Unsupervised Learning: Clustering

Prof. Mingkui Tan

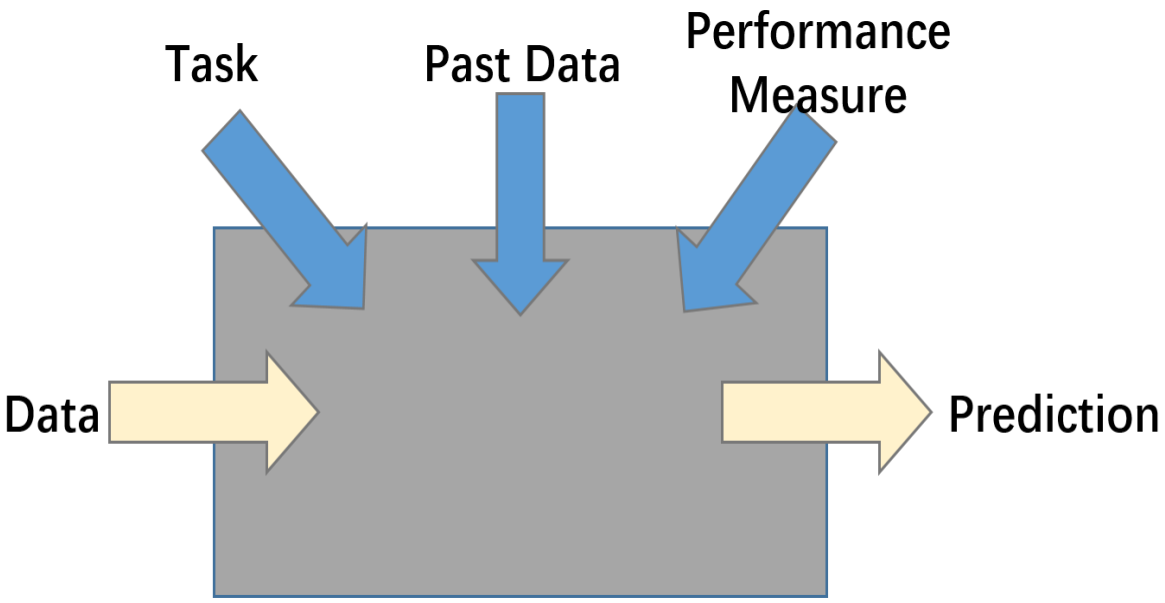
South China University of Technology
Southern Artificial Intelligence Laboratory(SAIL)

December 1, 2018

Contents

Supervised Learning

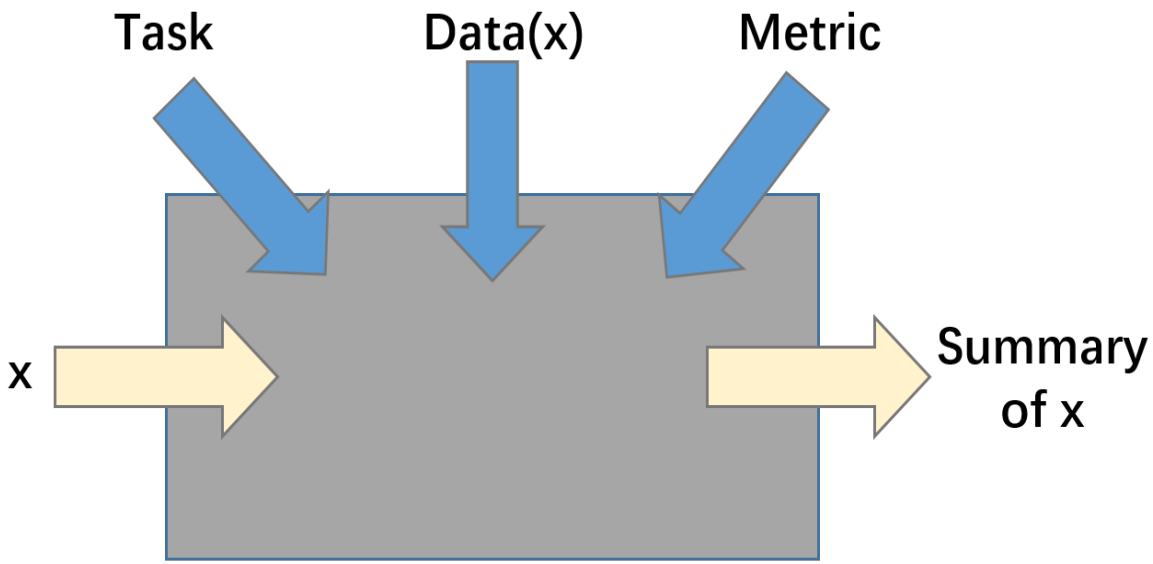
Data $D = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$



Solved via linear regression, logistic regression, Naive Bayes, neural nets, SVMs, etc.

Unsupervised Learning

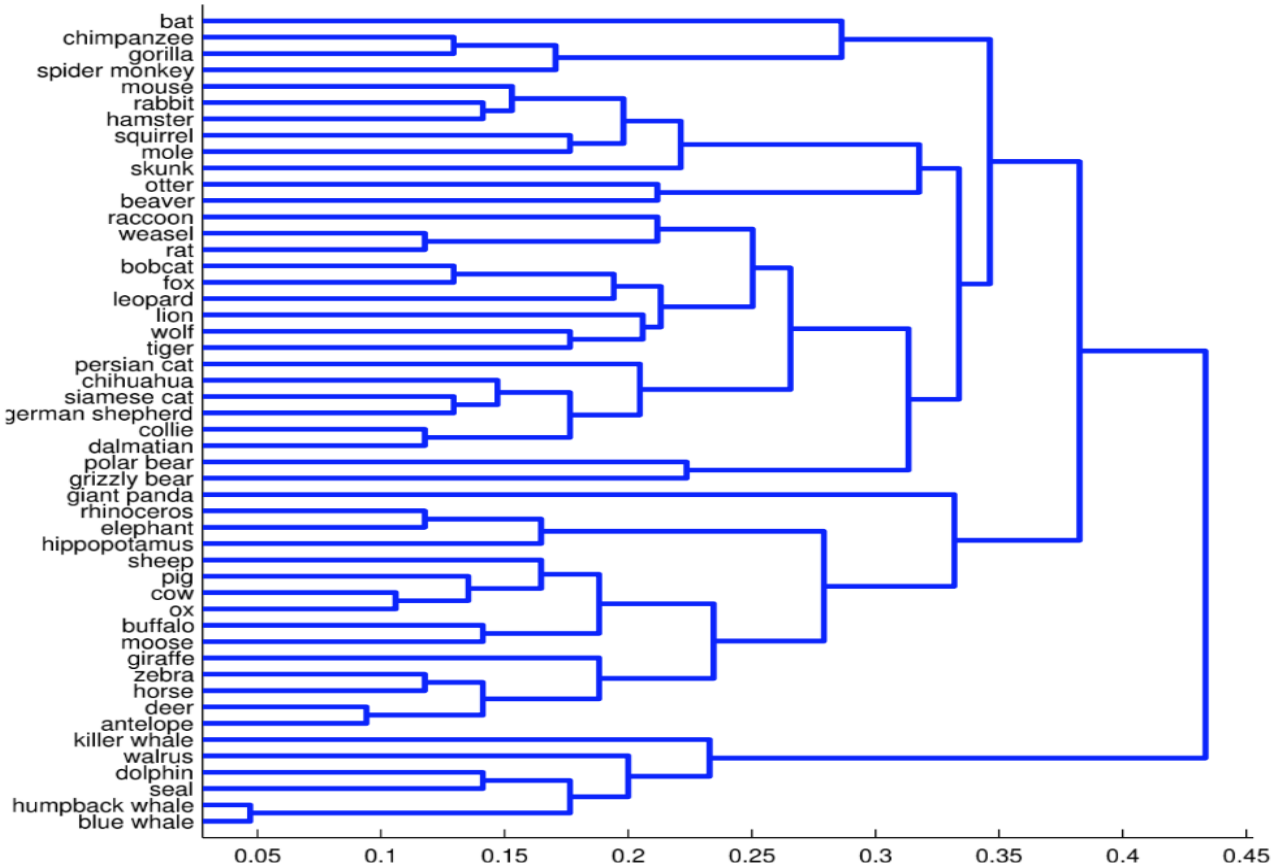
A different kind of task



- Data $D = \{\mathbf{x}\}_{i=1}^{n}$. No target values.
- Typical goals: understand data, summarize data, identify concepts.

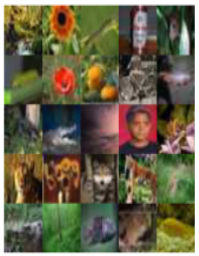
Application: Clustering Animals by Features

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

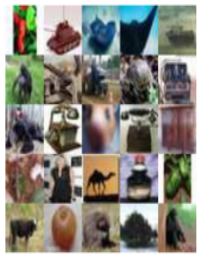


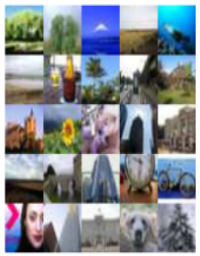
Application: Clustering Image Data

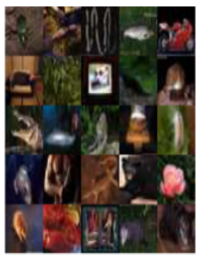

(a) Cluster Centers

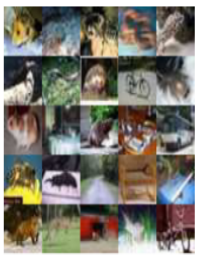

(b) Cluster 1

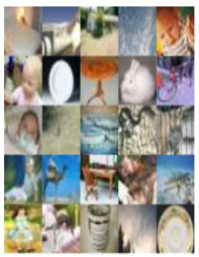

(c) Cluster 2

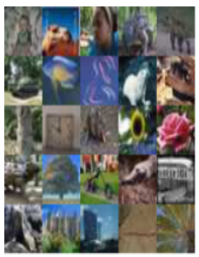

(d) Cluster 3


(e) Cluster 4

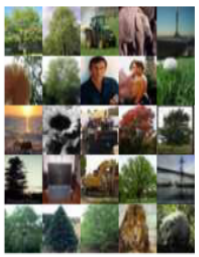

(f) Cluster 5


(g) Cluster 6

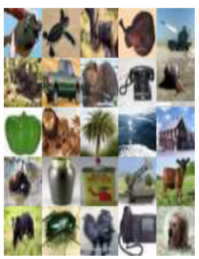

(h) Cluster 7

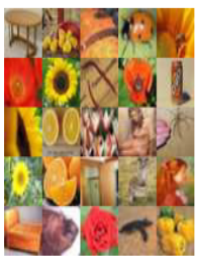

(i) Cluster 8


(j) Cluster 9

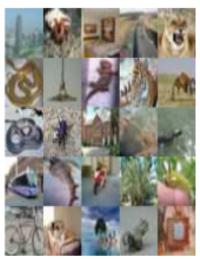

(k) Cluster 10


(l) Cluster 11


(m) Cluster 12

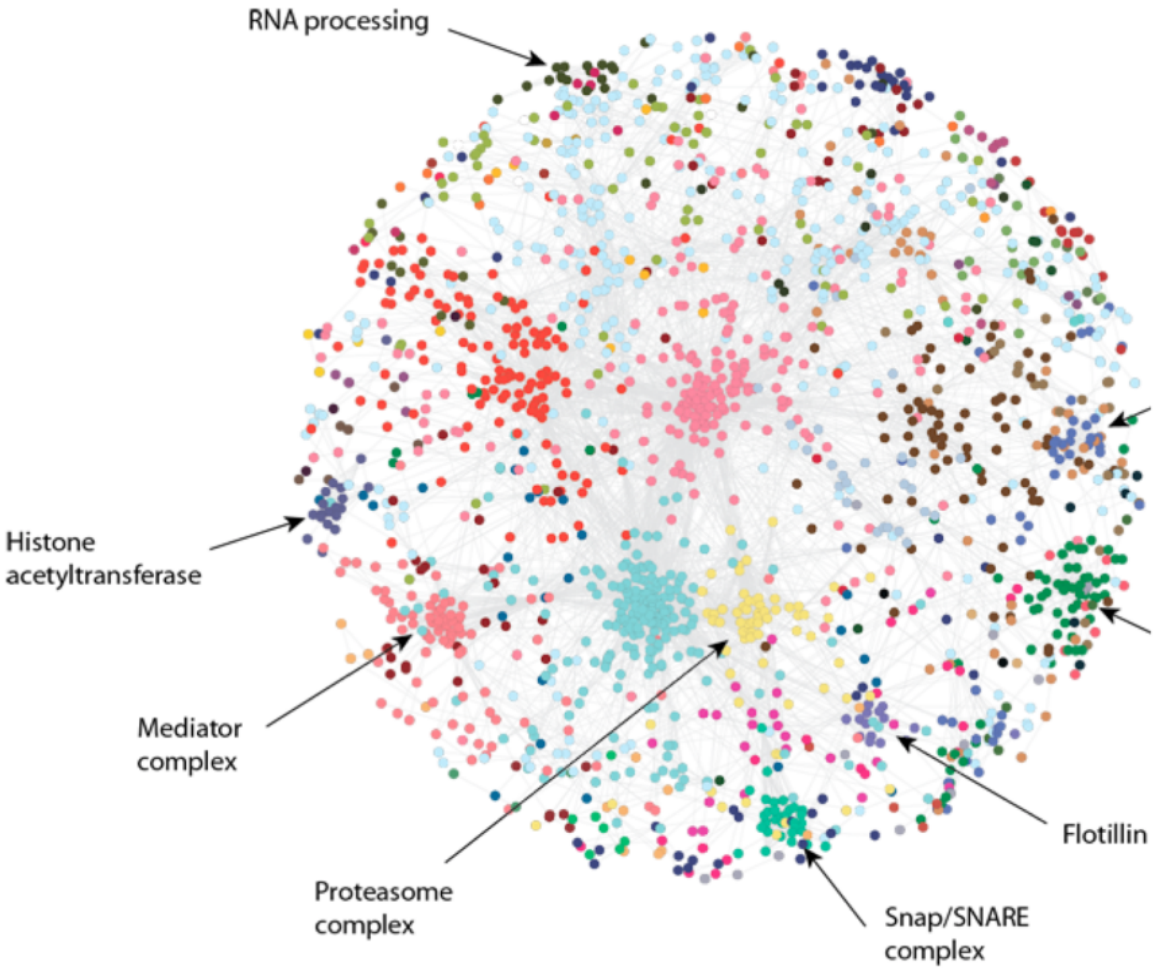

(n) Cluster 13


(o) Cluster 14


(p) Cluster 15


(q) Cluster 16

Clustering: Understanding Gene Regulation



Clustering

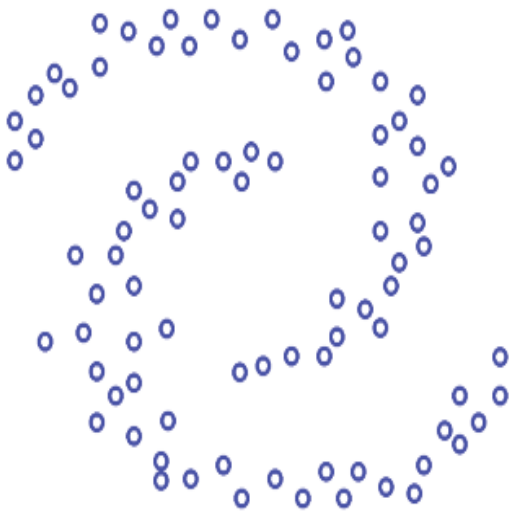
- Simplest idea for discovering structure.
- Find groups of similar examples:
  - To understand the data.
  - For dimensionality reduction.
  - To preprocess a lot of unlabeled data, find concepts to use for supervised learning.

