Integration of Classification and Pattern Mining: A Discriminative and Frequent Pattern-Based Approach

Hong Cheng
Chinese Univ. of Hong Kong
hcheng@se.cuhk.edu.hk

Jiawei Han
Univ. of Illinois at U-C
hanj@cs.uiuc.edu

Xifeng Yan
Univ. of California at Santa Barbara
xyan@cs.ucsb.edu

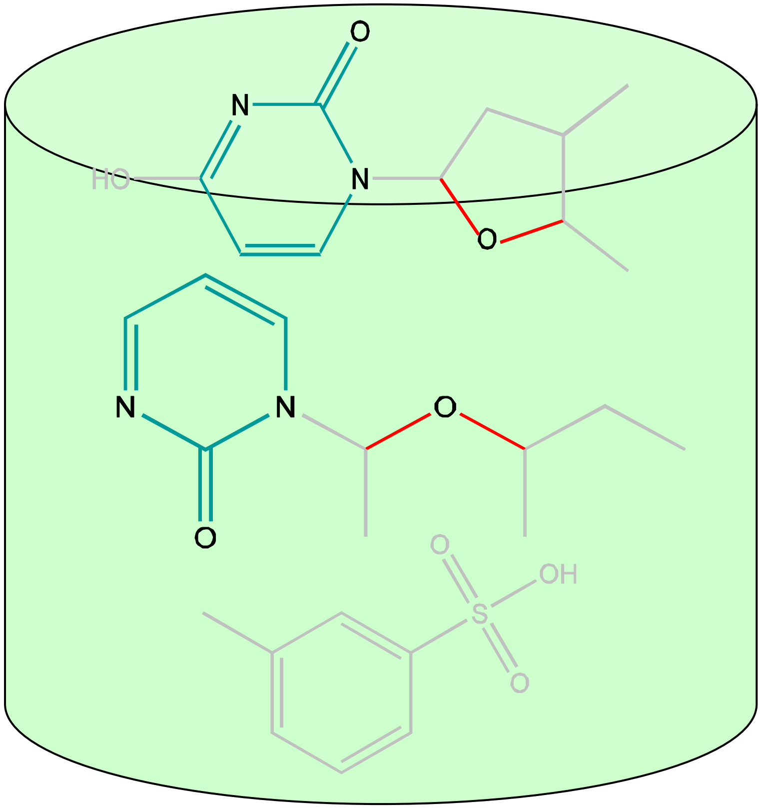
Philip S. Yu
Univ. of Illinois at Chicago
psyu@cs.uic.edu

Tutorial Outline

- Frequent Pattern Mining
- Classification Overview
- Associative Classification
- Substructure-Based Graph Classification
- Direct Mining of Discriminative Patterns
- Integration with Other Machine Learning Techniques
- Conclusions and Future Directions

Frequent Patterns

TID	Items bought
10	Beer, Nuts, Diaper
20	Beer, Coffee, Diaper
30	Beer, Diaper, Eggs
40	Nuts, Eggs, Milk
50	Nuts, Diaper, Eggs, Beer



Frequent Itemsets

Frequent Graphs

frequent pattern: support no less than min_sup

min_sup: the minimum frequency threshold

Major Mining Methodologies

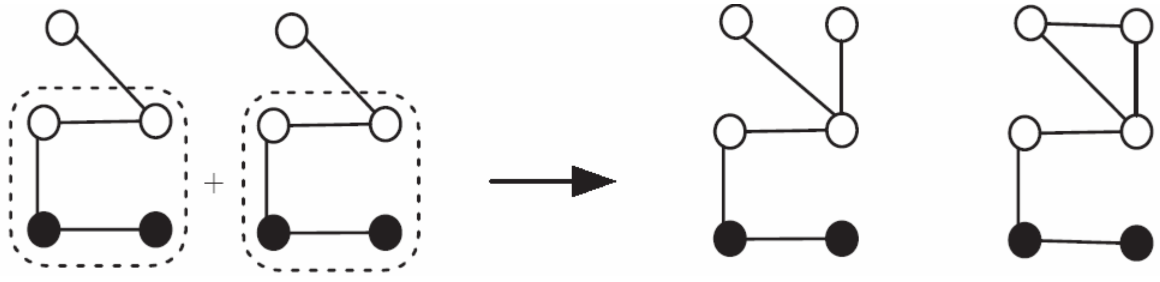
- **Apriori approach**
  - Candidate generate-and-test, breadth-first search
  - Apriori, GSP, AGM, FSG, PATH, FFSM
- **Pattern-growth approach**
  - Divide-and-conquer, depth-first search
  - FP-Growth, PrefixSpan, MoFa, gSpan, Gaston
- **Vertical data approach**
  - ID list intersection with (item: tid list) representation
  - Eclat, CHARM, SPADE

Apriori Approach

- Join two size-k patterns to a size-(k+1) pattern

- Itemset: {a,b,c} + {a,b,d} → {a,b,c,d}

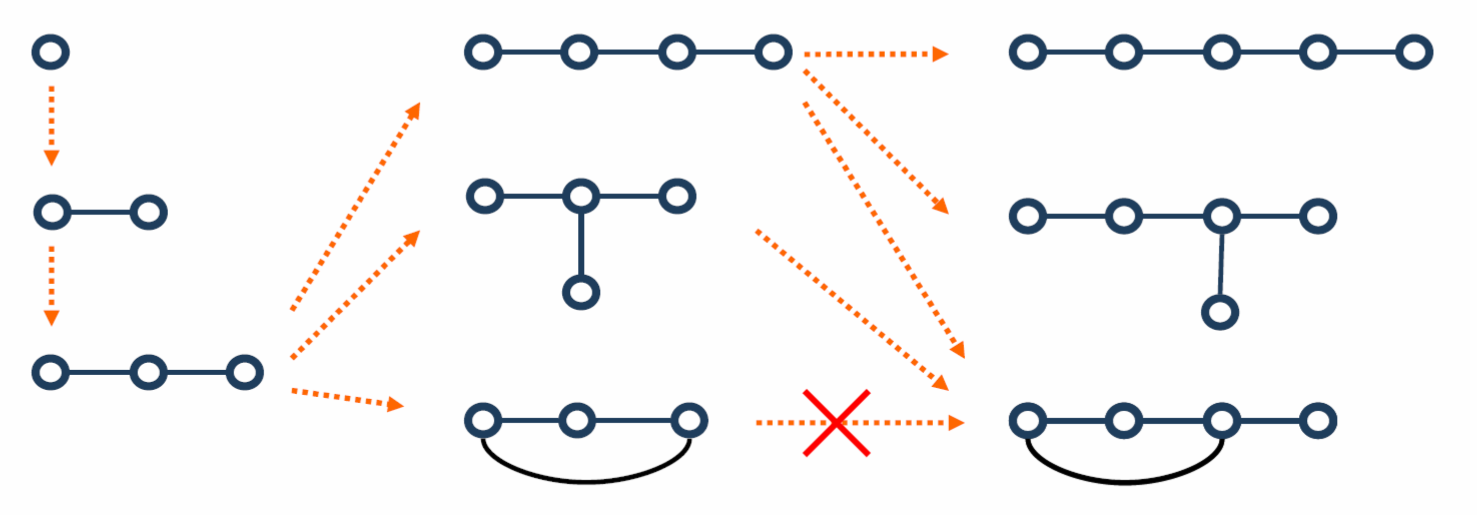
- Graph:



Pattern Growth Approach

- Depth-first search, grow a size-k pattern to size-(k+1) one by adding one element

- Frequent subgraph mining



Vertical Data Approach

- Major operation: transaction list intersection

$$t(AB) = t(A) \bigcap t(B)$$

Item	Transaction id
A	t1, t2, t3,…
B	t2, t3, t4,…
C	t1, t3, t4,…
…	…

Mining High Dimensional Data

- **High dimensional data**
  - Microarray data with 10,000 – 100,000 columns
- **Row enumeration rather than column enumeration**
  - CARPENTER [Pan et al., KDD'03]
  - COBBLER [Pan et al., SSDBM'04]
  - TD-Close [Liu et al., SDM'06]

Mining Colossal Patterns [Zhu et al., ICDE'07]

- **Mining colossal patterns: challenges**
  - A small number of colossal (i.e., large) patterns, but a very large number of mid-sized patterns
  - If the mining of mid-sized patterns is explosive in size, there is no hope to find colossal patterns efficiently by insisting "complete set" mining philosophy
- **A pattern-fusion approach**
  - Jump out of the swamp of mid-sized results and quickly reach colossal patterns
  - Fuse small patterns to large ones directly

Impact to Other Data Analysis Tasks

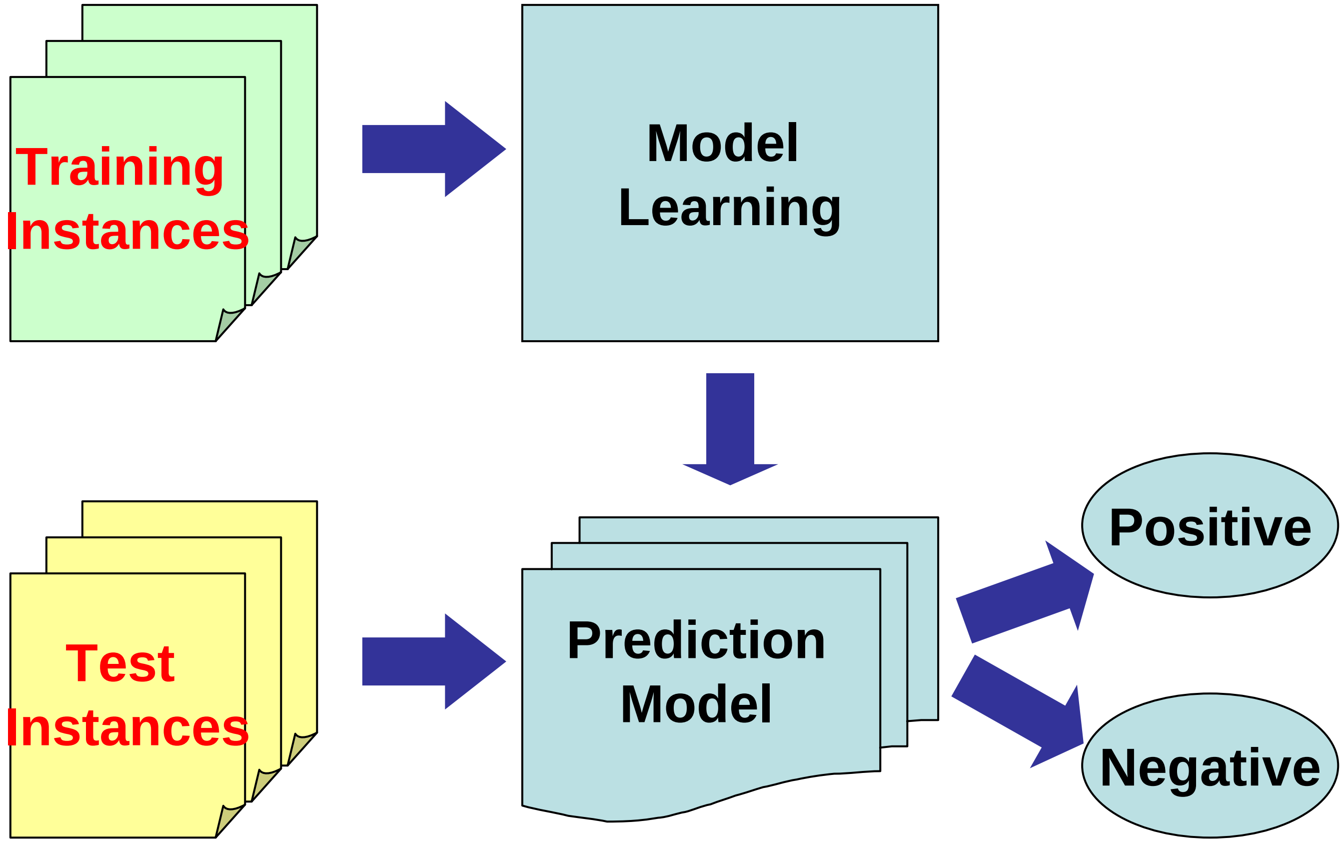
- **Association and correlation analysis**
  - Association: support and confidence
  - Correlation: lift, chi-square, cosine, all_confidence, coherence
  - A comparative study [Tan, Kumar and Srivastava, KDD'02]

- **Frequent pattern-based Indexing**
  - Sequence Indexing [Cheng, Yan and Han, SDM'05]
  - Graph Indexing [Yan, Yu and Han, SIGMOD'04; Cheng et al., SIGMOD'07; Chen et al., VLDB'07]

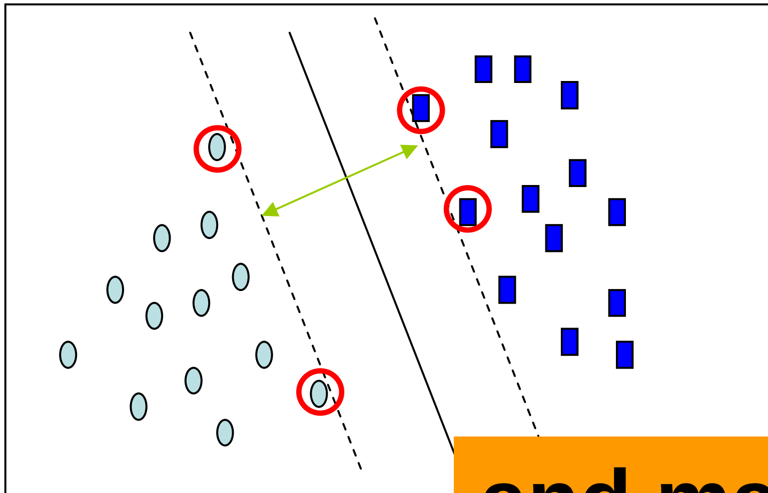
- **Frequent pattern-based clustering**
  - Subspace clustering with frequent itemsets
    - CLIQUE [Agrawal et al., SIGMOD'98]
    - ENCLUS [Cheng, Fu and Zhang, KDD'99]
    - pCluster [Wang et al., SIGMOD'02]

- **Frequent pattern-based classification**
  - Build classifiers with frequent patterns (our focus in this talk!)

