

CS4220: Knowledge Discovery Methods for Bioinformatics
Unit 1: Essence of Knowledge Discovery
(Part C: Data Mining)

Wong Limsoon



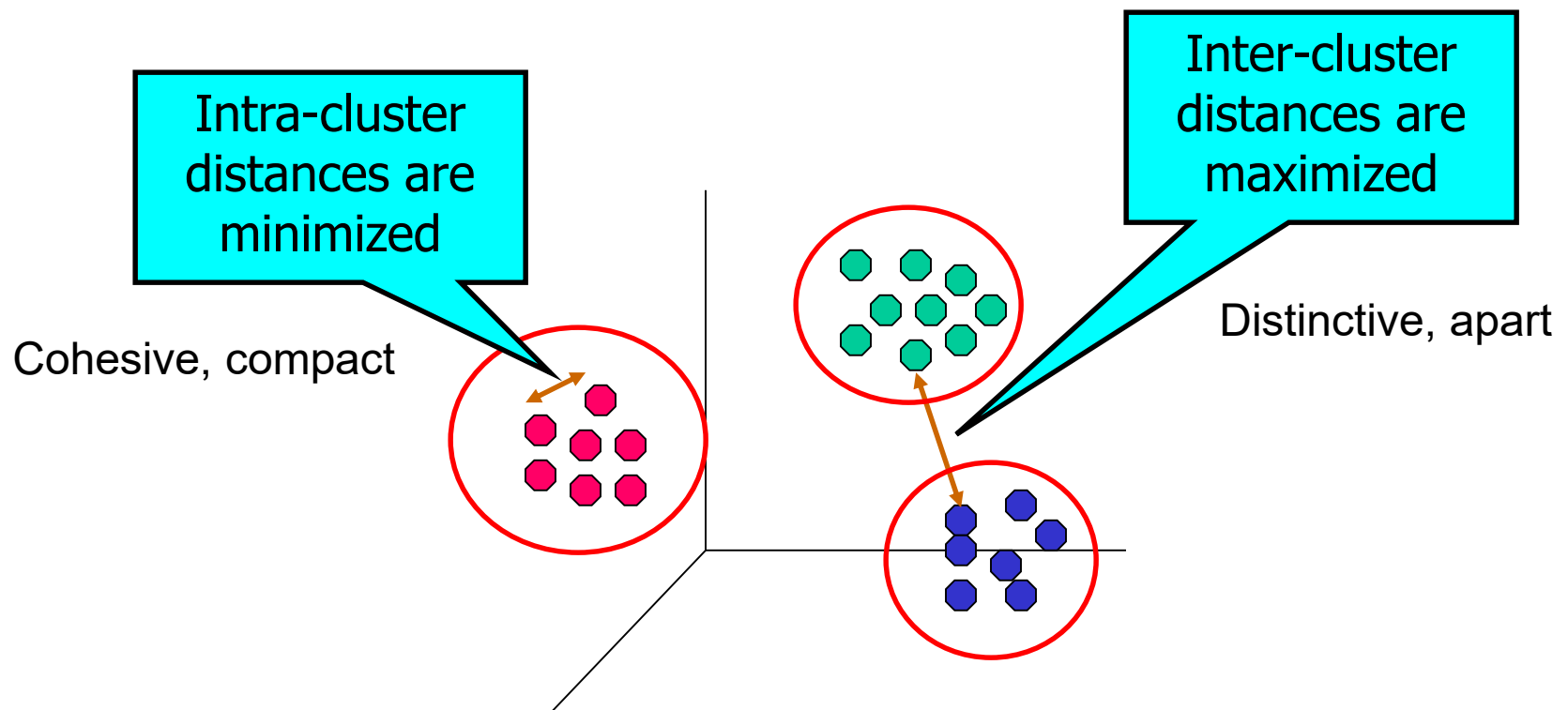
Outline

- **Clustering, aka unsupervised learning**
- **Association rule mining**
- **Classification, aka supervised learning**

CLUSTERING

Objective of cluster analysis

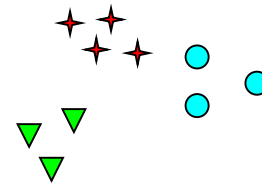
- Find groups of objects s.t. objects in a group are
 - Similar (or related) to one another
 - Diff from (or unrelated to) objects in other groups



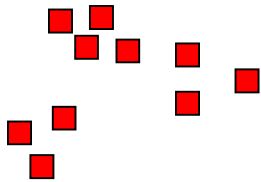
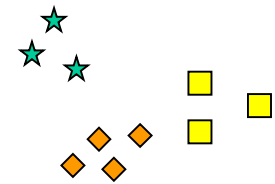
The notion of a “cluster” can be ambiguous



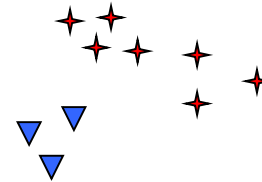
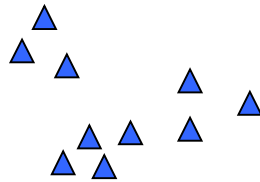
How many clusters?



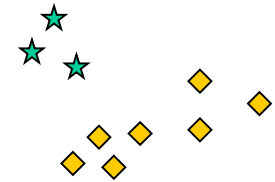
Six Clusters



Two Clusters



Four Clusters



Supervised vs unsupervised learning

- **Supervised learning (aka classification)**
 - Training data (observations, measurements, etc.) are accompanied by class
 - New data is classified based on training data
- **Unsupervised learning (aka clustering)**
 - Class labels of training data are unknown
 - Given a set of measurements, observations, etc., aim to establish existence of classes in the data

Clustering techniques

- **Partitional clustering: K-means**
 - Division of data objects into non-overlapping subsets (clusters) s.t. each data object is in exactly one subset
- **Hierarchical clustering: Agglomerative approach**
 - A set of nested clusters organized as a hierarchical tree
- **Subspace clustering and bi-/co-clustering**
 - Simultaneous clustering on a subset of tuples and a subset of attributes

Partitional clustering: K-means

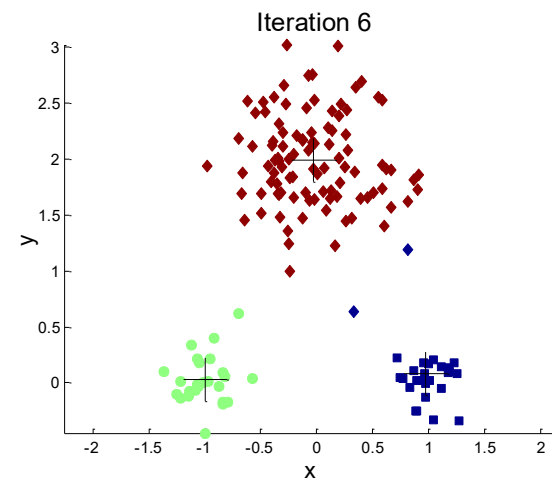
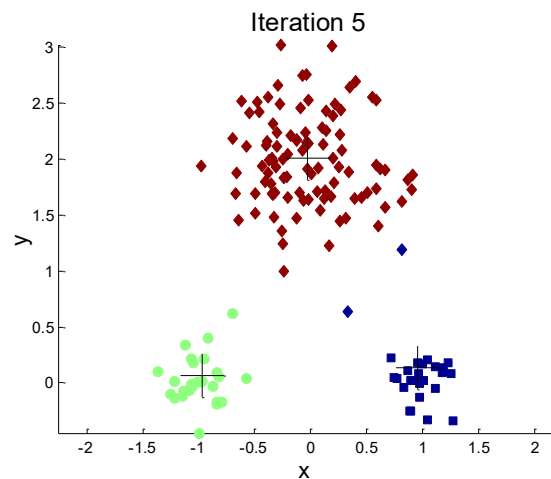
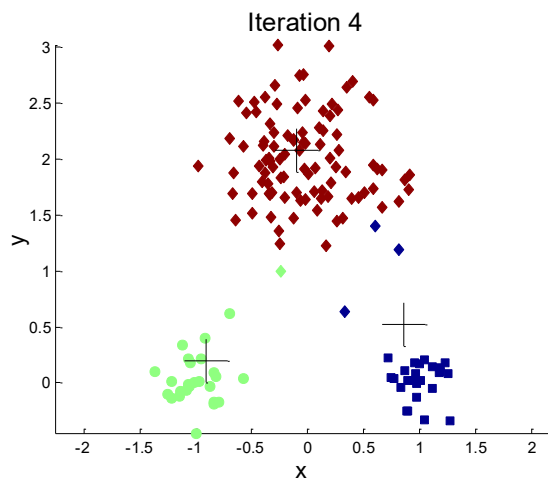
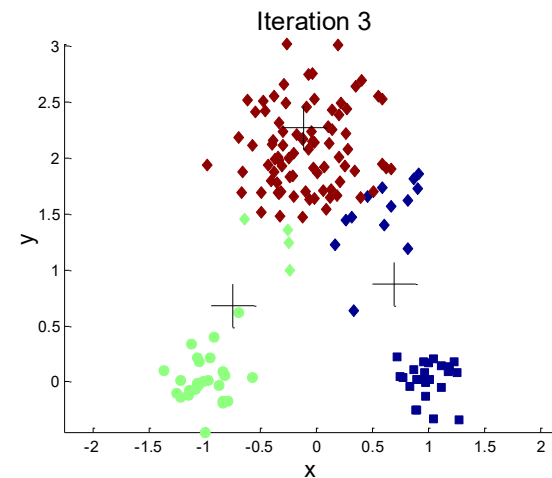
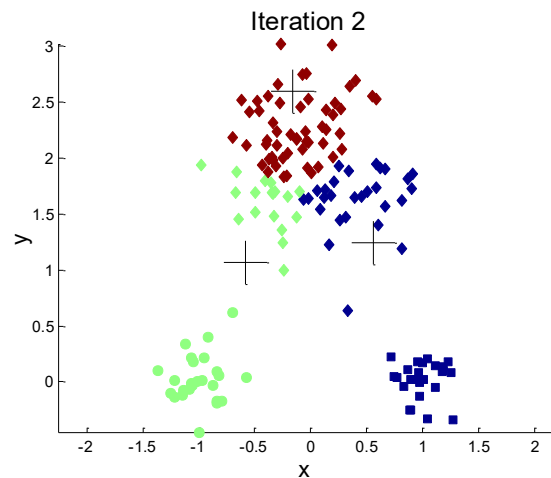
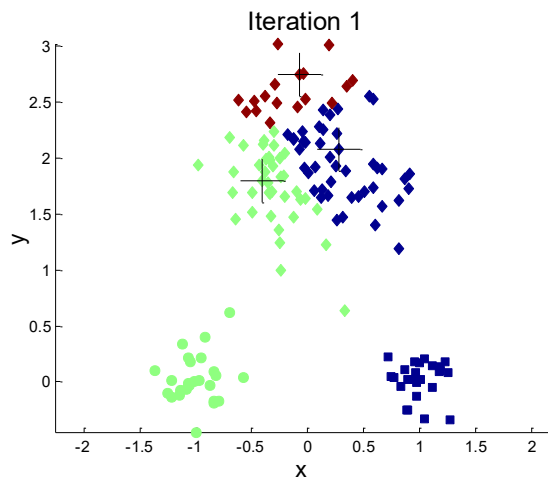
- Each cluster has a centroid
- Each point is assigned to a cluster based on closest centroid
- # of clusters, K , must be specified

-
- 1: Select K points as the initial centroids.
 - 2: **repeat**
 - 3: Form K clusters by assigning all points to the closest centroid.
 - 4: Recompute the centroid of each cluster.
 - 5: **until** The centroids don't change
-

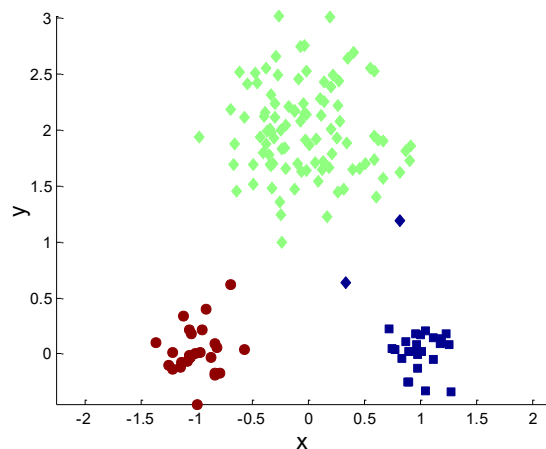
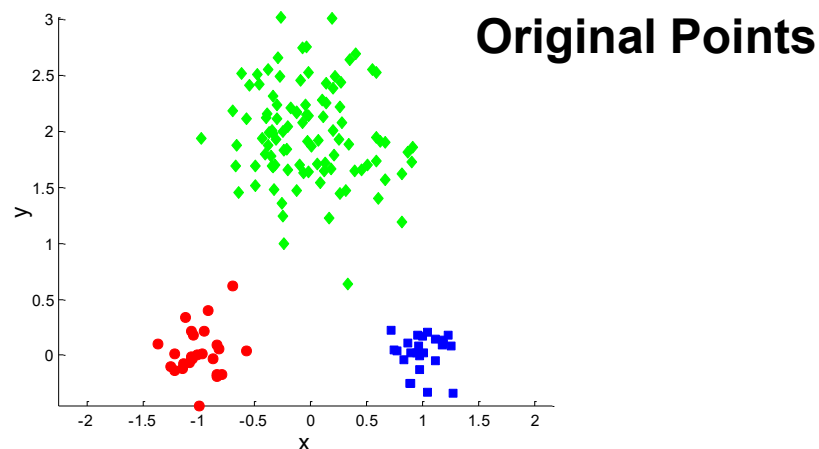
Details of K-means

