CS4220: Knowledge Discovery Methods for Bioinformatics

Unit 2: Essence of Data Mining

Limsoon Wong

NUS
National University
of Singapore

Lecture Outline

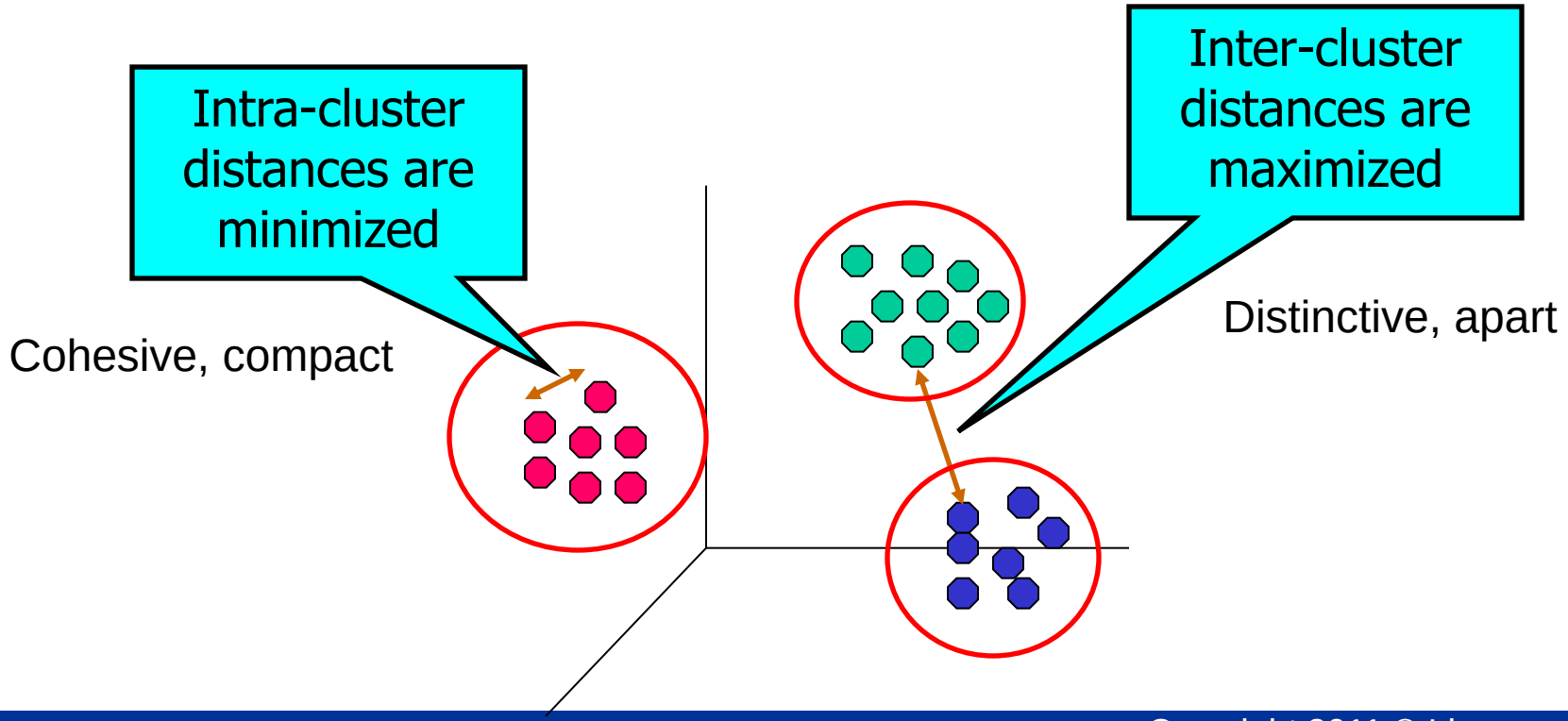
- **Clustering, aka unsupervised learning**
- **Association rule mining**
- **Classification, aka supervised learning**
- **Class-imbalance learning**

Clustering

NUS
National University
of Singapore

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

Typical 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

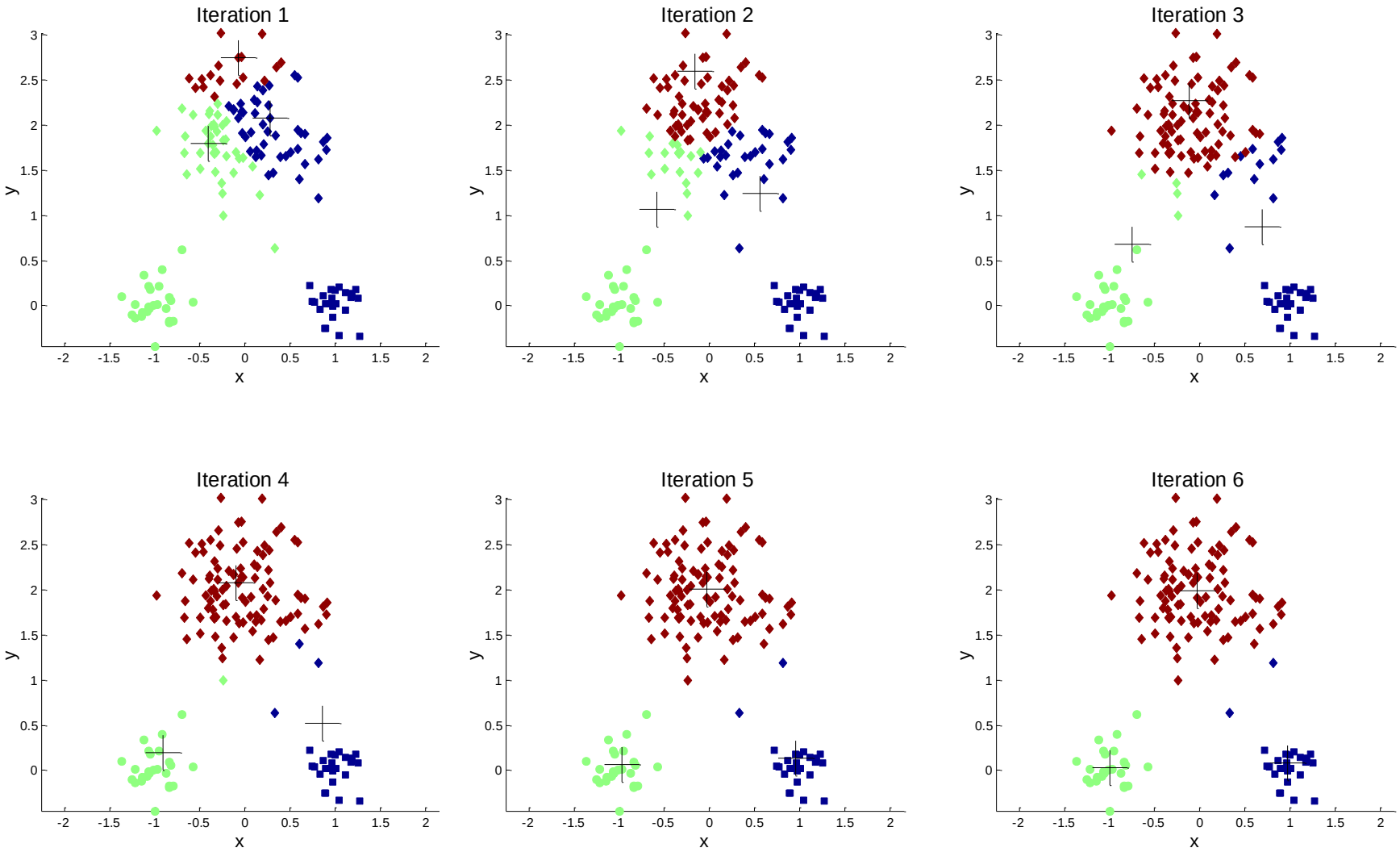
- **Each cluster has a centroid**
- **Each point is assigned to a cluster based on closest centroid**
- **# of clusters, K, must be specified**

Select K points as the initial centroids.
repeat
  Form K clusters by assigning all points to the closest centroid.
  Recompute the centroid of each cluster.
until The centroids don't change

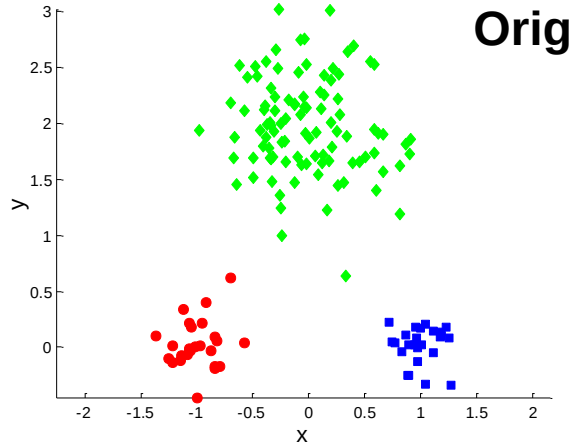
More Details of K-Means Clustering

- **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

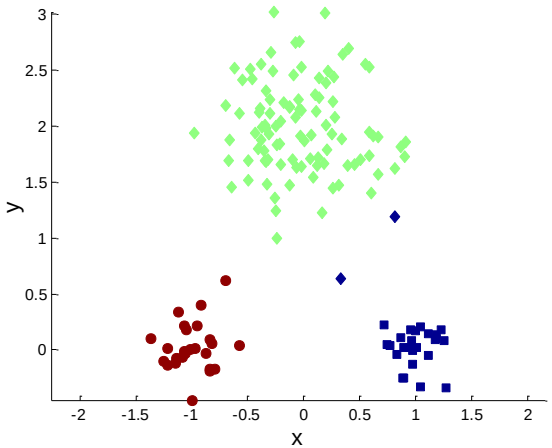
Example Iterations by K-Means



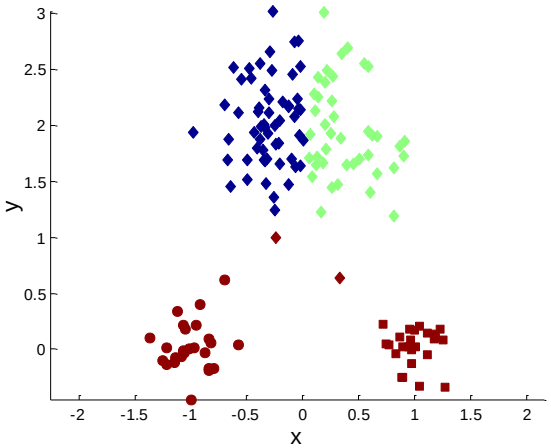
Two Different K-means Clusterings



Original Points



Optimal Clustering



Sub-optimal Clustering

Evaluating K-means Clusters

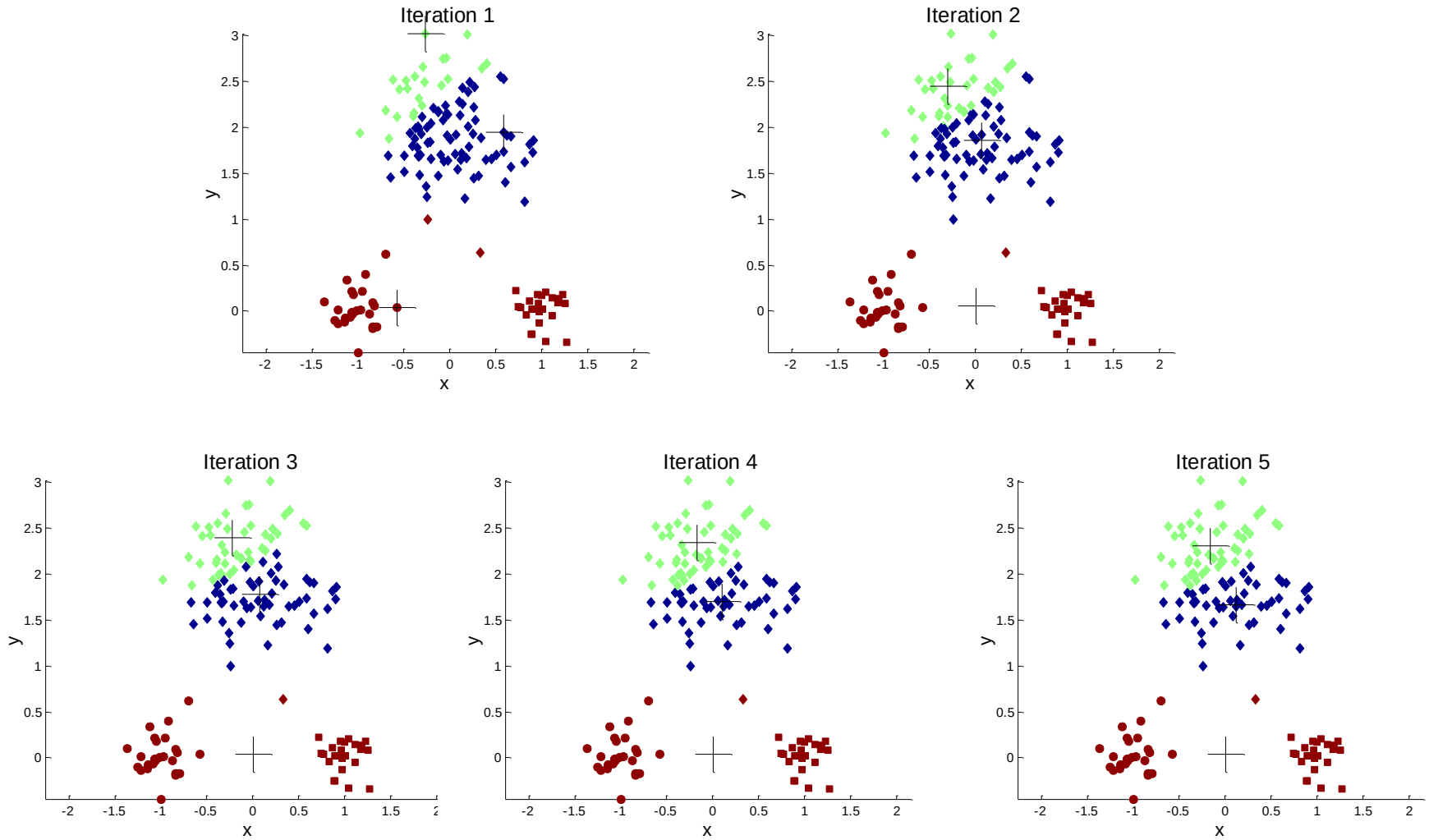
- **Sum of Squared Error (SSE) is commonly used**
  - Error of a point is its distance to nearest centroid
  - Square these errors and sum them to get SSE

