Data Mining Algorithms

Vipin Kumar

Department of Computer Science,

University of Minnesota,

Minneapolis, USA.

Tutorial Presented at IPAM 2002 Workshop on Mathematical Challenges in Scientific Data Mining
January 14, 2002

What is Data Mining?

- Search for Valuable Information in Large Volumes of Data.
- Draws ideas from machine learning/AI, pattern recognition, statistics, database systems, and data visualization.
- Traditional Techniques may be unsuitable
  - Enormity of data
  - High Dimensionality of data
  - Heterogeneous, Distributed nature of data

Why Mine Data? Commercial Viewpoints...

- Lots of data is being collected and warehoused.
- Computing has become affordable.
- Competitive Pressure is Strong
  - Provide better, customized services for an *edge*.
  - Information is becoming product in its own right.

Why Mine Data? Scientific Viewpoint...

- Data collected and stored at enormous speeds (Gbyte/hour)
  - remote sensor on a satellite
  - telescope scanning the skies
  - microarrays generating gene expression data
  - scientific simulations generating terabytes of data
- Traditional techniques are infeasible for raw data
- Data mining for data reduction..
  - cataloging, classifying, segmenting data
  - Helps scientists in Hypothesis Formation

Data Mining Tasks

- Prediction Methods
  - Use some variables to predict unknown or future values of other variables.

    Examples: Classification, Regression, Deviation detection.

- Description Methods
  - Find human-interpretable patterns that describe the data.

    Examples: Clustering, Associations, Classification.

From [Fayyad, et.al.] Advances in Knowledge Discovery and Data Mining, 1996

Association Rule Discovery: Definition

- Given a set of records each of which contain some number of items from a given collection;
  - Produce dependency rules which will predict occurrence of an item based on occurrences of other items.

TID	*Items*
1	Bread, Coke, Milk
2	Beer, Bread
3	Beer, Coke, Diaper, Milk
4	Beer, Bread, Diaper, Milk
5	Coke, Diaper, Milk

Rules Discovered:

{Milk} --> {Coke}

{Diaper, Milk} --> {Beer}

Association Rules: Support and Confidence

TID	*Items*
1	Bread, Milk
2	Beer, Diaper, Bread, Eggs
3	Beer, Coke, Diaper, Milk
4	Beer, Bread, Diaper, Milk
5	Coke, Bread, Diaper, Milk

Association Rule: $\mathrm{X} \Rightarrow_{s,\alpha} \mathrm{y}$

Support: $s = \dfrac{\sigma(\mathrm{X} \cup \mathrm{y})}{|T|} (s = P(\mathrm{X}, \mathrm{y}))$

Confidence: $\alpha = \dfrac{\sigma(\mathrm{X} \cup \mathrm{y})}{\sigma(\mathrm{X})|} (\alpha = P(\mathrm{y} \,|\, \mathrm{X}))$

Example:

$$\{\text{Diaper}, \text{Milk}\} \Rightarrow_{s,\alpha} \text{Beer}$$

$$s = \frac{\sigma(\text{Diaper}, \text{Milk}, \text{Beer})}{\text{Total Number of Transactions}} = \frac{2}{5} = 0.4$$

$$\alpha = \frac{\sigma(\text{Diaper}, \text{Milk}, \text{Beer})}{\sigma(\text{Diaper}, \text{Milk})|} = 0.66$$

Handling Exponential Complexity

- Given *n* transactions and *m* different items:
  - number of possible association rules: $O(m2^{m-1})$
  - computation complexity: $O(nm2^m)$
- Systematic search for all patterns, based on support constraint [Agarwal & Srikant]:
  - If {A,B} has support at least α, then both A and B have support at least α.
  - If either A or B has support less than α, then {A,B} has support less than α.
  - Use patterns of *n*-1 items to find patterns of *n* items.

Apriori Principle

- Collect single item counts. Find frequent items.
- Find candidate pairs, count them => frequent *pairs* of items.
- Find candidate triplets, count them => frequent *triplets* of items, And so on...
- Guiding Principle: *Every subset of a frequent itemset has to be frequent.*
  - **Used for pruning many candidates.**

Illustrating Apriori Principle

Item	Count
Bread	**4**
Coke	**2**
Milk	**4**
Beer	**3**
Diaper	**4**
Eggs	**1**

Items (1-itemsets)

Itemset	Count
{Bread,Milk}	**3**
{Bread,Beer}	**2**
{Bread,Diaper}	**3**
{Milk,Beer}	**2**
{Milk,Diaper}	**3**
{Beer,Diaper}	**3**

Pairs (2-itemsets)

Minimum Support = 3

Triplets (3-itemsets)

Itemset	Count
{Bread,Milk,Diaper}	**3**

If every subset is considered,

$^6C_1 + {}^6C_2 + {}^6C_3 = 41$

With support-based pruning,

$6 + 6 + 1 = 13$

Apriori Algorithm

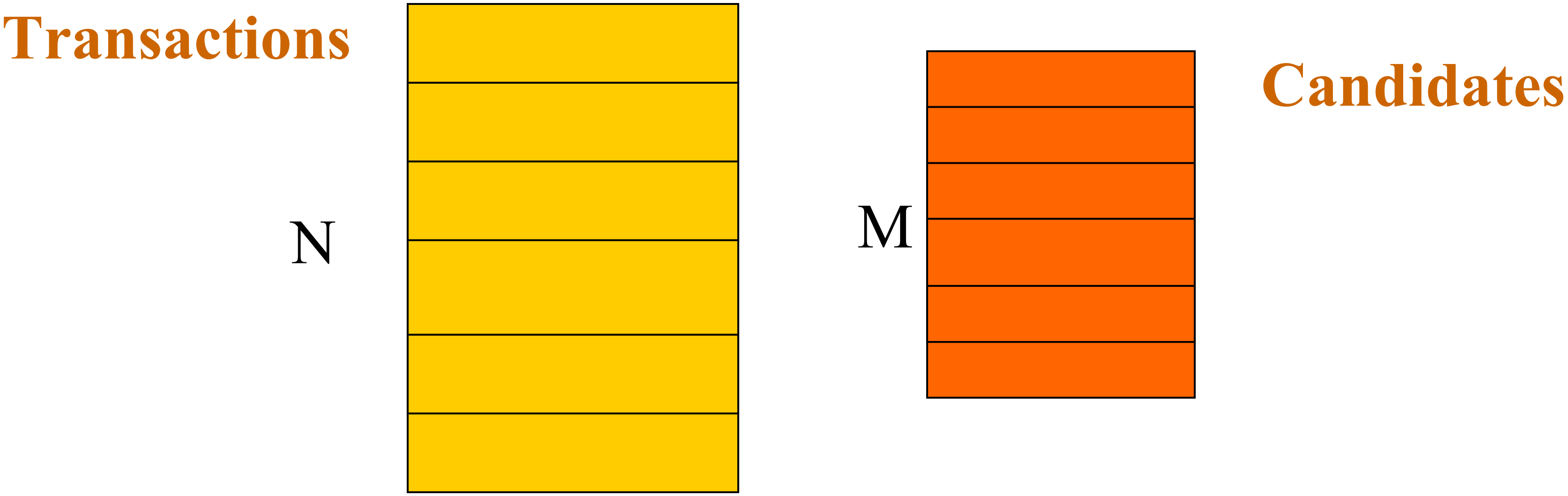
```
$F_1$ = {frequent 1-item sets};
k = 2;
while( $F_{k-1}$ is not empty ) {
        $C_k$ = Apriori_generate( $F_{k-1}$ );
        for all transactions t in T {
                Subset( $C_k$, t );
        }
        $F_k$ = { c in $C_k$ s.t. c.count >= minimum_support};
}
Answer = union of all sets $F_k$;
```

Association Rule Discovery: Apriori_generate

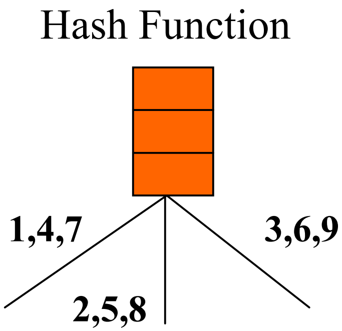
```
Apriori_generate( F(k-1) ) {
        join $F_{k-1}$ with $F_{k-1}$ such that,
            $c_1 = (i_1, i_2, .., i_{k-1})$ and $c_2 = (j_1, j_2, .., j_{k-1})$ join together if
                $i_p = j_p$ for $1 <= p <= k-1$,
        and then new candidate, c, has a form
            $c = (i_1, i_2, .., i_{k-1}, j_{k-1})$.
        c is then added to a hash-tree structure.
}
```

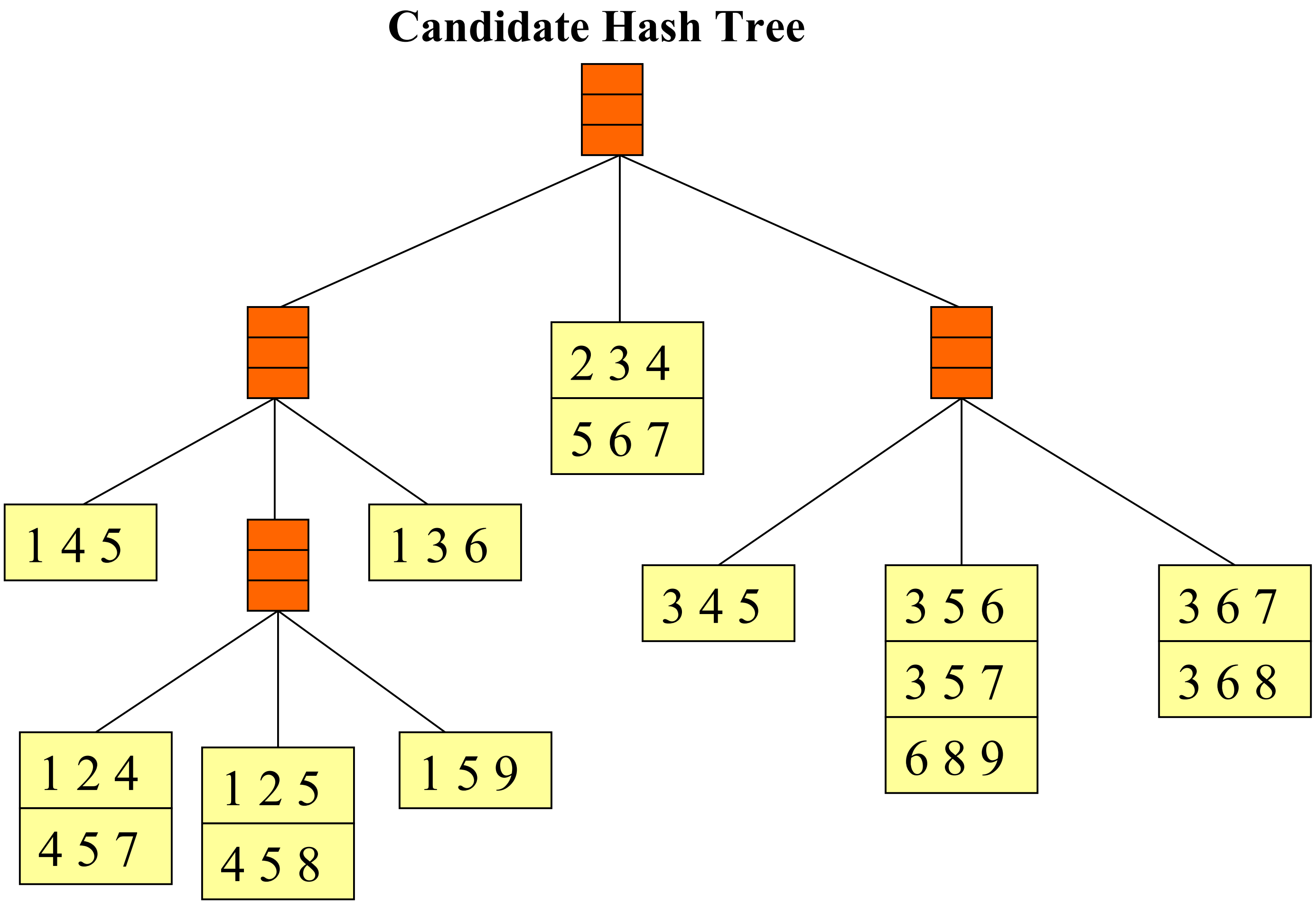
Counting Candidates

- Frequent Itemsets are found by counting candidates.
- Simple way:
  - Search for each candidate in each transaction. Expensive!!!

