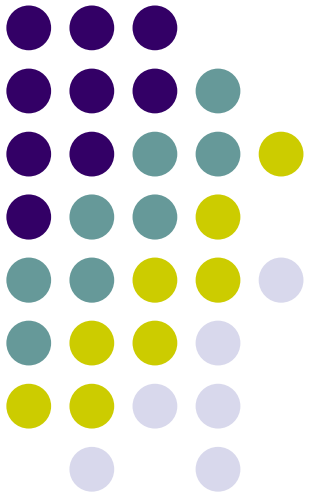
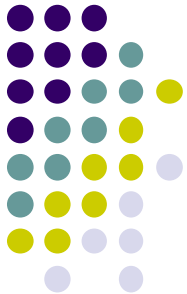
# Clustering

J. Savoy
University of Neuchatel



Ian H. Witten, Eibe Frank: Data Mining. Practical Machine Learning Tools and Techniques. Morgan Kaufmann, 2005.

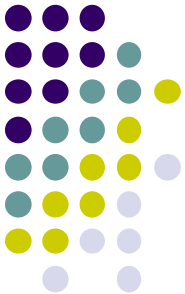
D. Ullman: Course on Data Mining.



# Overview

- **Examples and Applications**
- Hierarchical Clustering
- Cluster Representatives
- *k*-Means Algorithms

# The Problem of Clustering

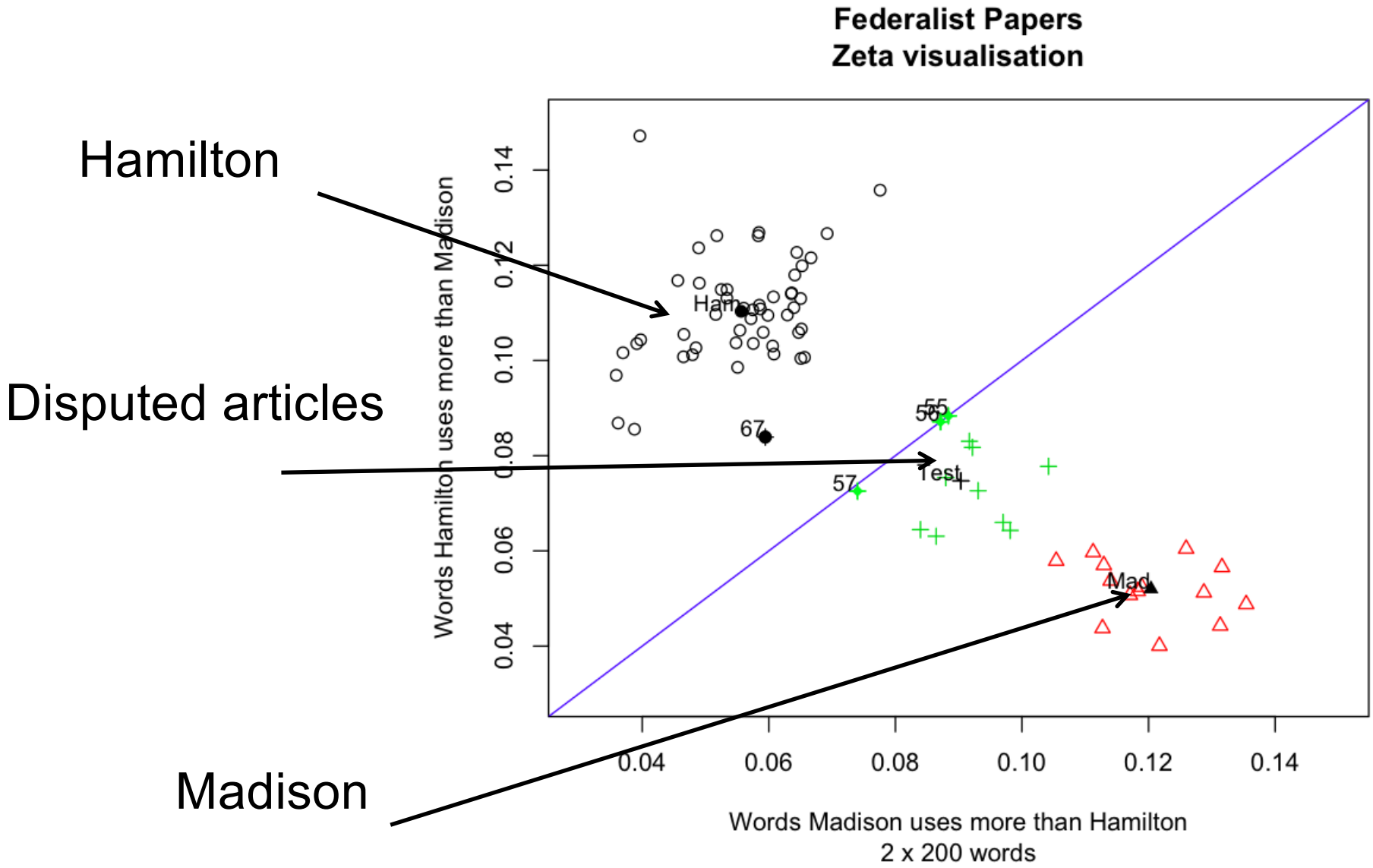


- Objective:
  Given a set of points, with a notion of *distance* between points, group the points into some number of *clusters (groups or classes)*, so that members of a cluster are in *some sense* as close to each other as possible.
- In other words: Having a (large) number of objects, can we organize them into different classes.
  We do not predict a class (it is not a text categorization problem) but we had to find the "natural" classes
- *Unsupervised* learning: we don't have a teaching signal (a set of examples and counter-examples)
- Some examples

# Example

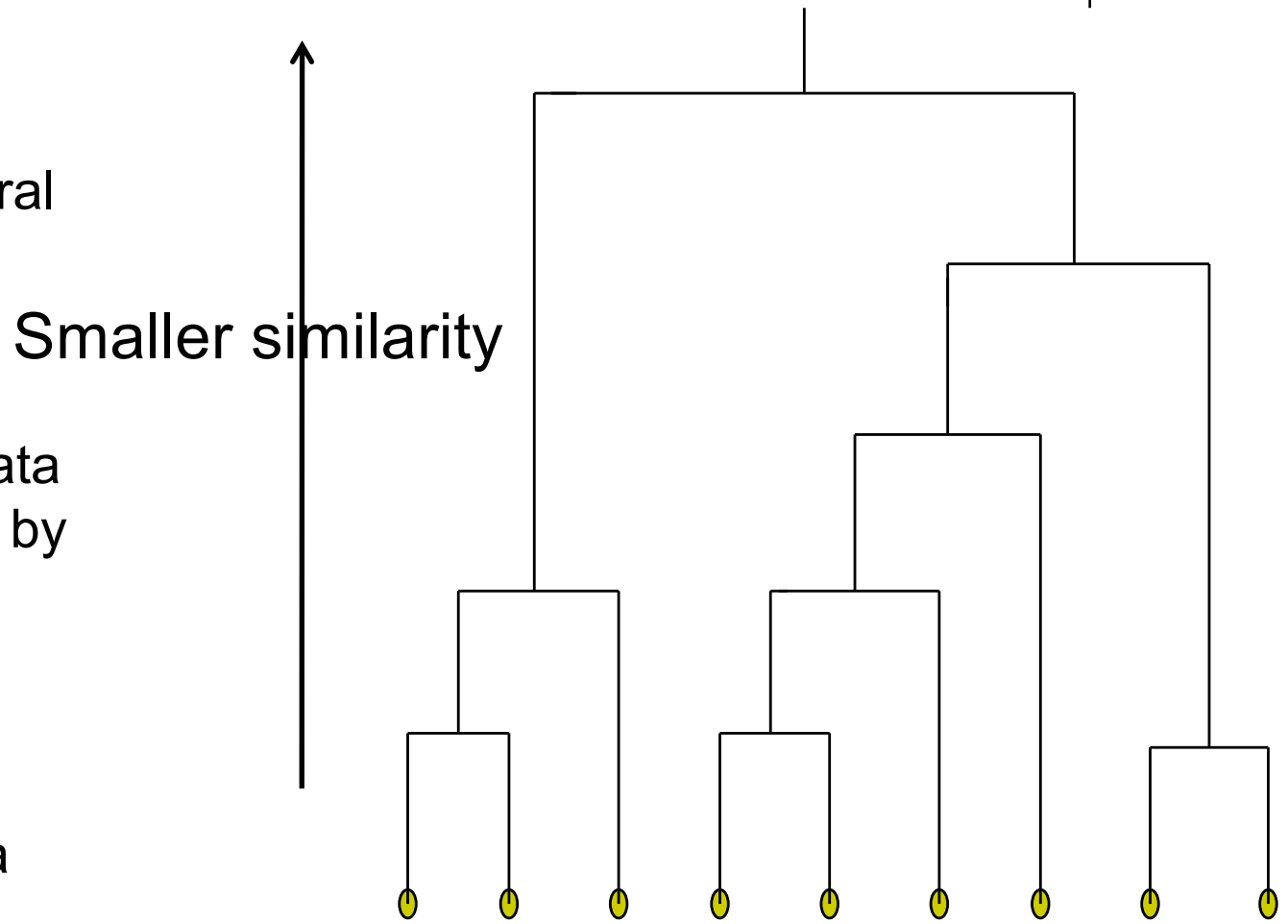
- Classify these pictures

# Authorship Attribution



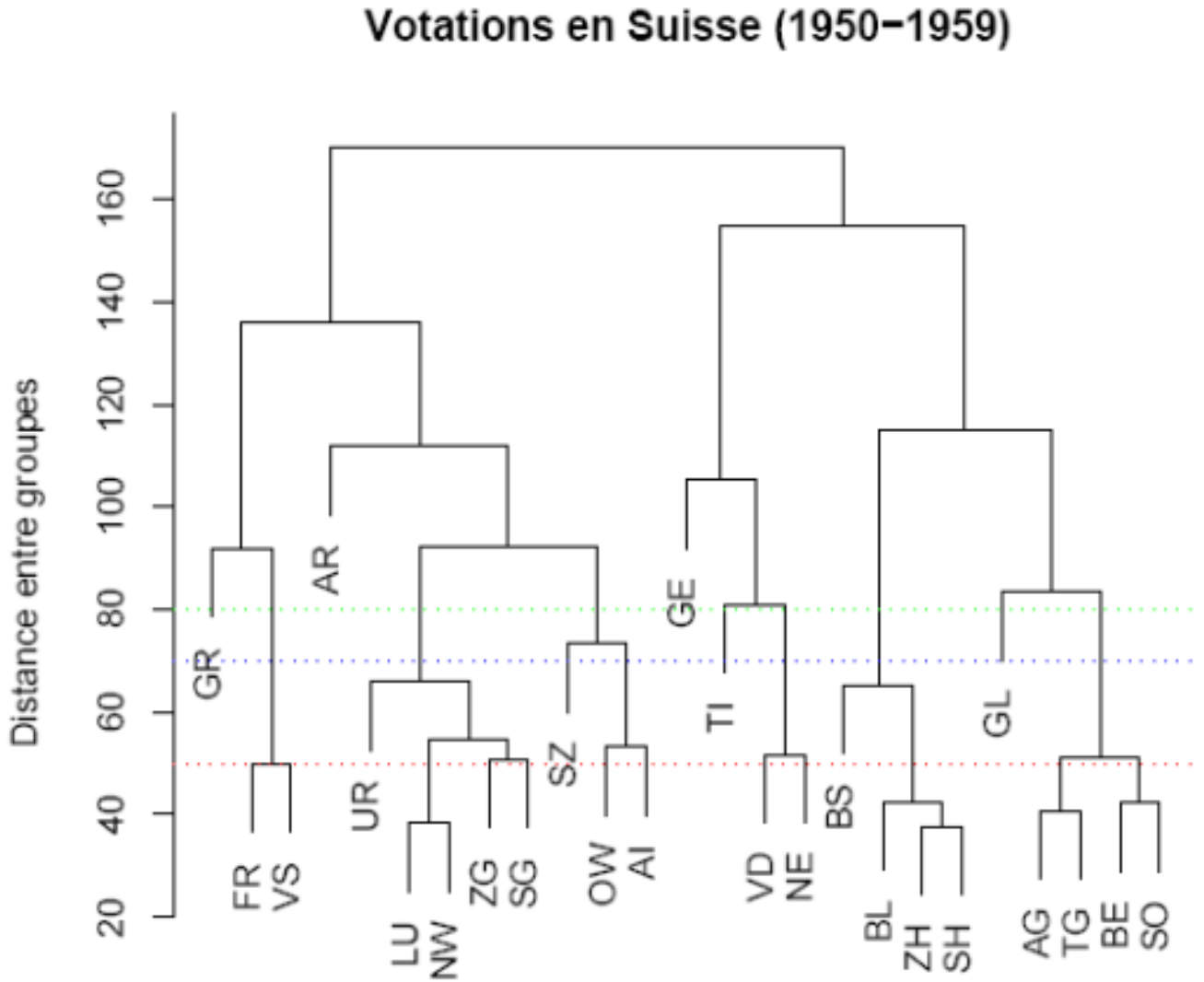
# Clustering

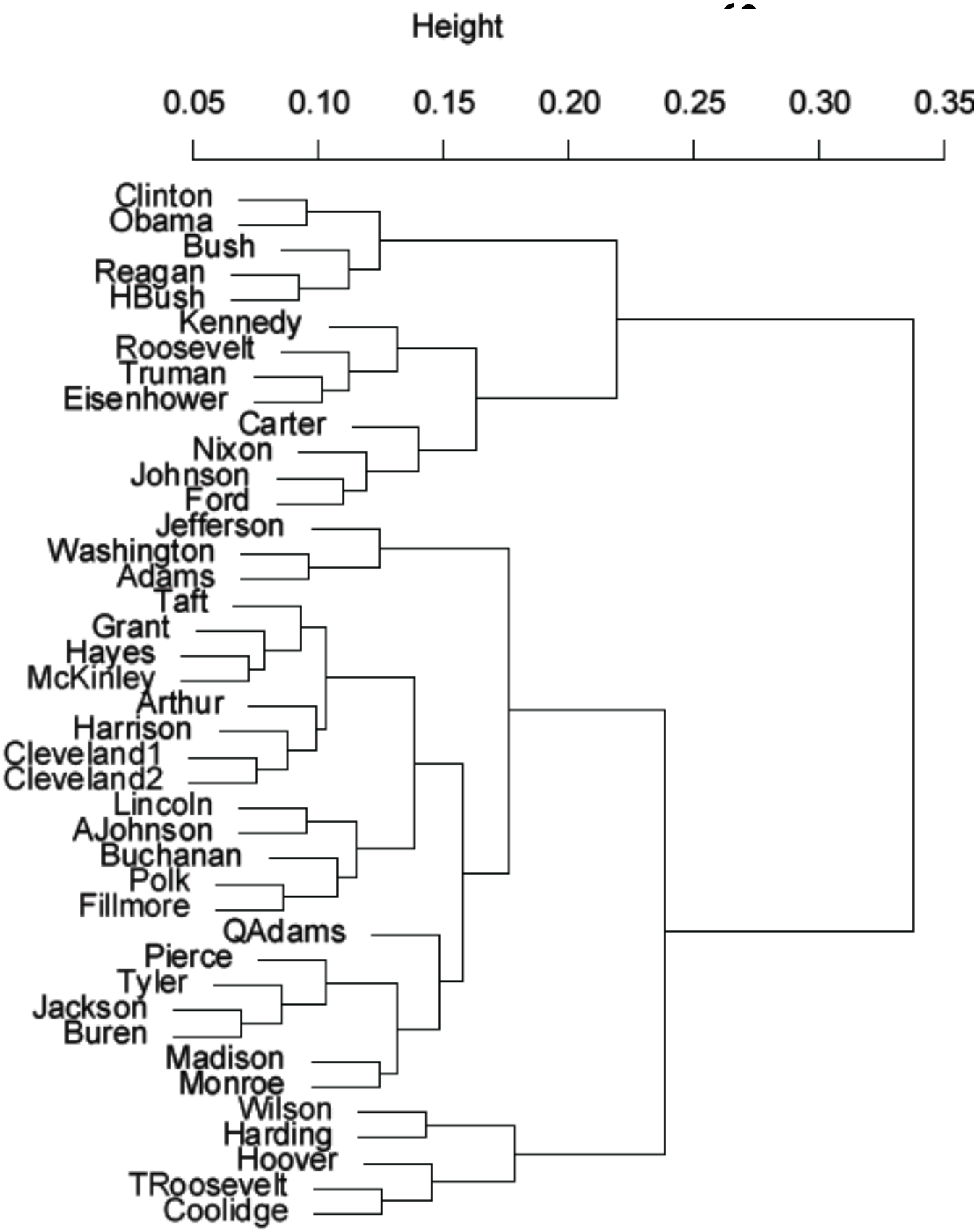
- Dendrogram: Decomposes data objects into a several levels of nested partitioning (tree of clusters).
- Clustering of the data objects is obtained by cutting the dendrogram at the desired level, then each connected component forms a cluster.