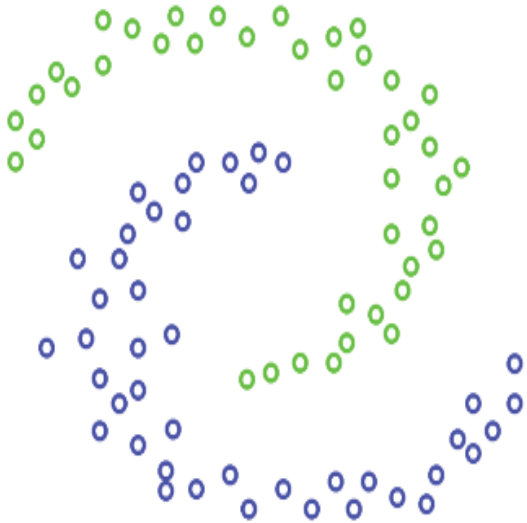
Today's lecture:

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

Example 1: How Would You Cluster these Points?

Example 2: How Would You Cluster these Points?





The Clustering Problem

- Each example $\mathbf{x} \in \mathbb{R}^m$. Data $D = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$.
- Number of clusters K (may not be given).
- Typical output: assignment of each example to a cluster.
- $\mathbf{r}_i$ is the binary responsibility vector for example $\mathbf{x}_i$. A one-hot encoding ($r_{ik} = 1$ for assigned cluster, $r_{ik} = 0$ otherwise).

What Is A Good Clustering?

- Examples are "more similar" to the examples in their cluster than to examples in other clusters.
- How to measure the similarity?
  - How to measure the similarity between one example $\mathbf{x}$ to another $\hat{\mathbf{x}}$?
  - How to measure the similarity between one group of examples to another?

For data in $\mathbb{R}^m$, a typical approach is l_2 metric:

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

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

The K-Means Objective

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

- Associate each cluster with a prototype $\boldsymbol{\mu}_k \in \mathbb{R}^m$, for $k \in \{1, \ldots, K\}$.
- Make an assignment $\mathbf{r}_i$ of each example $\mathbf{x}_i$ to a cluster.
- Objective: find prototypes and an assignment to minimize

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

where $||\mathbf{z}|| = \sqrt{\mathbf{z}^\top \mathbf{z}}$.

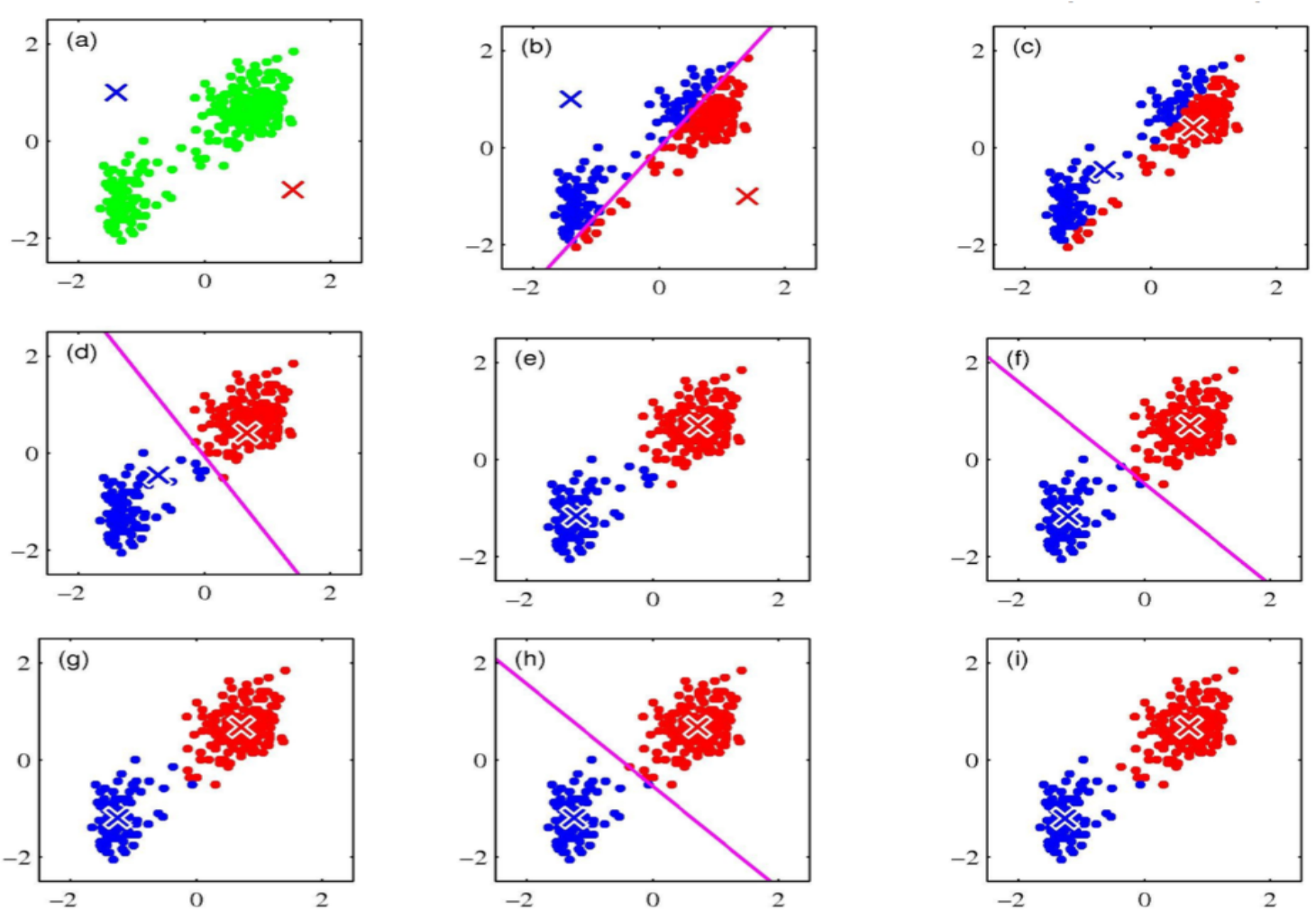
This is highly non-convex, with lots of local minima.

K-Means Clustering Algorithm (Lloyd's Algorithm)

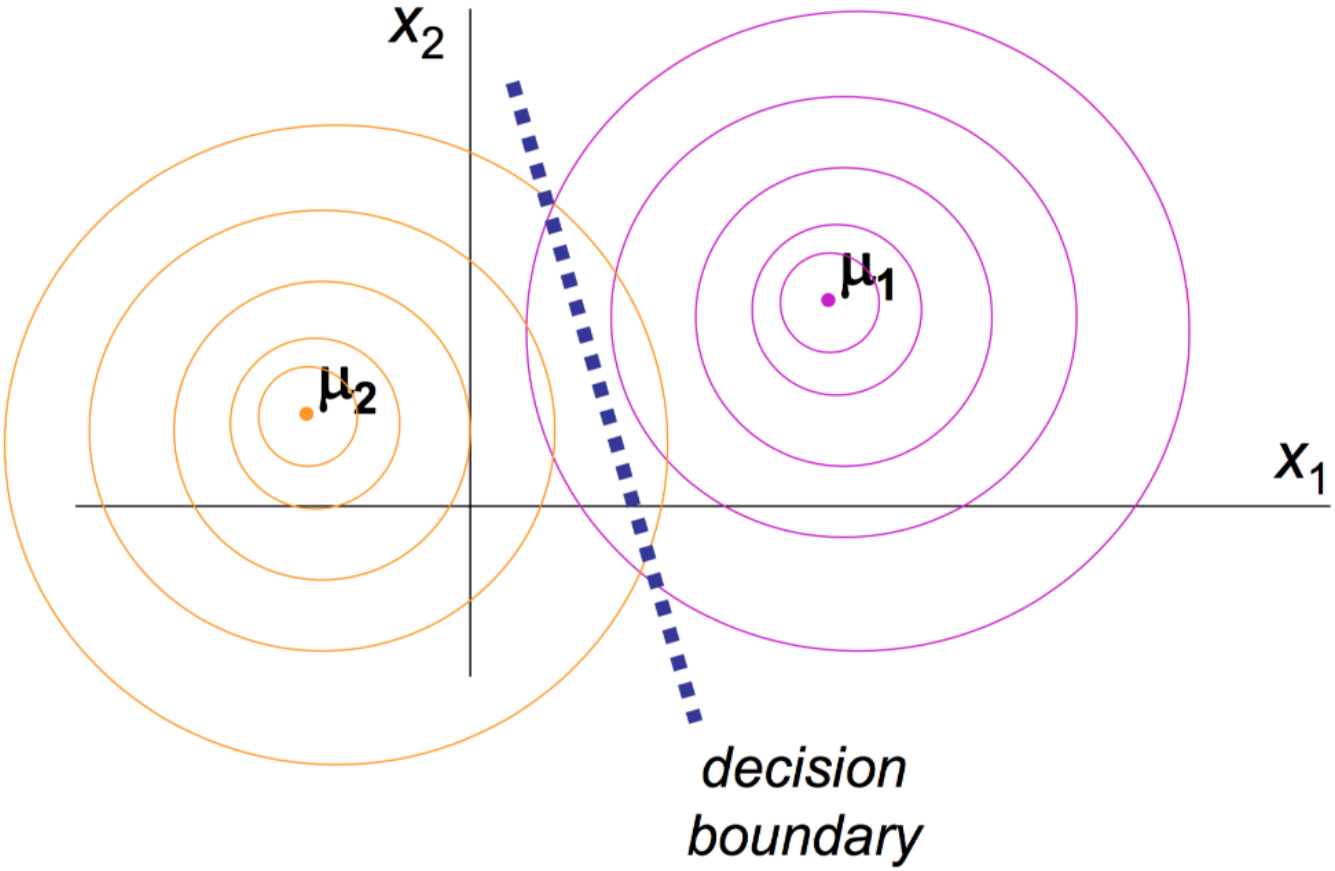
1. Initialize prototypes $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_k$ at random.
2. Repeat until converged:
   - Step 1: Assign each example to the closest prototype (breaking ties in favor of current assignment)
   $$\mathbf{r}_i := \arg \min_{k \in (1,\ldots,K)} ||\mathbf{x}_i - \boldsymbol{\mu}_k||$$
   - Step 2: For each k, set $\boldsymbol{\mu}_k$ to the centroid of assigned examples
   $$\boldsymbol{\mu}_k := \frac{1}{n_k} \sum_{i=1}^{n} r_{ik} \mathbf{x}_i,$$
   where $n_k = \sum_i r_{ik}$.

Typical to run this multiple times, with different initial conditions.

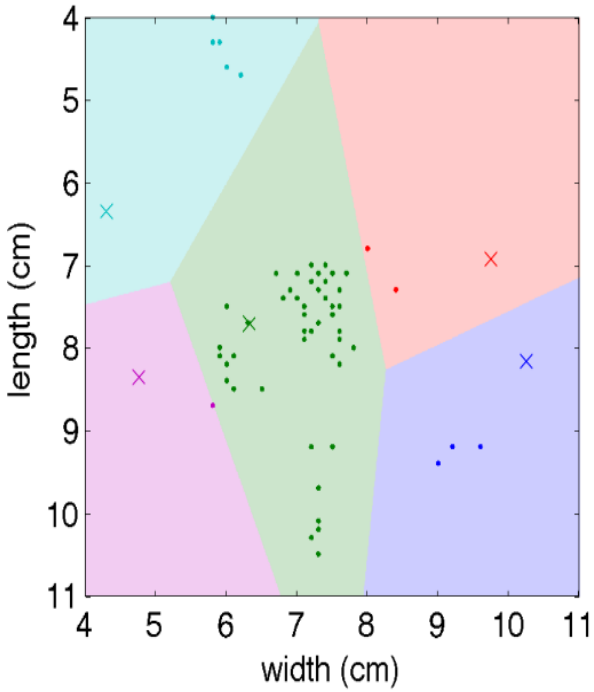
Example: K-Means on Old Faithful Eruptions



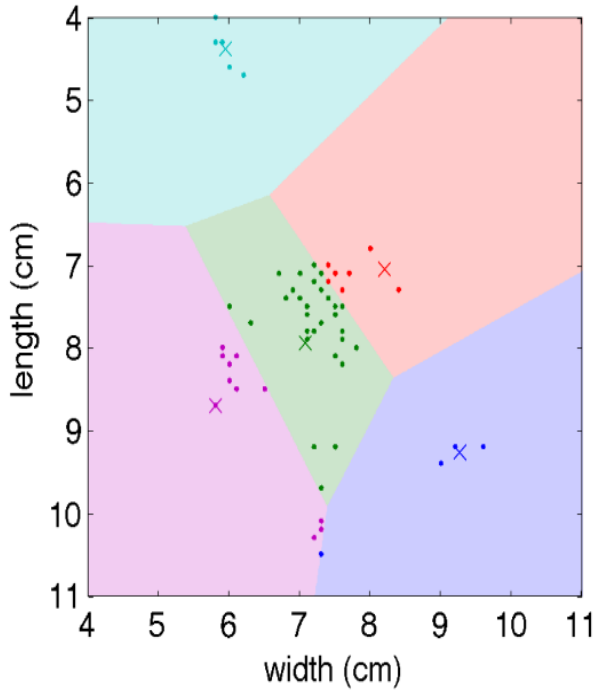
Note: Linear Decision Boundaries



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



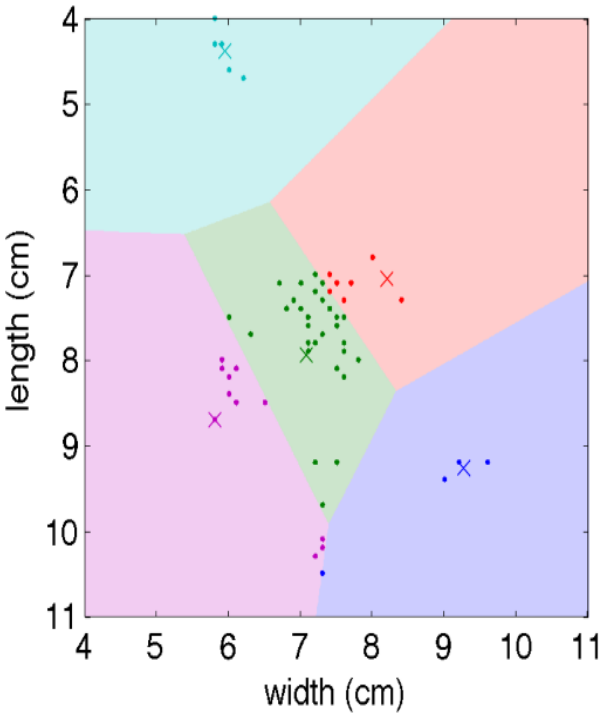
(a) Initialization



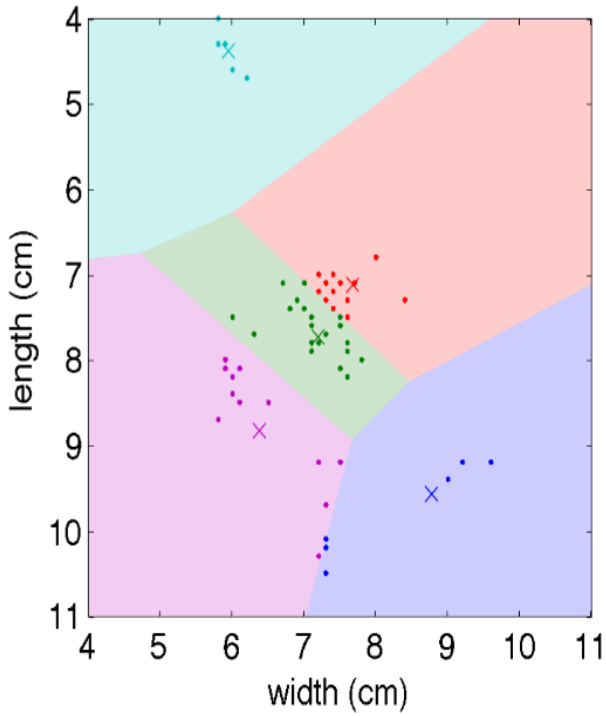
(b) Iteration 1

($K = 5$, Iain Murray data)

