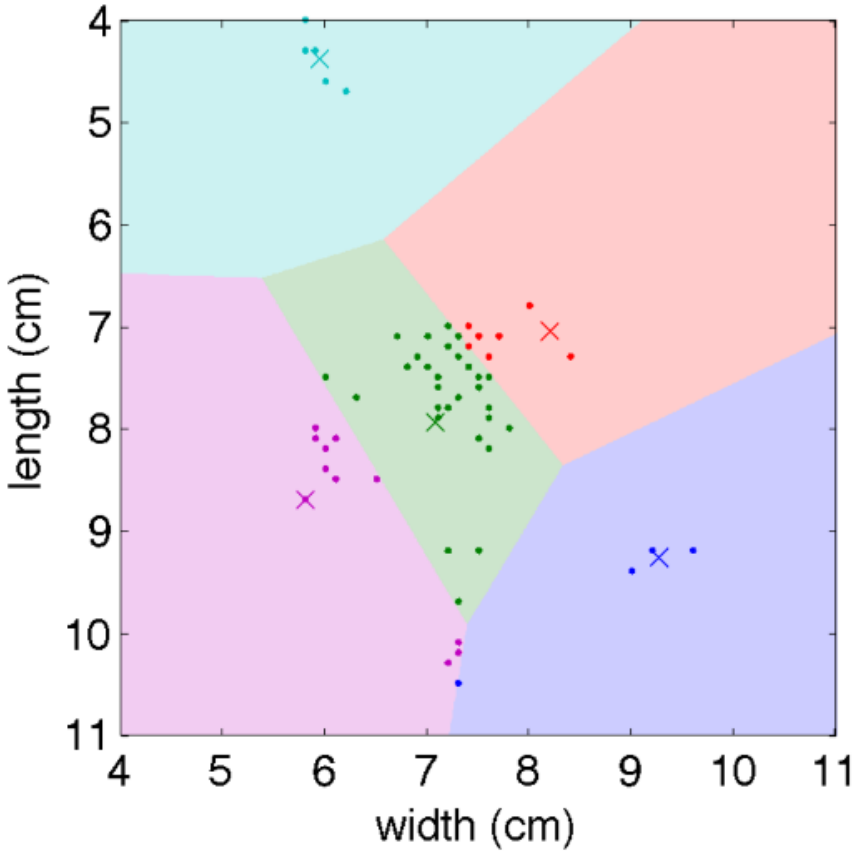
(a) Initialization



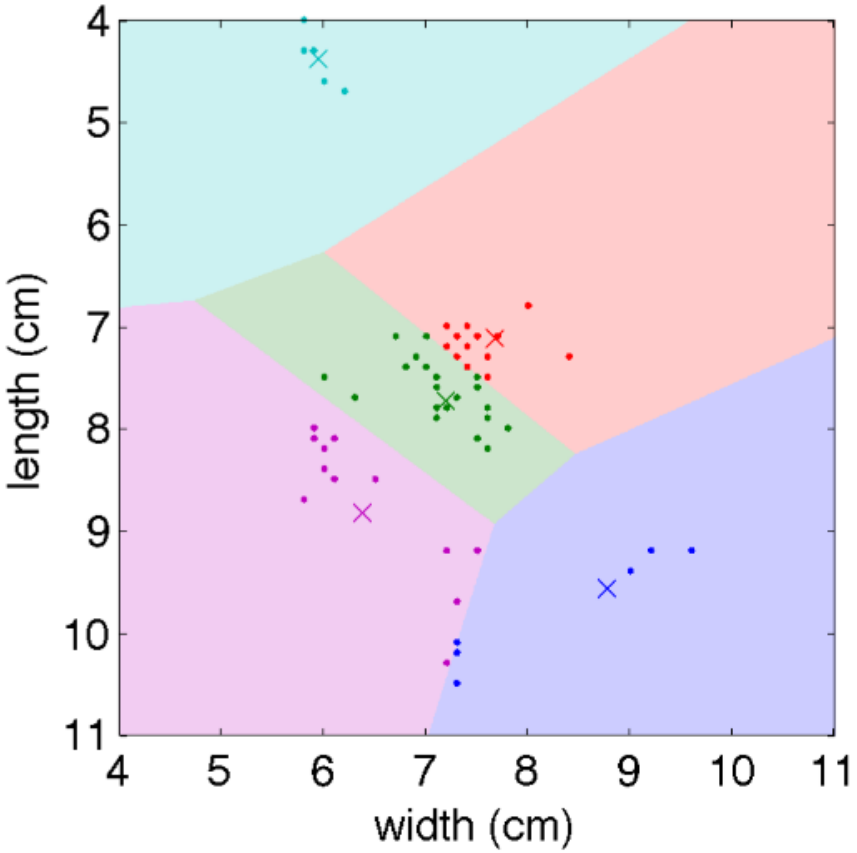
(b) Iteration 1

(K=5,lain Murray data)

Example: K-means on Oranges and Lemons(2 of 4)



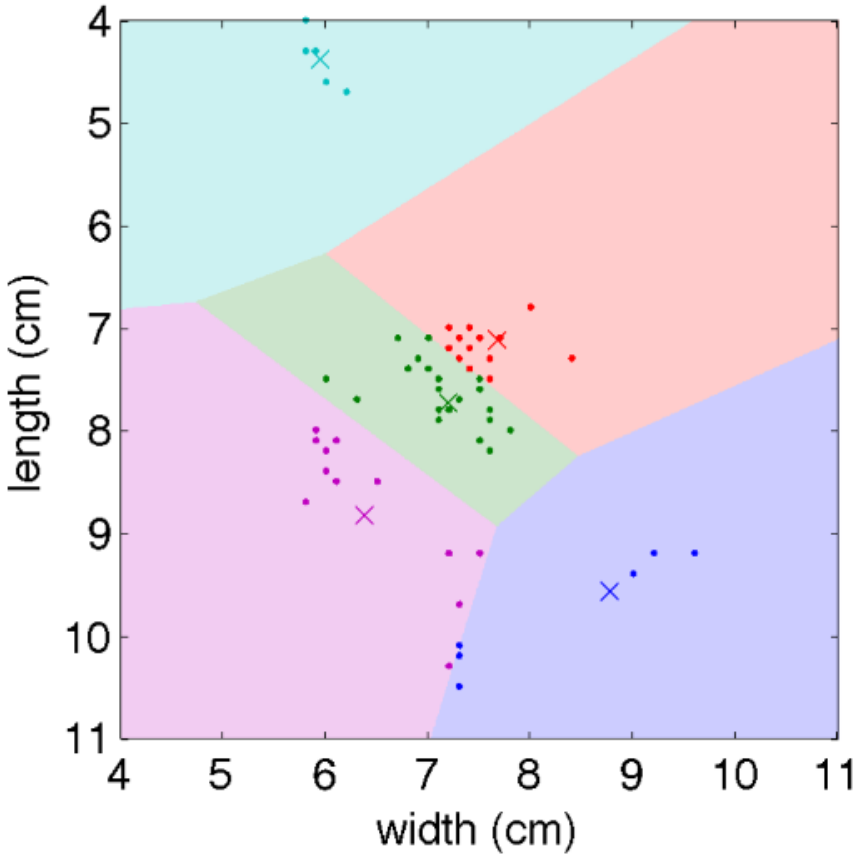
(b) Iteration 1



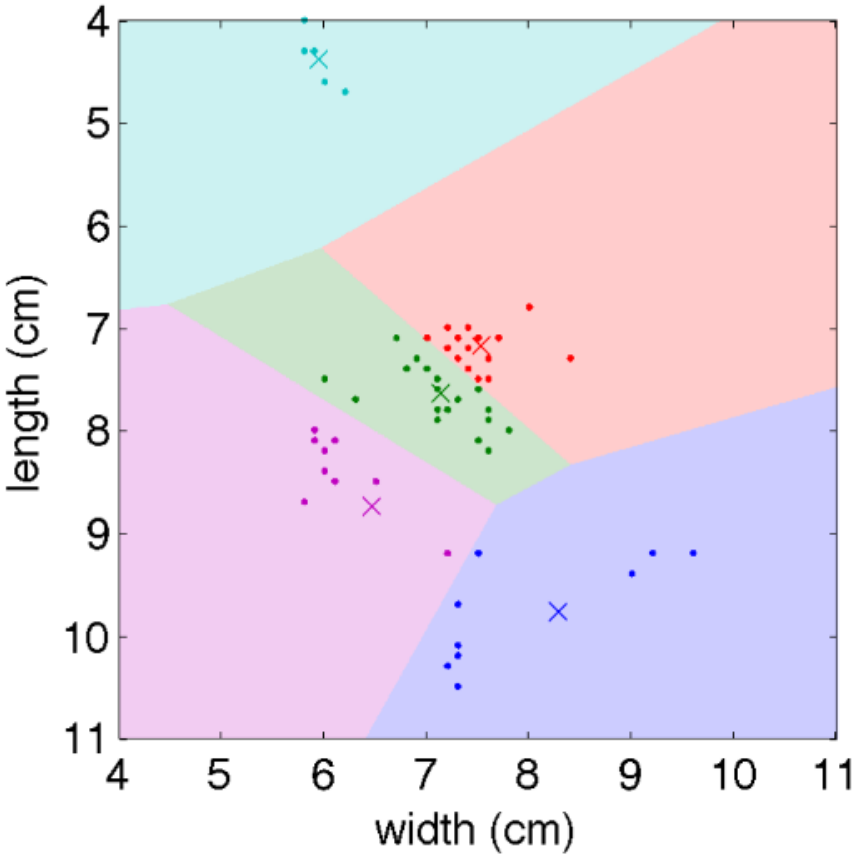
(c) Iteration 2

(K=5,lain Murray data)

Example: K-means on Oranges and Lemons(3 of 4)



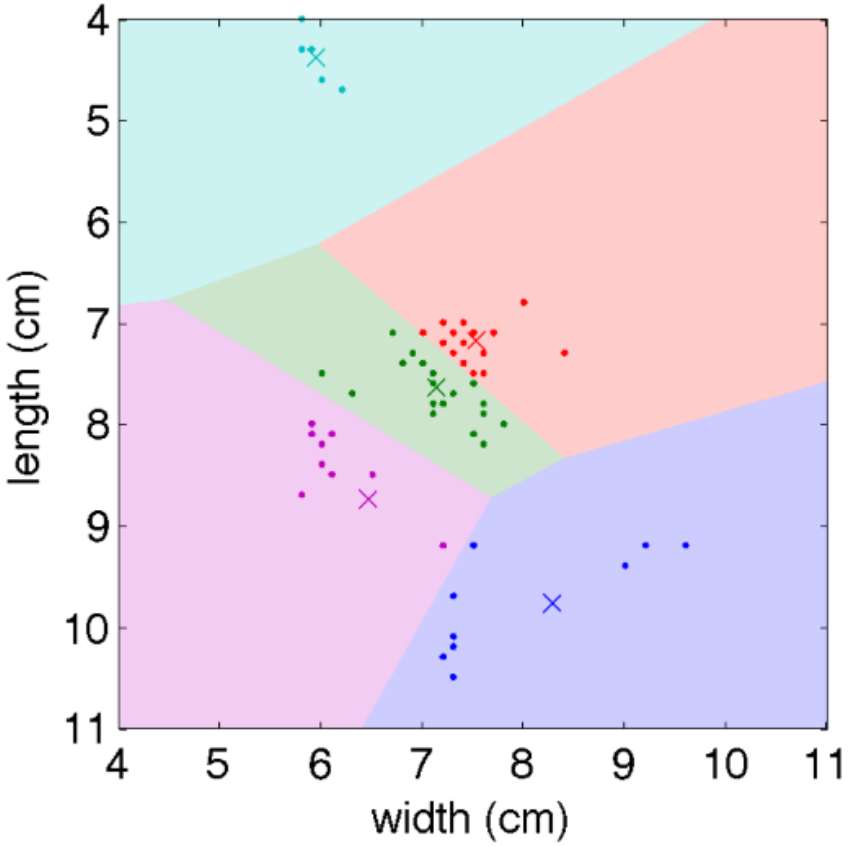
(c) Iteration 2



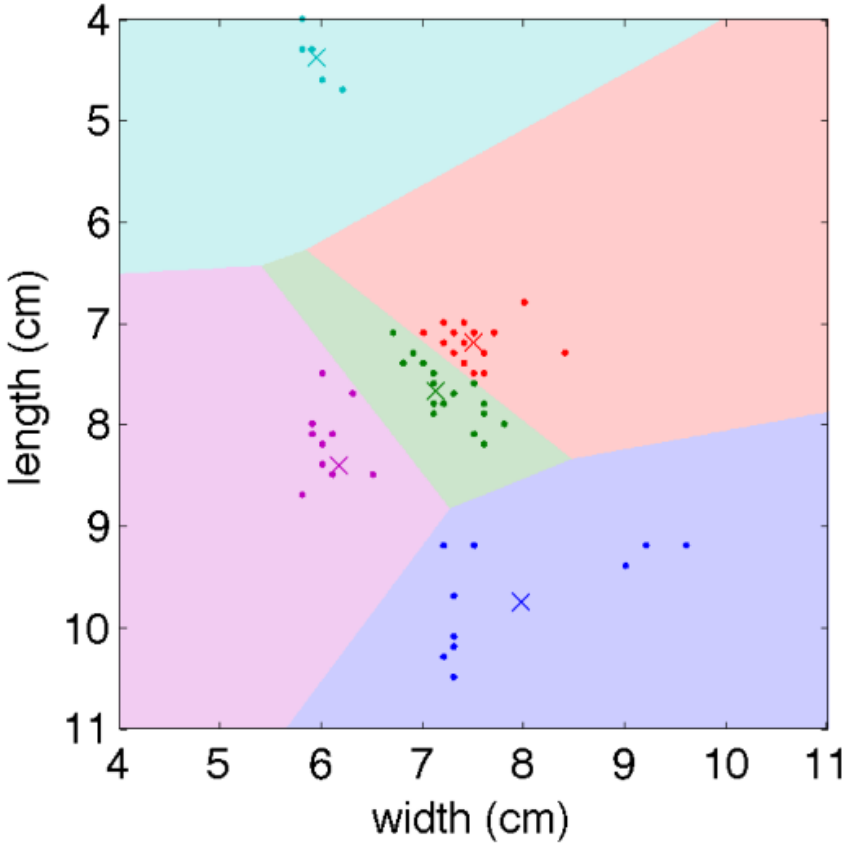
(d) Iteration 3

(K=5,lain Murray data)

Example: K-means on Oranges and Lemons(4 of 4)