# Example

Dendrogram
(tree diagram)



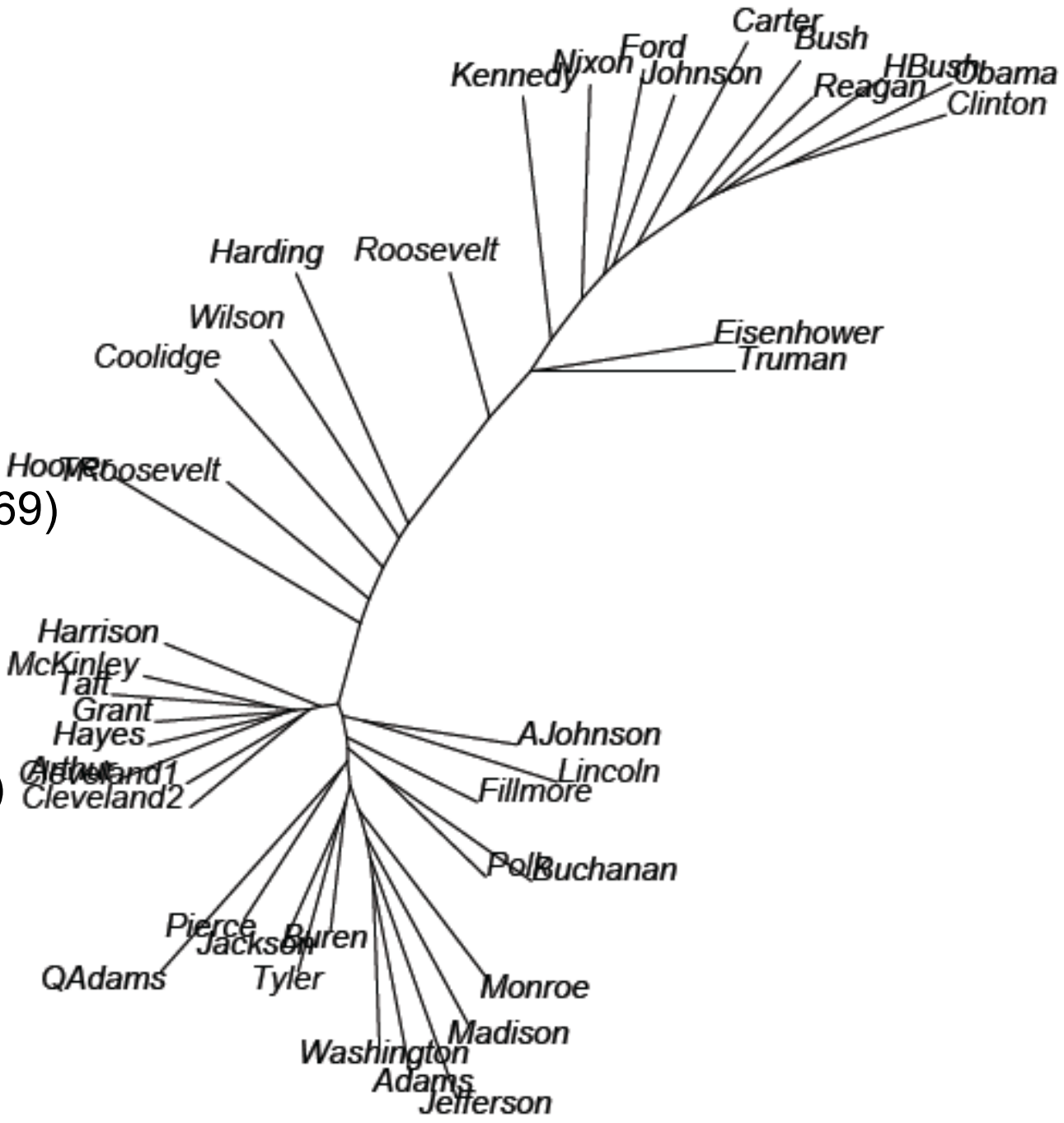

Height
0.05
0.10
0.15
0.20
0.25
0.30
0.35
Clinton
Obama
Bush
Reagan
HBush
Kennedy
Roosevelt
Truman
Eisenhower
Carter
Nixon
Johnson
Ford
Jefferson
Washington
Adams
Taft
Grant
Hayes
McKinley
Arthur
Harrison
Cleveland1
Cleveland2
Lincoln
AJohnson
Buchanan
Polk
Fillmore
QAdams
Pierce
Tyler
Jackson
Buren
Madison
Monroe
Wilson
Harding
Hoover
TRoosevelt
Coolidge

# Example

Stylistic features

Smallest distance:
Jackson – Van Buren (0.069)
Hayes – McKinley (0.072)

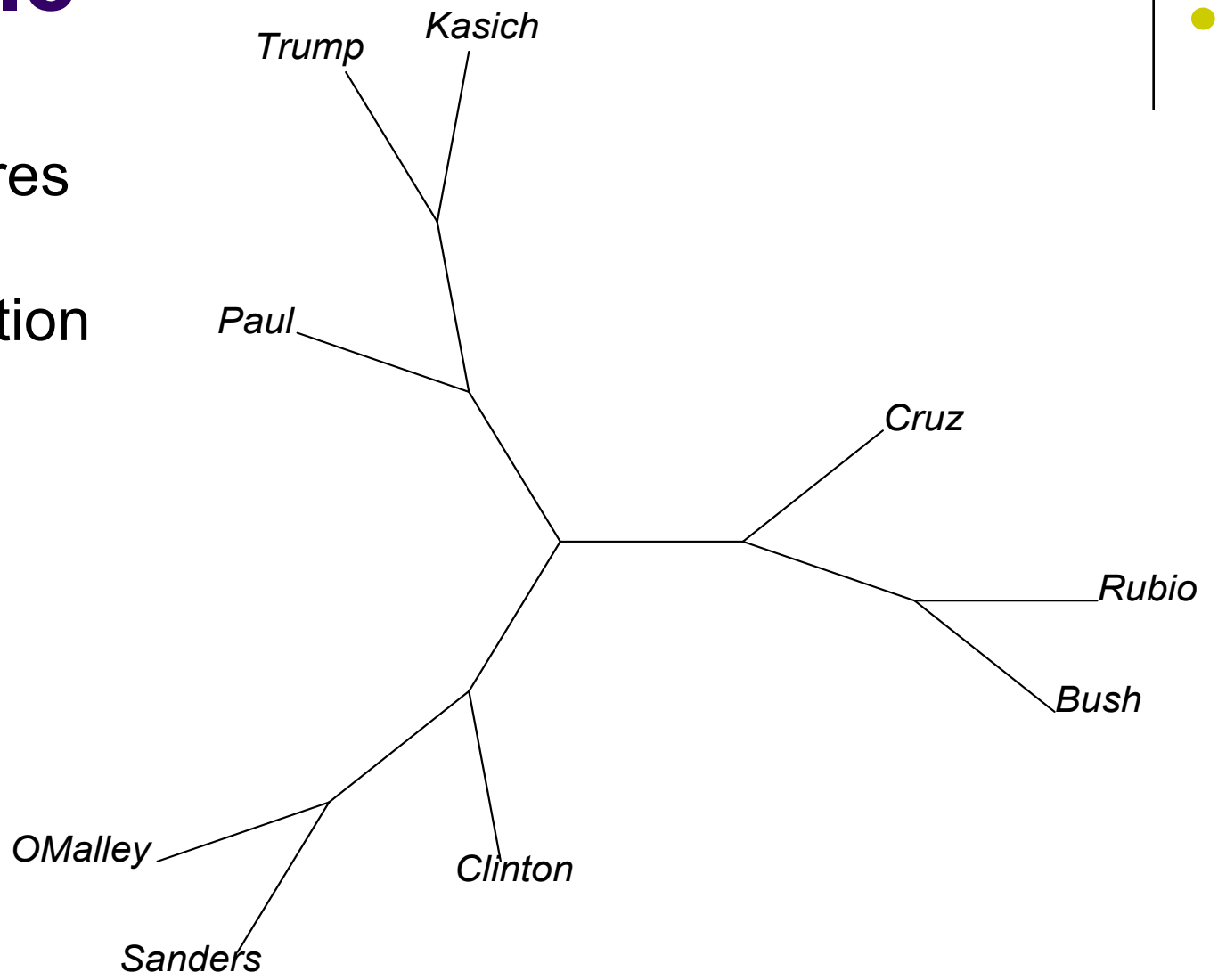
Longest distance:
QAdams – Obama (0.337)
QAdams – Clinton (0.336)



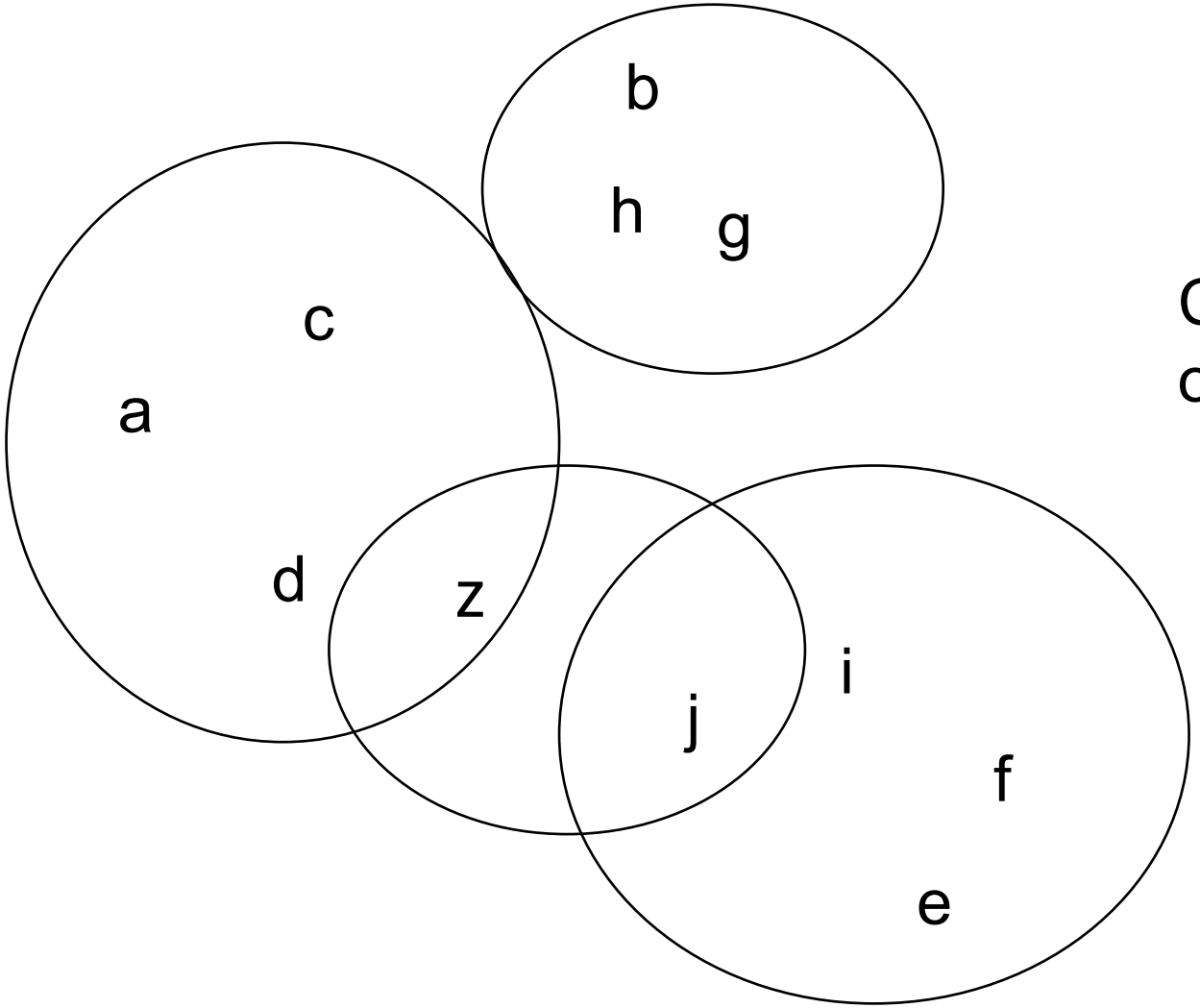
# Example

Stylistic features

2016 US election

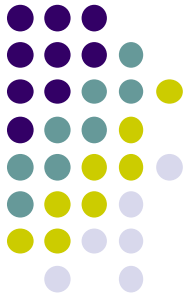


# Example




Clustering with overlapping

# Example



Objects (A, B, … F) with cluster probabilities

| | Group1 | Group2 | Group3 |
|---|---|---|---|
| A | 0.8 | 0.1 | 0.1 |
| B | 0.7 | 0.2 | 0.1 |
| C | 0.2 | 0.5 | 0.3 |
| D | 0.3 | 0.1 | 0.6 |
| E | 0.3 | 0.7 | 0.0 |
| F | 0.4 | 0.1 | 0.5 |

# Example

## Cantons suisses

Les chiffres se réfèrent en règle générale à l'année 2006

| | ZH | BE | LU | UR | SZ | OW | NW | GL | ZG | FR | SO | BS | BL | SH |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Entrée dans la Confédération | 1351 | 1353 | 1332 | 1291 | 1291 | 1291 | 1291 | 1352 | 1352 | 1481 | 1481 | 1501 | 1501 | 1501 |
| Nombre de communes[7] | 171 | 396 | 96 | 20 | 30 | 7 | 11 | 25 | 11 | 168 | 125 | 3 | 86 | 32 |
| Conseillers nationaux, nombre | 34 | 26 | 10 | 1 | 4 | 1 | 1 | 1 | 3 | 7 | 7 | 5 | 7 | 2 |
| Superficie[1] en km² | 1 729 | 5 959 | 1 493 | 1 077 | 908 | 491 | 276 | 685 | 239 | 1 671 | 791 | 37 | 518 | 298 |
| Surface agricole utile[1] en % | 43,4 | 43,3 | 54,7 | 24,4 | 40,9 | 37,9 | 37,9 | 30,5 | 44,8 | 57,3 | 43,4 | 12,1 | 41,3 | 45,0 |

## Cantons suisses

Les chiffres se réfèrent en règle générale à l'année 2006

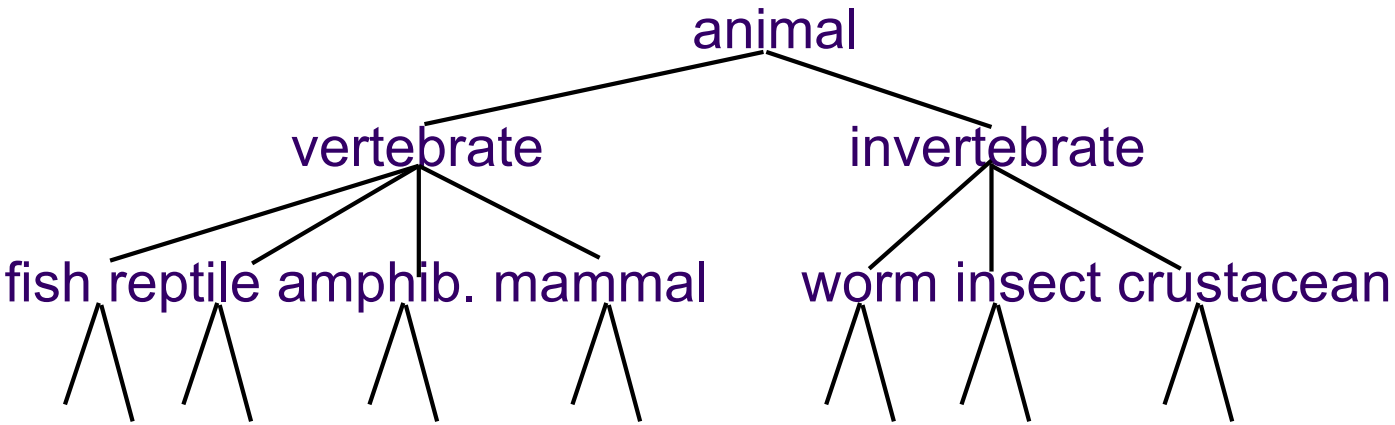
| | AR | AI | SG | GR | AG | TG | TI | VD | VS | NE | GE | JU | CH |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Entrée dans la Confédération | 1513 | 1513 | 1803 | 1803 | 1803 | 1803 | 1803 | 1803 | 1815 | 1815 | 1815 | 1979 | |
| Nombre de communes[7] | 20 | 6 | 88 | 206 | 229 | 80 | 190 | 378 | 153 | 62 | 45 | 83 | 2 721 |
| Conseillers nationaux, nombre | 1 | 1 | 12 | 5 | 15 | 6 | 8 | 18 | 7 | 5 | 11 | 2 | 200 |
| Superficie[1] en km² | 243 | 173 | 2 026 | 7 105 | 1 404 | 991 | 2 812 | 3 212 | 5 224 | 803 | 282 | 838 | 41 284 |
| Surface agricole utile[1] en % | 56,1 | 55,7 | 47,9 | 29,8 | 45,3 | 53,2 | 14,3 | 43,4 | 20,3 | 42,0 | 41,5 | 49,3 | 36,9 |

## Comparaison avec d'autres pays en 2006

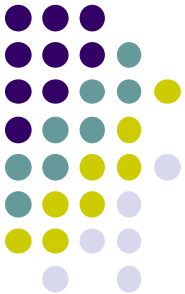
| | Zone euro (12) | Allemagne | France | Italie | Autriche | Grande-Bretagne | Etats-Unis | Inde | Japon | Chine | Suisse |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Résidents[18, 19] en millions | 314,6 | 82,4 | 62,9 | 58,8 | 8,3 | 60,4 | 299,4[20] | 1 095,4 | 127,8 | 1 314,0 | 7,6[21] |
| Surface en 1000 km² | 2 501,3 | 357,0 | 547,0 | 301,2 | 83,9 | 244,8 | 9 826,6 | 3 287,6 | 377,8 | 9 597,0 | 41,3 |
| Nombre d'habitants[19] par km² | 126 | 231 | 115 | 195 | 99 | 247 | 30[20] | 333 | 338 | 137 | 182[21] |
| Population active[18, 22] en millions | 147,6 | 40,9 | 27,6 | 24,8 | 4,1 | 30,0 | 151,4 | 509,3 | 66,5 | 766,7[11] | 4,2 |
| Taux de chômage[23] en % | 7,9 | 8,4 | 9,4 | 6,8 | 4,8 | 5,3 | 4,6 | 7,8 | 4,1 | 4,2[11] | 3,3 |
| Produit intérieur brut en mia. de USD | 10 553,2 | 2 894,3 | 2 227,3 | 1 837,4 | 3 21,6 | 2 370,6 | 13 183,0 | 854,5 | 4 364,8 | 2 631,8 | 378,8 |
| Variation en termes réels 05/06 en % | +2,8 | +2,7 | +2,1 | +1,9 | +2,8 | +2,8 | +3,3 | +7,8 | +2,2 | +10,0 | +2,7 |
| Produit intér. brut par habitant en USD[24] | 33 461 | 35 108 | 35 416 | 31 275 | 38 906 | 39 253 | 44 032 | 780 | 34 161 | 2 003 | 50 120 |

# Example

Build a tree-based hierarchical taxonomy (*dendrogram*) from a set of unlabeled examples.
Classification done by Carl von Linné (1707-1778)



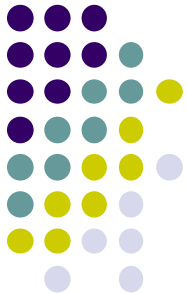

# Example



- Whole corpus analysis/navigation
  - Better user interface
  - Can organize a set of objects
  - Use in the web

    (set of labels not pre-defined!)
- For improving recall in search applications
  - Better search results
- For better navigation of search results
  - Effective “user recall” will be higher