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



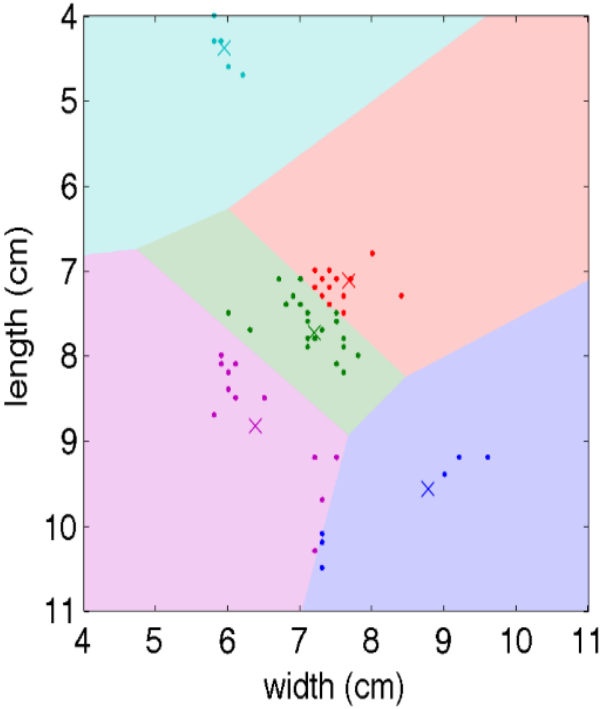
(b) Iteration 1



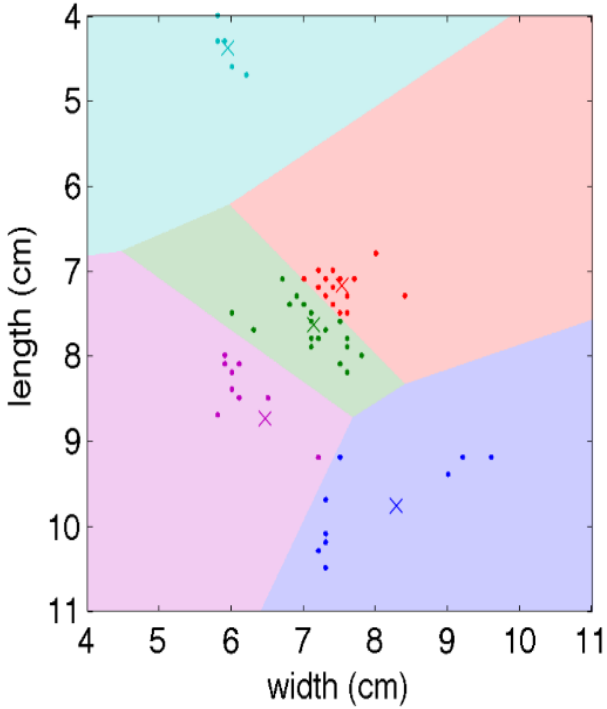
(c) Iteration 2

($K = 5$, Iain Murray data)

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



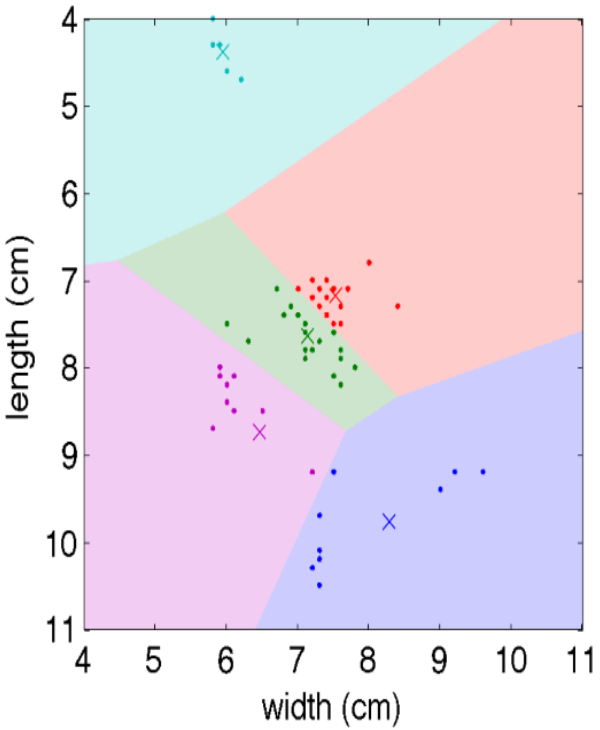
(c) Iteration 2



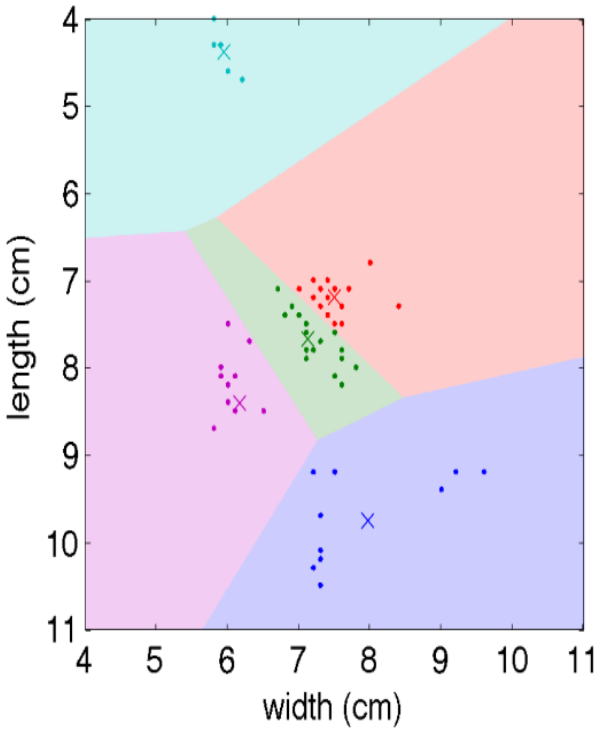
(d) Iteration 3

($K = 5$, Iain Murray data)

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



(d) Iteration 3

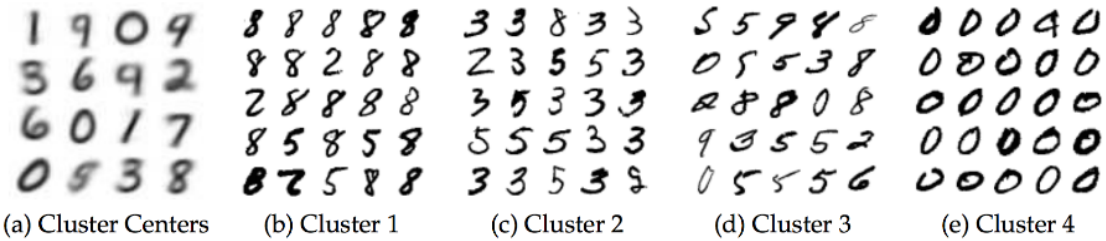


(e) Iteration 4

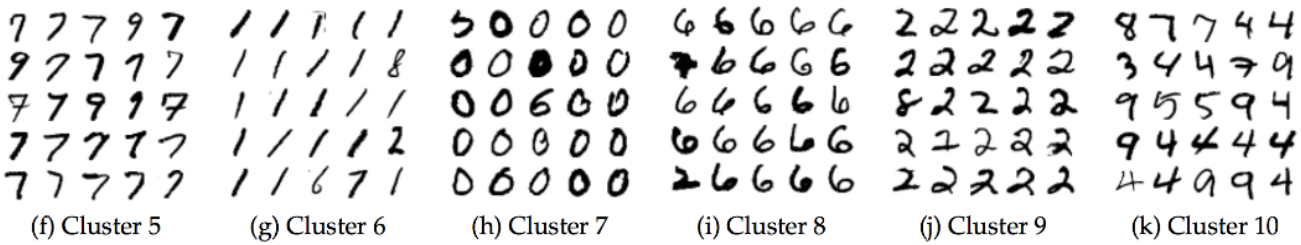
($K = 5$, Iain Murray data)

Example: K-means Clustering on Handwritten Digits

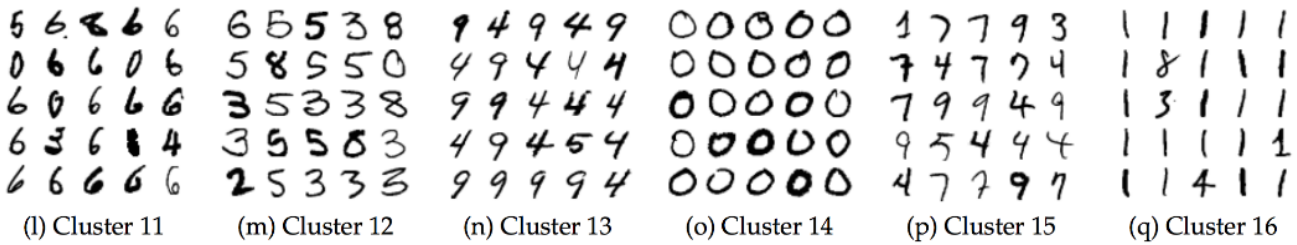
MNIST: 60,000 digits. 28 x 28 grayscale (0-255)



(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

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

Example: K-means Clustering on Image Data

CIFAR-100 color. 50,000 images. 32 x 32 x 3 (RGB), each 0-255.

(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

($K = 16$. Clusters pick up on low-freq color variations)

Example: K-Means Clustering on Documents

30,991 articles from Grolier's Encyclopedia. Articles are represented via a count vector of most common words (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:

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

Solve this via coordinate descent. Alternate $\{\mathbf{r}_i\}$ and $\{\boldsymbol{\mu}_k\}$ updates

- Step 1: $\{\mathbf{r}\}$ update. Fixing $\{\boldsymbol{\mu}\}$, minimize loss by assigning each example to the cluster that is closest.
- Step 2: $\{\boldsymbol{\mu}\}$ update. Fixing $\{\mathbf{r}\}$, work with squared distance, $\boldsymbol{\mathcal{L}}(\boldsymbol{\mu}_k) = \sum_{i=1}^{n} r_{ik}(\mathbf{x}_i - \boldsymbol{\mu}_k)^\top(\mathbf{x}_i - \boldsymbol{\mu}_k)$. We have:

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

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

K-Means Clustering

- Simple and popular method.
- Parametric (The effective number of parameters is mK, to define the cluster centroids).
- Not useful if linear decision boundaries fails.

K-Means Clustering

Computational complexity:

- Assignment step is O(nKm), since for each example need to compare to each center.
- Centroid update is O(nm), since need to take average of different partitions of the data.
- O($nKmT$) time over T iterations (generally $T \ll n$).

Setting K, The Number of Clusters:

- 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.

Can be useful to add class labels:
After clustering, it is common to want to associate a cluster with a name; e.g., by testing some examples and associating the cluster with the majority concept.

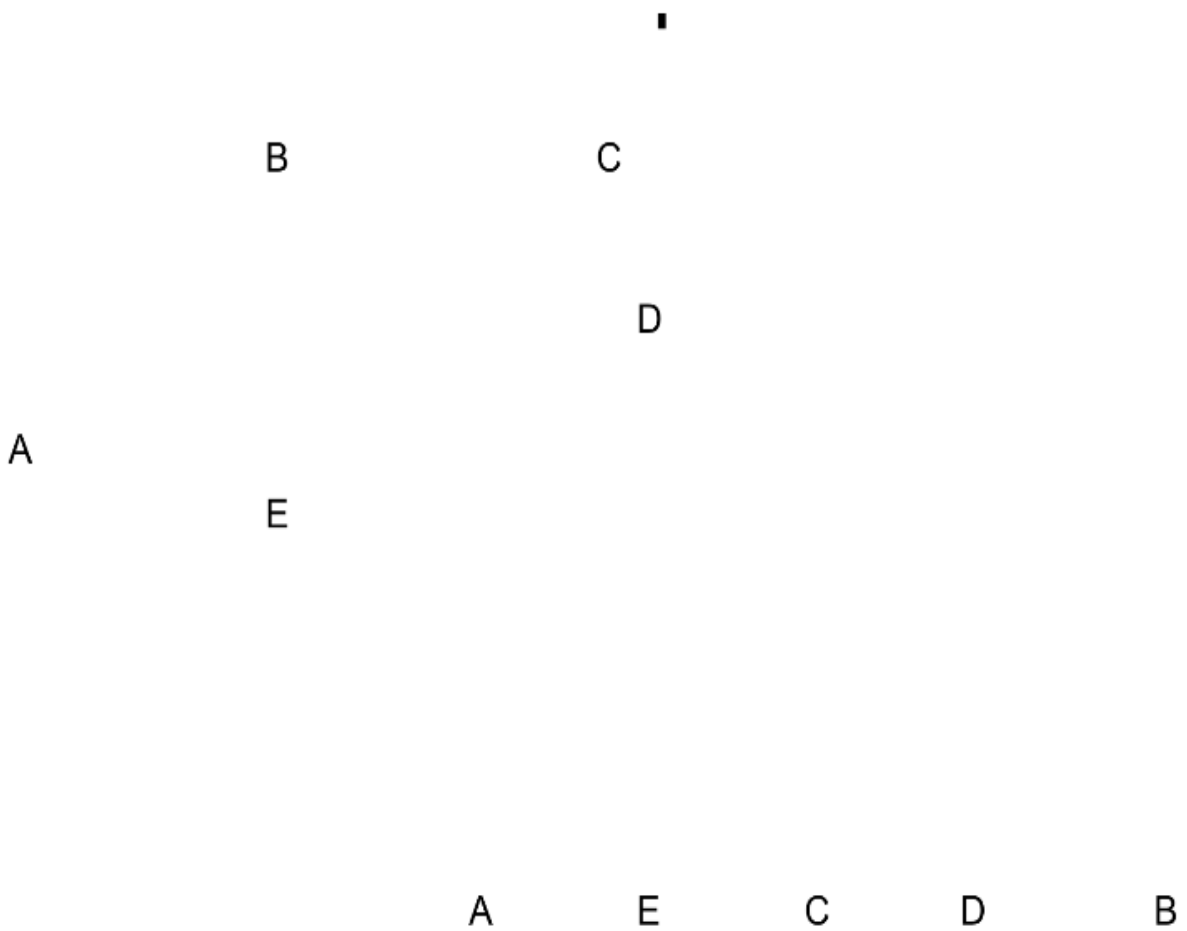
Hierarchical Agglomerative Clustering (HAC)