(d) Iteration 3

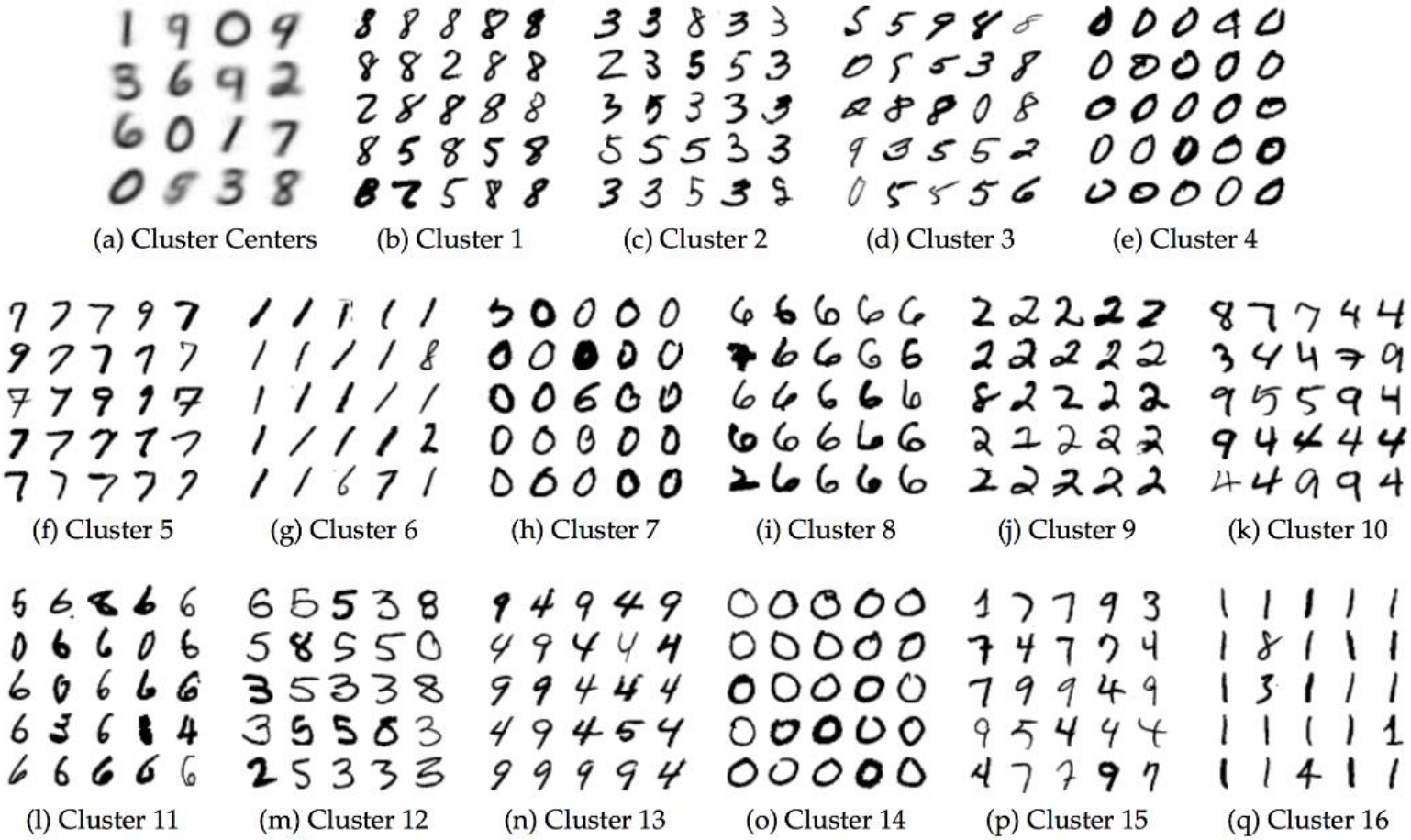


(e) Iteration 4

(K=5,lain Murray data)

Example: K-means Clustering on Handwritten Digits

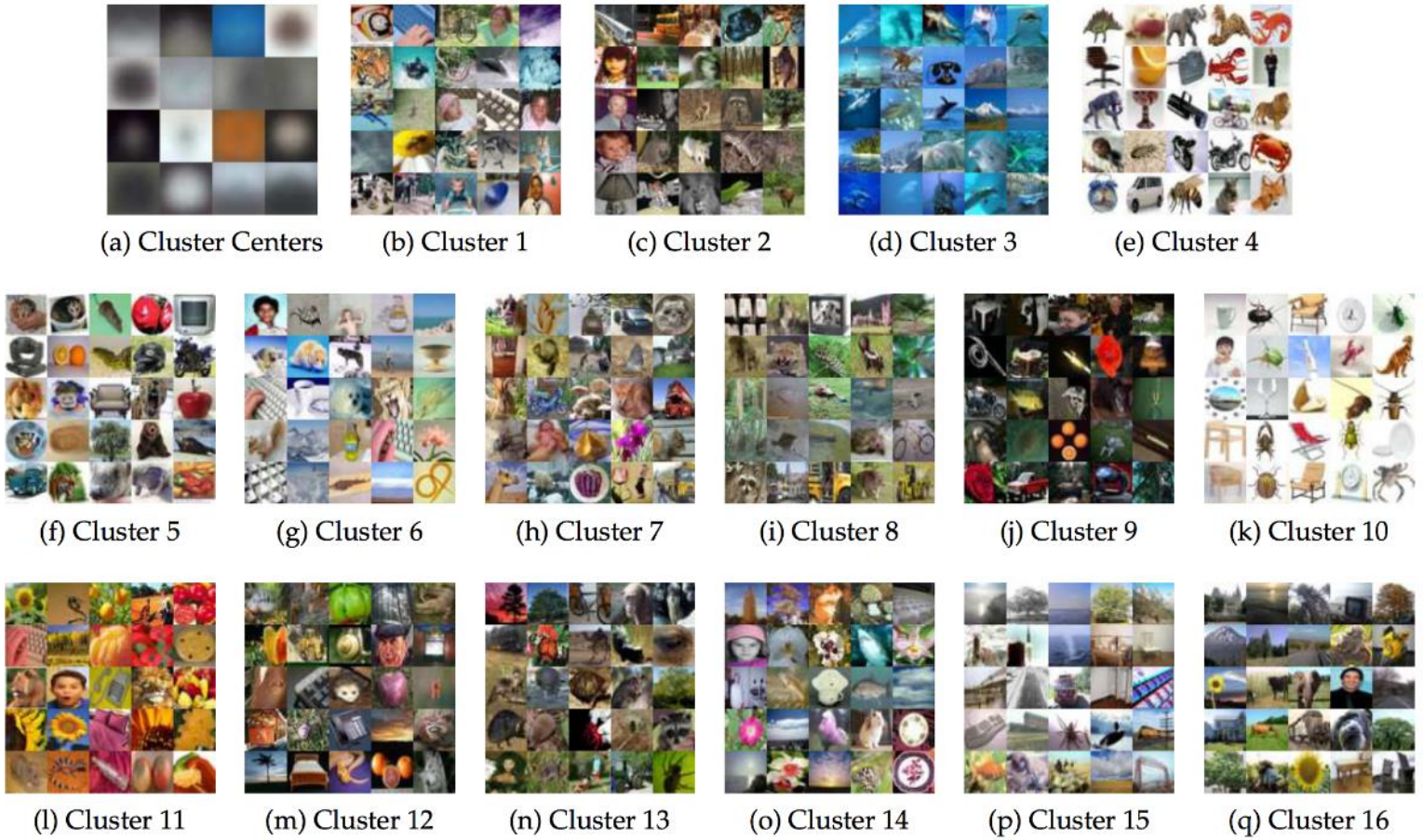
- MNIST: 60,000 digits. 28x28 grayscale



(K=16. Cluster pick up on similar stroke patterns)

Example: K-means Clustering on Image Data

- CIFAR-100 color. 50,000 images. 32x32x3 (RGB)



(K=16. Cluster pick up on low-freq color patterns)

Example: K-means Clustering on Documents

- 30,991 articles from Grolier's Encyclopedia. Articles are represented via a count vector of most common word (m = 15276)

Cluster 1	Cluster 2	Cluster 3	Cluster 4	Cluster 5	Cluster 6
education	south	war	war	art	light
united	population	german	government	century	energy
american	north	british	law	architecture	atoms
public	major	united	political	style	theory
world	west	president	power	painting	stars
social	mi	power	united	period	chemical
government	km	government	party	sculpture	elements
century	sq	army	world	form	electrons
schools	deg	germany	century	artists	hydrogen
countries	river	congress	military	forms	carbon

Cluster 7	Cluster 8	Cluster 9	Cluster 10	Cluster 11	Cluster 12
energy	god	century	city	population	cells
system	world	world	american	major	body
radio	century	water	century	km	blood
space	religion	called	world	mi	species
power	jesus	time	war	government	cell
systems	religious	system	john	deg	called
television	steel	form	life	sq	plants
water	philosophy	united	united	north	animals
solar	science	example	family	south	system
signal	history	life	called	country	human

(K=12)

Understanding Lloyd's Algorithm

- Loss function:

$$\boldsymbol{L}(\{\mathbf{r}\},\{\boldsymbol{\mu}\}) = \sum_{i=1}^{n}\sum_{k=1}^{K} r_{ik}\,\|\mathbf{x}_i - \boldsymbol{\mu}_k\|^2$$

- Solve this via coordinate descent:

  Step 1: Fix $\{\mathbf{r}\}$, update $\{\boldsymbol{\mu}\}$: minimize loss by assigning each sample to the cluster that is closest

  Step 2: Fix $\{\boldsymbol{\mu}\}$, update $\{\mathbf{r}\}$: work with squared distance

$$\boldsymbol{L} = \sum_{i=1}^{n}\sum_{k=1}^{K} r_{ik}\,(\mathbf{x}_i - \boldsymbol{\mu}_k)^{\mathrm{T}}(\mathbf{x}_i - \boldsymbol{\mu}_k)$$

$$\frac{\partial \boldsymbol{L}}{\partial \boldsymbol{\mu}_k} = -2\sum_{i=1}^{n} r_{ik}\,(\mathbf{x}_i - \boldsymbol{\mu}_k) = 0$$

$$\leftrightarrow \sum_{i=1}^{n} r_{ik}\,\mathbf{x}_i = \boldsymbol{\mu}_k \sum_{i=1}^{n} r_{ik} \leftrightarrow \boldsymbol{\mu}_k = \frac{\sum_{i=1}^{n} r_{ik}\mathbf{x}_i}{\sum_{i=1}^{n} r_{ik}}$$

K-means Clustering

- Simple and popular method
- The effective number of parameters is $m \times K$, where m is sample dimension, K is number of clusters
- Not useful if linear decision boundaries fail

K-means Clustering

Computational complexity:

- Assignment step is $\boldsymbol{O}(nKm)$, since it compares each sample with K centers
- Centroid update is $\boldsymbol{O}(nm)$, since it calculates the centroid of each cluster
- $\boldsymbol{O}(nKmT)$ time over T iterations (generally $T << n$)

How to Set the Number of Cluster K

- **Smaller**: may provide better interpretation
- **Larger**: useful if clustering is being used for feature extraction
- No principled way to do this. A heuristic approach is to plot loss against K, and look for a "Knee" in the plot

Contents

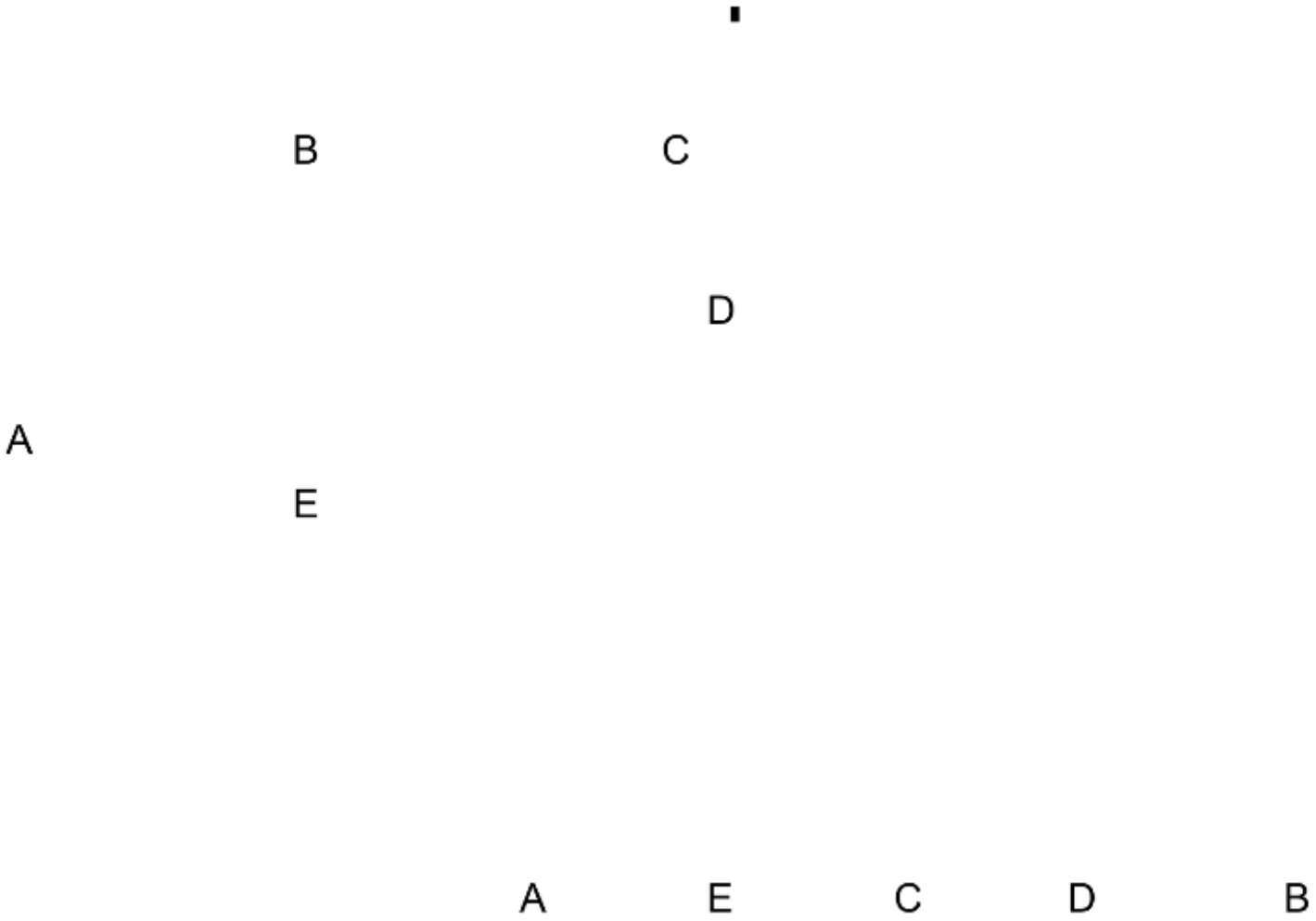
Hierarchical Agglomerative Clustering (HAC)

HAC will work by maintaining an "active set" of clusters, and repeatedly merging cluster. (Forming a tree)