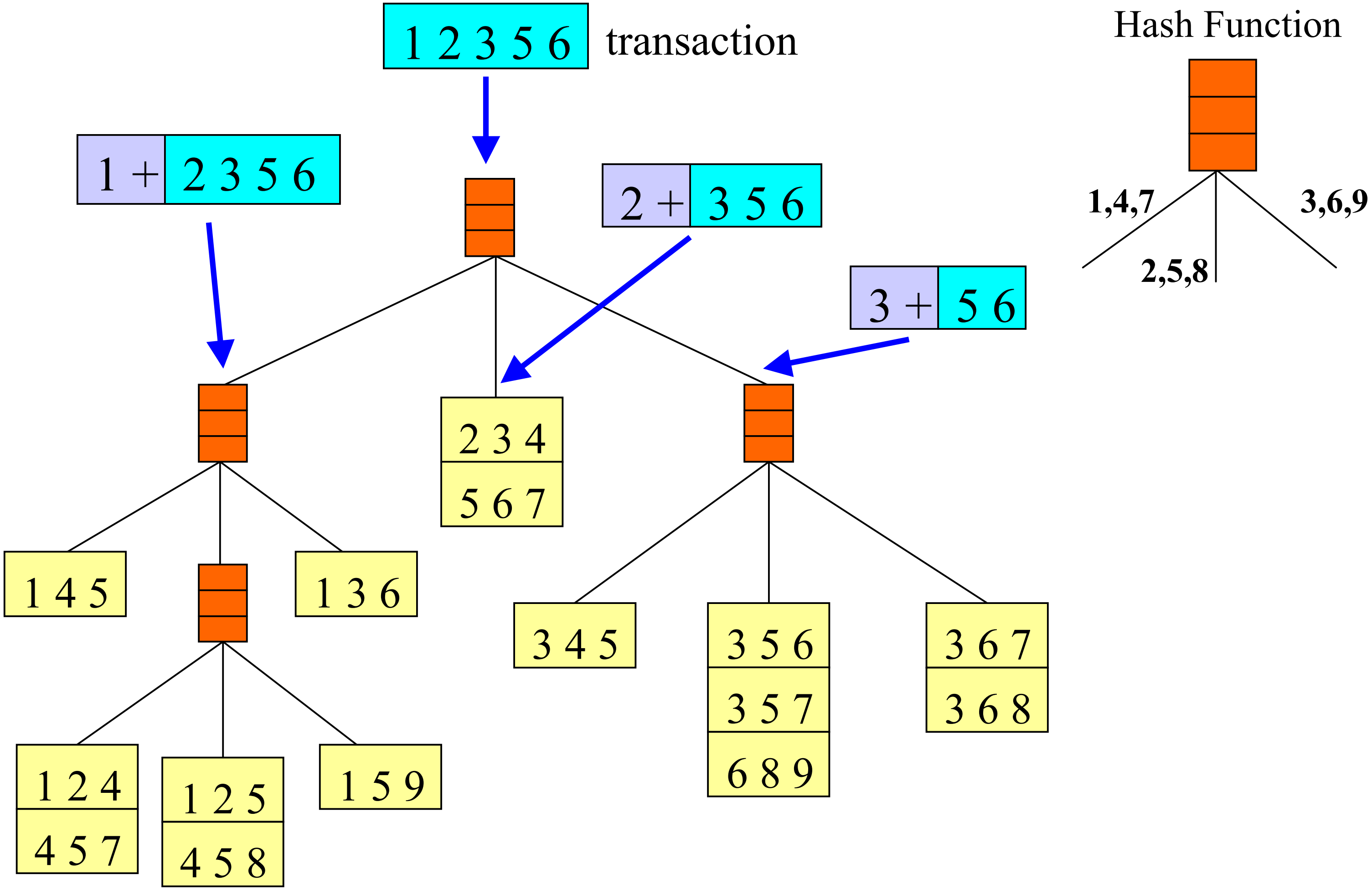


Association Rule Discovery: Hash tree for fast access.

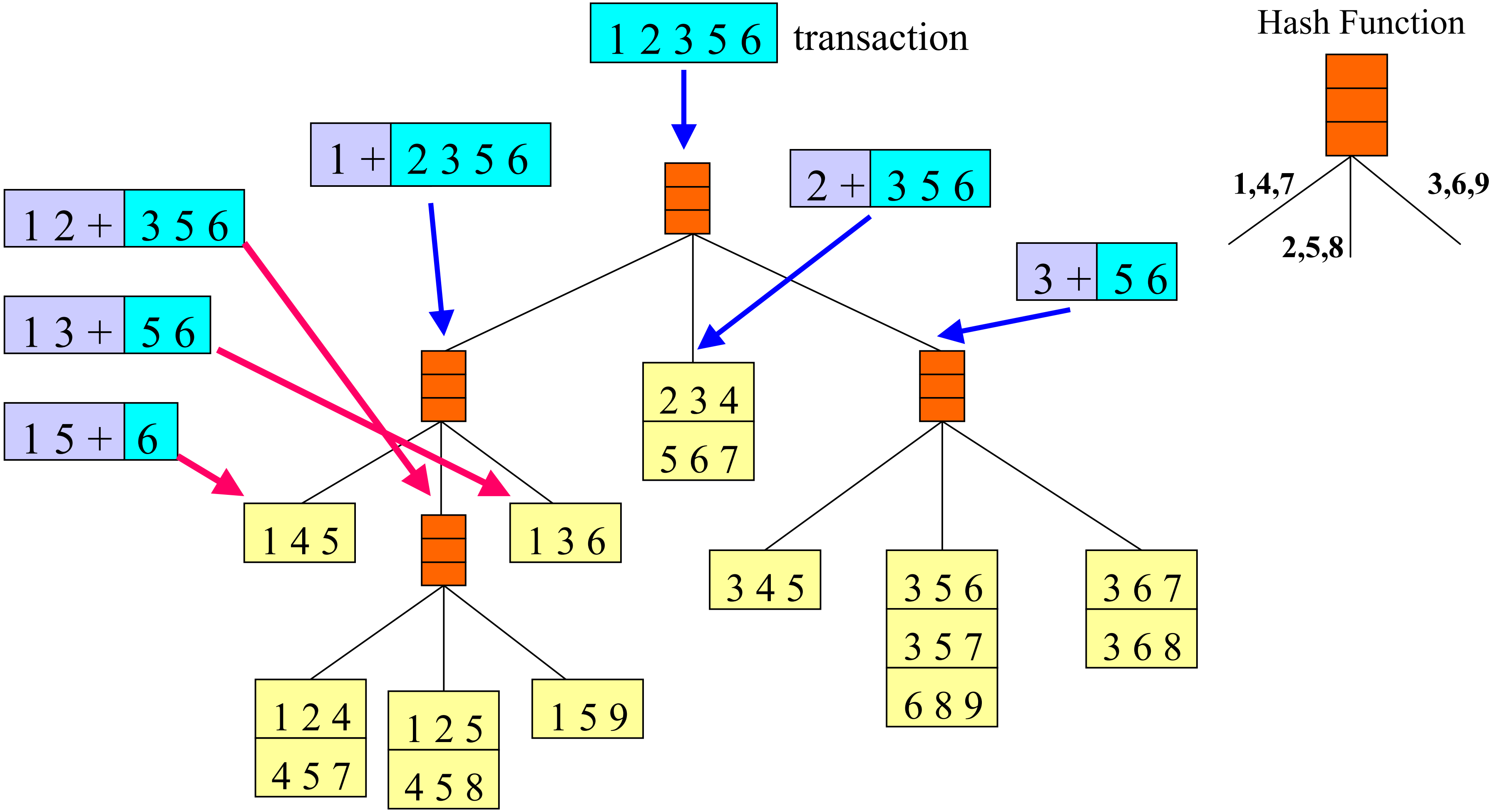




Association Rule Discovery: Subset Operation



Association Rule Discovery: Subset Operation (contd.)



Discovering Sequential Associations

Given:

A set of objects with associated event occurrences.

Object	Event Sequences
1	(A, B) → (C)
2	(B) → (C) → (D)
3	(A) → (CD)
4	(A) → (A) → (C)

Sequential Pattern Discovery: Examples

- In telecommunications alarm logs,
  - (Inverter_Problem Excessive_Line_Current)

    (Rectifier_Alarm) --> (Fire_Alarm)
- In point-of-sale transaction sequences,
  - Computer Bookstore:

    (Intro_To_Visual_C) (C++_Primer) -->
    (Perl_for_dummies,Tcl_Tk)
  - Athletic Apparel Store:

    (Shoes) (Racket, Racketball) --> (Sports_Jacket)

Discovery of Sequential Patterns : Complexity

- Much **higher computational complexity** than association rule discovery.
  - **O($m^k\ 2^{k-1}$) number of possible sequential patterns having *k* events, where *m* is the total number of possible events.**
- Time constraints offer some pruning. Further use of **support based pruning contains complexity**.
  - A subsequence of a sequence occurs at least as many times as the sequence.
  - A sequence has no more occurrences than any of its subsequences.
  - Build sequences in increasing number of events. [GSP algorithm by Agarwal & Srikant]

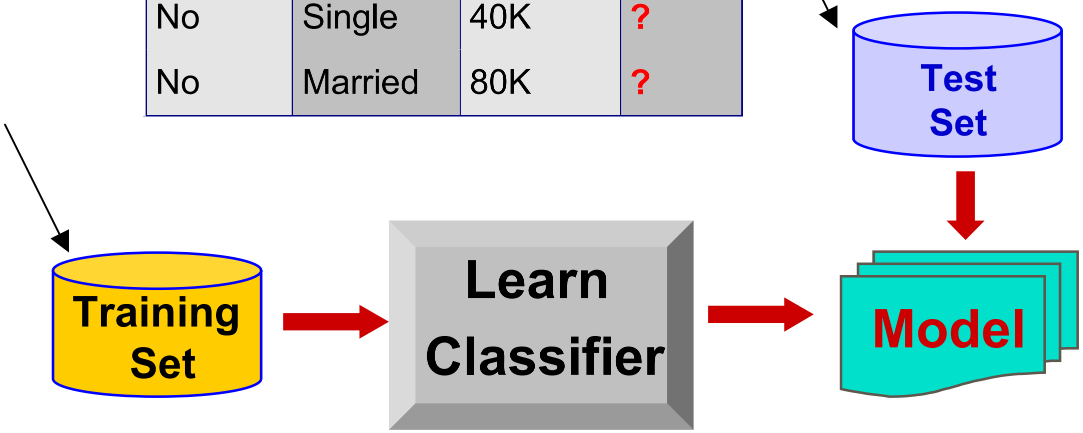
Classification: Definition

- Given a collection of records (*training set*)
  - Each record contains a set of *attributes*, one of the attributes is the *class*.
- Find a *model* for class attribute as a function of the values of other attributes.
- Goal: previously unseen records should be assigned a class as accurately as possible.
  - A *test set* is used to determine the accuracy of the model. Usually, the given data set is divided into training and test sets, with training set used to build the model and test set used to validate it.

Classification Example

		categorical	categorical	continuous	class
Tid	**Refund**	**Marital Status**	**Taxable Income**	**Cheat**	
1	Yes	Single	125K	**No**	
2	No	Married	100K	**No**	
3	No	Single	70K	**No**	
4	Yes	Married	120K	**No**	
5	No	Divorced	95K	**Yes**	
6	No	Married	60K	**No**	
7	Yes	Divorced	220K	**No**	
8	No	Single	85K	**Yes**	
9	No	Married	75K	**No**	
10	No	Single	90K	**Yes**	

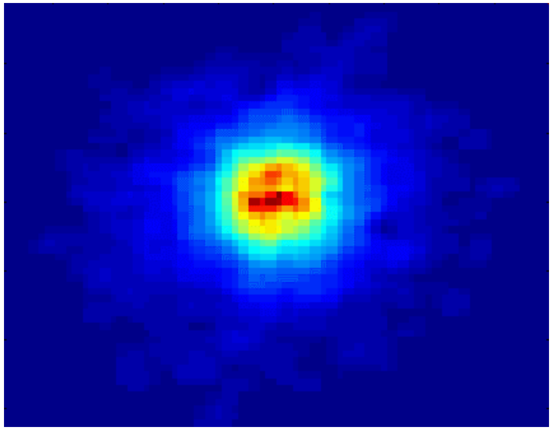
Refund	Marital Status	Taxable Income	Cheat
No	Single	75K	**?**
Yes	Married	50K	**?**
No	Married	150K	**?**
Yes	Divorced	90K	**?**
No	Single	40K	**?**
No	Married	80K	**?**



Classifying Galaxies

Courtsey: http://aps.umn.edu

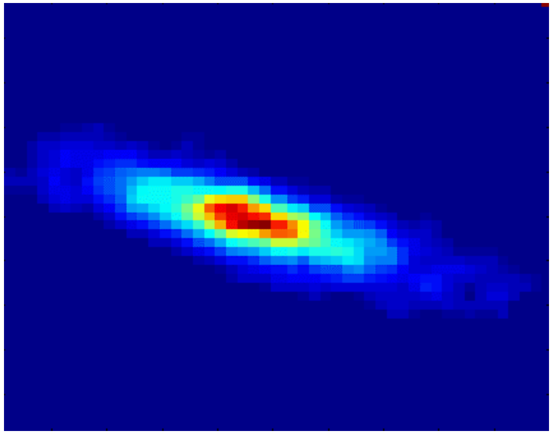
Early



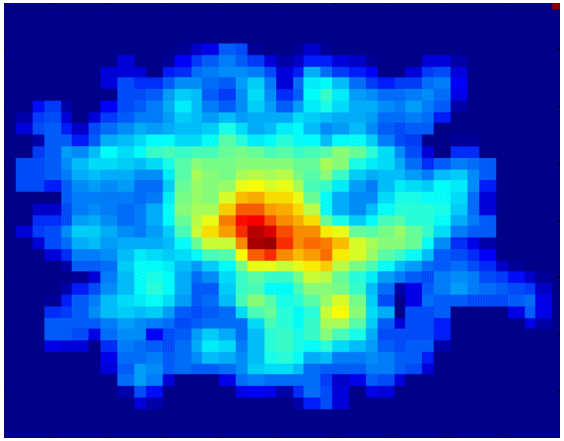
Class:
- Stages of Formation

Attributes:
- Image features,
- Characteristics of light waves received, etc.