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

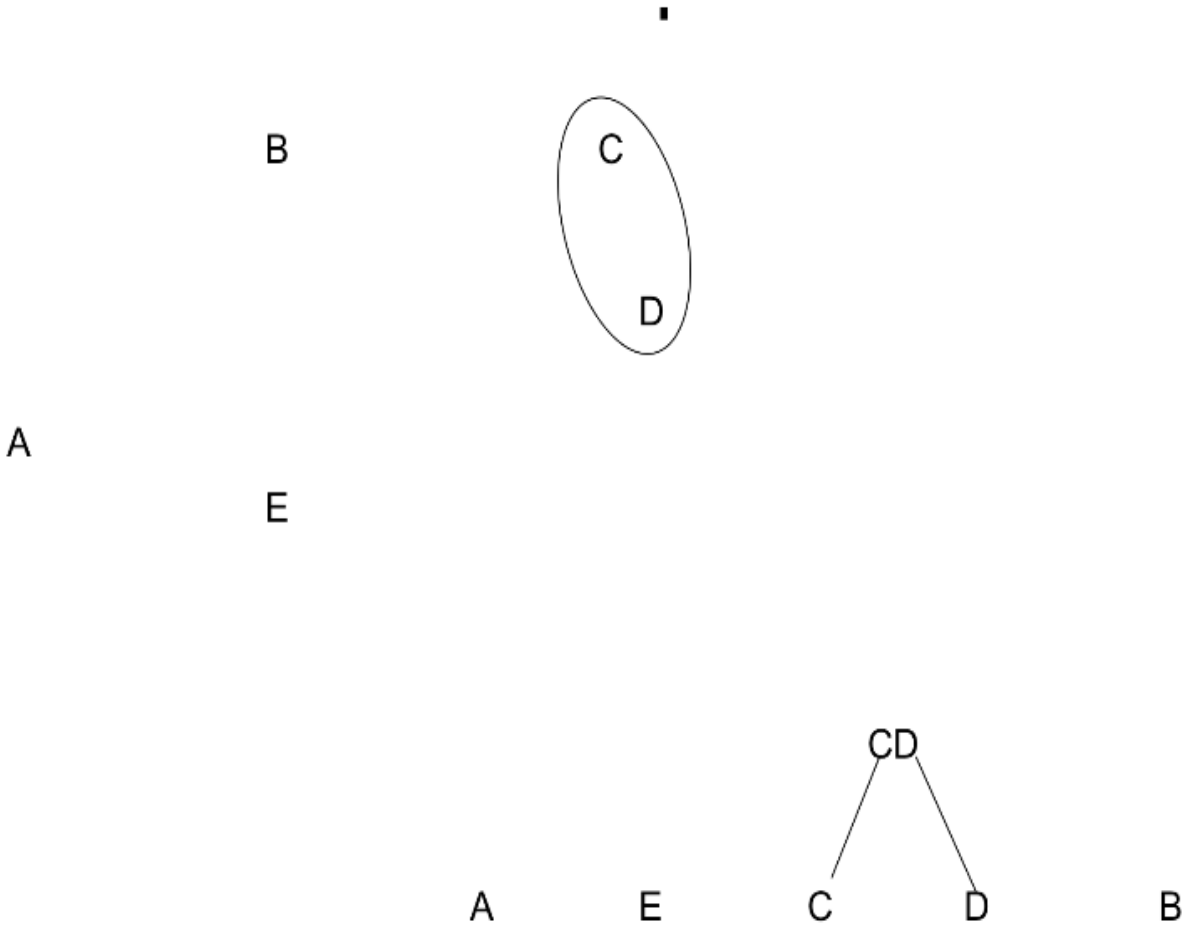
Compared with K-Means:

- 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.)

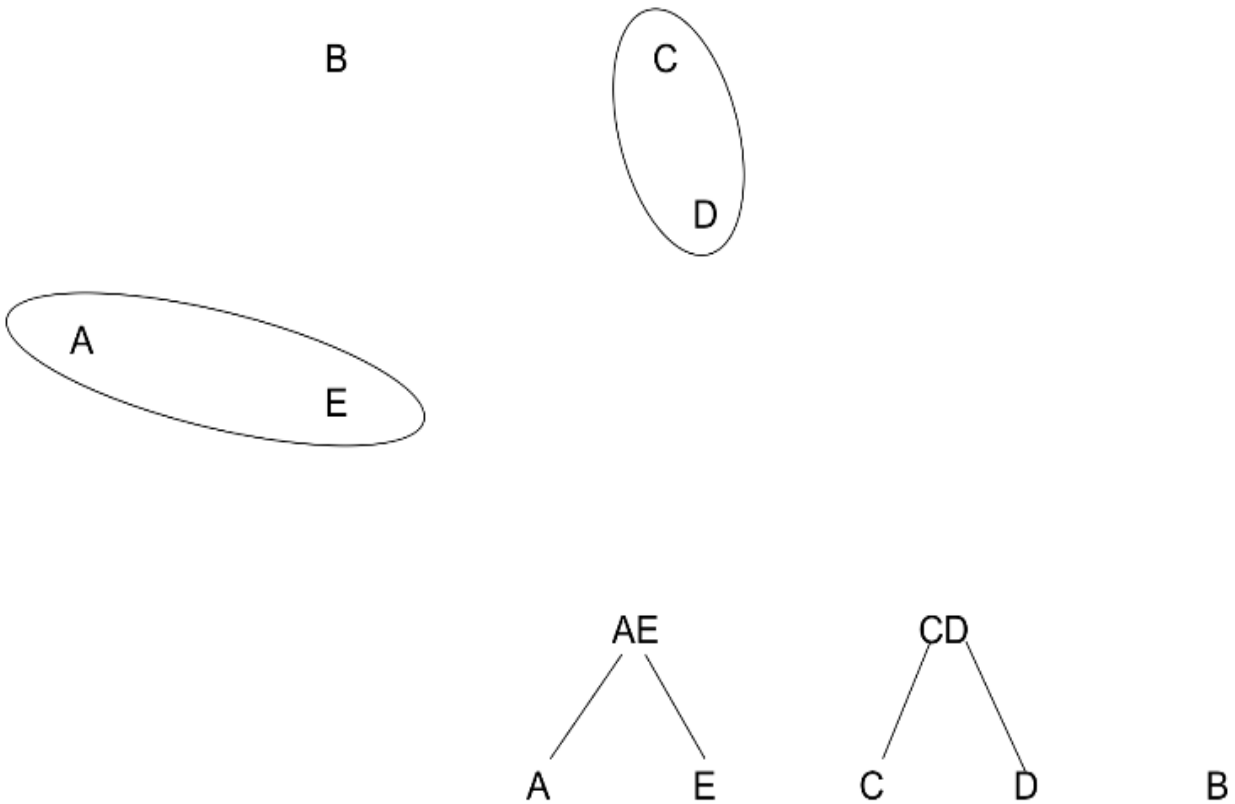
HAC Example (1 of 5)



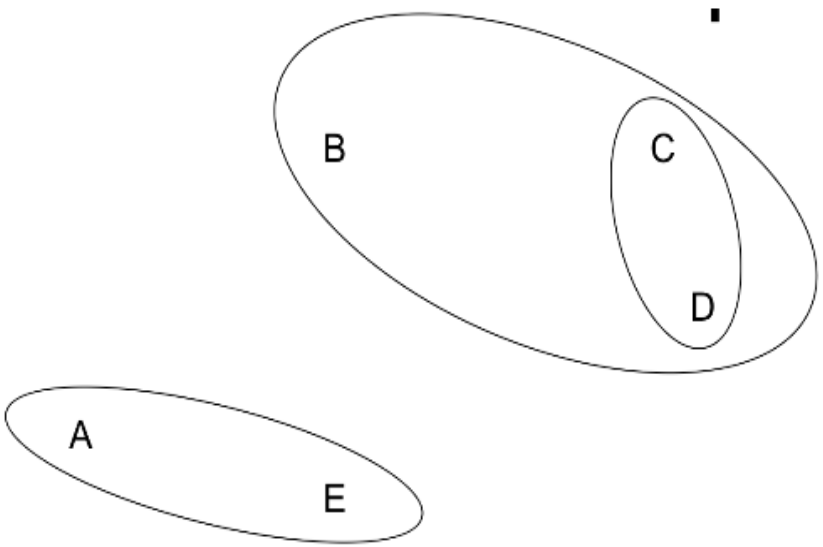
HAC Example (2 of 5)

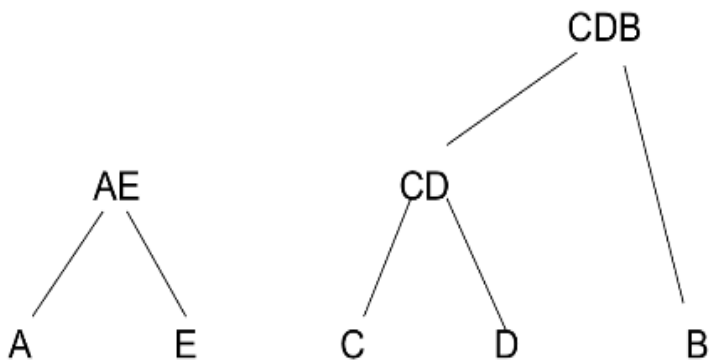


HAC Example (3 of 5)

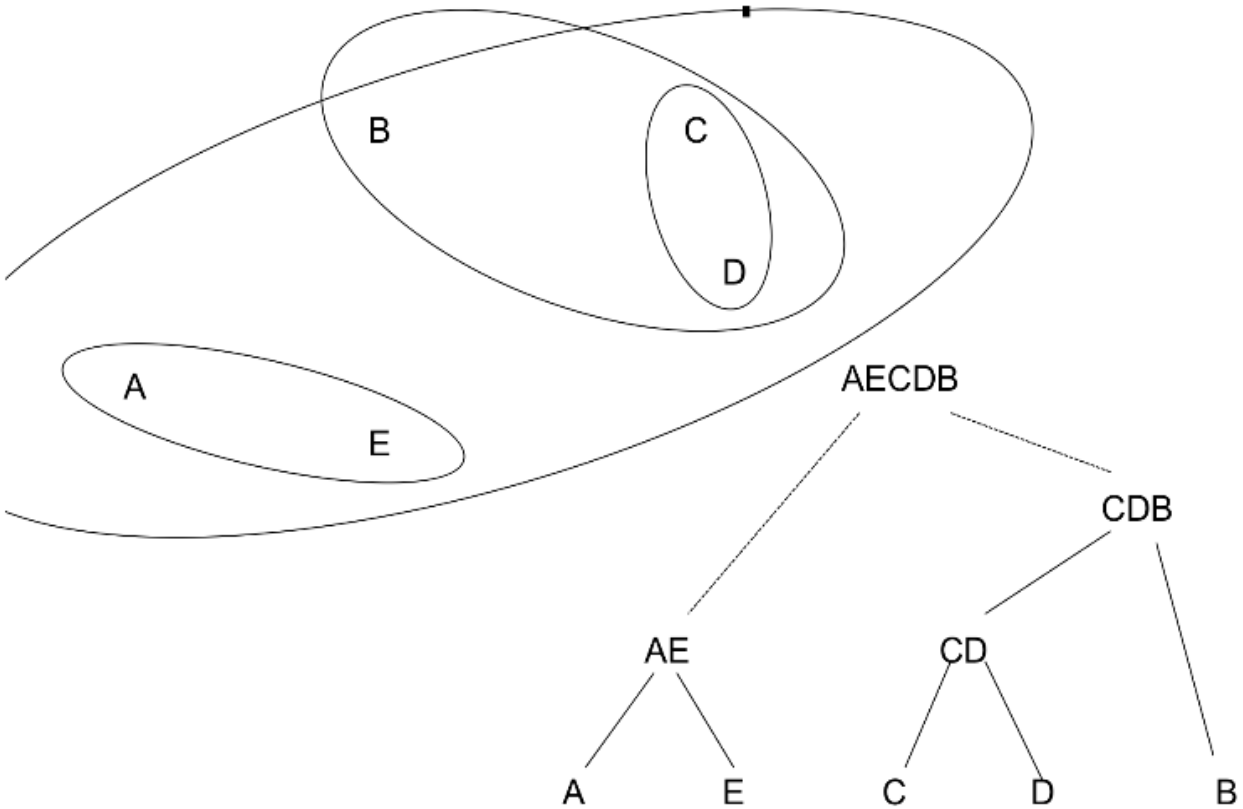


HAC Example (4 of 5)





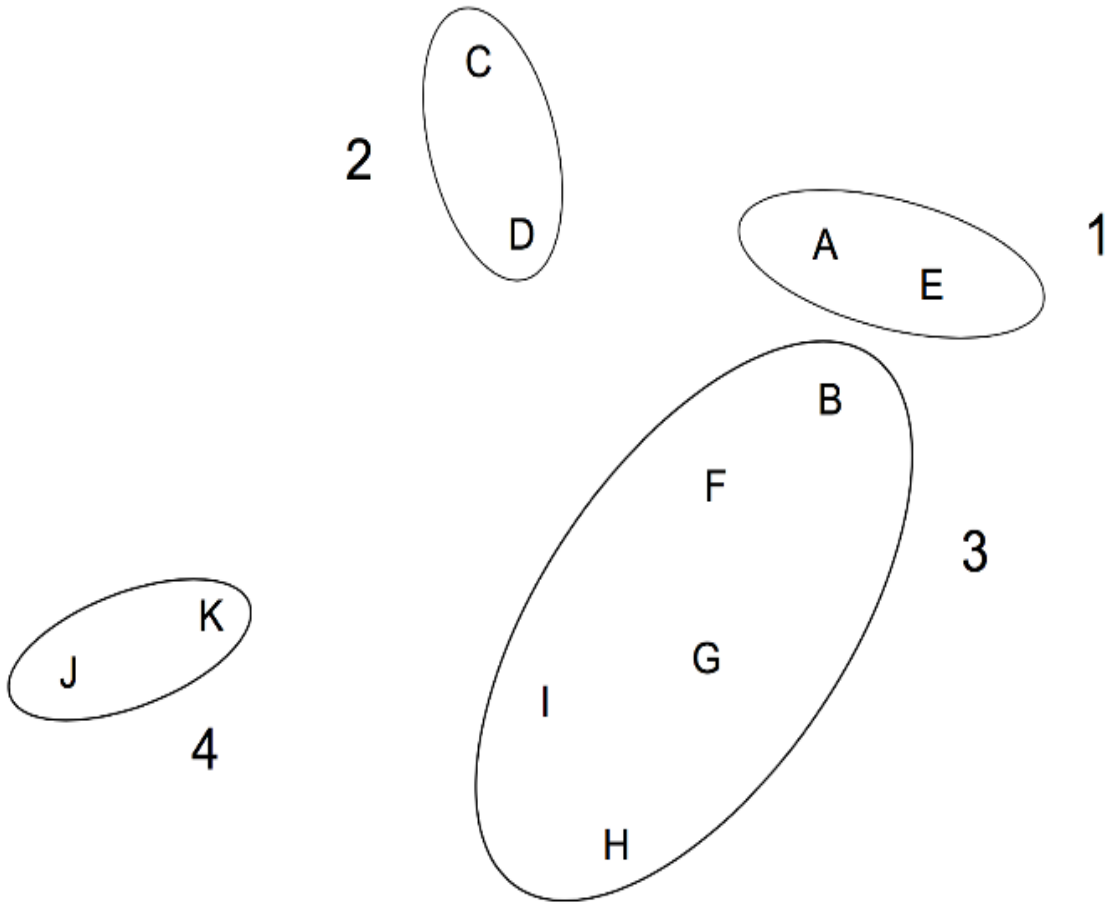
HAC Example (5 of 5)



HAC Algorithm

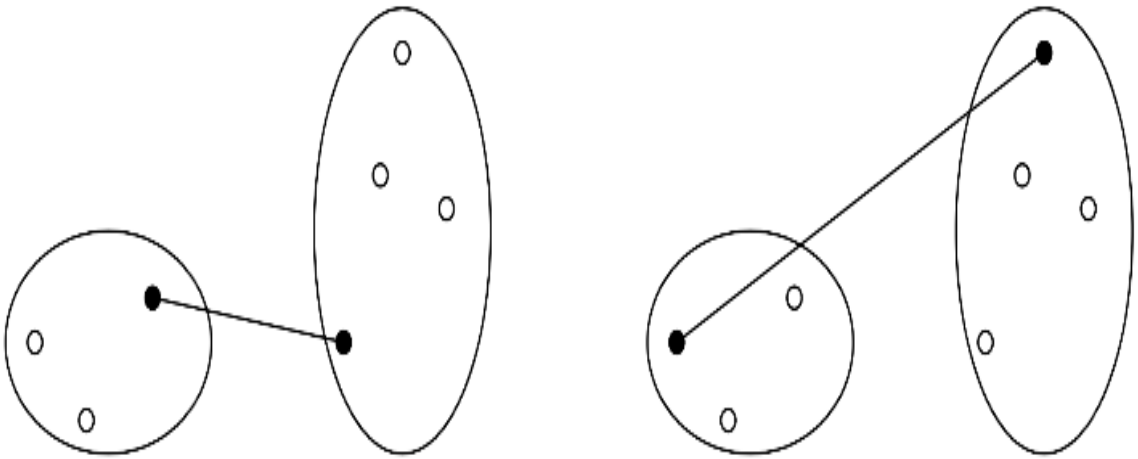
1. Start with each example in its own cluster.
2. Repeat until a single cluster:
    - Find the two "closest" clusters, and merge them.