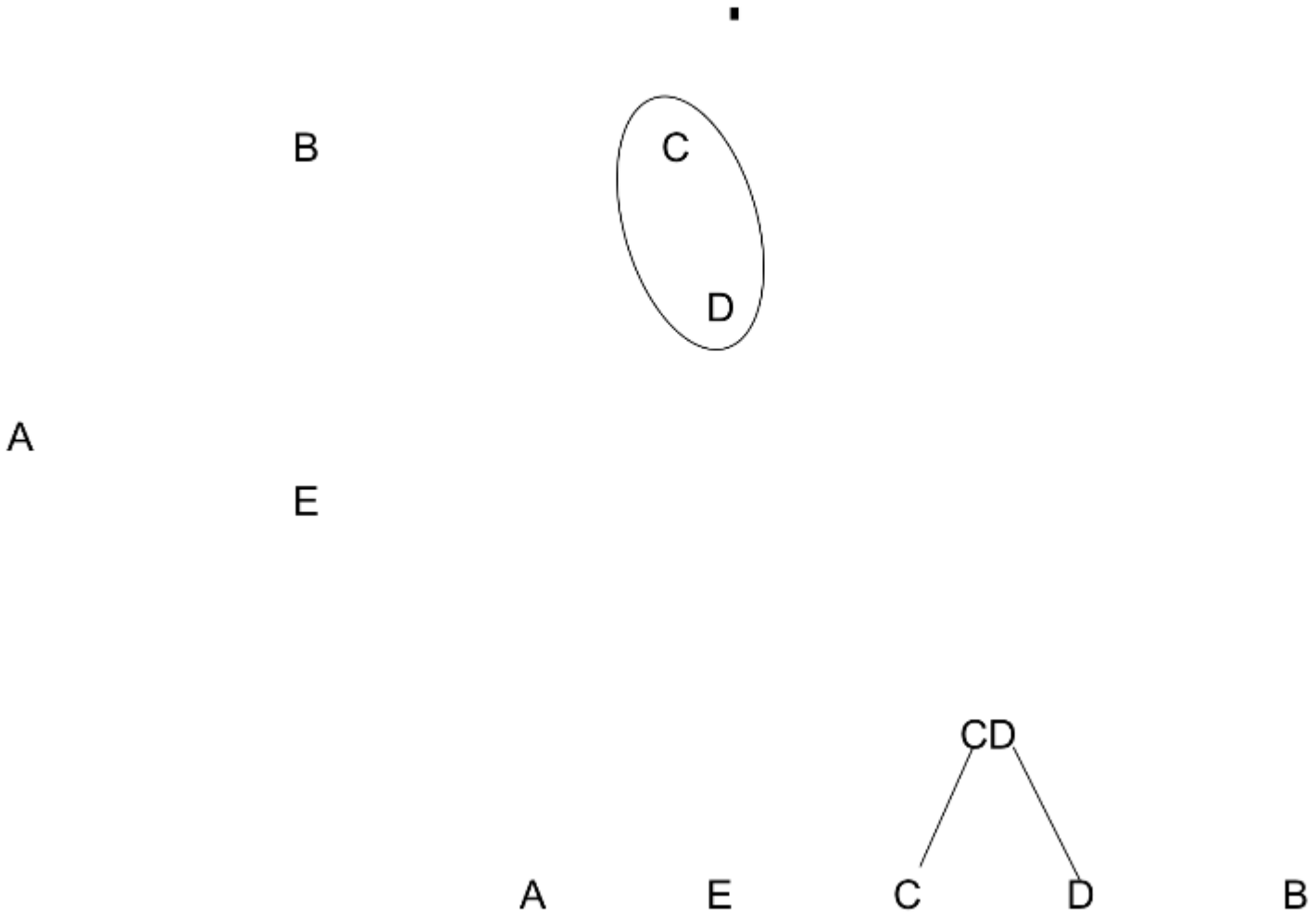
Compared with K-Means:

- Non parametric (instance based). Because of this, more flexible than K-means.
- Can generate arbitrary cluster shapes. (Can also over-fit!)
- Rather than a flat partition of data, it generates a hierarchy of clusters
- No need to specify the number of clusters up front
- Not randomized (This can be a problem with K-means)

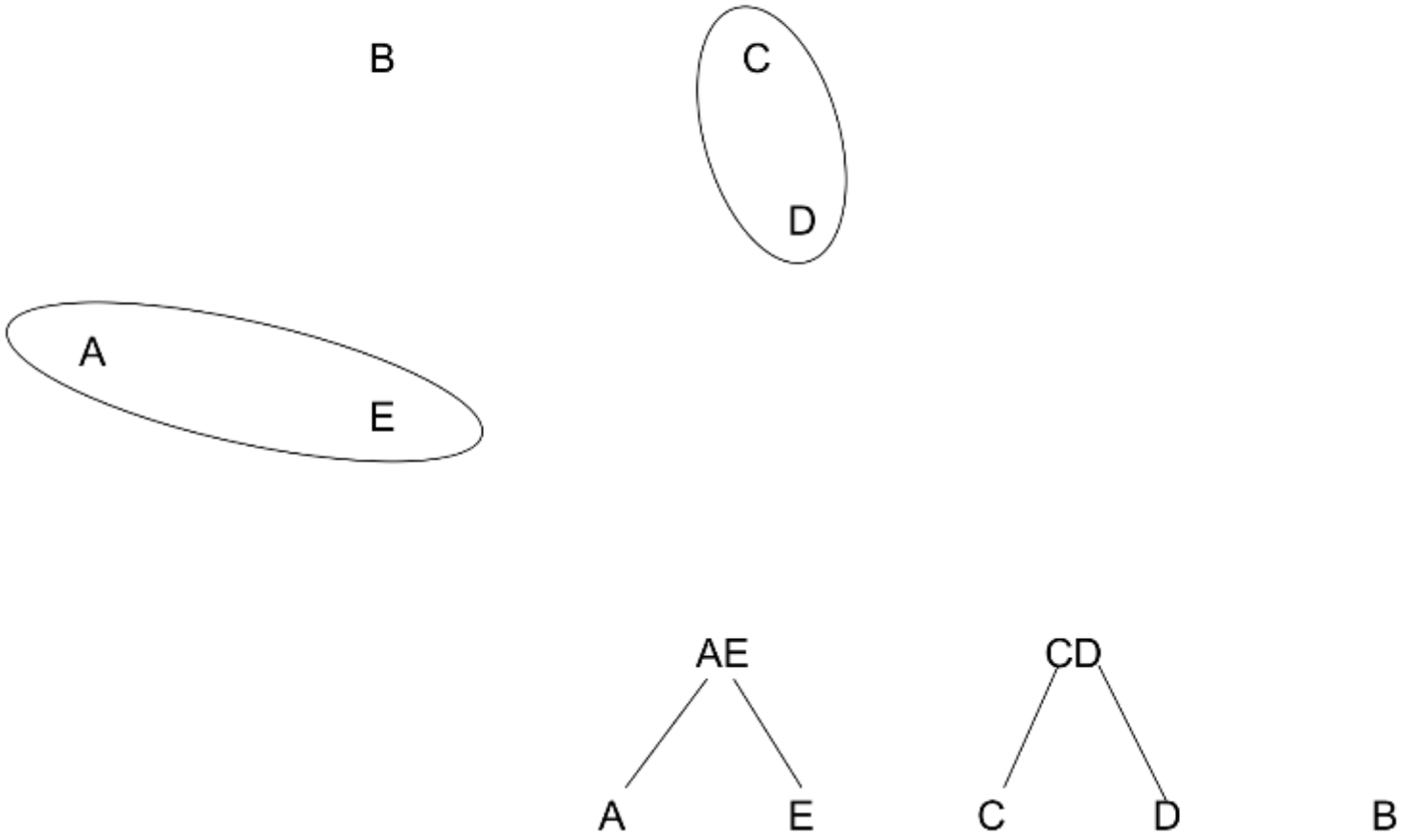
HAC Example (1 of 5)



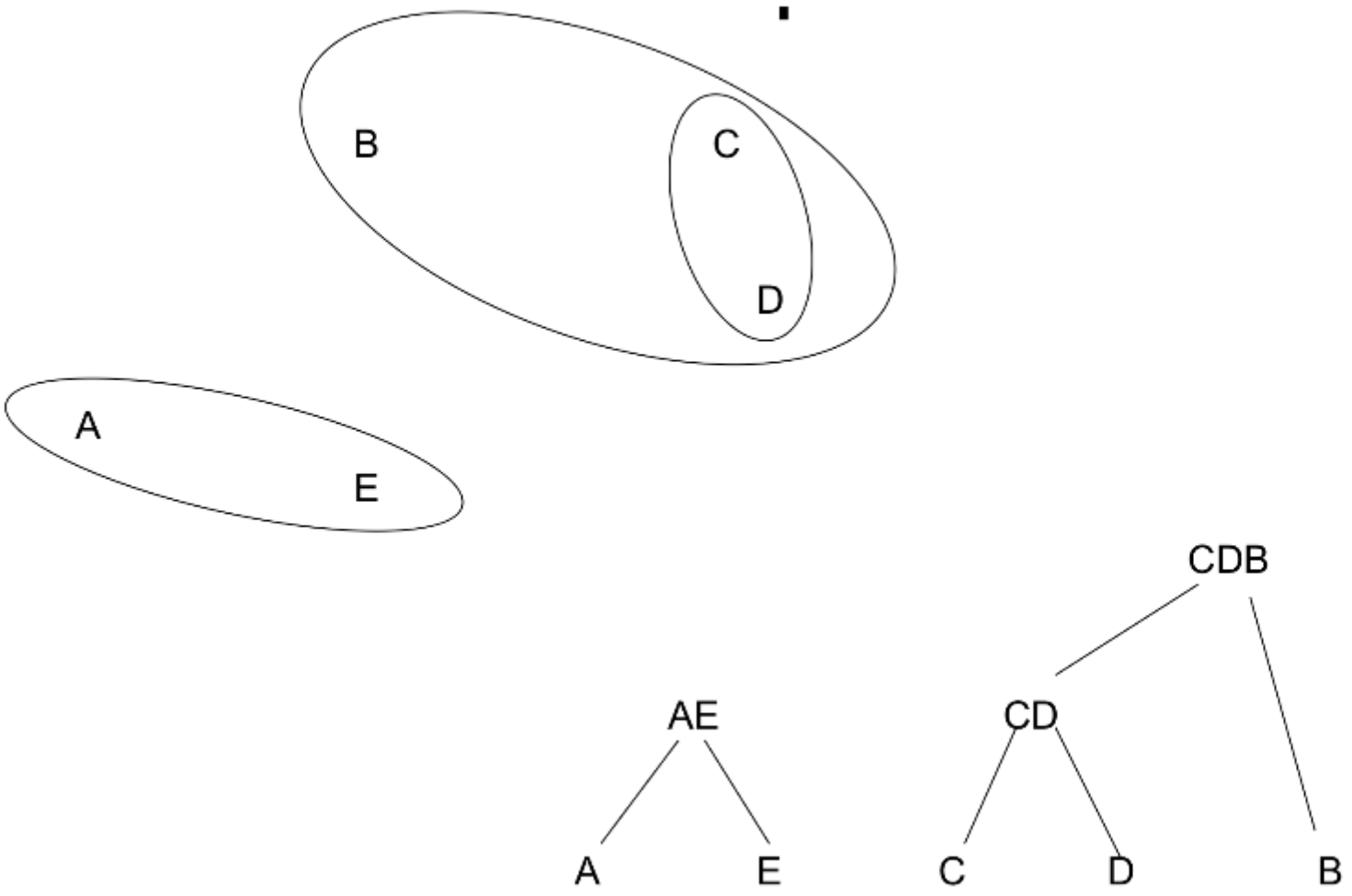
HAC Example (2 of 5)



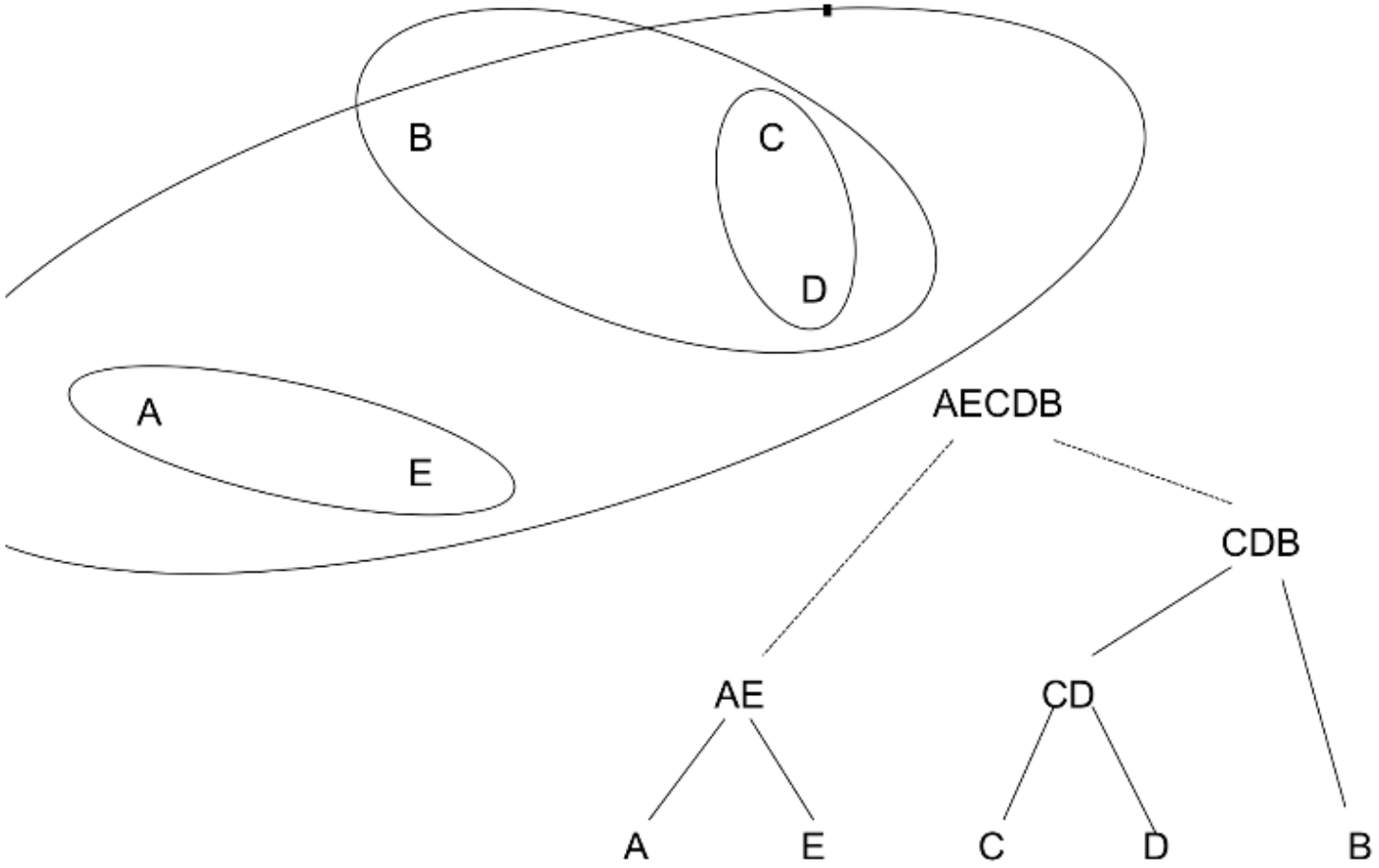
HAC Example (3 of 5)



HAC Example (4 of 5)