- **Initial centroids are often chosen randomly**
 - Clusters produced vary from one run to another
- **Centroid is the “mean” of points in the cluster**
- **“Closeness” is measured by Euclidean distance, cosine similarity, correlation, etc**
- **K-means usually converges in a few iterations**
 - Often the stopping condition is changed to “until relatively few points change clusters”
- **Complexity is $O(n * K * i * d)$**
 - n = # of points, K = # of clusters, i = # of iterations, d = # of attributes

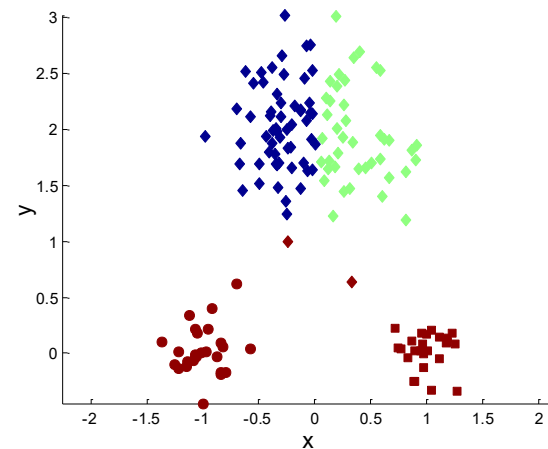
Example iterations by K-means



Two different K-means clusterings



Optimal Clustering



Sub-optimal Clustering

Evaluating K-means clusters

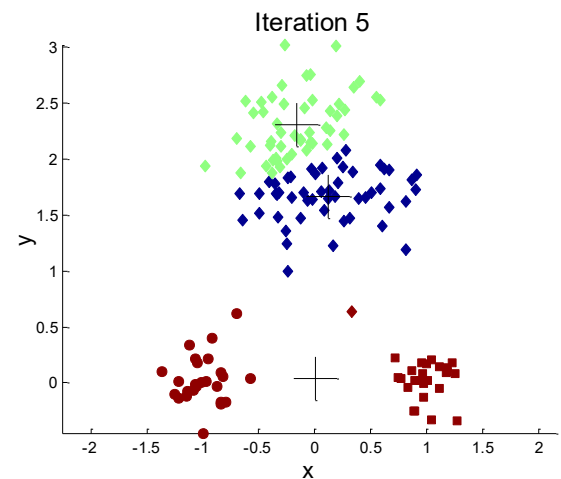
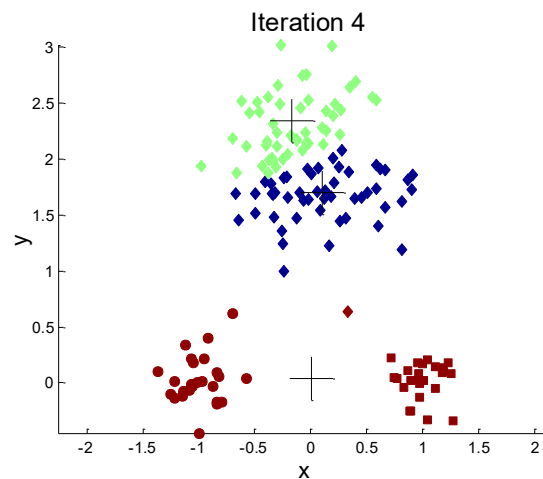
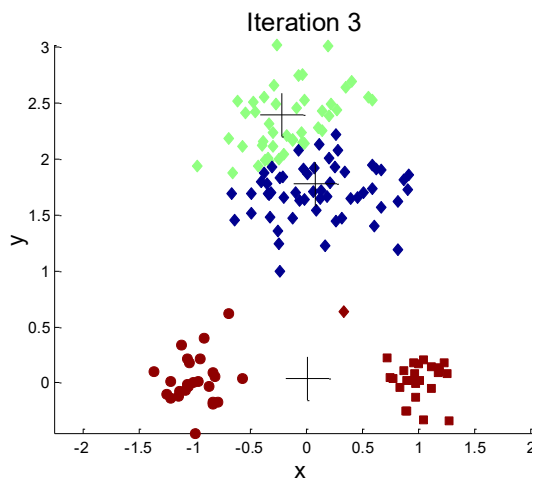
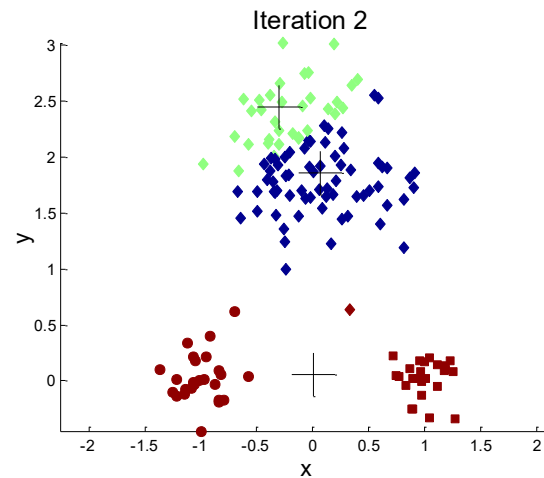
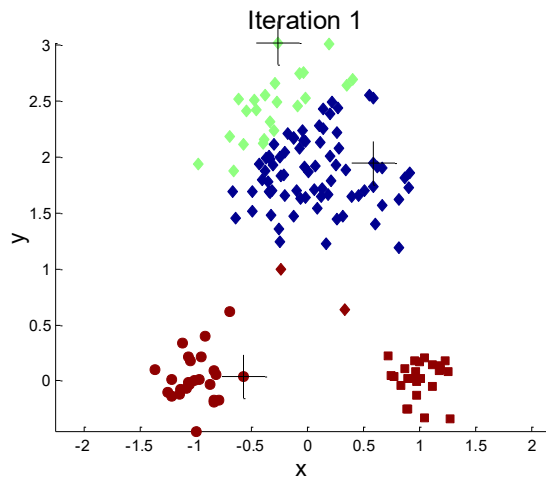
- **Sum of Squared Error (SSE) is commonly used**
 - Error of a point is its distance to nearest centroid
 - Square these errors and sum them to get SSE

$$SSE = \sum_{i=1}^K \sum_{x \in C_i} dist^2(m_i, x)$$

where C_i is a cluster, m_i is its centroid

- **Can reduce SSE by increasing K, the # of clusters**
- **A good clustering with smaller K can have a lower SSE than a poor clustering with higher K**

Importance of initial centroids



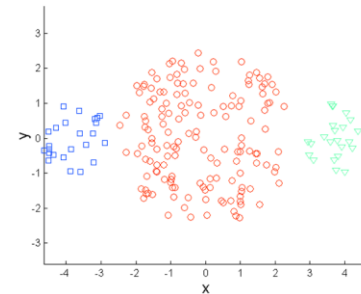
Solutions to initial-centroid problem

- **Multiple runs**
 - Helps, but probability is not on your side
- **Use hierarchical clustering to determine initial centroids**
- **Select $>k$ initial centroids and then select the most widely separated among these initial centroids**
- **Use more advanced algos, like “Bisecting K-Means”, that are not as susceptible to initialization issues**

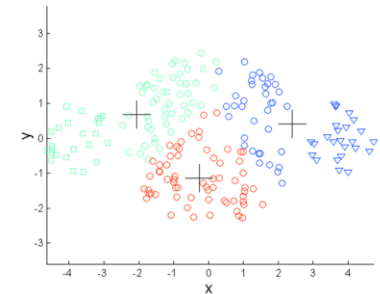
Limitations of K-means

- Has problems when clusters are of differing
 - Sizes
 - Densities
 - Non-globular shapes
- Also has problems when data contain outliers

Differing Sizes

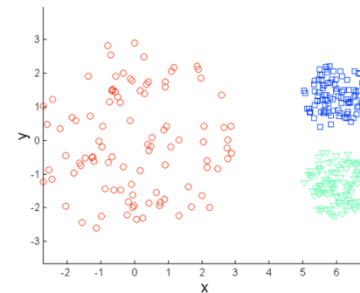


Original Points

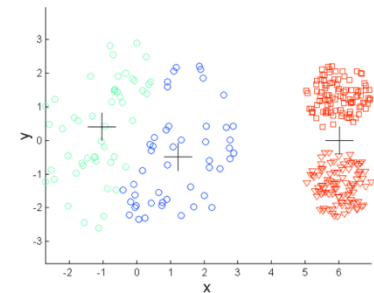


K-means (3 Clusters)

Differing Density

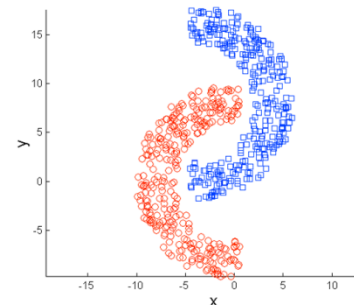


Original Points

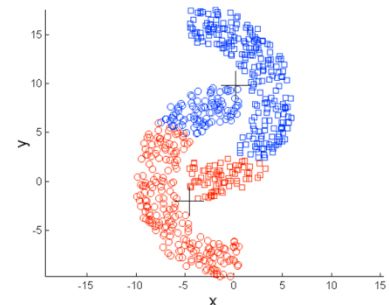


K-means (3 Clusters)

Non-globular Shapes



Original Points

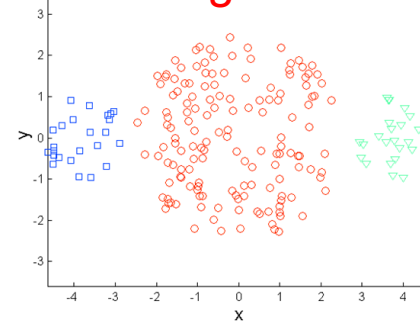


K-means (2 Clusters)

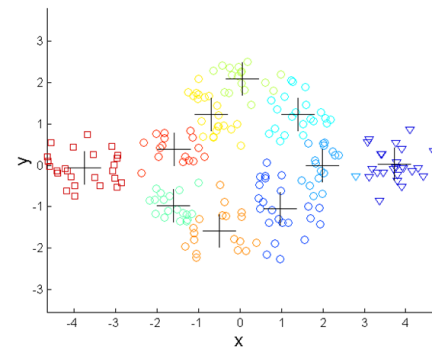
Overcoming K-means' limitations

- One solution is to use many clusters
 - Find parts of clusters
 - But need to put them together

Differing Sizes

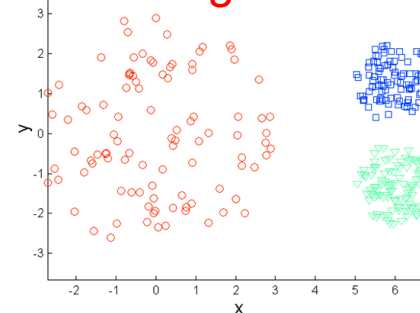


Original Points

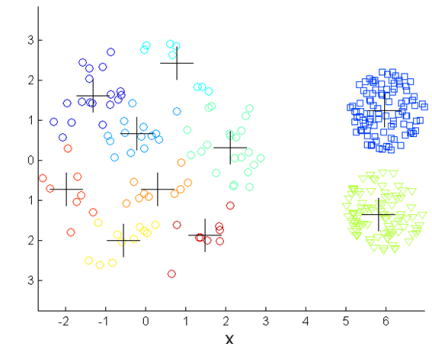


K-means Clusters

Differing Densities

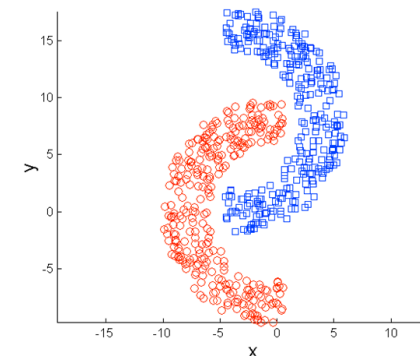


Original Points

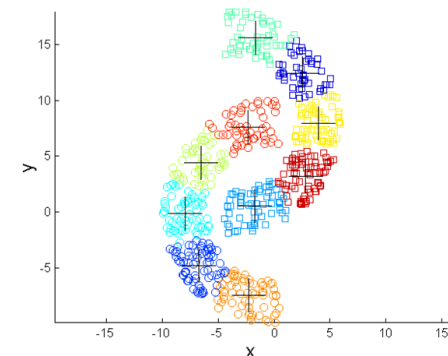


K-means Clusters

Nonglobular Shapes



Original Points



K-means Clusters

Hierarchical clustering

Hierarchical clustering

- Organize similar data into groups
 - Form groups into a hierarchical tree structure, termed a Dendrogram
-
- Offer useful visual descriptions of data
 - Two approaches
 - Agglomerative
 - **Build the tree by finding most related objects first**
 - Divisive
 - **Build the tree by finding most dissimilar objects first.**

Which pairs of tuples are similar
based on the data matrix?

n features (order of 1000)

m tuples

gene1	gene2	gene3	gene4	...	gene_n
x_{11}	x_{12}	x_{13}	x_{14}	...	x_{1n}
x_{21}	x_{22}	x_{23}	x_{24}	...	x_{2n}
x_{31}	x_{32}	x_{33}	x_{34}	...	x_{3n}
.....
x_{m1}	x_{m2}	x_{m3}	x_{m4}	...	x_{mn}

Similar?

Distance (similarity) Matrix

	p1	p2	p3	p4	p5	...
p1						
p2				P(2,4)		
p3						
p4						
p5						
.						
.						
.						