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

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 Choosing Initial Centroids



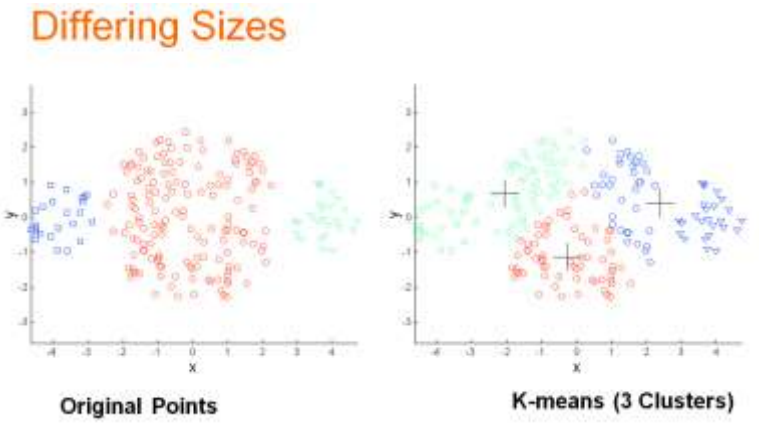
Solutions to Initial Centroid Problem

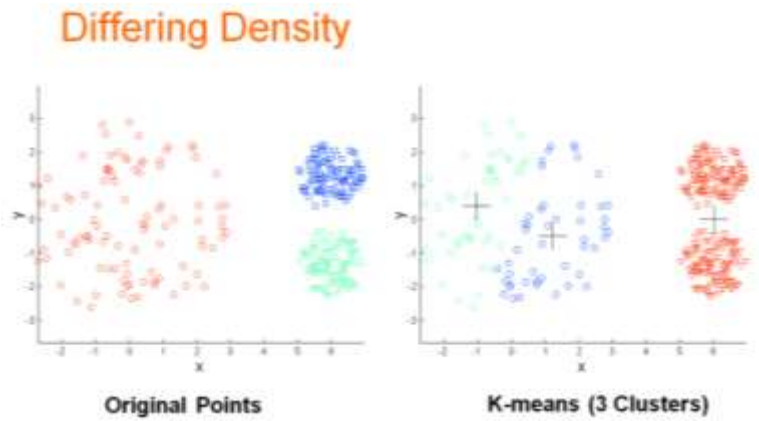
- **Multiple runs**
  - Helps, but probability is not on your side
- **Use hierarchical clustering to determine initial centroids**
- **Select >k initial centroids and then select the most widely separated among these initial centroids**
- **Use more advanced algos, like "Bisecting K-Means", that are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

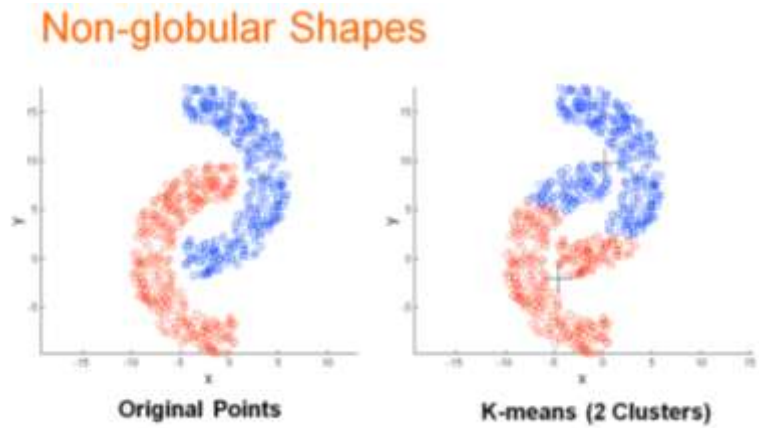
Limitations of K-means

- **Has problems when clusters are of differing**
  - Sizes
  - Densities
  - Non-globular shapes

- **Also has problems when data contain outliers**

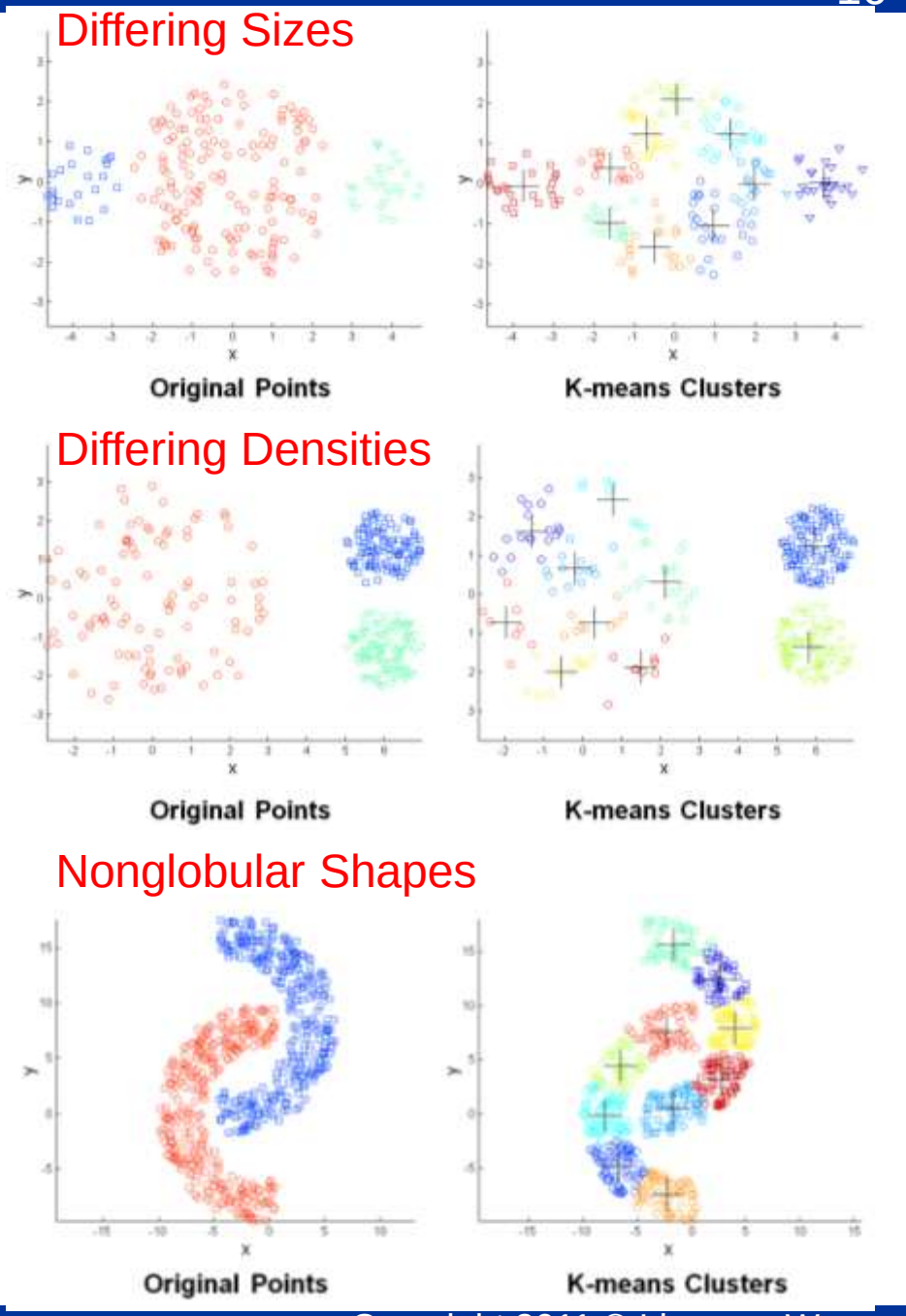






Overcoming K-means' Limitations

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



Hierachical 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.**

Distance Matrix

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

n features (order of 1000)

	gene1	gene2	gene3	gene4	…	gene_n
m tuples	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) = dist(pi, pj)