# Example



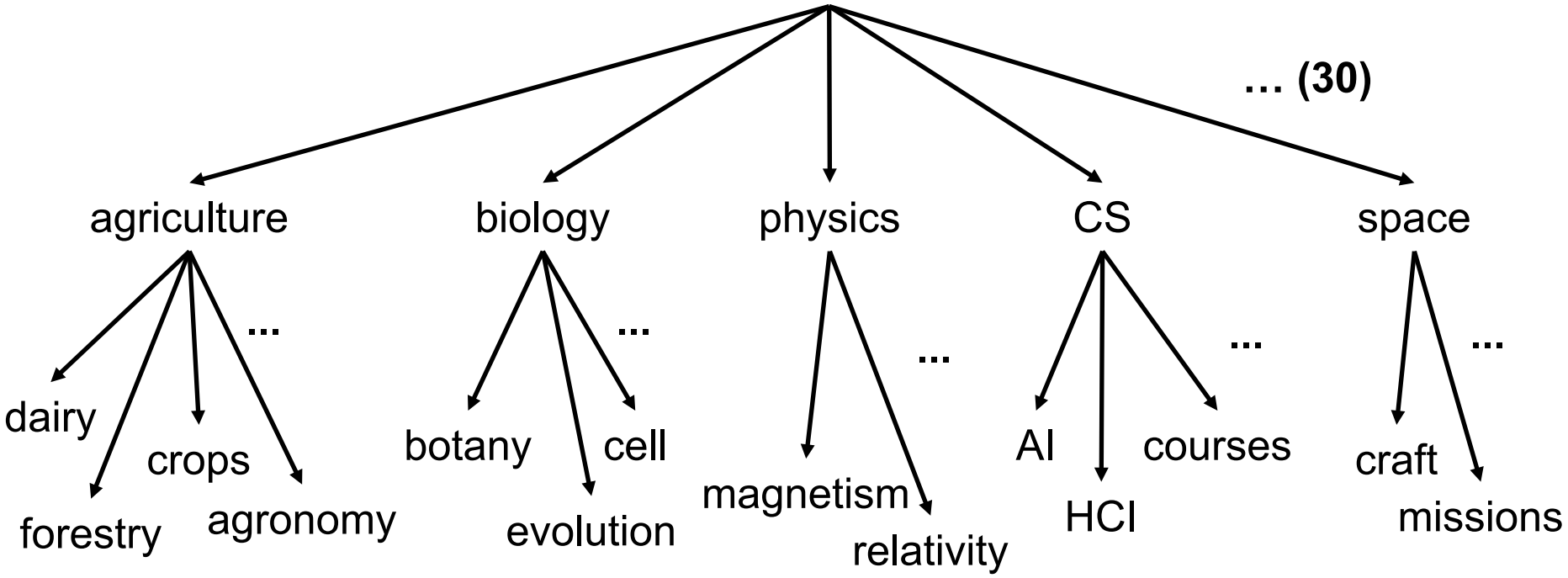
- Document clusters are like a table of contents
- People find having a table of contents useful

***Table of Contents***

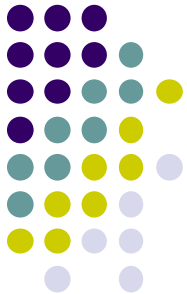
1. Science of Cognition
    - 1.a. Motivations
        - 1.a.i. Intellectual Curiosity
        - 1.a.ii. Practical Applications
    - 1.b. History of Cognitive Psychology
2. The Neural Basis of Cognition
    - 2.a. The Nervous System
    - 2.b. Organization of the Brain
    - 2.c. The Visual System
3. Perception and Attention
    - 3.a. Sensory Memory
    - 3.b. Attention and Sensory Information Processing

# Example: Yahoo! Hierarchy

www.yahoo.com/Science (other example DMOZ hierarchy)

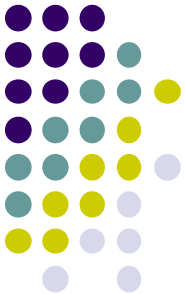


# The Problem of Clustering



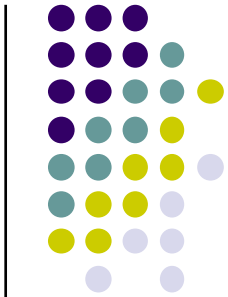
- Clustering is the process of grouping a set of physical or abstract objects into classes of similar objects
  - It is the commonest form of unsupervised learning
    - Unsupervised learning = learning from raw data, as opposed to supervised data where the correct classification of examples is given
  - It is a common and important task that finds many applications (Data Mining, IR, Web)
- Clustering in two dimensions looks easy.
- Clustering small amounts of data looks easy.
- And in most cases, looks are *not* deceiving.
- So where are the real problems?

# The Curse of Dimensionality



- The ***curse of dimensionality*** is a term coined by R. Bellman to describe the problem caused by the exponential increase in volume associated with adding extra dimensions to a (mathematical) space
  (e.g., 100 evenly-spaced sample points suffice to sample a unit interval with no more than 0.01 distance between points)
  The probability of random points being close drops quickly as the dimensionality grows.
- Many applications involve not 2, but 10 or 10,000 dimensions.
- High-dimensional spaces look different: almost all pairs of points are at about the same distance.

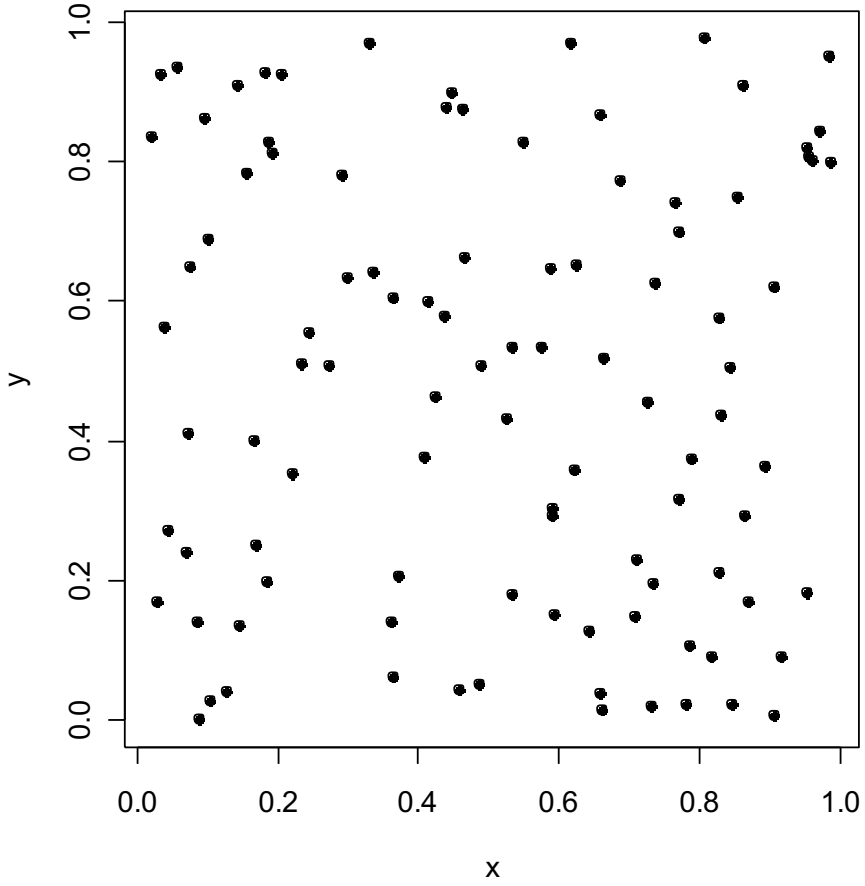
# The Curse of Dimensionality



Example: assume random points within a bounding box, e.g., values between 0 and 1 in each dimension.

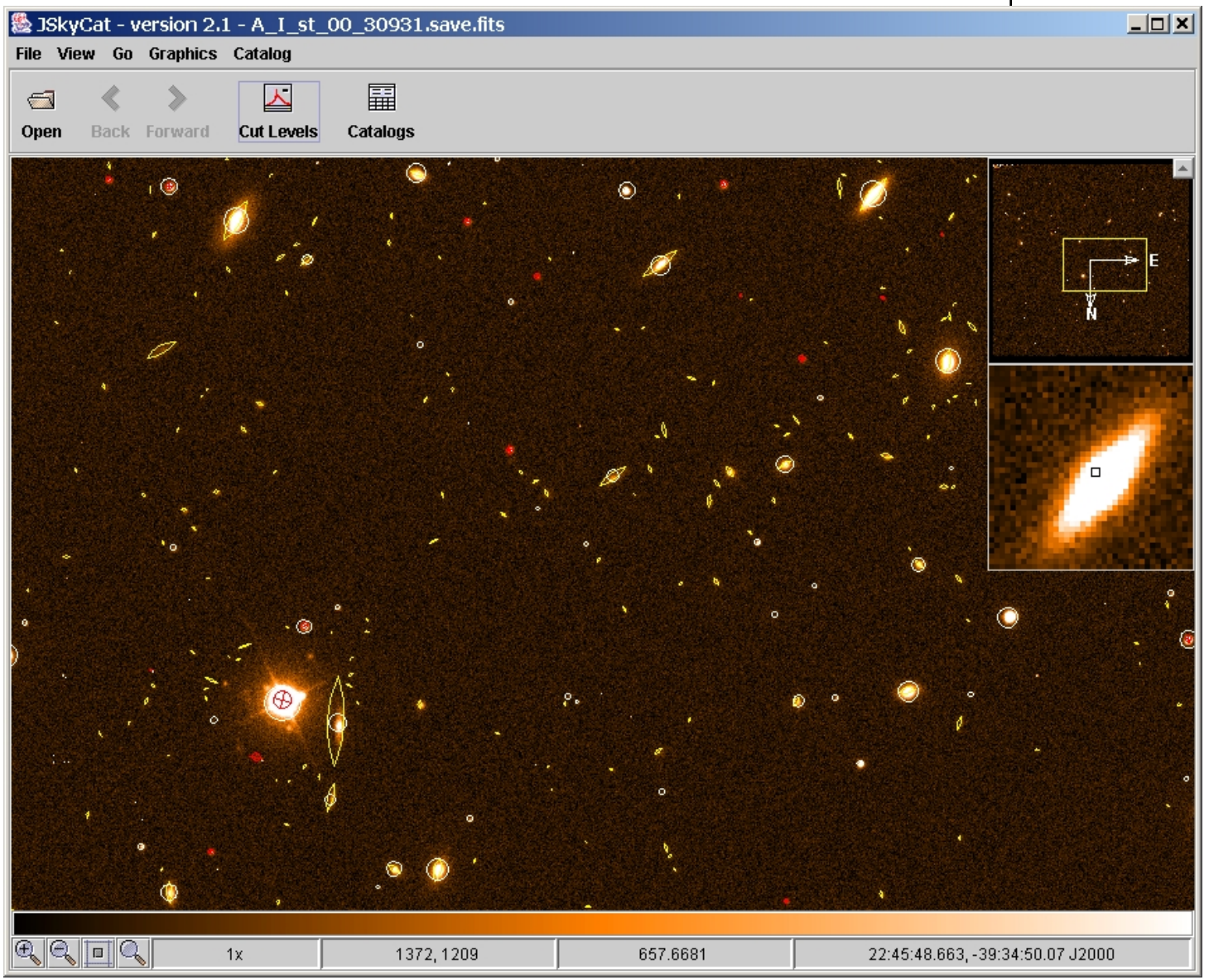
(example in 2D)

How can we cluster them?



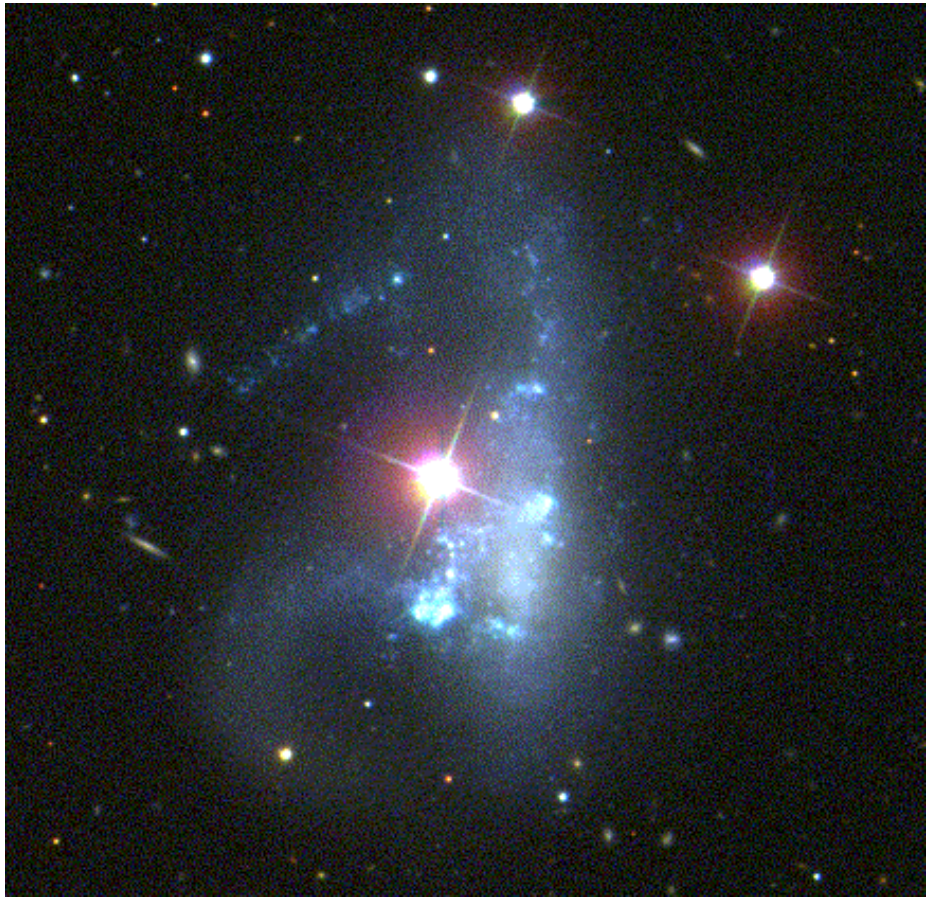
# Example: SkyCat

*SkyCat* is a tool that combines visualization of images and access to catalogs and archive data for astronomy.

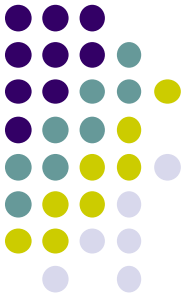


# Example: SkyCat

- *SkyCat* contains around 2 billion “sky objects” represents objects by their radiation in 9 dimensions (frequency bands).
- Problem: cluster into similar objects, e.g., galaxies, nearby stars, quasars, etc.

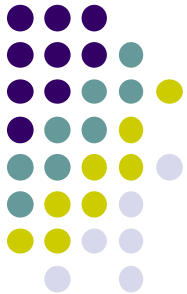


# Example: Clustering CDs (Collaborative Filtering)



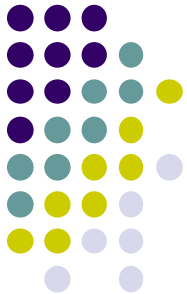
- Intuitively: music divides into categories, and customers prefer a few categories.
  - But what are categories really?
- Represent a CD by the customers who bought it.
- Similar CDs have similar sets of customers, and vice-versa.
- Could be useful to improve your sales (intelligent purchase suggestions)

# The Space of CDs



- Think of a space with one dimension for each customer.
  - Values in a dimension may be 0 or 1 only.
- A CD point in this space is $(x_1, x_2, \ldots, x_k)$, where $x_i = 1$ iff the $i^{th}$ customer bought the CD.
  - Compare with the "shingle/signature" matrix rows = customers; cols. = CDs.
- For Amazon, the dimension count is tens of millions.
- An option: use minhashing / LSH to get Jaccard similarity between "close" CDs.
- 1 minus Jaccard similarity can serve as a (non-Euclidean) distance.

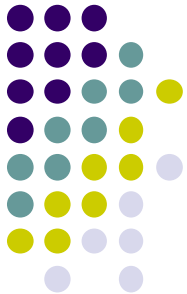
# Example: Clustering Documents



- Represent a document by a vector ($x_1$, $x_2$,…, $x_k$), where $x_i$ = 1 iff the $i^{th}$ word (in some order) appears in the document.
  - It actually doesn't matter if $k$ is infinite; i.e., we don't limit the set of words.