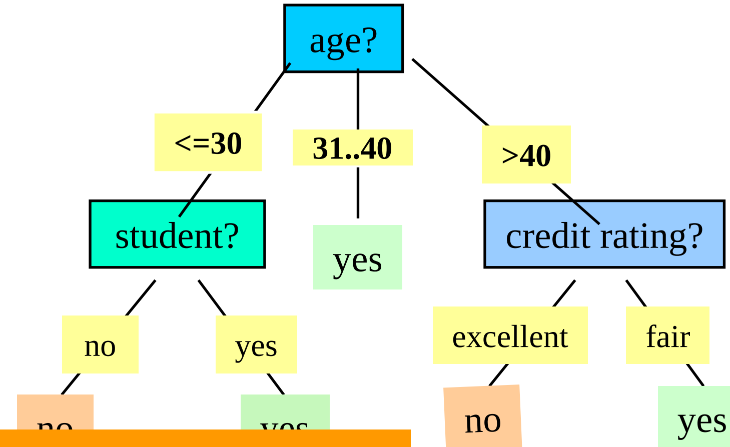
Classification Overview



Existing Classification Methods

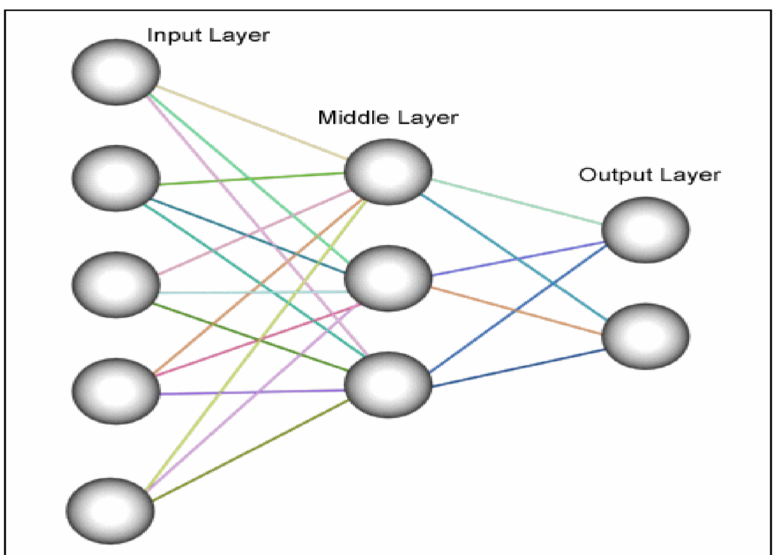


Support Vector

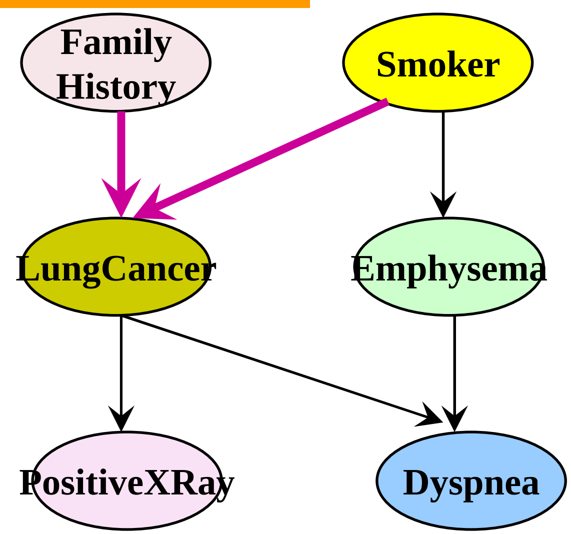


n Tree

and many more…

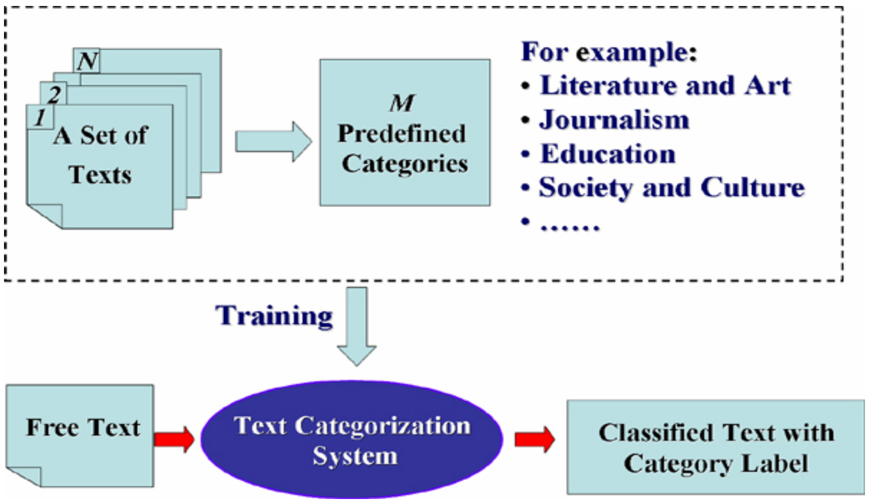


Neural Network

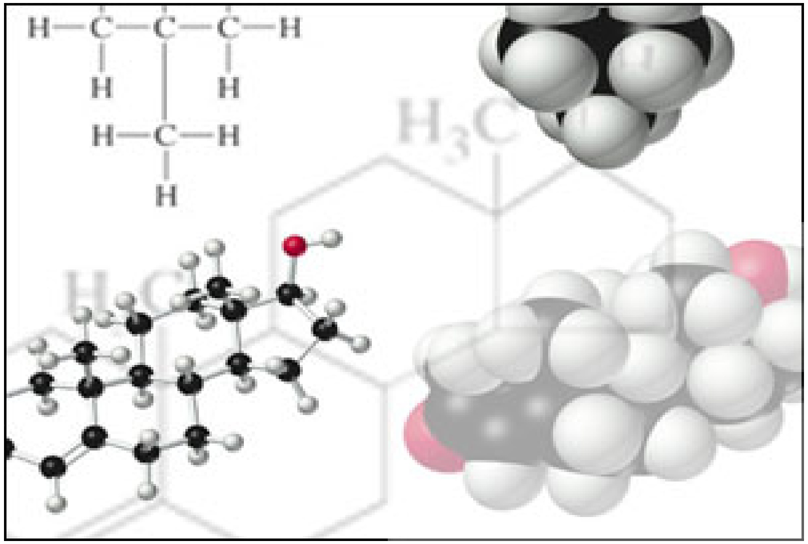


Bayesian Network

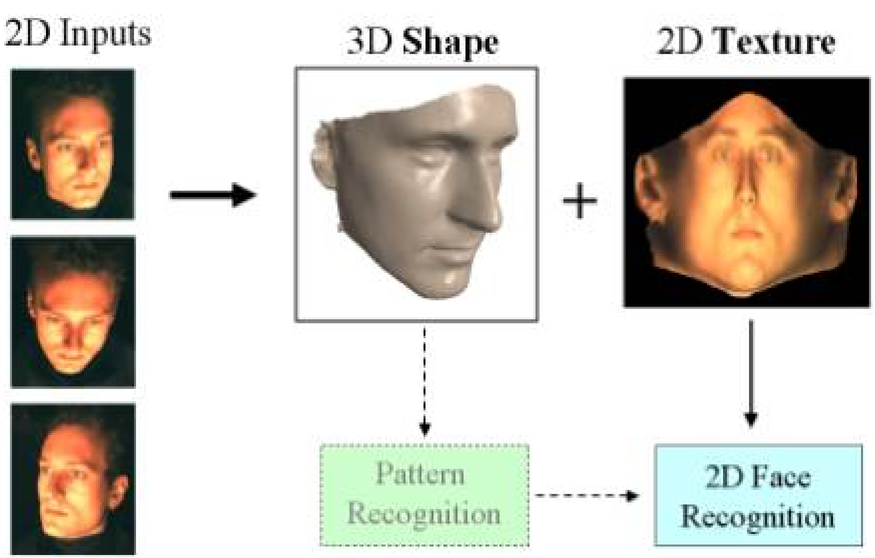
Many Classification Applications



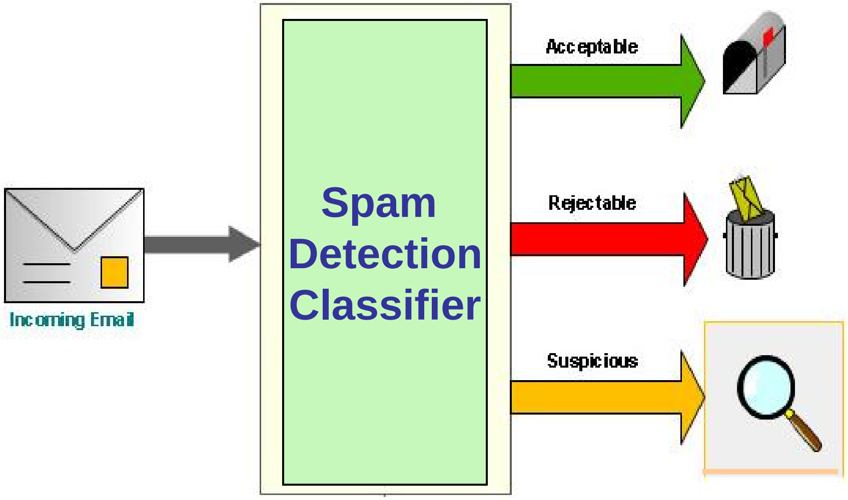
Text Categorization



Drug Design

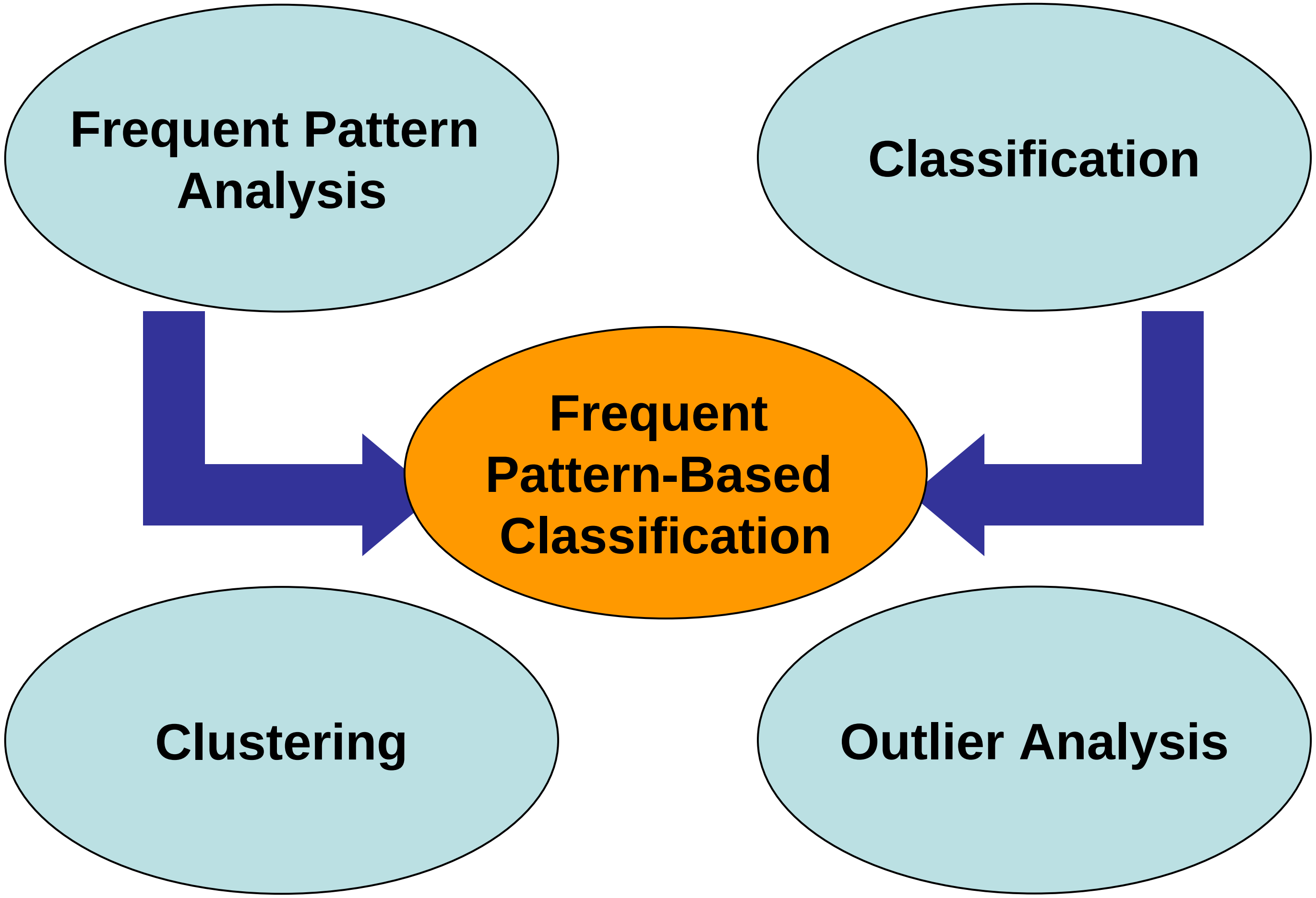


Face Recognition



Spam Detection

Major Data Mining Themes



Why Pattern-Based Classification?

- **Feature construction**
  - Higher order
  - Compact
  - Discriminative

- **Complex data modeling**
  - Sequences
  - Graphs
  - Semi-structured/unstructured data

Feature Construction

Phrases vs. single words

… the long-awaited **Apple iPhone** has arrived **…**

… the best **apple pie** recipe **…**

disambiguation

Sequences vs. single commands

… **login**, changeDir, **delFile, appendFile**, **logout** **…**

… **login**, setFileType, storeFile, **logout** **…**

temporal order

higher order, discriminative

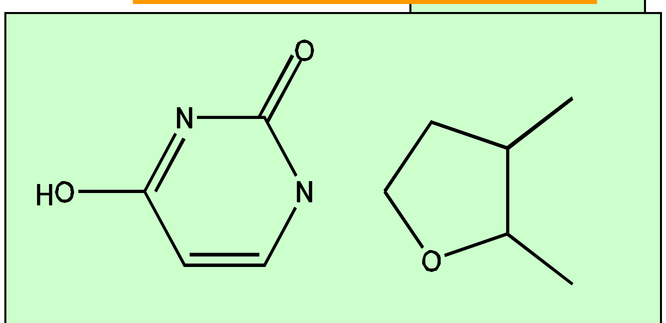
Complex Data Modeling

age	income	credit	Buy?
25	80k	good	Yes
50	200k	good	No
32	50k	fair	No

Predefined Feature vector

Classification model

Prediction Model



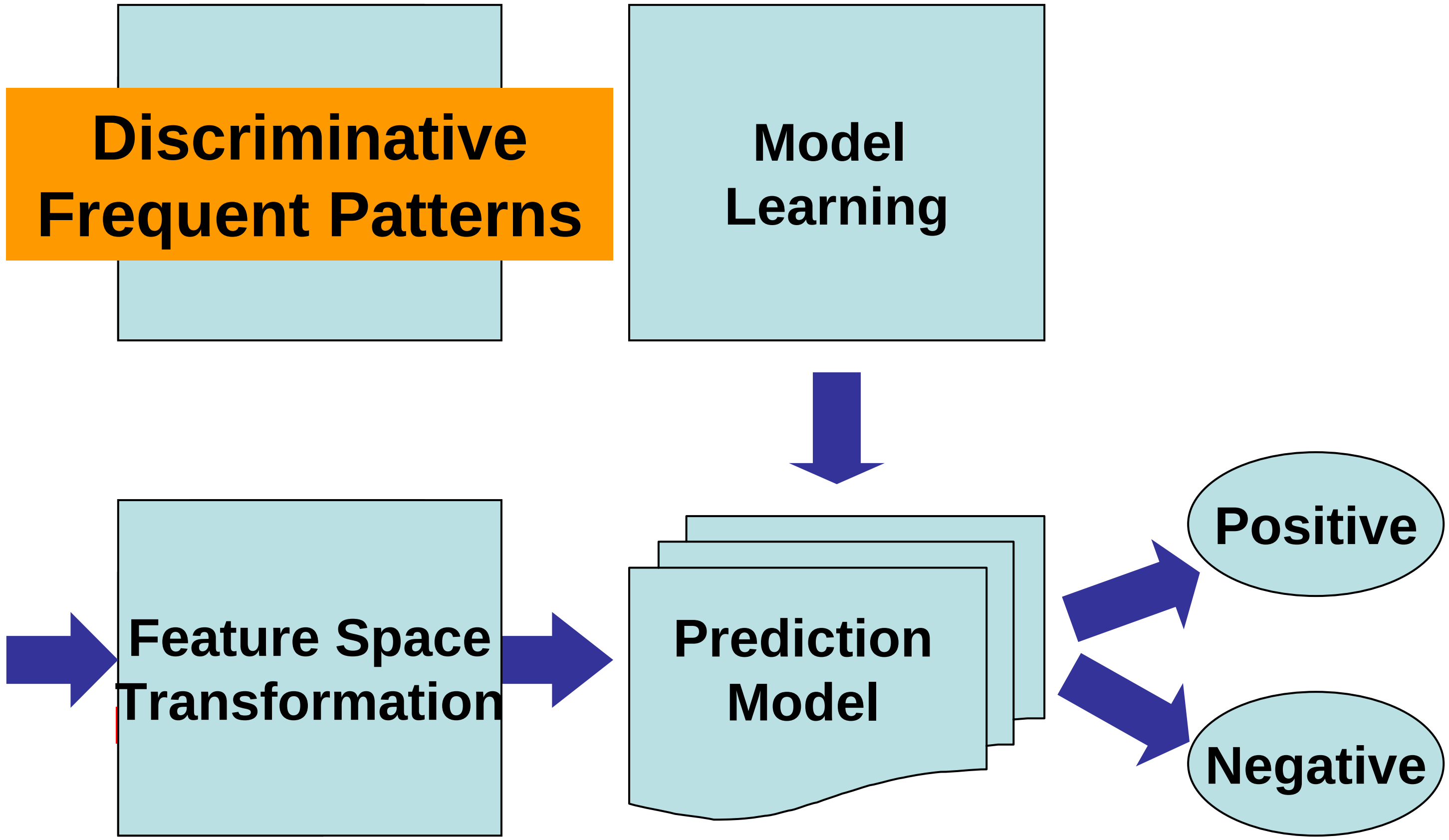
NO Predefined Feature vector

?