- **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
```

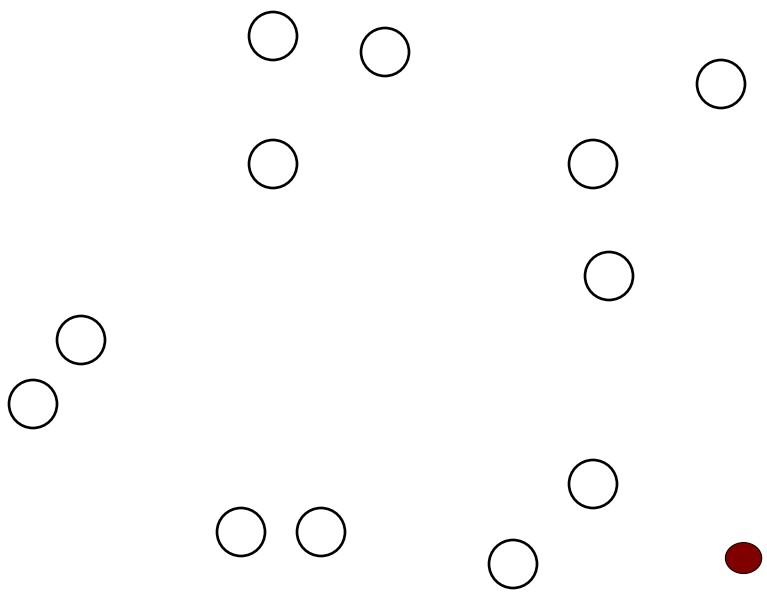
- **Key is computing proximity of two clusters**
  - Diff approaches to defining distance betw clusters distinguish the diff algos

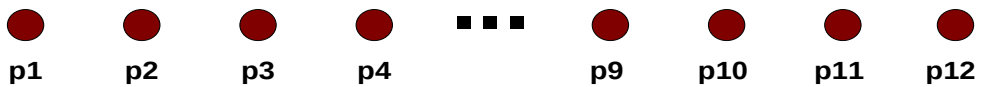
Starting Situation

- **Start with clusters of individual points and a proximity matrix**

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

Proximity Matrix



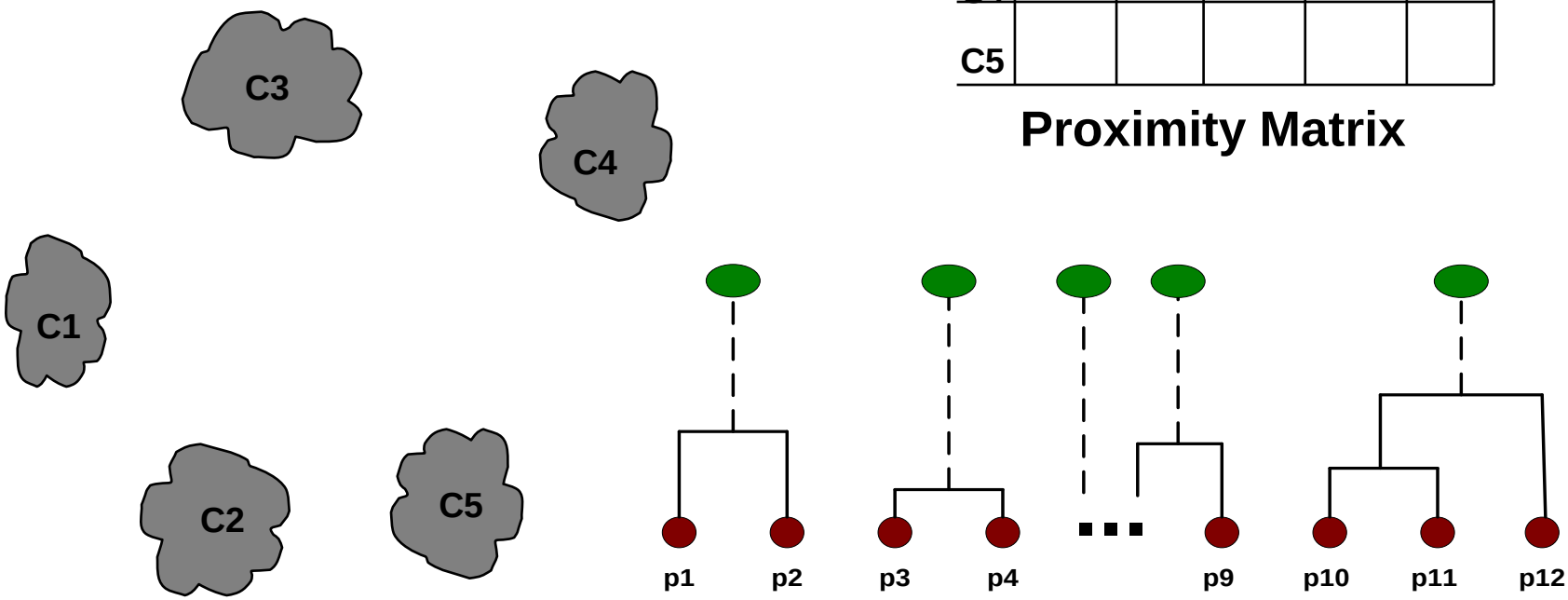


Intermediate Situation

- **After some merging steps, we have some clusters**

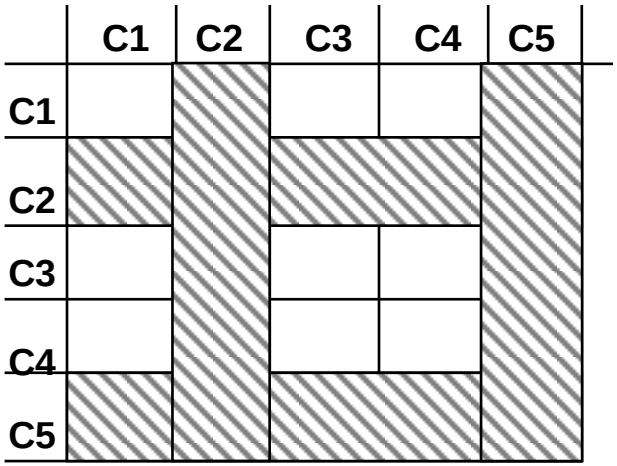
	C1	C2	C3	C4	C5
C1					
C2					
C3					
C4					
C5					

Proximity Matrix

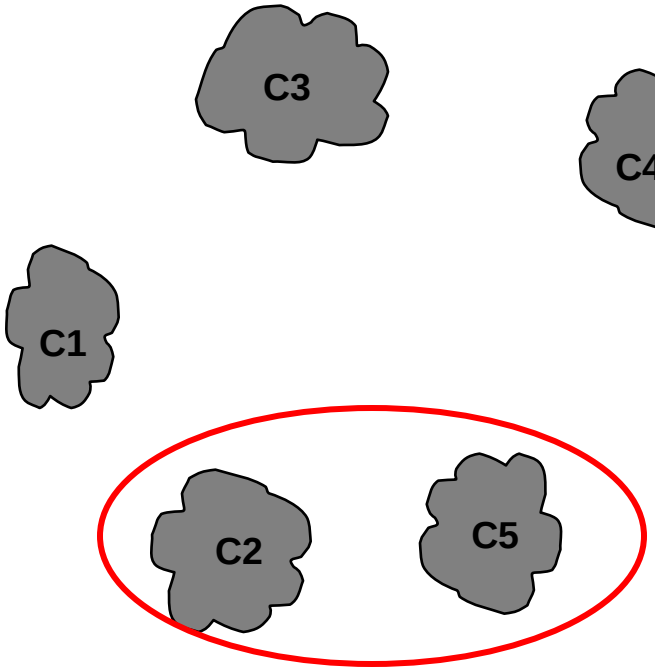


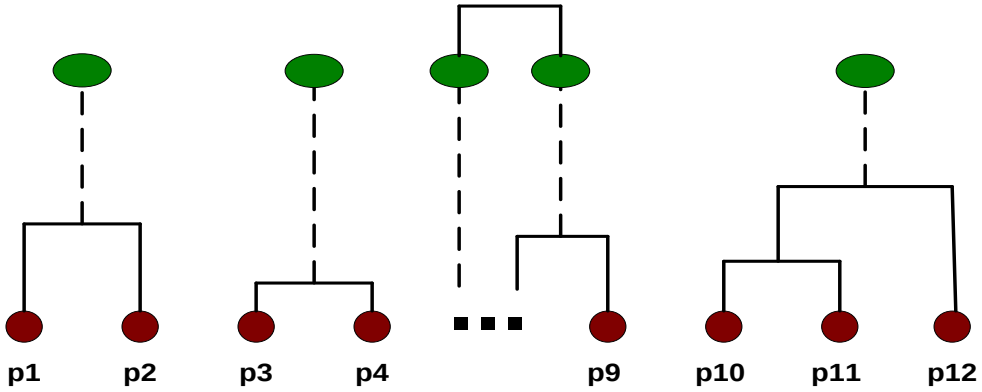
Intermediate Situation

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

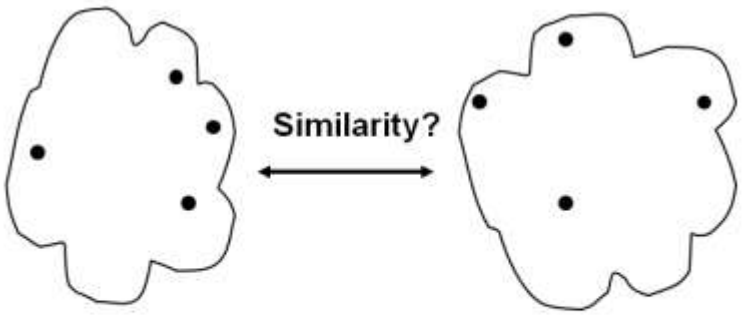


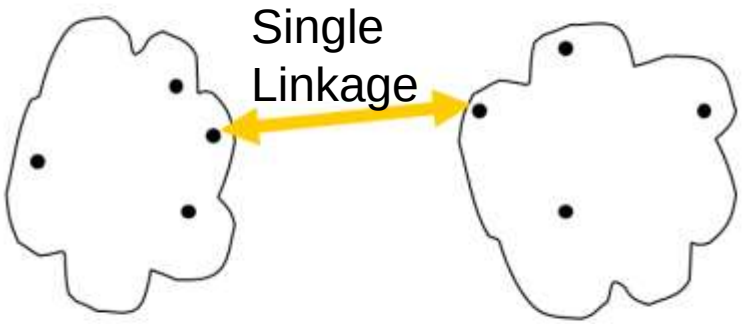
Proximity Matrix

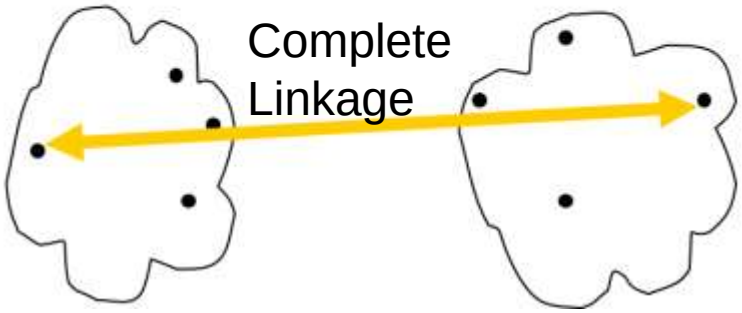


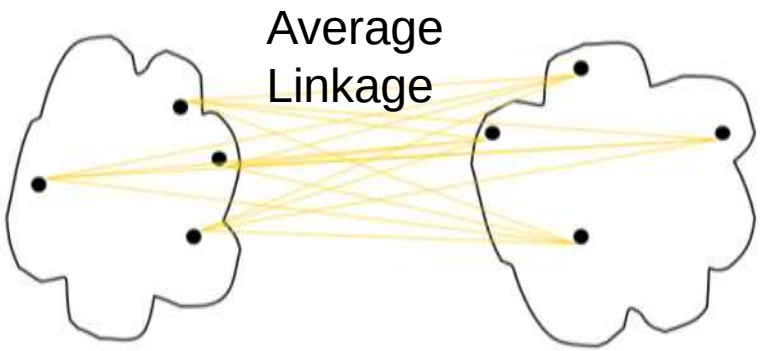


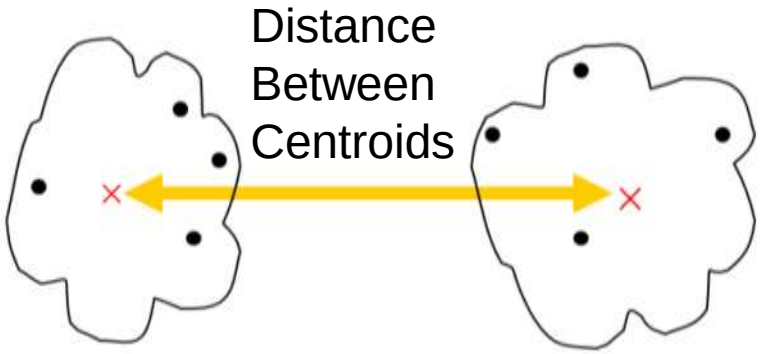
Defining Inter-Cluster Similarity





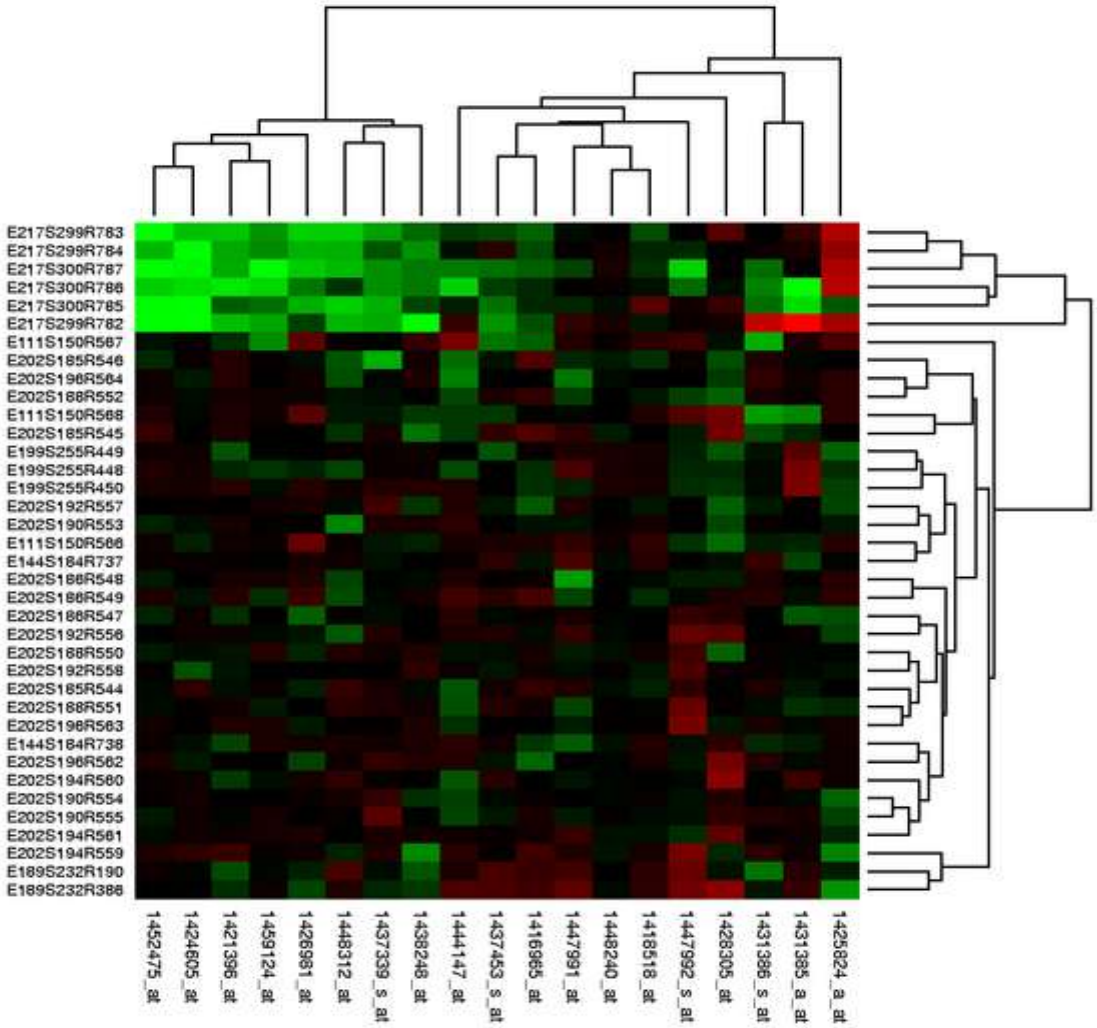






- **Other methods use an objective function**
  - Ward's method uses squared error

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 dendogram at the proper level
- **They may correspond to meaningful taxonomies**
  - Example in biological sciences (e.g., animal kingdom, phylogeny reconstruction, …)

Divisive Hierarchical Clustering

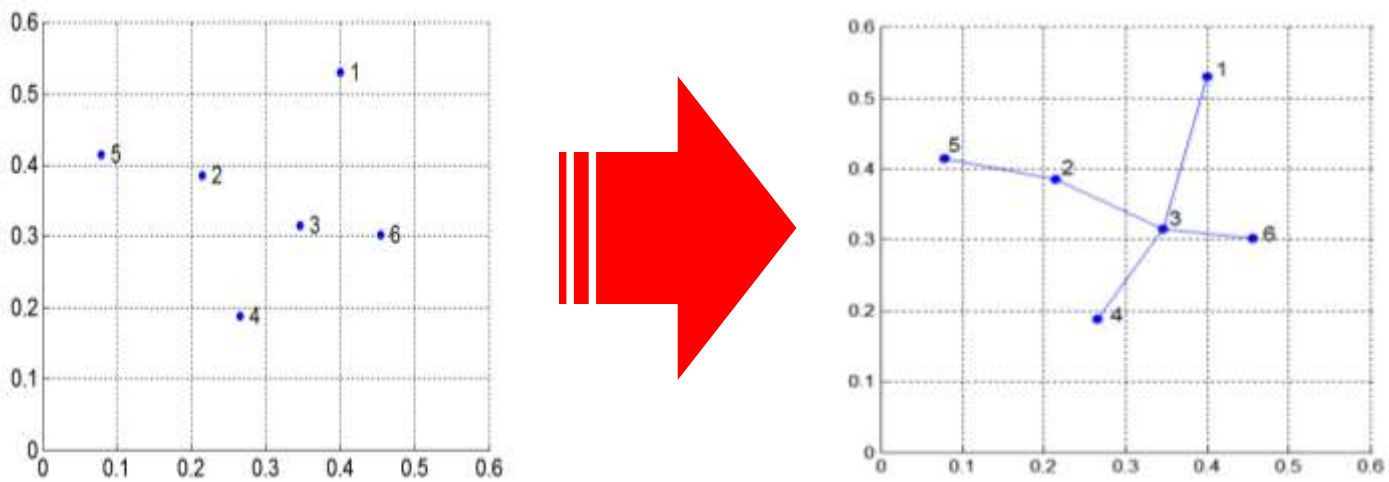
- **Start with one, all-inclusive cluster**
- **At each step, split a cluster until each cluster contains a point (or there are k clusters)**

Algorithm 7.5 MST Divisive Hierarchical Clustering Algorithm