Intermediate



Late



Data Size:
- 72 million stars, 20 million galaxies
- Object Catalog: 9 GB
- Image Database: 150 GB

Classification Approaches

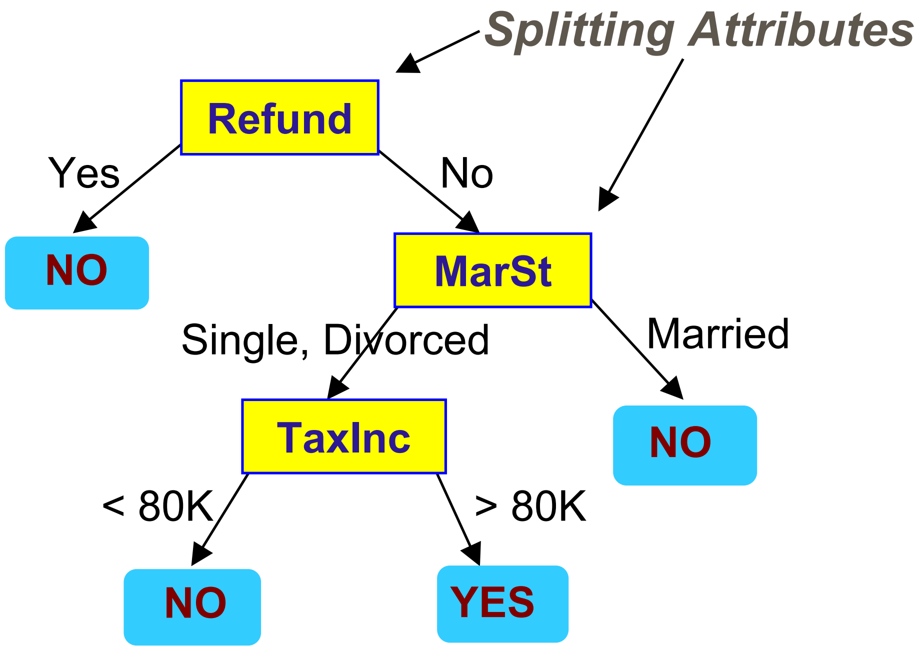
- Decision Tree based Methods
- Rule-based Methods
- Memory based reasoning
- Neural Networks
- Genetic Algorithms
- Bayesian Networks
- Support Vector Machines
- Meta Algorithms
  - Boosting
  - Bagging

Decision Tree Based Classification

- Decision tree models are better suited for data mining:
  - Inexpensive to construct
  - Easy to Interpret
  - Easy to integrate with database systems
  - Comparable or better accuracy in many applications

Example Decision Tree

	categorical	categorical	continuous	class
Tid	**Refund**	**Marital Status**	**Taxable Income**	**Cheat**
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes



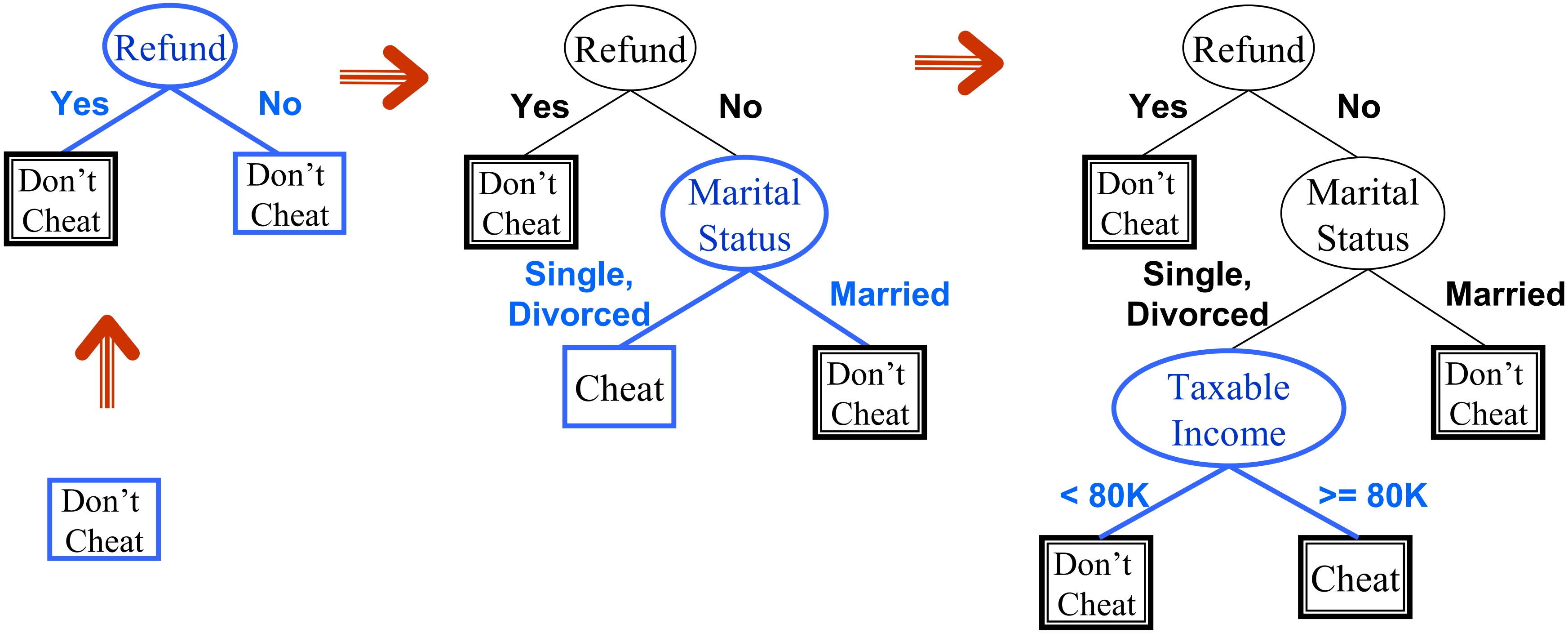
Decision Tree Algorithms

- Many Algorithms:
  - Hunt's Algorithm (one of the earliest).
  - CART
  - ID3, C4.5
  - SLIQ,SPRINT

- General Structure:
  - Tree Induction
  - Tree Pruning

Hunt’s Method

- An Example:
  - Attributes: Refund (Yes, No), Marital Status (Single, Married, Divorced), Taxable Income (Continuous)
  - Class: Cheat, Don’t Cheat

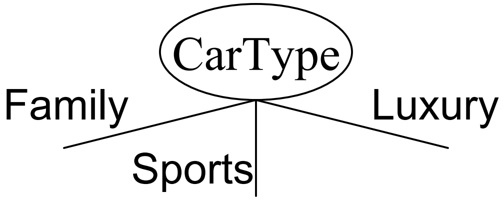


Tree Induction

- Greedy strategy.
  - Choose to split records based on an attribute that optimizes the splitting criterion.
- Two phases at each node:
  - Split Determining Phase:
    - How to Split a Given Attribute?
    - Which attribute to split on? Use Splitting Criterion.
  - Splitting Phase:
    - Split the records into children.

Splitting Based on Categorical Attributes

- Each partition has a subset of values signifying it.
- Simple method: Use as many partitions as distinct values.



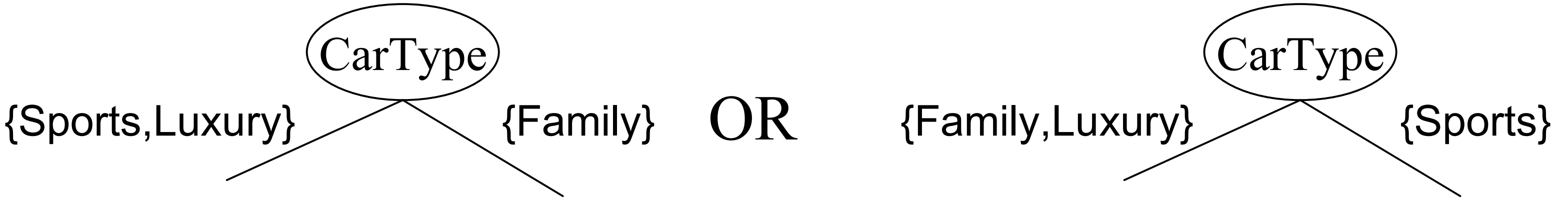
CarType

Family

Sports

Luxury

- Complex method: Two partitions. Each partitioning divides values into two subsets. Need to find optimal partitioning.



CarType

{Sports,Luxury}

{Family}

OR

CarType

{Family,Luxury}

{Sports}

Splitting Based on Continuous Attributes

- Different ways of handling
  - Static: Apriori Discretization to form a categorical attribute
    - may not be desirable in many situations
  - Dynamic: Make decisions as algorithm proceeds
    - complex but more powerful and flexible in approximating true dependency

Splitting Criterion: GINI

- Gini Index:

$$GINI(t) = 1 - \sum_j [p(j \mid t)]^2$$

(NOTE: $p(j \mid t)$ is the relative frequency of class j at node t).

- Measures impurity of a node.
  - Maximum ($1 - 1/n_c$) when records are equally distributed among all classes, implying least interesting information
  - Minimum (0.0) when all records belong to one class, implying most interesting information

C1	**0**
C2	**6**
Gini=0.000	

C1	**1**
C2	**5**
Gini=0.278	

C1	**2**
C2	**4**
Gini=0.444	

C1	**3**
C2	**3**
Gini=0.500	

Splitting Based on GINI

- Used in CART, SLIQ, SPRINT.
- Splitting Criterion: Minimize Gini Index of the Split.
- When a node p is split into k partitions (children), the quality of split is computed as,

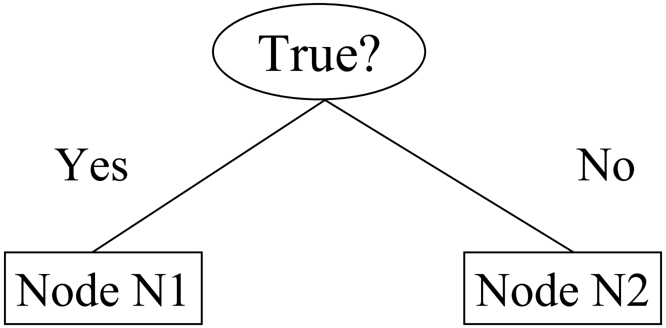
$$\boxed{GINI_{split} = \sum_{i=1}^{k} \frac{n_i}{n} GINI(i)}$$

where, n_i = number of records at child i,

n = number of records at node p.

Binary Attributes: Computing GINI Index

- Splits into two partitions
- Effect of Weighing partitions:
  - Larger and Purer Partitions are sought for.