Classification model

Prediction Model

Discriminative Frequent Pattern-Based Classification



Pattern-Based Classification on Transactions

Attributes	Class
A, B, C	1
A	1
A, B, C	1
C	0
A, B	1
A, C	0
B, C	0

Mining

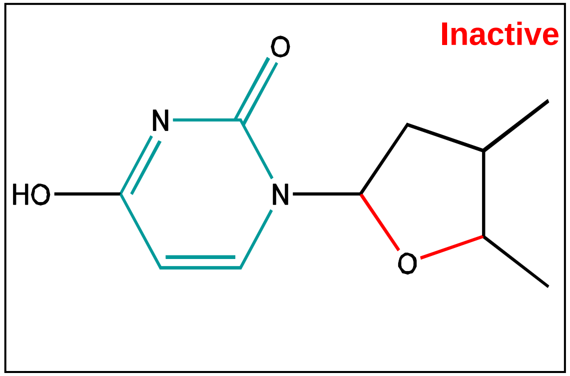
min_sup=3

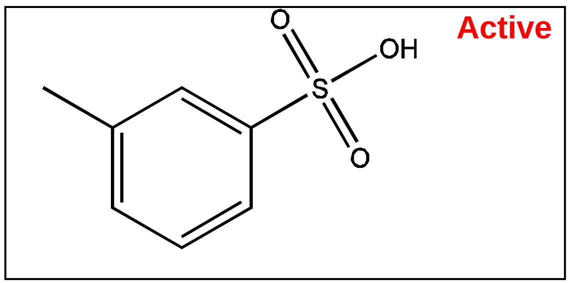
Frequent Itemset	Support
AB	3
AC	3
BC	3

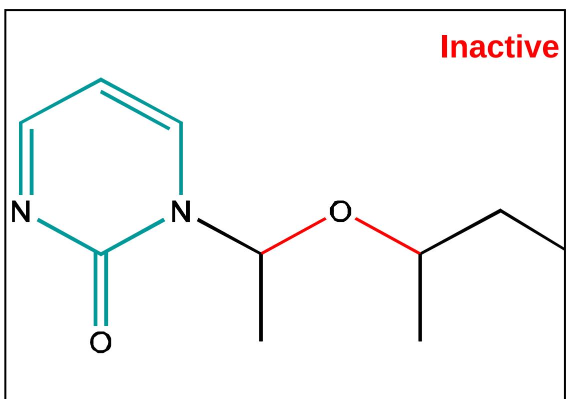
Augmented

A	B	C	AB	AC	BC	Class
1	1	1	1	1	1	1
1	0	0	0	0	0	1
1	1	1	1	1	1	1
0	0	1	0	0	0	0
1	1	0	1	0	0	1
1	0	1	0	1	0	0
0	1	1	0	0	1	0

Pattern-Based Classification on Graphs



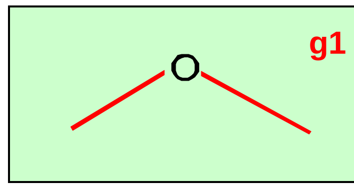


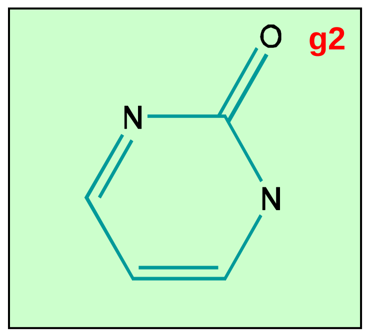


Frequent Graphs

Mining

min_sup=2

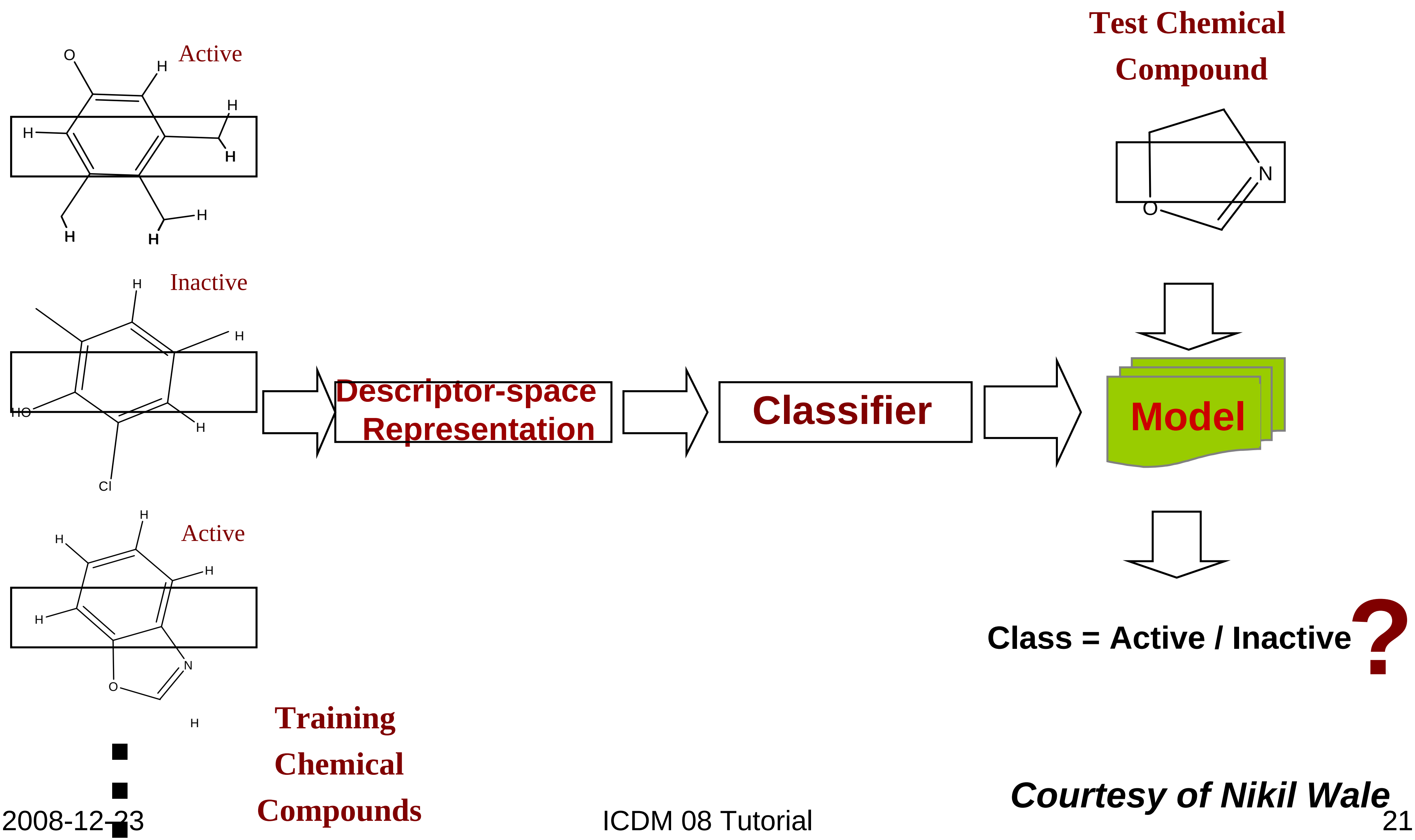




Transform

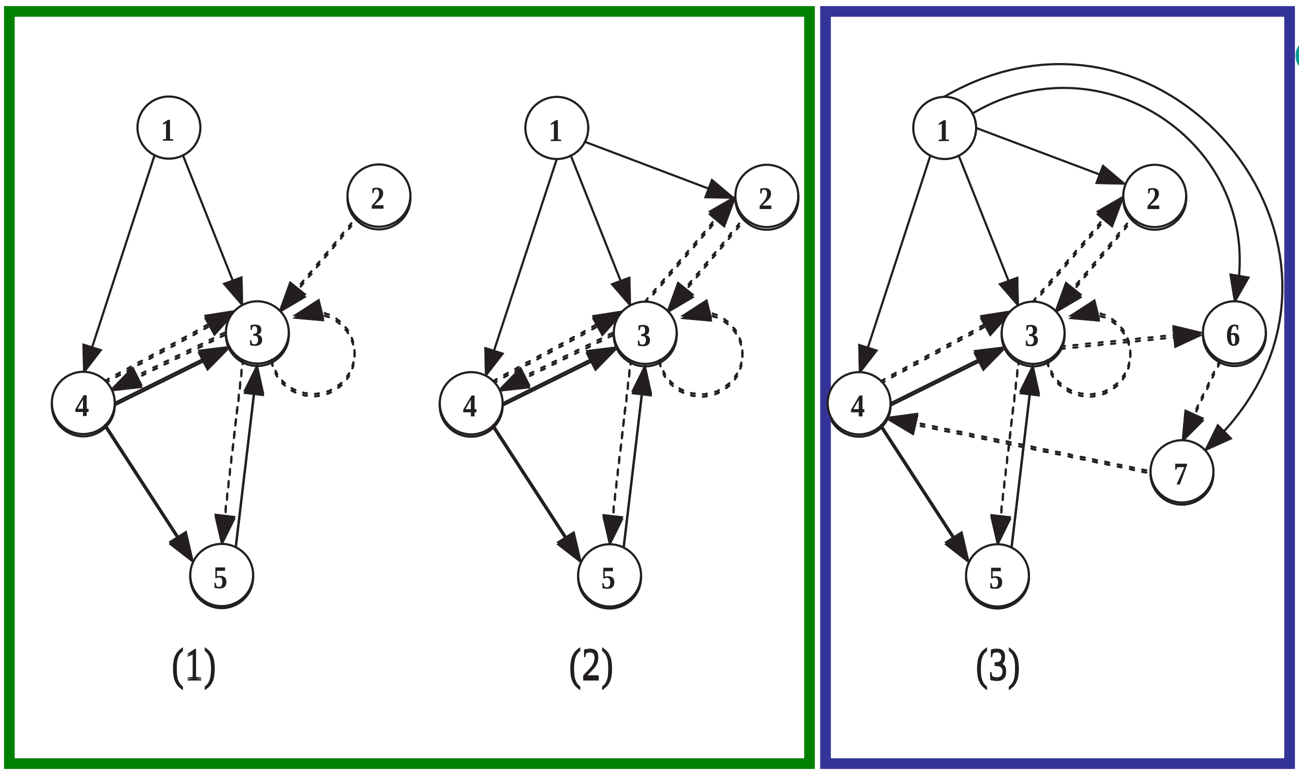
g1	g2	Class
1	1	0
0	0	1
1	1	0

Applications: Drug Design



Courtesy of Nikil Wale

Applications: Bug Localization



calling graph

1: makepat
2: esc
3: addstr
4: getccl
5: dodash
6: in_set_2
7: stclose

correct executions   incorrect executions

Courtesy of Chao Liu

Tutorial Outline

- Frequent Pattern Mining
- Classification Overview
- Associative Classification
- Substructure-Based Graph Classification
- Direct Mining of Discriminative Patterns
- Integration with Other Machine Learning Techniques
- Conclusions and Future Directions

Associative Classification

- **Data:** transactional data, microarray data

- **Pattern:** frequent itemsets and association rules

- **Representative work**
  - CBA [Liu, Hsu and Ma, KDD'98]
  - Emerging patterns [Dong and Li, KDD'99]
  - CMAR [Li, Han and Pei, ICDM'01]
  - CPAR [Yin and Han, SDM'03]
  - RCBT [Cong et al., SIGMOD'05]
  - Lazy classifier [Veloso, Meira and Zaki, ICDM'06]
  - Integrated with classification models [Cheng et al., ICDE'07]

CBA [Liu, Hsu and Ma, KDD'98]

- **Basic idea**
  - Mine high-confidence, high-support class association rules with Apriori
  - Rule LHS: a conjunction of conditions
  - Rule RHS: a class label
  - Example:

R1: age < 25 & credit = 'good' → buy iPhone (sup=30%, conf=80%)

R2: age > 40 & income < 50k → not buy iPhone (sup=40%, conf=90%)

CBA

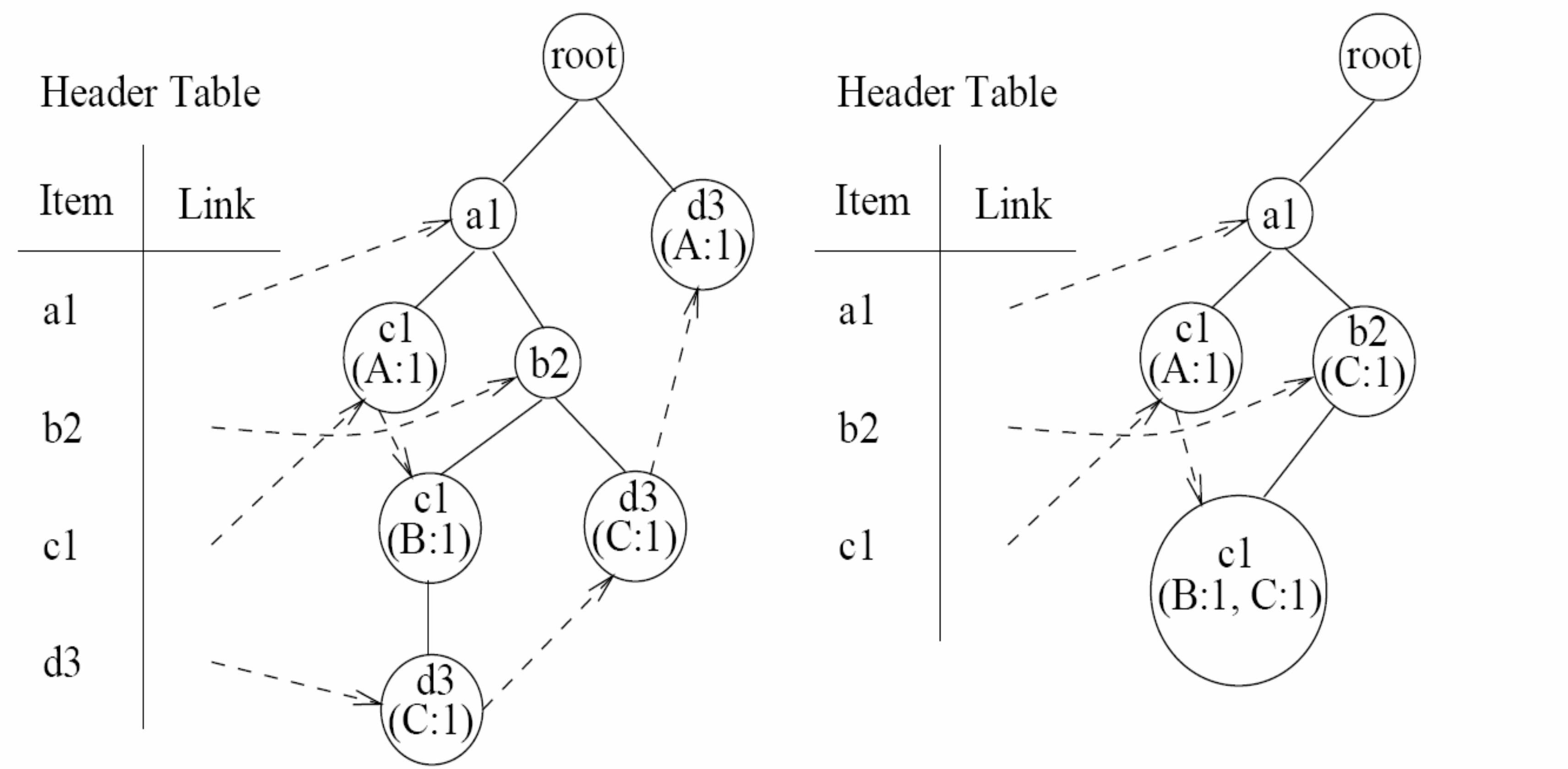
- **Rule mining**
  - Mine the set of association rules *wrt.* min_sup and min_conf
  - Rank rules in descending order of confidence and support
  - Select rules to ensure training instance coverage

- **Prediction**
  - Apply the first rule that matches a test case
  - Otherwise, apply the default rule

CMAR [Li, Han and Pei, ICDM'01]

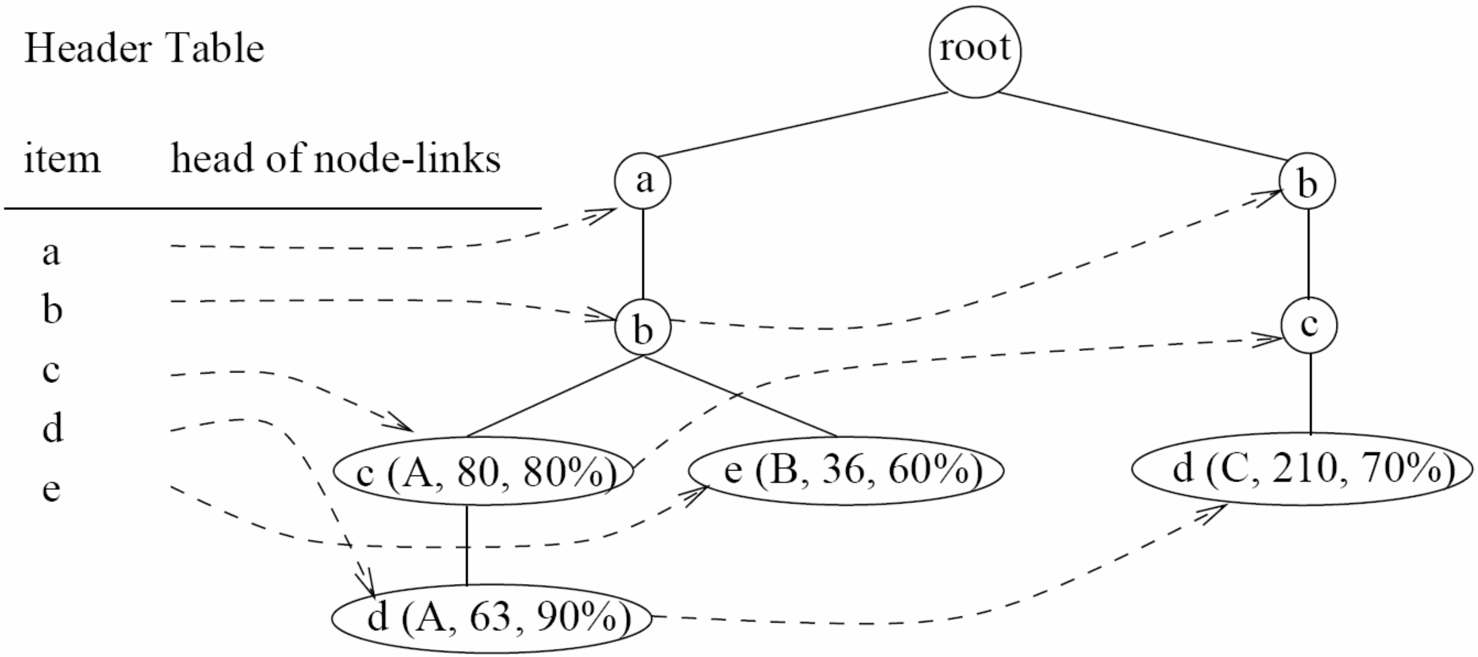
- **Basic idea**
  - Mining: build a class distribution-associated FP-tree
  - Prediction: combine the strength of multiple rules
- **Rule mining**
  - Mine association rules from a class distribution-associated FP-tree
  - Store and retrieve association rules in a CR-tree
  - Prune rules based on confidence, correlation and database coverage

Class Distribution-Associated FP-tree



(a) FP-tree (b) FP-tree after merging nodes of d3

CR-tree: A Prefix-tree to Store and Index Rules



Rule-id	Rule	Support	Confidence
1	$abc \rightarrow A$	80	80%
2	$abcd \rightarrow A$	63	90%
3	$abe \rightarrow B$	36	60%
4	$bcd \rightarrow C$	210	70%

Prediction Based on Multiple Rules

- All rules matching a test case are collected and grouped based on class labels. The group with the most strength is used for prediction

- Multiple rules in one group are combined with a weighted chi-square as:

$$\sum \frac{\chi^2 \chi^2}{\max \chi^2}$$

where $\max \chi^2$ is the upper bound of chi-square of a rule.