```
Compute a minimum spanning tree for the proximity graph.
repeat
    Create a new cluster by breaking the link corresponding to the largest distance
    (smallest similarity).
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 the subspaces**

Bi- or Co-Clustering

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

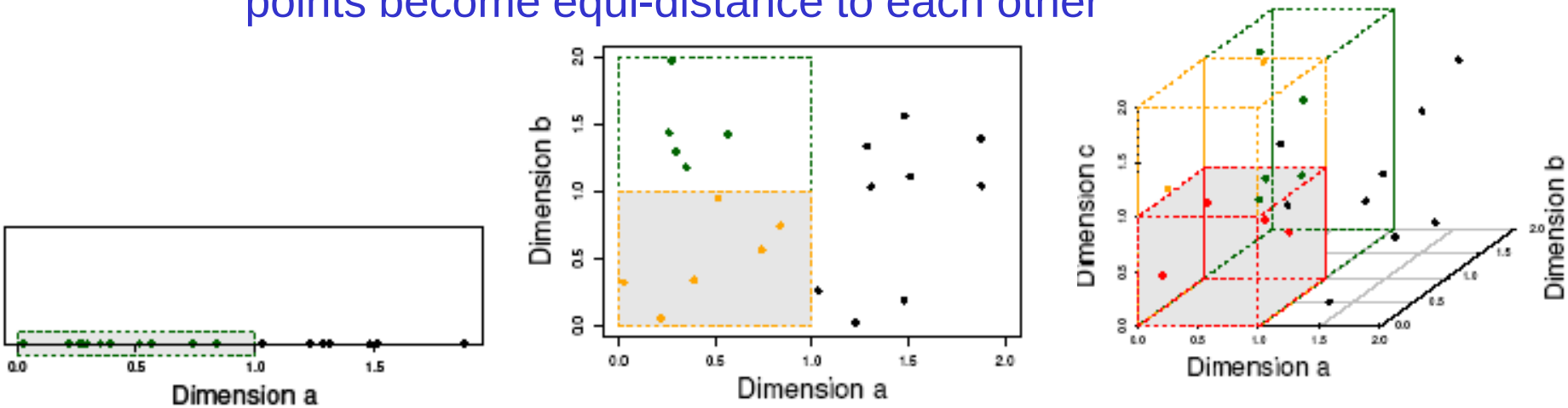
High-Dimensional Data

- **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

The 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

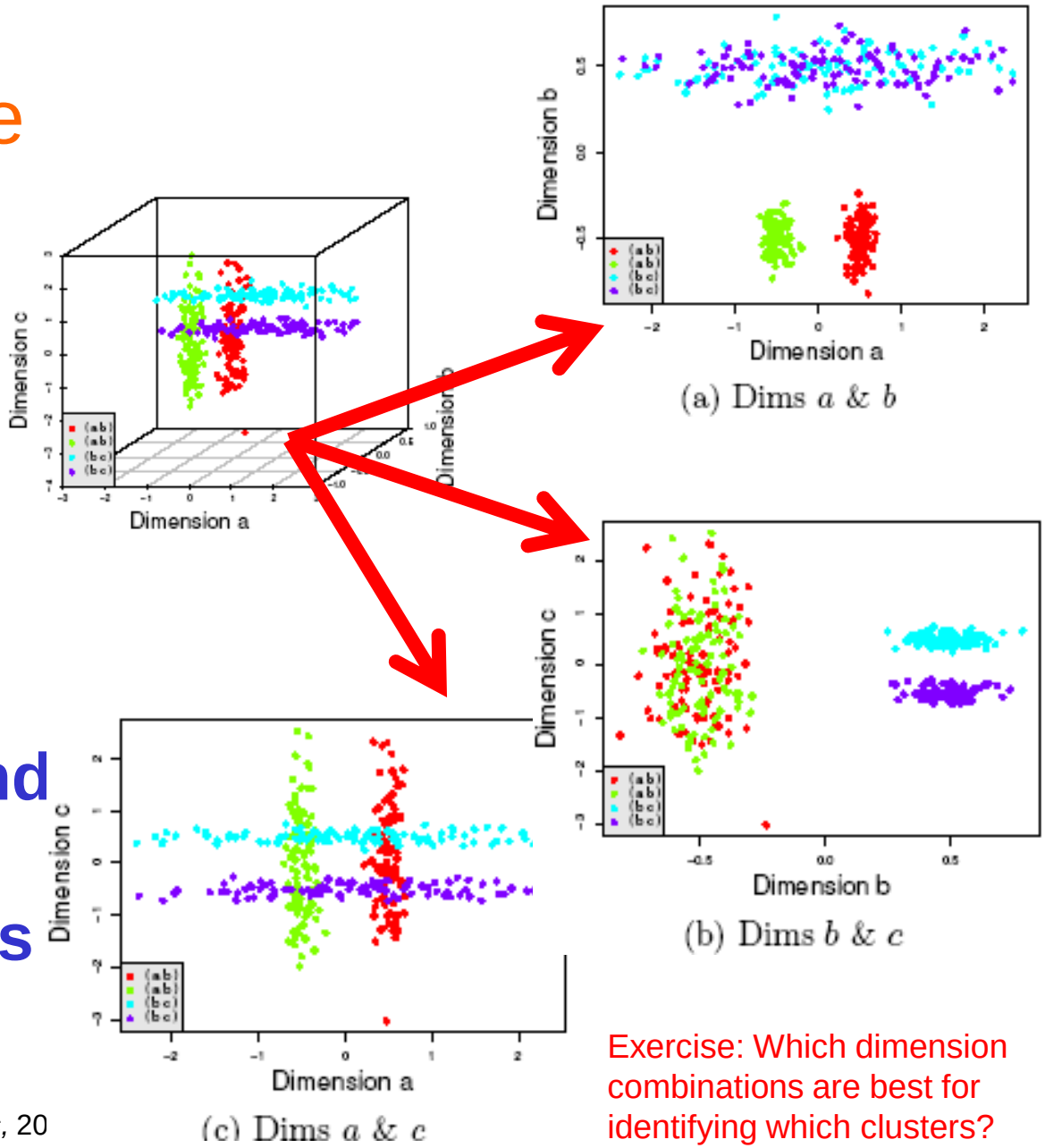


(a) 11 Objects in One Unit Bin (b) 6 Objects in One Unit Bin (c) 4 Objects in One Unit Bin

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**

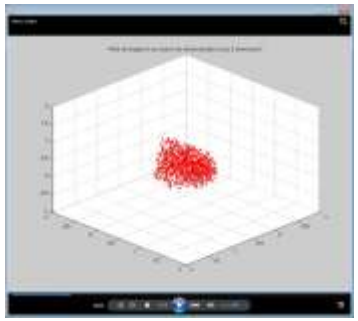


Exercise: Which dimension combinations are best for identifying which clusters?

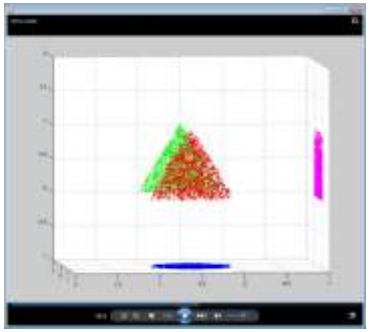
Image credit: Parsons et al. *KDD Explorations*, 20

However, inspect your subspace clusters carefully!

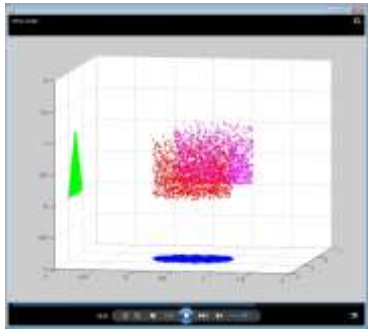
A cloud of points in 3D



In 2D XZ we see …



In 2D YZ we see…



In 2D XY we see …

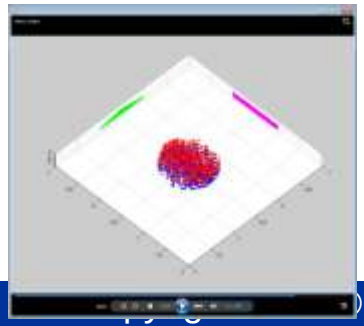


Image credit: Eamonn Keogh

CLIQUE (Clustering In QUEst)

- **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

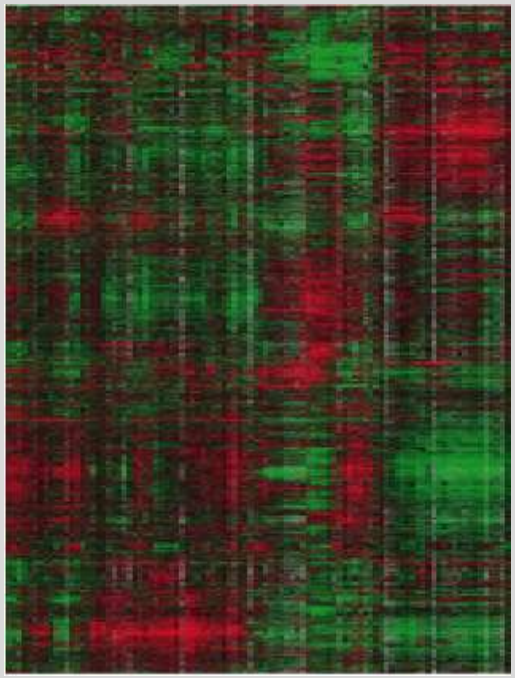
- **Partition the data space**
- **Identify subspaces that contain clusters**
  - Use the "Apriori Principle"
    - Find dense units in all subspaces
    - Form connected dense units in all subspaces
- **Generate minimal description for the clusters**
  - Determine maximal regions that cover a cluster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

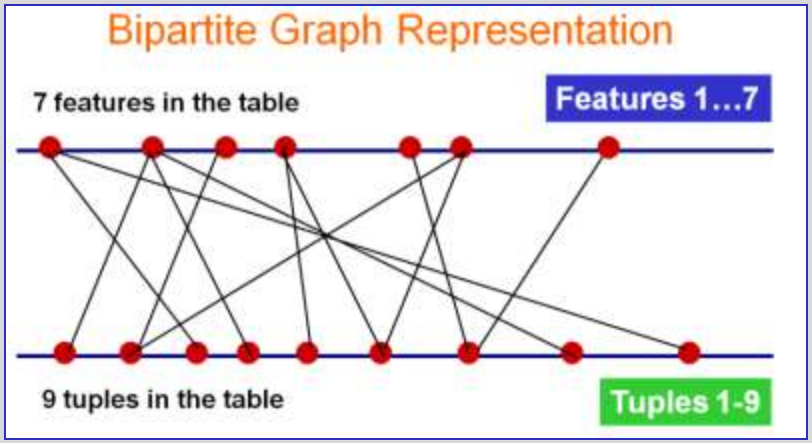
Biclusters = small boxes of homogeneity

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

NUS
National University
of Singapore

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, …, t_n} is a set of trans**
- **Each trans t_k is an itemset I = {i_1, …, 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
  - σ({bread, peanut-butter}) = 3
- **Support, s**
  - Fraction of transactions containing that itemset
  - s({bread, peanut-butter}) = 3/5

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

- **Frequent itemset**
  - An itemset whose support ≥ 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 ⇒ peanut-butter	0.60	0.75
peanut-butter ⇒ bread	0.60	1.00
beer ⇒ bread	0.20	0.50
peanut-butter ⇒ jelly	0.20	0.33
jelly ⇒ peanut-butter	0.20	1.00
jelly ⇒ milk	0.00	0.00