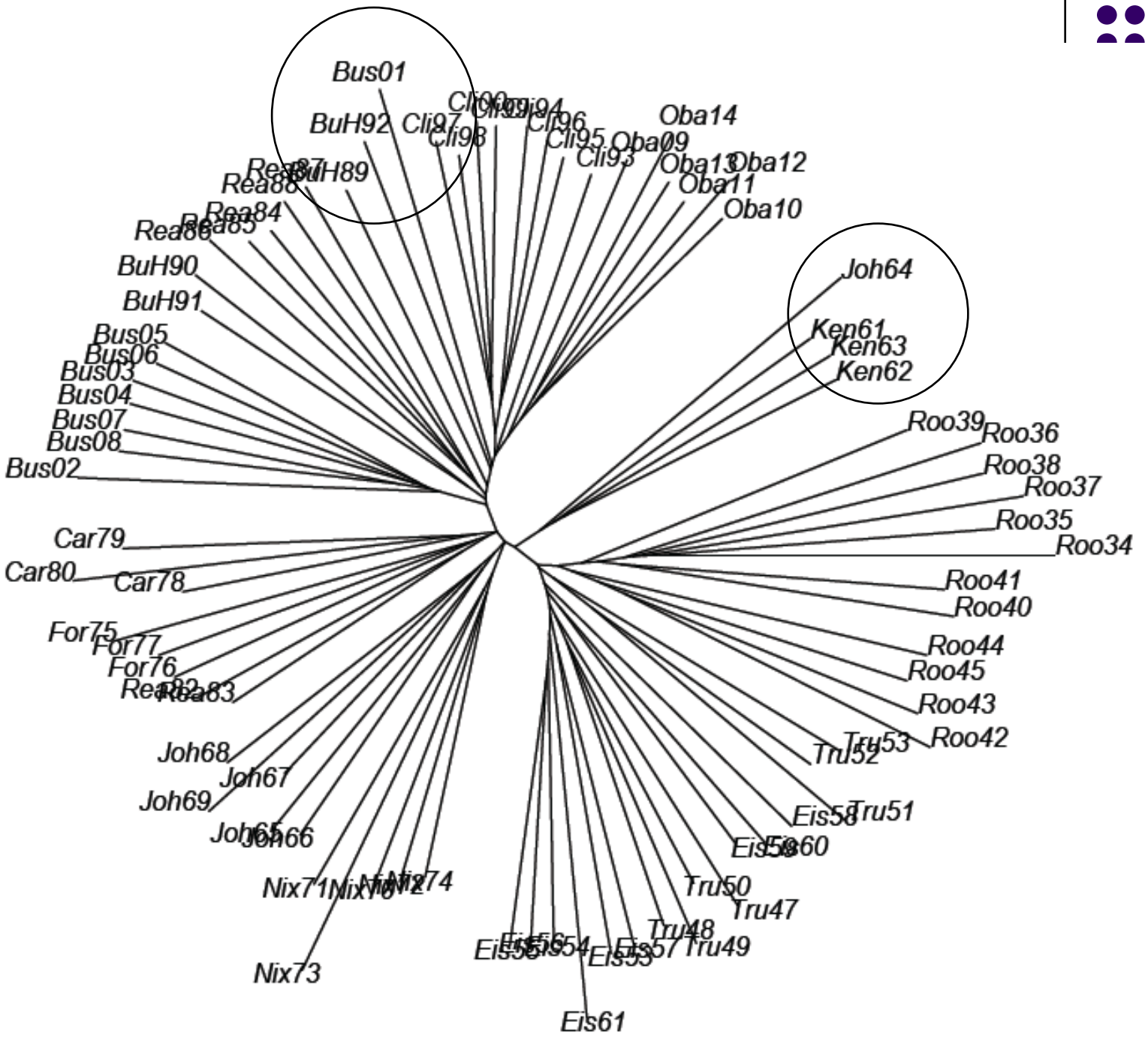
| | $D_1$ | $D_2$ | $D_3$ |
|---|---|---|---|
| **A** | 1 | 0 | 0 |
| **B** | 1 | 1 | 0 |
| **C** | 0 | 1 | 1 |
| **D** | 1 | 1 | 0 |
| **E** | 0 | 1 | 1 |

# Example: Clustering Documents

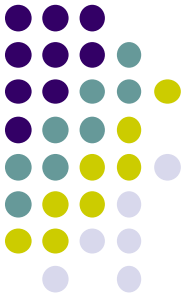


- Documents with similar sets of words may be about the same topic.
- Refine this idea by using a threshold (consider terms only if their frequency of occurrence is greater than a given threshold, remove stop-word like "the", "in", "of")

# Dista


Bus01
BuH92
Cli97
Cli98
Cli96
Cli95
Cli93
Oba14
Oba09
Oba13
Oba12
Oba11
Oba10
BuH89
Rea88
Rea84
Rea85
Rea86
BuH90
BuH91
Bus05
Bus06
Bus03
Bus04
Bus07
Bus08
Bus02
Joh64
Ken61
Ken63
Ken62
Roo39
Roo36
Roo38
Roo37
Roo35
Roo34
Roo41
Roo40
Roo44
Roo45
Roo43
Roo42
Tru53
Tru52
Tru51
Eis58
Eis60
Tru50
Tru47
Tru48
Tru49
Eis57
Eis53
Eis61
Nix73
Car79
Car80
Car78
For75
For77
For76
Rea83
Joh68
Joh67
Joh69
Joh66
Nix74

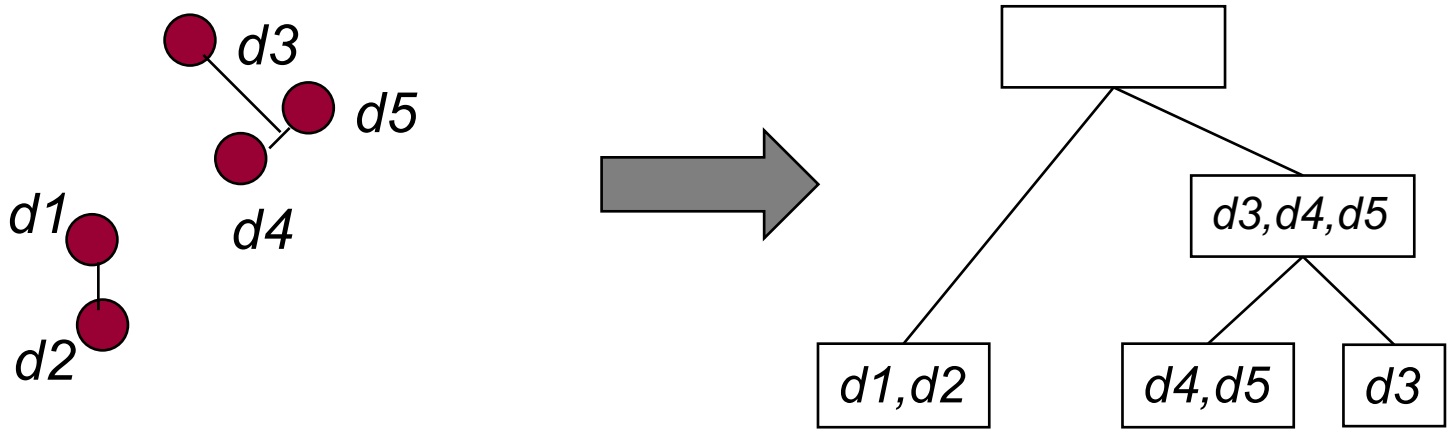
# Example: Gene Sequences

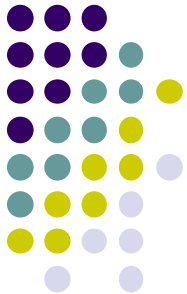


- Objects are sequences of {C,A,T,G}.
- Distance between sequences is *edit distance*, the minimum number of inserts and deletes needed to turn one into the other.
- Knowledge can be incorporated to define real differences between DNA sequences
  e.g., the sequence TAT or TAC defines the tyrosine
  the sequences TCT, TCC, TCA, TCG, AGT, AGC = serine

# The Idea

- As clusters *agglomerate*, objects likely to fall into a hierarchy of concepts.

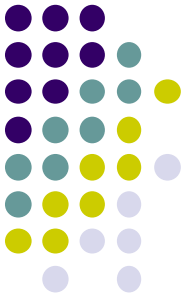




# Overview

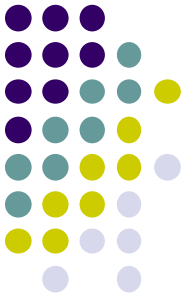
- Examples and Applications
- **Hierarchical Clustering**
- Cluster Representatives
- *k*-Means Algorithms

# What is a Good Clustering?



- Internal criterion: A good clustering will produce high quality clusters in which:
  - the intra-class (that is, intra-cluster) similarity is high
    (every member of a class is closed to all other)
  - the inter-class similarity is low
    (two members belonging to two different classes are very distant)
  - The measured quality of a clustering depends on both the object representation and the similarity measure used
- External criterion: The quality of a clustering is also measured by its ability to discover some or all of the hidden patterns or latent classes
  - Assessable with gold standard data

# External Evaluation of Cluster Quality

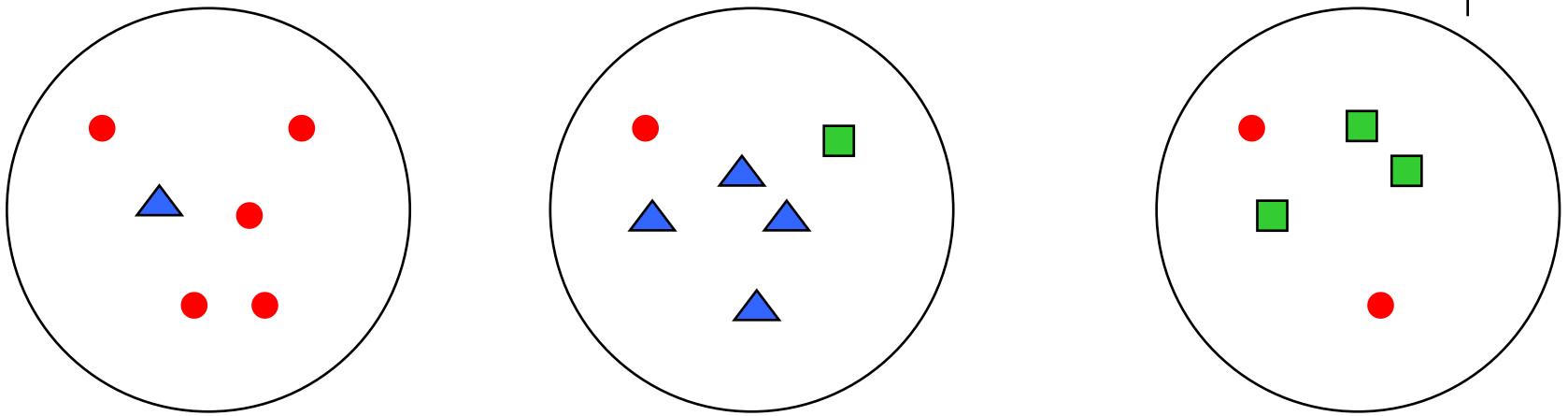


- Assesses clustering with respect to ground truth
- Assume that there are C gold standard classes, while our clustering algorithms produce *k* clusters, $\pi_1$, $\pi_2$, …, $\pi_k$ with $n_i$ members.
- Simple measure: purity, the ratio between the dominant class in the cluster $\pi_i$ and the size of cluster $\pi_i$

$$Purity(\pi_i) = \frac{1}{n_i} \cdot Max_j \left(n_{ij}\right) \quad j \in C$$

- Others are entropy of classes in clusters (or mutual information between classes and clusters)

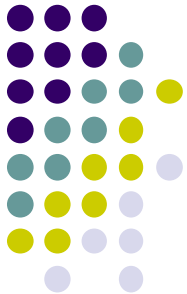
# Purity



Cluster I Cluster II Cluster III

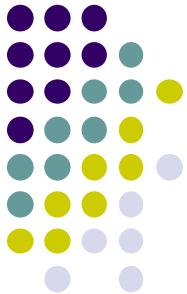
- Use the order (red point, blue triangle, green square)
- Cluster I: Purity = 1/6 (max(5, 1, 0)) = 5/6
- Cluster II: Purity = 1/6 (max(1, 4, 1)) = 4/6
- Cluster III: Purity = 1/5 (max(2, 0, 3)) = 3/5

# Methods of Clustering



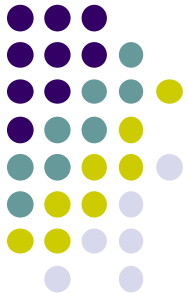
- Hierarchical algorithms (hierarchical agglomerative clustering, HAC)
  - Bottom-up, agglomerative
  - Top-down, divisive
- Partitioning "flat" algorithms
  - Usually start with a random (partial) partitioning
  - Refine it iteratively
    - *k* means/medoids clustering
    - Model based clustering

# Methods of Clustering



- Relationship between attributes and the classes
  - monothetic (each object possesses a large (all?) number of the attributes)
  - polythetic (each attribute in a class is possessed by many number of these members)
- Relationship between objects and the classes
  - disjoint vs. overlapping
  - deterministic vs. probabilistic
- Relationship between classes
  - ordered (hierarchy, e.g, Carl von Linné)
  - unordered (flat)

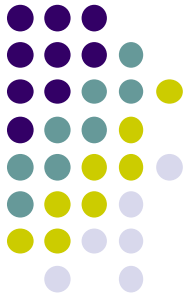
# Methods of Clustering



With 8 objects and 8 properties, we can define polythetic class $\{O_1, O_2, O_3, O_4\}$ and a monothetic class $\{O_5, O_6\}$

| | $O_1$ | $O_2$ | $O_3$ | $O_4$ | $O_5$ | $O_6$ | $O_7$ | $O_8$ |
|---|---|---|---|---|---|---|---|---|
| $p_1$ | + | + | + | | | | | |
| $p_2$ | + | + | | + | | | | |
| $p_3$ | + | | + | + | | | | |
| $p_4$ | + | + | + | | | | | |
| $p_5$ | | | | | + | + | + | |
| $p_6$ | | | | | + | + | + | |
| $p_7$ | | | | | + | + | | + |
| $p_8$ | | | | | + | + | | + |

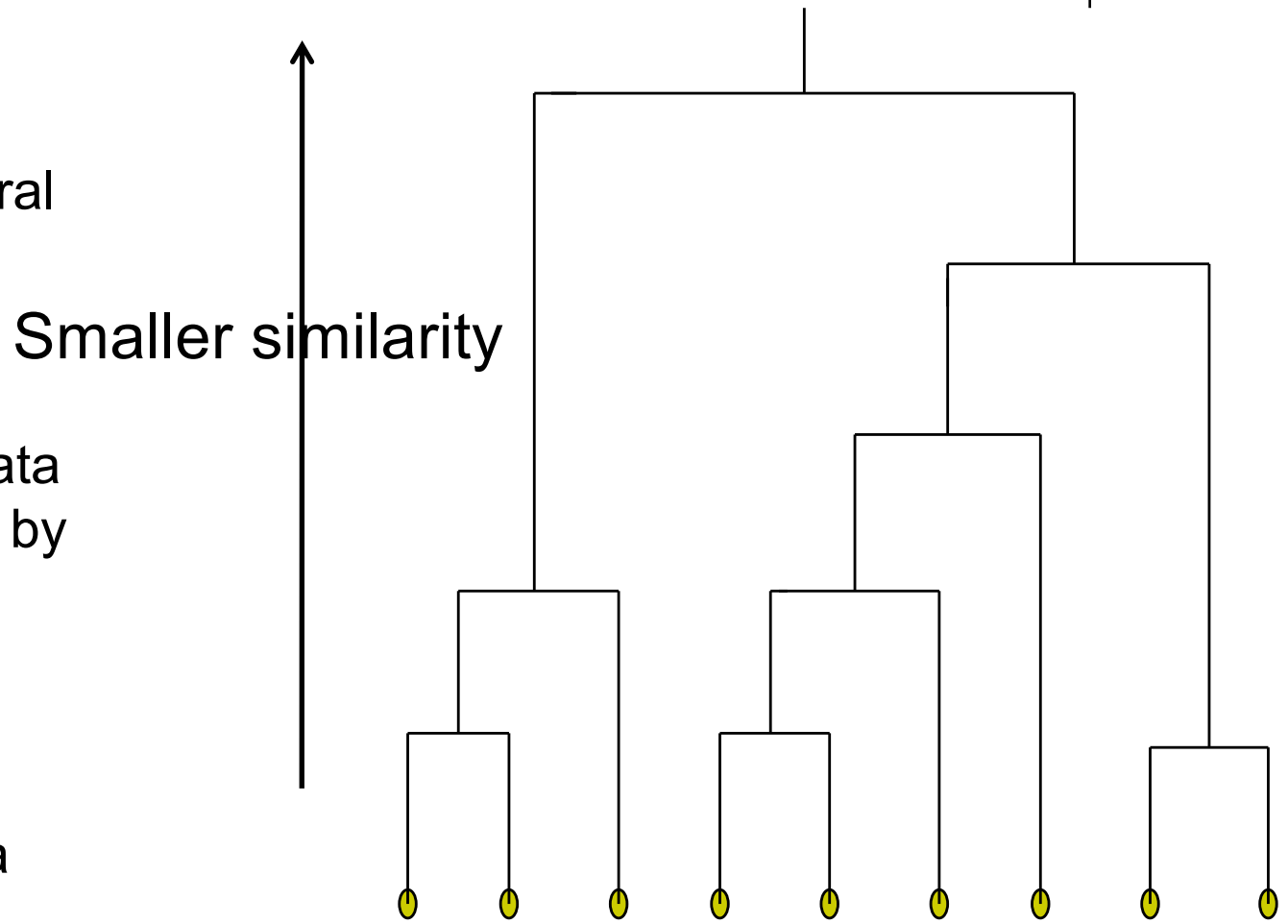
# Hierarchical Clustering

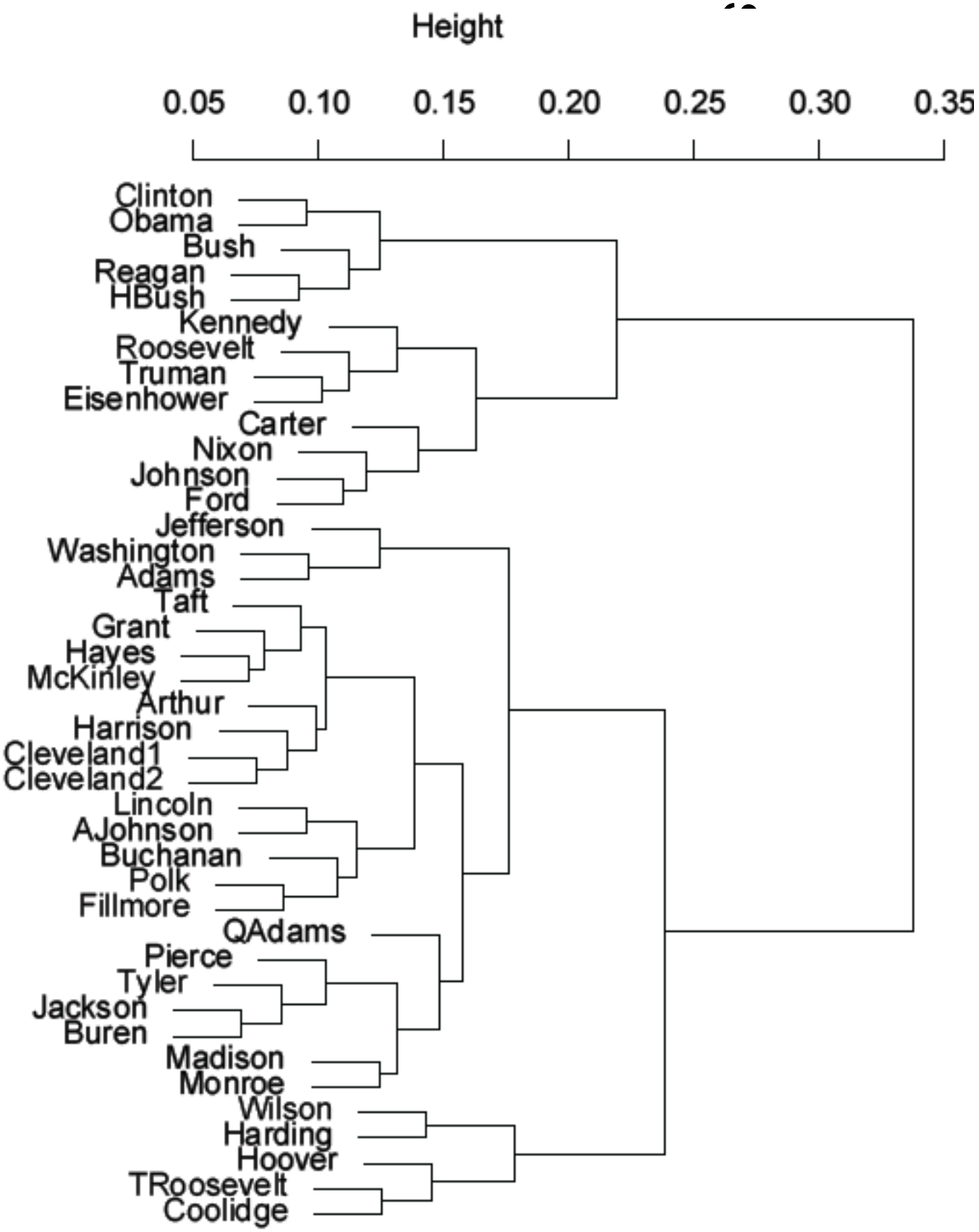


- Assumes a similarity function for determining the similarity of two instances.
- Start with all instances in their own cluster.
  Until there is only one cluster:
    Among the current clusters, determine the two clusters, $c_i$ and $c_k$, that are *most similar*. (MAX).
  Replace $c_i$ and $c_k$, with a single cluster $c_i \cup c_k$
- The history of merging forms a binary tree or hierarchy.
  The final result is presented as a dendrogram.