CPAR [Yin and Han, SDM'03]

- **Basic idea**
  - Combine associative classification and FOIL-based rule generation
  - **Foil gain**: criterion for selecting a literal

$$gain(p) = |P^*|(log\frac{|P^*|}{|P^*| + |N^*|} - log\frac{|P|}{|P| + |N|})$$

  - Improve accuracy over traditional rule-based classifiers
  - Improve efficiency and reduce number of rules over association rule-based methods

CPAR

- **Rule generation**
  - Build a rule by adding literals one by one in a greedy way according to foil gain measure
  - Keep all close-to-the-best literals and build several rules simultaneously
- **Prediction**
  - Collect all rules matching a test case
  - Select the best k rules for each class
  - Choose the class with the highest expected accuracy for prediction

Performance Comparison [Yin and Han, SDM'03]

Data	C4.5	Ripper	CBA	CMAR	CPAR
anneal	94.8	95.8	97.9	97.3	98.4
austral	84.7	87.3	84.9	86.1	86.2
auto	80.1	72.8	78.3	78.1	82.0
breast	95.0	95.1	96.3	96.4	96.0
cleve	78.2	82.2	82.8	82.2	81.5
crx	84.9	84.9	84.7	84.9	85.7
diabetes	74.2	74.7	74.5	75.8	75.1
german	72.3	69.8	73.4	74.9	73.4
glass	68.7	69.1	73.9	70.1	74.4
heart	80.8	80.7	81.9	82.2	82.6
hepatic	80.6	76.7	81.8	80.5	79.4
horse	82.6	84.8	82.1	82.6	84.2
hypo	99.2	98.9	98.9	98.4	98.1
iono	90.0	91.2	92.3	91.5	92.6
iris	95.3	94.0	94.7	94.0	94.7
labor	79.3	84.0	86.3	89.7	84.7
…	…	…	…	…	…
Average	83.34	82.93	84.69	85.22	85.17

Emerging Patterns
[Dong and Li, KDD'99]

- Emerging Patterns (EPs) are contrast patterns between two classes of data whose support changes significantly between the two classes.

- Change significance can be defined by:

big support ratio:
supp2(X)/supp1(X) >= minRatio

similar to RiskRatio

big support difference:
|supp2(X) – supp1(X)| >= minDiff

defined by Bay+Pazzani 99

- If supp2(X)/supp1(X) = infinity, then X is a *jumping EP*.
  - jumping EP occurs in one class but never occurs in the other class.

Courtesy of Bailey and Dong

A Typical EP in the Mushroom Dataset

- The Mushroom dataset contains two classes: edible and poisonous
- Each data tuple has several features such as: odor, ring-number, stalk-surface-bellow-ring, etc.
- Consider the pattern

  {odor = none,
  stalk-surface-below-ring = smooth,
  ring-number = one}

  Its support increases from 0.2% in the poisonous class to 57.6% in the edible class (a growth rate of 288).

Courtesy of Bailey and Dong

EP-Based Classification: CAEP [Dong et al, DS'99]

- Given a test case T, obtain T's scores for each class, by aggregating the discriminating power of EPs contained in T; assign the class with the maximal score as T's class.
- The discriminating power of EPs are expressed in terms of supports and growth rates. Prefer large supRatio, large support

- The contribution of one EP X (support weighted confidence):

strength(X) = sup(X) * supRatio(X) / (supRatio(X)+1)

- Given a test T and a set E(Ci) of EPs for class Ci, the aggregate score of T for Ci is

score(T, Ci) = Σ strength(X)
(over X of Ci matching T)

- For each class, may use median (or 85%) aggregated value to normalize to avoid bias towards class with more EPs

Courtesy of Bailey and Dong

Top-k Covering Rule Groups for Gene Expression Data [Cong et al., SIGMOD'05]

- **Problem**
  - Mine strong association rules to reveal correlation between gene expression patterns and disease outcomes
  - Example: $gene_1[a_1, b_1], \ldots, gene_n[a_n, b_n] \rightarrow class$
  - Build a rule-based classifier for prediction

- **Challenges**: high dimensionality of data
  - Extremely long mining time
  - Huge number of rules generated

- **Solution**
  - Mining top-k covering rule groups with row enumeration
  - A classifier RCBT based on top-k covering rule groups

A Microarray Dataset

1000 - 100,000 columns

100-500 rows

	Class	Gene1	Gene2	Gene3	Gene4	Gene5	Gene6	G
Sample1	Cancer							
Sample2	Cancer							
. . .								
SampleN-1	~Cancer							
SampleN	~Cancer							

- Find closed patterns which occur frequently among genes.
- Find rules which associate certain combination of the columns that affect the class of the rows
  - Gene1,Gene10,Gene1001 -> Cancer

Courtesy of Anthony Tung

Top-k Covering Rule Groups

- **Rule group**
  - A set of rules which are supported by the same set of transactions $G = \{A_i \rightarrow C \mid A_i \subseteq I\}$
  - Rules in one group have the same sup and conf
  - Reduce the number of rules by clustering them into groups

- **Mining top-k covering rule groups**
  - For a row r_i , the set of rule groups $\{\gamma_{r_i j}\}, j \in [1, k]$ satisfying minsup and there is no more significant rule groups

Row Enumeration

i	r_i	class
1	a, b, c, d, e	C
2	a, b, c, o, p	C
3	c, d, e, f, g	C
4	c, d, e, f, g	¬C
5	e, f, g, h, o	¬C

(a) Example Table

i_j	$\mathcal{R}(i_j)$	
	C	$\neg C$
a	1, 2	
b	1, 2	
c	1, 2, 3	4
d	1, 3	4
e	1, 3	4, 5
f	3	4, 5
g	3	4, 5
h		5
o	2	5
p	2	

(b) $TT|_{\emptyset}$ (or TT)

i_j	$\mathcal{R}(i_j)$	
	C	$\neg C$
a	2	
b	2	
c	2, 3	4
d	3	4
e	3	4, 5

(c) $TT|_{\{1\}}$

i_j	$\mathcal{R}(i_j)$	
	C	$\neg C$
c		4
d		4
e		4, 5

(d) $TT|_{\{1,3\}}$

Figure 1: Running Example

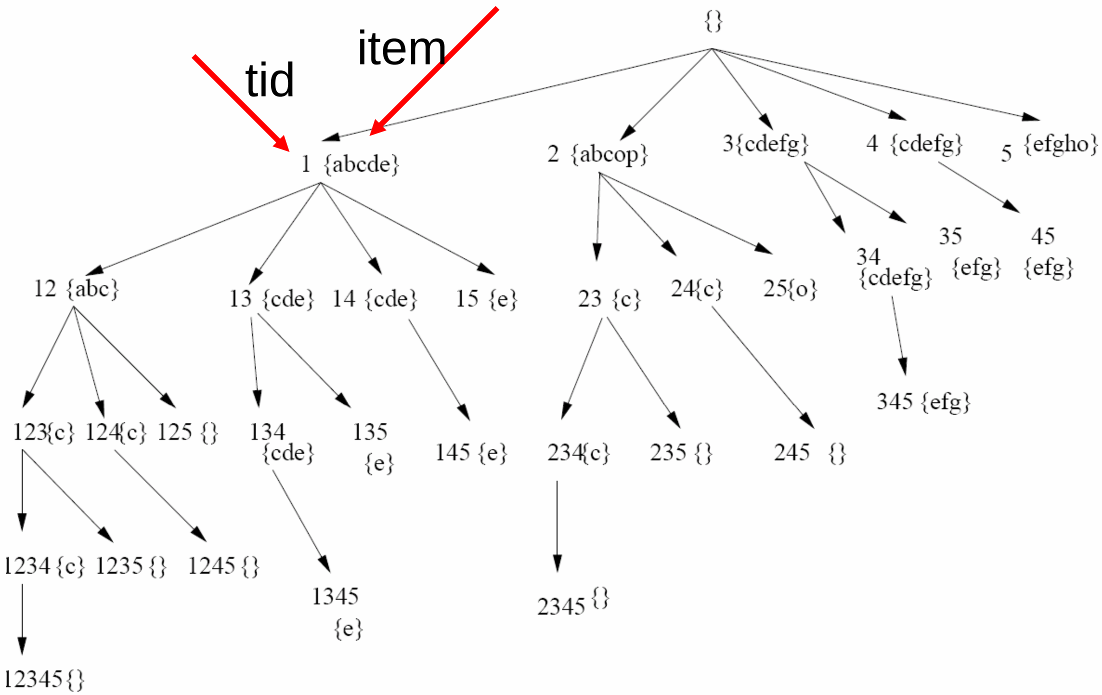


Figure 2: Row Enumeration Tree.

TopkRGS Mining Algorithm

- Perform a depth-first traversal of a row enumeration tree
- $\{\gamma_{r_i j}\}$ for row r_i are initialized
- **Update**
  - If a new rule is more significant than existing rule groups, insert it
- **Pruning**
  - If the confidence upper bound of a subtree X is below the minconf of current top-k rule groups, prune X

RCBT

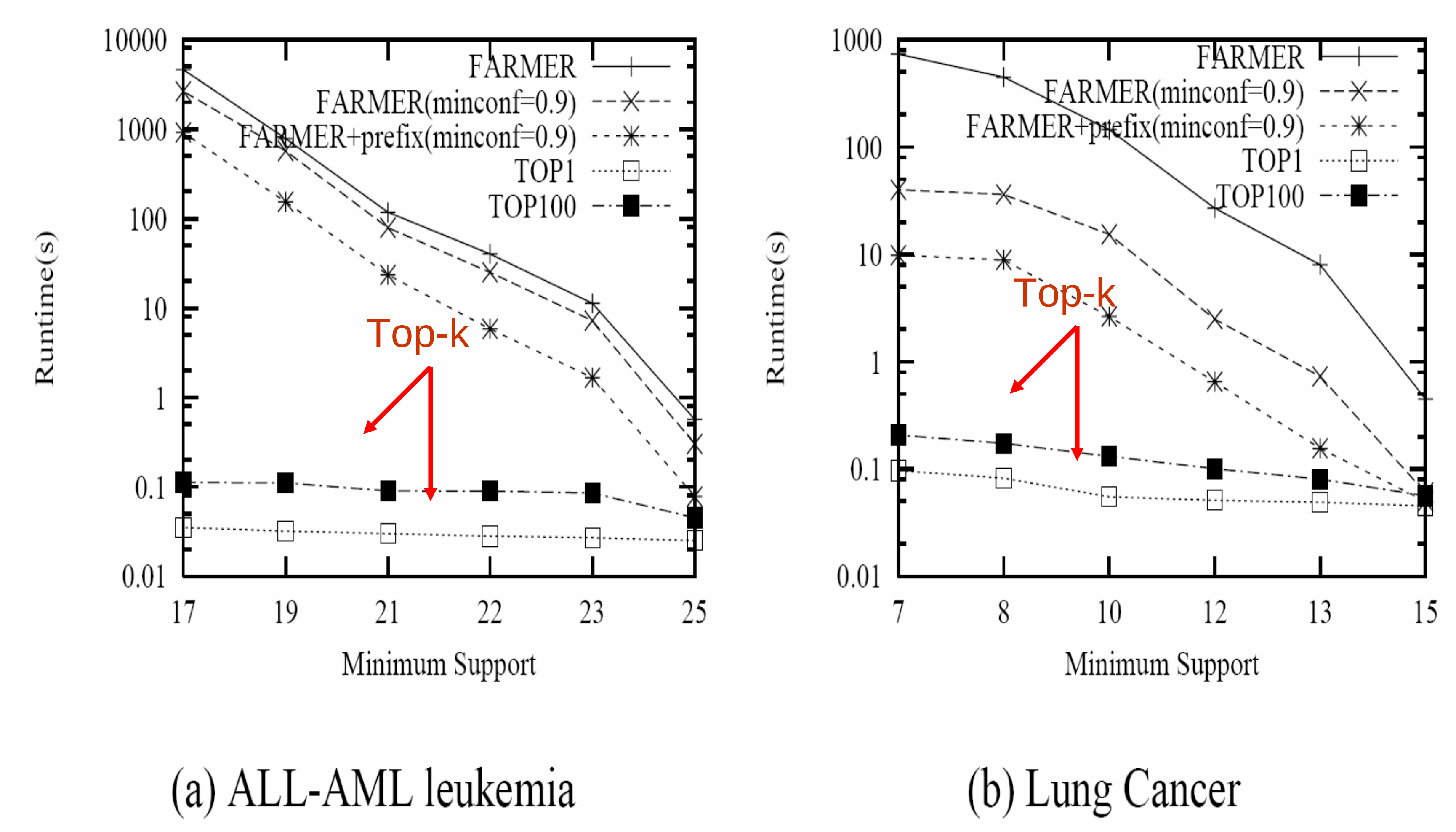
- RCBT uses a set of matching rules for a collective decision
- Given a test data t, assume t satisfies m_i rules of class c_i , the classification score of class c_i is

$$Score(t)^{c_i} = (\sum_{i=1}^{m_i} S(\gamma(t)_i^{c_i})) / S_{norm}^{c_i}$$

where the score of a single rule is

$$S(\gamma^{c_i}) = \gamma^{c_i}.conf * \gamma^{c_i}.sup / d_{c_i}$$

Mining Efficiency



Classification Accuracy

Dataset	RCBT	CBA	IRG Classifier	C4.5 family			SVM
				single tree	bagging	boosting	
AML/ALL (ALL)	91.18%	91.18%	64.71%	91.18%	91.18%	91.18%	97.06%
Lung Cancer(LC)	97.99%	81.88%	89.93%	81.88%	96.64%	81.88%	96.64%
Ovarian Cancer(OC)	97.67%	93.02%	-	97.67%	97.67%	97.67%	97.67%
Prostate Cancer(PC)	97.06%	82.35%	88.24%	26.47%	26.47%	26.47%	79.41%
Average Accuracy	95.98%	87.11%	80.96%	74.3%	77.99%	74.3%	92.70%

Table 2: Classification Results

Lazy Associative Classification [Veloso, Meira, Zaki, ICDM'06]

- **Basic idea**
  - Simply stores training data, and the classification model (CARs) is built after a test instance is given
    - For a test case t, project training data D on t
    - Mine association rules from D_t
    - Select the best rule for prediction
  - Advantages
    - Search space is reduced/focused
      - Cover small disjuncts (support can be lowered)
    - Only applicable rules are generated
      - A much smaller number of CARs are induced
  - Disadvantages
    - Several models are generated, one for each test instance
    - Potentially high computational cost

Courtesy of **Mohammed Zaki**

Caching for Lazy CARs

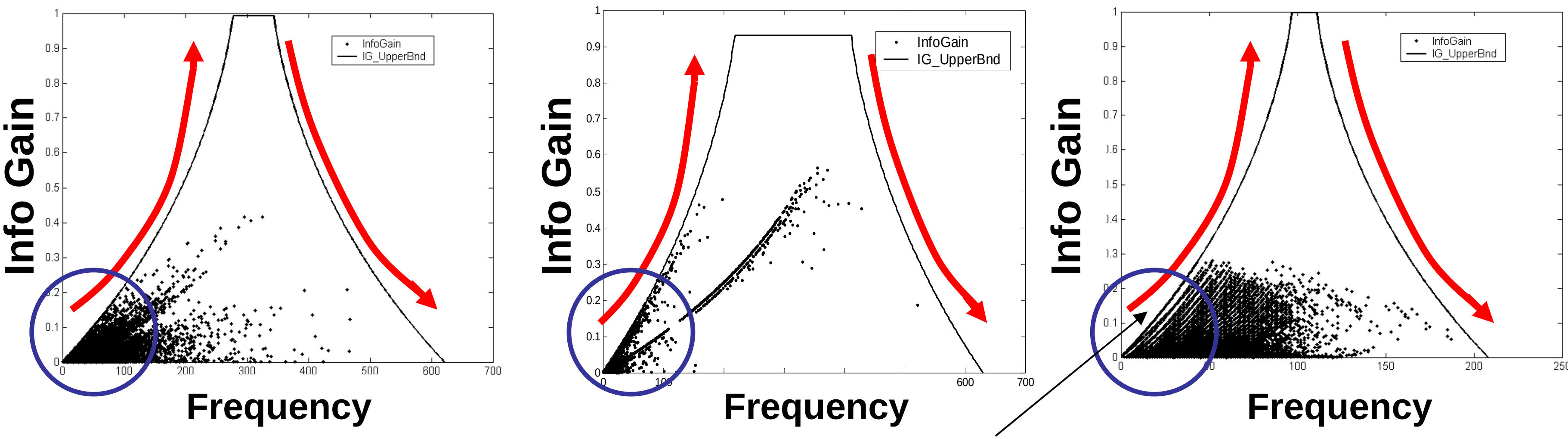
- Models for different test instances may share some CARs
  - Avoid work replication by caching common CARs
- Cache infrastructure
  - All CARs are stored in main memory
  - Each CAR has only one entry in the cache
  - Replacement policy
    - LFU heuristic

***Courtesy of* Mohammed Zaki**

Integrated with Classification Models [Cheng et al., ICDE'07]

- **Framework**
  - Feature construction
    - Frequent itemset mining
  - Feature selection
    - Select discriminative features
    - Remove redundancy and correlation
  - Model learning
    - A general classifier based on SVM or C4.5 or other classification model

Information Gain vs. Frequency?