$P(i, j) = \text{dist}(p_i, p_j)$

Distance matrix

- Square, symmetrical
- Element value is based on a similarity function, e.g., Euclidian distance
- Sometimes, it's called a **Similarity Matrix** or a **Proximity Matrix**

Agglomerative hierarchical clustering

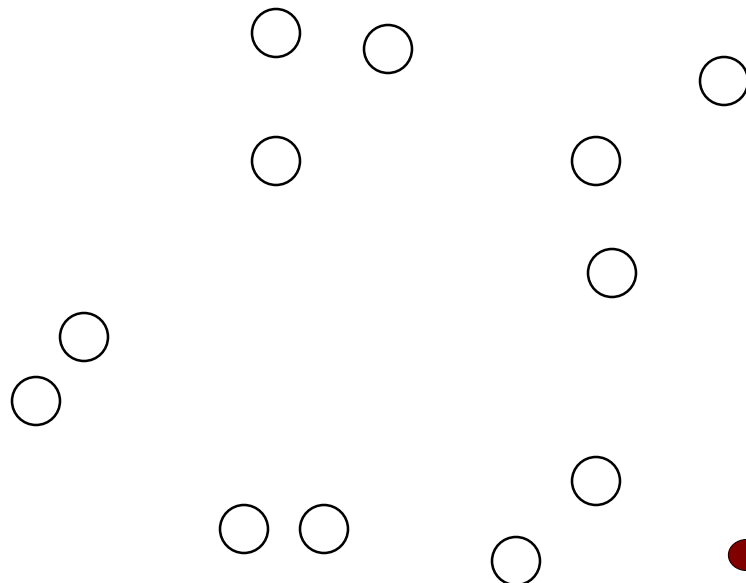
- **Basic algo is straightforward**

Compute proximity matrix
Let each data point be a cluster
Repeat
 Merge the two closest clusters
 Update the proximity matrix
Until only a single cluster remains

- **Key is computing proximity of two clusters**
 - Diff approaches to defining distance betw clusters distinguish the diff algos

Starting situation

- Start with clusters of individual points and a proximity matrix



	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

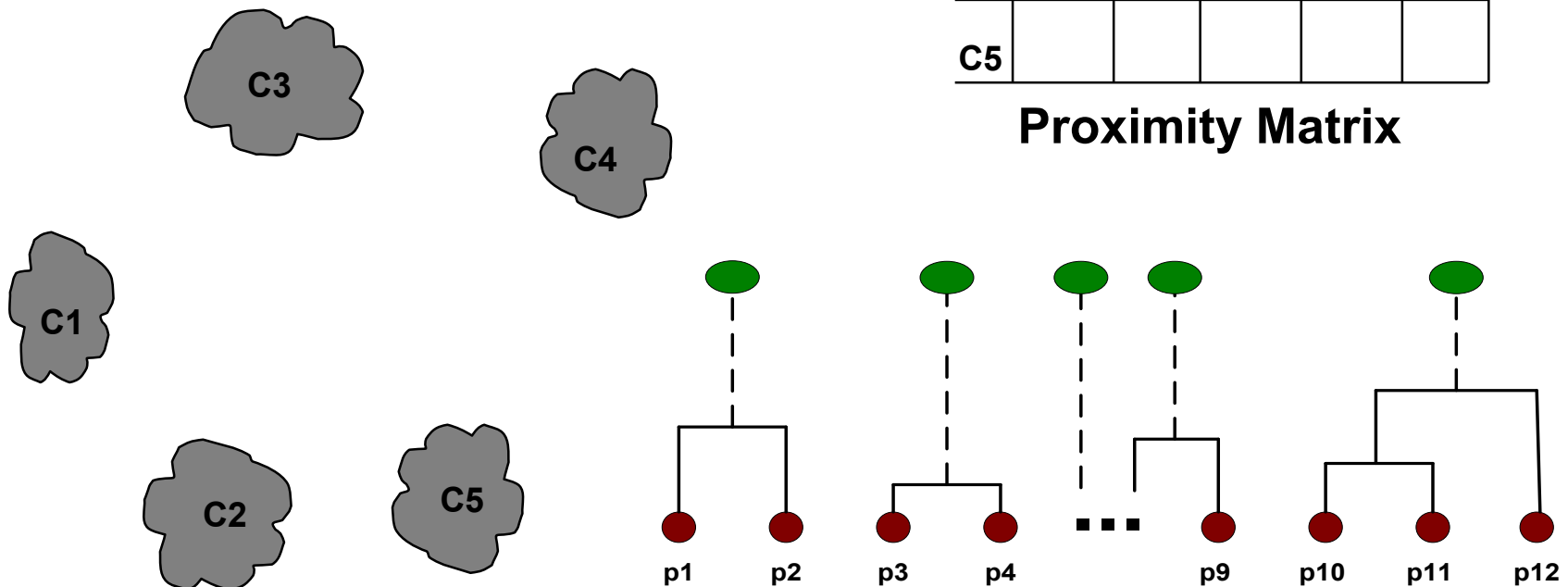


Intermediate situation

- After some merging steps, we have some clusters

	C1	C2	C3	C4	C5
C1					
C2					
C3					
C4					
C5					

Proximity Matrix

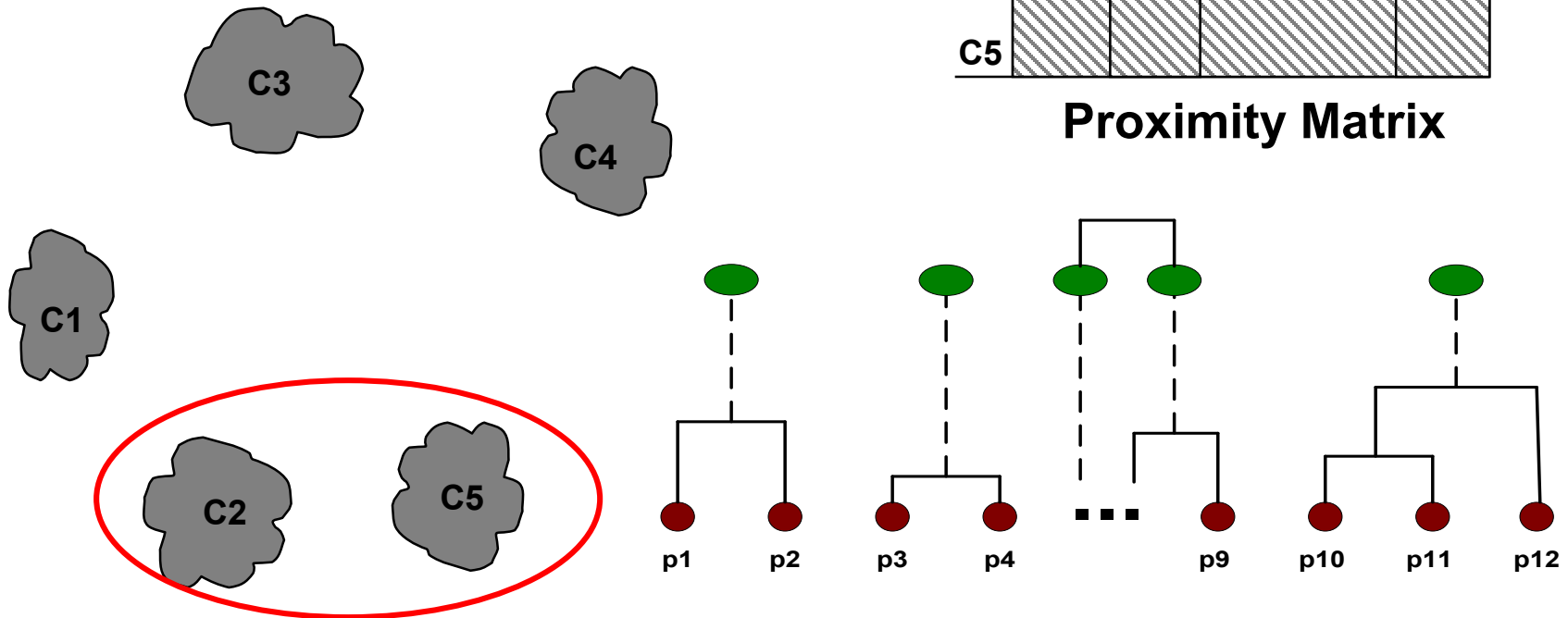


Intermediate situation

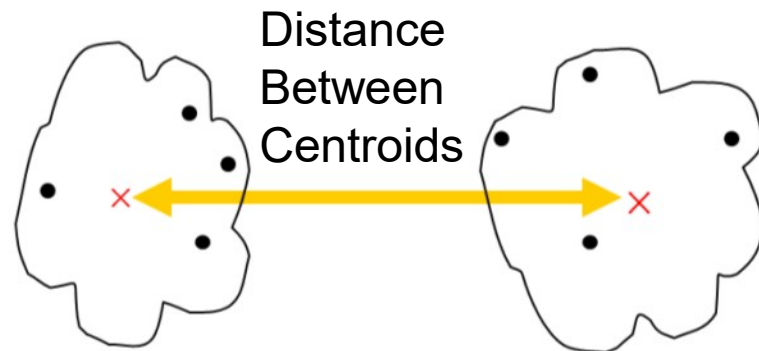
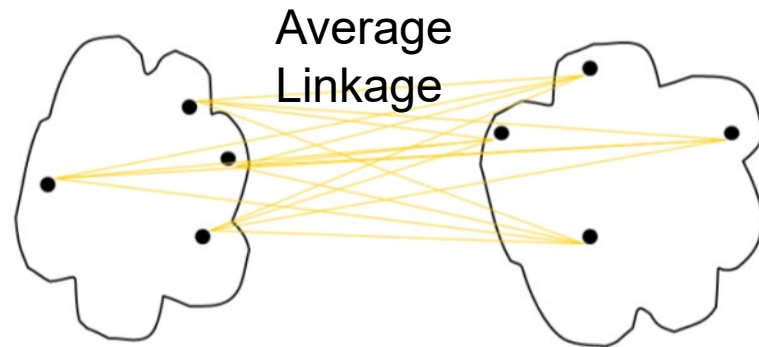
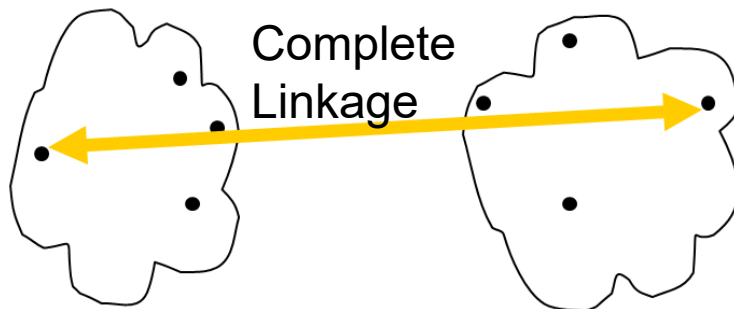
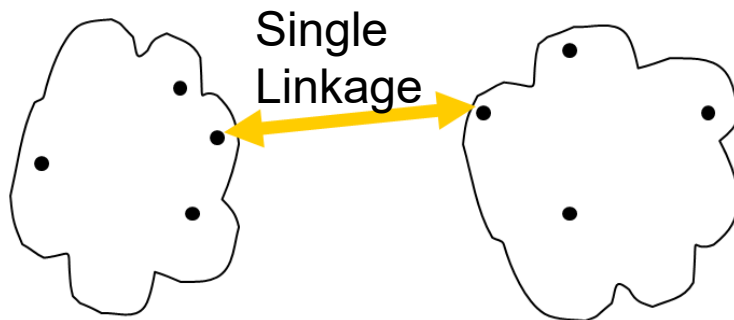
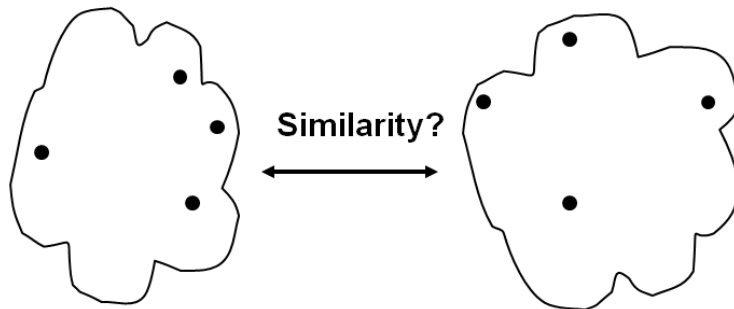
- We want to merge two closest clusters (C2, C5) and update the proximity matrix