	N1	N2
C1	0	4
C2	6	0
Gini=0.000		

	N1	N2
C1	3	4
C2	3	0
Gini=0.300		

	N1	N2
C1	4	2
C2	4	0
Gini=0.400		

	N1	N2
C1	6	2
C2	2	0
Gini=0.300		

Categorical Attributes: Computing Gini Index

- For each distinct value, gather counts for each class in the dataset
- Use the count matrix to make decisions

Multi-way split

	CarType		
	Family	Sports	Luxury
C1	1	2	1
C2	4	1	1
Gini	0.393		

Two-way split
(find best partition of values)

	CarType	
	{Sports, Luxury}	{Family}
C1	3	1
C2	2	4
Gini	0.400	

	CarType	
	{Sports}	{Family, Luxury}
C1	2	2
C2	1	5
Gini	0.419	

Continuous Attributes: Computing Gini Index

- Use Binary Decisions based on one value
- Several Choices for the splitting value
  - Number of possible splitting values = Number of distinct values
- Each splitting value has a count matrix associated with it
  - Class counts in each of the partitions, A < v and A >= v
- Simple method to choose best v
  - For each v, scan the database to gather count matrix and compute its Gini index
  - Computationally Inefficient! Repetition of work.

Continuous Attributes: Computing Gini Index...

- For efficient computation: for each attribute,
  - Sort the attribute on values
  - Linearly scan these values, each time updating the count matrix and computing gini index
  - Choose the split position that has the least gini index

Cheat	No	No	No	Yes	Yes	Yes	No	No	No	No
	Taxable Income									
Sorted Values →	60	70	75	85	90	95	100	120	125	220

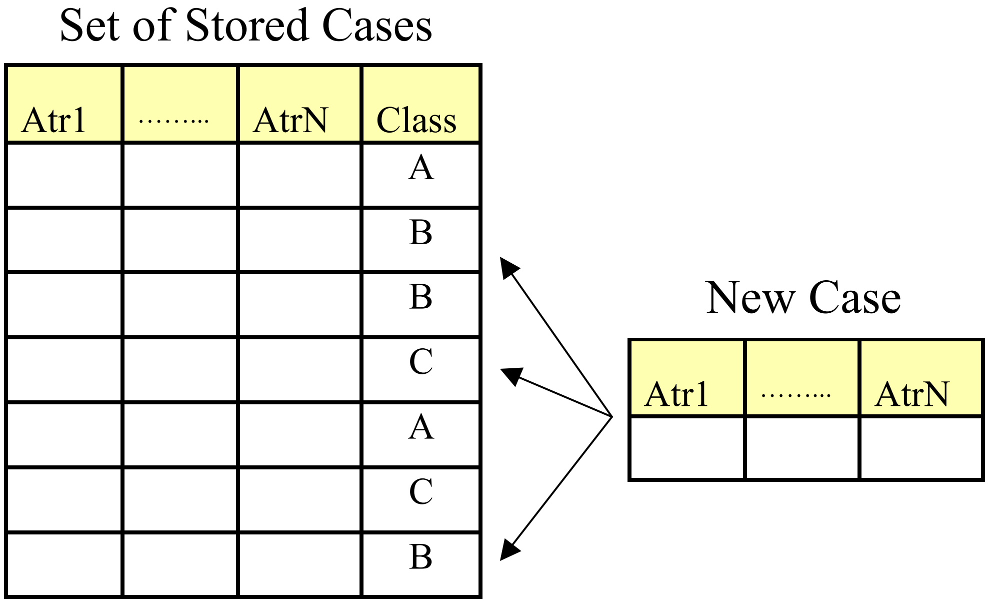
Split Positions →	55		65		72		80		87		92		97		110		122		172		230	
	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>
Yes	0	3	0	3	0	3	0	3	1	2	2	1	3	0	3	0	3	0	3	0	3	0
No	0	7	1	6	2	5	3	4	3	4	3	4	3	4	4	3	5	2	6	1	7	0
Gini	0.420		0.400		0.375		0.343		0.417		0.400		*0.300*		0.343		0.375		0.400		0.420	

C4.5

- Simple depth-first construction.
- Sorts Continuous Attributes at each node.
- Needs entire data to fit in memory.
- Unsuitable for Large Datasets.
  - Needs out-of-core sorting.

- Classification Accuracy shown to improve when *entire* datasets are used!

Classification: Memory Based Reasoning



Set of Stored Cases

Atr1	………	AtrN	Class
			A
			B
			B
			C
			A
			C
			B

New Case

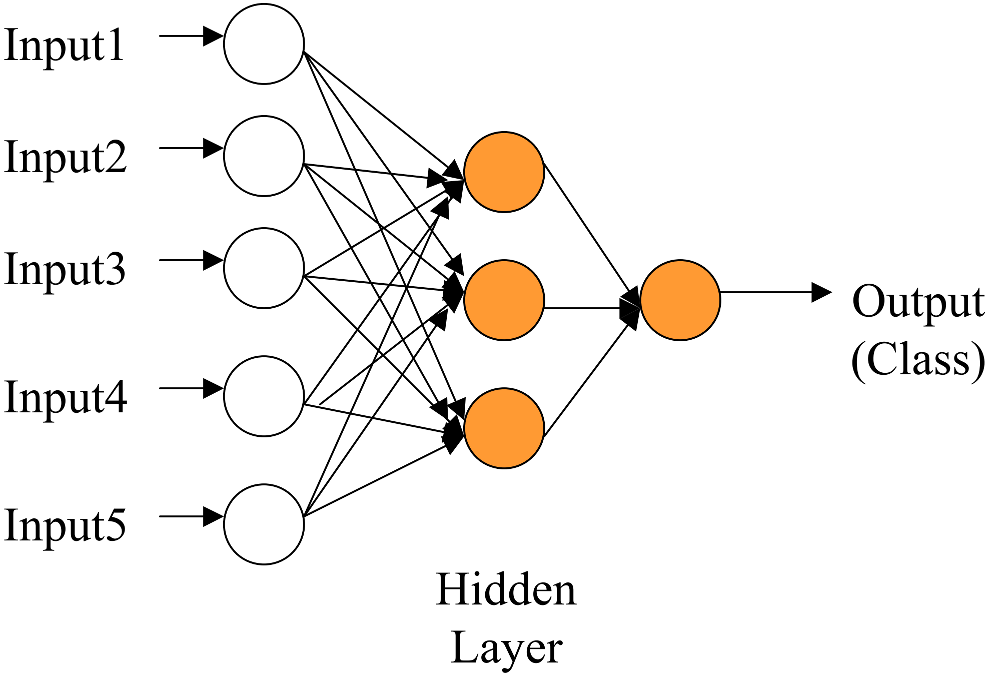
Atr1	………	AtrN

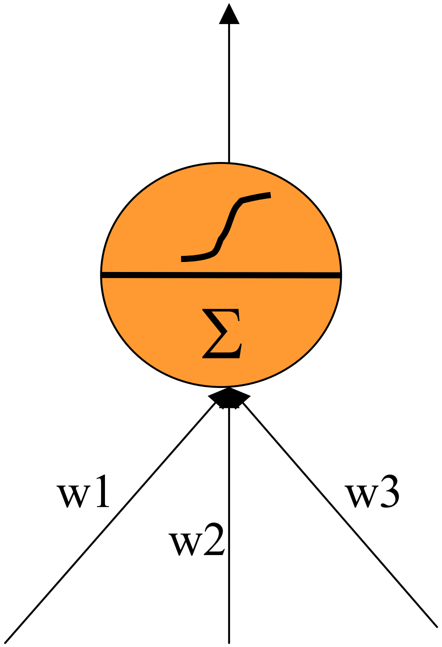
K-Nearest Neighbor

- Needs three things.
  - The set of stored cases
  - Distance Metric is used to compute distance between cases.
  - The value of *k*, the number of nearest neighbors to retrieve

- For classification :
  - *k* nearest neighbors are retrieved.
  - The class label assigned to the largest number of the *k* cases is selected.

Classification: Neural Networks





Nonlinear Optimization techniques (back propagation) used for *learning* the weights

Bayesian Classifiers

- Each attribute and class label are random variables.
- Objective is to classify a given record of attributes $(A_1, A_2, \ldots, A_n)$ to class C s.t. $P(C \mid A_1, A_2, \ldots, A_n)$ is maximal.
- Naïve Bayesian Approach:
  - Assume independence among attributes A_i.
  - Estimate $P(A_i \mid C_j)$ for all A_i and C_j.
  - New point is classified to C_j if $P(C_j) \Pi_i P(A_i| C_j)$ is maximal.
- Generic Approach based on Bayesian Networks:
  - Represent dependencies using a direct acyclic graph (child conditioned on all its parents). Class variable is a child of all the attributes.
  - Goal is to get compact and accurate representation of the joint probability distribution of all variables. Learning Bayesian Networks is an active research area.

Evaluation Criteria

Predicted Actual	C_1	C_2
C_1	a	b
C_2	c	d

$$\text{Accuracy (A)} = \frac{a+d}{a+b+c+d}$$

$$\text{Precision (P)} = \frac{a}{a+c}$$

$$\text{Recall (R)} = \frac{a}{a+b}$$

$$F = \frac{2PR}{P+R}$$

Accuracy Unsuitable for Skewed Class Distributions

Predicted Actual	C_1	C_2
C_1	0	10
C_2	0	90

A = 90/100

P = /

R = 0

F = 0

Predicted Actual	C_1	C_2
C_1	3	7
C_2	10	80

A = 83/100

P = 3/13

R = 3/10

F = 6/23

Predicted Actual	C_1	C_2
C_1	8	2
C_2	42	48

A = 56/100

P = 8/50

R= 8/10

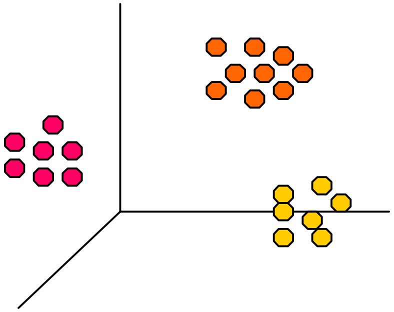
F = 4/15

Clustering Definition

- ⌘ Given a set of data points, each having a set of attributes, and a similarity measure among them, find clusters such that
  - Data points in one cluster are more similar to one another.
  - Data points in separate clusters are less similar to one another.
- ⌘ Similarity Measures:
  - Euclidean Distance
  - Jaccard Coefficient
  - Cosine Similarity
  - Other Problem-specific Measures.

Input Data for Clustering

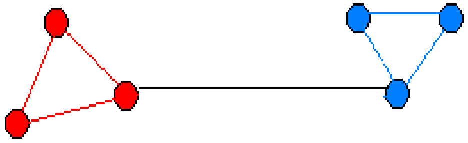
- A set of N points in an M dimensional space



OR

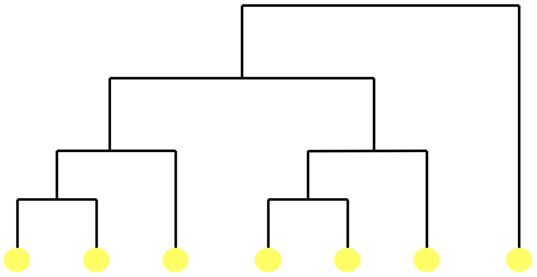
- A proximity matrix that gives the pairwise distance or similarity between points.
  - Can be viewed as a weighted graph.

	I1	I2	I3	I4	I5	I6
I1	1.00	0.70	0.80	0.00	0.00	0.00
I2	0.70	1.00	0.65	0.25	0.00	0.00
I3	0.80	0.65	1.00	0.00	0.00	0.00
I4	0.00	0.25	0.00	1.00	0.90	0.85
I5	0.00	0.00	0.00	0.90	1.00	0.95
I6	0.00	0.00	0.00	0.85	0.95	1.00



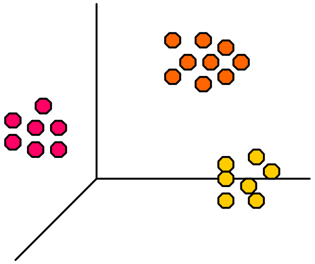
Types of Clustering: Partitional and Hierarchical

- Partitional Clustering (K-means and K-medoid) finds a one-level partitioning of the data into K disjoint groups.
- Hierarchical Clustering finds a hierarchy of nested clusters (dendogram).
  - May proceed either bottom-up (agglomerative) or top-down (divisive).
  - Uses a proximity matrix.
  - Can be viewed as operating on a proximity graph.