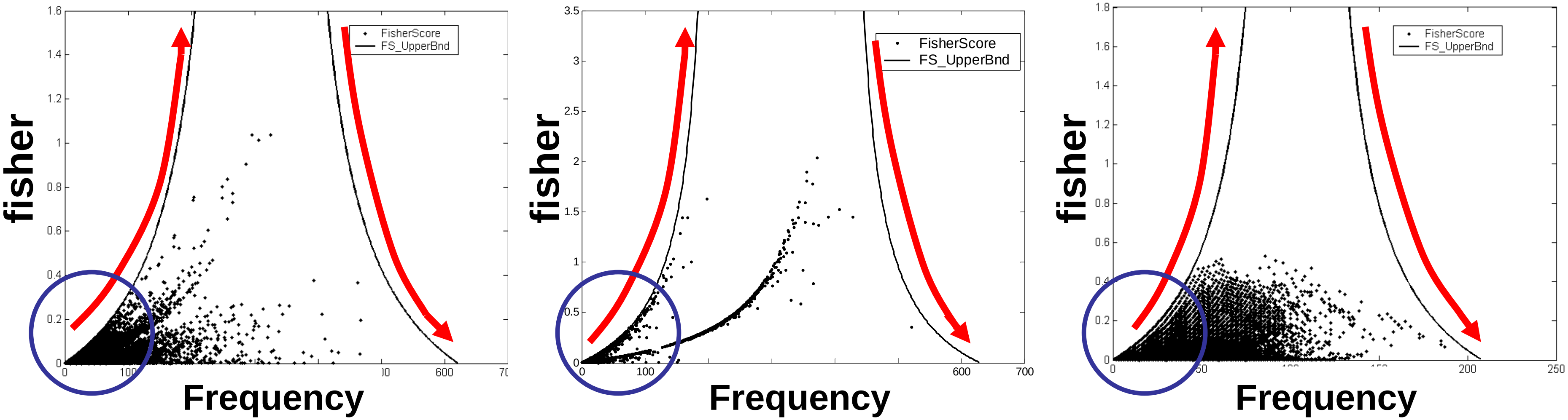
(a) Austral (b) Breast (c) Sonar

Low support, low info gain

Information Gain Formula:

$$IG(C \mid X) = H(C) - H(C \mid X)$$

Fisher Score vs. Frequency?



(a) Austral (b) Breast (c) Sonar

Fisher Score Formula:

$$Fr = \frac{\sum_{i=1}^{c} n_i (u_i - u)^2}{\sum_{i=1}^{c} n_i \sigma_i^2}$$

Analytical Study on Information Gain

$$IG(C|X) = H(C) - H(C|X)$$

$$H(C) = -\sum_{i=1}^{m} p_i \log_2(p_i)$$

$$H(C|X) = \sum_j P(X = x_j) H(Y | X = x_j)$$

Entropy

Constant given data

Conditional Entropy

Study focus

Information Gain Expressed by Pattern Frequency

X: feature; C: class labels

$$H(C\mid X) = -\sum P(x) \sum_{c\in\{0,1\}} P(c\mid x)\log P(c\mid x)$$

Entropy when feature appears (x=1)

$$H(C\mid X) = -\theta q\log q - \theta(1-q)\log(1-q)$$

$$+(\theta q - p)\log\frac{p-\theta q}{1-\theta} + (\theta(1-q)-(1-p))\log\frac{(1-p)-\theta(1-q)}{1-\theta}$$

Entropy when feature not appears (x=0)

Conditional prob. of the positive class when pattern appears $q = P(c=1\mid x=1)$

Pattern frequency $\theta = P(x=1)$

Prob. of Positive Class $p = P(c=1)$

Conditional Entropy in a Pure Case

- When $q=1$ (or $q=0$)

$$H(C|X) = -\theta q \log q - \theta(1-q)\log(1-q) \Rightarrow 0$$

$$+(\theta q - p)\log\frac{p-\theta q}{1-\theta} + (\theta(1-q)-(1-p))\log\frac{(1-p)-\theta(1-q)}{1-\theta}$$

$$\Downarrow$$

$$H(C|X)_{|q=1} = (\theta-1)\left(\frac{p-\theta}{1-\theta}\log\frac{p-\theta}{1-\theta} + \frac{1-p}{1-\theta}\log\frac{1-p}{1-\theta}\right)$$

Frequent Is Informative

$$H(C|X)_{|q=1} = (\theta - 1)\left(\frac{p-\theta}{1-\theta}\log\frac{p-\theta}{1-\theta} + \frac{1-p}{1-\theta}\log\frac{1-p}{1-\theta}\right)$$

the H(C|X) minimum value when $\theta \leq p$ (similar for q=0)

Take a partial derivative

$$\boxed{\frac{\partial H(C|X)_{|q=1}}{\partial \theta} = \log\frac{p-\theta}{1-\theta} \leq \log 1 = 0}$$ since $\theta \leq p \leq 1$

H(C|X) lower bound is monotonically decreasing with frequency

IG(C|X) upper bound is monotonically increasing with frequency

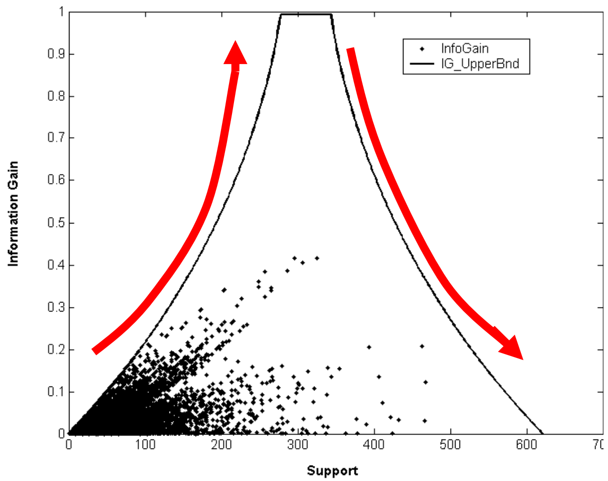
Too Frequent is Less Informative

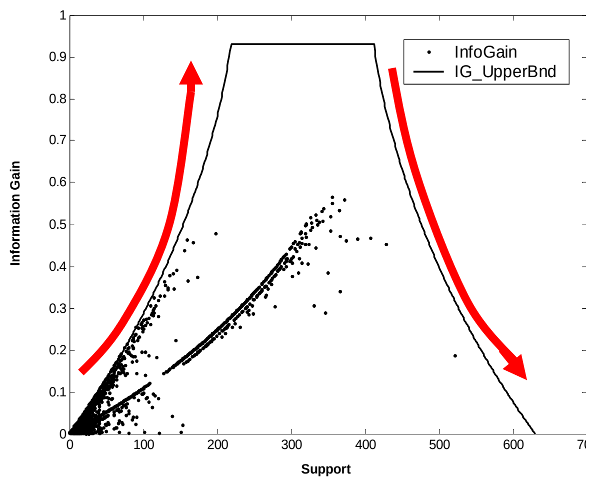
- For $\theta \geq p$, we have a similar conclusion:

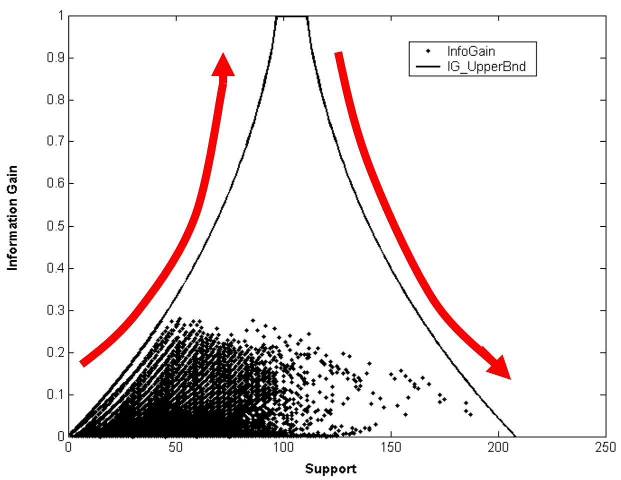
H(C|X) lower bound is monotonically increasing with frequency

IG(C|X) upper bound is monotonically decreasing with frequency

- Similar analysis on Fisher score







Accuracy

	Single Feature		Frequent Pattern	
Data	Item_All*	Item_FS	Pat_All	Pat_FS
austral	85.01	85.50	81.79	91.14
auto	83.25	84.21	74.97	90.79
cleve	84.81	84.81	78.55	95.04
diabetes	74.41	74.41	77.73	78.31
glass	75.19	75.19	79.91	81.32
heart	84.81	84.81	82.22	88.15
iono	93.15	94.30	89.17	95.44

Accuracy based on SVM

	Single Feature		Frequent Pattern	
Data	Item_All	Item_FS	Pat_All	Pat_FS
austral	84.53	84.53	84.21	88.24
auto	71.70	77.63	71.14	78.77
Cleve	80.87	80.87	80.84	91.42
diabetes	77.02	77.02	76.00	76.58
glass	75.24	75.24	76.62	79.89
heart	81.85	81.85	80.00	86.30
iono	92.30	92.30	92.89	94.87

Accuracy based on Decision Tree

* **Item_All**: all single features

Item_FS: single features with selection

Pat_All: all frequent patterns

Pat_FS: frequent patterns with selection

Classification with A Small Feature Set

min_sup	# Patterns	Time	SVM (%)	Decision Tree (%)
1	N/A	N/A	N/A	N/A
2000	68,967	44.70	92.52	97.59
2200	28,358	19.94	91.68	97.84
2500	6,837	2.91	91.68	97.62
2800	1,031	0.47	91.84	97.37
3000	136	0.06	91.90	97.06

Accuracy and Time on Chess

Tutorial Outline

- ❑ Frequent Pattern Mining
- ❑ Classification Overview
- ❑ Associative Classification
- ❑ Substructure-Based Graph Classification
- ❑ Direct Mining of Discriminative Patterns
- ❑ Integration with Other Machine Learning Techniques
- ❑ Conclusions and Future Directions

Substructure-Based Graph Classification

- **Data:** graph data with labels, e.g., chemical compounds, software behavior graphs, social networks

- **Basic idea**
  - Extract graph substructures $F = \{g_{1,\ldots,} g_n\}$
  - Represent a graph with a feature vector $\mathbf{x} = \{x_1, \ldots, x_n\}$, where x_i is the frequency of g_i in that graph
  - Build a classification model

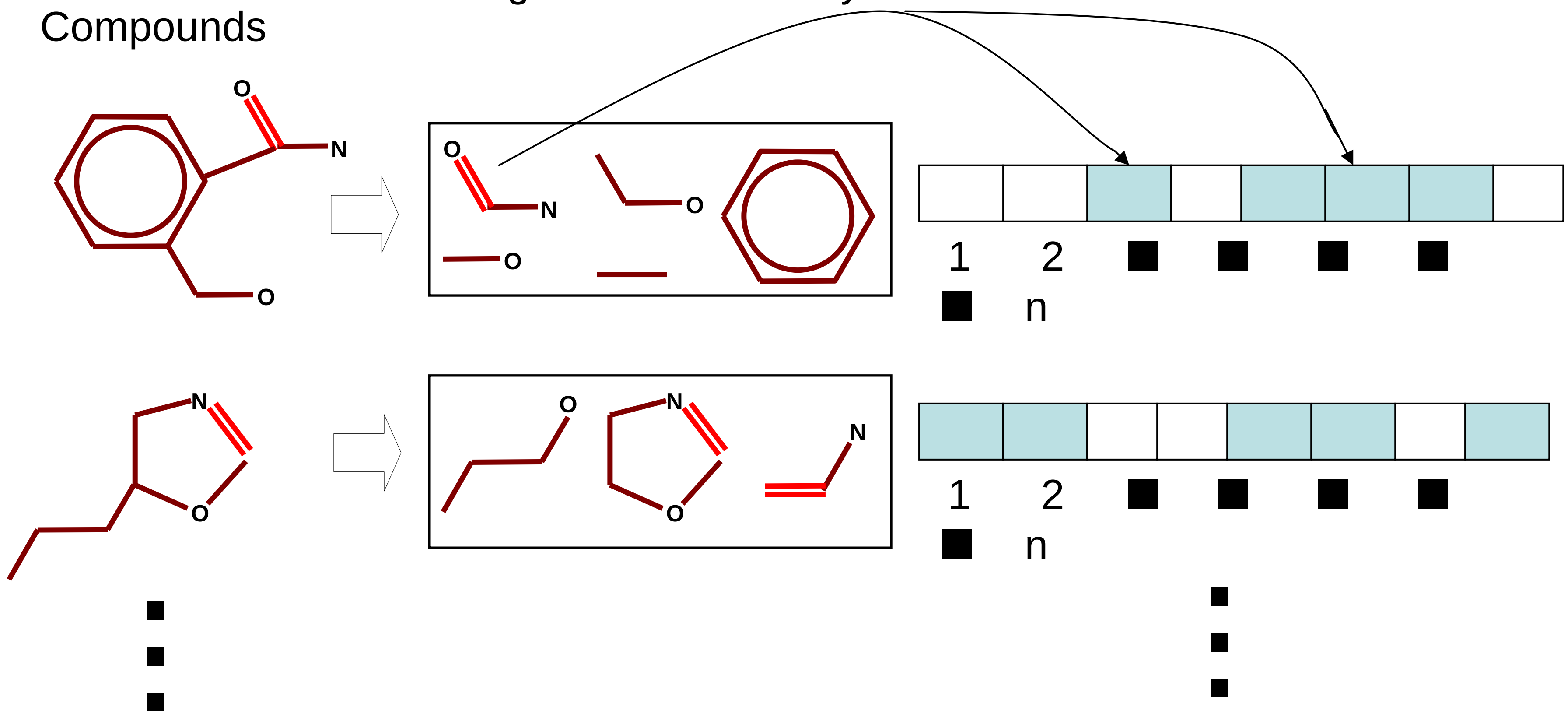
- **Different features and representative work**
  - Fingerprint
  - Maccs keys
  - Tree and cyclic patterns [Horvath et al., KDD'04]
  - Minimal contrast subgraph [Ting and Bailey, SDM'06]
  - Frequent subgraphs [Deshpande et al., TKDE'05; Liu et al., SDM'05]
  - Graph fragments [Wale and Karypis, ICDM'06]

Fingerprints (fp-n)

Chemical Compounds

Enumerate all paths up to length ***l*** and certain cycles

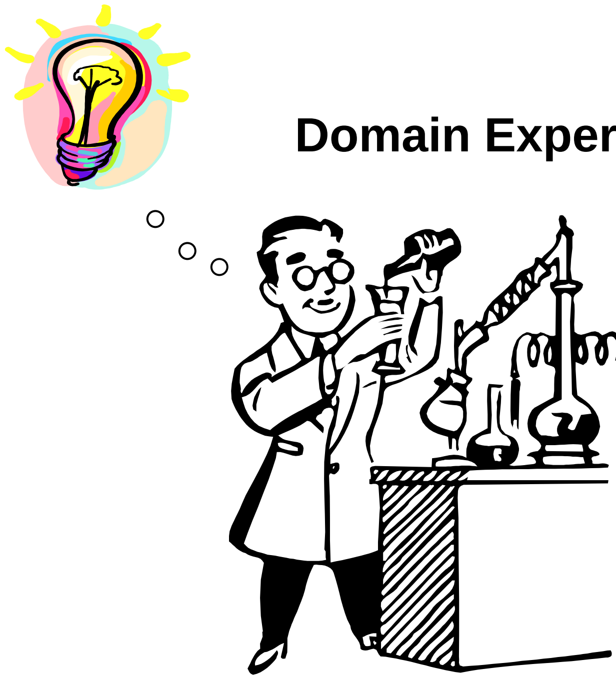
Hash features to position(s) in a fixed length bit-vector



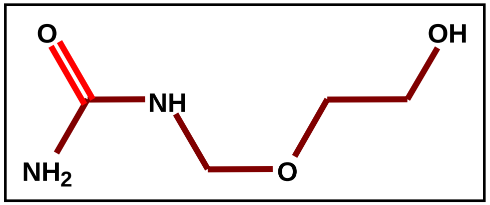
Courtesy of Nikil Wale

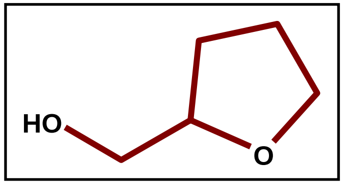
Maccs Keys (MK)

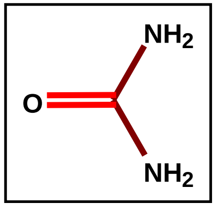
Each Fragment forms a fixed dimension in the descriptor-space