What Is Closest to Cluster {AE}?



Max? Min? Average? Centroid?

HAC: Min and Max Group Distances

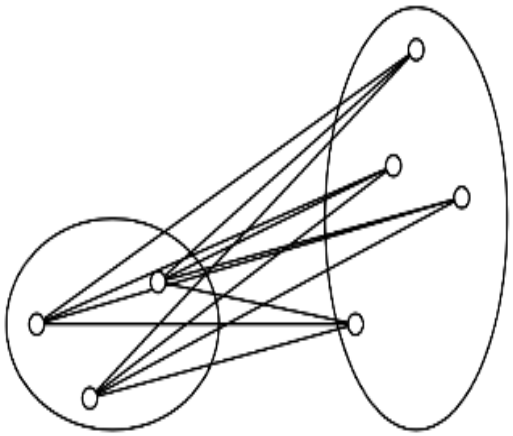


(a) min distance (b) max distance

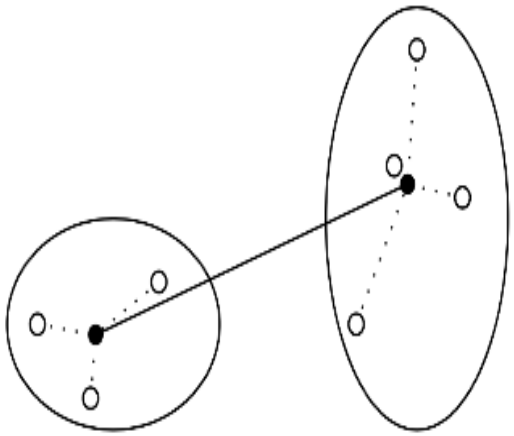
$$d_{min}(\mathbf{G}, \hat{\mathbf{G}}) = \min_{\mathbf{x}\in\mathbf{G}, \hat{\mathbf{x}}\in\hat{\mathbf{G}}} ||\mathbf{x} - \hat{\mathbf{x}}||$$

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

HAC: Average and Centroid Group Distances



(c) average distance

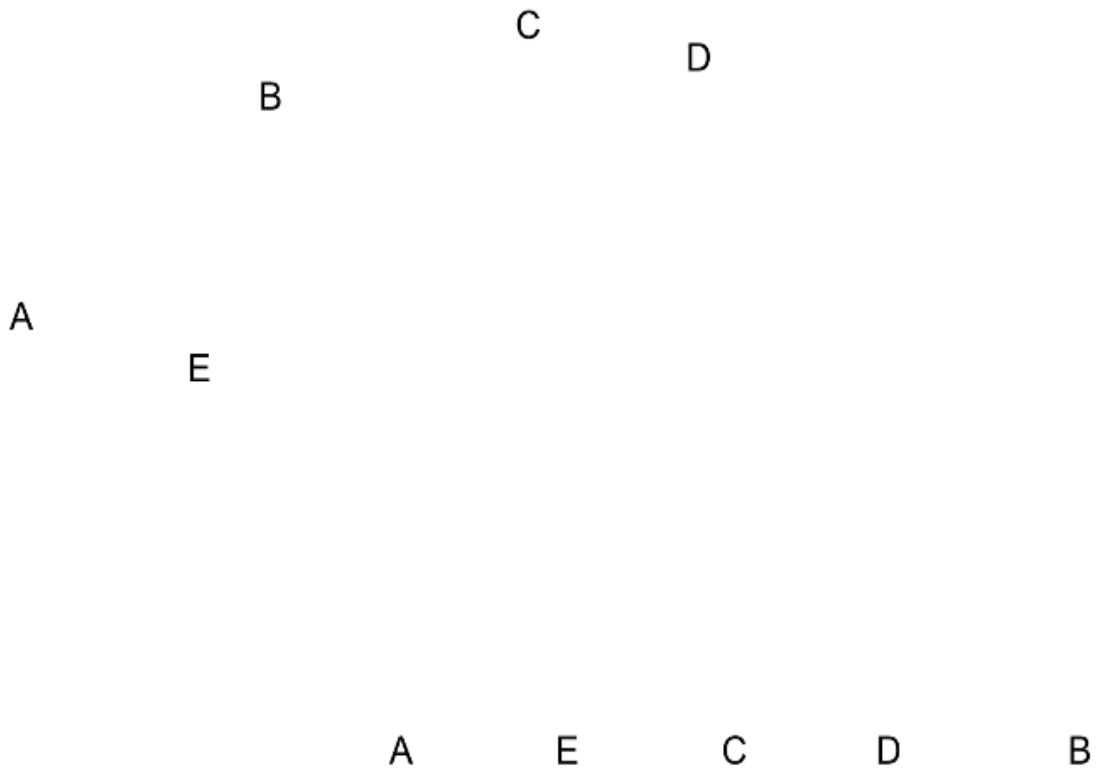


(d) centroid distance

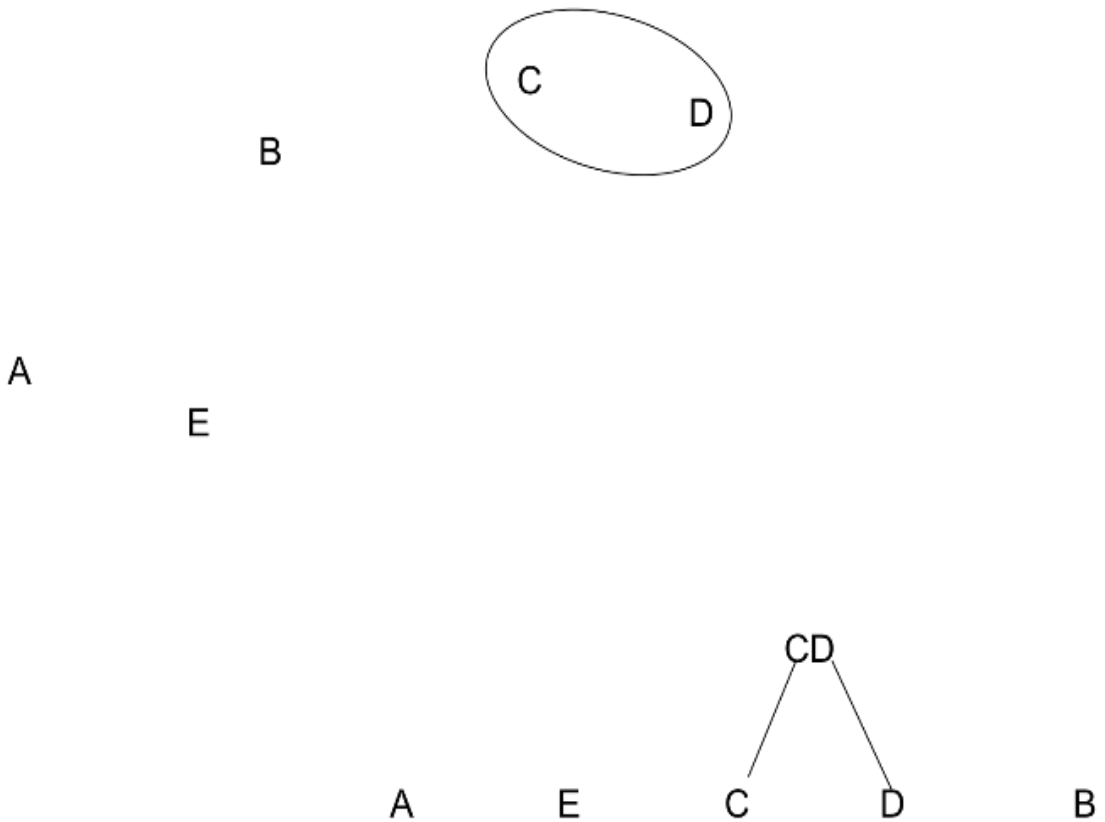
$$d_{avg}(\mathbf{G}, \hat{\mathbf{G}}) = \frac{1}{|\mathbf{G}||\hat{\mathbf{G}}|} \sum_{\mathbf{x} \in \mathbf{G}, \hat{\mathbf{x}} \in \hat{\mathbf{G}}} ||\mathbf{x} - \hat{\mathbf{x}}||$$

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

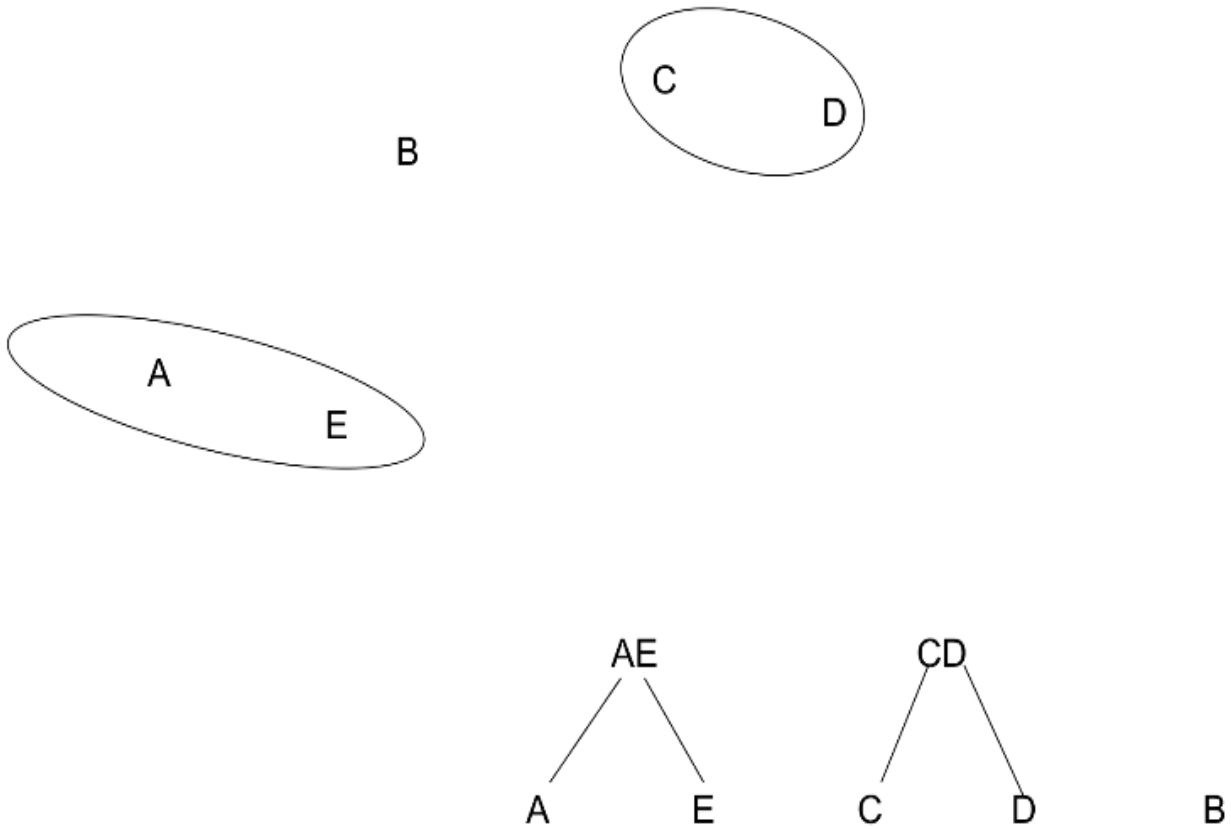
Example: HAC with Min Distance (1 of 6)



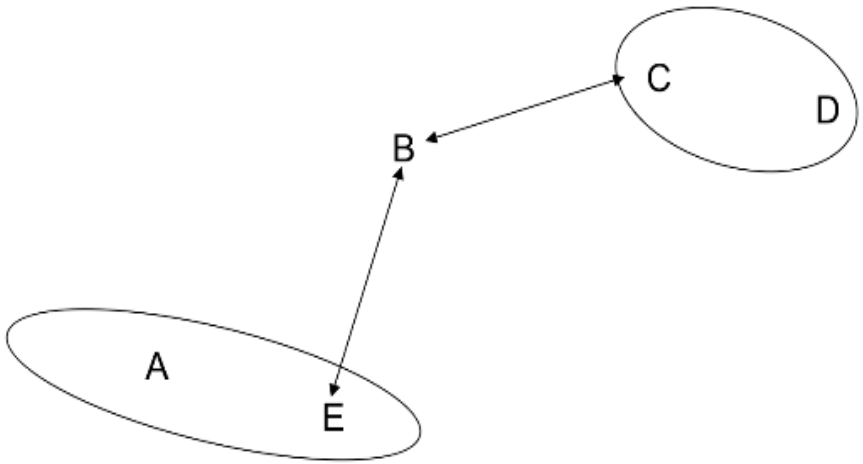
Example: HAC with Min Distance (2 of 6)

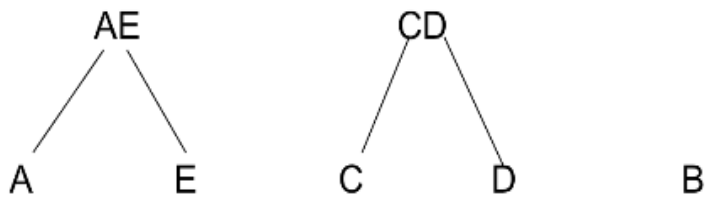


Example: HAC with Min Distance (3 of 6)

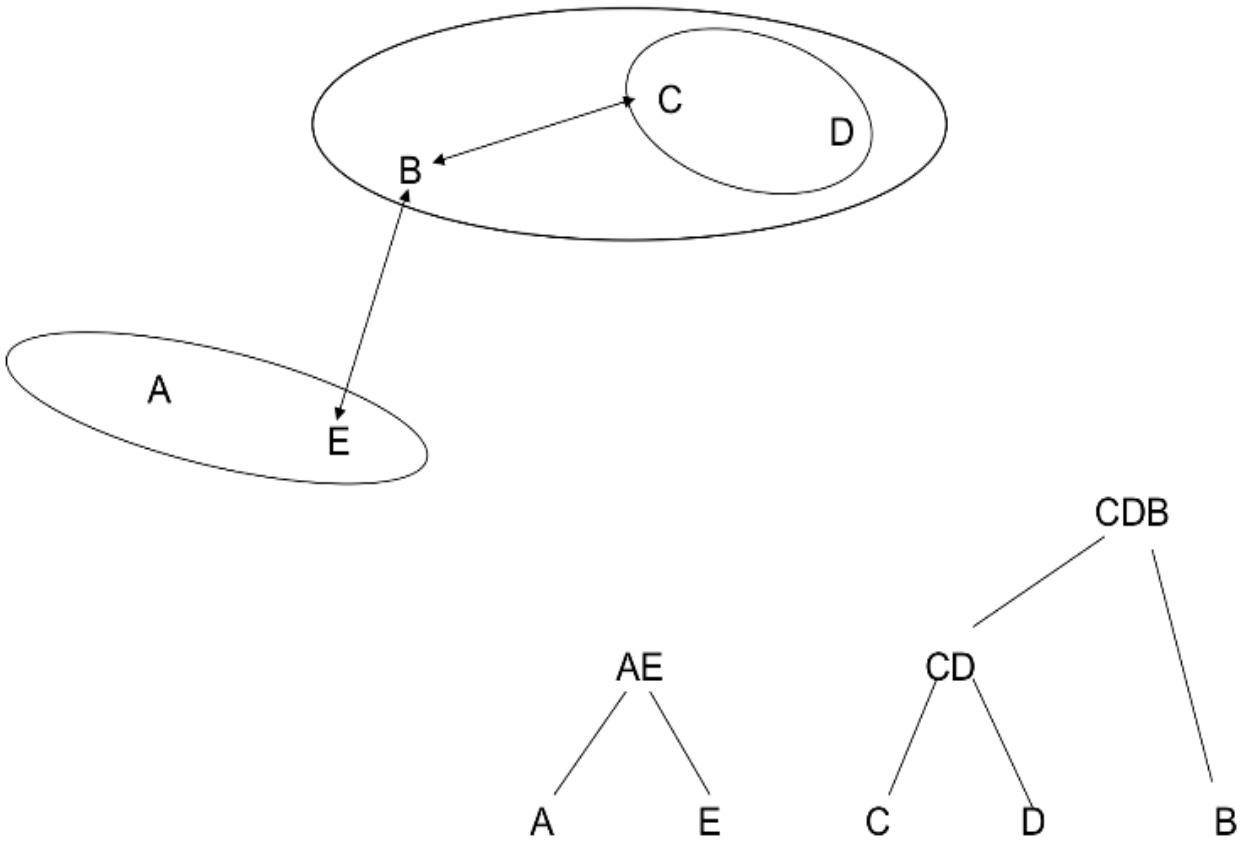


Example: HAC with Min Distance (4 of 6)