K-means Clustering

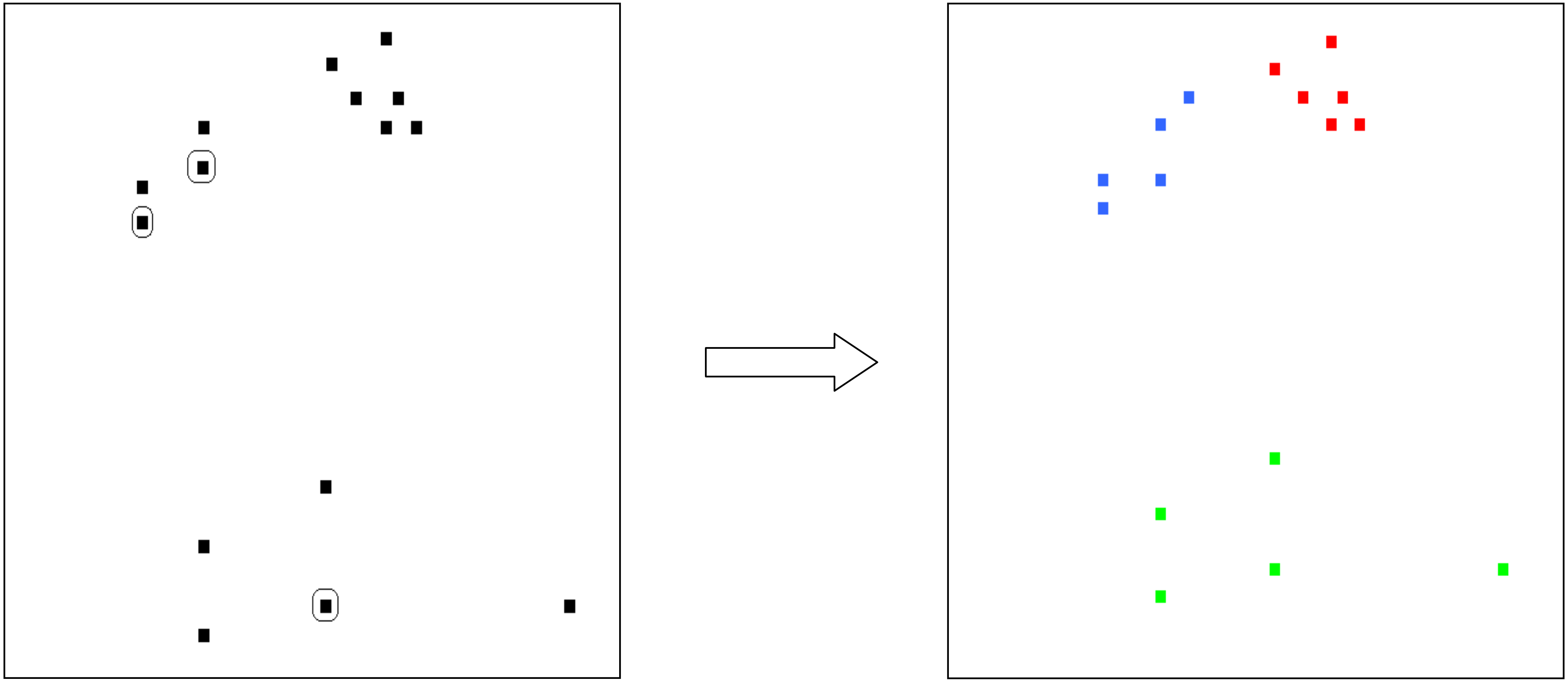
- Find a single partition of the data into K clusters such that the within cluster error, e.g., $\sum_{i=1}^{K}\sum_{\vec{x}\in C_i}\|\vec{x}-\vec{c}_i\|^2$, is minimized.
- Basic K-means Algorithm:



  1. Select K points as the initial centroids.
  2. Assign all points to the closest centroid.
  3. Recompute the centroids.
  4. Repeat steps 2 and 3 until the centroids don’t change.
- K-means is a gradient-descent algorithm that always converges - perhaps to a local minimum.

(*Clustering for Applications*, Anderberg)

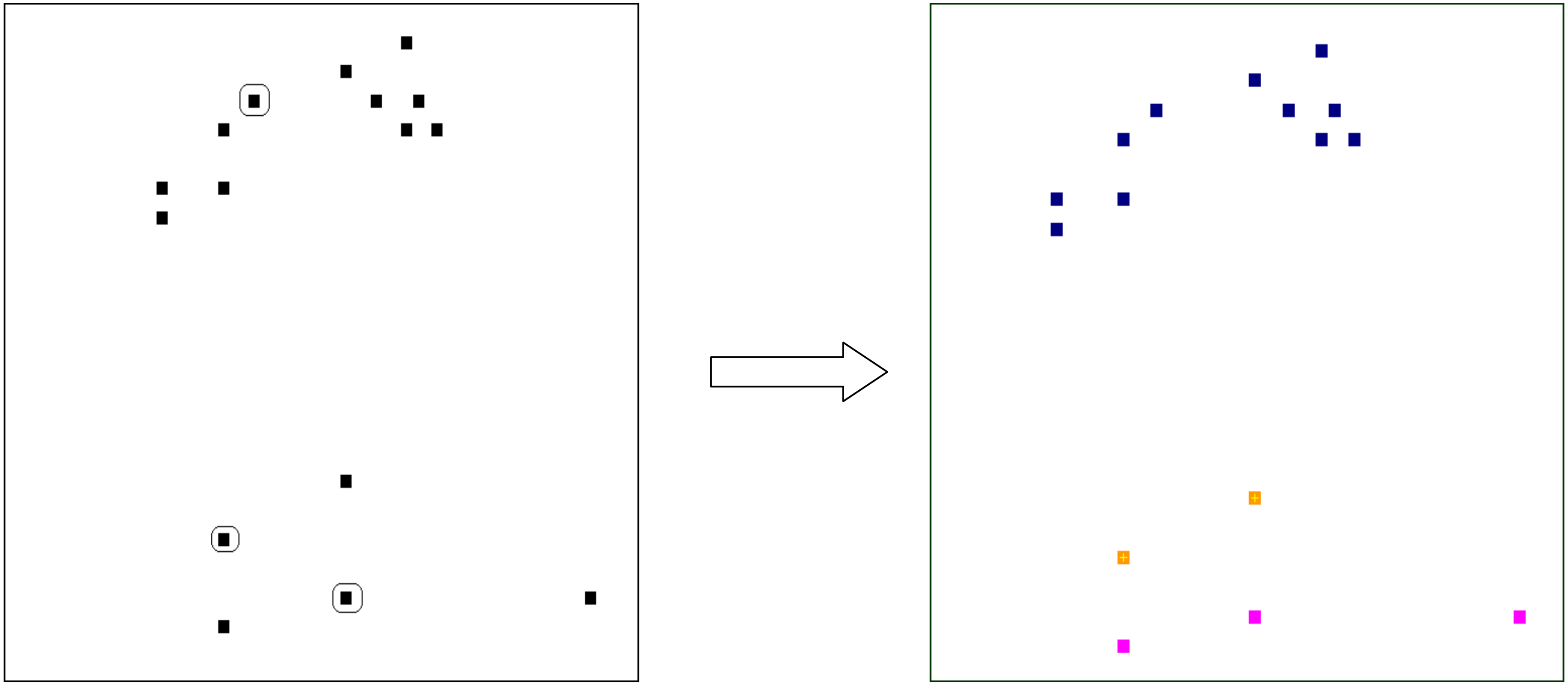
Example: Kmeans



Initial Data and Seeds

Final Clustering

Example: K-means



Initial Data and Seeds

Final Clustering

K-means: Initial Point Selection

- Bad set of initial points gives a poor solution.
- Random selection
  - Simple and efficient.
  - Initial points don't cover clusters with high probability.
  - Many runs may be needed for optimal solution.
- Choose initial points from
  - Dense regions so that the points are "well-separated."
- Many more variations on initial point selection.

K-means: How to Update Centroids

- Depends on the exact error criterion used.
- If trying to minimize the squared error,

$$\sum_{i=1}^{K} \sum_{\vec{x} \in C_i} \left\| \vec{x} - \vec{c}_i \right\|^2$$

, then the new centroid is the mean of the points in a cluster.

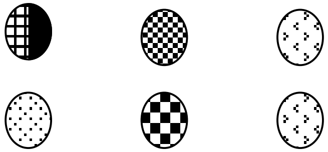
- If trying to minimize the sum of distances,

$$\sum_{i=1}^{K} \sum_{\vec{x} \in C_i} \left\| \vec{x} - \vec{c}_i \right\|$$

, then the new centroid is the median of the points in a cluster.

K-means: Pre and Post Processing

- Outliers can dominate the clustering and, in some cases, are eliminated by preprocessing.
- Post-processing attempts to "fix-up" the clustering produced by the K-means algorithm.
  - Merge clusters that are "close" to each other.
  - Split "loose" clusters that contribute most to the error.
  - Permanently eliminate "small" clusters since they may represent groups of outliers.



- Approaches are based on heuristics and require the user to choose parameter values.

K-means: Time and Space requirements

- O(MN) space since it uses just the vectors, not the proximity matrix.
  - M is the number of attributes.
  - N is the number of points.
  - Also keep track of which cluster each point belongs to and the K cluster centers.
- Time for basic K-means is O(T*K*M*N),
  - T is the number of iterations. (T is often small, 5-10, and can easily be bounded, as few changes occur after the first few iterations).

K-means: Determining the Number of Clusters

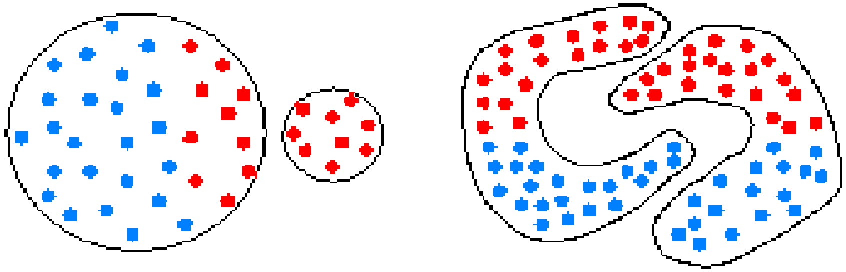
- Mostly heuristic and domain dependant approaches.

- Plot the error for 2, 3, … clusters and find the knee in the curve.

- Use domain specific knowledge and inspect the clusters for desired characteristics.

K-means: Problems and Limitations

- Based on minimizing within cluster error - a criterion that is not appropriate for many situations.
  - Unsuitable when clusters have widely different sizes or have convex shapes.



- Restricted to data in Euclidean spaces, but variants of K-means can be used for other types of data.
- Sensitive to outliers

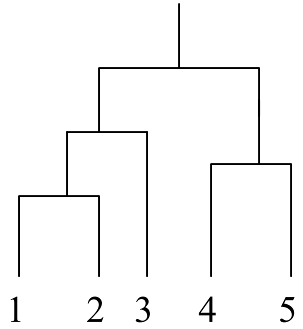
Hierarchical Clustering Algorithms

- Hierarchical Agglomerative Clustering
  1. Initially each item belongs to a single cluster.
  2. Combine the two *most similar* clusters.
  3. Repeat step 2 until there is only a single cluster.
  - Most popular approach.
- Hierarchical Divisive Clustering
  - Starting with a single cluster, divide clusters until only single item clusters remain.
  - Less popular, but equivalent in functionality.

Cluster Similarity: MIN or Single Link

- Similarity of two clusters is based on the two most similar (closest) points in the different clusters.
  - Determined by one pair of points, i.e., by one link in the proximity graph.
- Can handle non-elliptical shapes.
- Sensitive to noise and outliers.

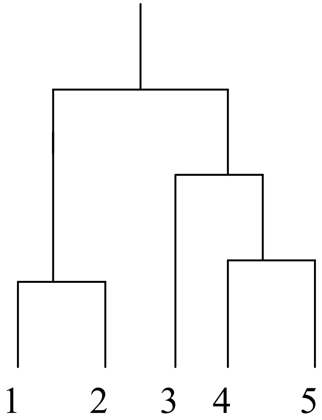
	I1	I2	I3	I4	I5
I1	1.00	0.90	0.10	0.65	0.20
I2	0.90	1.00	0.70	0.60	0.50
I3	0.10	0.70	1.00	0.40	0.30
I4	0.65	0.60	0.40	1.00	0.80
I5	0.20	0.50	0.30	0.80	1.00



Cluster Similarity: MAX or Complete Linkage

- Similarity of two clusters is based on the two least similar (most distant) points in the different clusters.
  - Determined by all pairs of points in the two clusters.
  - Tends to break large clusters.
  - Less susceptible to noise and outliers.