Domain Expert



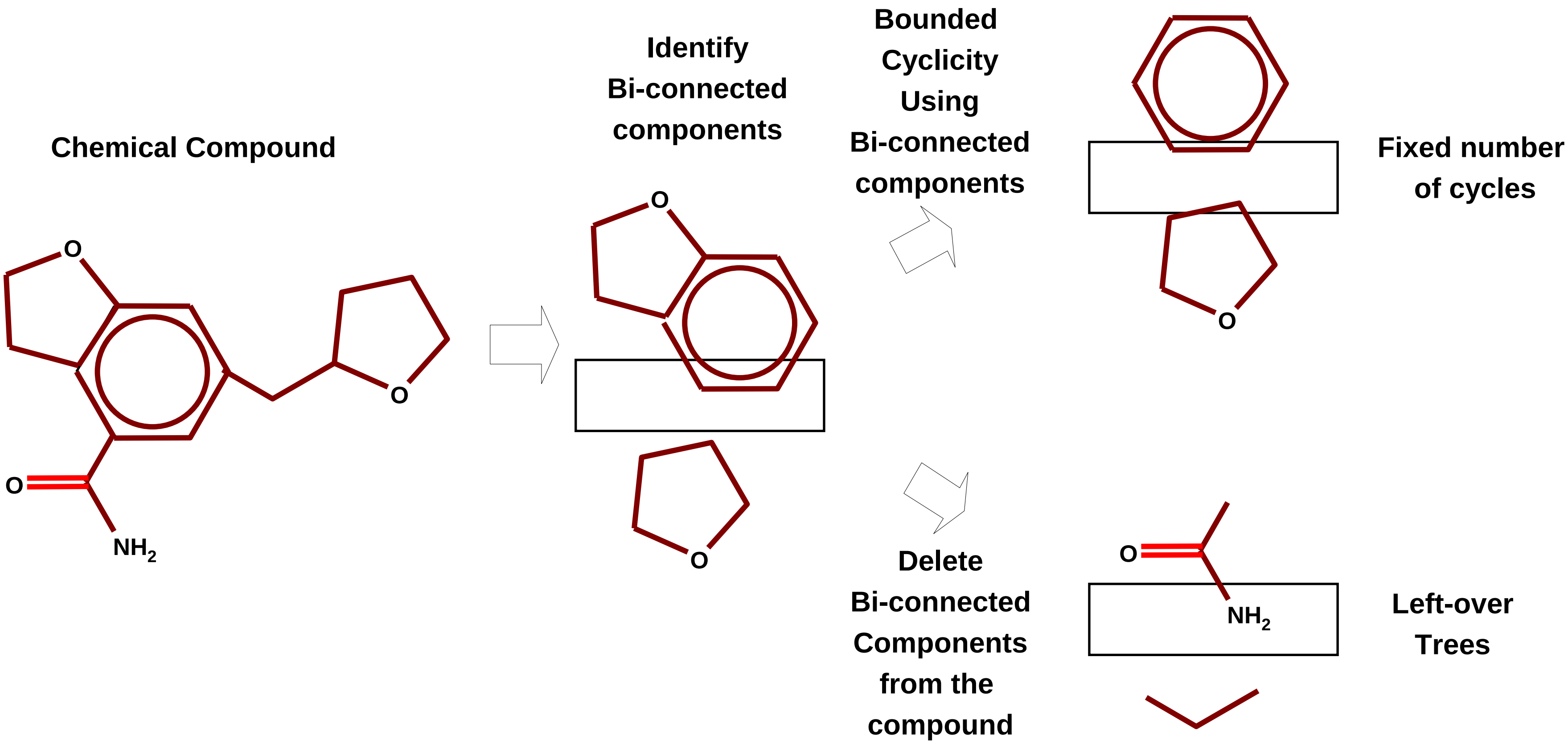




Identify "Important" Fragments for bioactivity

Courtesy of Nikil Wale

Cycles and Trees (CT) [Horvath et al., KDD'04]



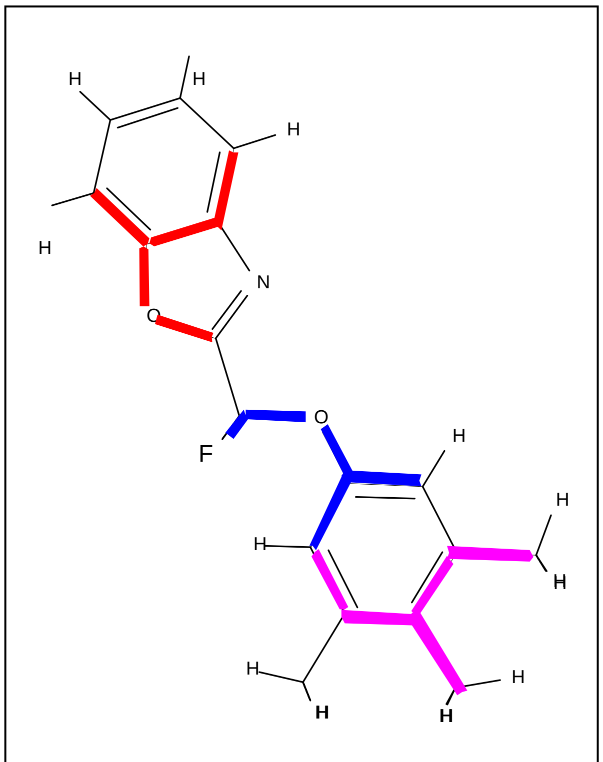
Courtesy of Nikil Wale

Frequent Subgraphs (FS) [Deshpande et al., TKDE'05]

Discovering Features

Topological features – captured by graph representation

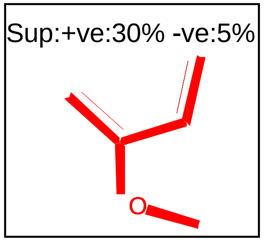
Chemical Compounds

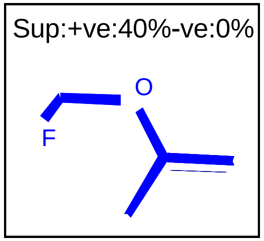


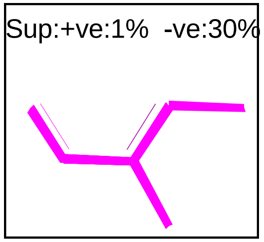
Frequent Subgraph Discovery

Min. Support.

Discovered Subgraphs



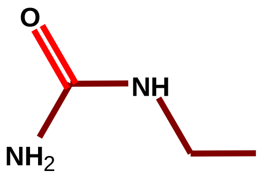




Courtesy of Nikil Wale

Graph Fragments (GF) [Wale and Karypis, ICDM'06]

- Tree Fragments (TF): At least one node of the tree fragment has a degree greater than 2 (no cycles).



- Path Fragments (PF): All nodes have degree less than or equal to 2 but does not include cycles.

- Acyclic Fragments (AF): TF U PF
  - Acyclic fragments are also termed as free trees.

Courtesy of Nikil Wale

Comparison of Different Features [Wale and Karypis, ICDM'06]

Table 1. Design choices made by the descriptor spaces.

Previously developed descriptors

	Topological Complexity	Generation	Precise	Complete Coverage
MK	Low to High	static	Yes	Maybe
fp	Low	dynamic	No	Yes
CT	Medium	dynamic	Yes	Yes
FS	Low to High	dynamic	Yes	Maybe

GF-based descriptors

	Topological Complexity	Generation	Precise	Complete Coverage
PF	Low	dynamic	Yes	Yes
TF	Medium	dynamic	Yes	Maybe
AF	Medium	dynamic	Yes	Yes
GF	Low to High	dynamic	Yes	Yes

Minimal Contrast Subgraphs [Ting and Bailey, SDM'06]

- A contrast graph is a subgraph appearing in one class of graphs and never in another class of graphs
  - Minimal if none of its subgraphs are contrasts
  - **May be disconnected**
    - Allows succinct description of differences
    - But requires larger search space

Courtesy of Bailey and Dong

Mining Contrast Subgraphs

- Main idea
  - Find the maximal common edge sets
    - These may be disconnected
  - Apply a minimal hypergraph transversal operation to derive the minimal contrast edge sets from the maximal common edge sets
  - Must compute minimal contrast vertex sets separately and then minimal union with the minimal contrast edge sets

Courtesy of Bailey and Dong

Frequent Subgraph-Based Classification [Deshpande et al., TKDE'05]

- **Frequent subgraphs**
  - A graph is **frequent** if its support (occurrence frequency) in a given dataset is no less than a minimum support threshold
- **Feature generation**
  - Frequent topological subgraphs by FSG
  - Frequent geometric subgraphs with 3D shape information
- **Feature selection**
  - Sequential covering paradigm
- **Classification**
  - Use SVM to learn a classifier based on feature vectors
  - Assign different misclassification costs for different classes to address skewed class distribution

Varying Minimum Support

TABLE 2
Varying Minimum Support Threshold (σ)

D	σ=10.0%				σ = 15.0%				σ = 20.0%			
	Topo.		Geom.		Topo.		Geom.		Topo.		Geom.	
	A	N_f	A	N_f	A	N_f	A	N_f	A	N_f	A	N_f
P1	66.0	1211	65.5	1317	66.0	513	64.1	478	64.4	254	60.2	268
P2	65.0	967	64.0	1165	65.1	380	63.3	395	64.2	217	63.1	235
P3	60.5	597	60.7	808	59.4	248	61.3	302	59.9	168	60.9	204
P4	54.3	275	55.4	394	56.2	173	57.4	240	57.3	84	58.3	104
H1	81.0	27034	82.1	29554	77.4	13531	79.2	8247	78.4	7479	79.5	7700
H2	70.1	1797	76.0	3739	63.6	307	62.2	953	59.0	139	58.1	493
H3	83.9	27019	89.5	30525	83.6	13557	88.8	11240	84.6	7482	87.7	7494
A1	78.2	476	79.0	492	78.2	484	77.6	332	77.1	312	76.1	193

"A" denotes the area under the ROC curve and "N_f" denotes the number of discovered frequent subgraphs.

Varying Misclassification Cost

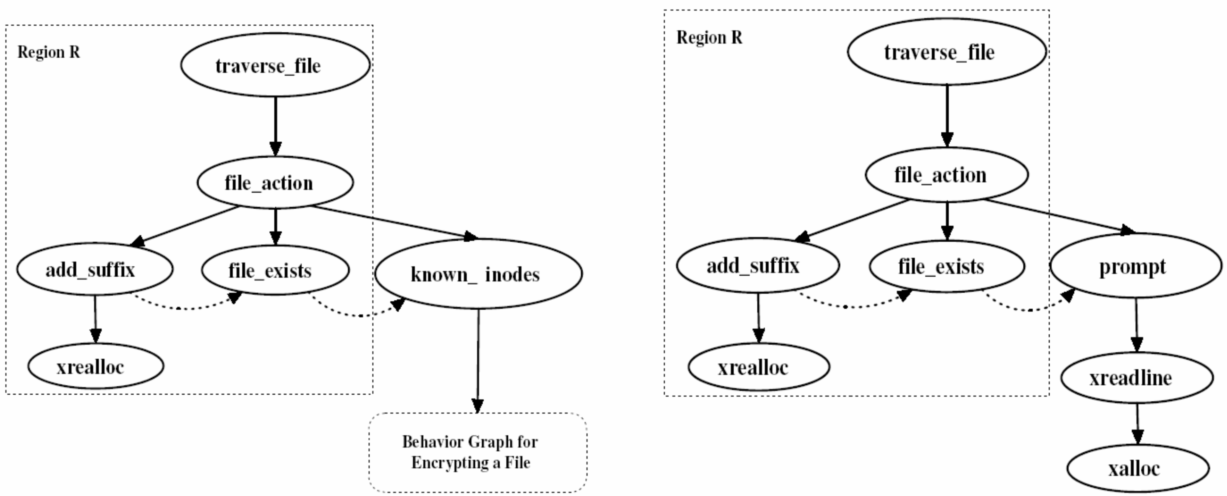
TABLE 4
The Area under the ROC Curve Obtained
by Varying the Misclassification Cost

Dataset	Topo		Geom	
	$\beta = 1.0$	$\beta = EqCost$	$\beta = 1.0$	$\beta = EqCost$
P1	65.5	65.3	65.0	66.7
P2	67.3	66.8	69.9	69.2
P3	62.6	62.6	64.8	64.6
P4	63.4	65.2	63.7	66.1
H1	81.0	79.2	82.1	81.1
H2	76.5	79.4	79.1	81.9
H3	83.9	90.8	89.5	94.0
A1	81.7	82.1	82.6	83.0

"$\beta = 1.0$" indicates the experiments in which each positive and negative example had a weight of one, and "$\beta = EqCost$" indicates the experiments in which the misclassification cost of the positive examples was increased to match the number of negative examples.

Frequent Subgraph-Based Classification for Bug Localization [Liu et al., SDM'05]

- **Basic idea**
  - Mine closed subgraphs from software behavior graphs
  - Build a graph classification model for software behavior prediction
  - Discover program regions that may contain bugs
- **Software behavior graphs**
  - Node: functions
  - Edge: function calls or transitions

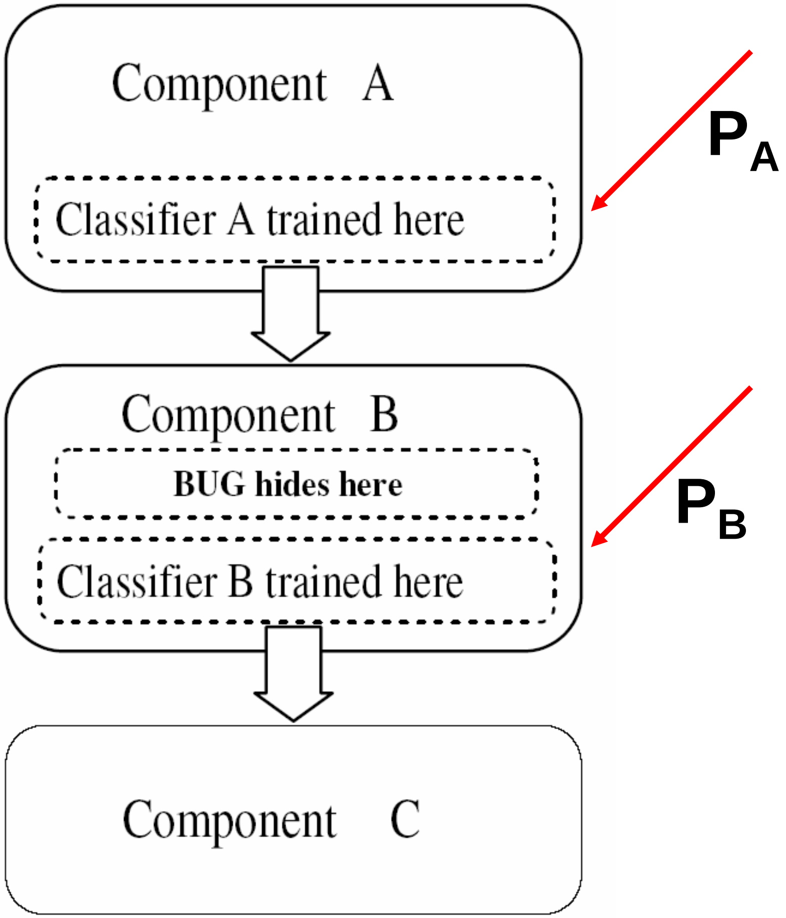


(a) one correct run (b) one incorrect run

Figure 1: Software Behavior Graphs

Bug Localization

- **Identify suspicious functions relevant to incorrect runs**
  - Gradually include more trace data
  - Build multiple classification models and estimate the accuracy boost
  - A function with a significant precision boost could be bug relevant



P_B-P_A is the accuracy boost of function B

Case Study

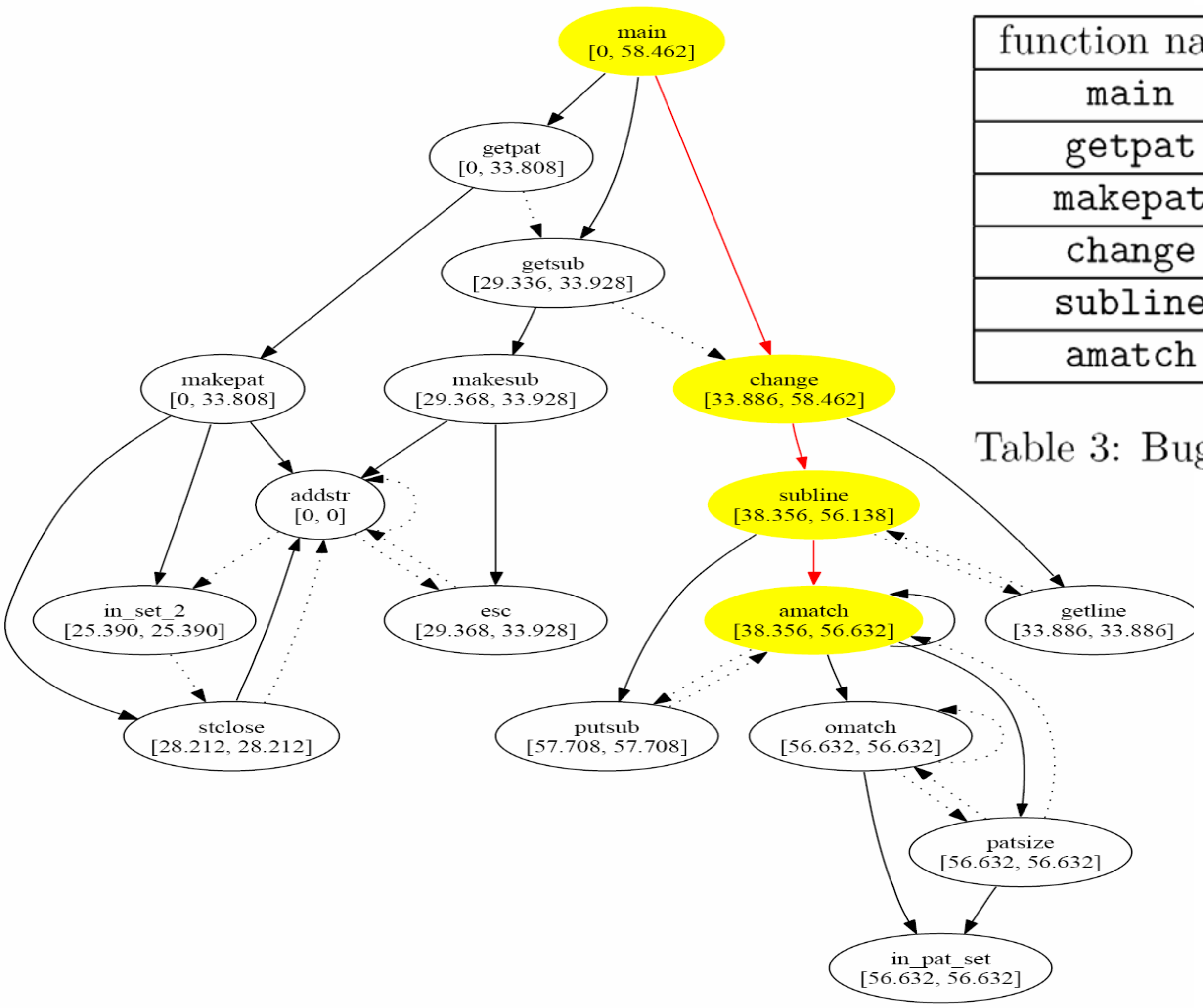


Figure 9: Entrance Precision and Exit Precision

function name	$Precision_{in}$	$Precision_{out}$
main	0	58.462
getpat	0	33.808
makepat	0	33.808
change	33.886	58.462
subline	38.356	56.318
amatch	38.356	56.632