	C1	C2	C3	C4	C5
C1					
C2					
C3					
C4					
C5					

Proximity Matrix

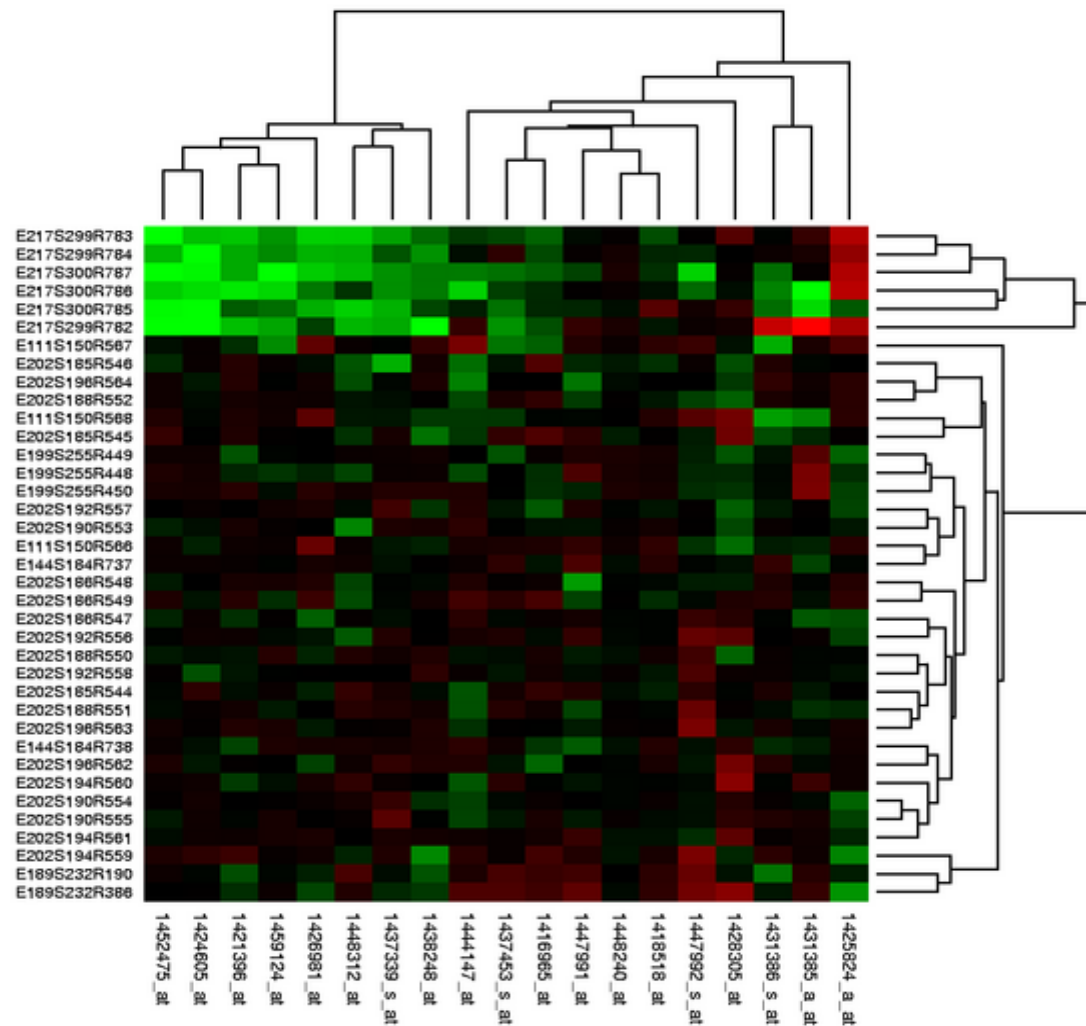


Defining inter-cluster similarity



- **Other methods use an objective function**
 - Ward's method uses squared error

Finally, get a resulting dendrogram



Strengths of hierarchical clustering

- **No need to assume any particular # of clusters**
 - Any desired number of clusters can be obtained by 'cutting' the dendrogram at the proper level
- **They may correspond to meaningful taxonomies**
 - Example in biological sciences (e.g., animal kingdom, phylogeny reconstruction, ...)

Divisive hierarchical clustering

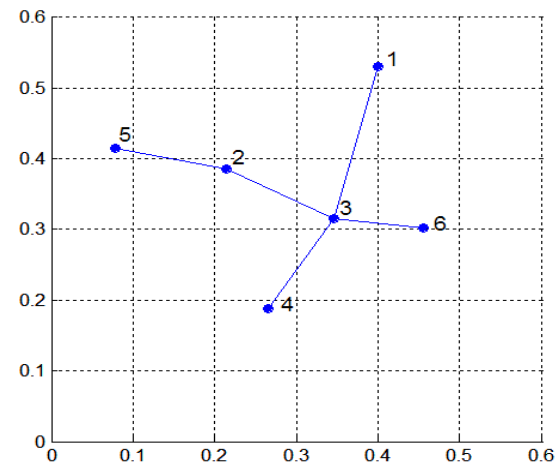
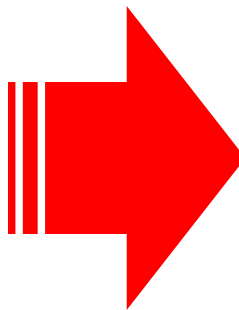
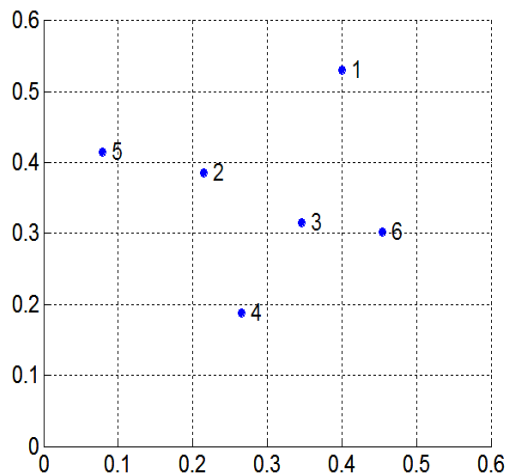
- **Start with one, all-inclusive cluster**
- **At each step, split a cluster until each cluster contains a point (or there are k clusters)**

Algorithm 7.5 MST Divisive Hierarchical Clustering Algorithm

- 1: Compute a minimum spanning tree for the proximity graph.
 - 2: **repeat**
 - 3: Create a new cluster by breaking the link corresponding to the largest distance (smallest similarity).
 - 4: **until** Only singleton clusters remain
-

In case you don't know what a MST is...

- **To build a MST (Minimum Spanning Tree)**
 - Start with a tree that consists of any point
 - In successive steps, look for the closest pair of points (p, q) s.t. p is in the current tree but q is not
 - Add q to the tree and put an edge betw p and q



Subspace clustering

- **Cluster boundaries clear only wrt the subspaces**

Bi- or co-clustering

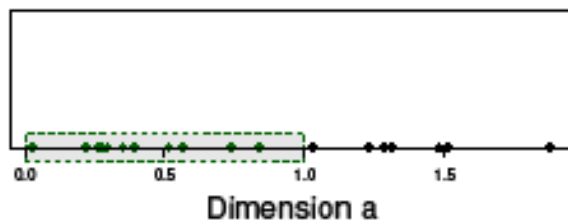
- **Simultaneous clustering on a subset of attributes and a subset of tuples**

High-dimensional data

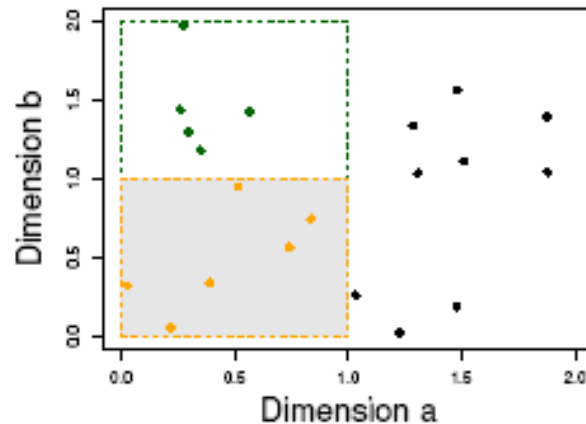
- **Many applications need clustering on high-dimensional data**
 - Text documents
 - Microarray data
- **Major challenges:**
 - Many irrelevant dimensions may mask clusters
 - Distance measure becomes meaningless
 - **The “equi-distance” phenomenon**
 - Clusters may exist only in some subspaces

Curse of dimensionality

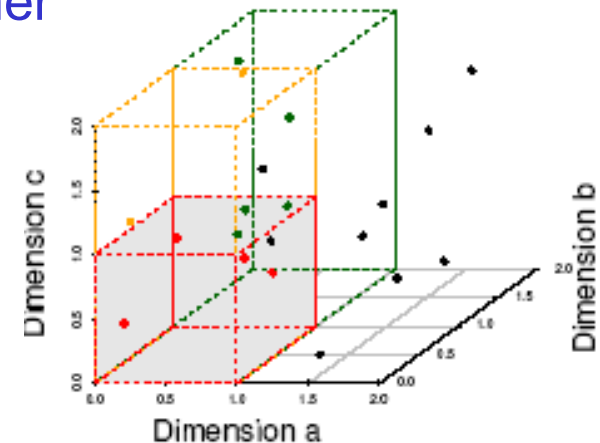
- Data in only one dimension is relatively packed
 - Adding a dimension “stretches” the points across that dimension, making them further apart
 - Adding more dimensions makes the points further apart
 - High-dimensional data is sparse
- ⇒ Distance measure becomes meaningless, as most data points become equi-distance to each other



(a) 11 Objects in One Unit Bin



(b) 6 Objects in One Unit Bin



(c) 4 Objects in One Unit Bin

Image credit: Parsons et al. *KDD Explorations*, 2004

Why subspace clustering?

- Clusters may exist only in some subspaces
- Subspace-clustering: find clusters in all the subspaces

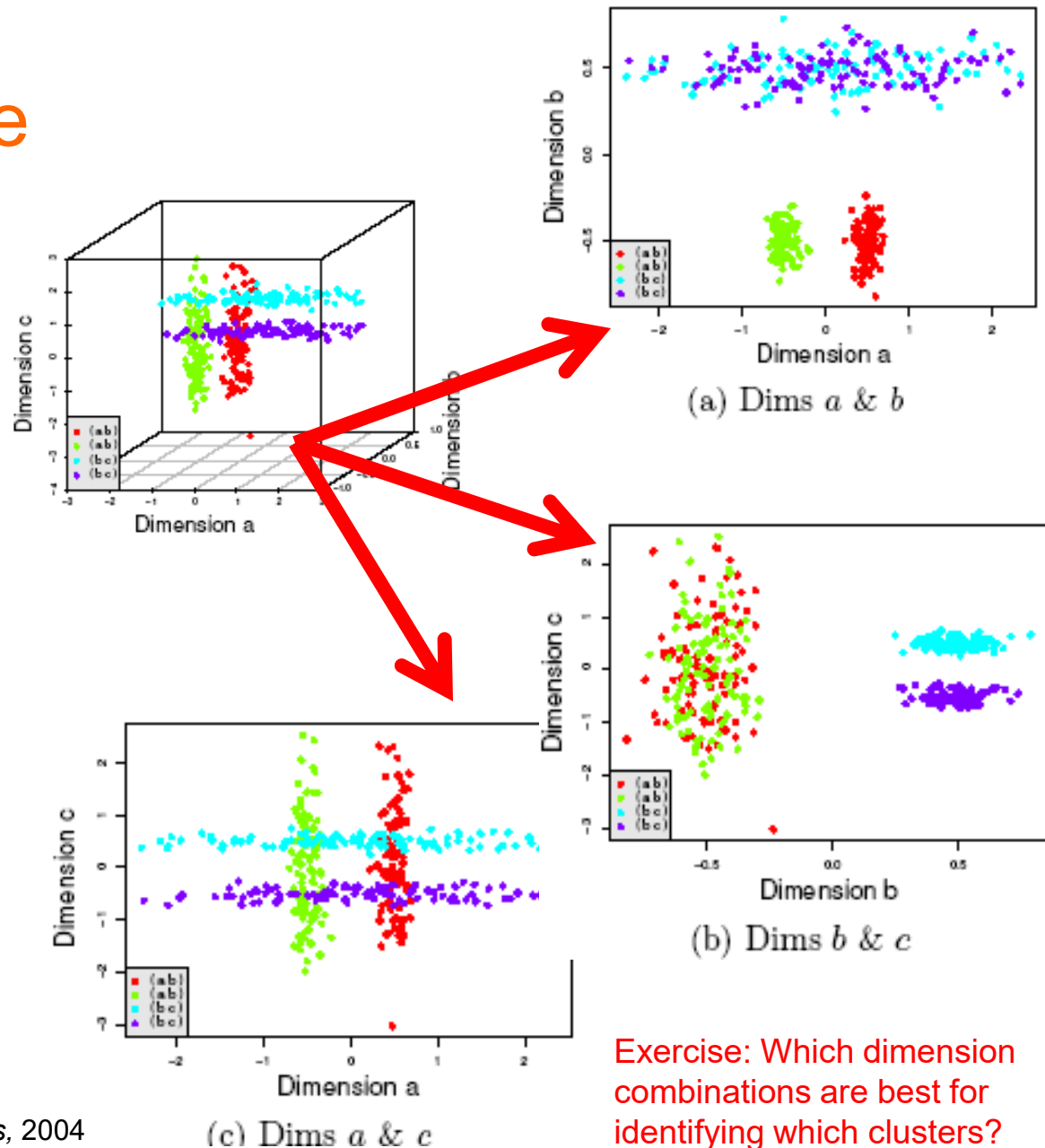
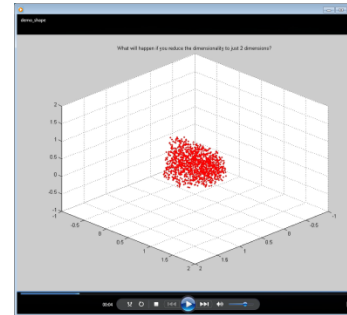
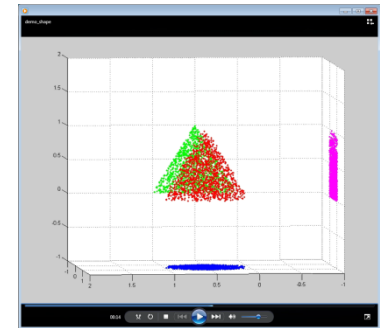


Image credit: Parsons et al. *KDD Explorations*, 2004

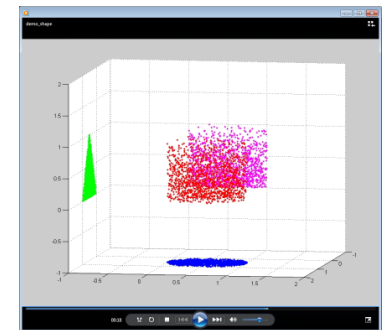
A cloud of points in 3D



In 2D **XZ** we see ...