	I1	I2	I3	I4	I5
I1	1.00	0.90	0.10	0.65	0.20
I2	0.90	1.00	0.70	0.60	0.50
I3	0.10	0.70	1.00	0.40	0.30
I4	0.65	0.60	0.40	1.00	0.80
I5	0.20	0.50	0.30	0.80	1.00



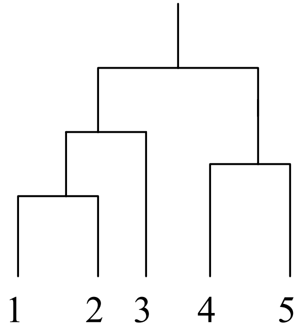
Cluster Similarity: Group Average

- Similarity of two clusters is the average of pairwise similarities between points in the two clusters.

$$\text{Similarity}(\text{Cluster}_i, \text{Cluster}_j) = \frac{\displaystyle\sum_{\substack{p_i \in \text{Cluster}_i \\ p_j \in \text{Cluster}_j}} \text{Similarity}(p_i, p_j)}{|\text{Cluster}_i| * |\text{Cluster}_j|}$$

- Compromise between Single and Complete Link.
- Need to use average connectivity for scalability since total connectivity favors large clusters.

	I1	I2	I3	I4	I5
I1	1.00	0.90	0.10	0.65	0.20
I2	0.90	1.00	0.70	0.60	0.50
I3	0.10	0.70	1.00	0.40	0.30
I4	0.65	0.60	0.40	1.00	0.80
I5	0.20	0.50	0.30	0.80	1.00

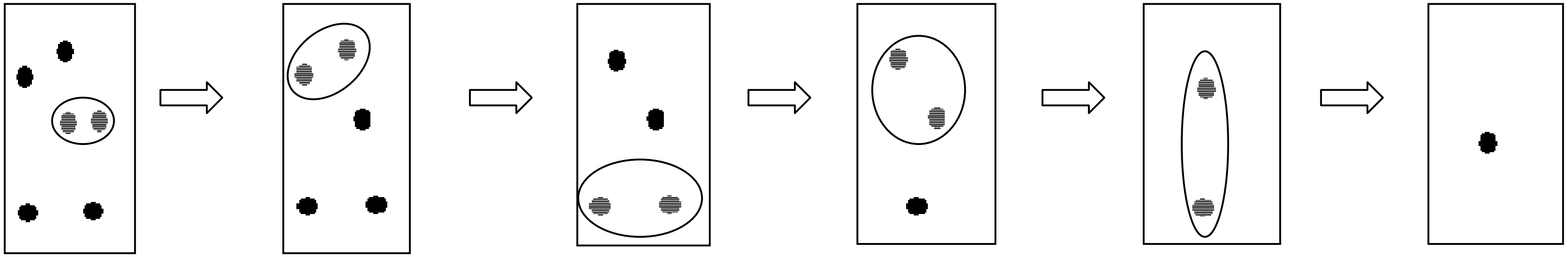


Cluster Similarity: Centroid Methods

- Similarity of two clusters is based on the distance of the centroids of the two clusters.

- Similar to K-means
  - Euclidean distance requirement
  - Problems with different sized clusters and convex shapes.

- Variations include “median” based methods.



Hierarchical Clustering: Time and Space requirements

- $O(N^2)$ space since it uses the proximity matrix.
  - N is the number of points.

- $O(N^3)$ time in many cases.
  - There are N steps and at each step the size, N^2, proximity matrix must be updated and searched.
  - By being careful, the complexity can be reduced to $O(N^2 \log(N))$ time for some approaches.

Hierarchical Clustering: Problems and Limitations

- Once a decision is made to combine two clusters, it cannot be undone.
- No objective function is directly minimized.
- Different schemes have problems with one or more of the following:
  - Sensitivity to noise and outliers.
  - Difficulty handling different sized clusters and convex shapes.
  - Breaking large clusters.

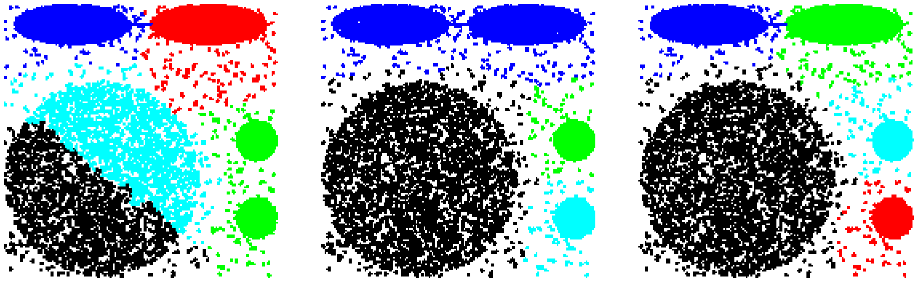
Recent Approaches: CURE

- Uses a number of points to represent a cluster.
- Representative points are found by selecting a constant number of points from a cluster and then “shrinking” them toward the center of the cluster.
- Cluster similarity is the similarity of the closest pair of representative points from different clusters.
- Shrinking representative points toward the center helps avoid problems with noise and outliers.
- CURE is better able to handle clusters of arbitrary shapes and sizes.

(*CURE*, Guha, Rastogi, Shim)

Experimental Results

CURE



a) BIRCH (centroid)

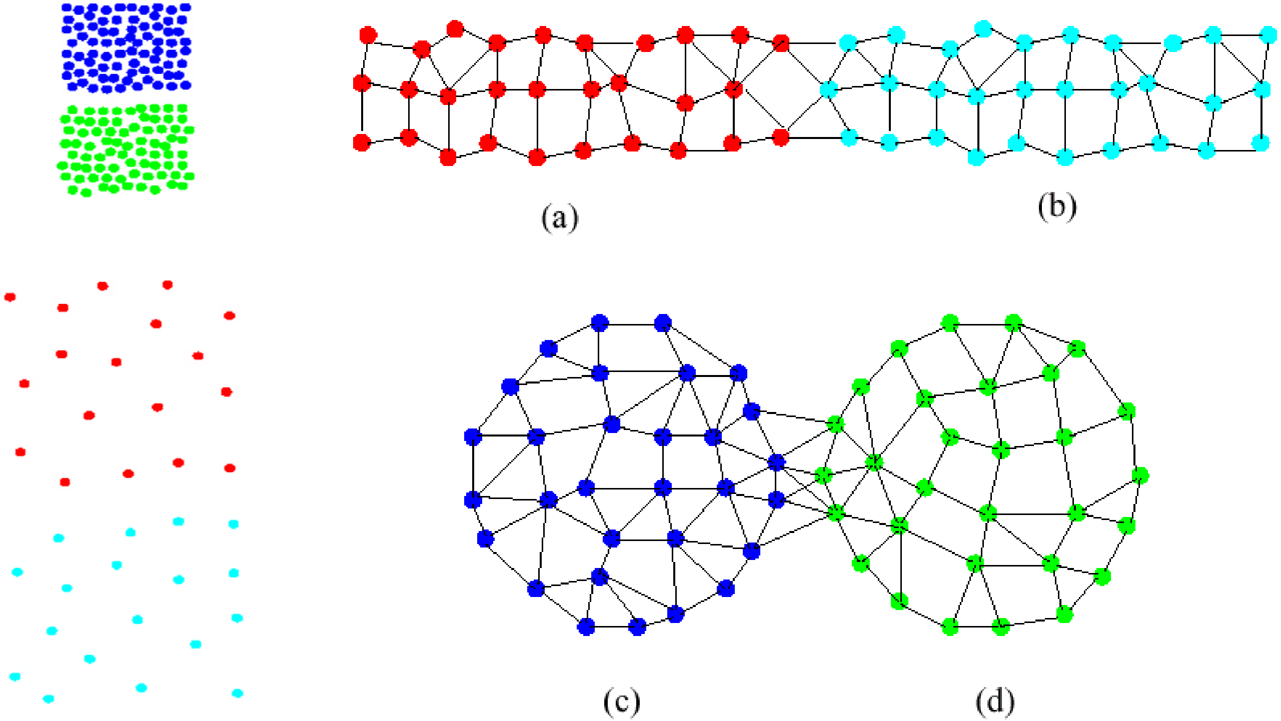
b) MST METHOD (single link)

c) CURE

Picture from *CURE*, Guha, Rastogi, Shim.

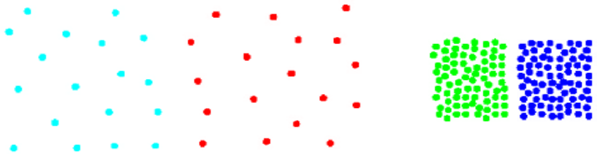
Limitations of Current Merging Schemes

- Existing merging schemes are static in nature.



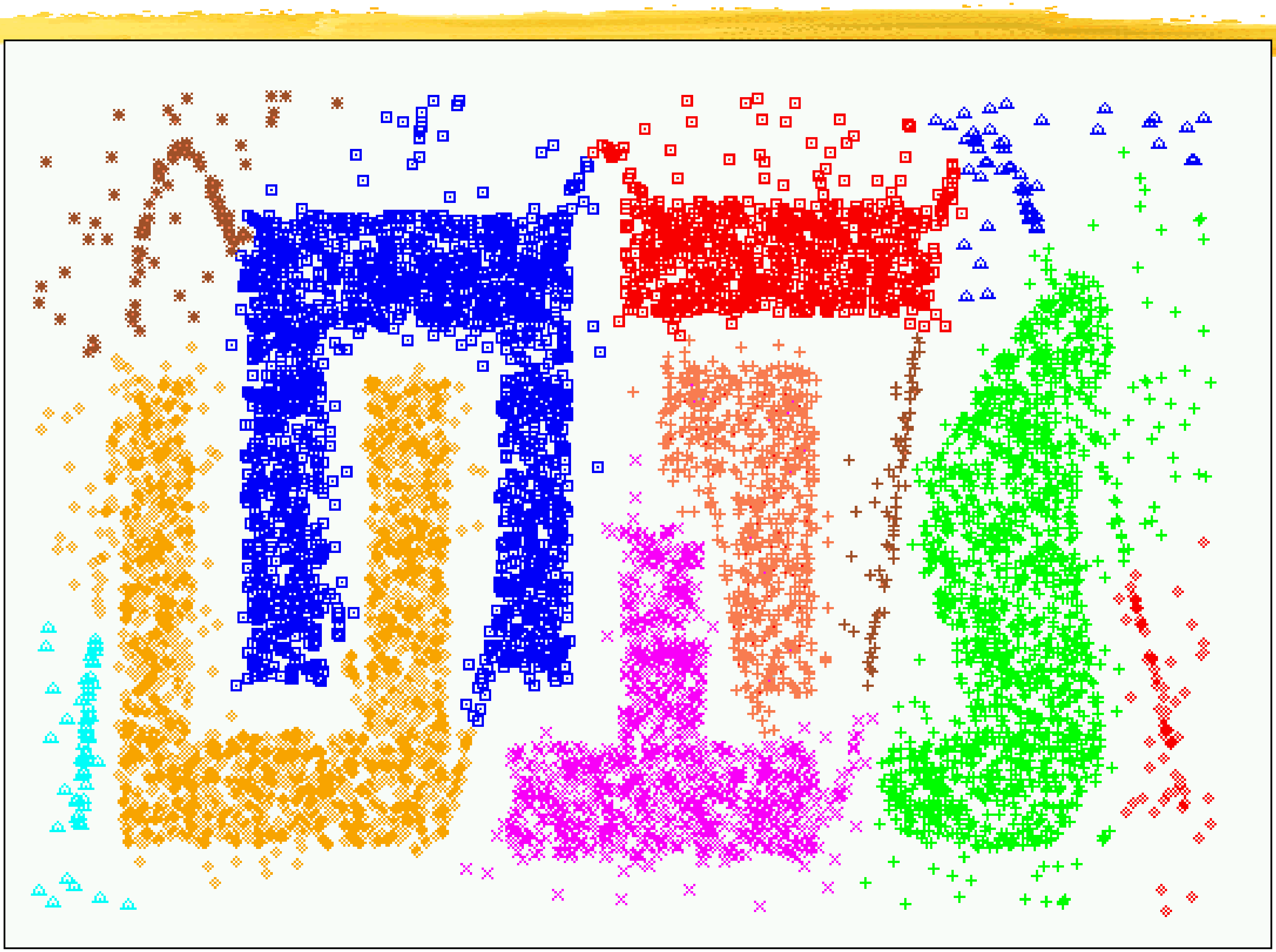
Chameleon: Clustering Using Dynamic Modeling

- Adapt to the characteristics of the data set to find the natural clusters.
- Use a dynamic model to measure the similarity between clusters.
  - Main property is the relative closeness and relative inter-connectivity of the cluster.
  - Two clusters are combined if the resulting cluster shares certain *properties* with the constituent clusters.
  - The merging scheme preserves *self-similarity*.