Source: A. Puig

Apriori

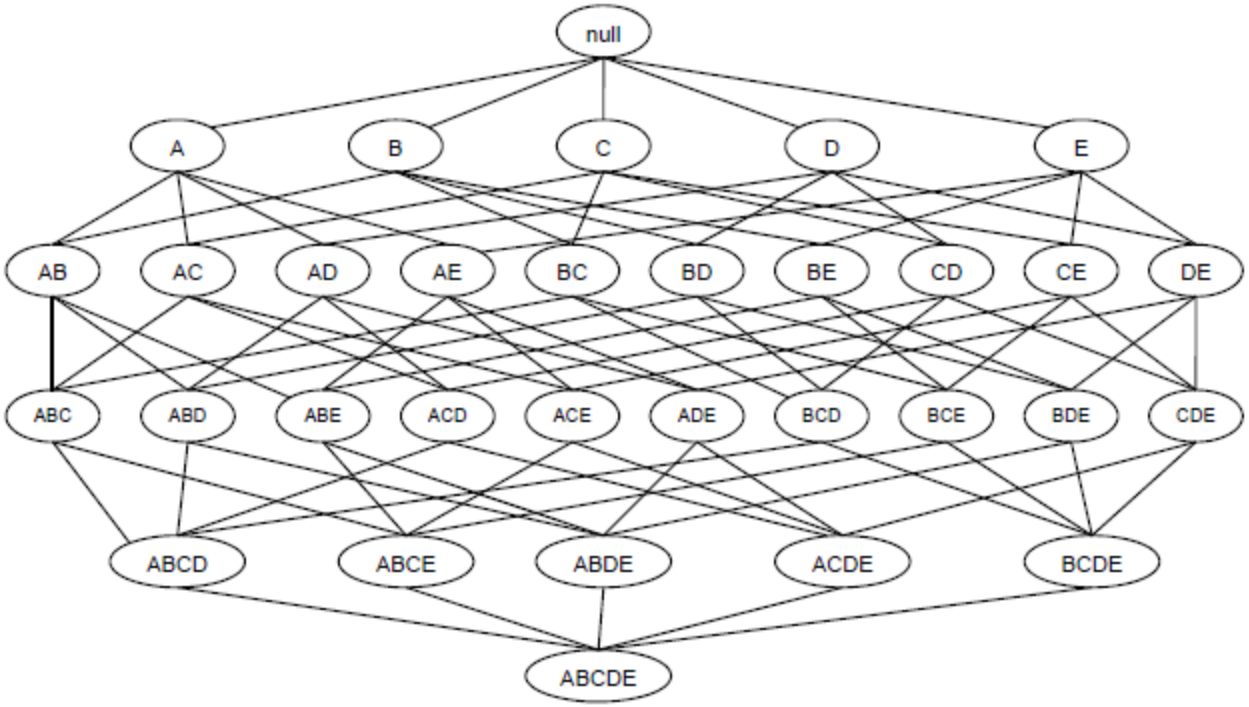
- **Apriori is the classic assoc rule mining algo**
  - Agrawal & Srikant. "Fast algorithms for mining association rules in large databases". *VLDB 1994*, pp. 487-499

- **Mines assoc rules in two steps**
  1. Generate all freq itemsets with support ≥ minsup
  2. Generate assoc rules using these freq itemsets

Let's work on Step 1 first…

Step 1 of Apriori:

Generate freq itemsets with support ≥ 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:**

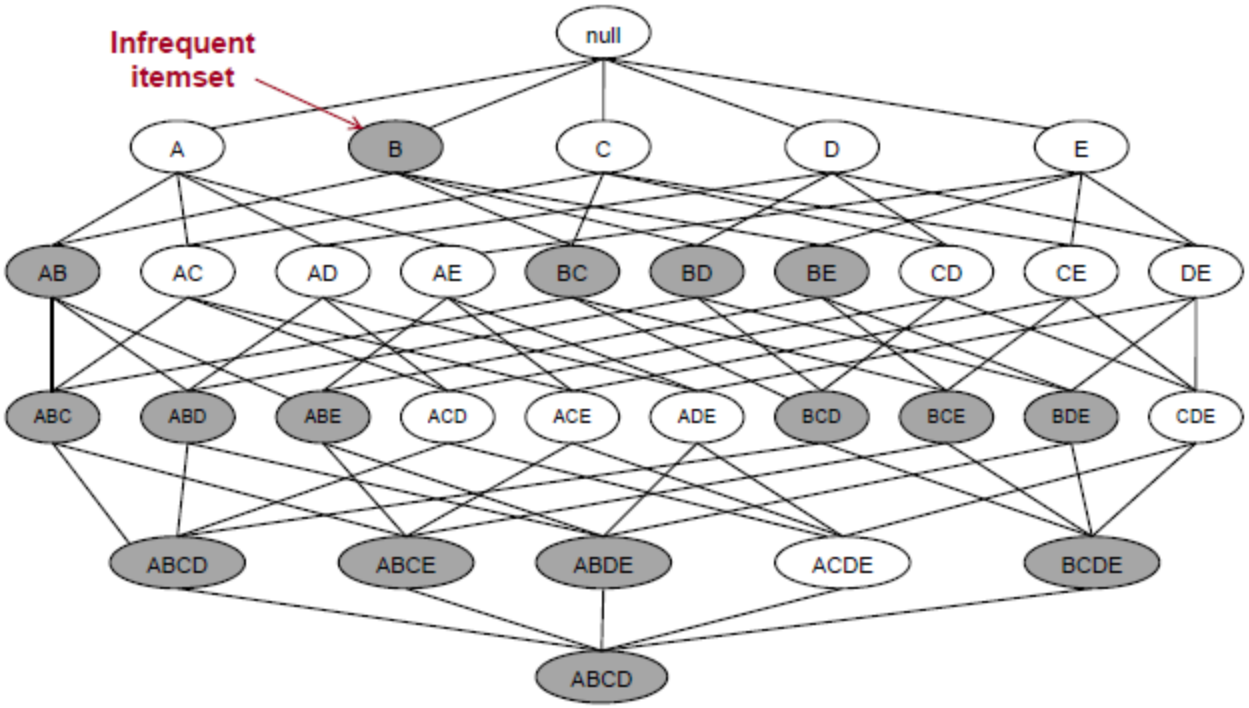
Any subset of a frequent itemset is frequent

⇒ **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 support ≥ minsup



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

Source: A. Puig

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)
    C1 ← init-pass(T);
    F1 ← {f | f ∈ C1, f.count/n ≥ minsup};
    for (k = 2; Fk-1 ≠ ∅; k++) do
        Ck ← candidate-gen(Fk-1);
        for each transaction t ∈ T do
            for each candidate c ∈ Ck do
                if c is contained in t then
                    c.count++;
            end
        end
        Fk ← {c ∈ Ck | c.count/n ≥ minsup}
    end
return F ← ∪k Fk;
```