In 2D **YZ** we see...



In 2D **XY** we see ...

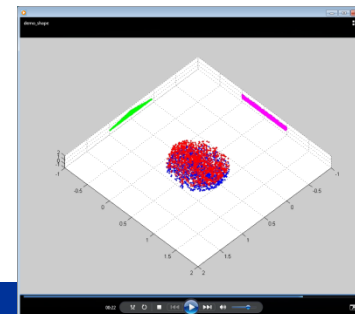


Image credit: Eamonn Keogh

Time for Exercise #1

- The picture shows random sets of 3x3 cells selected from randomly generated $n \times n$ samples for various n . The darker the cell, the higher the value it contains
- Based on this picture, discuss the effect of high dimensionality on clustering and feature selection

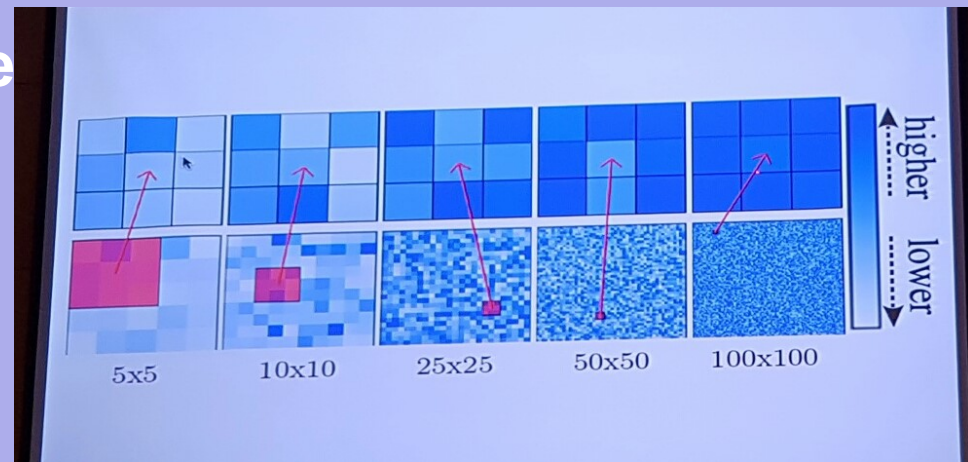


image of a slide from a talk by Koji Tsuda

CLIQUE (Clustering In QUES)

- **Automatically identify subspaces of a high dimensional data space that allow better clustering than original space**
 - Agrawal et al. “Automatic Subspace Clustering of High Dimensional Data”. *Data Min. Knowl. Discov.*, 11(1):5-33, 2005

CLIQUE: The Major Steps

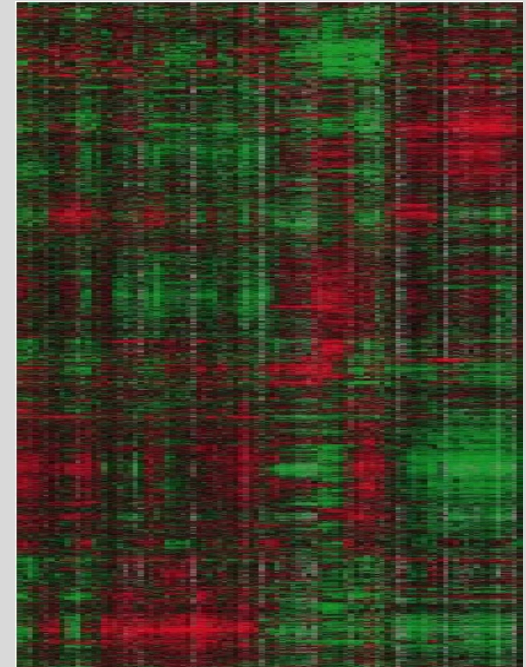
- **Partition the data space**
- **Identify subspaces that contain clusters**
 - Use the “Apriori Principle”
 - Find dense units in all subspaces
 - Form connected dense units in all subspaces
- **Generate minimal description for the clusters**
 - Determine maximal regions that cover a cluster of connected dense units
 - Determination of minimal cover for each cluster

Biclustering

- **Please read these two papers yourself ☺**
 - Cheng & Church. “Biclustering of expression data”. *ISMB 2000*
 - Madeira & Oliveira. “Biclustering algorithms for biological data analysis: A survey”. *TCBB*, vol.1, 2004

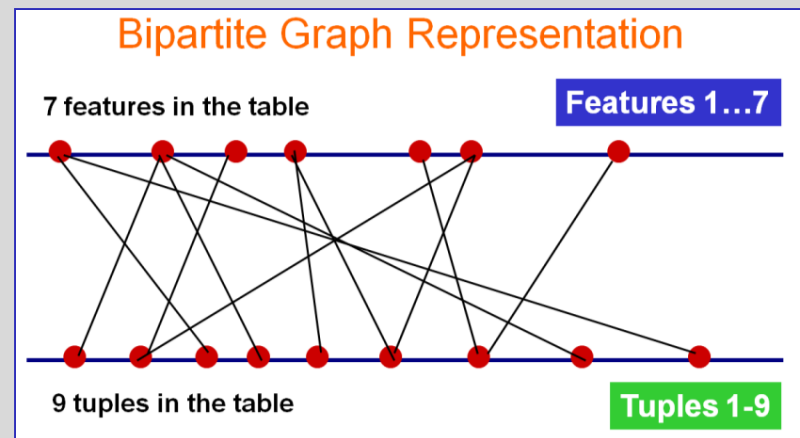
**Biclusters = small
boxes of homogeneity**

**A small box =
A subset of attributes X
A subset of tuples**



A special case of biclustering: Biclique detection

- When the table is a binary matrix of 0s and 1s
- Convert the table into a bipartite graph



- Then, a max biclique corresponds to a bicluster
- A good algo for max biclique can be found at
 - Li et al. “Maximal biclique subgraphs and closed pattern pairs of the adjacency matrix: A one-to-one correspondence and mining algorithms”. *TKDE*, 19:1625-1637, 2007

What have we learned?

- **Partitional clustering**
 - K-means
- **Hierarchical clustering**
 - Agglomerative approach
 - Divisive approach
- **Subspace clustering and bi-/co-clustering, albeit rather briefly!**
- **How to evaluate quality of clusters**
 - SSE
- **A general strategy for some difficult-to-cluster situations**
 - Differing sizes
 - Differing densities
 - Non-globular

References

- **Must read**

- Jain et al. “Data clustering: A review”. *ACM computing Surveys*, 31(3):264-323, 1999

- **Good to read**

- Agrawal et al. “Automatic Subspace Clustering of High Dimensional Data”. *Data Mining & Knowledge Discovery*, 11(1):5-33, 2005
- Cheng & Church. “Biclustering of expression data”. *ISMB 2000*, pp. 93-103
- Madeira & Oliveira. “Biclustering algorithms for biological data analysis: A survey”. *ACM TCBB*, 1(1):24-45, 2004
- Li et al. “Maximal biclique subgraphs and closed pattern pairs of the adjacency matrix: A one-to-one correspondence and mining algorithms”. *TKDE*, 19:1625-1637, 2007

For those who want to go further...

- **Much progress has been made in scalable clustering methods**
 - Partitioning: k-means, k-medoids, CLARANS
 - Hierarchical: BIRCH, ROCK, CHAMELEON
 - Density-based: DBSCAN, OPTICS, DenClue
 - Grid-based: STING, WaveCluster, CLIQUE
 - Model-based: EM, Cobweb, SOM
 - Frequent pattern-based: pCluster
 - Constraint-based: COD, constrained-clustering

ASSOCIATION RULE MINING

Market-basket analysis

- What do my customers buy?
- Which products are bought together?



- Find associations and correlations between the different items that customers buy

Source: A. Puig

Association rule mining

TID	Items
T1	bread, jelly, peanut-butter
T2	bread, peanut-butter
T3	bread, milk, peanut-butter
T4	beer, bread
T5	beer, milk

- **Frequent itemsets**
 - Items that often appear together
 - {bread, peanut-butter}
- **Association rules**
 - bread \Rightarrow peanut-butter

- Transaction db $T = \{t_1, \dots, t_n\}$ is a set of trans
- Each trans t_k is an **itemset** $I = \{i_1, \dots, i_m\}$
- Find **freq patterns**, associations, ... among sets of items in T
- Represent these relationships as **association rules** $X \Rightarrow Y$

What is an interesting rule?

- **Support count, σ**
 - # of occurrence of an itemset
 - $\sigma(\{\text{bread, peanut-butter}\}) = 3$

TID	Items
T1	bread, jelly, peanut-butter
T2	bread, peanut-butter
T3	bread, milk, peanut-butter
T4	beer, bread
T5	beer, milk

- **Support, s**
 - Fraction of transactions containing that itemset
 - $s(\{\text{bread, peanut-butter}\}) = 3/5$

- **Frequent itemset**
 - An itemset whose support \geq a threshold minsup

What is an interesting rule?

- **Association rule**
 - $X \Rightarrow Y$
- **Support, s**
 - # of trans containing X, Y
- **Confidence, c**
 - How often Y occurs in trans containing X

$$s = \frac{\sigma(X \cup Y)}{\# \text{ of trans.}} \quad c = \frac{\sigma(X \cup Y)}{\sigma(X)}$$

TID	Items
T1	bread, jelly, peanut-butter
T2	bread, peanut-butter
T3	bread, milk, peanut-butter
T4	beer, bread
T5	beer, milk

TID	s	c
bread \Rightarrow peanut-butter	0.60	0.75
peanut-butter \Rightarrow bread	0.60	1.00
beer \Rightarrow bread	0.20	0.50
peanut-butter \Rightarrow jelly	0.20	0.33
jelly \Rightarrow peanut-butter	0.20	1.00
jelly \Rightarrow milk	0.00	0.00

Source: A. Puig

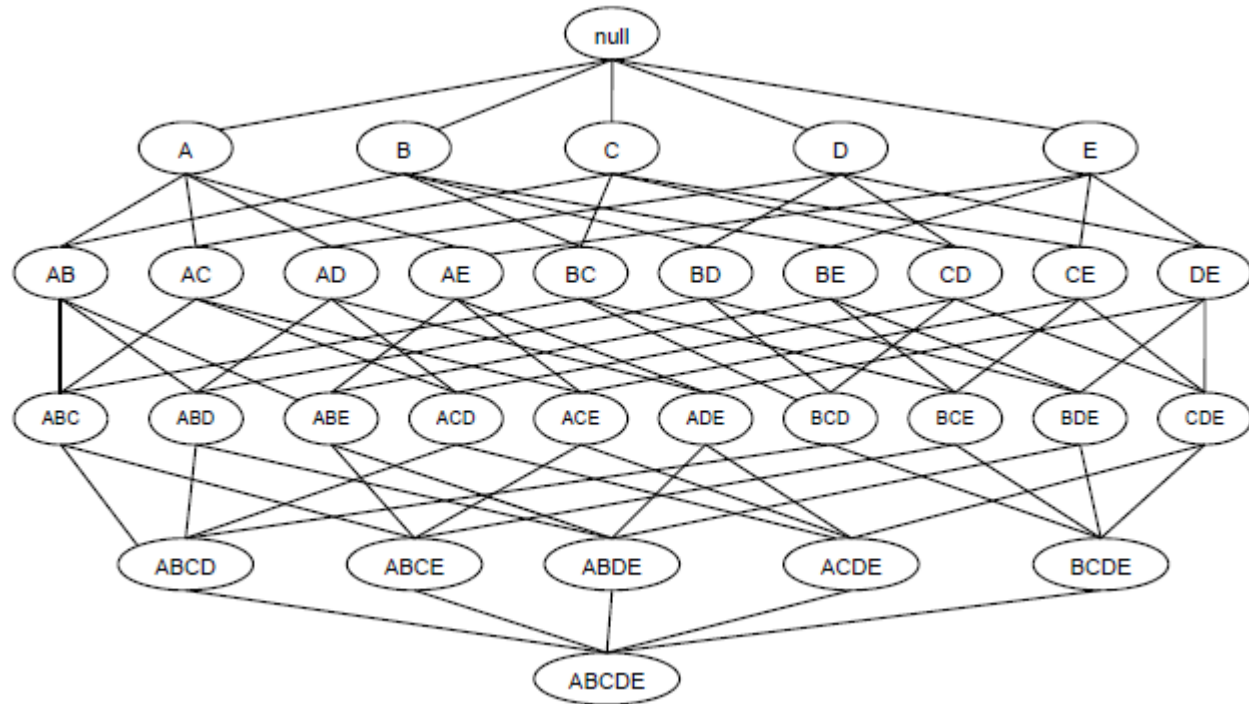
Apriori

- **Apriori is the classic assoc rule mining algo**
 - Agrawal & Srikant. “Fast algorithms for mining association rules in large databases”. *VLDB 1994*, pp. 487-499
- **Mines assoc rules in two steps**
 1. Generate all freq itemsets with support \geq minsup
 2. Generate assoc rules using these freq itemsets

Let's work on Step 1 first...

Step 1 of Apriori:

Generate freq itemsets with
 $\text{support} \geq \text{minsup}$



- Given d items. There are 2^d possible itemsets
- Do we need to generate them all?