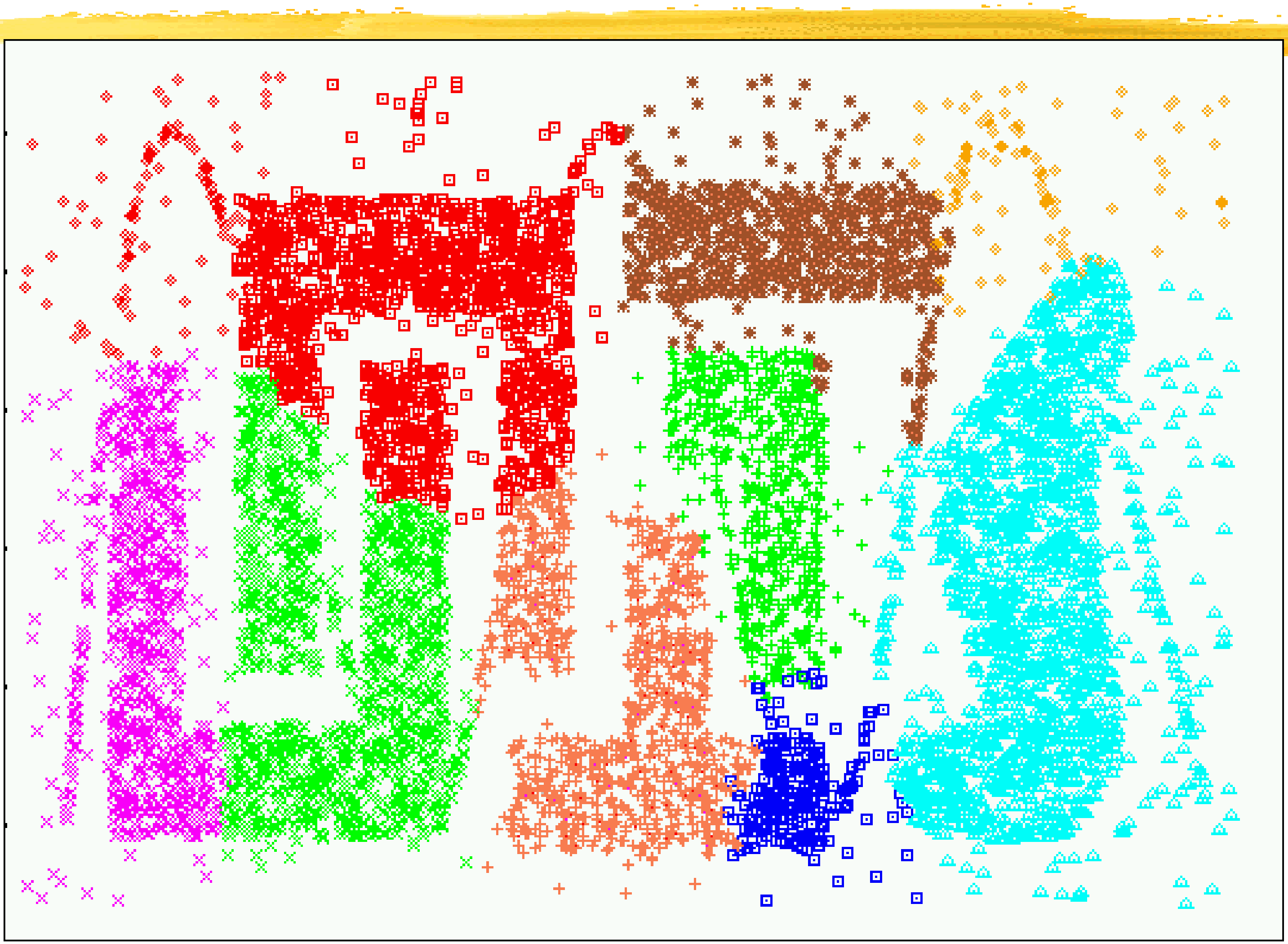


- One of the areas of application is spatial data.

Experimental Results
CHAMELEON

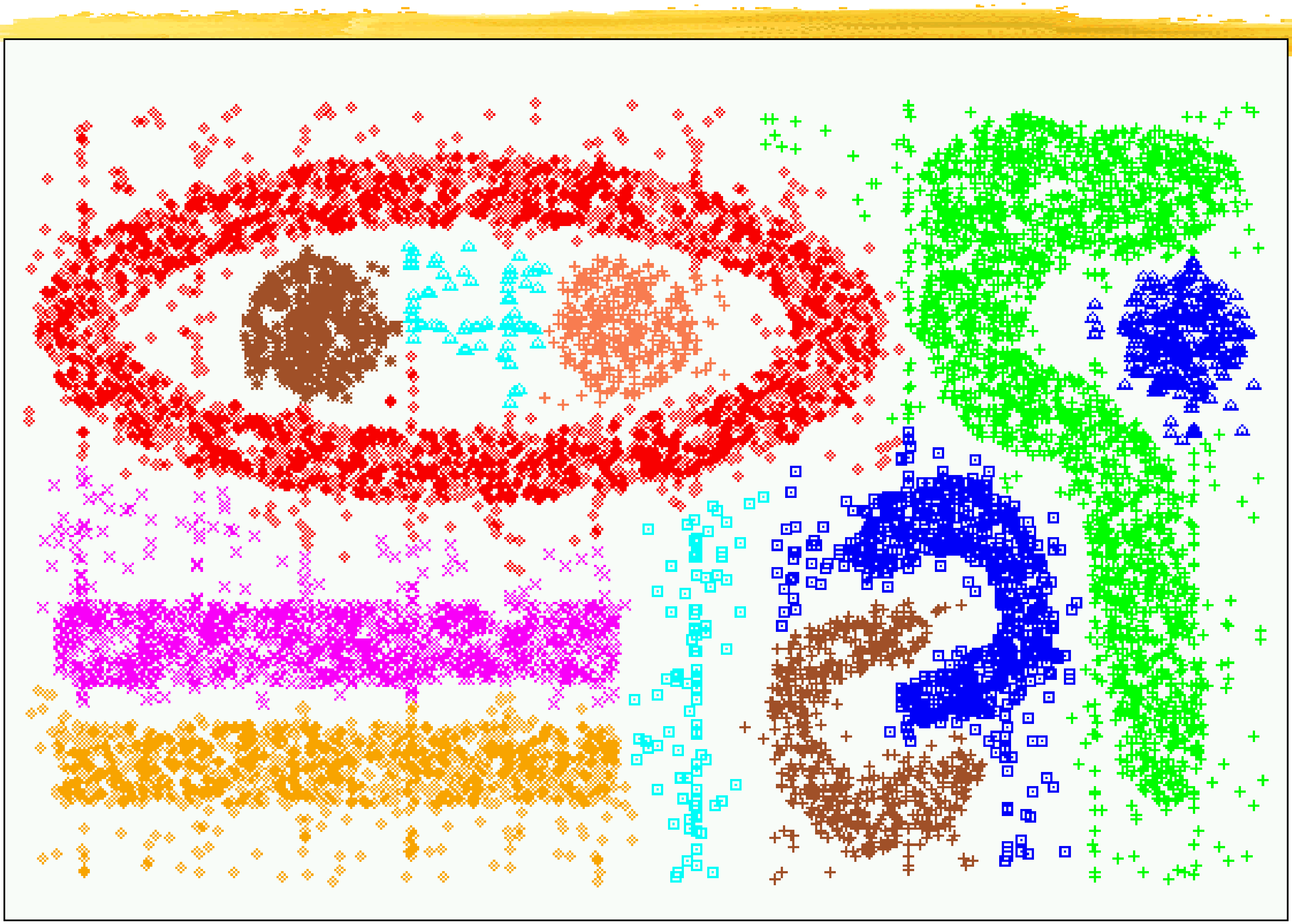


Experimental Results
CURE (*10 clusters*)



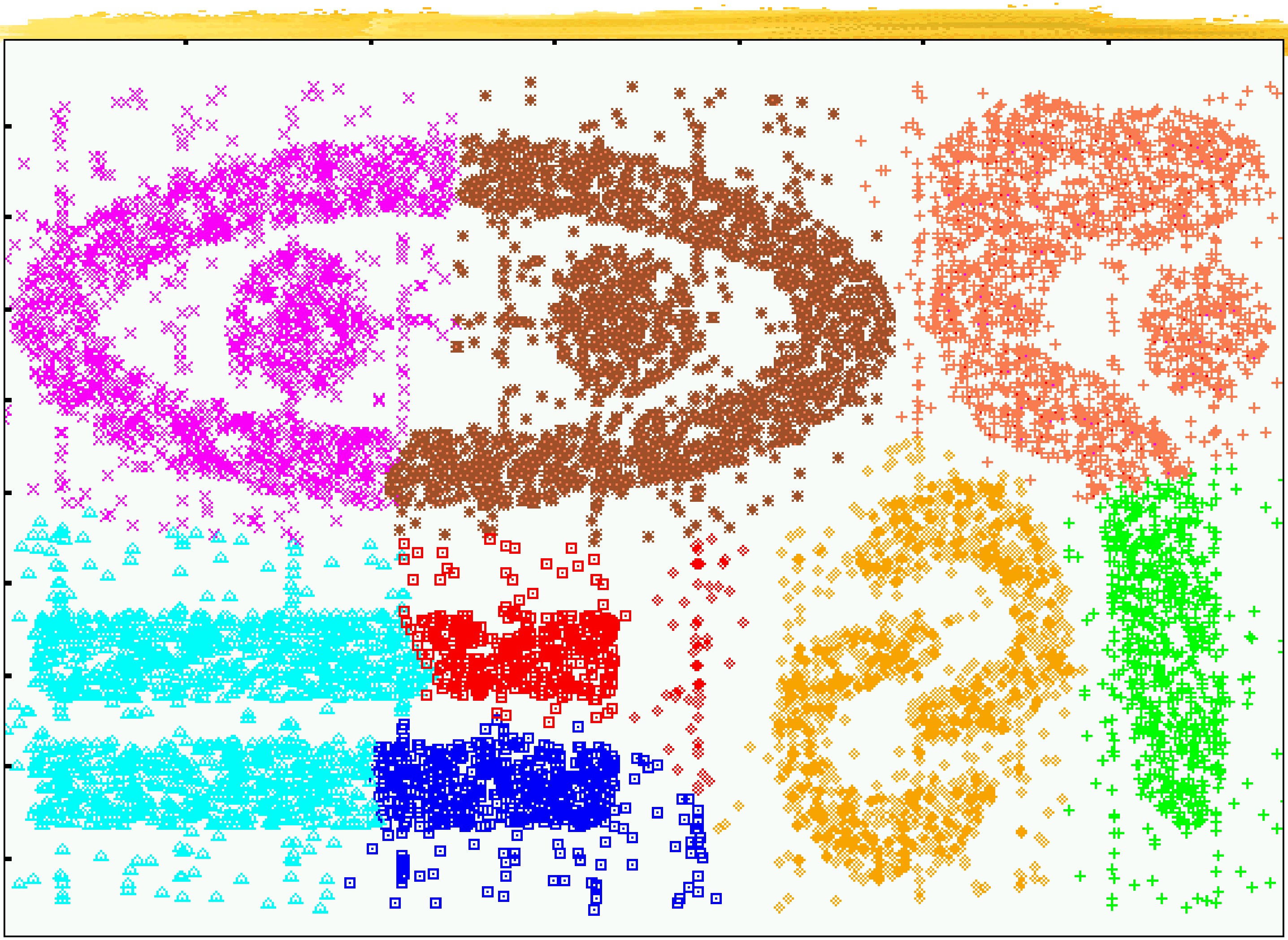
Experimental Results

CHAMELEON



Experimental Results

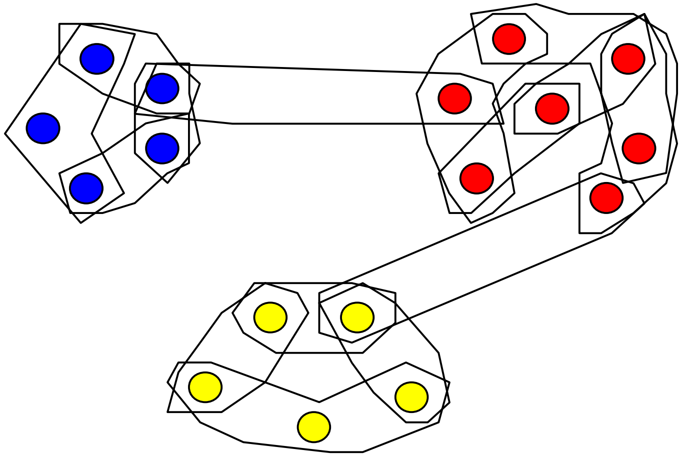
CURE (*9 clusters*)



Hypergraph-Based Clustering

Construct a hypergraph in which *related* data are connected via hyperedges.

Partition this hypergraph in a way such that each partition contains highly connected data.