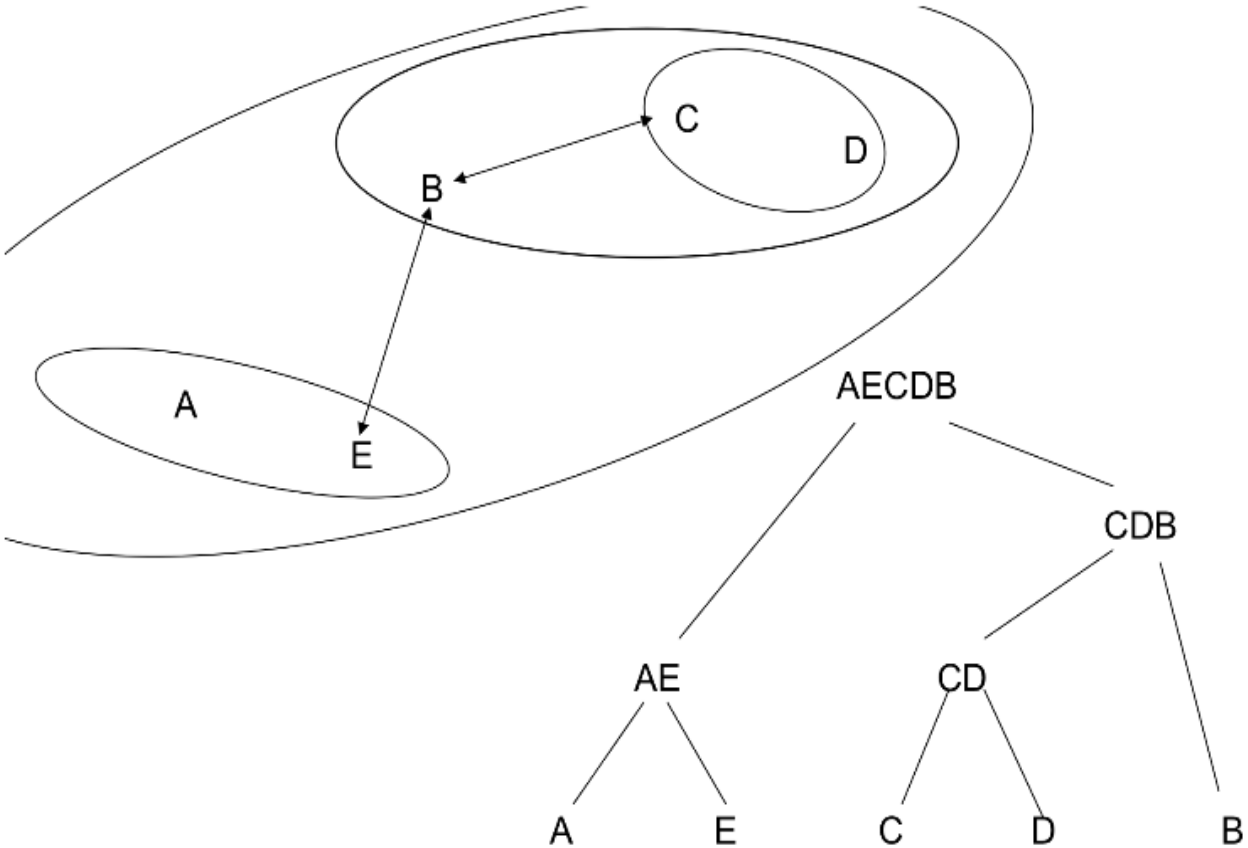




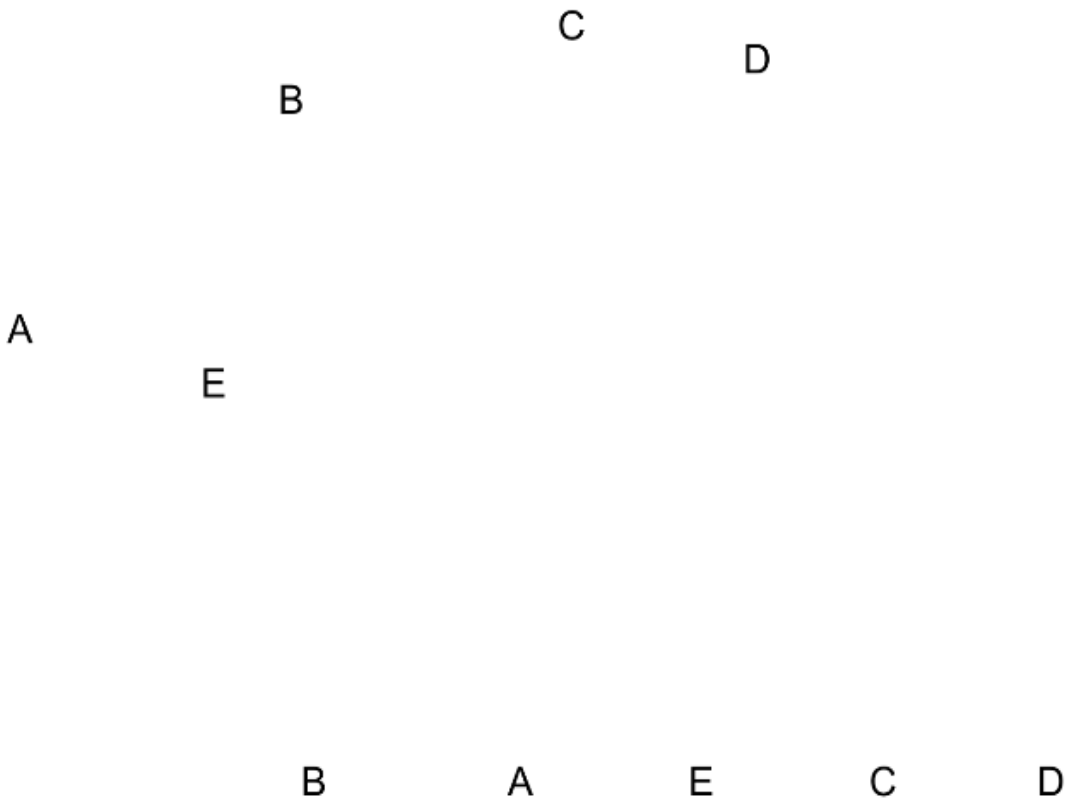
Example: HAC with Min Distance (5 of 6)



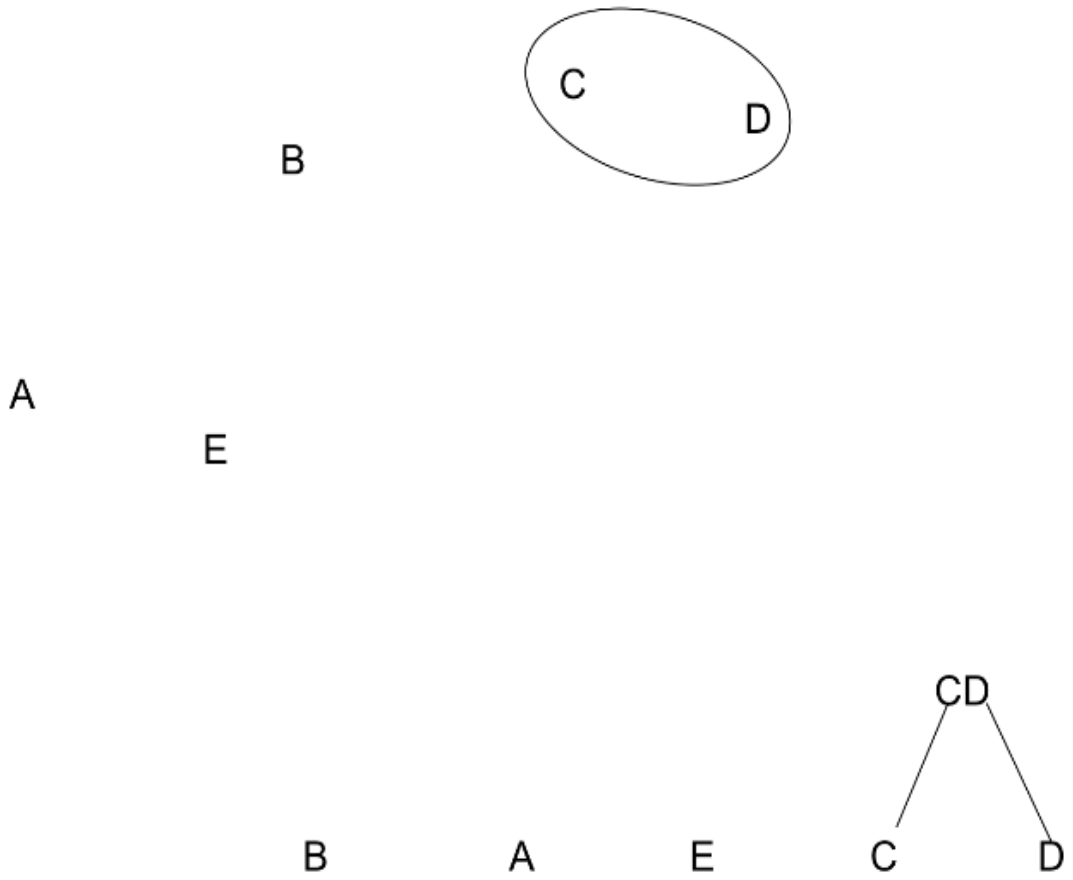
Example: HAC with Min Distance (6 of 6)



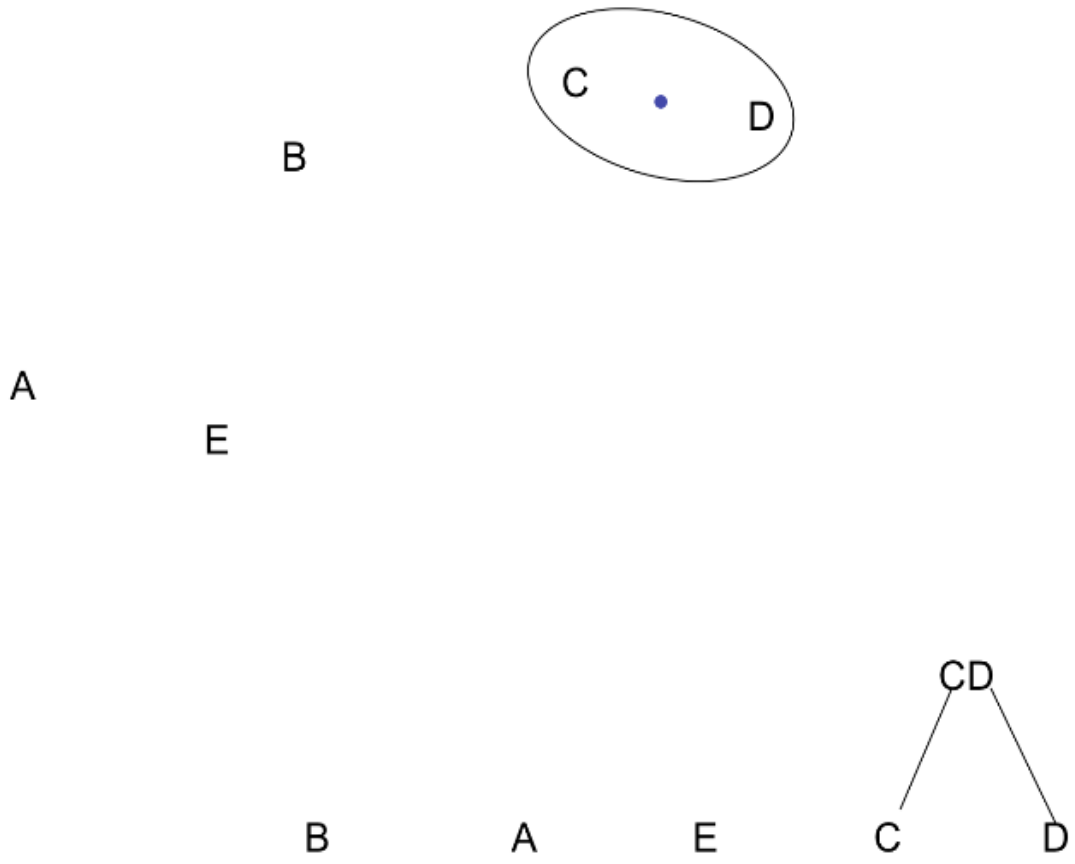
Example: HAC with Centroid Distance (1 of 7)



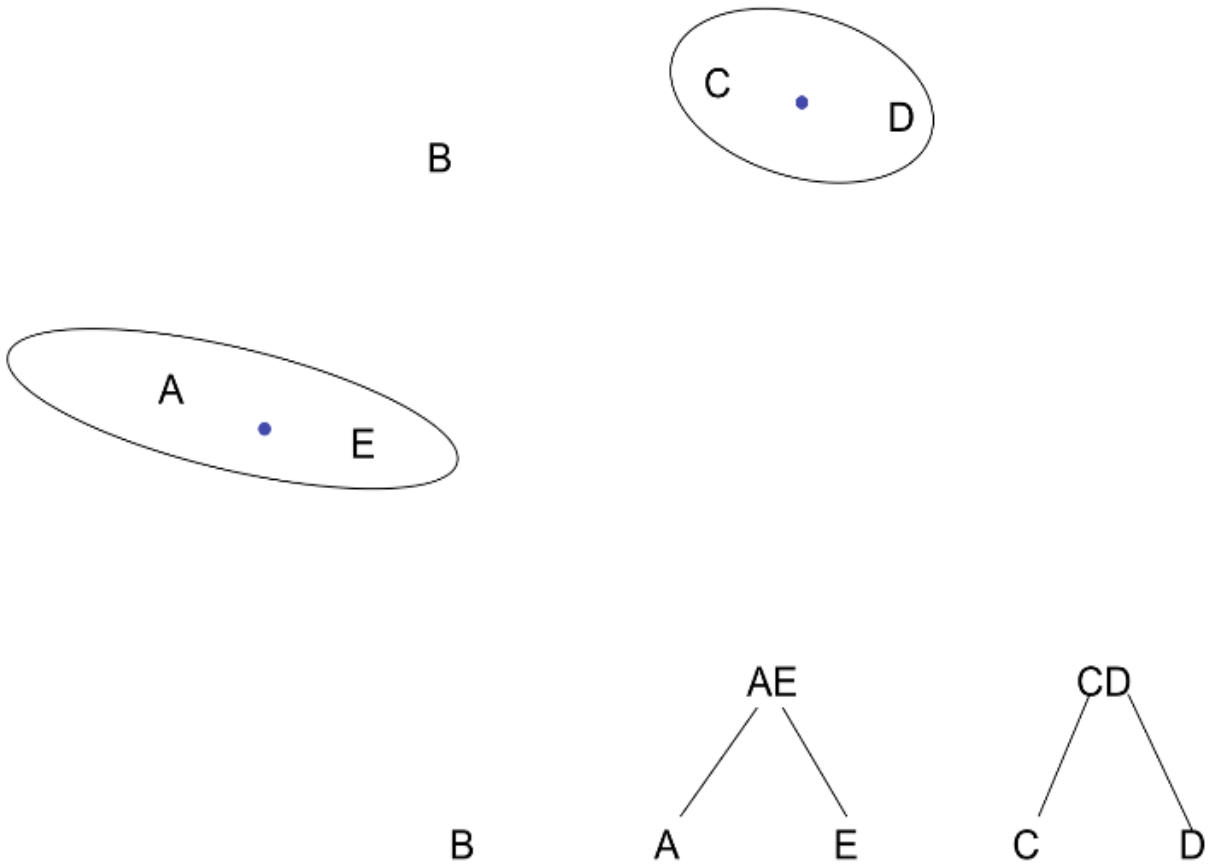
Example: HAC with Centroid Distance (2 of 7)