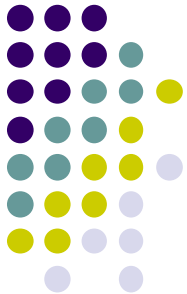
# Hierarchical Clustering

- Dendrogram: Decomposes data objects into a several levels of nested partitioning (tree of clusters).
- Clustering of the data objects is obtained by cutting the dendrogram at the desired level, then each connected component forms a cluster.



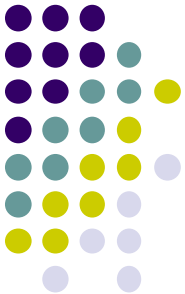

Height
0.05
0.10
0.15
0.20
0.25
0.30
0.35
Clinton
Obama
Bush
Reagan
HBush
Kennedy
Roosevelt
Truman
Eisenhower
Carter
Nixon
Johnson
Ford
Jefferson
Washington
Adams
Taft
Grant
Hayes
McKinley
Arthur
Harrison
Cleveland1
Cleveland2
Lincoln
AJohnson
Buchanan
Polk
Fillmore
QAdams
Pierce
Tyler
Jackson
Buren
Madison
Monroe
Wilson
Harding
Hoover
TRoosevelt
Coolidge

# Hierarchical Clustering



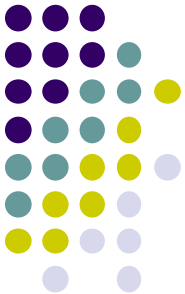
- Two important questions:
  1. How do you determine the “nearness” of clusters?
  2. How do you represent a cluster of more than one point?

# Hierarchical Clustering



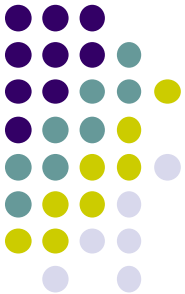
- Key problem: as you build clusters, how do you represent the location of each cluster, to tell which pair of clusters is closest?
- Euclidean case: each cluster has a *centroid* = average of its points.
  - Measure intercluster distances by distances of centroids.

# “Closest pair” of clusters



- Many variants to defining closest pair of clusters
- “Center of gravity”
  - Clusters whose centroids (centers of gravity) are the most similar
- Average-link (Ward)
  - Average similarity between pairs of elements
- Single-link
  - Similarity of the most similar pair of elements (single-link)
- Complete-link
  - Similarity of the “furthest” points, the least similar

# Single Link Agglomerative Clustering



- Between two groups, use maximum similarity of pairs:

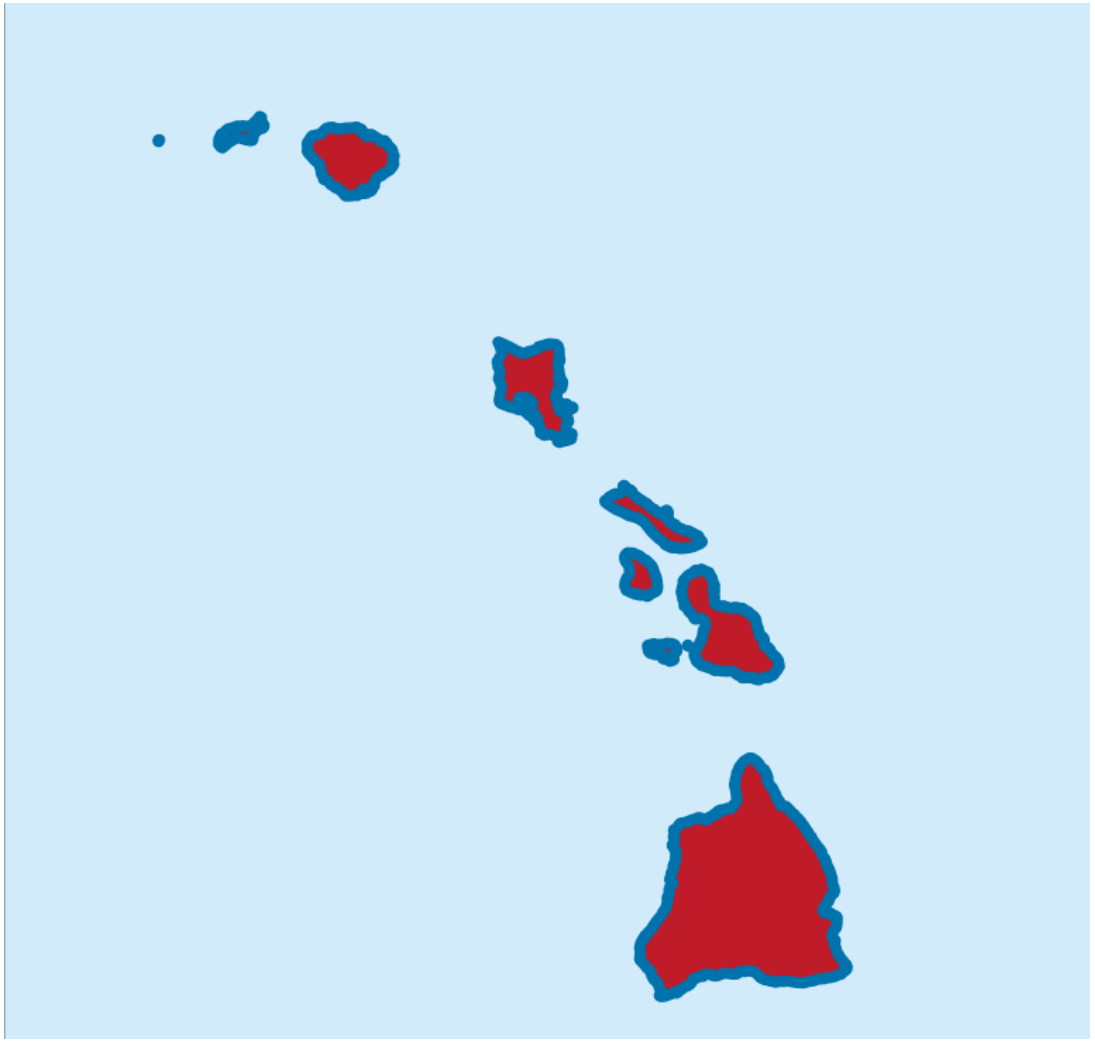
$$sim(c_i, c_j) = \max_{x \in c_i, y \in c_j} sim(x, y)$$

- Can result in "straggly" (long and thin) clusters due to chaining effect.
  - Appropriate in some domains, such as clustering islands: "Hawaii's clusters"
- After merging $c_i$ and $c_j$, the similarity of the resulting cluster to another cluster, $c_k$, is:

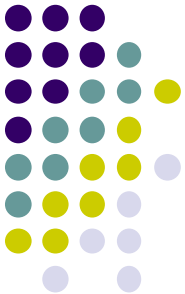
$$sim(c_i \cup c_j, c_k) = max(sim(c_i, c_k), sim(c_j, c_k))$$

# Single Link Agglomerative Clustering

Hawaii's cluster



# Complete Link Agglomerative Clustering



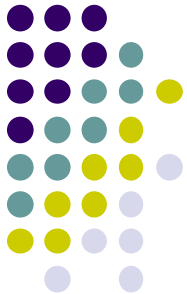
- Between two groups, use minimum similarity of pairs:

$$sim(c_i, c_j) = \min_{x \in c_i, y \in c_j} sim(x, y)$$

- Makes "tighter," spherical clusters that are typically preferable.
- After merging $c_i$ and $c_j$, the similarity of the resulting cluster to another cluster, $c_k$, is:

$$sim(c_i \cup c_j, c_k) = min(sim(c_i, c_k), sim(c_j, c_k))$$

# Example



- Similarity matrix
- The most similar pair of object is A & F
- The new similarity matrix is … depends if you're using the single or complete link…
- Use the single link

| | A | B | C | D | E | F |
|---|---|---|---|---|---|---|
| **A** | - | 0.3 | 0.5 | 0.6 | 0.8 | 0.9 |
| **B** | 0.3 | - | 0.4 | 0.5 | 0.7 | 0.8 |
| **C** | 0.5 | 0.4 | - | 0.3 | 0.5 | 0.2 |
| **D** | 0.6 | 0.5 | 0.3 | - | 0.4 | 0.1 |
| **E** | 0.8 | 0.7 | 0.5 | 0.4 | - | 0.3 |
| **F** | 0.9 | 0.8 | 0.2 | 0.1 | 0.3 | - |

# Example

The new similarity matrix depend of how we compute the similarity between classes of objects, classes having more than one element.

| | A,F | B | C | D | E |
|---|---|---|---|---|---|
| **A,F** | | 0.8 | 0.5 | 0.6 | 0.8 |
| **B** | | | 0.4 | 0.5 | 0.7 |
| **C** | | | | 0.3 | 0.5 |
| **D** | | | | | 0.4 |
| **E** | | | | | |

| | A,F,E | B | C | D |
|---|---|---|---|---|
| **A,F,E** | | 0.8 | 0.5 | 0.6 |
| **B** | | | 0.4 | 0.5 |
| **C** | | | | 0.3 |
| **D** | | | | |

With the single link (max), we have the following matrices

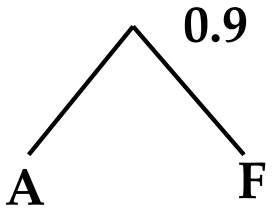
# Example

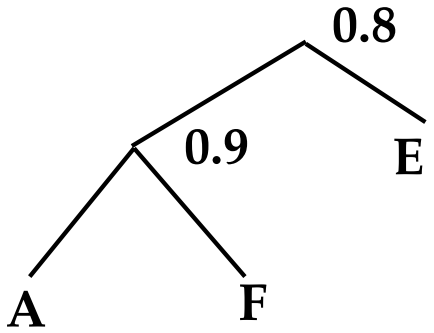
The new similarity matrix

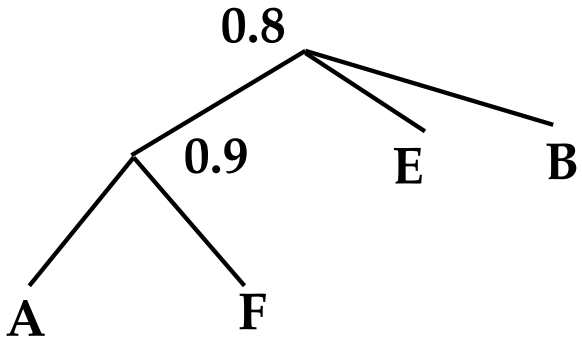
| | A,F,E,B | C | D |
|---|---|---|---|
| **A,F,E,B** | | 0.5 | 0.6 |
| **C** | | | 0.3 |
| **D** | | | |

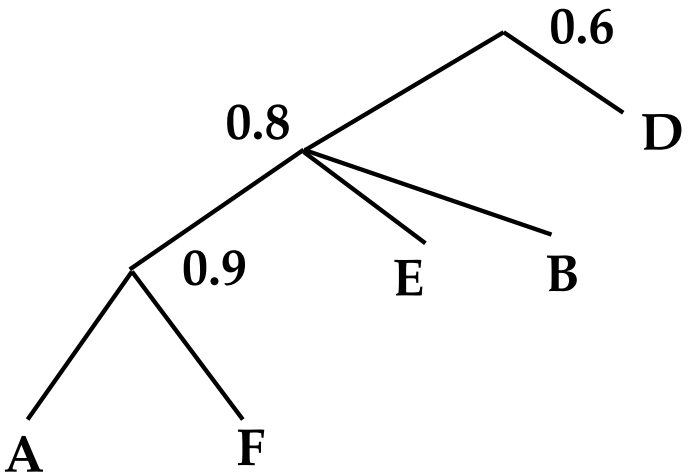
# Example

The result

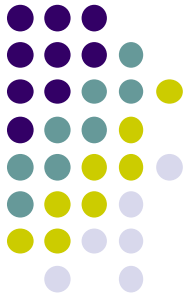


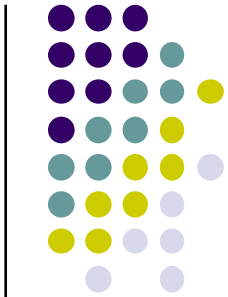



# Computational Complexity



- In the first iteration, all hierarchical clustering methods need to compute similarity of all pairs of $n$ individual instances which is $O(n^2)$.
- In each of the subsequent $n-2$ merging iterations, it must compute the distance between the most recently created cluster and all other existing clusters.
  - Since we can just store unchanged similarities
- In order to maintain an overall $O(n^2)$ performance, computing similarity to each other cluster must be done in constant time.
  - Else $O(n^2 \log n)$ or $O(n^3)$ if done naively

# Use R to compute



With a simple example

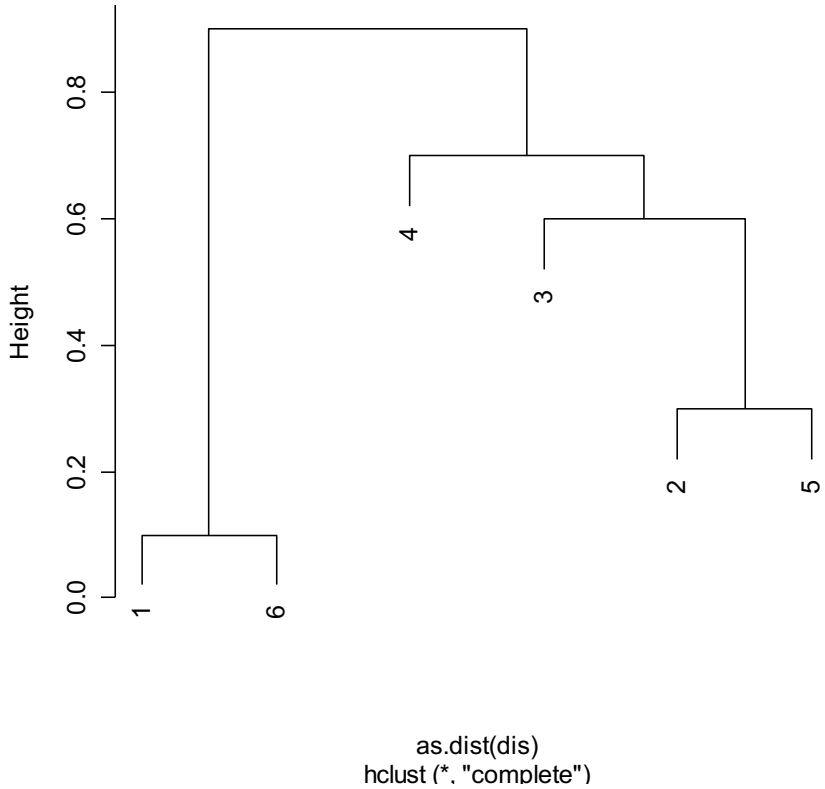
Building the distance matrix

```
l1 <- c(1,0.3,0.5,0.6,0.8,0.9)
l2 <- c(0.3,1,0.4,0.5,0.7,0.8)
l3 <- c(0.5,0.4,1, 0.3,0.5,0.2)
l4 <- c(0.6,0.5,0.3,1,0.4,0.1)
l5 <- c(0.8,0.7,0.5,0.4,1,0.3)
l6 <- c(0.9,0.8,0.2,0.1,0.3,1)
dat <- c(l1,l2,l3,l4,l5,l6)
sim <- matrix(dat, nrow=6)
c("A", "B", "C", "D", "E", "F")
dis <- 1 - sim
```

# Use R to compute

With a simple example

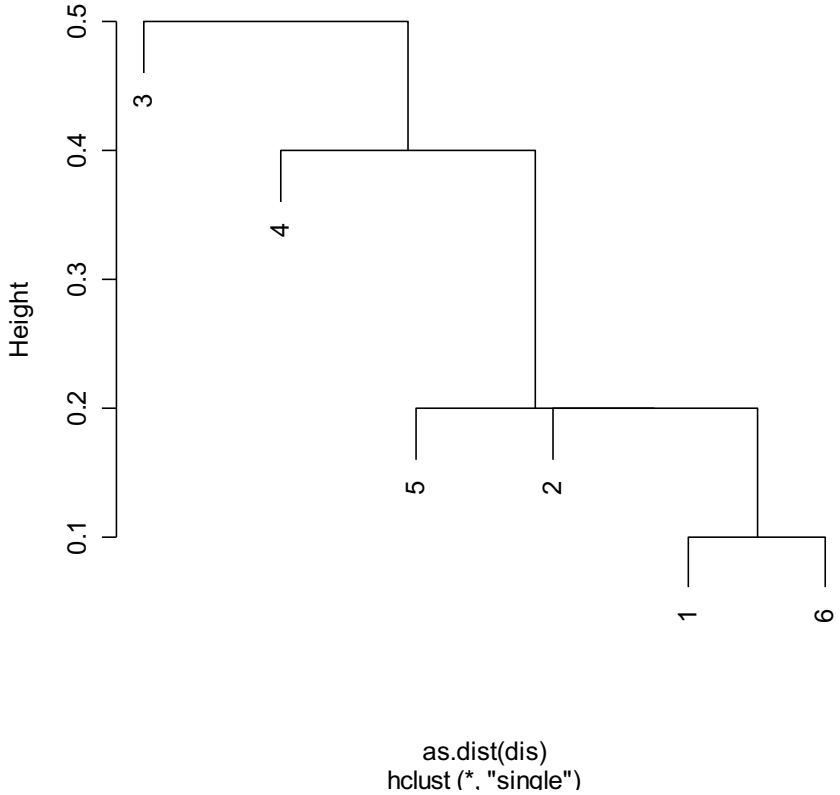
```
> as.dist(dis)
    1   2   3   4   5
2 0.7
3 0.5 0.6
4 0.4 0.5 0.7
5 0.2 0.3 0.5 0.6
6 0.1 0.2 0.8 0.9 0.7
> h <- hclust(as.dist(dis),
  method="complete")
> plclust(h)
```



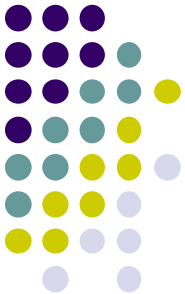
# Use R to compute

With the same example, another distance measure between clusters …

```
> as.dist(dis)
    1   2   3   4   5
2 0.7
3 0.5 0.6
4 0.4 0.5 0.7
5 0.2 0.3 0.5 0.6
6 0.1 0.2 0.8 0.9 0.7
> h <- hclust(as.dist(dis), method="single")
> plclust(h)
```



# Use R to compute



- Function `hclust(d,method="complete")`
  - `method="single"`
    `"complete"`
    `"average"`
    `"centroid"`
- Function `dist(x,method="euclidian",p=2)` returns a distance matrix
  - `method="euclidian"`
    `"manhanttan"`
    `"maximum"`
    `"canberra"`
    `"minkowski"`
- A library (`cluster`) is also available (use with Kaufman & Rousseeuw (1990) book).