Source: A. Puig

Anti-monotonicity

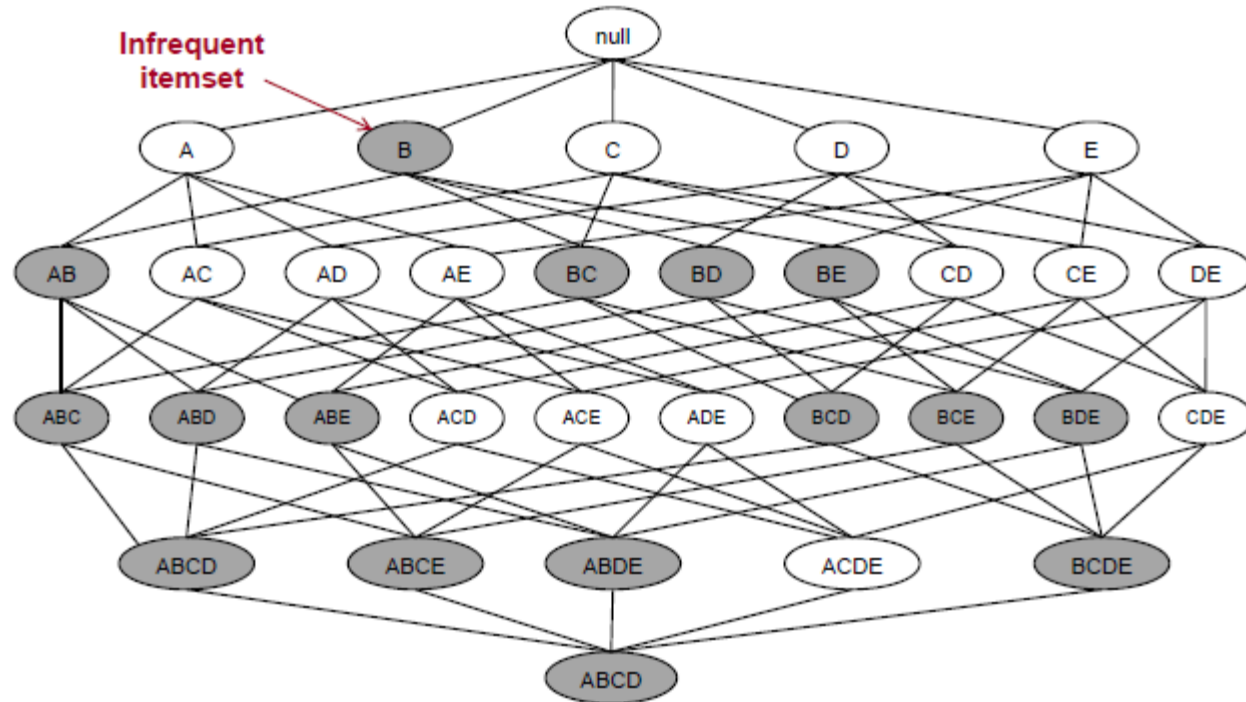
- **Downward Closure Property:**

Any subset of a frequent itemset is frequent

- ⇒ If an itemset is not frequent, none of its supersets can be frequent
- ⇒ If an itemset is not frequent, there is no need to explore its supersets

Step 1 of Apriori:

Generate freq itemsets with
 $\text{support} \geq \text{minsup}$



- By anti-monotonicity, if B's support $< \text{minsup}$, we can prune all its supersets. I.e., no need to generate these itemsets

Source: A. Puig

Apriori's Step 1 in Pseudo Codes

- **k=1**
- **Generate frequent itemsets of length 1**
- **Repeat until no frequent itemsets are found**
 - $k := k+1$
 - Generate itemsets of size k from the k-1 frequent itemsets
 - Compute the support of each candidate by scanning DB

Algorithm Apriori(T)

```

 $C_1 \leftarrow \text{init-pass}(T);$ 
 $F_1 \leftarrow \{f \mid f \in C_1, f.\text{count}/n \geq \text{minsup}\};$ 
for ( $k = 2; F_{k-1} \neq \emptyset; k++$ ) do
     $C_k \leftarrow \text{candidate-gen}(F_{k-1});$ 
    for each transaction  $t \in T$  do
        for each candidate  $c \in C_k$  do
            if  $c$  is contained in  $t$  then
                 $c.\text{count}++;$ 
            end
        end
     $F_k \leftarrow \{c \in C_k \mid c.\text{count}/n \geq \text{minsup}\}$ 
end
return  $F \leftarrow \bigcup_k F_k;$ 
  
```

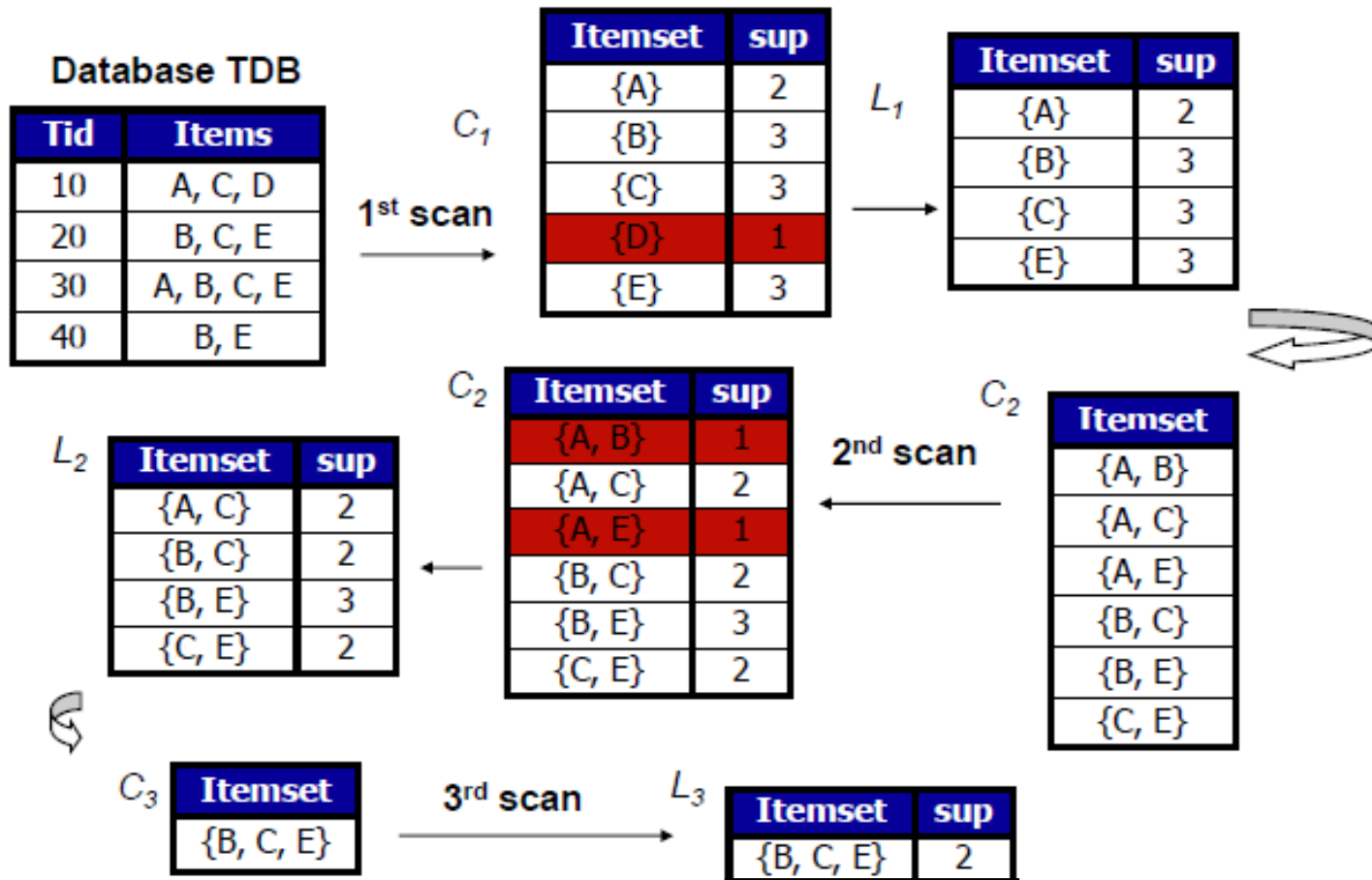
Function candidate-gen(F_{k-1})

```

 $C_k \leftarrow \emptyset;$ 
forall  $f_1, f_2 \in F_{k-1}$ 
    with  $f_1 = \{i_1, \dots, i_{k-2}, i_{k-1}\}$ 
    and  $f_2 = \{j_1, \dots, j_{k-2}, j'_{k-1}\}$ 
    and  $i_{k-1} < j'_{k-1}$  do
         $c \leftarrow \{i_1, \dots, i_{k-1}, j'_{k-1}\};$ 
         $C_k \leftarrow C_k \cup \{c\};$ 
        for each ( $k-1$ )-subset  $s$  of  $c$  do
            if ( $s \notin F_{k-1}$ ) then
                delete  $c$  from  $C_k;$ 
            end
        end
    end
return  $C_k;$ 
  
```

anti-monotonicity is used here

Example run of Apriori's step 1



Source: A. Puig

Apriori

- **Apriori is the classic assoc rule mining algo**
 - Agrawal & Srikant. “Fast algorithms for mining association rules in large databases”. *VLDB 1994*, pp. 487-499
- **Mines assoc rules in two steps**
 1. Generate all freq itemsets with support \geq minsup
 2. Generate assoc rules using these freq itemsets

Now that we have settled Step 1,
Let's work on Step 2 next...

Step 2 of Apriori:

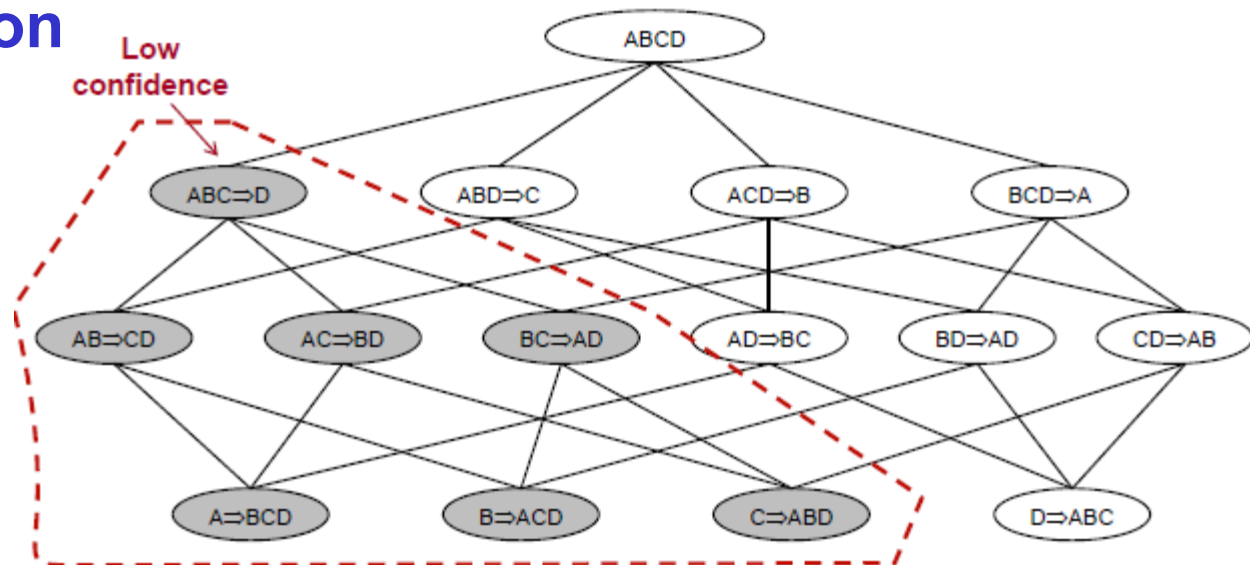
Generate association rules using freq itemsets

- **Given a frequent itemset L**
 - Find all non-empty subsets F of L
 - Output each rule $F \Rightarrow \{L-F\}$ that satisfies the threshold on confidence
- **Example: L = {A, B, C}**
 - The candidate itemsets are: $AB \Rightarrow C$, $AC \Rightarrow B$, $BC \Rightarrow A$, $A \Rightarrow BC$, $B \Rightarrow AC$, $C \Rightarrow AB$
 - In general, there are $2^{|L|} - 2$ candidates!

Can we be more efficient?

- Confidence of rules generated from the same itemset does have the anti-monotone property
 - $c(ABC \Rightarrow D) \geq c(AB \Rightarrow CD) \geq c(A \Rightarrow BCD)$
- We can apply this property to prune rule generation

Exercise:
Prove this.