```
Function candidate-gen(Fk-1)
    Ck ← ∅;
    forall f1, f2 ∈ Fk-1
        with f1 = {i1, … , ik-2, ik-1}
        and f2 = {i1, … , ik-2, i'k-1}
        and ik-1 < i'k-1 do
        c ← {i1, …, ik-1, i'k-1};
        Ck ← Ck ∪ {c};
        for each (k-1)-subset s of c do
            if (s ∉ Fk-1) then
                delete c from Ck;
        end
    end
    return Ck;
```

anti-monotonicity is used here

Example Run of Apriori's Step 1

Database TDB

Tid	Items
10	A, C, D
20	B, C, E
30	A, B, C, E
40	B, E

1st scan →

C_1

Itemset	sup
{A}	2
{B}	3
{C}	3
{D}	1
{E}	3

→ L_1

Itemset	sup
{A}	2
{B}	3
{C}	3
{E}	3

C_2

Itemset
{A, B}
{A, C}
{A, E}
{B, C}
{B, E}
{C, E}

2nd scan →

C_2

Itemset	sup
{A, B}	1
{A, C}	2
{A, E}	1
{B, C}	2
{B, E}	3
{C, E}	2

→ L_2

Itemset	sup
{A, C}	2
{B, C}	2
{B, E}	3
{C, E}	2

C_3

Itemset
{B, C, E}

3rd scan →

L_3

Itemset	sup
{B, C, E}	2

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 ≥ 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

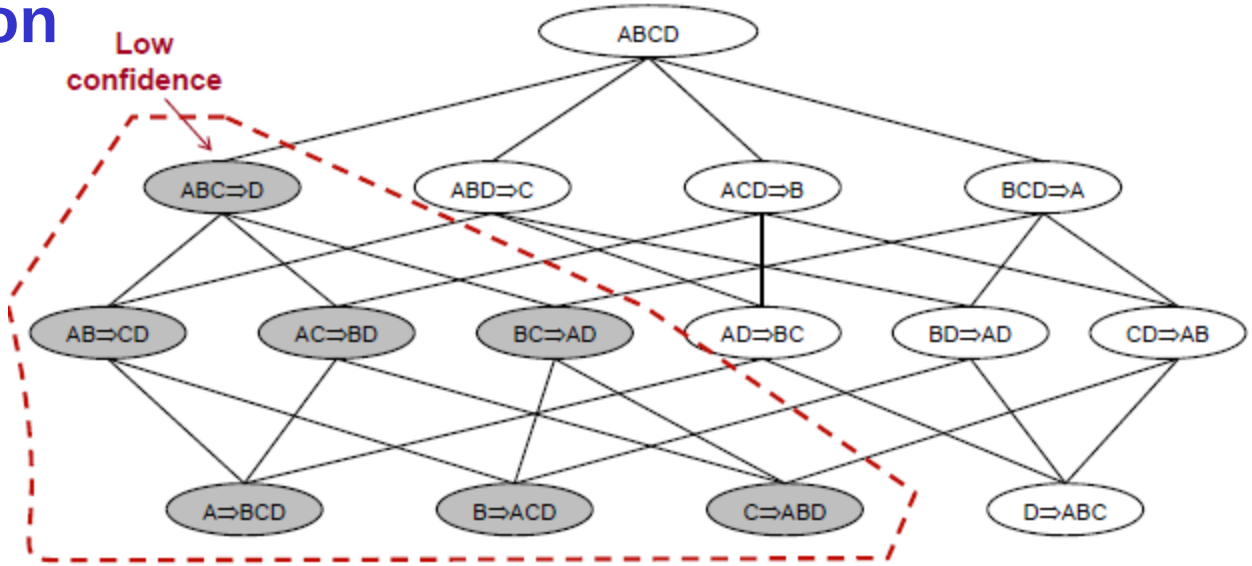
- **Example: L = {A, B, C}**
  - The candidate itemsets are: AB $\Rightarrow$C, AC $\Rightarrow$B, BC $\Rightarrow$A, A $\Rightarrow$BC, B $\Rightarrow$AC, C $\Rightarrow$AB
  - In general, there are $2^{|L|} - 2$ candidates!

Can we be more efficient?

- **Confidence of rules generated from the same itemset does have the anti-monotone property**
  - $c(ABC \Rightarrow D) \geq c(AB \Rightarrow CD) \geq c(A \Rightarrow BCD)$

Exercise: Prove this.

- **We can apply this property to prune rule generation**



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**

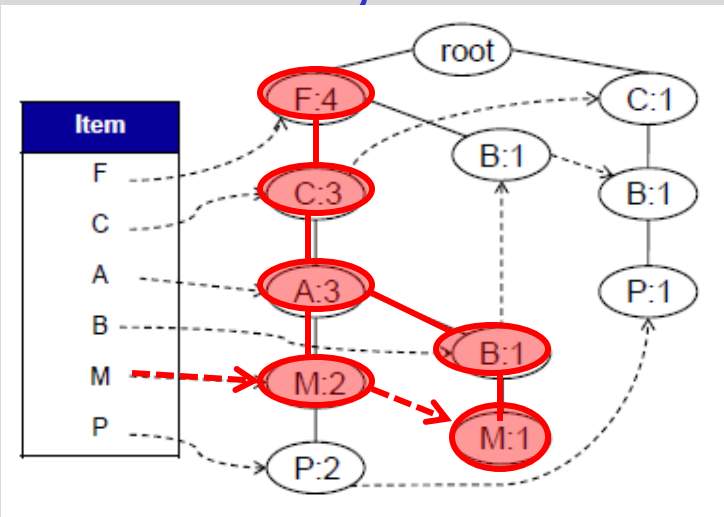
- **Newer methods try to improve on these points**
  - Reduce the # of scans of the db
  - Reduce the # of candidates
  - Count the support of candidates more efficiently

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**

TID	Items	Sorted FIS
1	{F,A,C,D,G,I,M,P}	{F,C,A,M,P}
2	{A,B,C,F,L,M,O}	{F,C,A,B,M}
3	{B,F,H,J,O}	{F,B}
4	{B,C,K,S,P}	{C,B,P}
5	{A,F,C,E,L,P,M,N}	{F,C,A,M,P}



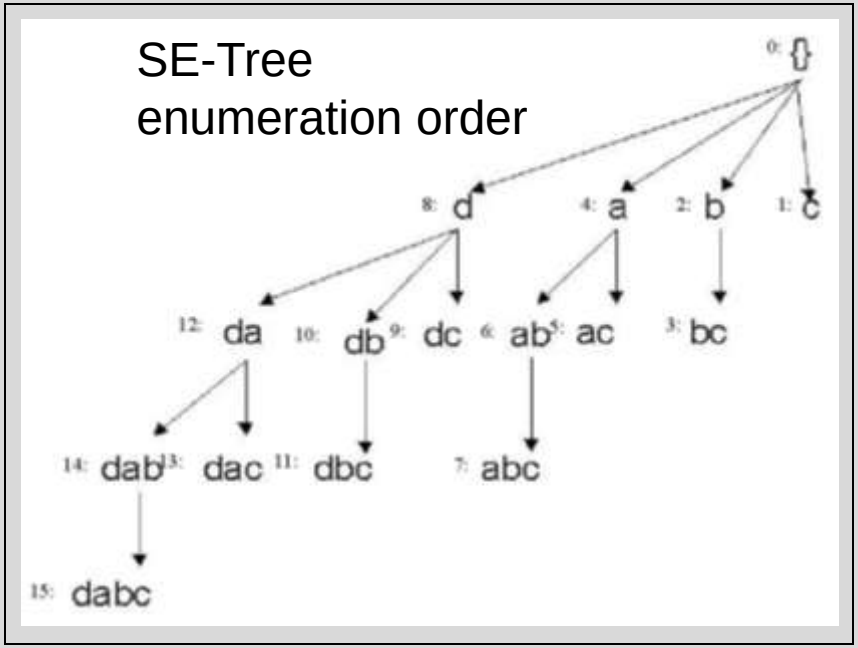
- **Use it for fast support counting**
  - To count the support of an itemset {FCM}, follow the “dotted” links on M. At each node M:*n*, note its support *n* & visit its prefix chain; if FCM is found in the prefix, add *n* to the support

Source: A. Puig

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

Classification

NUS
National University
of Singapore

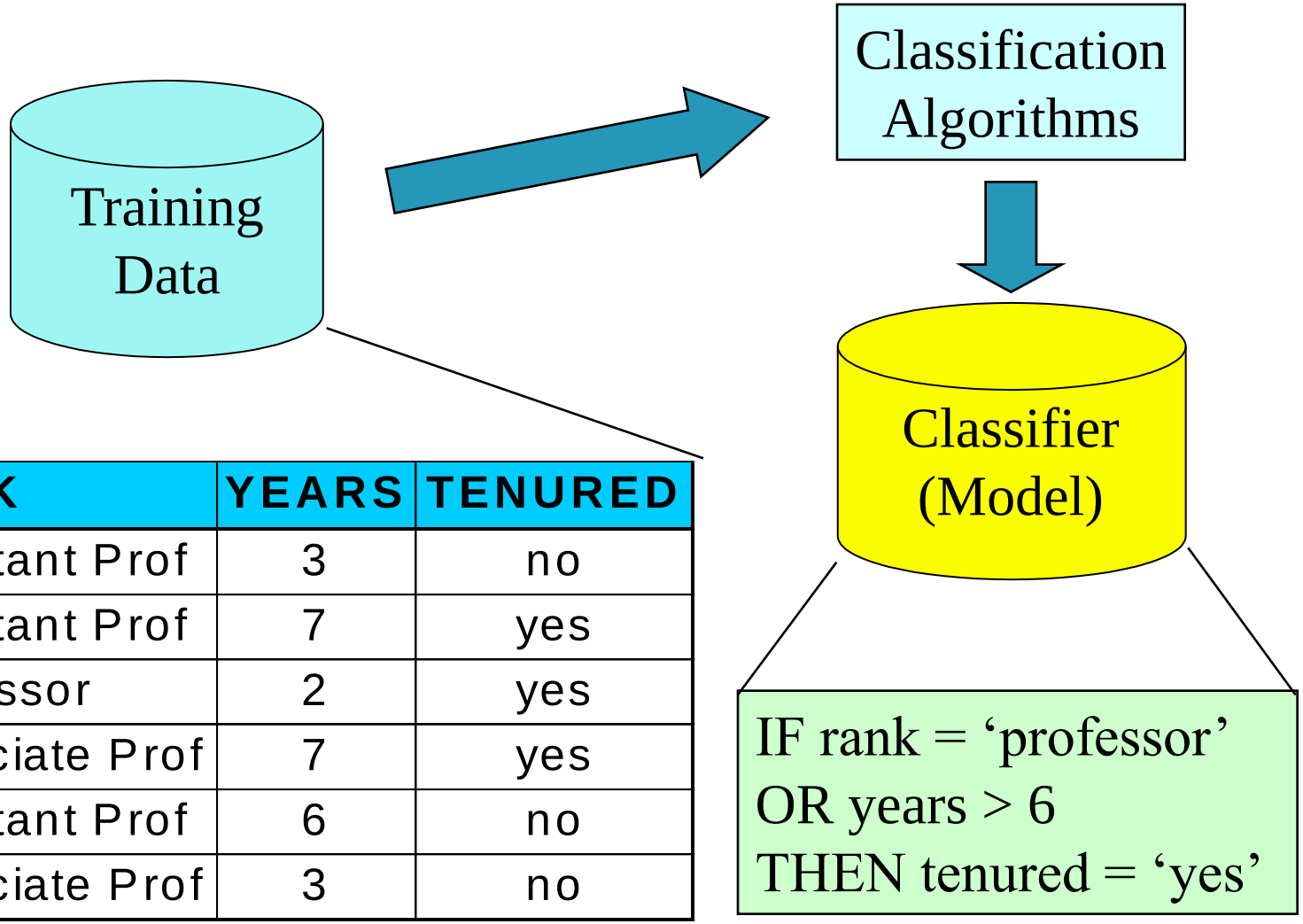
Classification, aka Supervised Learning

- **Model construction**
  - For describe 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

Source: Karl Aberer

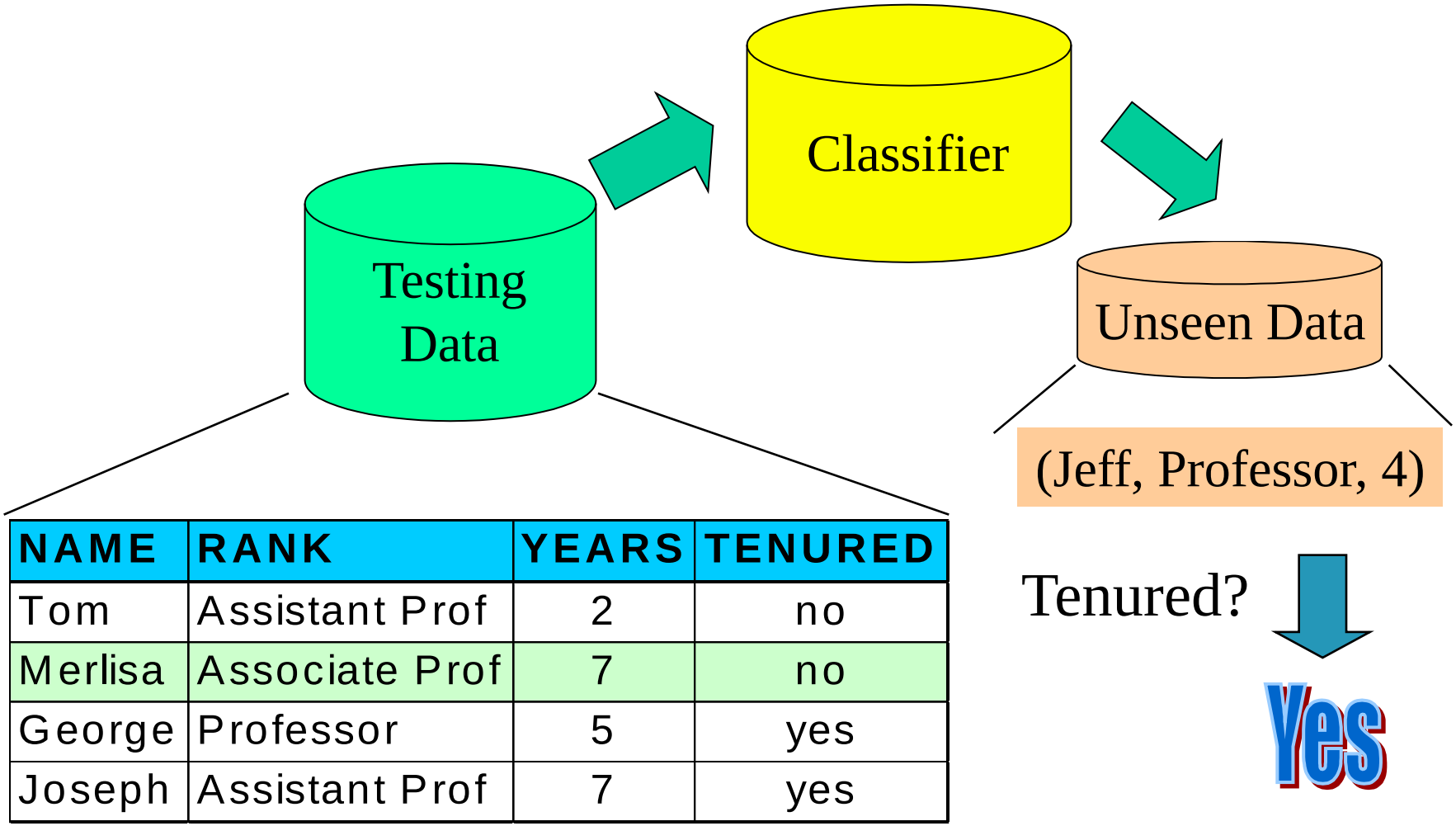
Model Construction



NAME	RANK	YEARS	TENURED
Mike	Assistant Prof	3	no
Mary	Assistant Prof	7	yes
Bill	Professor	2	yes
Jim	Associate Prof	7	yes
Dave	Assistant Prof	6	no
Anne	Associate Prof	3	no

Source: Karl Aberer

Use the Model for Prediction



NAME	RANK	YEARS	TENURED
Tom	Assistant Prof	2	no
Merlisa	Associate Prof	7	no
George	Professor	5	yes
Joseph	Assistant Prof	7	yes

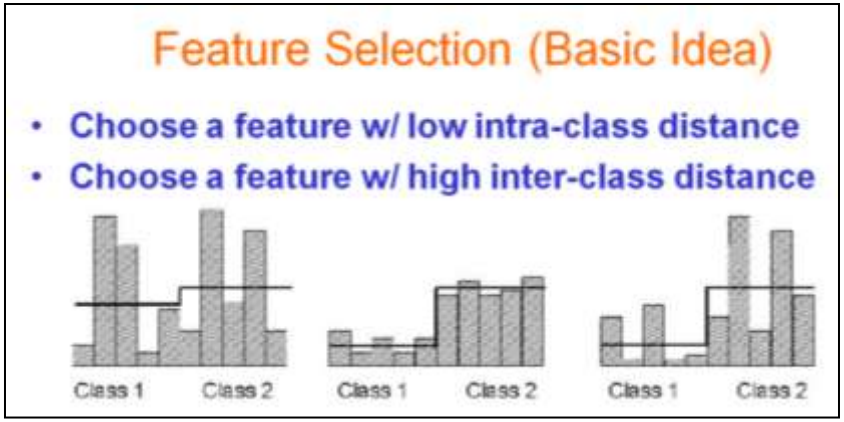
The Steps of Model Construction

- **Training data gathering**
- **Feature generation**
  - k-grams, colour, texture, domain know-how, ...
- **Feature selection**
  - Entropy, $\chi 2$, 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, χ2-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

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

$$\text{Specificity} = \frac{\#\text{ correct -ve predictions}}{\#\text{ -ve}} = \frac{TN}{TP + FN}$$

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

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

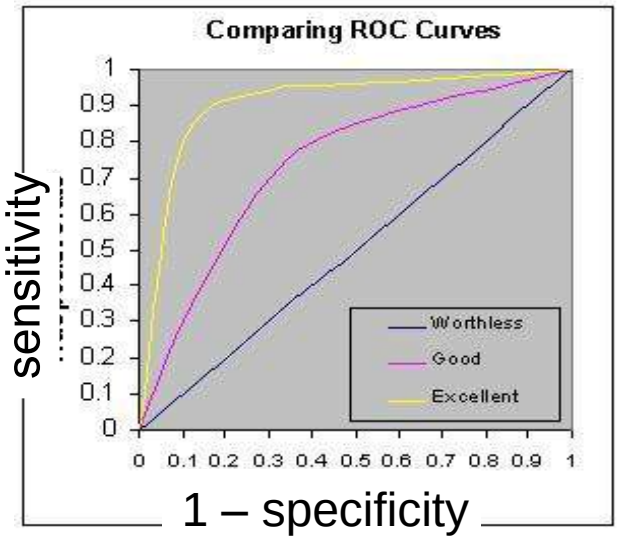
- **Accuracy is not a good measure if the (class) distribution of test data has bias**

- **Sensitivity (SE), specificity (SP), & precision (PPV) are better; but they must be used together**

- **How to combine SE, SP, and/or PPV?**

Combining SE, SP, and/or PPV

- **Area under the ROC (AUC-ROC)**



- **Area under the Precision-Recall Curve (AUC-PR)**

- **F-measure or Harmonic mean (Fm_1)**

$$Fm_1 = \frac{2 * PPV * SE}{PPV + SE}$$

- **Geometric mean (Gm)**

$$Gm = \sqrt{SE * SP}$$

- **Adjusted geometric mean (AGm)**

$$AGm_2 = \frac{Gm + SP * N_n}{1 + N_n}$$

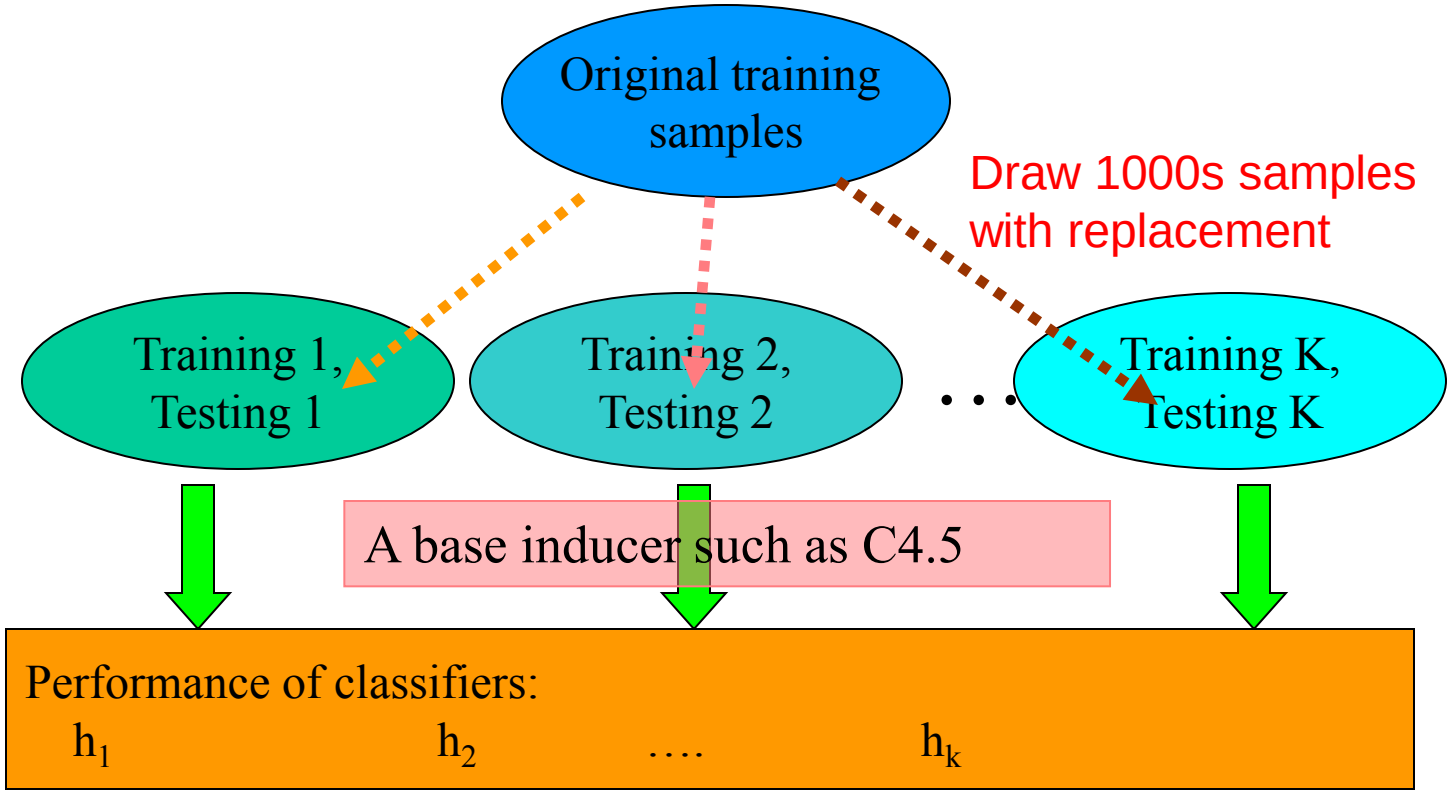
N_n is fraction of negatives in data

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, …, 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**

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

Class-Imbalance Learning

NUS
National University
of Singapore

Many classification problems in bioinformatics have very imbalanced training & testing data

Source: Batuwita & Palade, *JBCB*, 2012

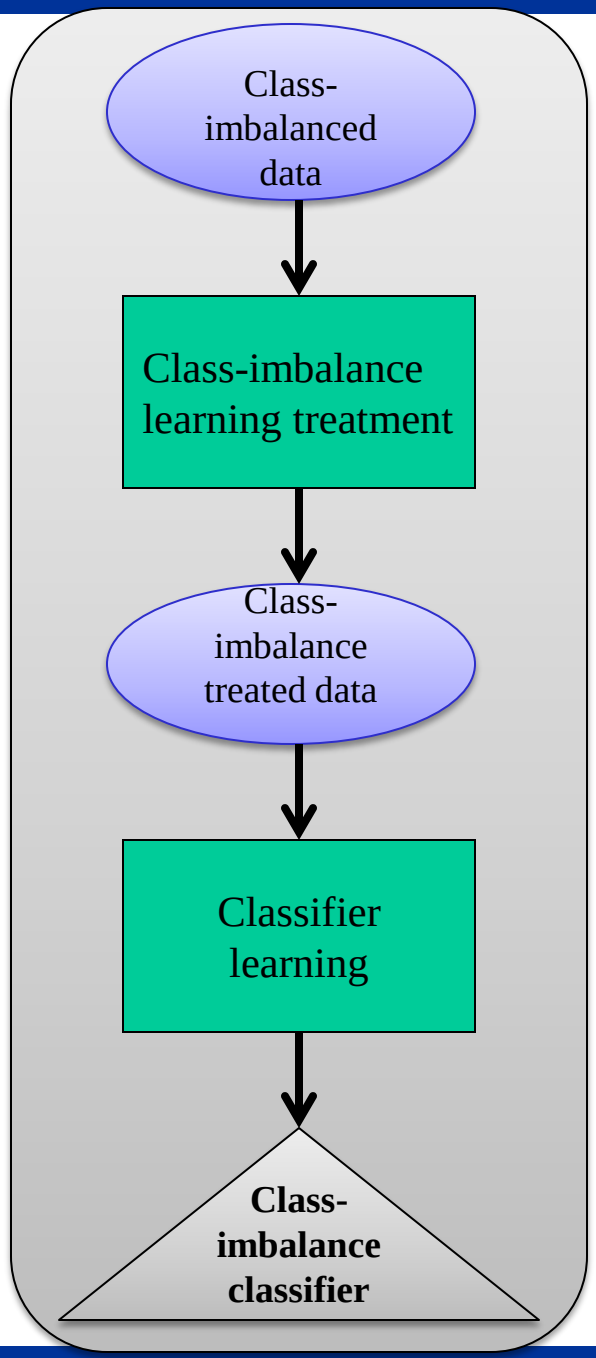