# Distance Definition

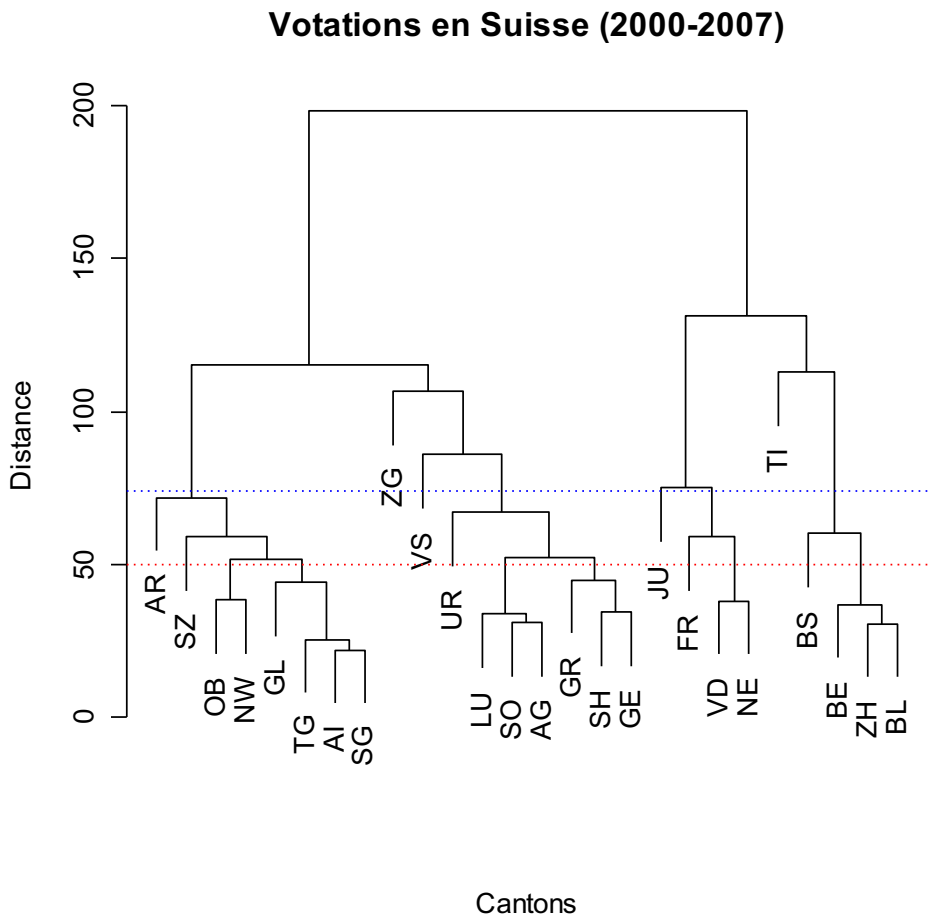
$$Dist_{Minkowski}(X,Y) = \sqrt[p]{\sum_{i=1}^{n} |x_i - y_i|^p}$$

With *p* = 1, Minkowski = Manhattan
With *p* = 2, Minkowski = Euclidian
With p = ∞, Minkowski = maximum

$$Dist_{Canberra}(X,Y) = \sum_{i=1}^{n} \frac{|x_i - y_i|}{|x_i| + |y_i|}$$

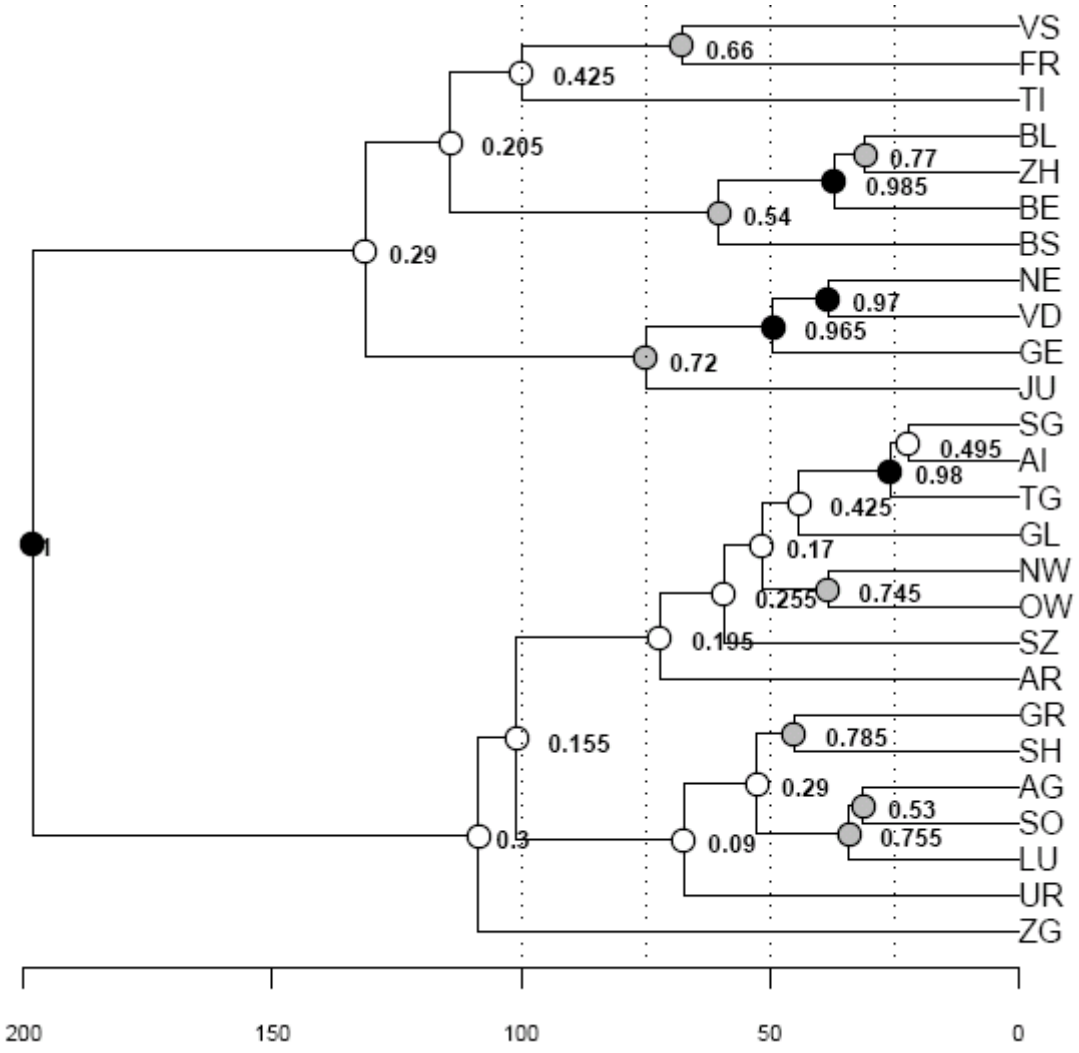
# Use R to compute

```
# Votation in CH
> h <-
hclust(dist(t(c)),
method="complete")

> plclust(h,
ylab="Distance",
xlab="Cantons",
main="Votations en
Suisse (2000-2007)")

> abline(h=50,lty=3,
col="red")
> abline(h=74,lty=3,
col="blue")
```



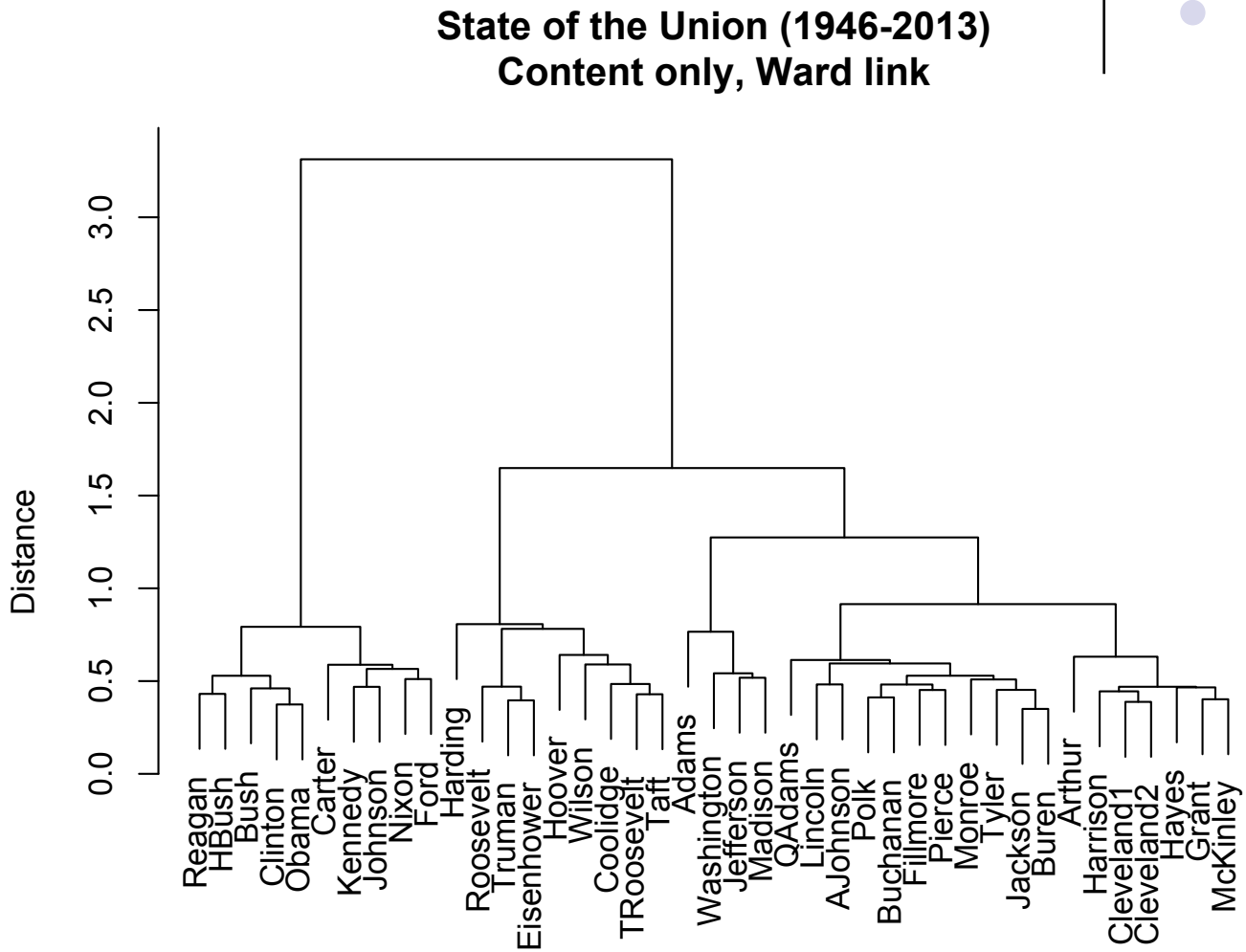
# Example with the Swiss vote

Complete link
73 votes
(2000-2007)
& Bootstrap



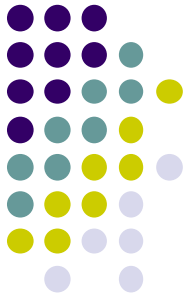
# State of the Union Adresses

Complete link
223 speeches
41 US presidents





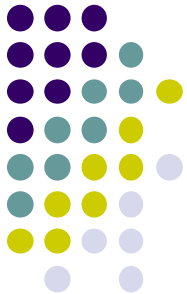

as.dist(data)
hclust (*, "ward")



# Overview

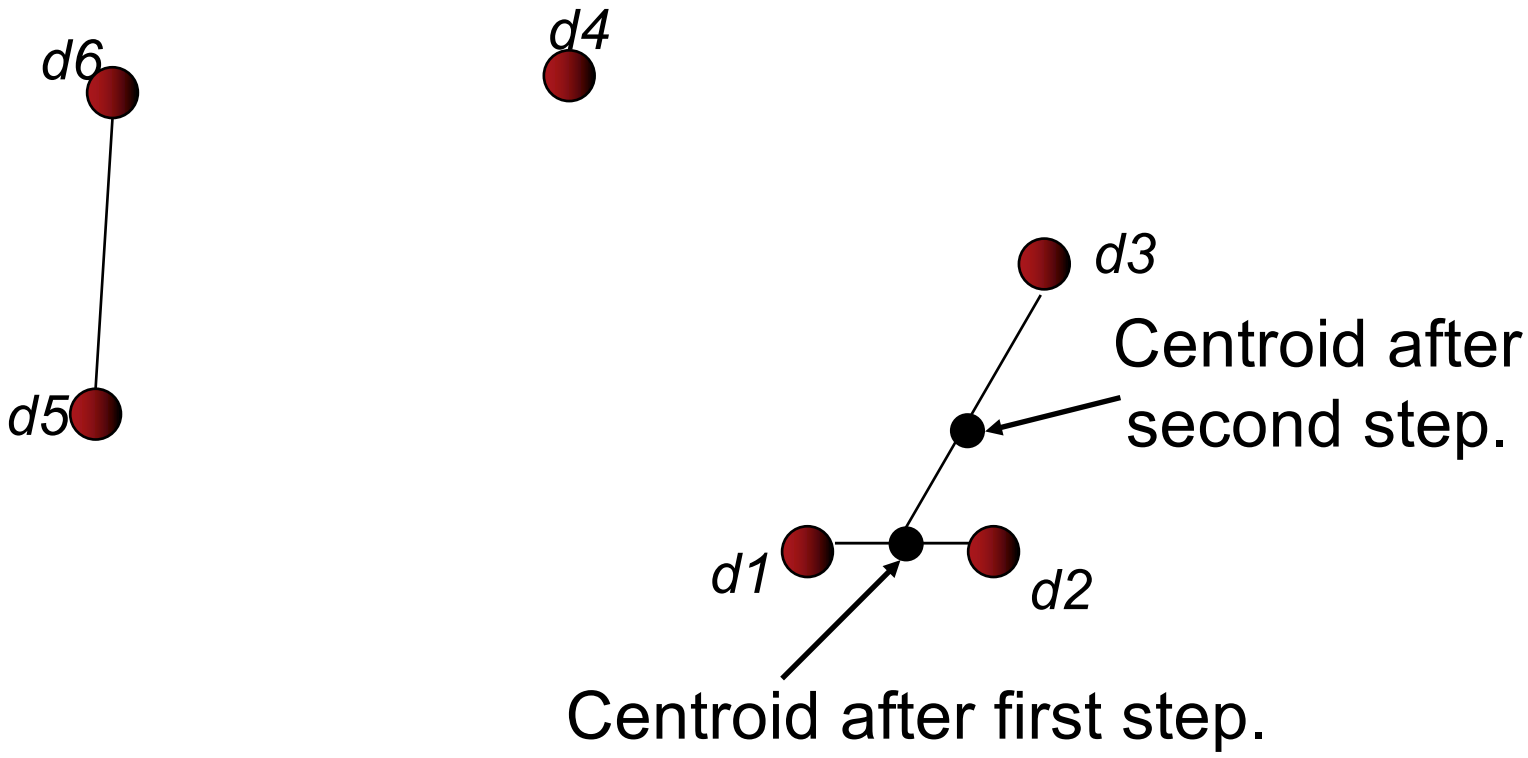
- Examples and Applications
- Hierarchical Clustering
- **Cluster Representatives**
- *k*-Means Algorithms

# Key notion: cluster representative

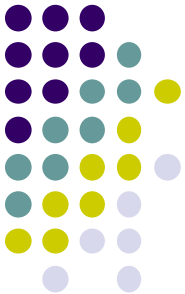


- We want a notion of a representative point in a cluster
- Representative should be some sort of "typical" or central point in the cluster, e.g.,
  - point inducing smallest radii to objects in cluster
  - smallest squared distances, etc.
  - point that is the "average" of all objects in the cluster
    - Centroid or center of gravity

# Example: *n*=6, *k*=3, closest pair of centroids

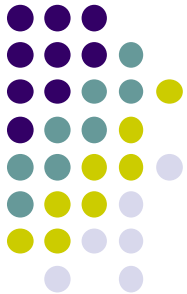


# Outliers in centroid computation



- Can ignore outliers when computing centroid.
- What is an outlier?
  - Lots of statistical definitions, e.g. four times the standard deviation.

# Group Average Agglomerative Clustering

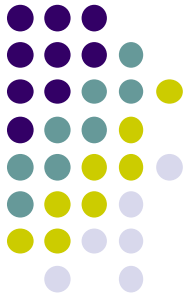


- Use average similarity across all pairs within the merged cluster to measure the similarity of two clusters.

$$sim(c_i, c_j) = \frac{1}{|c_i \cup c_j| \cdot (|c_i \cup c_j| - 1)} \cdot \sum_{\vec{x} \in (c_i \cup c_j)} \sum_{\vec{y} \in (c_i \cup c_j) : \vec{x} \neq \vec{y}} sim(\vec{x}, \vec{y})$$

- Compromise between single and complete link.