Table 3: Bug-Relevant Functions with $\theta = 20\%$

Graph Fragment [Wale and Karypis, ICDM'06]

- **All graph substructures up to a given length (size or # of bonds)**
  - Determined dynamically → Dataset dependent descriptor space
  - Complete coverage → Descriptors for every compound
  - Precise representation → One to one mapping
  - Complex fragments → Arbitrary topology

- **Recurrence relation to generate graph fragments of length l**

$$F(G,l) = \begin{cases} \emptyset, & \text{if } G \text{ has fewer than } l \text{ edges or } l = 0 \\ eF(G \backslash e, l-1) \cup F(G \backslash e, l), & \text{otherwise,} \end{cases}$$

Courtesy of Nikil Wale

Performance Comparison

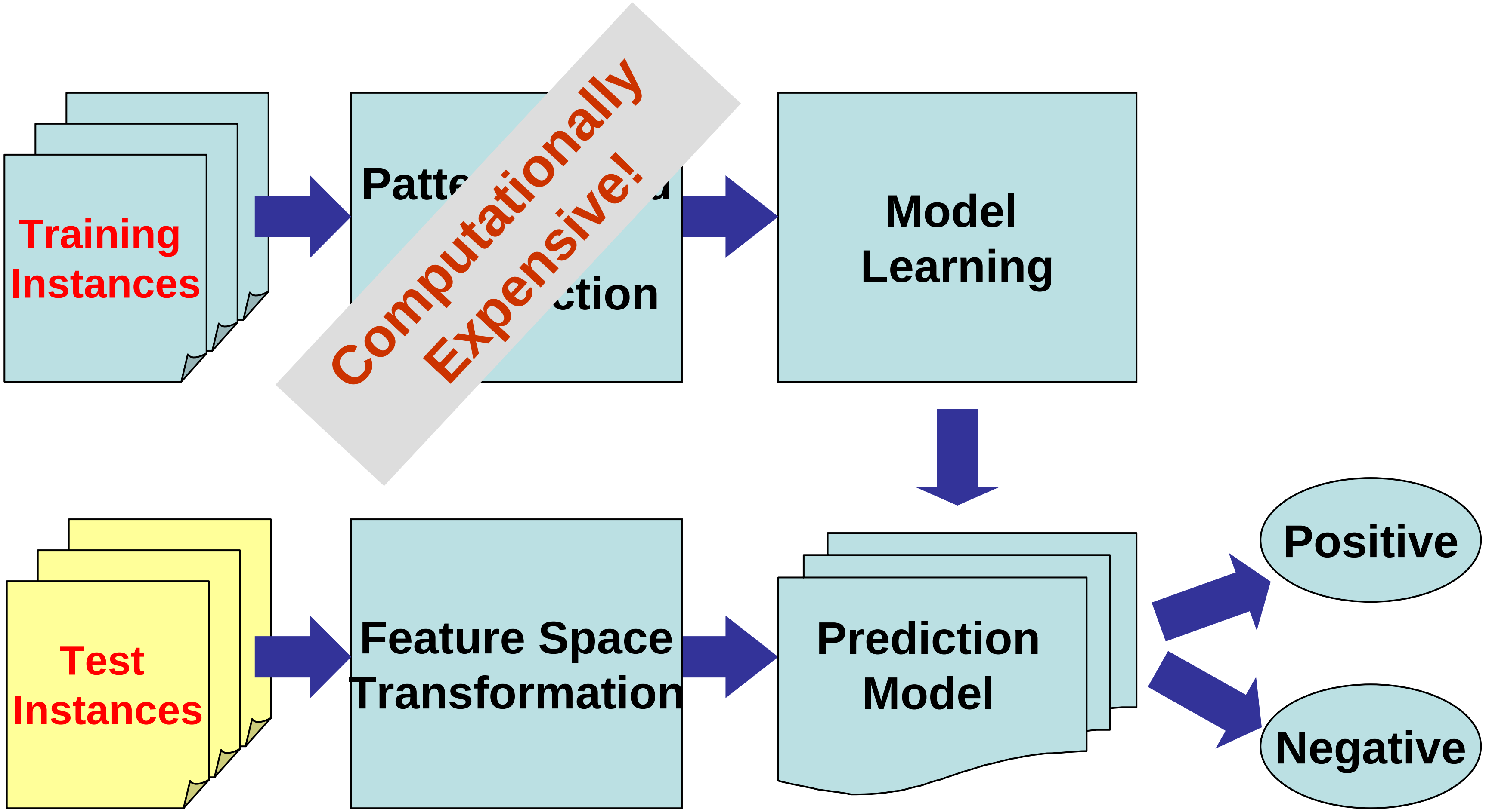
Table 9. ROC50 values for the eight descriptors using kernels derived from RBF.

Datasets	GF ($\mathcal{K}_b^*$)	AF ($\mathcal{K}_b^*$)	TF ($\mathcal{K}_b^*$)	PF ($\mathcal{K}_b^*$)	fp-8192 ($\mathcal{K}_b$)	CT ($\mathcal{K}_b$)	MK ($\mathcal{K}_f$)	FS ($\mathcal{K}_b$)
NCI1	0.303	**0.305**	0.302	0.303	0.198	0.256	0.192	0.249
NCI109	0.285	0.292	**0.293**	0.288	0.199	0.228	0.202	0.232
NCI123	0.244	0.247	0.240	**0.249**	0.177	0.223	0.173	0.234
NCI145	0.318	**0.322**	0.321	0.317	0.203	0.255	0.194	0.258
NCI167	0.060	**0.062**	**0.062**	0.053	0.043	0.041	0.043	0.047
NCI220	0.248	0.263	0.266	0.261	0.272	0.218	**0.393**	0.198
NCI33	**0.305**	0.304	0.297	0.286	0.186	0.238	0.210	0.242
NCI330	0.313	**0.317**	0.306	0.311	0.235	0.305	0.241	0.241
NCI41	0.341	**0.346**	0.344	0.344	0.237	0.267	0.213	0.294
NCI47	**0.298**	0.295	0.271	0.289	0.194	0.232	0.186	0.227
NCI81	**0.287**	0.284	0.279	**0.286**	0.188	0.230	0.194	0.231
NCI83	0.298	**0.301**	0.298	0.300	0.197	0.258	0.204	0.253
H1	0.263	0.264	0.259	**0.265**	0.229	0.223	0.233	0.220
H2	0.633	**0.636**	0.629	0.635	0.573	0.556	0.545	0.575
A1	0.202	0.195	0.167	**0.212**	0.125	0.128	0.062	0.123
H3	**0.637**	0.631	0.628	0.632	0.578	0.589	0.584	0.554
D1	**0.374**	0.357	0.362	0.358	0.345	0.317	0.340	0.307
D2	0.585	**0.592**	0.571	0.545	0.567	0.558	0.551	0.486
D3	**0.512**	0.506	0.497	0.507	0.430	0.454	0.424	0.482
D4	0.465	**0.470**	0.458	0.460	0.401	0.426	0.380	0.400
P1	0.603	0.599	**0.604**	**0.604**	0.544	0.542	0.563	0.553
P2	0.502	0.500	0.468	0.492	**0.532**	0.493	0.512	0.465
P3	0.572	0.582	0.458	0.580	0.553	0.499	**0.583**	0.558
P4	0.623	**0.625**	0.605	0.622	0.542	0.559	0.536	0.594
C1	0.814	**0.815**	0.810	0.808	0.794	0.744	**0.815**	0.813
M1	**0.440**	0.439	0.414	0.429	0.428	0.343	0.411	0.410
M2	**0.613**	0.606	0.573	0.600	0.567	0.484	0.577	0.584
M3	0.786	0.773	0.779	0.768	0.785	0.749	**0.788**	0.775
ARQB	0.976	0.978	0.948	0.965	0.803	0.832	0.800	0.843

Tutorial Outline

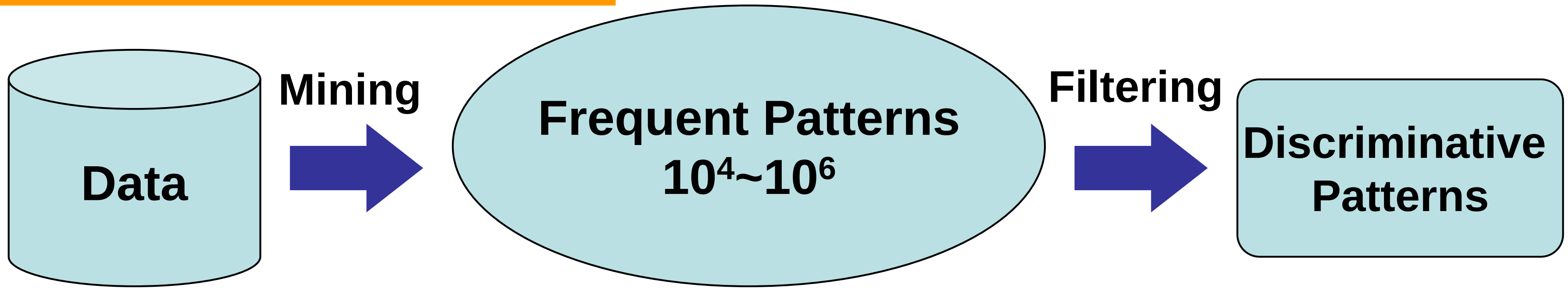
- ❑ **Frequent Pattern Mining**
- ❑ **Classification Overview**
- ❑ **Associative Classification**
- ❑ **Substructure-Based Graph Classification**
- ❑ **Direct Mining of Discriminative Patterns**
- ❑ **Integration with Other Machine Learning Techniques**
- ❑ **Conclusions and Future Directions**

Re-examination of Pattern-Based Classification

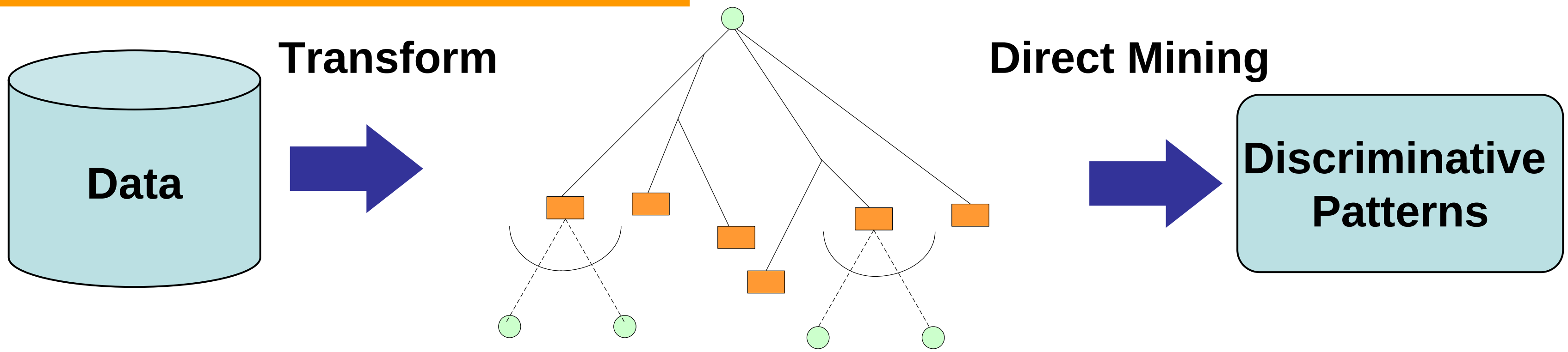


The Computational Bottleneck

Two steps, expensive

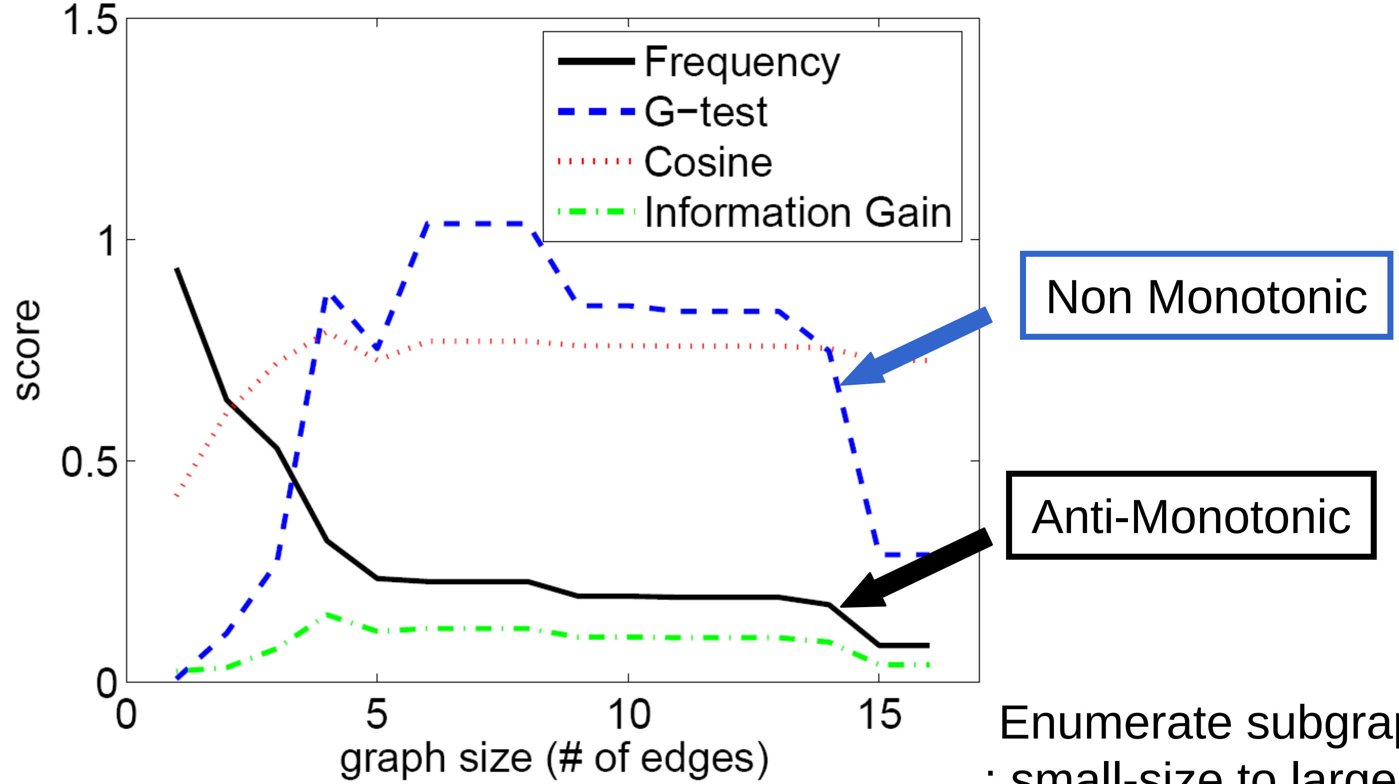


Direct mining, efficient



FP-tree

Challenge: Non Anti-Monotonic



Enumerate subgraphs
: small-size to large-size

Non-Monotonic: Enumerate all subgraphs then check their score?