Dataset	Positives	Negatives
miRNA	691	9248
Human promoter	471	5081
snoRNA	98	977
Drug-target	521	5019
Human-splice site-acceptor (Human acceptor)	tr=1116	tr=4672
	ts=208	ts=881
Human-splice site-donor (Human donor)	tr=1116	tr=4140
	ts=208	ts=782
Drosophila-splice site-acceptor (Dros acceptor)	tr=450	tr=1426
	ts=105	ts=355
Drosophila-splice site-donor (Dros donor)	tr=450	tr=1824
	ts=105	ts=298
E.coli protein localization sites (Ecoli)	77	259
Yeast protein localization sites (Yeast)	51	1433
tr = training, ts = testing		

- **–ve samples outnumber +ve samples many times**
- **Normal classifiers don't work well on them**

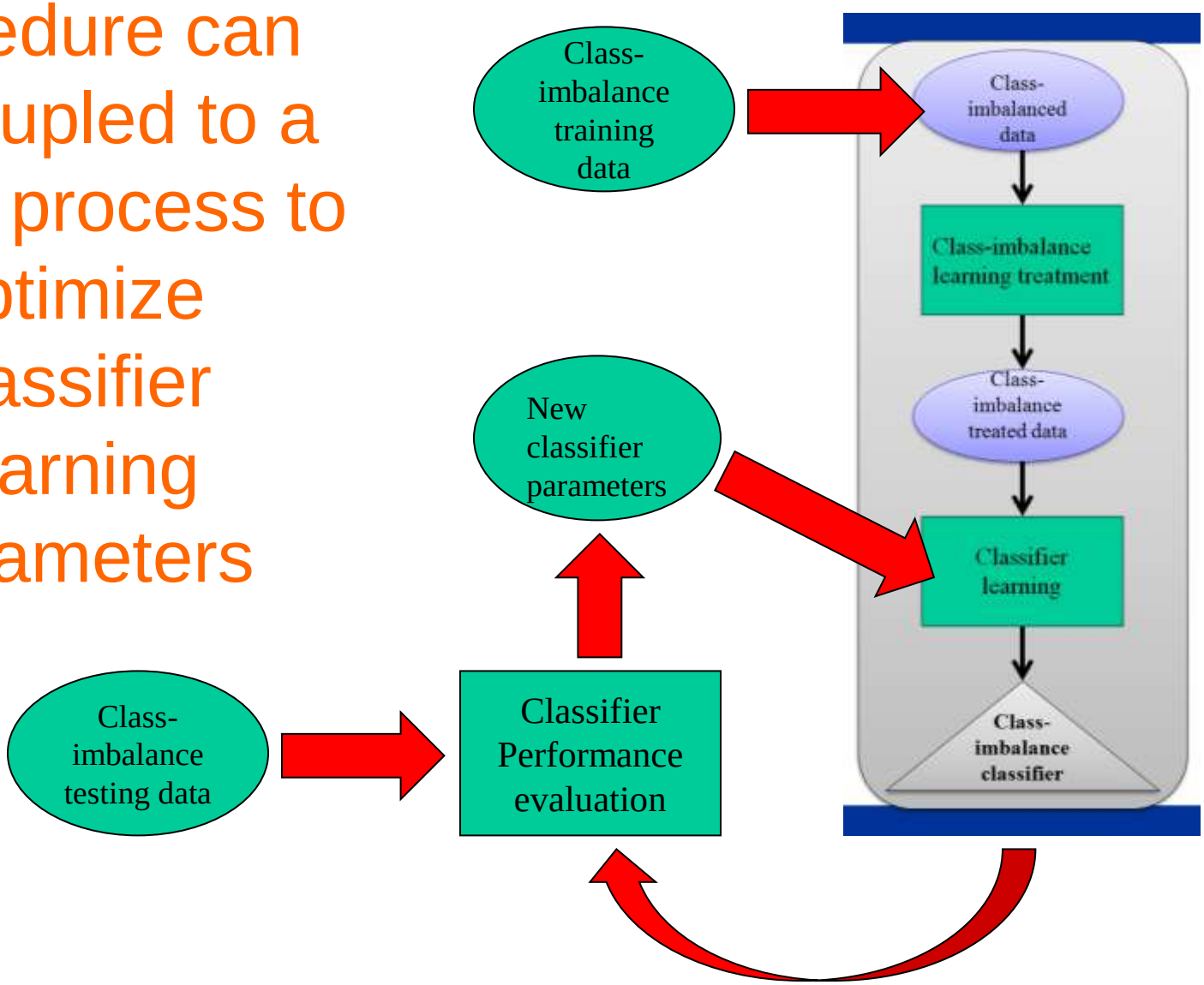
Class-Imbalance Learning

- **Random under-sampling**
  - Remove majority-class samples randomly to balance the data
- **Random over-sampling**
  - Duplicate minority-class samples randomly to balance the data
- **Synthetic minority over-sampling technique (SMOTE)**
  - Generate new synthetic minority-class samples rather than directly duplicating them
- **Different error costs (DEC)**
  - Modify a classifier's cost function

Basic procedure for handling class-imbalanced data



The basic procedure can be coupled to a tuning process to optimize classifier learning parameters



SVM under different class-imbalance treatments & performance evaluation measures

Method	Results for miRNA dataset (%)					Results for Promoter dataset (%)					Results for snoRNA dataset (%)				
	SE	SP	**Gm**	dSE	dSP	SE	SP	**Gm**	dSE	dSP	SE	SP	**Gm**	dSE	dSP
N. SVM	82.78	99.45	90.73			25.69	98.75	50.37			73.27	98.05	84.76		
Under	91.03	93.02	92.02	+8.25	-6.43	70.08	80.67	75.19	+44.39	-18.08	89.97	91.13	90.55	+16.70	-6.93
Over	89.93	96.53	93.17	+7.15	-2.93	68.89	82.56	75.42	+43.20	-16.19	85.75	93.23	89.41	+12.48	-4.83
SMOTE	89.15	97.04	93.01	+6.37	-2.41	67.88	80.57	73.95	+42.18	-18.19	85.10	93.02	88.97	+11.83	-5.04
DEC	90.02	96.30	93.11	+7.24	-3.15	67.74	82.75	74.87	+42.05	-16.00	87.78	94.58	91.11	+14.51	-3.48
	Average			**+7.25**	**-3.73**	Average			**+42.95**	**-17.11**	Average			**+13.08**	**-5.07**
	SE	SP	**Fm**	dSE	dSP	SE	SP	**Fm**	dSE	dSP	SE	SP	**Fm**	dSE	dSP
N. SVM	81.92	99.59	87.41			25.69	98.81	37.06			71.05	98.26	74.22		
Under	90.83	95.00	70.51	+8.91	-4.59	61.46	84.76	37.79	+35.76	-14.05	89.41	91.50	63.43	+18.37	-6.76
Over	82.70	97.93	78.62	+0.78	-1.66	68.55	83.48	39.44	+42.86	-15.33	71.71	97.95	73.19	+ 0.67	-0.31
SMOTE	85.54	97.90	80.05	+3.62	-1.69	64.17	82.76	36.67	+38.48	-16.05	75.49	97.91	75.44	+ 4.44	-0.35
DEC	84.66	97.33	76.89	+2.74	-2.26	66.68	84.84	40.25	+40.99	-13.97	72.22	97.34	71.09	+ 1.18	-0.92
	Average			**+4.01**	**-2.55**	Average			**+39.52**	**-14.85**	Average			**+ 6.16**	**-2.08**
	SE	SP	**AUC-ROC**	dSE	dSP	SE	SP	**AUC-ROC**	dSE	dSP	SE	SP	**AUC-ROC**	dSE	dSP
N. SVM	79.52	99.73	0.9693			25.49	99.36	0.8177			71.05	98.26	0.9816		
Under	91.03	94.39	0.9678	+11.52	-5.34	69.66	81.03	0.8303	+44.17	-18.32	90.67	91.61	0.9713	+19.62	-6.65
Over	89.76	95.19	0.9805	+ 9.69	-3.88	68.94	82.43	0.8386	+43.45	-16.93	82.65	92.00	0.9858	+11.60	-6.26
SMOTE	89.21	95.85	0.9771	+10.24	-4.54	68.85	80.76	0.8303	+43.37	-18.60	79.04	93.02	0.9854	+ 7.99	-5.24
DEC	90.02	95.45	0.9813	+10.50	-4.28	69.02	82.52	0.8398	+43.53	-16.84	78.01	95.98	0.9844	+ 6.96	-2.28
	Average			**+10.49**	**-4.51**	Average			**+43.63**	**-17.67**	Average			**+11.54**	**-5.11**
	SE	SP	**AUC-PR**	dSE	dSP	SE	SP	**AUC-PR**	dSE	dSP	SE	SP	**AUC-PR**	dSE	dSP
N. SVM	79.60	99.71	0.7734			24.62	99.36	0.6298			69.93	98.51	0.7130		
Under	89.44	94.45	0.7356	+9.84	-5.26	69.87	80.78	0.7276	+45.25	-18.57	89.93	85.56	0.6279	+20.00	-12.94
Over	84.13	97.45	0.7544	+4.52	-2.26	67.31	81.76	0.7534	+42.69	-17.60	69.93	98.05	0.7154	+ 0.00	- 0.45
SMOTE	85.13	97.45	0.7554	+5.52	-2.26	64.56	80.90	0.7524	+39.95	-18.46	72.09	98.26	0.7164	+ 2.16	- 0.25
DEC	81.63	99.60	0.7750	+2.02	-0.11	67.96	83.80	0.7324	+43.34	-15.55	70.52	98.40	0.7504	+ 0.58	- 0.11
	Average			**+5.62**	**-2.54**	Average			**+42.81**	**-17.55**	Average			**+ 5.68**	**- 3.44**
	SE	SP	**AGm**	dSE	dSP	SE	SP	**AGm**	dSE	dSP	SE	SP	**AGm**	dSE	dSP
N. SVM	82.49	99.40	94.82			25.69	98.75	73.55			72.16	98.16	90.87		
Under	90.89	94.63	93.65	+8.40	-4.77	65.07	83.53	78.43	+39.38	-15.20	89.41	91.49	90.94	+17.25	-6.67
Over	89.93	96.86	95.03	+7.44	-2.54	68.51	83.49	79.40	+42.82	-15.26	80.16	95.70	91.47	+ 8.00	-2.46
SMOTE	88.75	97.31	95.04	+6.26	-2.09	64.47	82.76	77.70	+38.78	-16.00	79.10	96.82	91.98	+ 6.94	-1.33
DEC	90.02	96.30	94.64	+7.53	-3.10	66.68	84.86	79.84	+40.99	-13.90	83.33	95.39	92.07	+11.18	-2.87
	Average			**+7.41**	**-3.01**	Average			**+40.49**	**-15.09**	Average			**+ 9.07**	**-3.86**

Source: Batuwita & Palade, *JBCB*, 2012

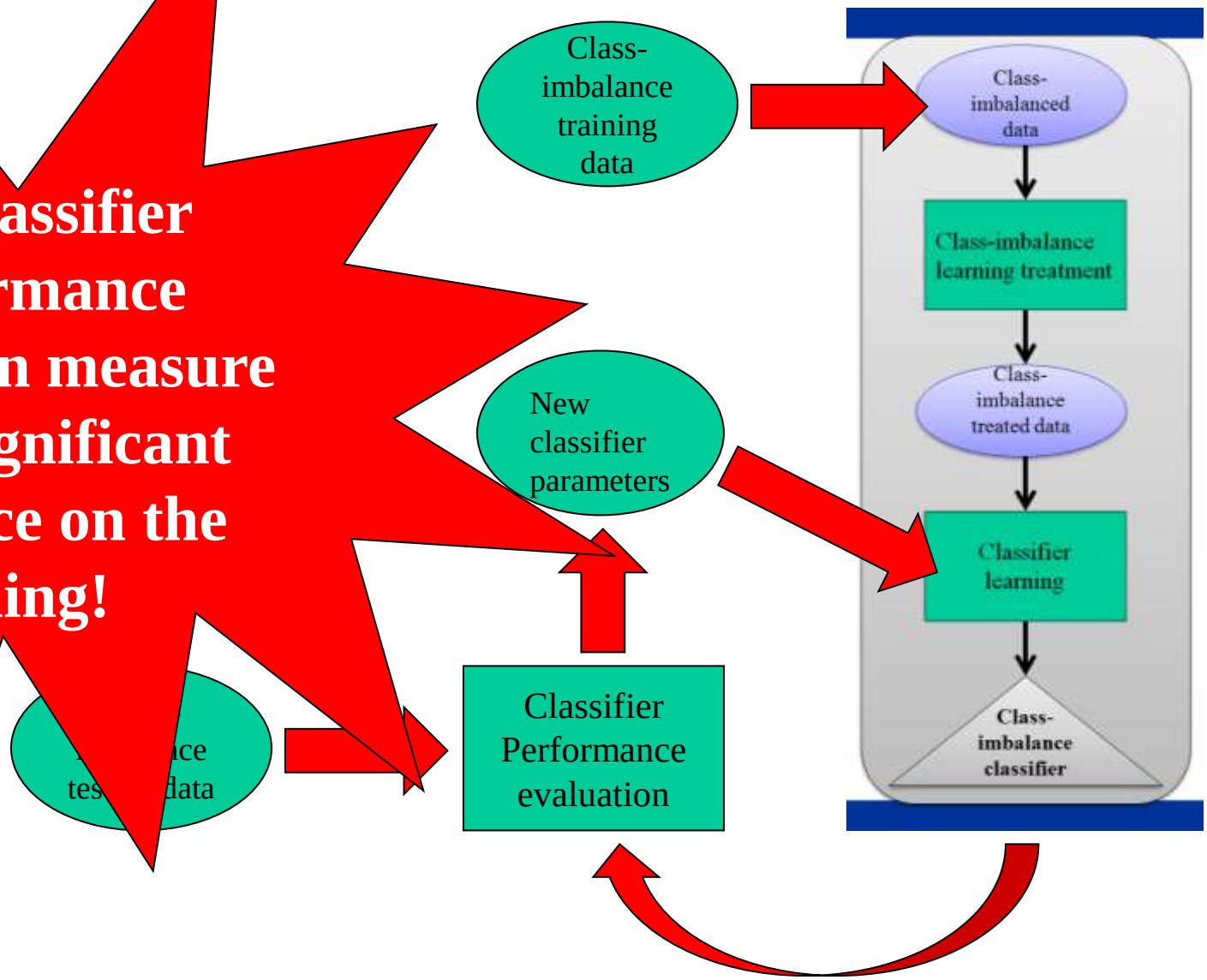
How to choose class-imbalance treatment?

Method	Results for miRNA dataset (%)				