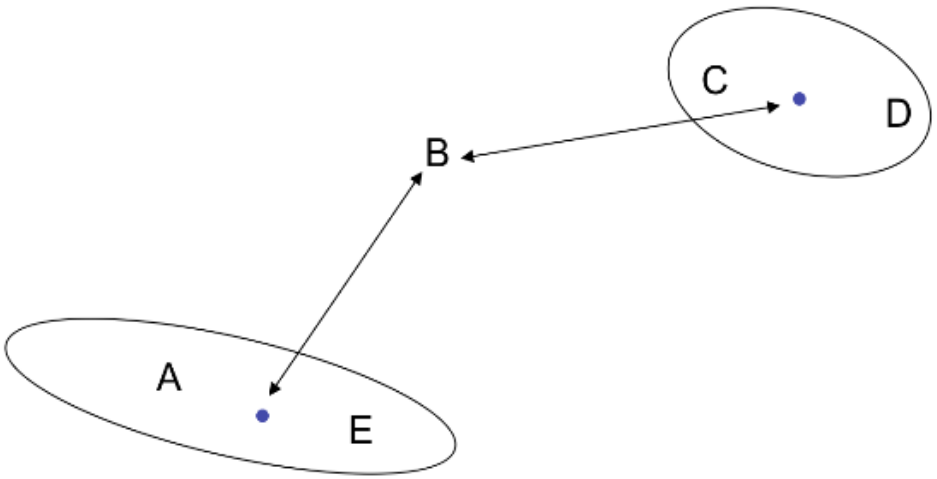
Example: HAC with Centroid Distance (3 of 7)

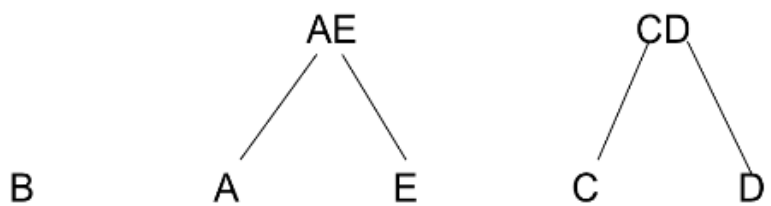


Example: HAC with Centroid Distance (4 of 7)

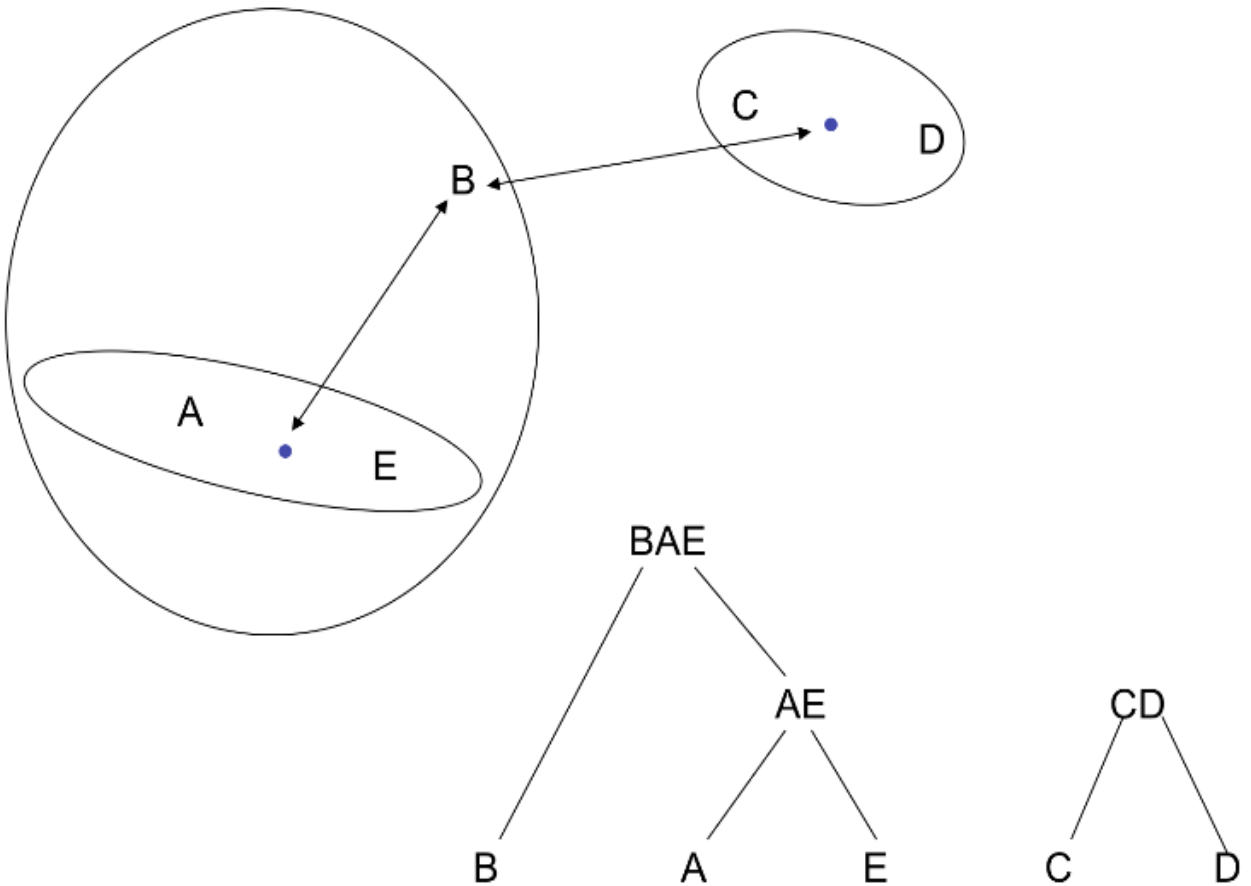


Example: HAC with Centroid Distance (5 of 7)

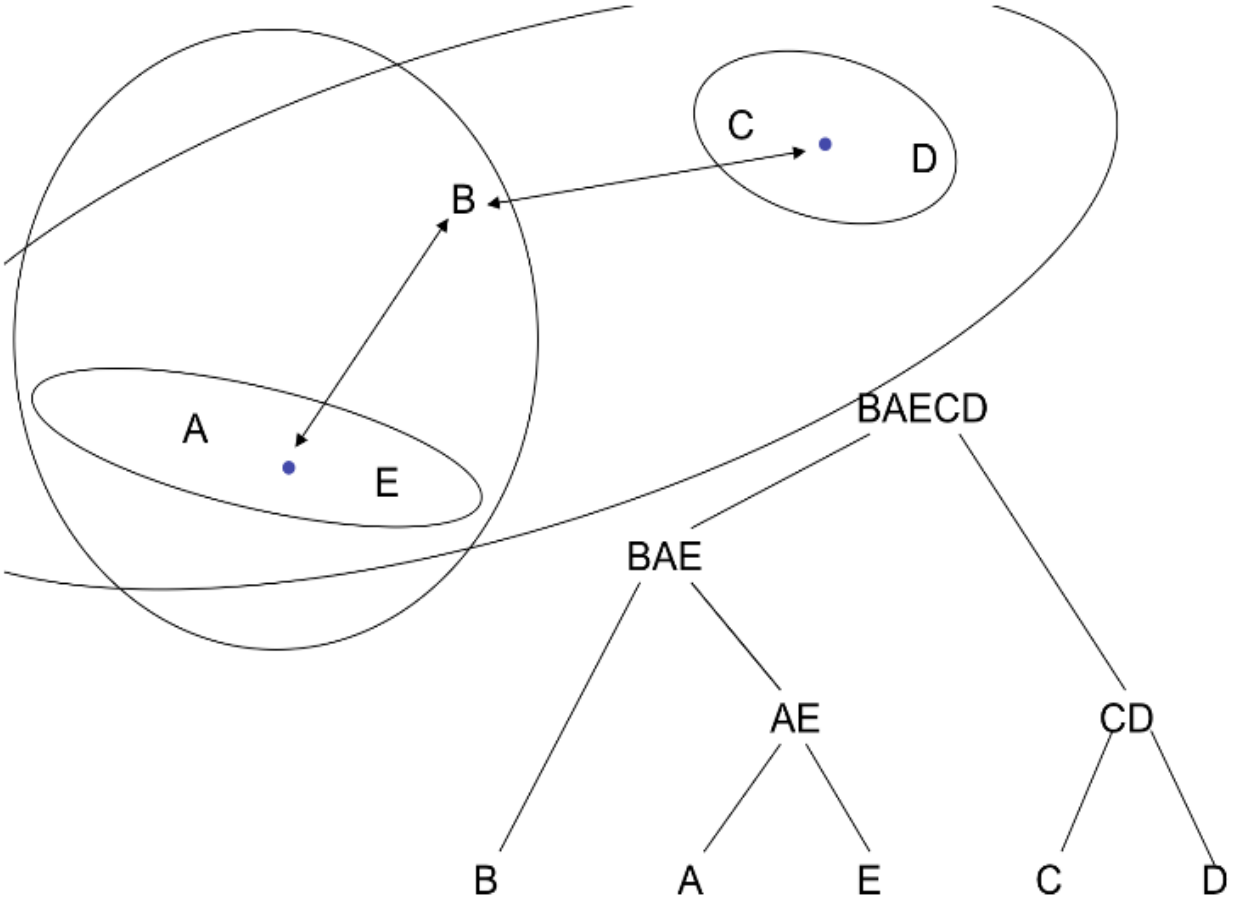




Example: HAC with Centroid Distance (6 of 7)



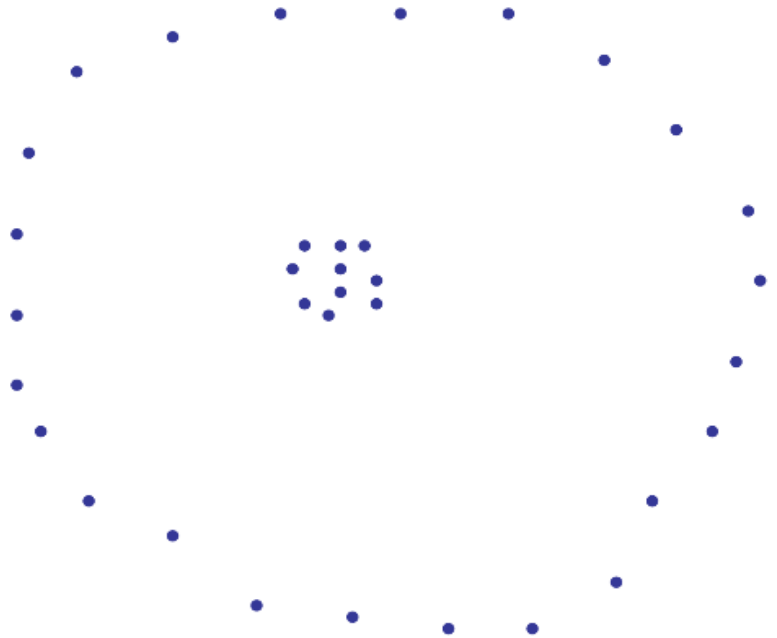
Example: HAC with Centroid Distance (7 of 7)



Comparing HAC Group Distance Criteria

1. Which distance will tend to merge large clusters with each other?
   A: min, because larger clusters are more likely to have a pair of examples that are close.
2. Which distance will tend to have a "chaining effect" and lead to long, stringy clusters?
   A: min, since only one distance has to be small.
3. Which distance will tend to prefer compact clusters?
   A: max, since all distances have to be small to merge.