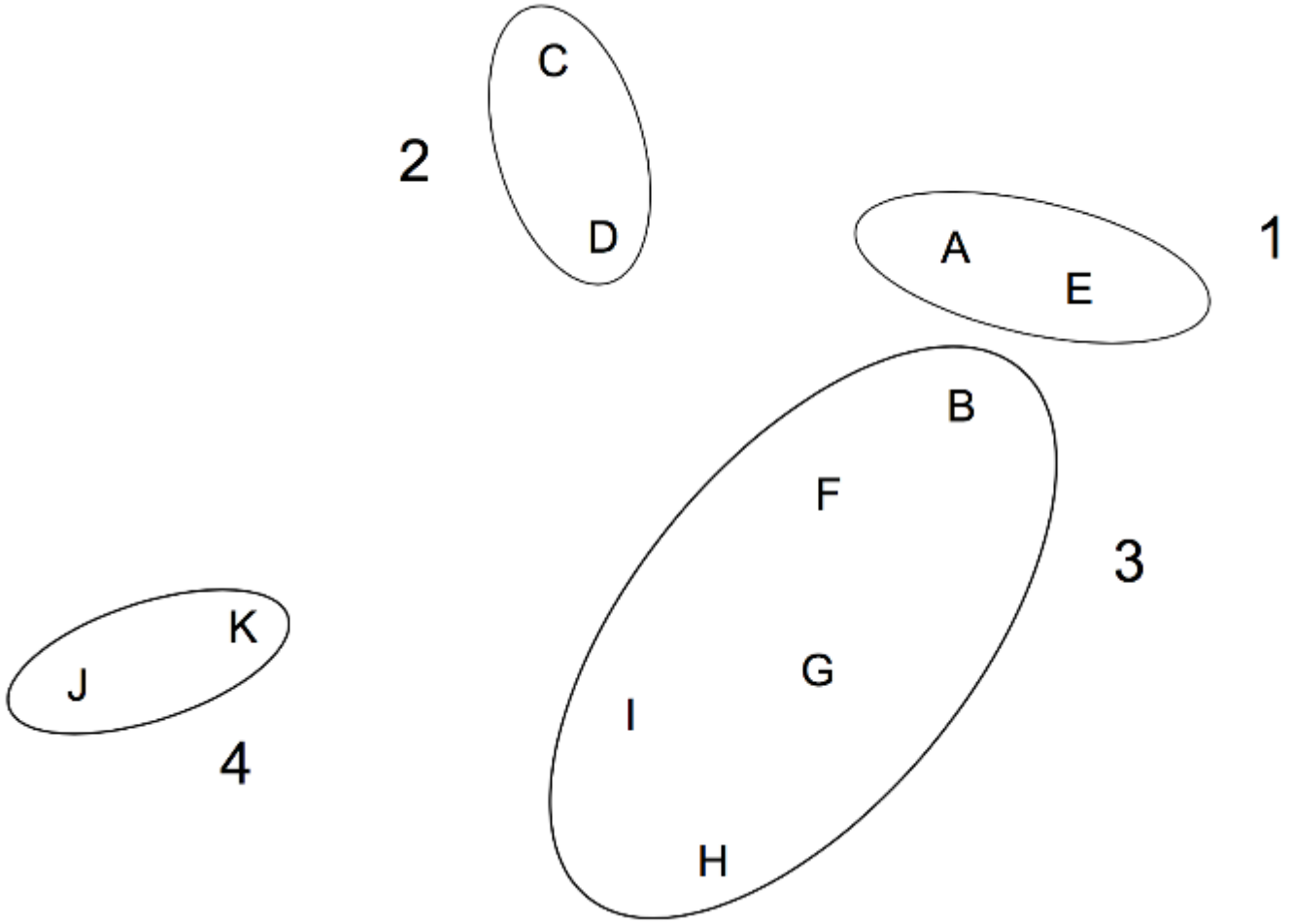
HAC Example (5 of 5)



HAC Algorithm

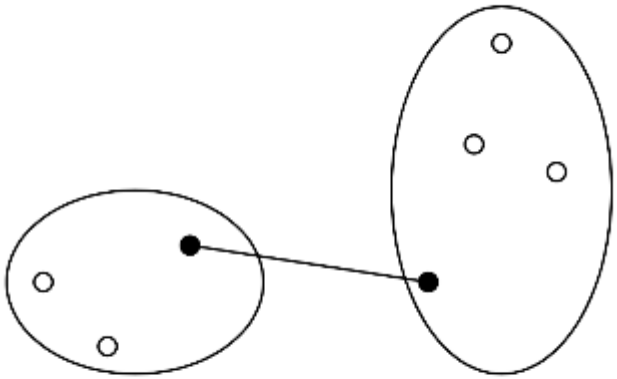
- Treat each sample as a cluster
- Merge two “closest” clusters
- Repeat until most clusters are merged

What is Closest to Cluster?

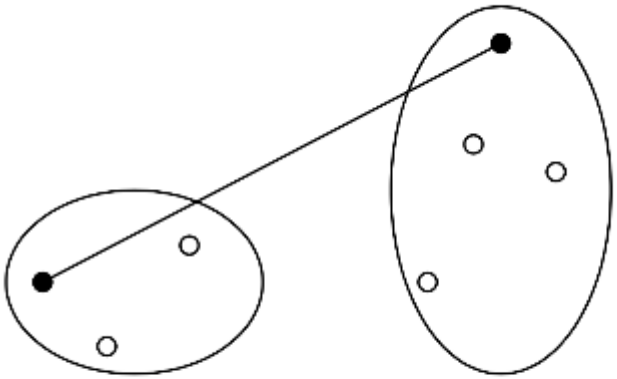


Max? Min? Average? Centroid?

HAC: Min and Max Group Distance



(a) min distance

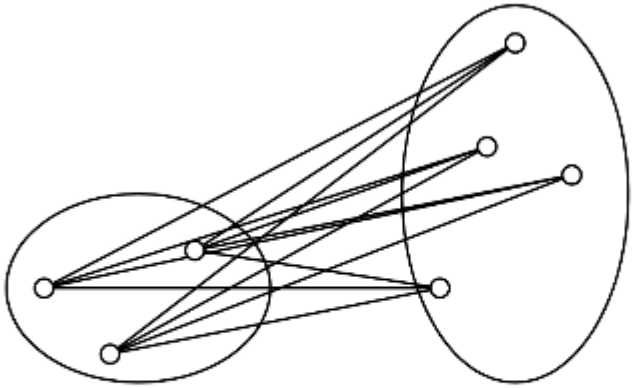


(b) max distance

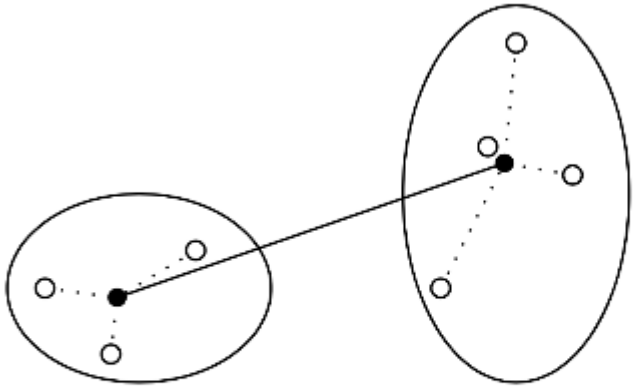
$$\min d(\boldsymbol{G}, \widehat{\boldsymbol{G}}) = \min_{\mathbf{x}\in\boldsymbol{G}, \hat{\mathbf{x}}\in\widehat{\boldsymbol{G}}} \|\mathbf{x} - \hat{\mathbf{x}}\|$$

$$\max d(\boldsymbol{G}, \widehat{\boldsymbol{G}}) = \max_{\mathbf{x}\in\boldsymbol{G}, \hat{\mathbf{x}}\in\widehat{\boldsymbol{G}}} \|\mathbf{x} - \hat{\mathbf{x}}\|$$

HAC: Average and Centroid Group Distances



(c) average distance

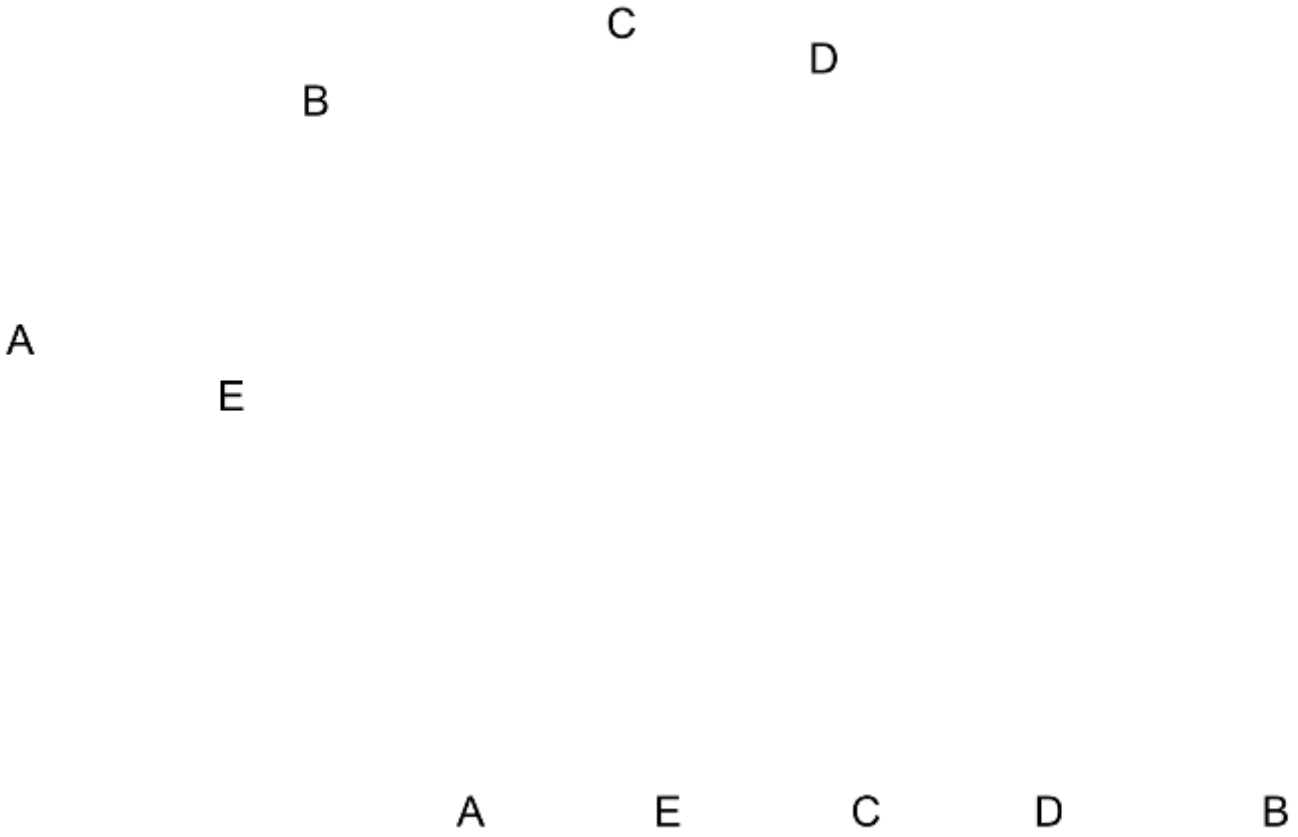


(d) centroid distance

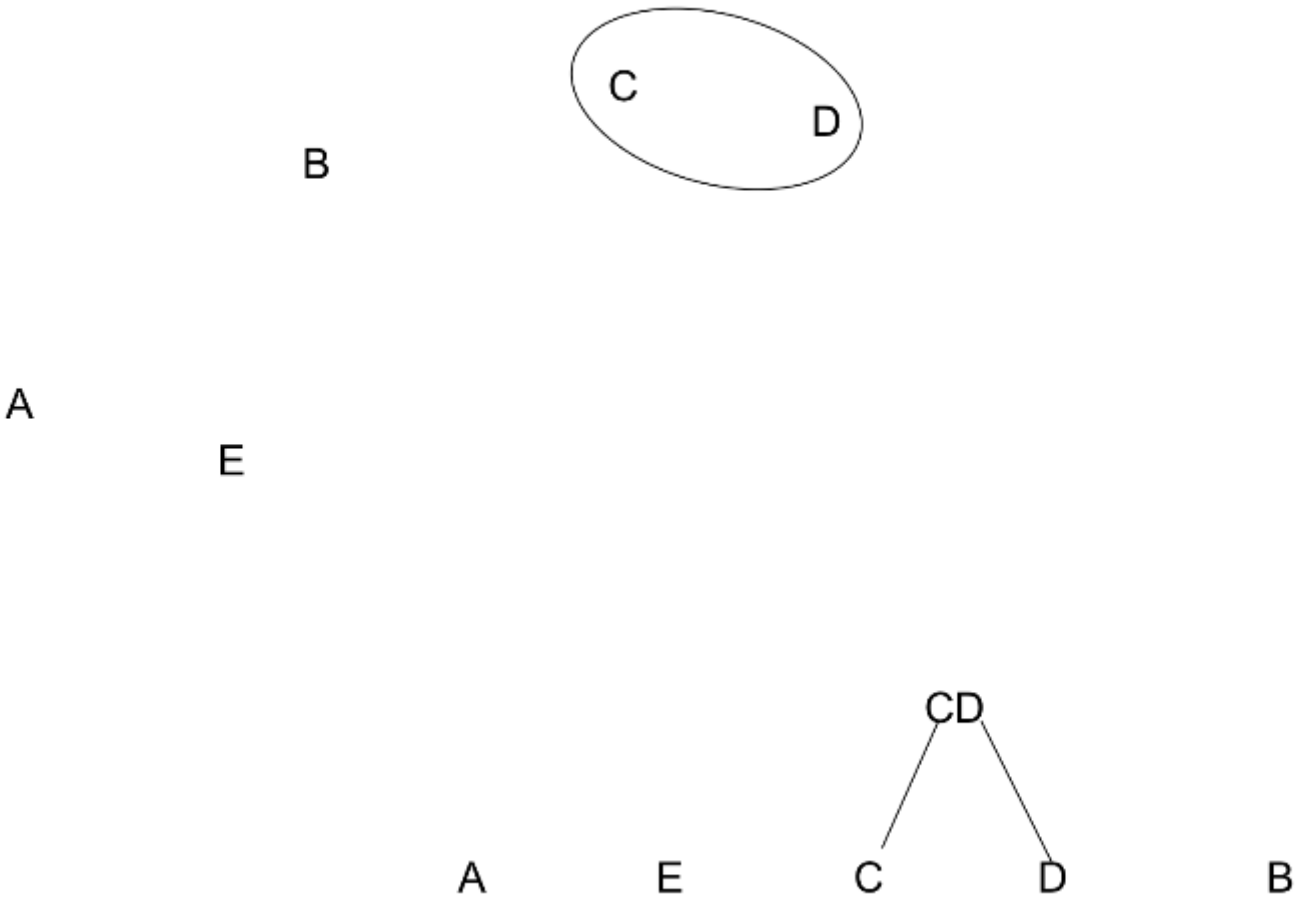
$$\min d(\boldsymbol{G}, \widehat{\boldsymbol{G}}) = \frac{1}{|\boldsymbol{G}||\widehat{\boldsymbol{G}}|} \sum_{\mathbf{x} \in \boldsymbol{G}, \hat{\mathbf{x}} \in \widehat{\boldsymbol{G}}} |\mathbf{x} - \hat{\mathbf{x}}|$$

$$d_{centroid}(\boldsymbol{G}, \widehat{\boldsymbol{G}}) = \|\boldsymbol{\mu} - \hat{\boldsymbol{\mu}}_{\boldsymbol{G}}\|$$