Source: A. Puig

Shortcomings of Apriori

- **Apriori scans the db multiple times**
- **There is often a high # of candidates**
- **Support counting for candidates takes a lot of time**
- **Newer methods try to improve on these points**
 - Reduce the # of scans of the db
 - Reduce the # of candidates
 - Count the support of candidates more efficiently

Han et al. “Mining frequent patterns without candidate generation”.
SIGMOD 2000, pp.1–12



FP-Growth

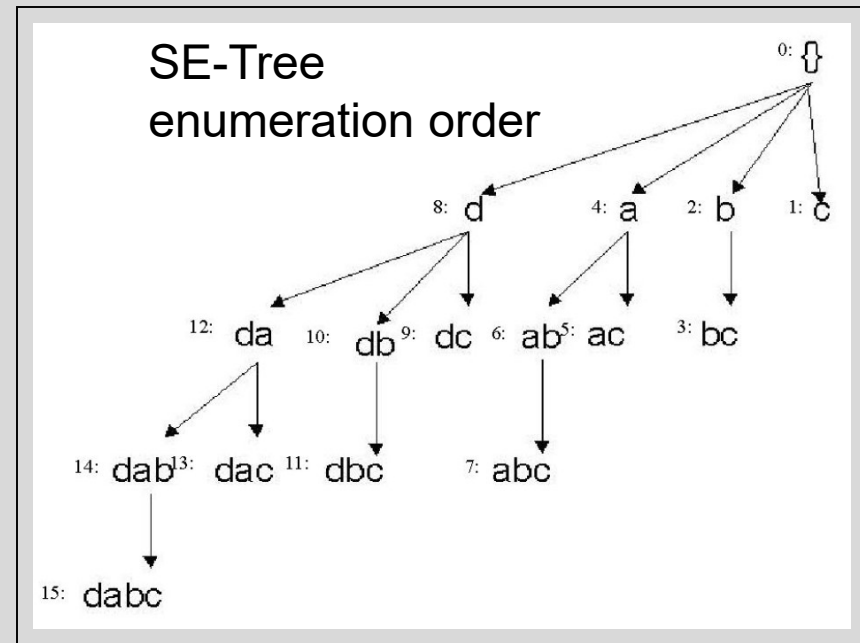
- **Build in one scan a data structure, FP-Tree**

Li et al. "Mining Statistically Important Equivalence Classes and Delta-Discriminative Emerging Patterns". *KDD 2007*, pp. 430--439



Gr-Growth

- **Build FP-Tree on the db**
- **Visit itemsets non-redundantly by following the right-to-left top-to-bottom SE-Tree order**
- **When visiting an itemset**
 - Use the FP-tree to count its support efficiently
 - If it is frequent, output it, & visit its supersets
 - Otherwise skip visiting its supersets



How do you mine association rules across multiple tables?

Single vs. Multidimensional Association Rules

- Single-dimensional rules
 $\text{buys}(X, \text{"milk"}) \Rightarrow \text{buys}(X, \text{"bread"})$
- Multi-dimensional rules: more than 2 dimensions or predicates
 $\text{age}(X, \text{"19-25"}) \wedge \text{buys}(X, \text{"popcorn"}) \Rightarrow \text{buys}(X, \text{"coke"})$
- Transformation into single-dimensional rules:
 use predicate/value pairs as items
 $\text{customer}(X, [\text{age}, \text{"19-25"}]) \wedge \text{customer}(X, [\text{buys}, \text{"popcorn"}])$
 $\Rightarrow \text{customer}(X, [\text{buys}, \text{"coke"}])$
- Simplified Notation for single dimensional rules
 $\{\text{milk}\} \Rightarrow \{\text{bread}\}$
 $\{[\text{age}, \text{"19-25"}], [\text{buys}, \text{"popcorn"}]\} \Rightarrow \{[\text{buys}, \text{"coke"}]\}$

Multidimensional association rules can be mined using the same method by transforming the problem. The items and the corresponding item values are encoded into a tuple. This results again in a finite number of possible (modified) item values, and therefore the same techniques as for single-dimensional rules apply.

Source: Karl Aberer

What have we learned?

- **Frequent itemsets & association rules**
- **Support & confidence**
- **Apriori, a classic association rule mining algo**
 - Anti-monotonicity
 - Search space pruning
- **Advanced methods**, albeit rather briefly
 - FP-Growth
 - Gr-Growth
 - Multidimensional association rule mining

References

- **Must read**

- Goethals. “Survey on frequent pattern mining”, published online, 2003. http://www.adrem.ua.ac.be/~goethals/publications/pubs/fpm_survey.pdf
- Karl Aberer. “Data mining: A short intro (Association rules)”, lecture notes, 2008. <http://lsirwww.epfl.ch/courses/dis/2007ws/lecture/week%2013%20Datamining-Association%20rules.pdf>

- **Good to read**

- Han. *Data Mining: Concepts and Techniques*, Morgan Kaufman, 2000
- Agrawal et al. “Mining association rules between sets of items in large databases”. *SIGMOD* 1993, 207-216
- Agrawal & Srikant. “Fast algorithms for mining association rules in large databases”. *VLDB* 1994, pp. 487-499
- Han et al. “Mining frequent patterns without candidate generation”. *SIGMOD* 2000, pp.1–12
- Li et al. “Mining Statistically Important Equivalence Classes and Delta-Discriminative Emerging Patterns”. *KDD* 2007, pp. 430-439

For those who want to go further ...

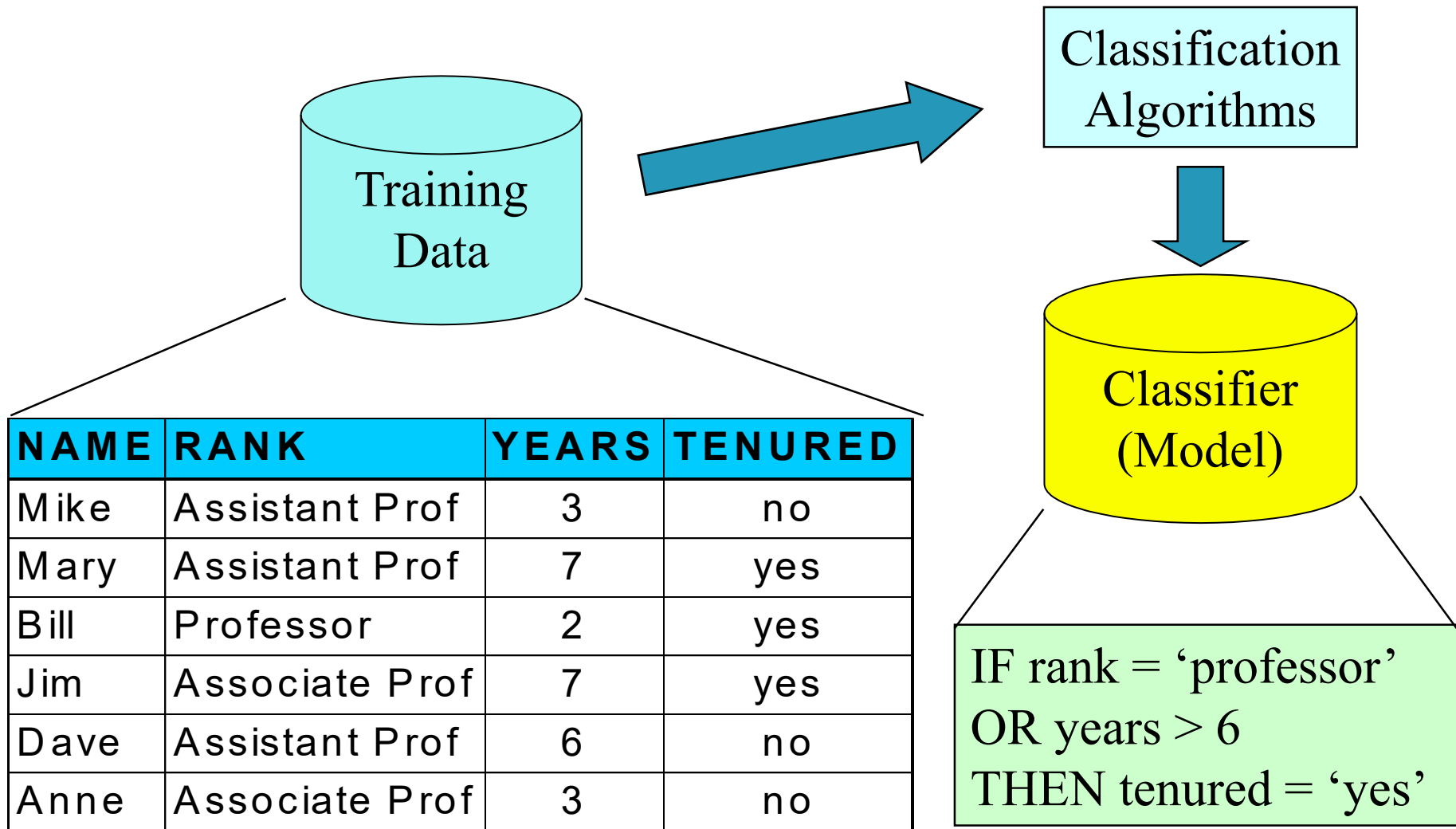
- **Association rule mining has been extended in many interesting directions**
 - Mining multilevel association
 - R. Srikant and R. Agrawal. “Mining generalized association rules”. VLDB 1995
 - Mining multidimensional association
 - Mining quantitative association
 - R. Srikant and R. Agrawal. “Mining quantitative association rules in large relational tables”. SIGMOD 1996
 - Hypothesis exploration, testing, and analysis
 - G. Liu, et al. “Towards Exploratory Hypothesis Testing and Analysis”. ICDE 2011

CLASSIFICATION

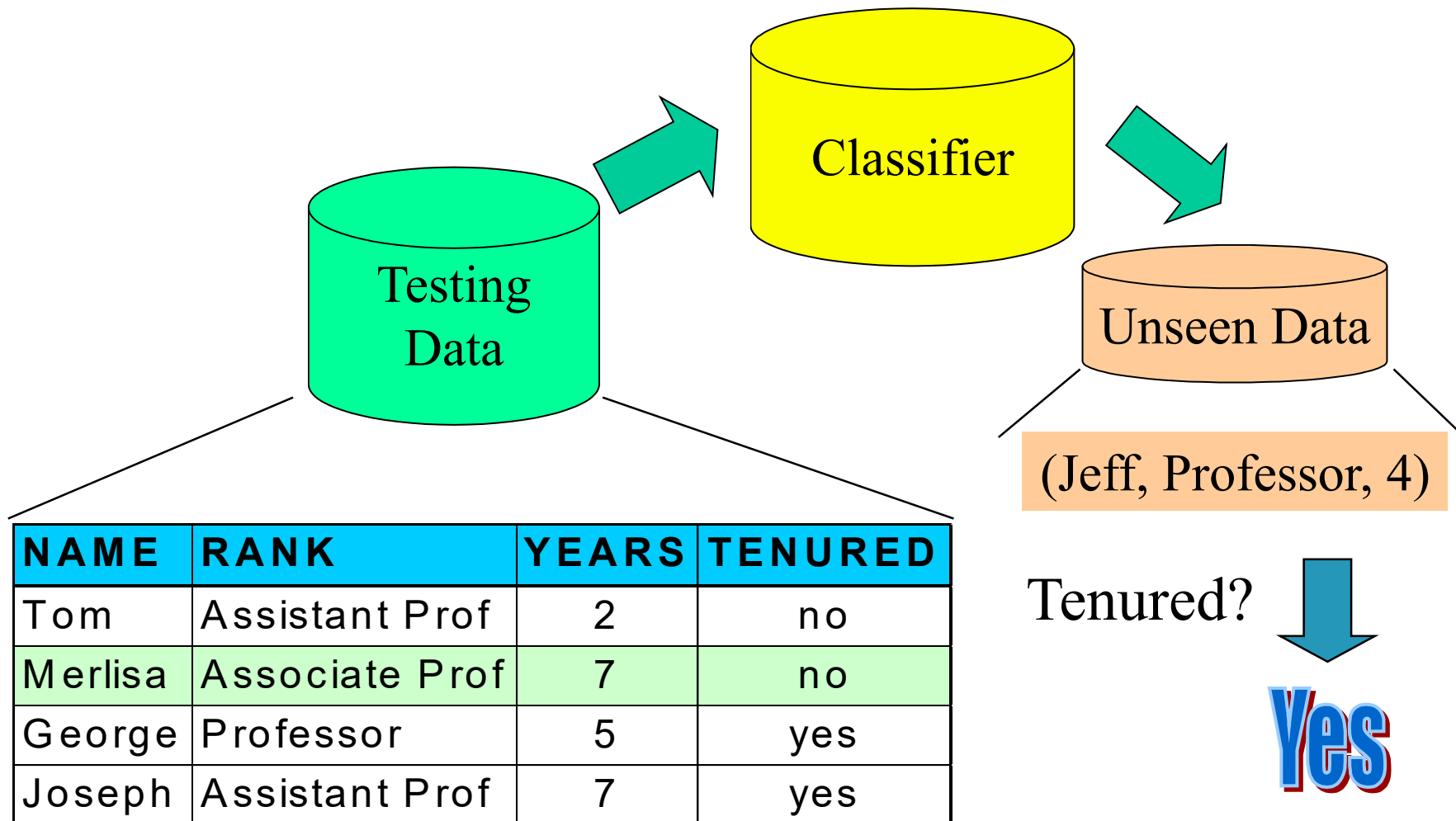
Classification, aka supervised learning

- **Model construction**
 - For describing a set of predetermined classes
 - **The model is represented as classification rules, decision trees, or mathematical formulae**
- **Model usage**
 - For classifying future or unknown objects
 - Estimate accuracy of the model
 - **The known label of test sample is compared with the classification result from the model**
 - If accuracy is acceptable, use the model to classify data tuples whose class labels are unknown

Model construction



Use the model for prediction



Steps of model construction

- **Training data gathering**
- **Feature generation**
 - k-grams, colour, texture, domain know-how, ...
- **Feature selection**
 - Entropy, χ^2 , t-test, domain know-how...
- **Feature integration**
 - SVM, ANN, PCL, CART, C4.5, kNN, ...

You should have already learned this stuff from CS2220. Here is just a quick revision...