# Computing Group Average Similarity



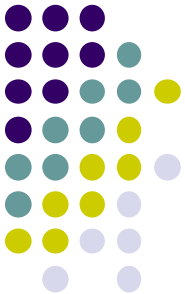
- Assume cosine similarity and normalized vectors with unit length.
- Always maintain sum of vectors in each cluster.

$$\vec{s}(c_j) = \sum_{\vec{x} \in c_j} \vec{x}$$

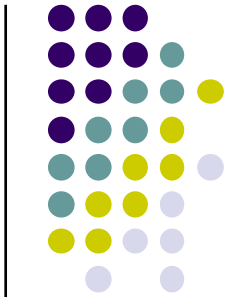
- Compute similarity of clusters in constant time:

$$sim(c_i, c_j) = \frac{(\vec{s}(c_i) + \vec{s}(c_j)) \cdot (\vec{s}(c_i) + \vec{s}(c_j)) - (|c_i| + |c_j|)}{(|c_i| + |c_j|) \cdot (|c_i| + |c_j| - 1)}$$

# Efficiency: Medoid as Cluster Representative



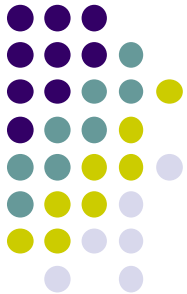
- The centroid does not have to be a document.
- Medoid: A cluster representative that is *one* of the documents
- For example: the document closest to the centroid
- One reason this is useful
  - Consider the representative of a large cluster (> 1,000 documents)
  - The centroid of this cluster will be a dense vector
  - The medoid of this cluster will be a sparse vector
- Compare: mean/centroid vs. median/medoid



# Overview

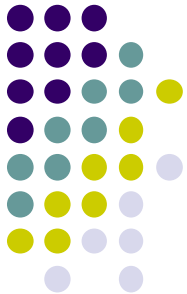
- Examples and Applications
- Hierarchical Clustering
- Cluster Representatives
- ***k*-Means Algorithms**

# *k* –Means Algorithm



- Well known approach to define clusters
- Assumes Euclidean space
- Clusters will be disjoint, deterministic and flat
- *k*-Means Algorithm
- You must specify *k*, the number of clusters (*k* predefined, sometimes difficult to specify)

# *k* –Means Algorithm

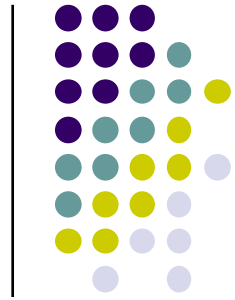


Algorithm

1. Initialize clusters by picking one point per cluster
   - For instance, pick one point at random, then *k*-1 other points, each as far away as possible from the previous points.
2. For each instance, place it in the cluster whose current centroid it is nearest.
3. After all points are assigned, compute the centroids of the *k* clusters.
4. Optional: reassign all points to their closest centroid.
   - Sometimes moves points between clusters.

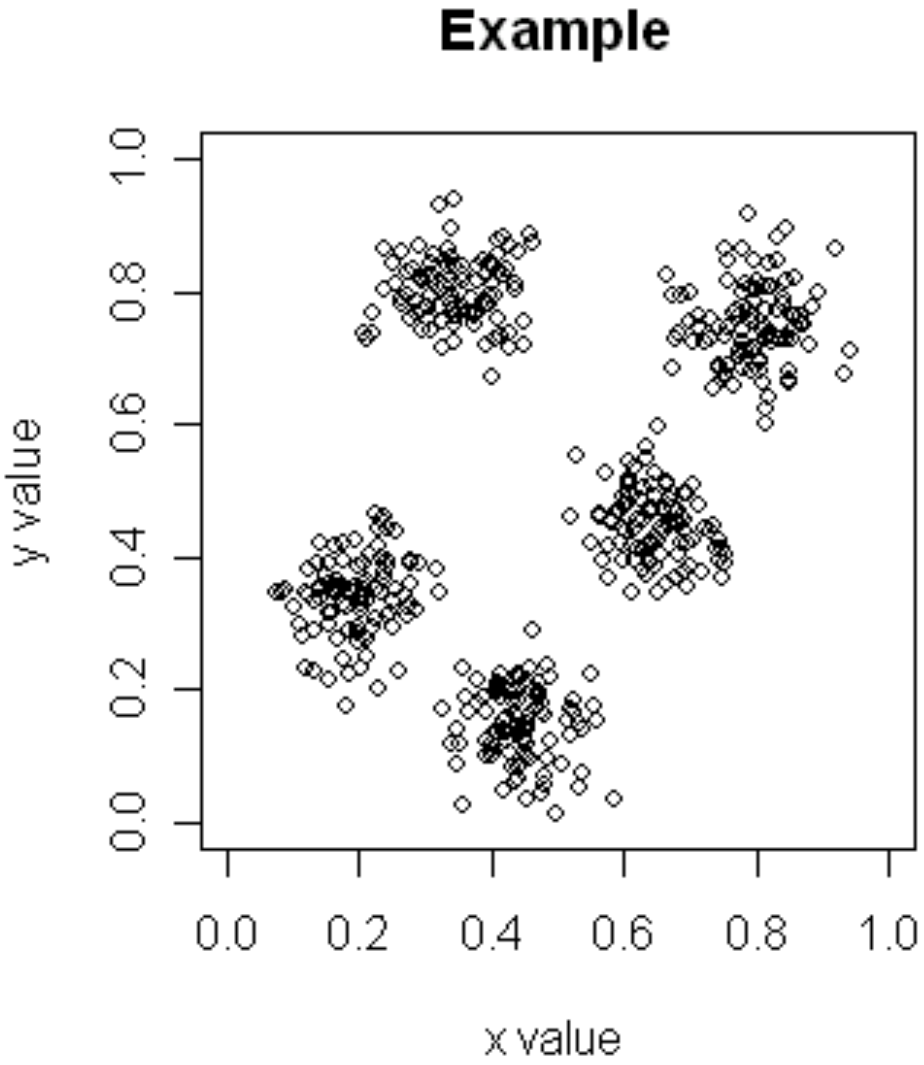
- Or go to 2 until convergence

# *k* –Means Algorithm

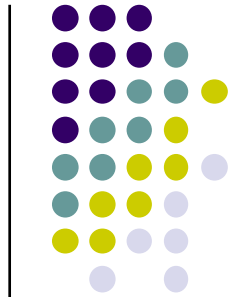


Some data

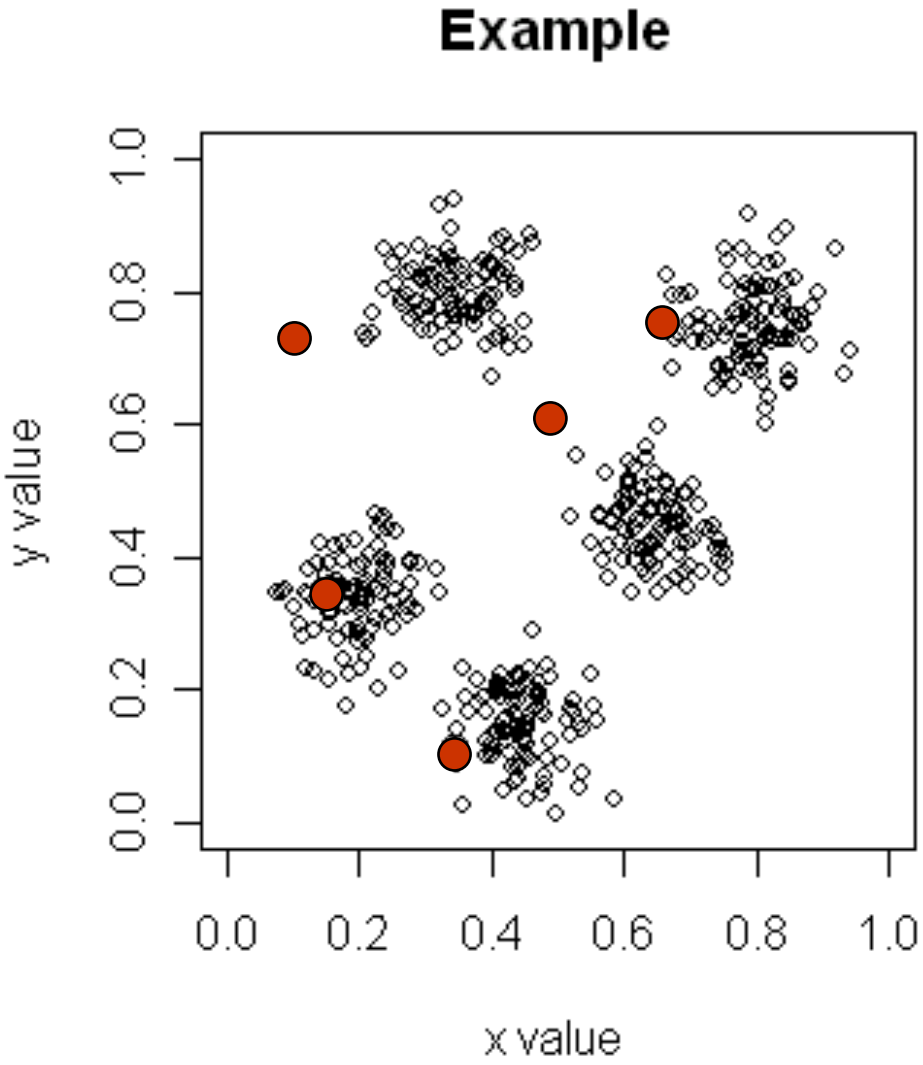
Could easily be a modeled by a Gaussian mixture (with 5 components)



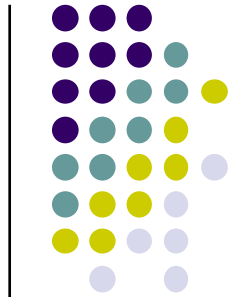
# *k* –Means Algorithm



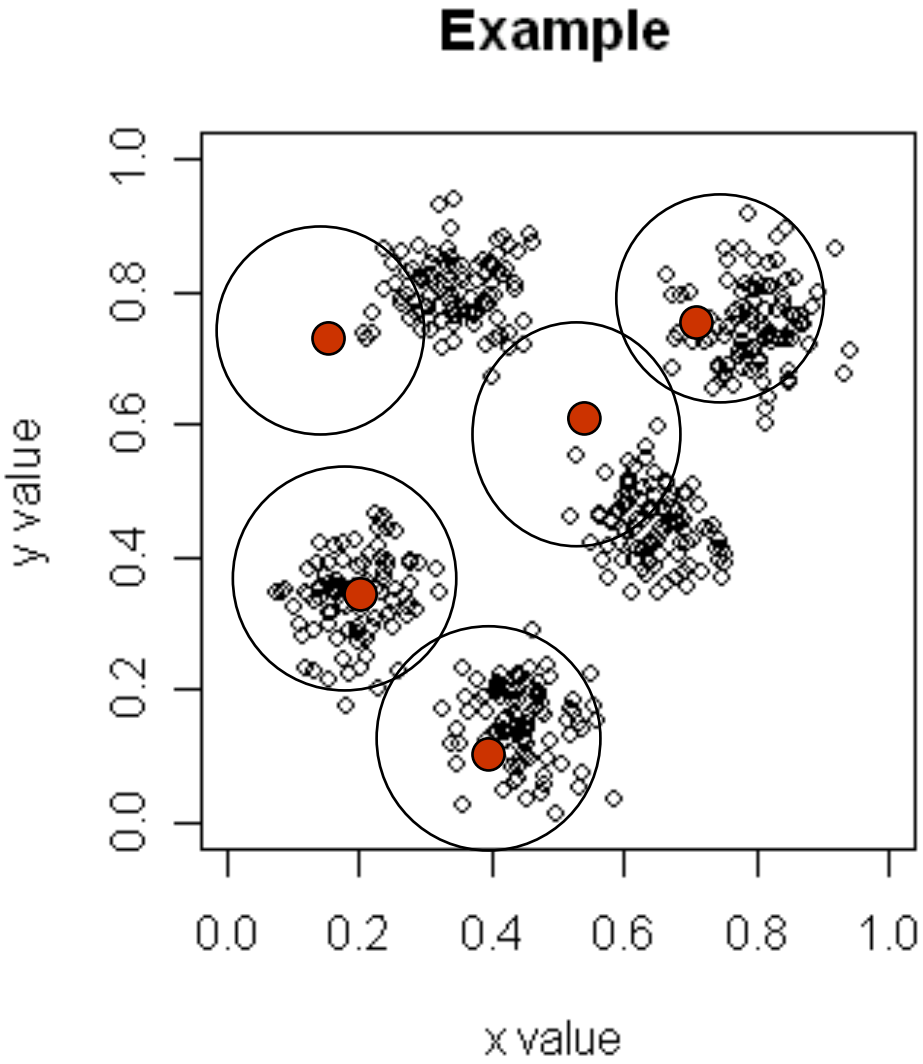
1. Ask the user how many clusters they'd like (e.g., *k*=5)
2. Randomly guess *k* cluster center locations



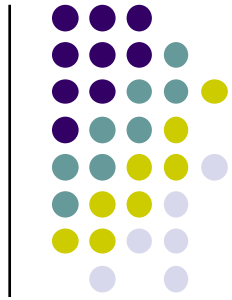
# *k* –Means Algorithm



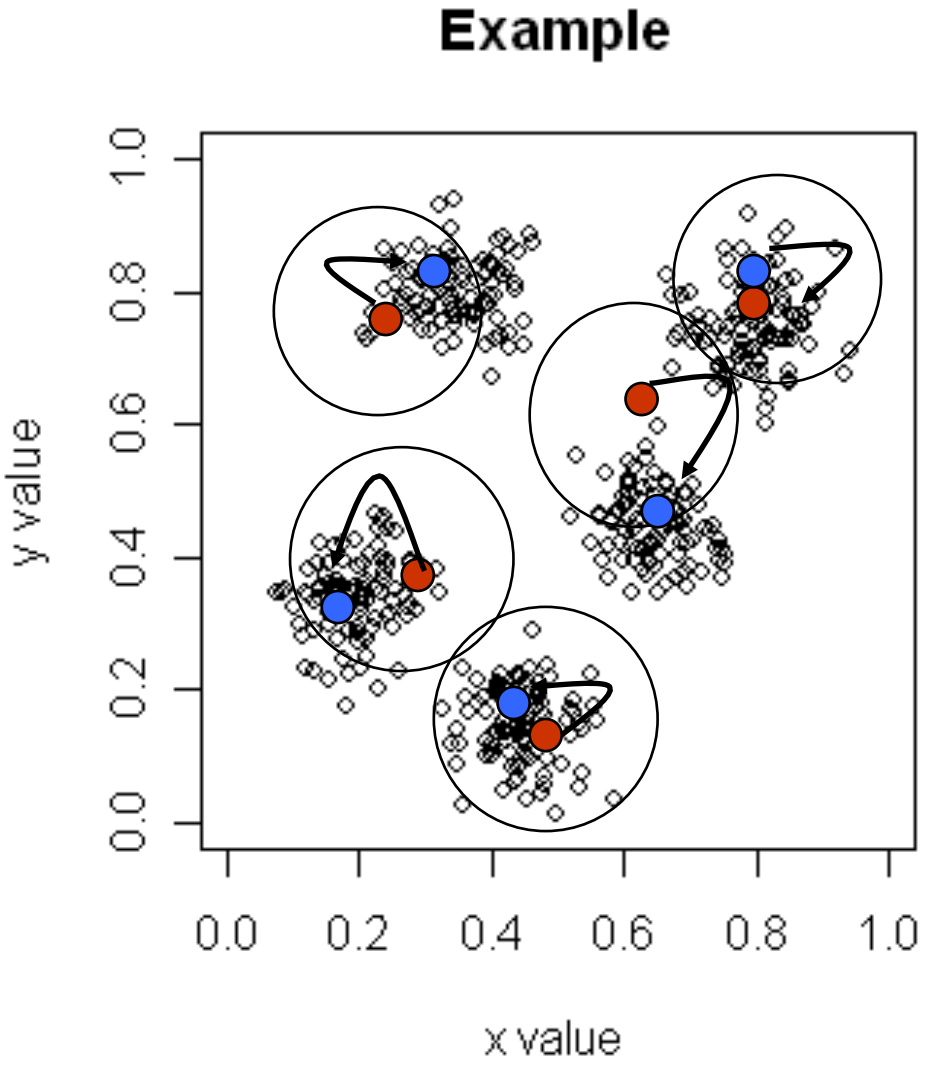
1. Ask the user how many clusters they'd like (e.g., *k*=5)
2. Randomly guess *k* cluster center locations
3. Each datapoint finds out which center it's closets to



# *k* –Means Algorithm

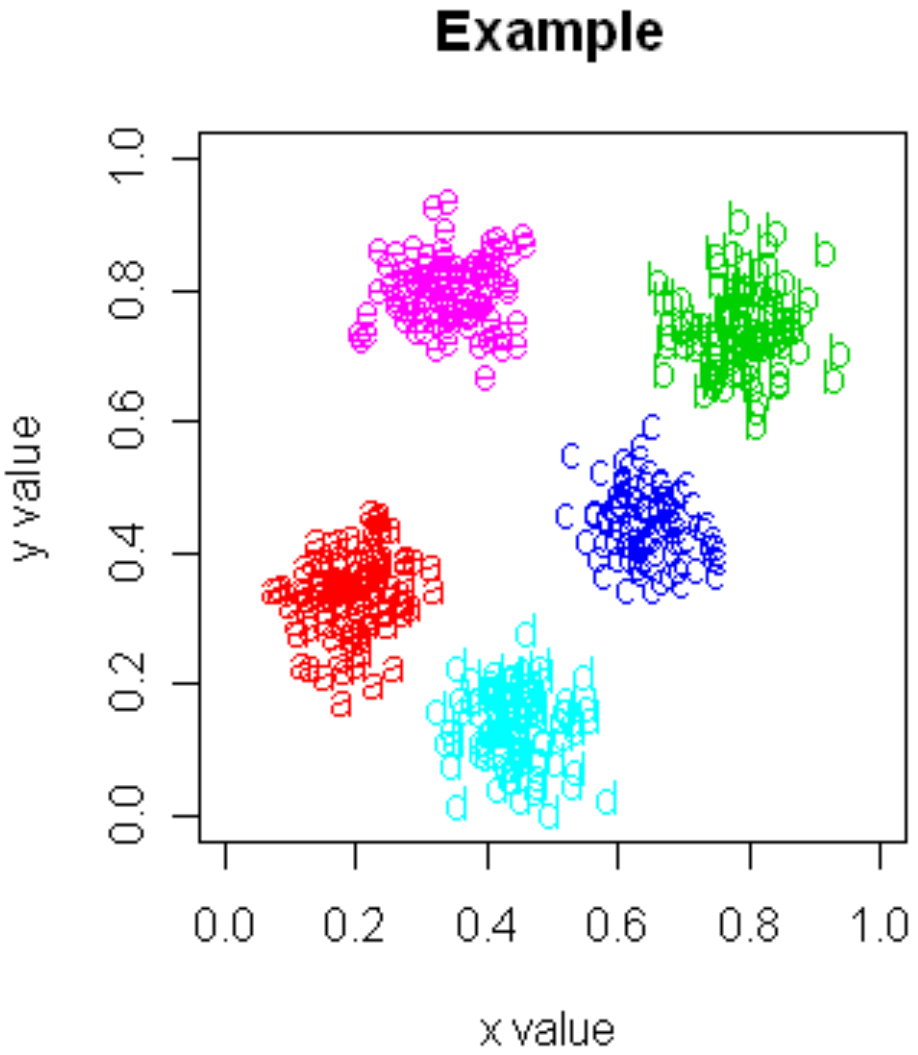


1. Ask the user how many clusters they'd like (e.g., *k*=5)
2. Randomly guess *k* cluster center locations
3. Each datapoint finds out which center it's closets to.
4. Each center finds the centroid of the points it owns
5. … and jumps there