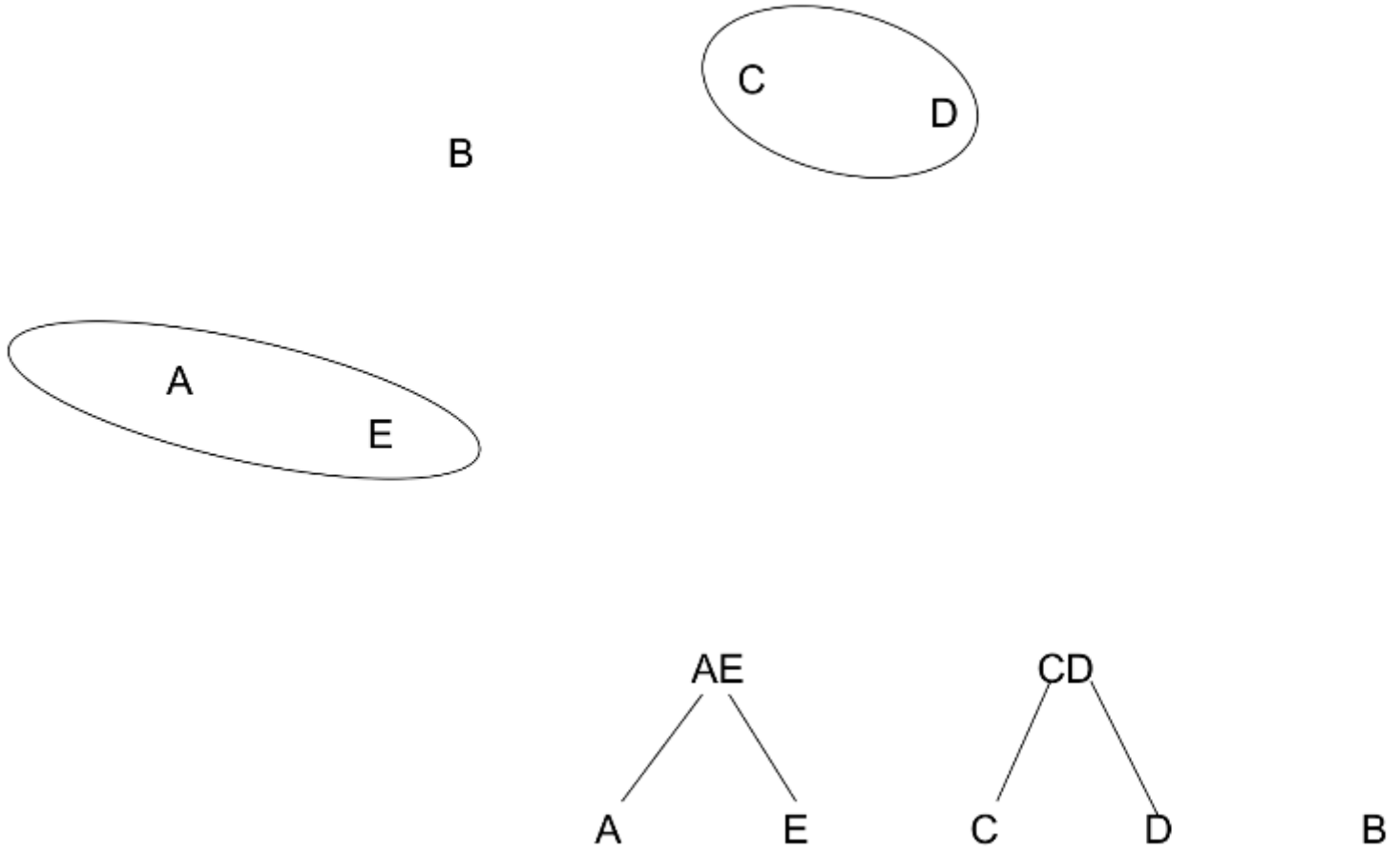
Example: HAC with Min Distance (1 of 6)



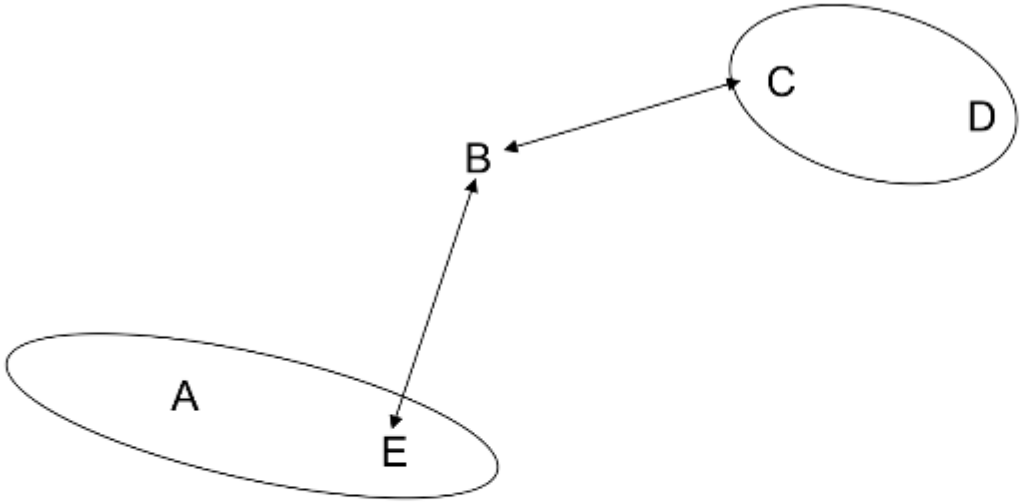
Example: HAC with Min Distance (2 of 6)

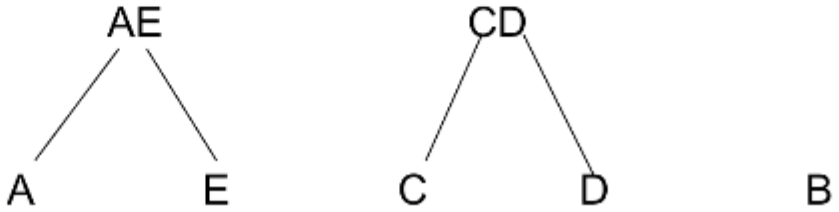


Example: HAC with Min Distance (3 of 6)

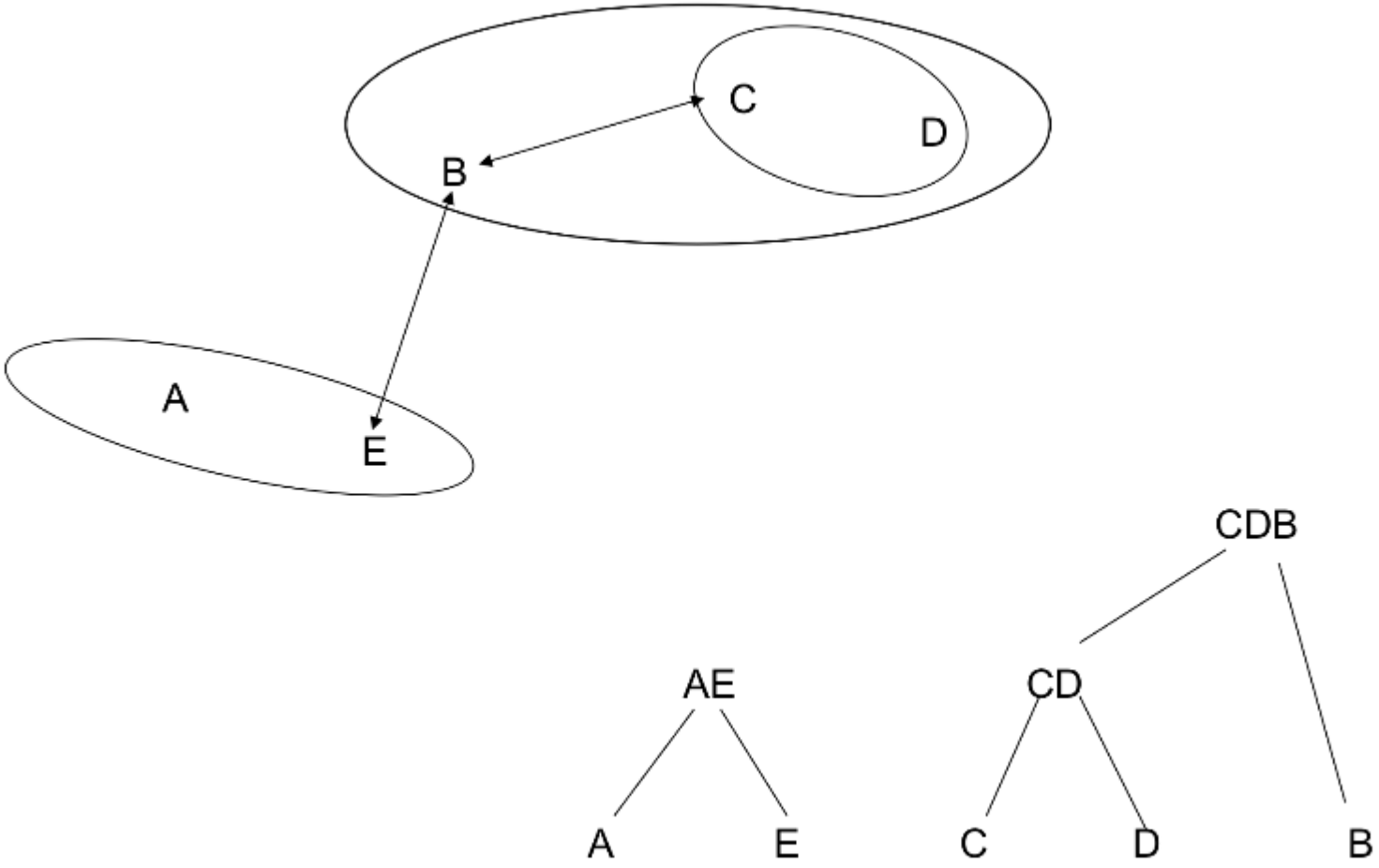


Example: HAC with Min Distance (4 of 6)

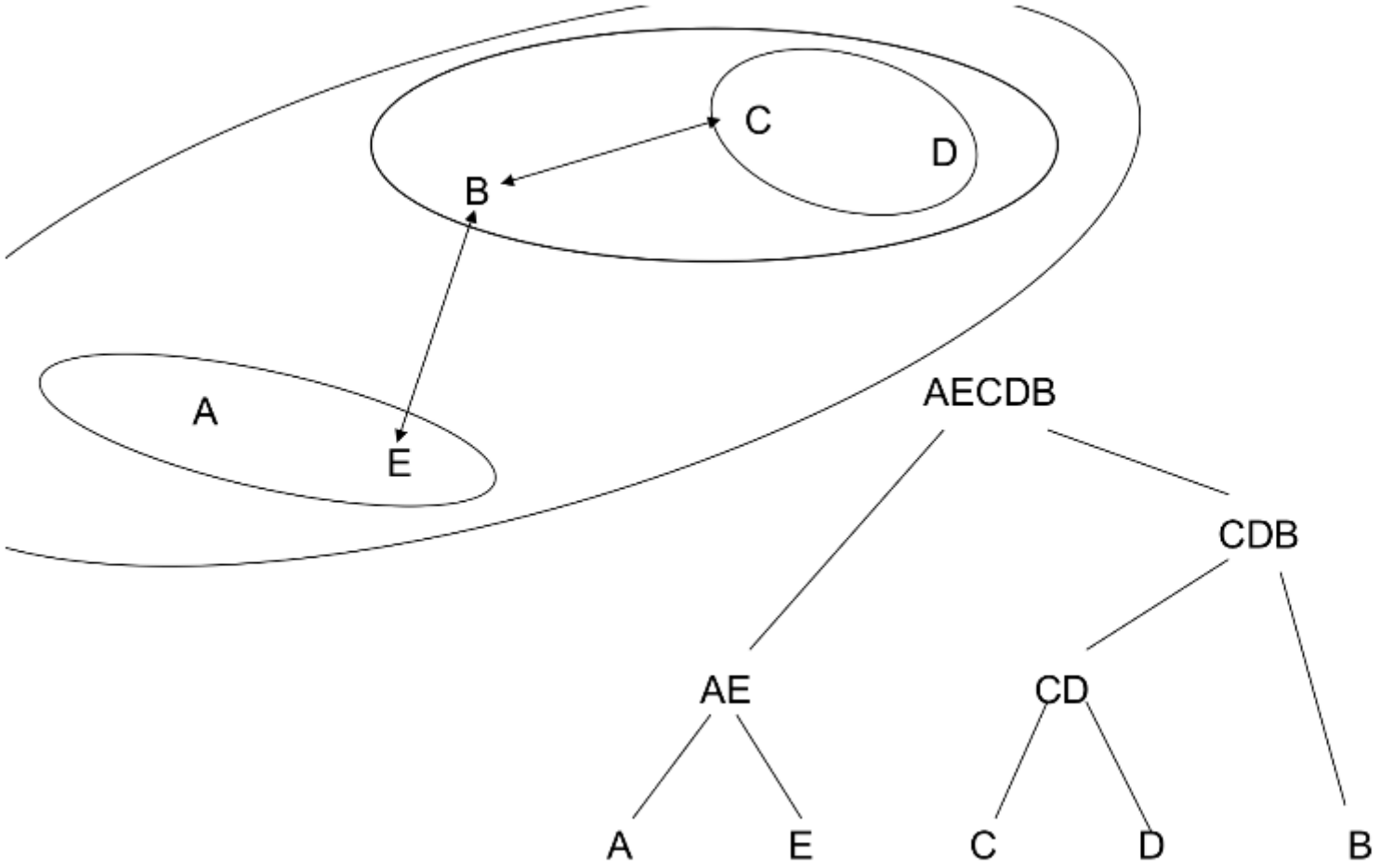




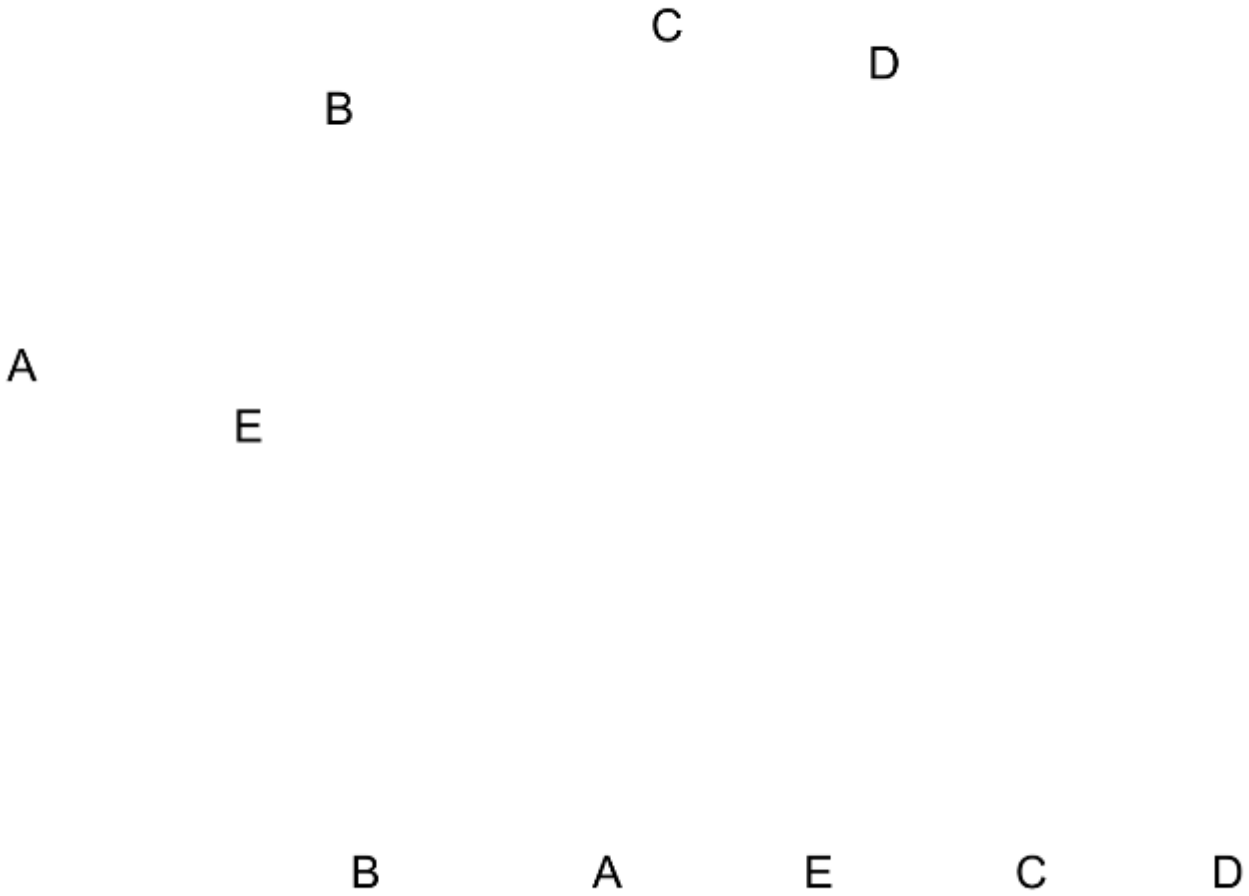
Example: HAC with Min Distance (5 of 6)