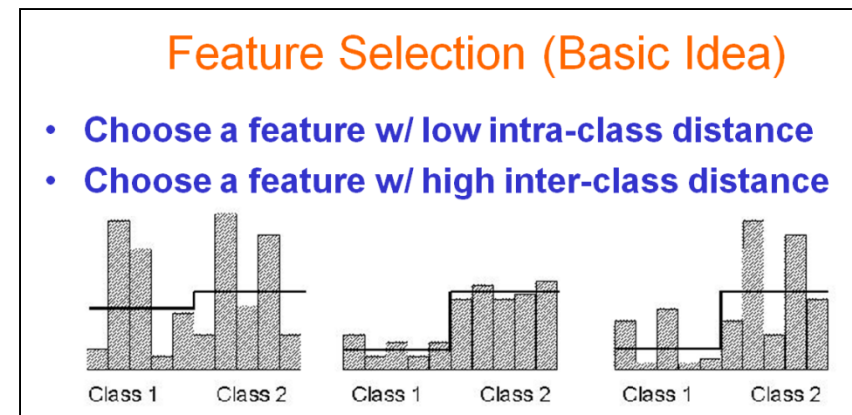
Feature selection

- **Purpose**

- Measure the diff betw two classes, and rank the features according to the degree of the difference
- Get rid of noisy & irrelevant features

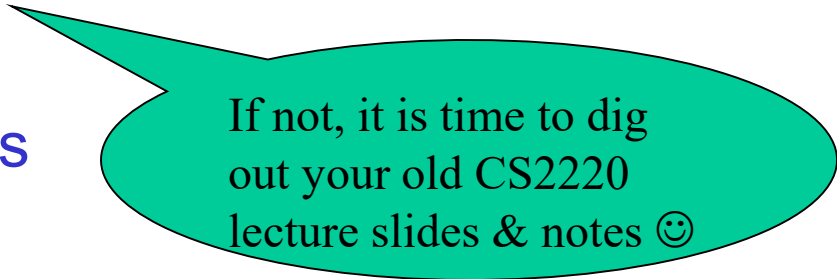
- **Approaches**

- Statistical tests
 - E.g., t-test, χ^2 -test
- Information theory
 - E.g., Gini index, entropy, info gain



Feature integration

- I hope you still remember the various classifiers you came across in CS2220
 - Decision Trees
 - Decision Trees Ensembles
 - E.g., Bagging
 - K-Nearest Neighbour
 - Support Vector Machines
 - Bayesian Approach
 - Hidden Markov Models



If not, it is time to dig out your old CS2220 lecture slides & notes 😊

Measures of classifier performance

	predicted as positive	predicted as negative
positive	TP	FN
negative	FP	TN

$$\begin{aligned}
 \text{Accuracy} &= \frac{\# \text{ correct predictions}}{\# \text{ predictions}} \\
 &= \frac{TP + TN}{TP + TN + FP + FN}
 \end{aligned}$$

$$\text{Sensitivity} = \frac{\# \text{ correct +ve predictions}}{\# \text{ +ve}}$$

$$= \frac{TP}{TP + FN}$$

$$\text{Specificity} = \frac{\# \text{ correct -ve predictions}}{\# \text{ -ve}}$$

$$= \frac{TN}{TN + FP}$$

$$\text{Precision} = \frac{\# \text{ correct +ve predictions}}{\# \text{ +ve predictions}}$$

$$= \frac{TP}{TP + FP}$$

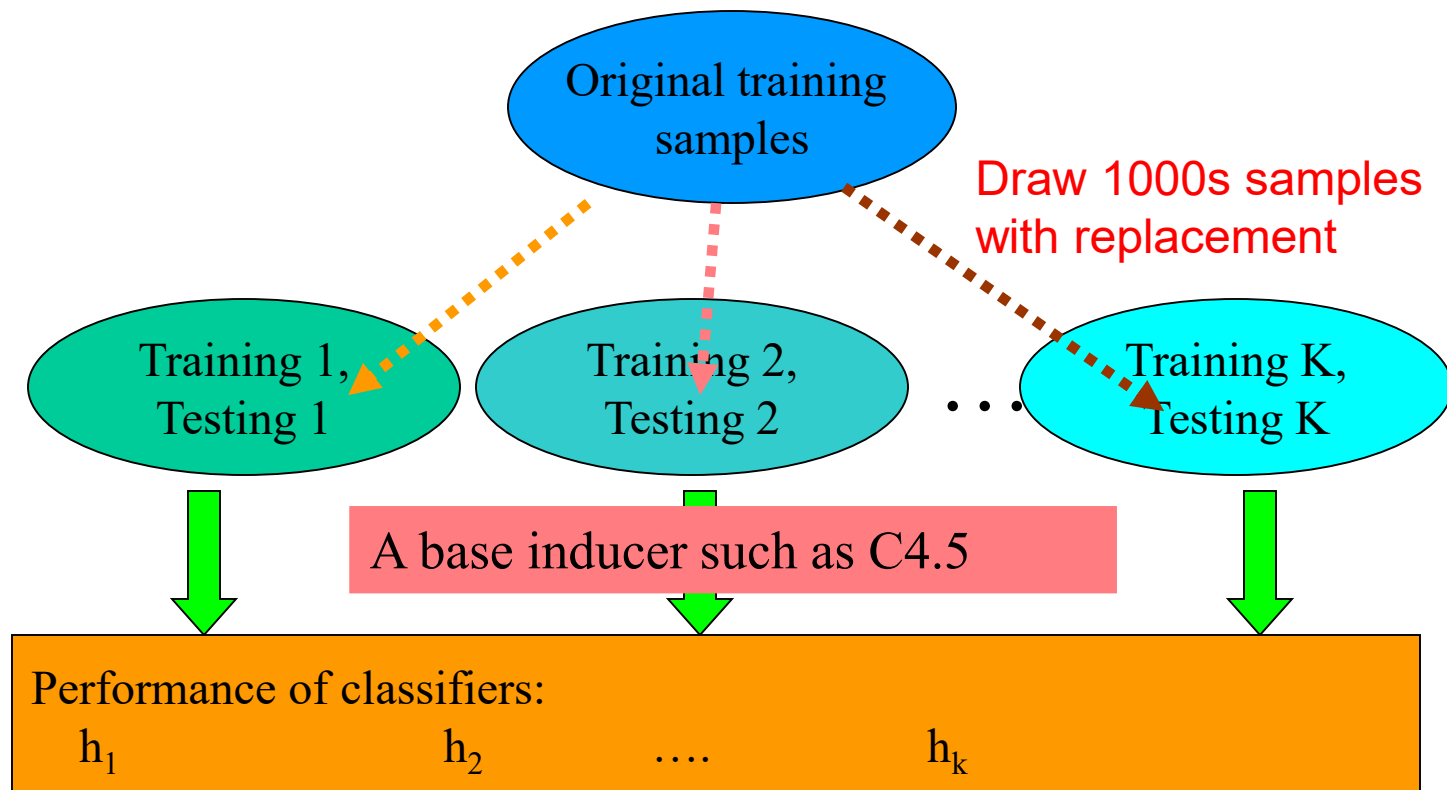
Time for Exercise #2

- **Accuracy is not a good measure if the (class) distribution of test data has bias**
- **Sensitivity (SE), specificity (SP), & precision (PPV) are better; but they must be used together**
- **How to combine SE, SP, and/or PPV?**

Evaluation

- **Accuracy, sensitivity, precision, etc of a classifier are generally evaluated based on blind test sets**
- **If adequate blind test set is unavailable, evaluate the expected performance of the learning algorithm instead**
 - Sampling and apply Central Limit Theorem (CLT)
 - Cross validation
 - P-value

Evaluation by sampling & CLT



- By CLT, ave accuracy of h_1, h_2, \dots, h_k is the expected accuracy of the classifier produced by the based inducer on the original samples

Evaluation by cross validation

1.Test	2.Train	3.Train	4.Train	5.Train
--------	---------	---------	---------	---------

1.Train	2.Test	3.Train	4.Train	5.Train
---------	--------	---------	---------	---------

1.Train	2.Train	3.Test	4.Train	5.Train
---------	---------	--------	---------	---------

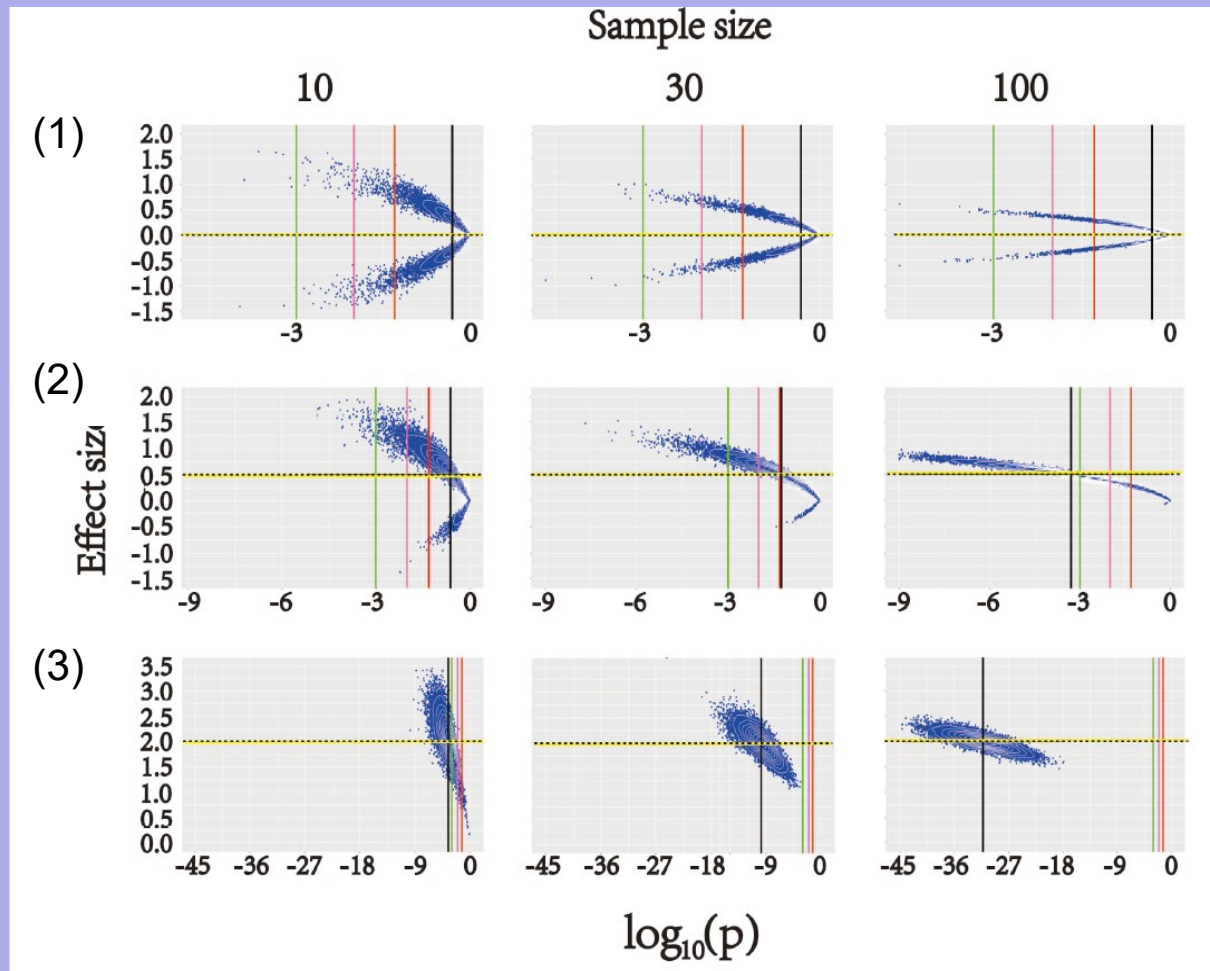
1.Train	2.Train	3.Train	4.Test	5.Train
---------	---------	---------	--------	---------

1.Train	2.Train	3.Train	4.Train	5.Test
---------	---------	---------	---------	--------

- Divide samples into k roughly equal parts
- Each part has similar proportion of samples from different classes
- Use each part to test other parts
- Total up the accuracy

Scenario	Distribution		Mean		Standard deviation		Sample size		
	A	B	A	B	A	B			
(1)	Normal	Normal	0	0	1	1	10	30	100
(2)	Normal	Normal	0	0.5	1	1	10	30	100
(3)	Normal	Normal	0	2	1	1	10	30	100

Time for Exercise #3



Time for Exercise #3

- You are analyzing a gene expression dataset with two phenotypes, say cancer vs normal, to identify genes that behave differently between the two phenotypes
- To help you make various decisions, you decide to do some simulations under three scenarios. In scenario (1), a gene is simulated to have the same expression distribution---specifically, $N(0,1)$, the normal distribution with mean 0 and standard deviation 1---in classes A and B. In scenario (2), the gene has the distribution $N(0,1)$ in class A and $N(0.5,1)$ in B. In scenario (3), the gene is has the distribution $N(0,1)$ in class A and $N(2,1)$ in B. The scenarios are simulated 10,000 times and the (observed) effect size---defined as the (observed) mean of class A minus the (observed) mean of class B---and (observed) t-statistic p-value are computed for each simulation and plotted in slide #74
- Discuss the threshold to use on the observed effect size when sample size is 10
- Discuss the threshold to use on the observed effect size when sample size is 100
- Discuss the threshold to use on the observed p-value when sample size is 10
- Discuss the threshold to use on the observed p-value when sample size is 100

References

- **Must read**

- Li et al. “Data Mining Techniques for the Practical Bioinformatician”, *The Practical Bioinformatician*, Chapter 3, pp. 35-70, WSPC, 2004
- Karl Aberer. “Data mining: A short intro (Classifiers)”, lecture notes, 2008. <http://lsirwww.epfl.ch/courses/dis/2007ws/lecture/week%2014%20Datamining-Clustering-Classification-Wrap-up.pdf>

- **Good to read**

- Swets. “Measuring the accuracy of diagnostic systems”, *Science* 240:1285--1293, 1988

Acknowledgements

- The set of slides on clustering was mostly adapted from Jinyan Li
- The set of slides on association rule mining was mostly adapted from Albert Puig