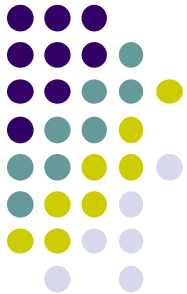


# *k* –Means Algorithm

The real underlying distribution for this data set

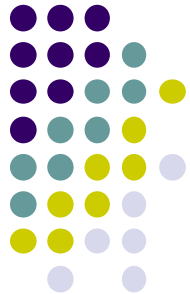


# *k* –Means Algorithm

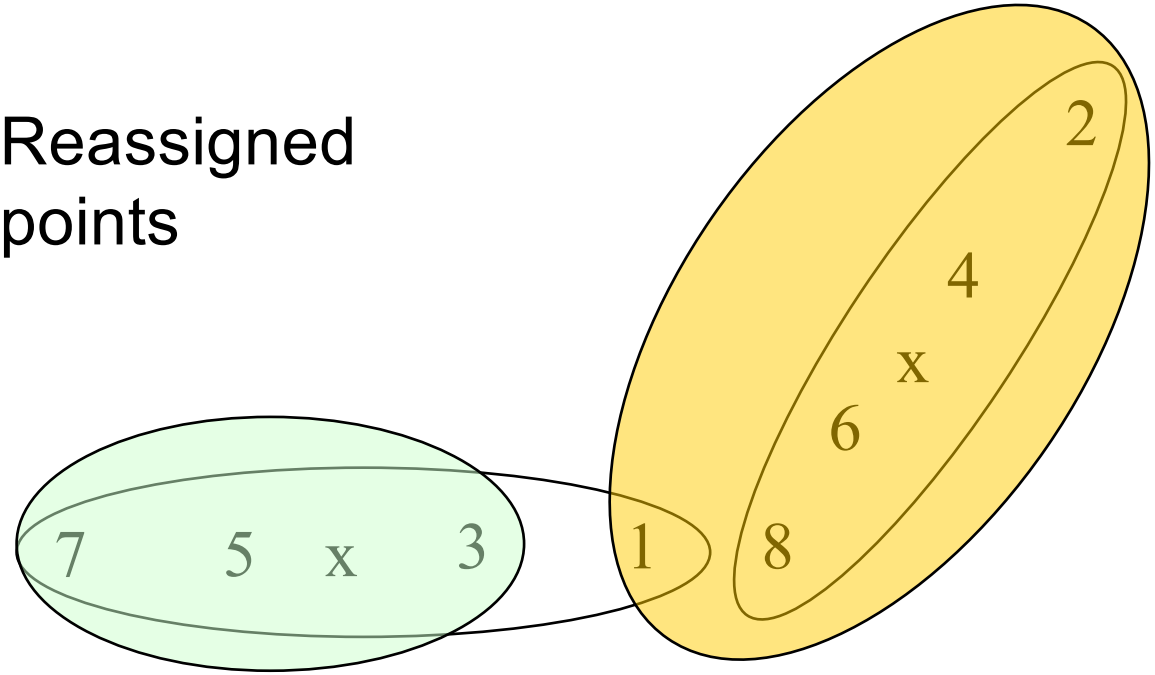


- Algorithm minimizes squared distance to cluster centers
- Result can vary significantly
  - based on initial choice of seeds
- Can get trapped in local minimum
- To increase chance of finding global optimum: restart with different random seeds
  and select the clustering having the smallest total squared distance
- Can we applied recursively with $k = 2$

# Example

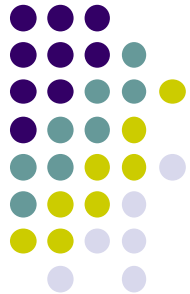


Reassigned points

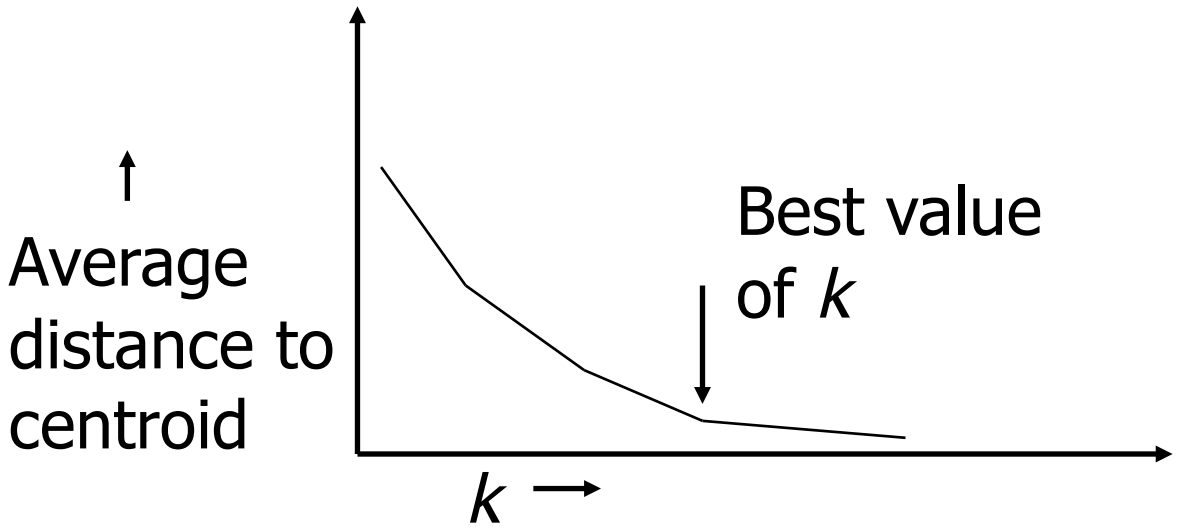



Clusters after first round

# Getting *k* Right

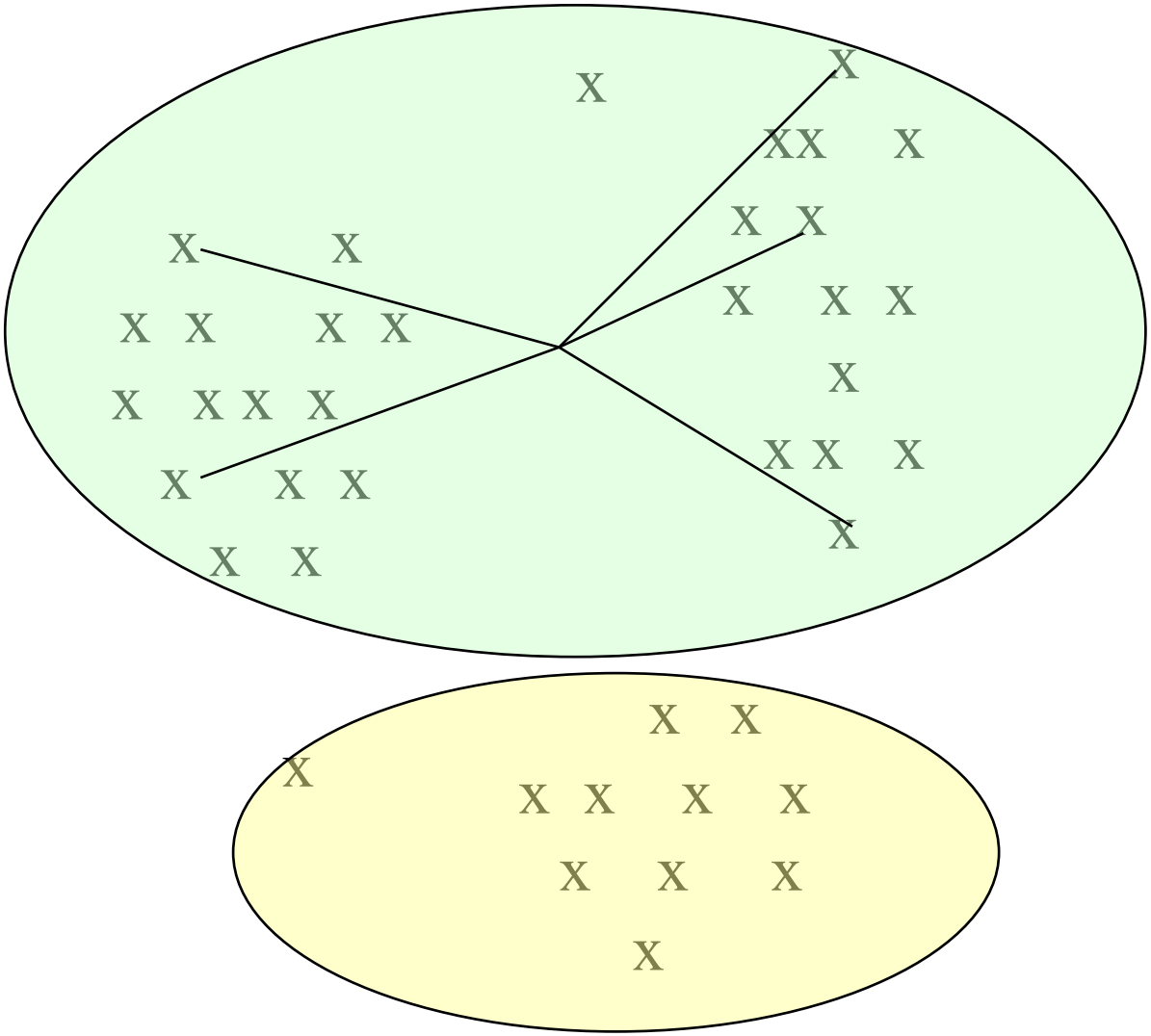


- Try different *k*, looking at the change in the average distance to centroid, as *k* increases.
- Average falls rapidly until right *k*, then changes little.



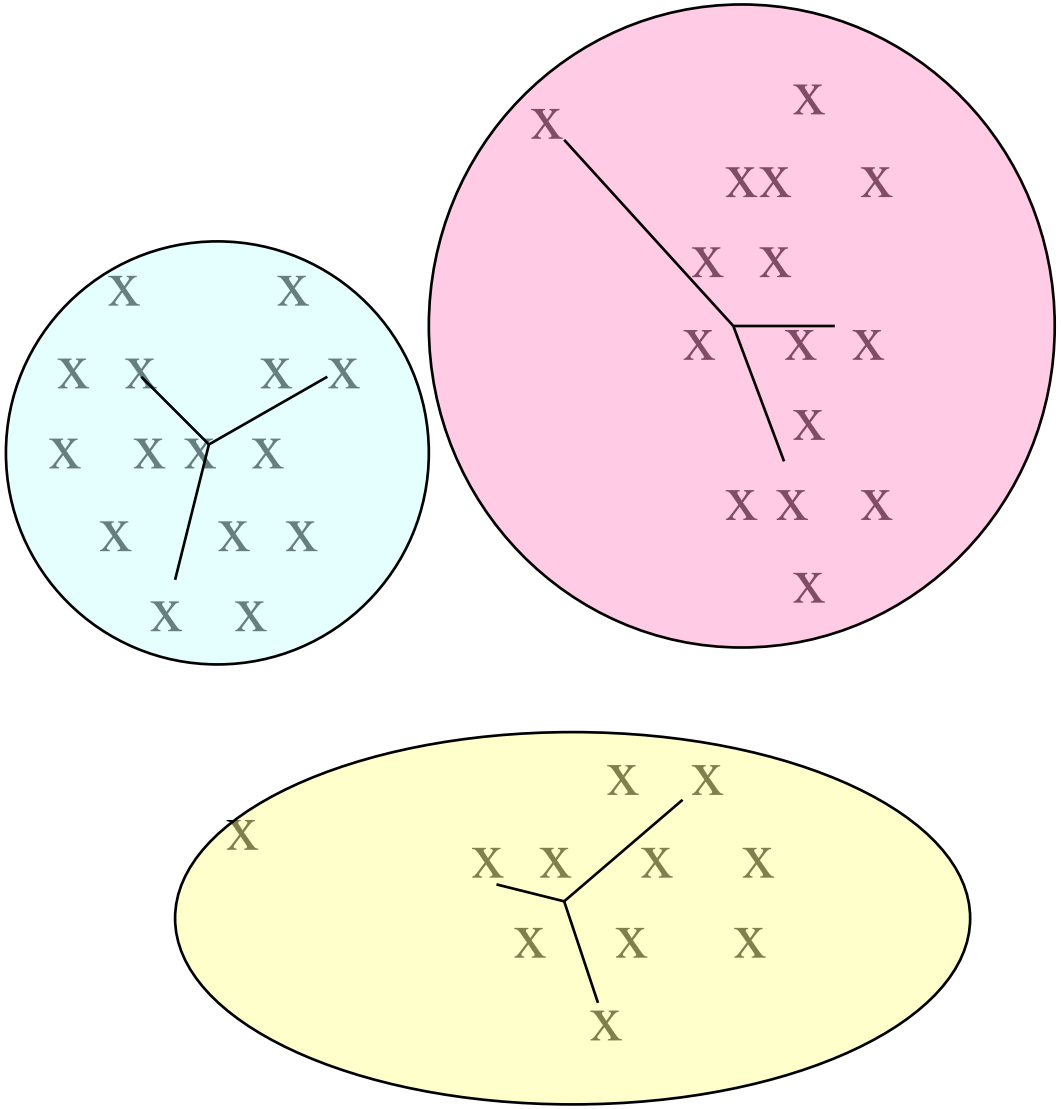
# Example

Too few;
many long
distances
to centroid.



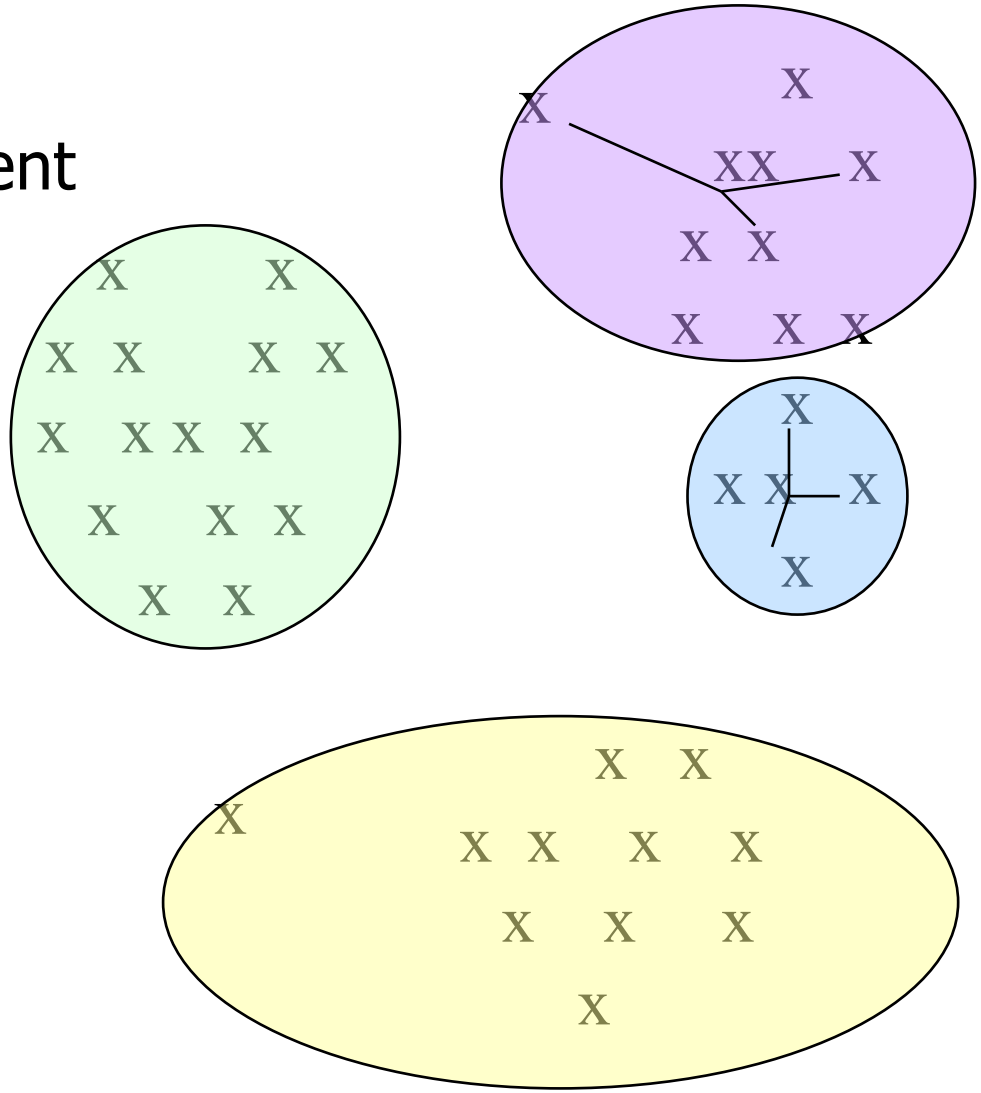
# Example

Just right; distances rather short.

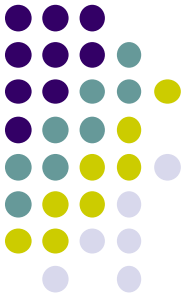


# Example

Too many;
little improvement
in average
distance.



# *k* –Means Algorithm

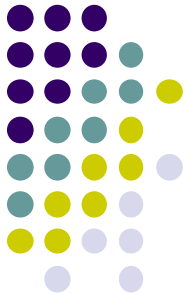


- With the *k*-means algorithm

  You need at each step to compute the distance between the *k* centroids and every instance to determine its cluster.

- But finding the closest cluster is not really different than finding the closest neighbor …

# Conclusion



- The most known unsupervised approach
  - No teaching signal, no “good” answer
- Notion of distance / similarity
  - Different measures
  - Combining different types (integer, nominal data, text)
- Notion of distance between groups
  - Single, complete, average
- Stable vs. heuristic
  - Hierarchical methods
  - K-means