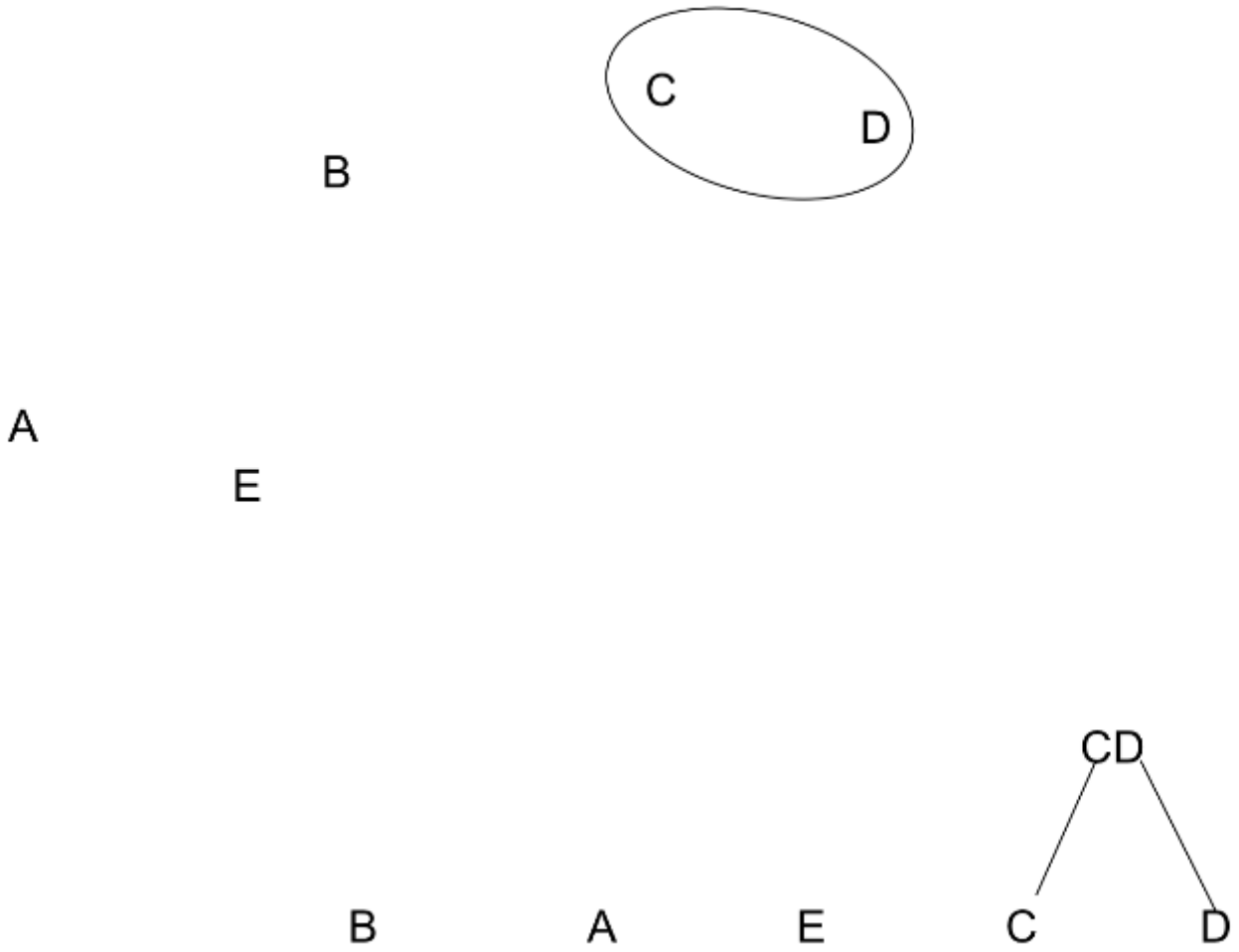
Example: HAC with Min Distance (6 of 6)



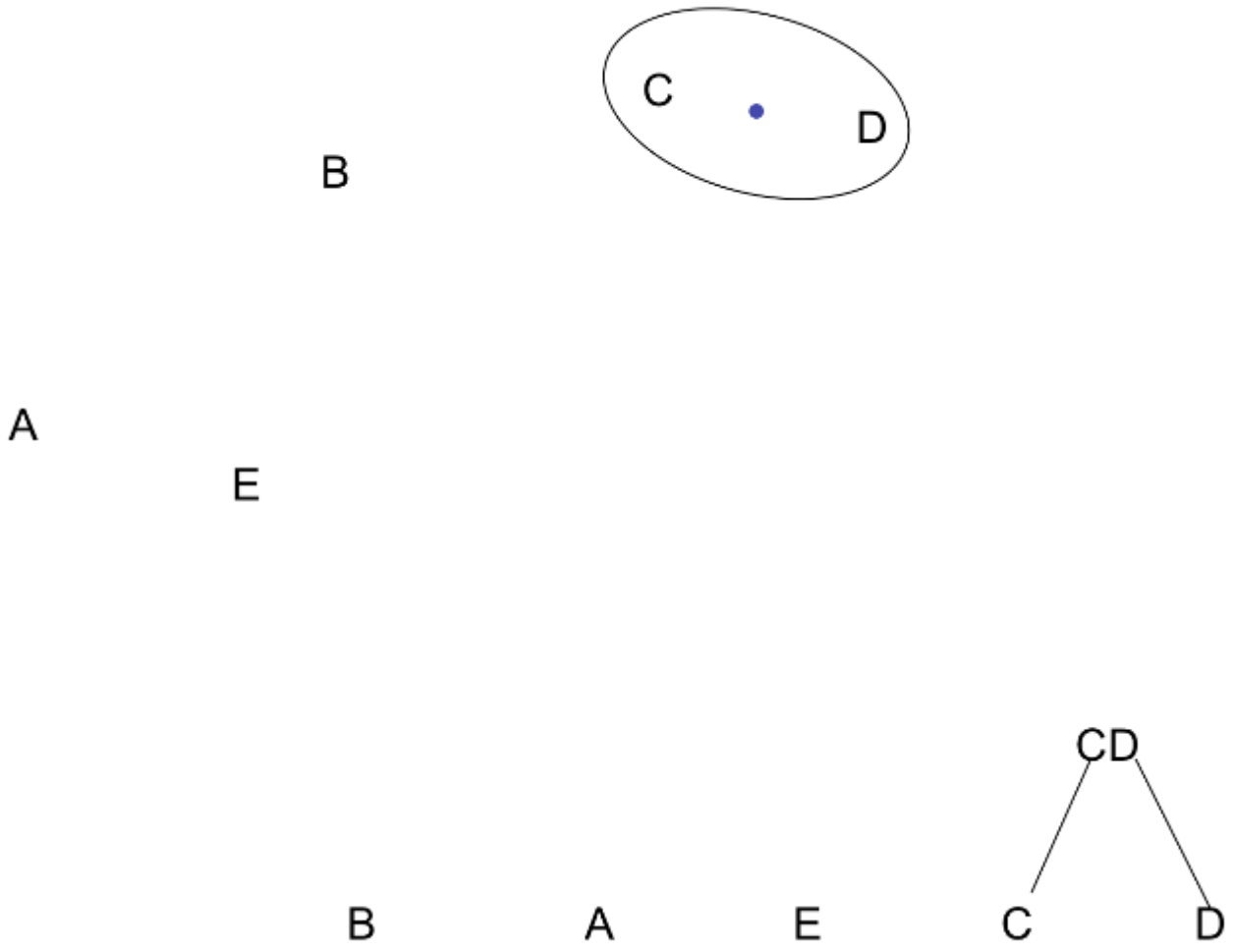
Example: HAC with Centroid Distance (1 of 7)



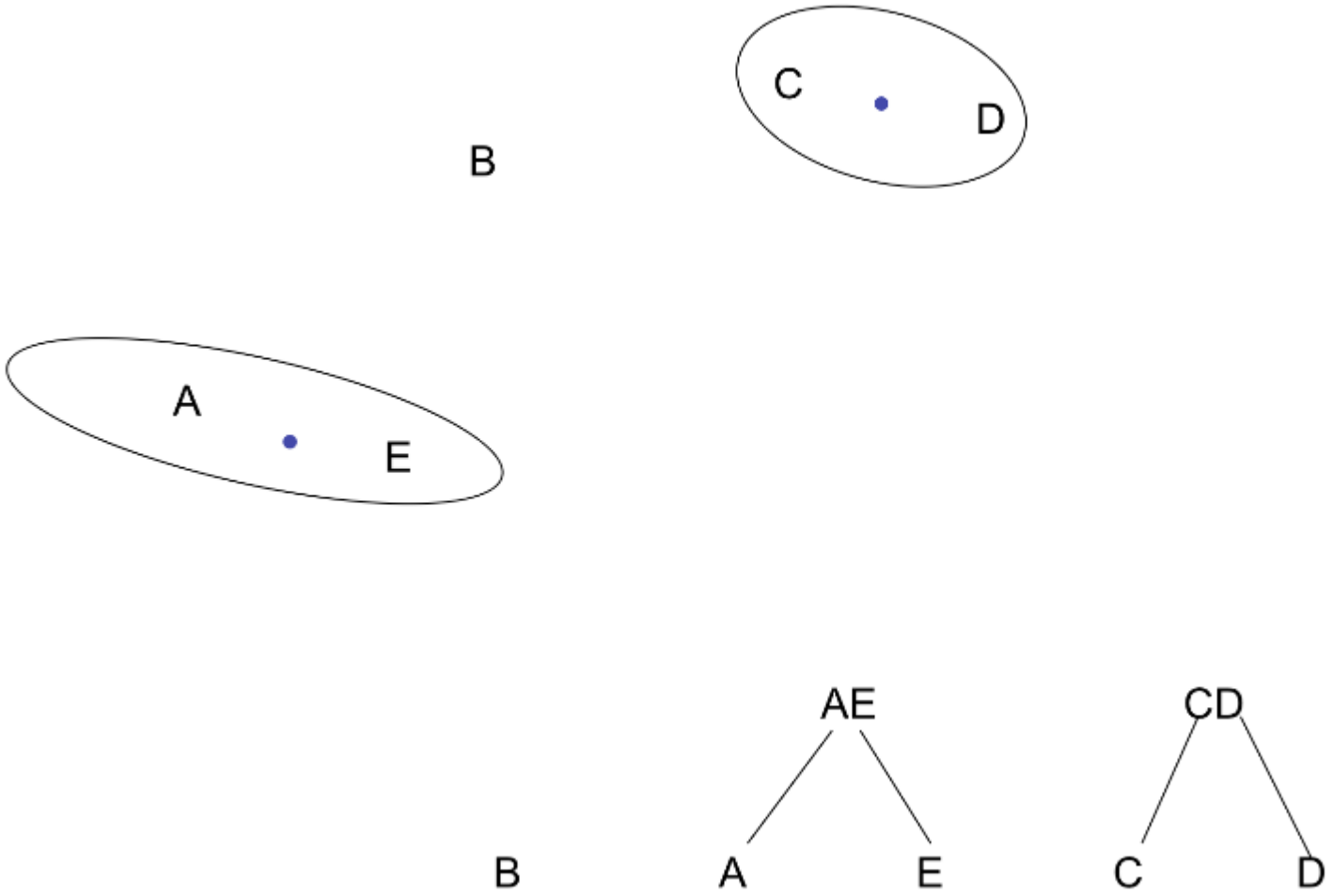
Example: HAC with Centroid Distance (2 of 7)



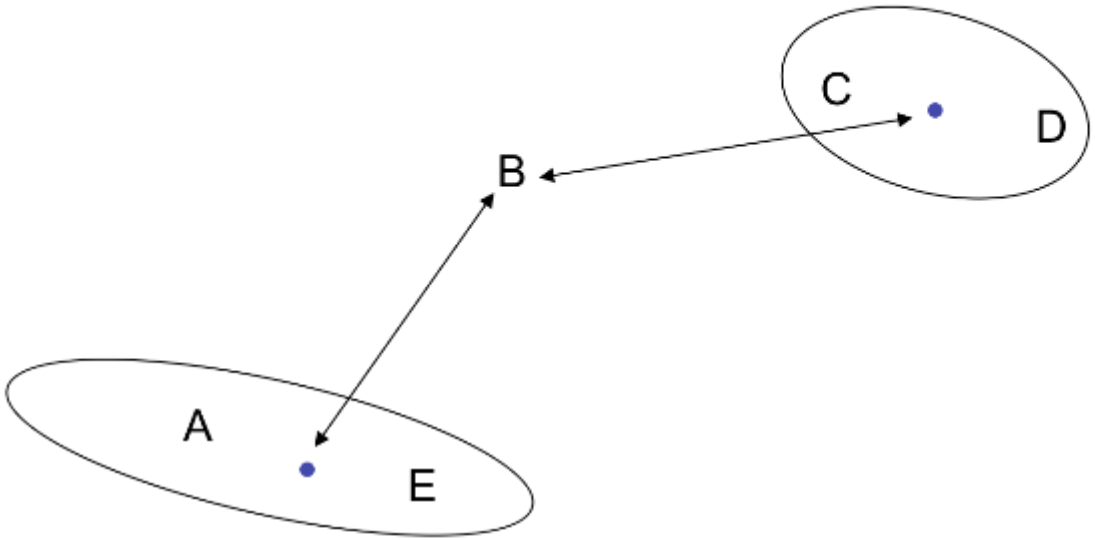
Example: HAC with Centroid Distance (3 of 7)

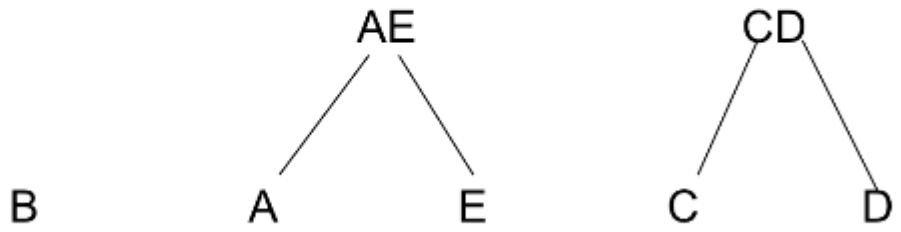


Example: HAC with Centroid Distance (4 of 7)