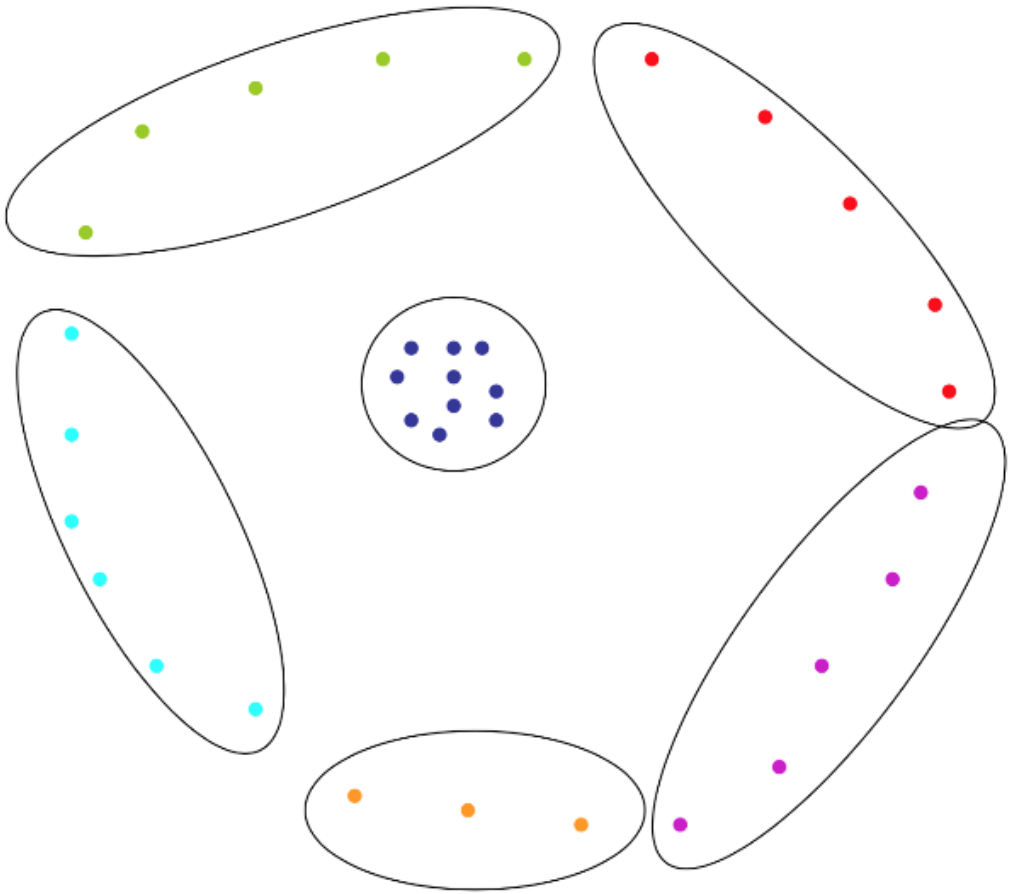
Simple HAC Example



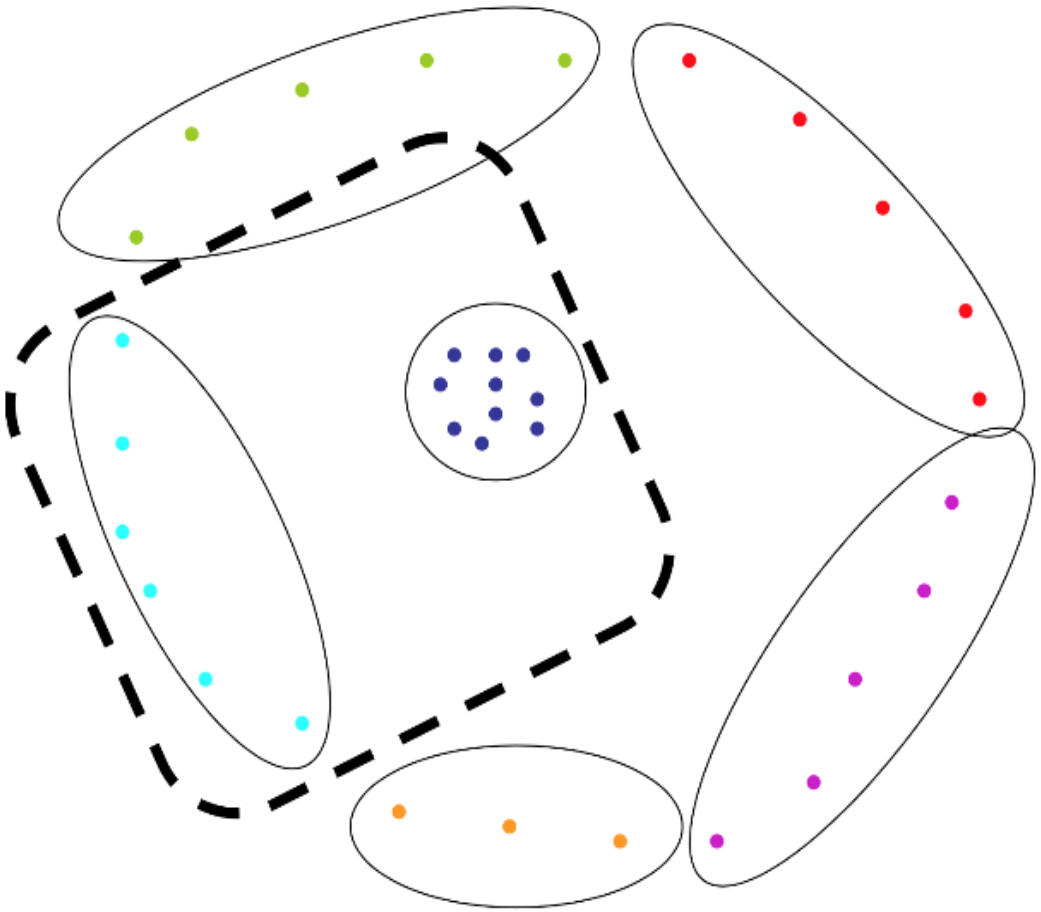
Which HAC distance will find the clusters?

A: min. What will max do?

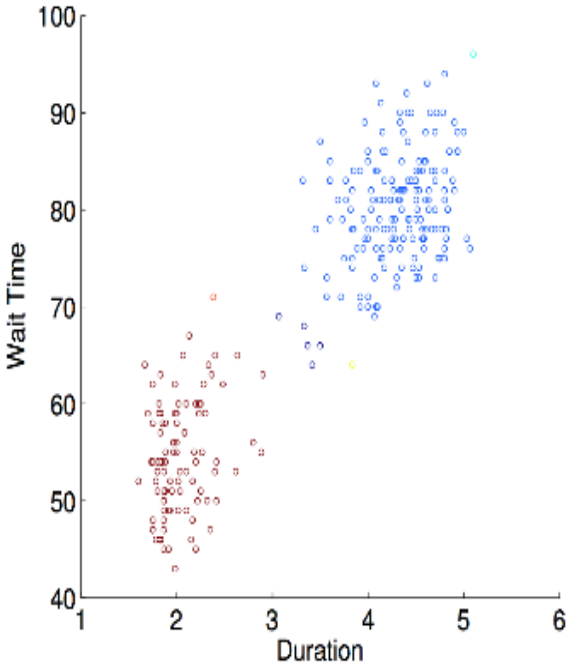
Simple HAC Example {Max (1 of 2)}



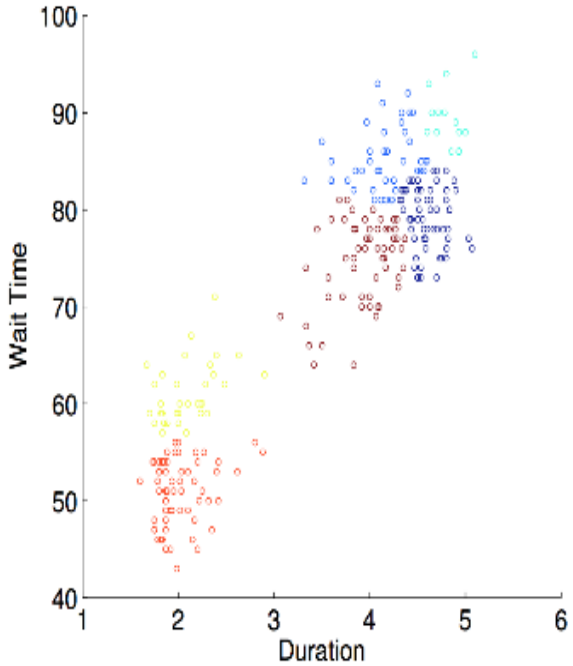
Simple HAC Example {Max (2 of 2)}



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

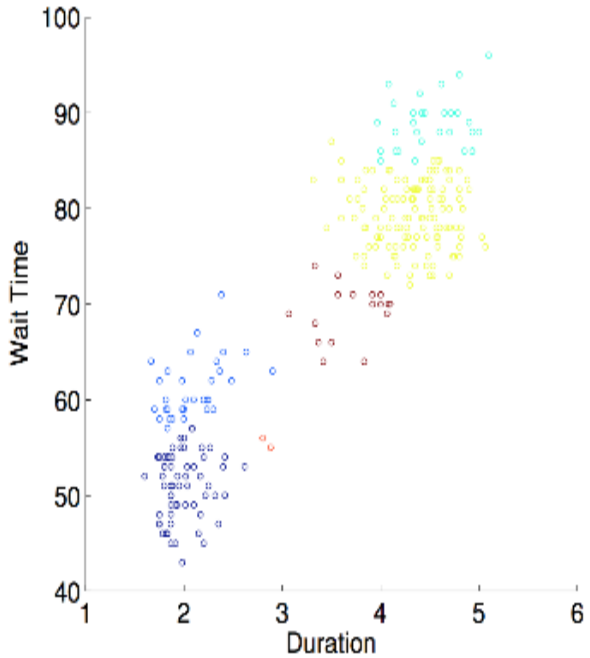


(a) min distance

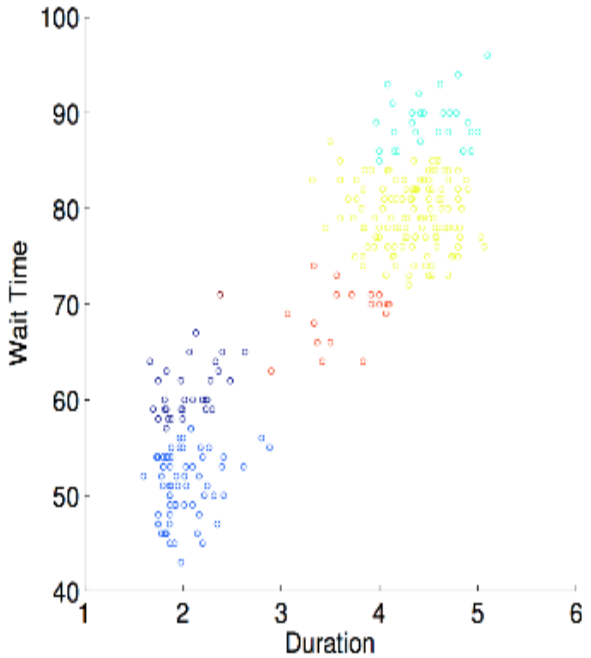


(b) max distance

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

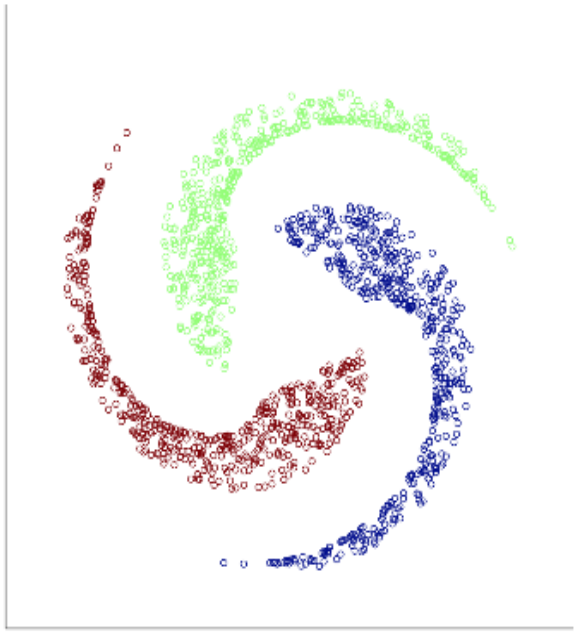


(c) average distance

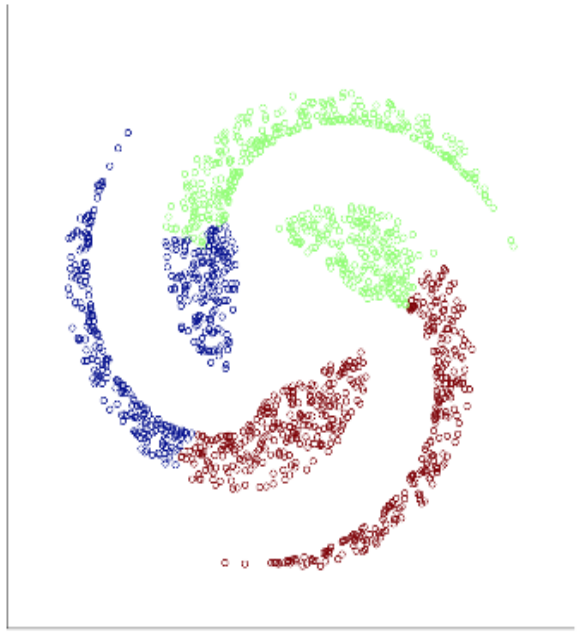


(d) centroid distance

Example: HAC Clustering on Pinwheel example (1 of 2)

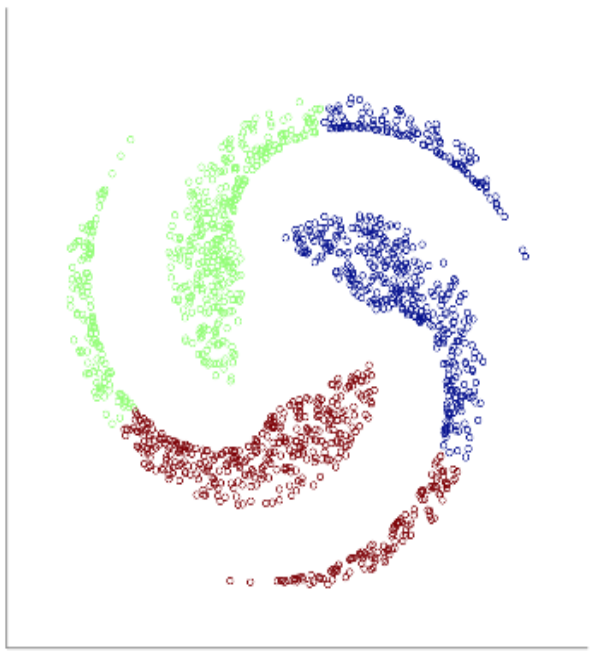


(a) min distance

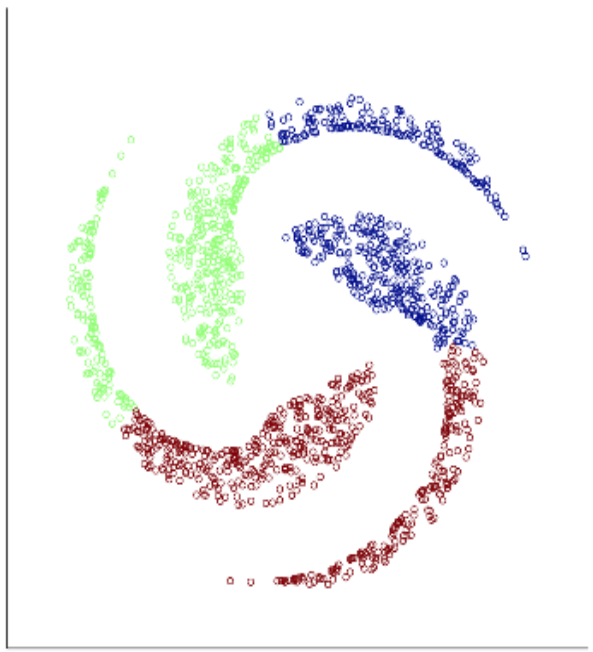


(b) max distance

Example: HAC Clustering 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 checks. $T \ll m$, and thus O(n^2m). (c.f., O($nKmT$) for K-means.)

Noise

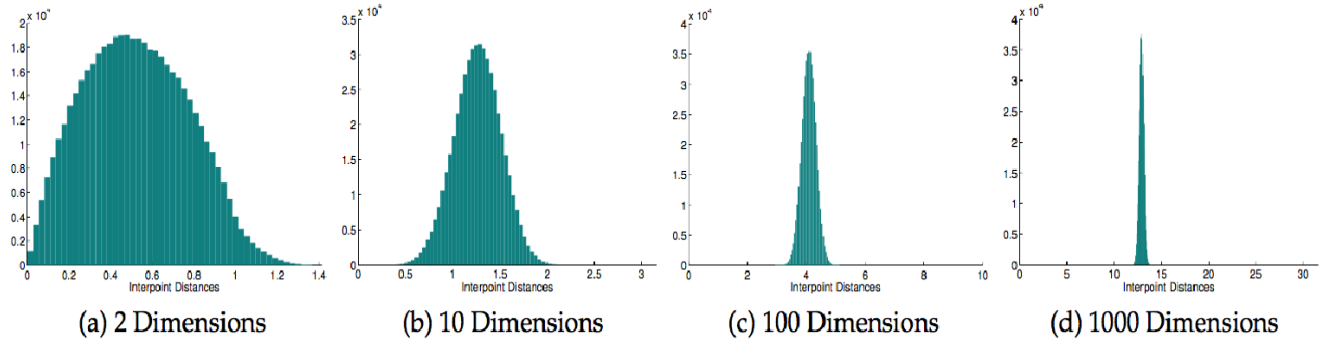
Suppose the data looks like this:

	ℓ				$m-\ell$
$\mathbf{x}_1$	1	1	1	1	random
$\mathbf{x}_2$	1	1	1	1	random
$\mathbf{x}_3$	0	0	0	0	random
$\mathbf{x}_4$	1	1	1	1	random
$\mathbf{x}_5$	0	0	0	0	random

Given $\mathbf{x}, \hat{\mathbf{x}}$ in the "1s" cluster and $\mathbf{z}$ in the "0s" cluster, the probability that $\mathbf{x}$ is closer to $\hat{\mathbf{x}}$ than to $\mathbf{z}$ goes to $1/2$ as $m \to \infty$, because the noise comes to dominate.
Note: K-means is OK here! $\boldsymbol{\mu}_1 = (1, 1, 1, 1, 0.5, 0.5, \ldots)$ and $\boldsymbol{\mu}_2 = (0, 0, 0, 0, 0.5, 0.5, \ldots)$, and it correctly clusters the data.

Curse of Dimensionality

Histograms of inter-example distances for 1000 examples 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 (distances are 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.

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.