	SE	SP	Gm	dSE	dSP
N. SVM	82.78	99.45	90.73		
Under	91.03	93.02	92.02	+8.25	-6.43
Over	89.93	96.53	93.17	+7.15	-2.93
SMOTE	89.15	97.04	93.01	+6.37	-2.41
DEC	90.02	96.30	93.11	+7.24	-3.15
	Average			**+7.25**	**-3.73**

- **There are usually a lot more –ve samples; often normal classifiers (N.SVM) have low SE & high SP**
  - Users generally want higher SE, w/o sacrificing too much SP
- **Class-imbalance treatment aim to maximize improvement in SE (dSE) & minimize loss in SP (dSP)**

⇒ **Choose one that maximizes R = |dSP / dSE|**

The classifier performance evaluation measure has a significant influence on the tuning!



Geometric Mean

$$Gm = \sqrt{SE * SP}$$

- **Gm weighs changes in SE and SP equally**

$\Rightarrow$ **Gm may select sub-optimal models for imbalanced data**

Example A

- ModelNormal: SE=70.00%, SP=99.00%, Gm=83.25%

- **After class-imbalanced treatment under Gm**
  - *Model1: SE = 92%, SP = 98%, Gm = 94.95%*
  - *Model2: SE = 94.95%, SP = 94.95%, Gm = 94.95%*
  - *Model3: SE = 98%, SP = 92%, Gm = 94.95%*
- **All 3 treated models have the same Gm**
- **But Model1 (dSE=22%, dSP= –1%) is the best**

Source: Batuwita & Palade, *JBCB*, 2012

F-Measure

$$Fm_1 = \frac{2 * PPV * SE}{PPV + SE}$$

- **Fm works well if training set is imbalance but testing set is balanced. If testing set is also imbalanced, Fm is more sensitive to changes in PPV, leading to sub-optimal models**

Example B

- **Let –ve testing samples = 950, +ve testing samples = 50**

  Let Model1 & Model2 be resulted when applying class imbalance learning for diff model parameters
  - ModelNormal: SE = 70%, SP = 99%
  - Model1: SE = 70%, SP = 99%, Then PPV = 80%, Fm = 74%
  - Model2: SE = 90%, SP = 97%, Then PPV = 62%, Fm = 73%
- **Here Model2 is better; but Fm picks Model1**

Source: Batuwita & Palade, *JBCB*, 2012

AUC-ROC

- **AUC-ROC plots SE vs (1- SP) and compute area under the curve. Thus it weighs changes in SE and SP equally**

$\Rightarrow$ **Similar problem as Gm**

AUC-PR

- **AUC-PR plots SE vs PPV and compute area under the curve. Thus if test set is imbalanced, it is more sensitive to changes in PPV than SE**

$\Rightarrow$ **Similar problem as Fm**

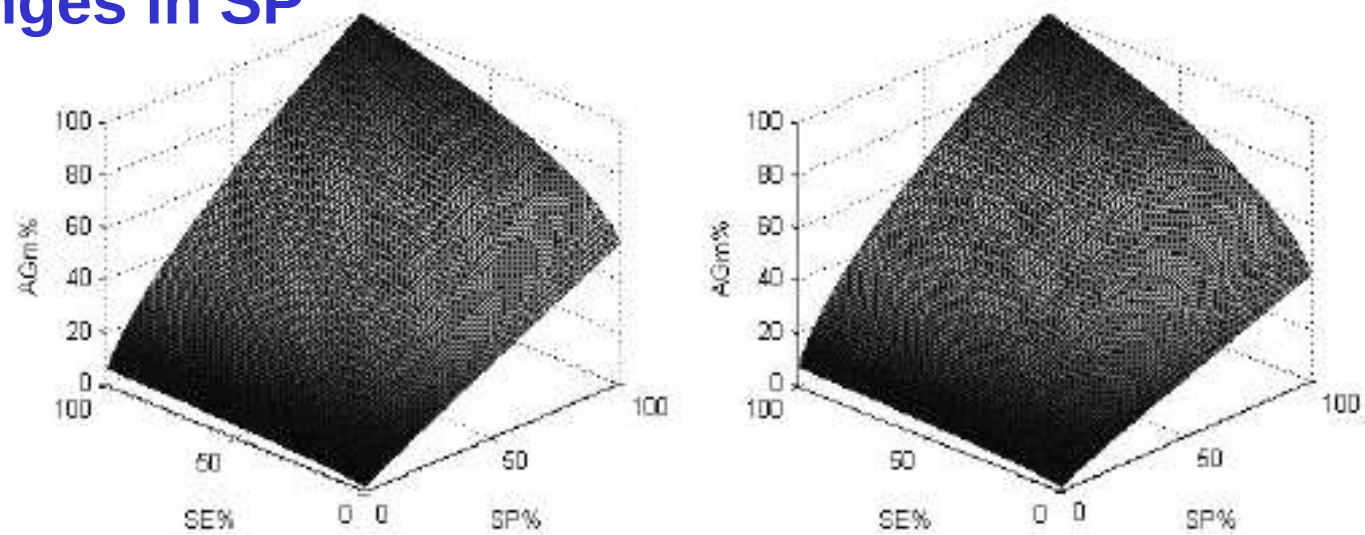
Source: Batuwita & Palade, *JBCB*, 2012

Adjusted Geometric Mean

$$AGm_2 = \frac{Gm + SP * N_n}{1 + N_n}$$

N_n = proportion of –ve samples

- **AGm metric is more sensitive to changes in SP than to changes in SE. Moreover, the higher the imbalance (N_n), the higher its sensitiveness to changes in SP**



Distribution of AGm over SE and SP when (a) $N_n = 0.9$ and (b) $N_n = 0.5$.

Adjusted Geometric Mean

$$AGm_2 = \frac{Gm + SP * N_n}{1 + N_n}$$

N_n = proportion of –ve samples

Example A – Revisited (N_n = 95%)

- ModelNormal: SE=70.00%, SP=99.00%, Gm=83.25%

- **After class-imbalanced treatment under Gm**
  - *Model1: SE = 92%, SP = 98%, Gm = 94.95%*
    - AGm = (94.95+98*0.95)/(1+0.95) = **96.44%**
  - *Model2: SE = 94.95%, SP = 94.95%, Gm = 94.95%*
    - AGm = (94.95+94.95*0.95)/(1+0.95) = 94.95%
  - *Model3: SE = 98%, SP = 92%, Gm = 94.95%*
    - AGm = (94.95+92*0.95)/(1+0.95) = 93.51%
- **All 3 treated models have the same Gm**
- **But Model1 (dSE=22%, dSP=–1%) is the best**
- **AGm correctly picks Model1**

Source: Batuwita & Palade, *JBCB*, 2012

Adjusted Geometric Mean

$$AGm_2 = \frac{Gm + SP * N_n}{1 + N_n}$$

N_n = proportion of –ve samples

Example B – Revisited (N_n = 95%)

- **Let –ve testing samples = 950, +ve testing samples = 50**

  Let Model1 & Model2 be resulted when applying class imbalance learning for diff model parameters
  - ModelNormal: SE = 70%, SP = 99%
  - Model1: SE = 70%, SP = 99%, Then PPV = 80%, Fm = 74%
    - AGm = 91%
  - Model2: SE = 90%, SP = 97%, Then PPV = 62%, Fm = 73%
    - **AGm = 95%**
- **Here Model2 is better; but Fm picks Model1**
- **AGm correctly picks Model2**

Source: Batuwita & Palade, *JBCB*, 2012

What have we learned?

- **Class-imbalance problems**
- **Class-imbalance treatment procedure**
- **Better appreciation of classifier performance measure, especially under class-imbalance situation**

References

- **Must read**
  - Batuwita & Palade. “Adjusted geometric mean: A novel performance measure for imbalanced bioinformatics datasets learning”, *JBCB*, 10(4):???-???, 2012. To appear.

Journal of Bioinformatics and Computational Biology

ADJUSTED GEOMETRIC-MEAN: A NOVEL PERFORMANCE MEASURE FOR IMBALANCED BIOINFORMATICS DATASETS LEARNING

RUKSHAN BATUWITA
University of Oxford, Department of Computer Science
Oxford, OX1 3QD, United Kingdom
manb@cs.ox.ac.uk

VASILE PALADE
University of Oxford, Department of Computer Science
Oxford, OX1 3QD, United Kingdom
vasile.palade@cs.ox.ac.uk

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.**