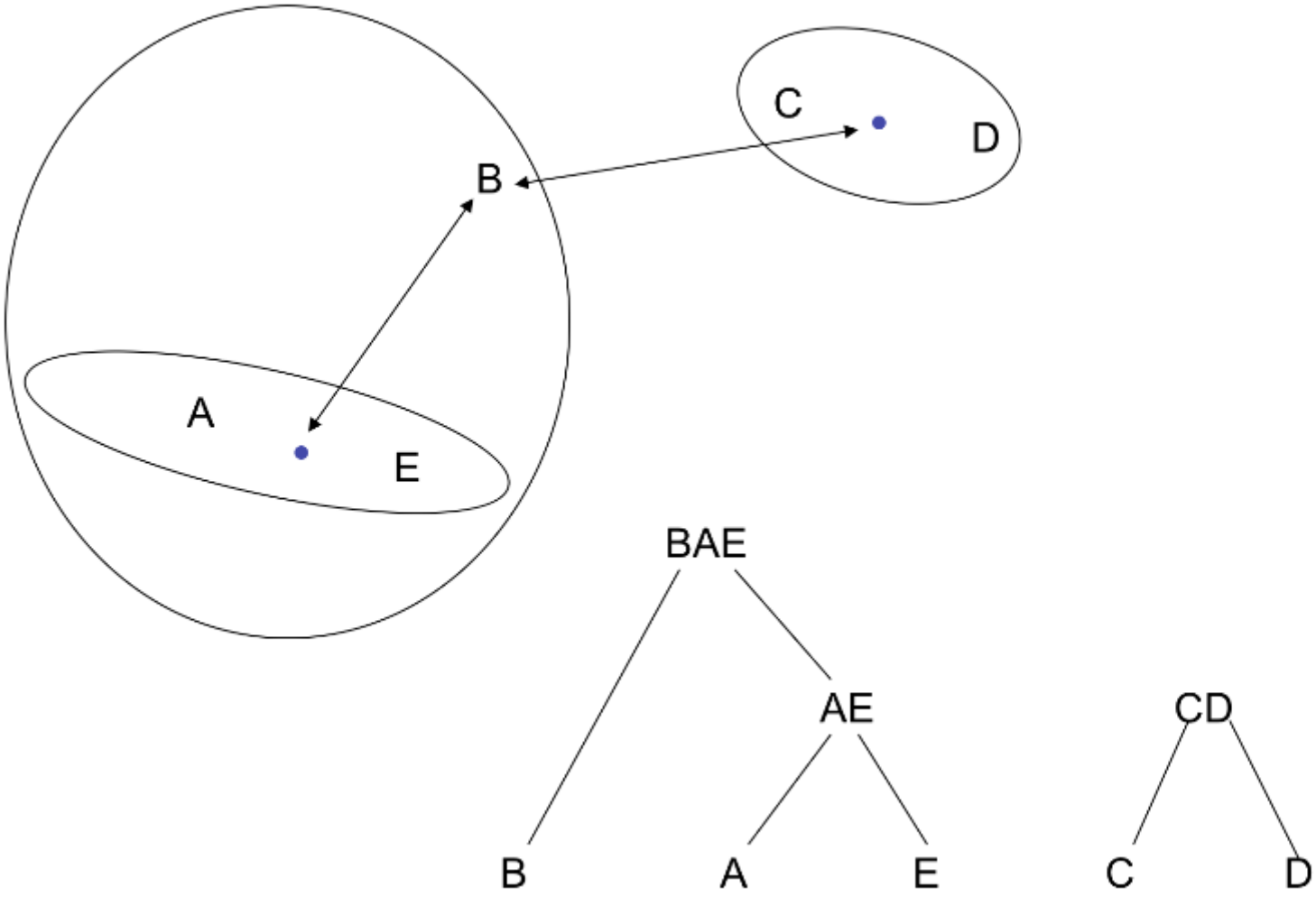


Example: HAC with Centroid Distance (5 of 7)

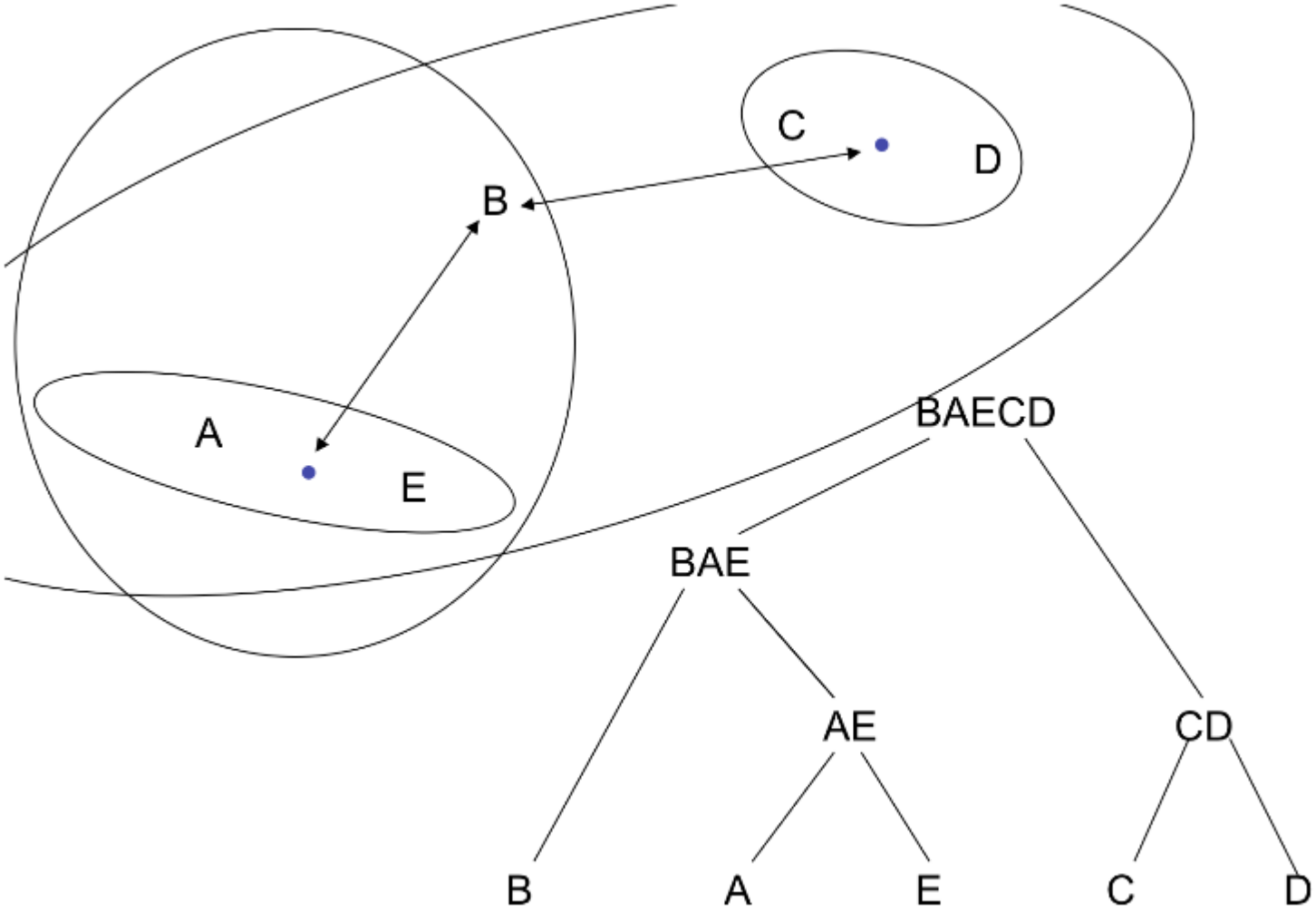




Example: HAC with Centroid Distance (6 of 7)



Example: HAC with Centroid Distance (7 of 7)



Comparing HAC Group Distance Criteria

- Which distance will tend to merge large clusters with each other?

A: min, because larger clusters are more likely to have a pair of sample that are close

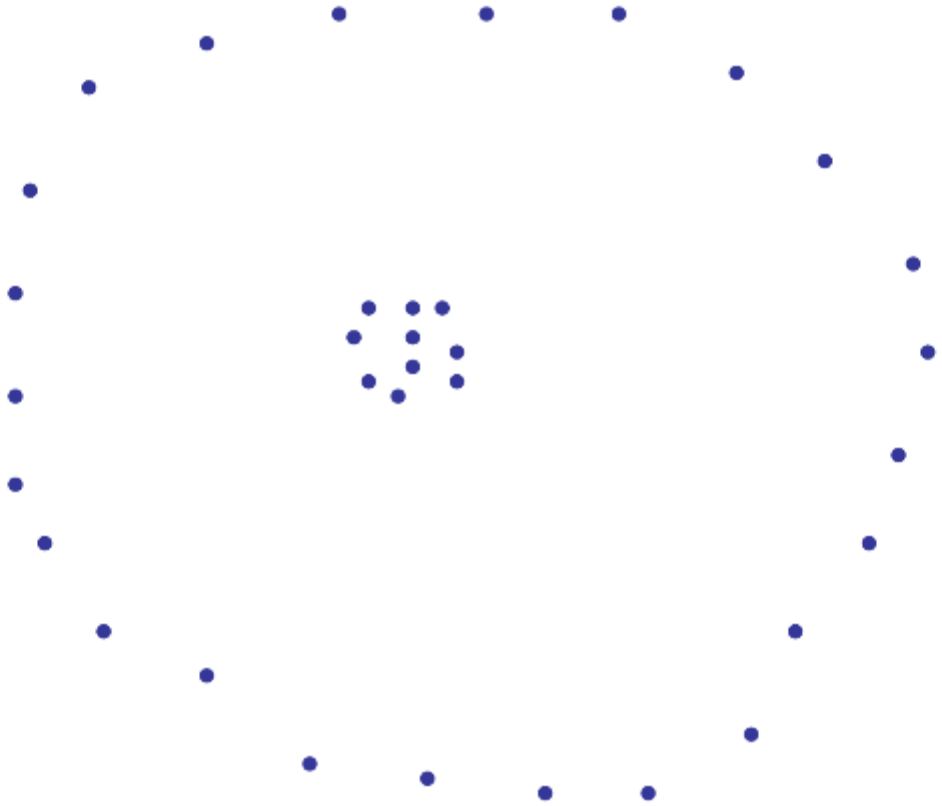
- Which distance will tend to have a "chain effect" and lead to long, string clusters?

A: min, since only one distance has to be small

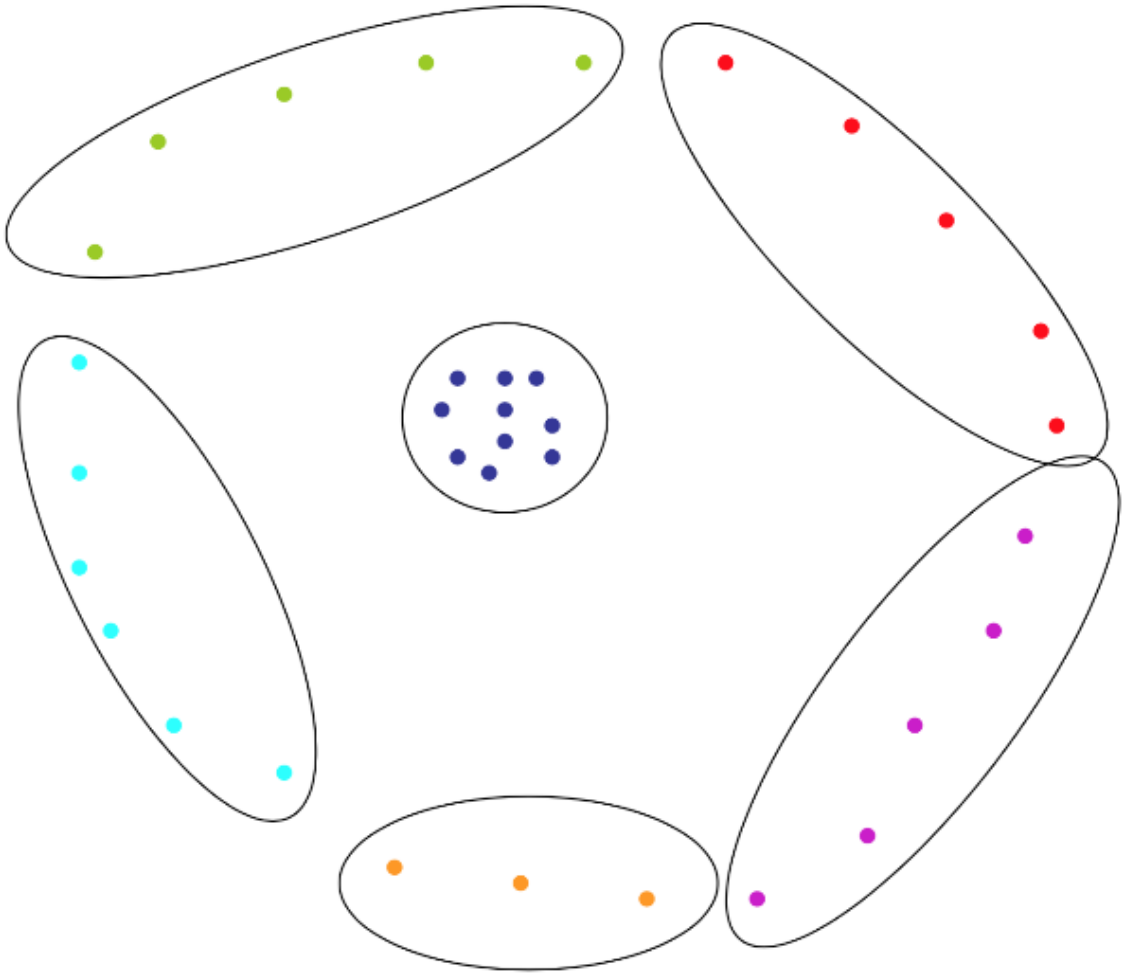
- Which distance will tend to prefer compact cluster?