How do we find related sets of data items? **Use Association Rules!**

S&P 500 Stock Data

- S&P 500 stock price movement from Jan. 1994 to Oct. 1996.

ay 2: Intel-DOWN Microsoft-DOWN Morgan-S y-UP …

ay 3: Intel-UP Microsoft-DOWN Morgan-Stanley-DOWN …

…

- Frequent item sets from the stock data.

{Intel-up, Microsoft-UP}

{Intel-down, Microsoft-DOWN, Morgan-Stanley-UP}

{Morgan-Stanley-UP, MBNA-Corp-UP, Fed-Home-Loan-UP}

…

Clustering of S&P 500 Stock Data

	Discovered Clusters	*Industry Group*
1	Applied-Matl-DOWN,Bay-Network-Down,3-COM-DOWN, Cabletron-Sys-DOWN,CISCO-DOWN,HP-DOWN, DSC-Comm-DOWN,INTEL-DOWN,LSI-Logic-DOWN, Micron-Tech-DOWN,Texas-Inst-Down,Tellabs-Inc-Down, Natl-Semiconduct-DOWN,Oracl-DOWN,SGI-DOWN, Sun-DOWN	Technology1-DOWN
2	Apple-Comp-DOWN,Autodesk-DOWN,DEC-DOWN, ADV-Micro-Device-DOWN,Andrew-Corp-DOWN, Computer-Assoc-DOWN,Circuit-City-DOWN, Compaq-DOWN, EMC-Corp-DOWN, Gen-Inst-DOWN, Motorola-DOWN,Microsoft-DOWN,Scientific-Atl-DOWN	Technology2-DOWN
3	Fannie-Mae-DOWN,Fed-Home-Loan-DOWN, MBNA-Corp-DOWN,Morgan-Stanley-DOWN	Financial-DOWN
4	Baker-Hughes-UP,Dresser-Inds-UP,Halliburton-HLD-UP, Louisiana-Land-UP,Phillips-Petro-UP,Unocal-UP, Schlumberger-UP	Oil-UP
5	Barrick-Gold-UP,Echo-Bay-Mines-UP Homestake-Mining-UP,Newmont-Mining-UP, Placer-Dome-Inc-UP	Gold-UP
6	Alcan-Aluminum-DOWN,Asarco-Inc-DOWN, Cyprus-Amax-Min-DOWN,Inland-Steel-Inc-Down, Inco-LTD-DOWN,Nucor-Corp-DOWN,Praxair-Inc-DOWN, Reynolds-Metals-DOWN,Stone-Container-DOWN, USX-US-Steel-DOWN	Metal-DOWN

Other clusters found: Bank, Paper/Lumber, Motor/Machinery, Retail, Telecommunication, Tech/Electronics

Word Clusters Using Hypergraph-Based Method

Cluster 1	Cluster 2	Cluster 3	Cluster 4	Cluster 5
http	access	act	data	action
internet	approach	busi	engineer	administrate
mov	comput	check	includes	agenci
please	electron	enforc	manag	complianc
site	goal	feder	network	establish
web	manufactur	follow	services	health
ww	power	govern	softwar	law
	step	informate	support	laws
		page	systems	nation
		public	technologi	offic
			wide	regulations

Other Clustering Approaches

- Modeling clusters as a “mixture” of Multivariate Normal Distributions. (Raftery and Fraley)
- Bayesian Approaches (*AutoClass*, Cheeseman)
- Density-Based Clustering (*DB-SCAN*, Kriegel)
- Neural Network Approaches (*SOM*, Kohonen)
- Subspace Clustering (CLIQUE, Agrawal)
- Many, many other variations and combinations of approaches.

Other Important Topics

- Dimensionality Reduction
  - Latent Semantic Indexing (LSI)
  - Principal Component Analysis (PCA)
- Feature transformation.
  - Normalizing features to the same scale by subtracting the mean and dividing by the standard deviation.
- Feature Selection
  - As in classification, not all features are equally important.

References

Book References:

[1] Hillol Kargupta and Philip Chan (Edotors), *Advances in Distributed and Parallel Knowledge Discovery*, AAAI Press, 2000.

[2] Usama Fayyad, Gregory Piatetsky-Shapiro, Padhraic Smyth, and Ramasamy Uthurasamy (eds.), *Advances in Knowledge Discovery and Data Mining*, AAAI Press/ The MIT Press, 1996.

[3] A. K. Jain and R. C. Dubes, *Algorithms for Clustering Data*, Prentice Hall, 1988.

[4] Michael Anderberg, *Clustering for Applications.* Academic Press, 1973.

[5] Jaiwei Han and Micheline Kamber, *Data Mining: Concepts and Techniques*, Morgan Kaufman, 2001.

[6] Robert L. Grossman, Chandrika Kamath, Philip Kegelmeyer,Vipin Kumar, and Raju Namburu (Editors), *Data Mining for Scientific and Engineering Applications,* Kluwer Academic Publishers, 2001.

[7] Michael Berry and Gordon Linoff, *Data Mining Techniques (For Marketing, Sales, and Customer Support)*, John Wiley & Sons, 1997.

[8] Kaufman and Rousseeuw, *Finding Groups in Data: An Introduction to Cluster Analysis,* Wiley, 1990.

[9] Vipin Kumar, Ananth Grama, Anshul Gupta, and George Karypis, *Introduction to Parallel Computing: Algorithm Design and Analysis*, Benjamin Cummings/Addison Wesley, Redwood City, 1994.

References

Book References:

[10] Tom M. Mitchell, *Machine Learning*, WCB/McGraw-Hill, 1997.**[8]** Alex Freitas and Simon Lavington, *Mining Very Large Databases with Parallel Processing*, Kluwer Academic Publishers, 1998.

[11] Sholom M. Weiss and Nitin Indurkhya, *Predictive Data Mining (a practical guide)*, Morgan Kaufmann Publishers,1998.

[12] David J. Hand, Heikki Mannila and Padhraic Smyth, *Principles of Data Mining*, The MIT Press, 2001.

[13] J. Ross Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann, 1993.

[14] T. Kohonen, *Self-Organizing Maps.*, Second Extended Edition, Springer Series in Information Sciences, Vol. 30, Springer, Berlin, Heidelberg, New York, 1997.

References…

Research Paper References:

[1] M. Mehta, R. Agarwal, and J. Rissanen, *SLIQ: A Fast Scalable Classifier for Data Mining*, Proc. Of the fifth Int. Conf. On Extending Database Technology (EDBT), Avignon, France, 1996.

[2] J. Shafer, R. Agrawal, and M. Mehta, *SPRINT: A Scalable Parallel Classifier for Data Mining*, Proc. 22nd Int. Conf. On Very Large Databases, Mumbai, India, 1996.

[3] A. Srivastava, E.H. Han, V. Kumar, and V.Singh, *Parallel Formulations of Decision-Tree Classification Algorithms*, Proc. 12th International Parallel Processing Symposium (IPPS), Orlando, 1998.

[4] M.Joshi, G.Karypis, and V. Kumar, *ScalParC: A New Scalable and Efficient Parallel Classification Algorithms for Mining Large Datasets*, Proc. 12th International Parallel Processing Symposium (IPPS), Orlando, 1998.

[5] N. Friedman, D. Geiger, and M. Goldszmidt, *Bayesian Network Classifiers,* Machine Learning 29:131--163, 1997.

[6] R. Agrawal, T.Imielinski, and A.Swami, *Mining Association Rules Between Sets of Items in Large Databases,* Proc. 1993 ACM-SIGMOD Int.Conf. On Management of Data, Washington, D.C., 1993.

[7] R. Agrawal and R. Srikant, *Fast Algorithms for Mining Association Rules*, Proc. Of 20th VLDB Conference, 1994.

References…

[8] R. Agrawal and J.C. Shafer, *Parallel Mining of Association Rules*, IEEE Trans. On Knowledge and Data Eng., 8(6):962-969, December 1996.

[9] E.H.Han, G.Karypis, and V.Kumar, *Scalable Parallel Data Mining for Association Rules*, Proc. 1997 ACM-SIGMOD Int. Conf. On Management of Data, Tucson, Arizona, 1997.

[10] R. Srikant and R. Agrawal, *Mining Sequential Patterns: Generalizations and Performance Improvements*, Proc. Of 5th Int. Conf. On Extending Database Technology (EDBT), Avignon, France, 1996.

[11] M. Joshi, G. Karypis, and V. Kumar, *Parallel Algorithms for Sequential Associations: Issues and Challenges*, Mini-symposium Talk at Ninth SIAM International Conference on Parallel Processing (PP'99), San Antonio, 1999.

[12] George Karypis, Eui-Hong (Sam) Han, and Vipin Kumar, *CHAMELEON: A Hierarchical Clustering Algorithm Using Dynamic Modeling (1999),* IEEE Computer, Vol. 32, No. 8, August, 1999. pp. 68-75.

[13] George Karypis, Eui-Hong (Sam) Han, and Vipin Kumar, *Multilevel Refinement for Hierarchical Clustering (1999).*, Technical Report # 99-020.

[14] Daniel Boley, Maria Gini, Robert Gross, Eui-Hong (Sam) Han, Kyle Hastings, George Karypis, Vipin Kumar, Bamshad Mobasher, and Jerome Moore, *Partitioning-Based Clustering for Web Document Categorization (1999).* To appear in Decision Support Systems Journal.

References…

[15] Eui-Hong (Sam) Han, George Karypis, Vipin Kumar and B. Mobasher, *Hypergraph Based Clustering in High-Dimensional Data Sets: A Summary of Results (1998).* Bulletin of the Technical Committee on Data Engineering, Vol. 21, No. 1, 1998.

[16] K. C. Gowda and G. Krishna, Agglomerative Clustering Using the Concept of Mutual Nearest Neighborhood, Pattern Recognition, Vol. 10, pp. 105-112, 1978.

[17] Rakesh Agrawal, Johannes Gehrke, Dimitrios Gunopulos, Prabhakar Raghavan, Automatic Subspace Clustering of High Dimensional Data for Data Mining Applications, *Proc. of the ACM SIGMOD Int'l Conference on Management of Data*, Seattle, Washington, June 1998.

[18] Peter Cheeseman and John Stutz, "Bayesian Classification (AutoClass): Theory and Results", in U. M. Fayyad, G. Piatetsky-Shapiro, P. Smith, and R. Uthurusamy (eds.), "Advances in Knowledge Discovery and Data Mining", pp. 153-180, AAAI/MIT Press, 1996.

[19] Tian Zhang, Raghu Ramakrishnan, Miron Livny, "BIRCH: An Efficient Data Clustering Method for Very Large Databases", Proc. of ACM SIGMOD Int'l Conf. on Data Management, Canada, June 1996.

References…

[20] Venkatesh Ganti, Raghu Ramakrishnan, and Johannes Gehrke, "Clustering Large Datasets in Arbitrary Metric Spaces", Proceedings of IEEE Conference on Data Engineering, Australia, 1999.

[21] Jarvis and E. A. Patrick, Clustering Using a Similarity Measure Based on Shared Nearest Neighbors, IEEE Transactions on Computers, Vol. C-22, No. 11, November, 1973.

[22] Sudipto Guha, Rajeev Rastogi, Kyuseok Shim, "CURE: An Efficient Clustering Algorithm for Large Databases", ACM SIGMOD Conference, 1998, pp. 73-84.

[23] Sander J., Ester M., Kriegel H.-P., Xu X., *Density-Based Clustering in Spatial Databases: The Algorithm GDBSCAN and its Applications,* Data Mining and Knowledge Discovery, An International Journal, Kluwer Academic Publishers, Vol. 2, No. 2, 1998, pp. 169-194.

[24] R. Ng and J. Han., Efficient and effective clustering method for spatial data mining, In *Proc. 1994 Int. Conf. Very Large Data Bases*, pp. 144--155, Santiago, Chile, September, 1994.

[25] Chris Fraley and Adrian E. Raftery, How many clusters? Which clustering method? - Answers via Model-Based Cluster Analysis, *Computer Journal*, 41(1998):578-588.

References…

[26] Sudipto Guha, Rajeev Rastogi, Kyuseok Shim, "ROCK: A Robust Clustering Algorithm for Categorical Attributes", Proceedings of IEEE Conference on Data Engineering, Australia, 1999.

[27] Paul S. Bradley, Usama M. Fayyad and Cory A. Reina, Scaling Clustering Algorithms to Large Databases, Proceedings of the 4th International Conference on Knowledge Discovery & Data Mining (KDD98).

Over 100 More Data Mining References are available at
http://www.cs.umn.edu/~mjoshi/dmrefs.html

Our group's papers are available via **http://www.cs.umn.edu/~kumar**