A: max, since all distance have to be small to merge

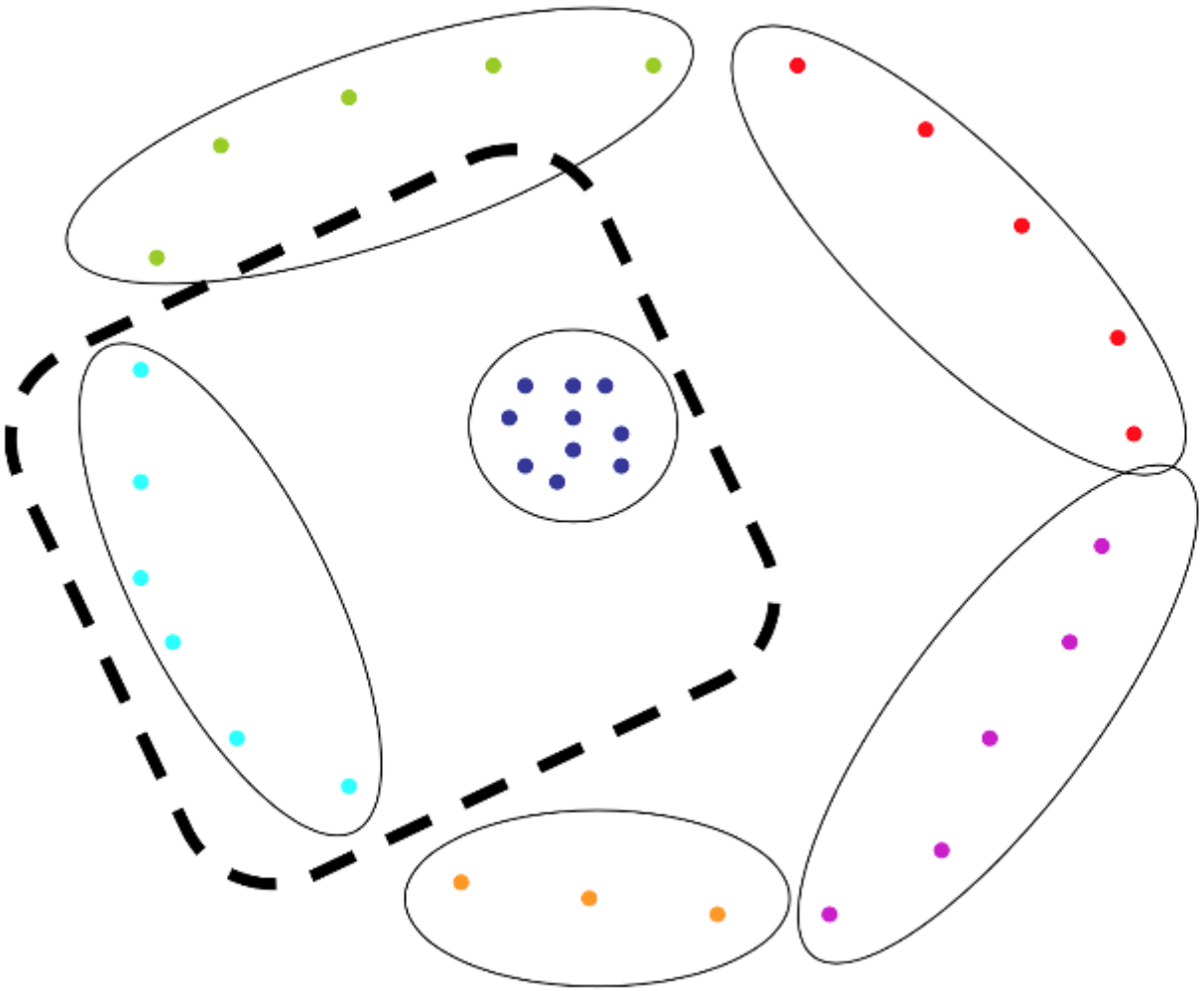
Simple HAC Example



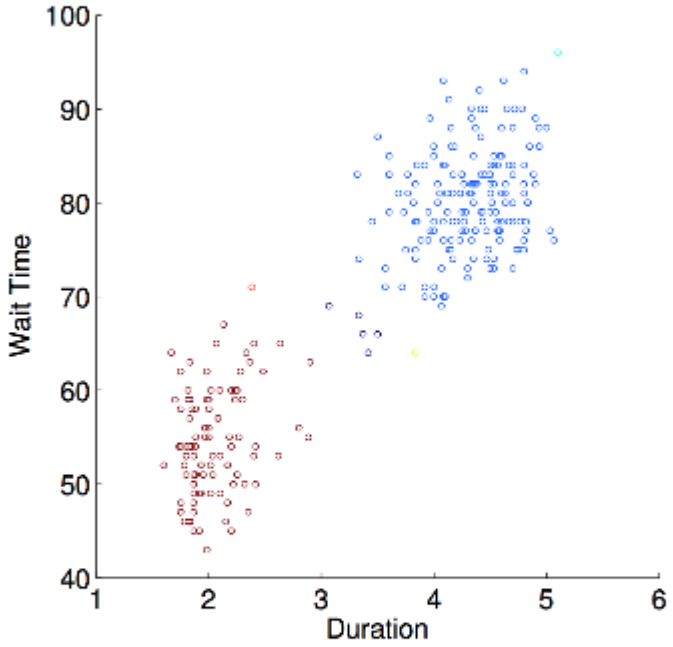
Simple HAC Example Max (1 of 2)



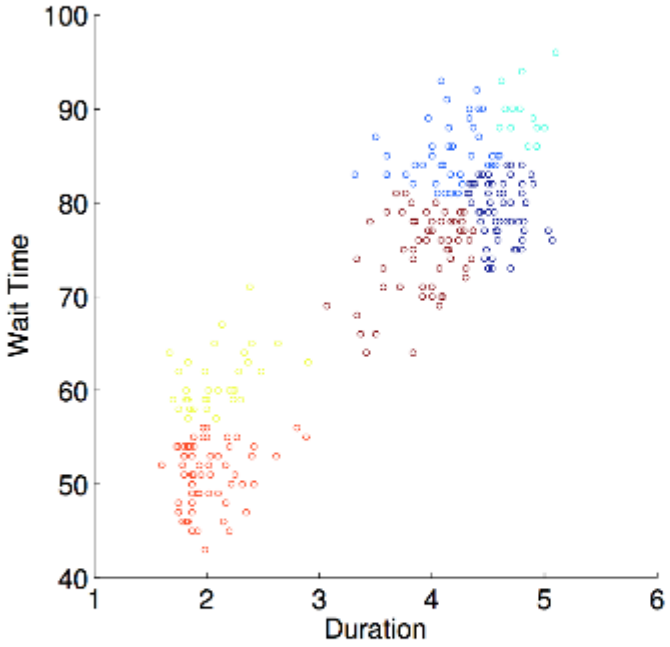
Simple HAC Example Max (2 of 2)



Example: HAC on Old Faithful Eruptions (1 of 2)

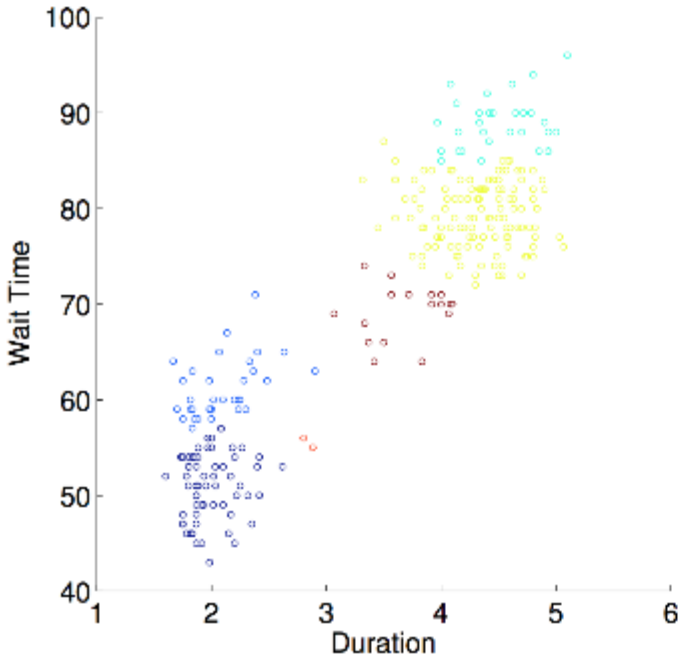


(a) min distance

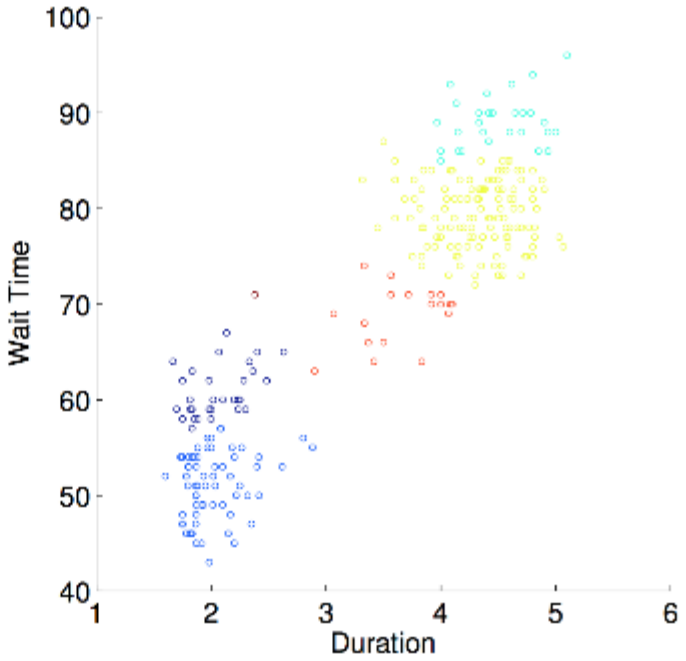


(b) max distance

Example: HAC on Old Faithful Eruptions (2 of 2)

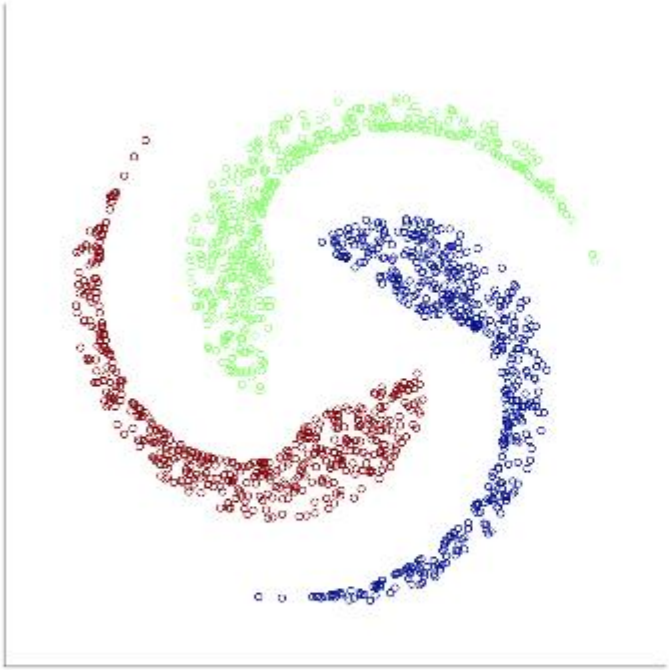


(c) average distance

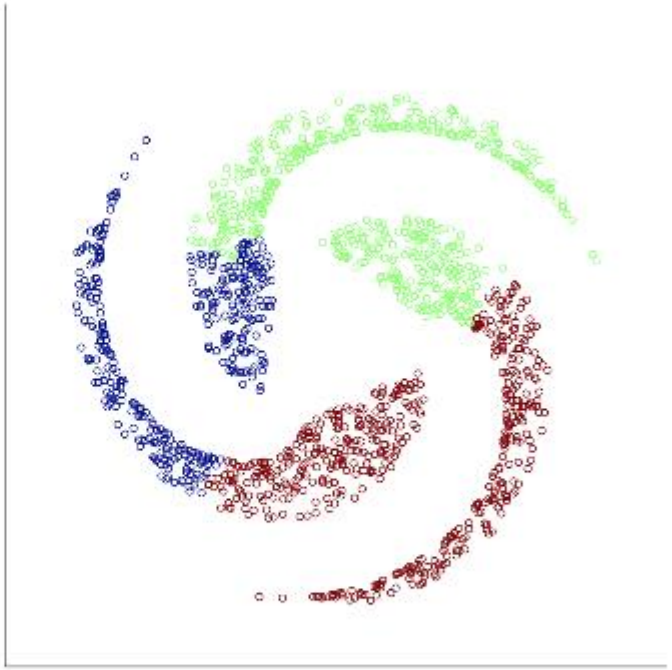


(d) centroid distance

Example: HAC on Pinwheel example (1 of 2)

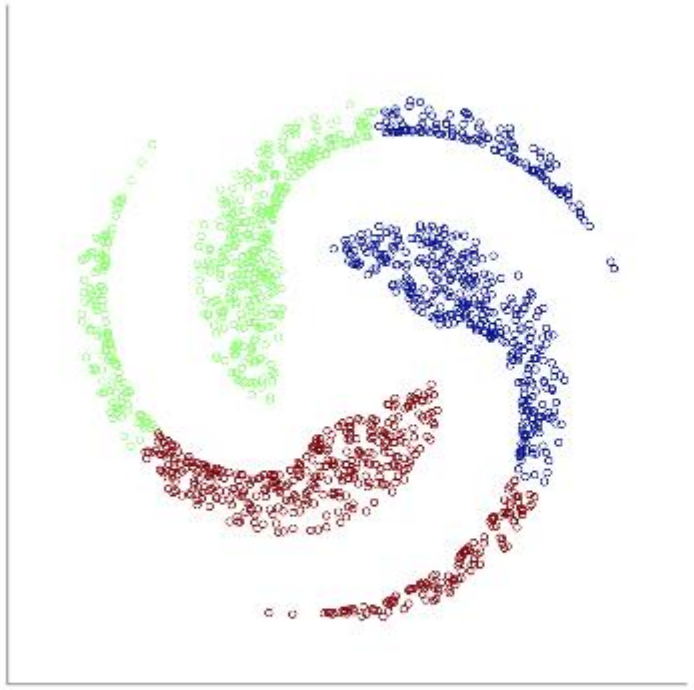


(a) min distance

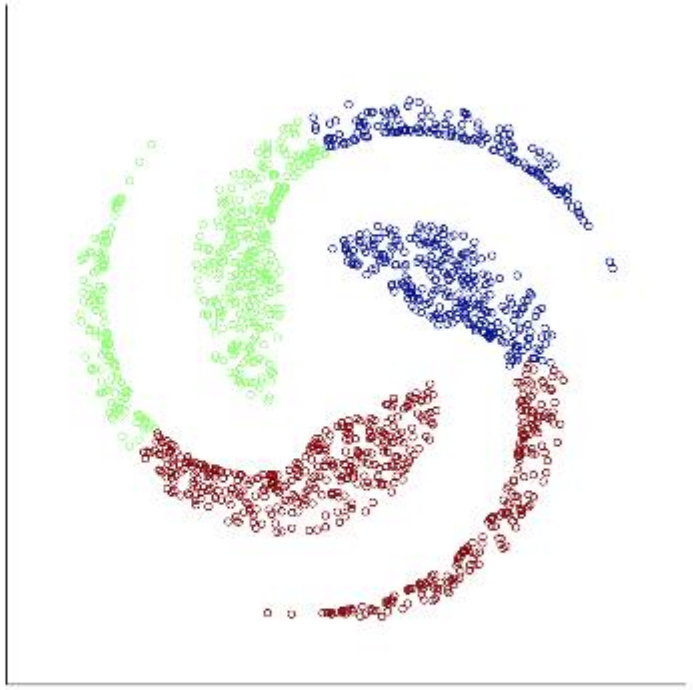


(b) max distance

Example: HAC on Pinwheel Example (2 of 2)



(c) average distance



(d) centroid distance

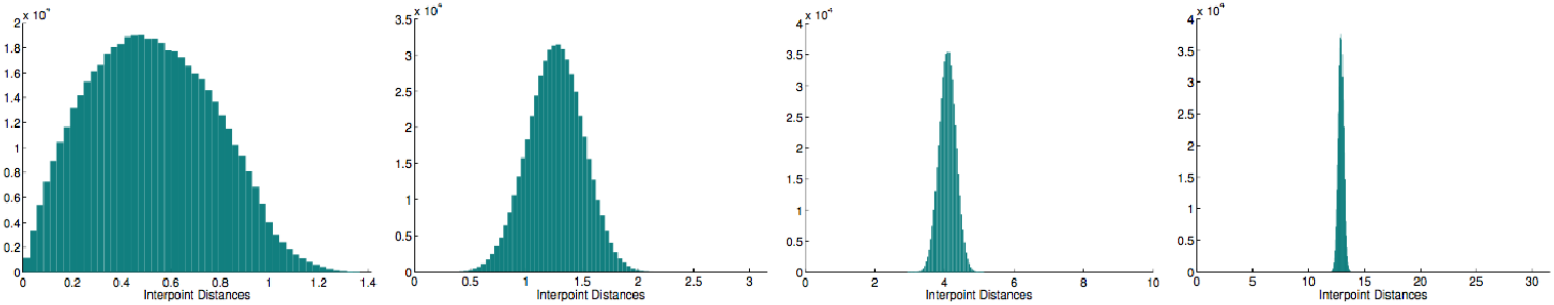
HAC: Discussion

- Average and centroid distances provide a compromise between min and max
- Non-parametric. Can get arbitrary cluster shapes
- Can over-fit in high dimensional spaces that have irrelevant

Computational complexity is $n(n-1)m$ to get pairwise distance between all examples, and then n^2T for T rounds since need to do pairwise check. $T \ll m$ and thus $\boldsymbol{O}(n^2m)$. (c.f., $\boldsymbol{O}(nKmT)$ for K-means)

Curse of Dimensionality

- Histograms of inter-sample distance for 1000 samples in unit Hypercube



(a) 2 Dimensions (b) 10 Dimensions (c) 100 Dimensions (d) 1000 Dimensions

As dimensions increase, we get concentration of the distances relative to min (0) and max($\sqrt{m}$) distances (distance is sum of IID r.v.s)

Because of this, HAC suffers the "curse of dimensionality"

Becomes less useful as the dimensionality of the data grows

Contents

Clustering

- A very natural , unsupervised learning problem
- K-means and HAC are two simple, popular algorithms
- HAC is more flexible, but has poor performance in high dimensional problems

Thank You