Direct Mining of Discriminative Patterns

- **Avoid mining the whole set of patterns**
  - Harmony [Wang and Karypis, SDM'05]
  - DDPMine [Cheng et al., ICDE'08]
  - LEAP [Yan et al., SIGMOD'08]
  - MbT [Fan et al., KDD'08]
- **Find the most discriminative pattern**
  - A search problem?
  - An optimization problem?
- **Extensions**
  - Mining top-k discriminative patterns
  - Mining approximate/weighted discriminative patterns

Harmony [Wang and Karypis, SDM'05]

- **Direct mining the best rules for classification**
  - Instance-centric rule generation: the highest confidence rule for each training case is included
  - Efficient search strategies and pruning methods
    - Support equivalence item (keep "generator itemset")
      - e.g., prune (ab) if sup(ab)=sup(a)
    - Unpromising item or conditional database
      - Estimate confidence upper bound
      - Prune an item or a conditional db if it cannot generate a rule with higher confidence
  - Ordering of items in conditional database
    - Maximum confidence descending order
    - Entropy ascending order
    - Correlation coefficient ascending order

Harmony

- **Prediction**
  - For a test case, partition the rules into k groups based on class labels
  - Compute the score for each rule group
  - Predict based the rule group with the highest score

Accuracy of Harmony

Database	FOIL	CPAR	SVM	HARMONY
adult	82.5	76.7	**84.16**	81.9
chess	42.6	32.8	29.83	**44.87**
connect	65.7	54.3	**72.5**	68.05
led7	62.3	71.2	73.78	**74.56**
letRecog	57.5	59.9	67.76	**76.81**
mushroom	99.5	98.8	99.67	**99.94**
nursery	91.3	78.5	91.35	**92.83**
pageBlocks	**91.6**	76.2	91.21	**91.6**
penDigits	88.0	83.0	93.2	**96.23**
waveform	75.6	75.4	**83.16**	80.46
average	75.66	70.68	78.663	**80.725**

Runtime of Harmony

Database	FOIL	CPAR	SVM	HARMONY
adult	10251.0	**809.0**	2493.1	1395.5
chess	10122.8	1736.0	13289.4	**11.34**
connect	35572.5	24047.1	74541.1	**85.44**
led7	11.5	5.7	17.12	**1.29**
letRecog	4365.6	**764.0**	17825.2	778.91
mushroom	38.3	15.4	16.6	**8.78**
nursery	73.1	51.7	322.4	**6.21**
pageBlocks	43.1	15.5	11.2	**2.5**
penDigits	821.1	101.9	512.7	**82.6**
waveform	295.3	38.1	**36.2**	130.0
total	61594.3	27584.4	109065.02	**2502.57**

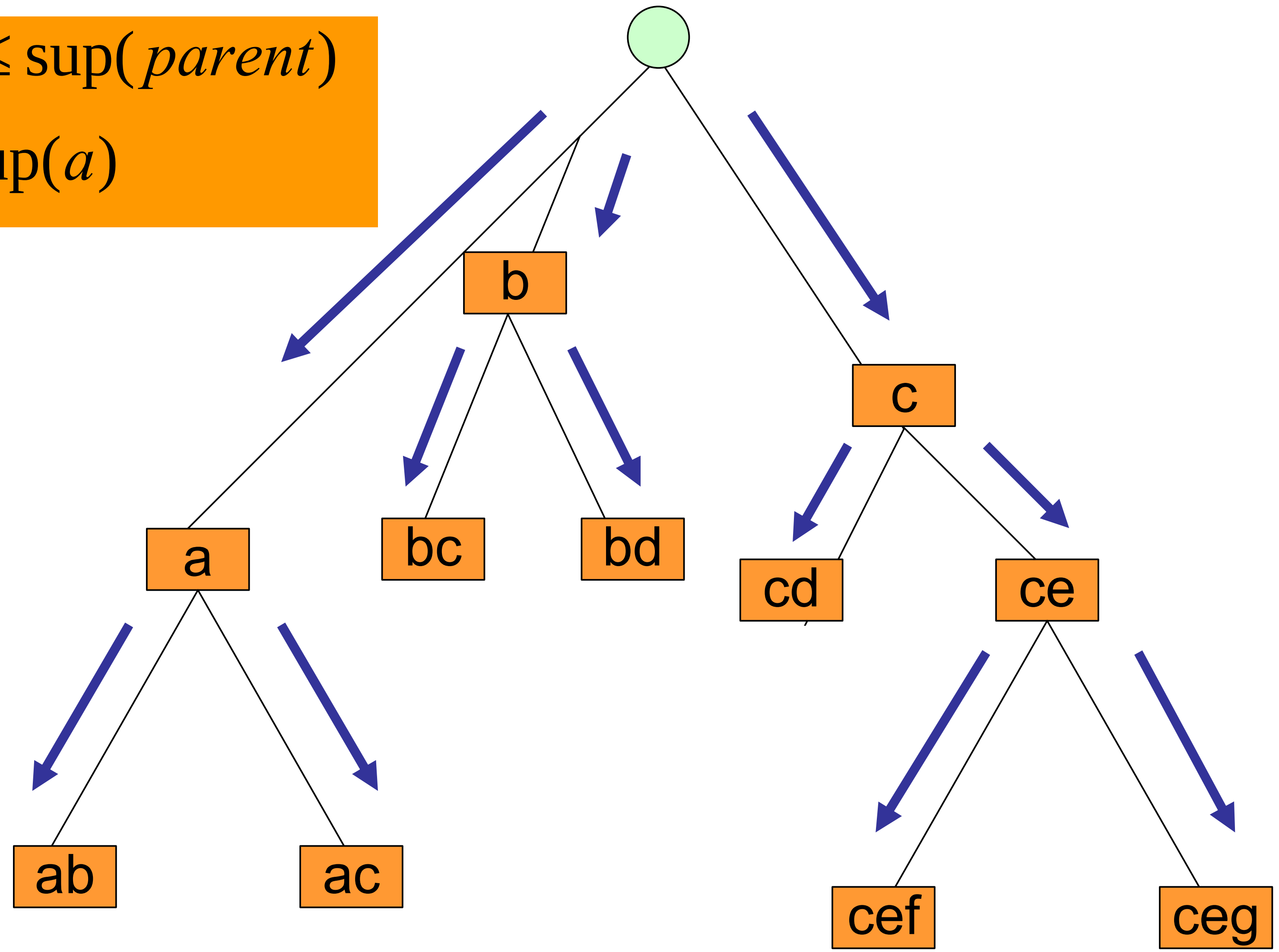
DDPMine [Cheng et al., ICDE'08]

- **Basic idea**
  - Integration of branch-and-bound search with FP-growth mining
  - Iteratively eliminate training instance and progressively shrink FP-tree
- **Performance**
  - Maintain high accuracy
  - Improve mining efficiency

FP-growth Mining with Depth-first Search

$$\sup(child) \leq \sup(parent)$$

$$\sup(ab) \leq \sup(a)$$



Branch-and-Bound Search

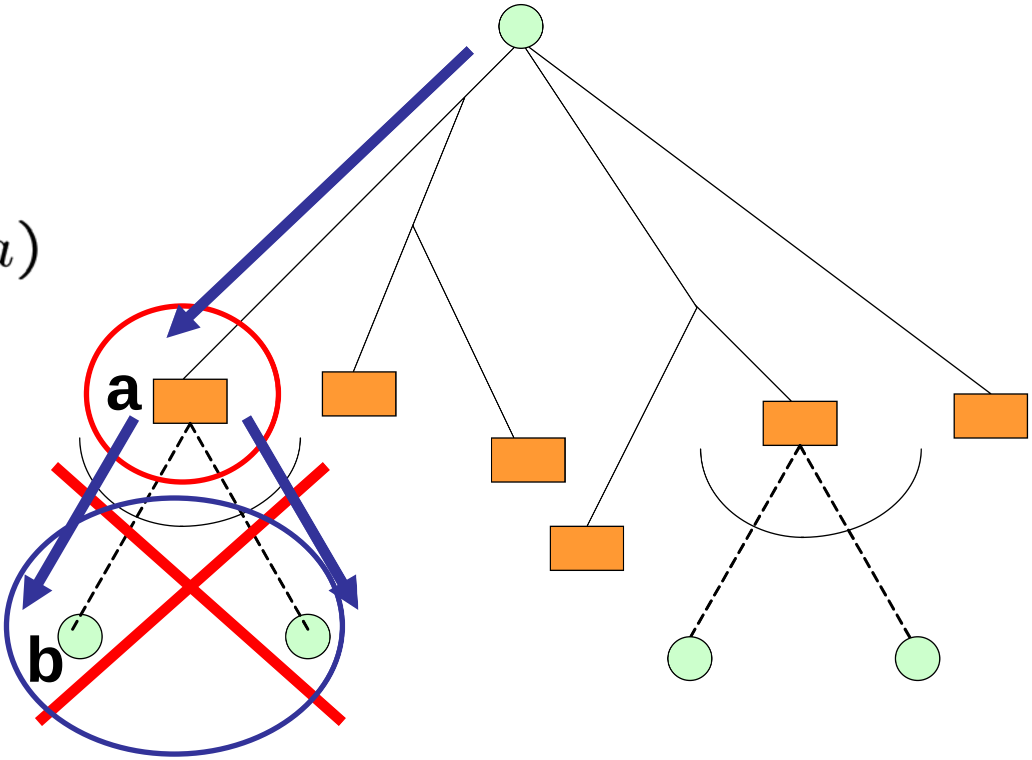
maximize $IG(C|b)$

subject to

$min\_sup \leq sup(b) \leq sup(a)$

$0 \leq sup_{+}(b) \leq sup_{+}(a)$

$0 \leq sup_{-}(b) \leq sup_{-}(a)$

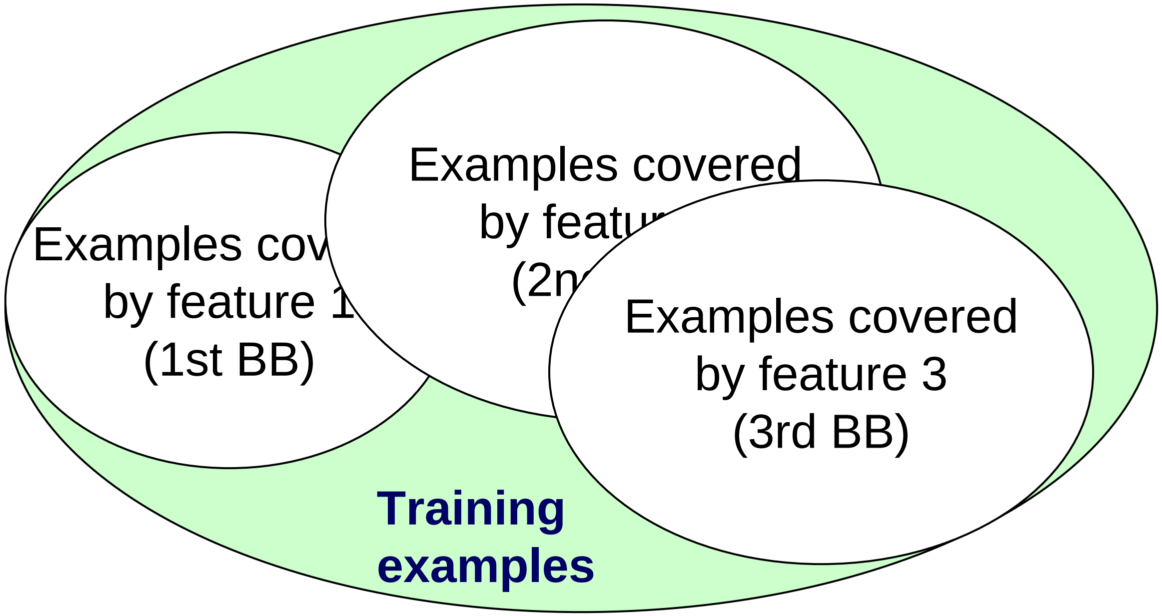


a: constant, a parent node

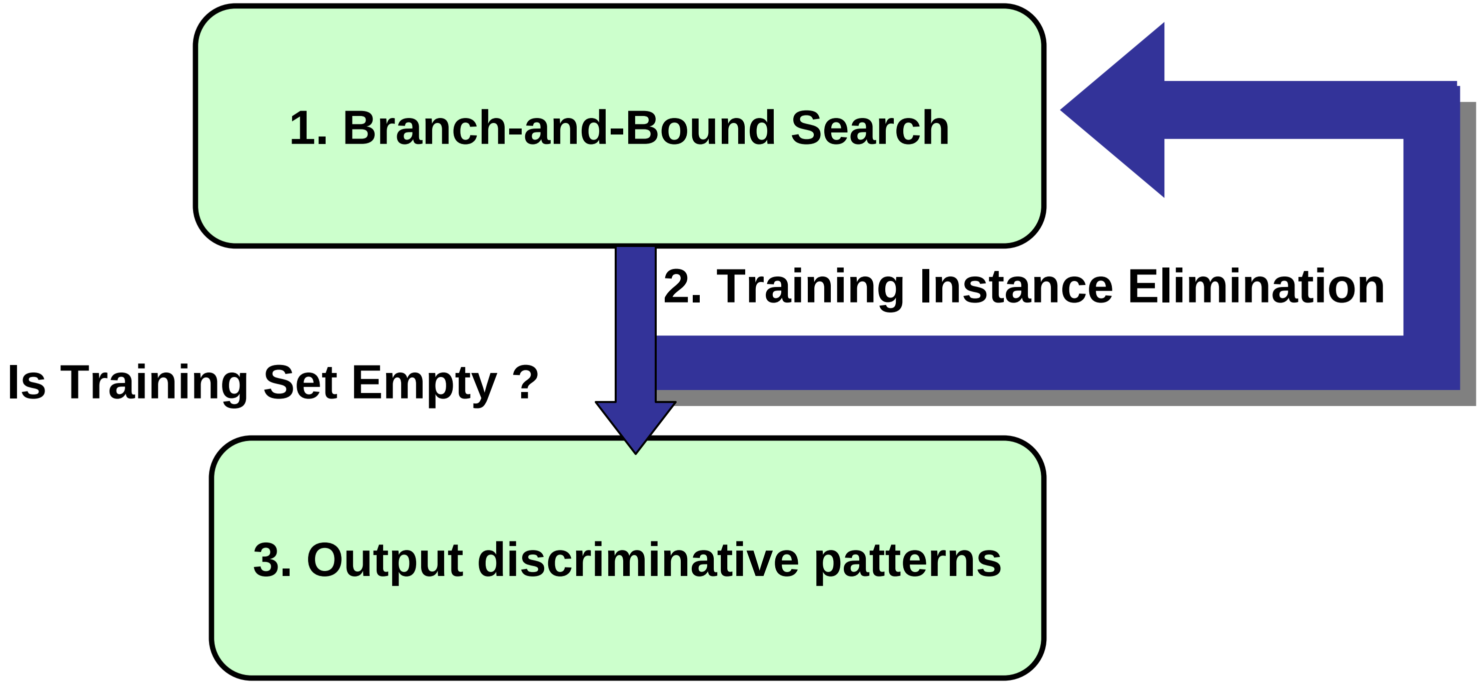
b: variable, a descendent

Association between information gain and frequency

Training Instance Elimination



DDPMine Algorithm Pipeline



Efficiency Analysis: Iteration Number

- $min\_sup = \theta_0$; frequent itemset α_i at i-th iteration since $|T(\alpha_i)| \geq \theta_0 |D_{i-1}|$

$$\boxed{|D_i| = |D_{i-1}| - |T(\alpha_i)| \leq (1-\theta_0)|D_{i-1}| \leq \ldots \leq (1-\theta_0)^i |D_0|}$$

- **Number of iterations:**

$$\boxed{n \leq \log_{\frac{1}{1-\theta_0}} |D_0|}$$

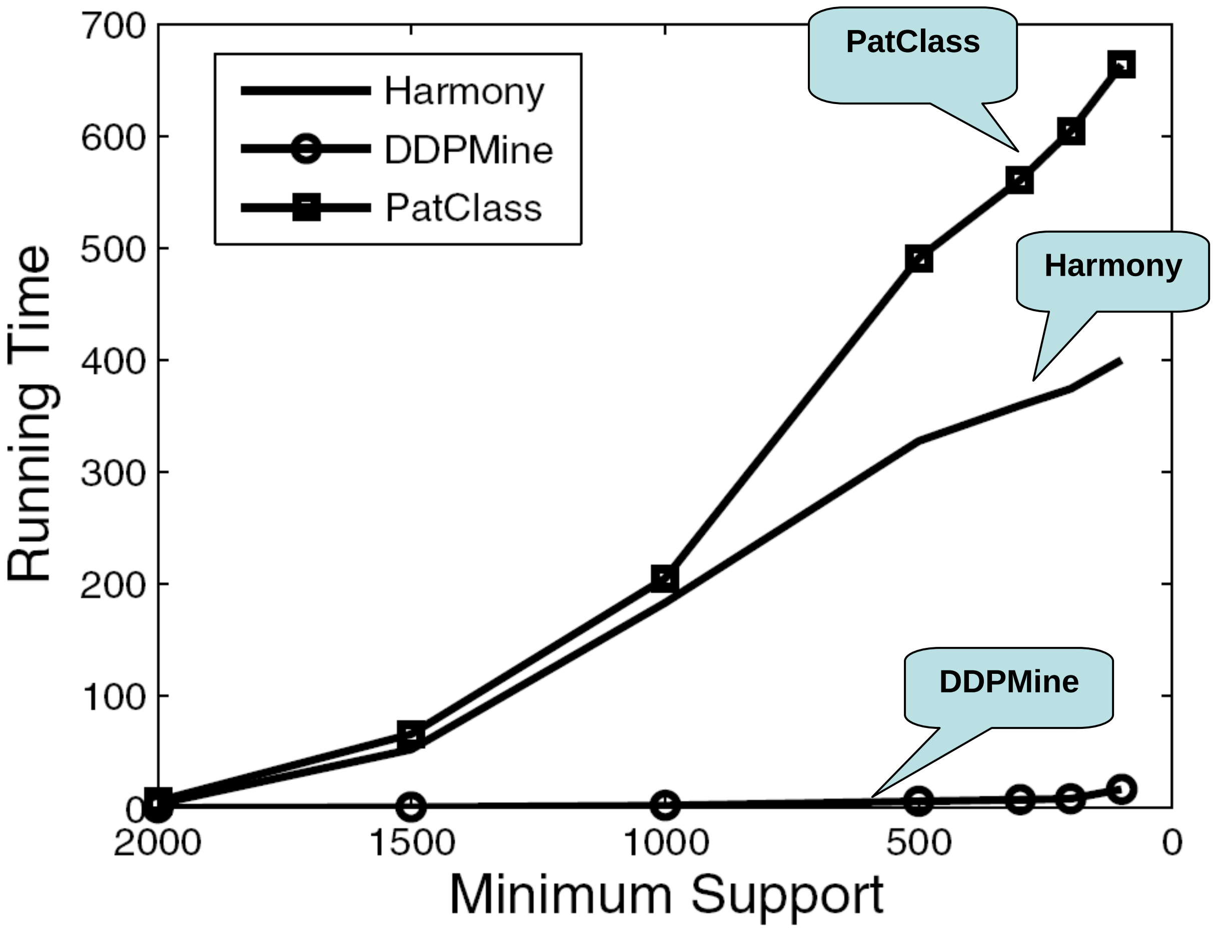
- If $\theta_0 = 0.5$ $n \leq \log_2 |D_0|$; $\theta_0 = 0.2$ $n \leq \log_{1.25} |D_0|$

Accuracy

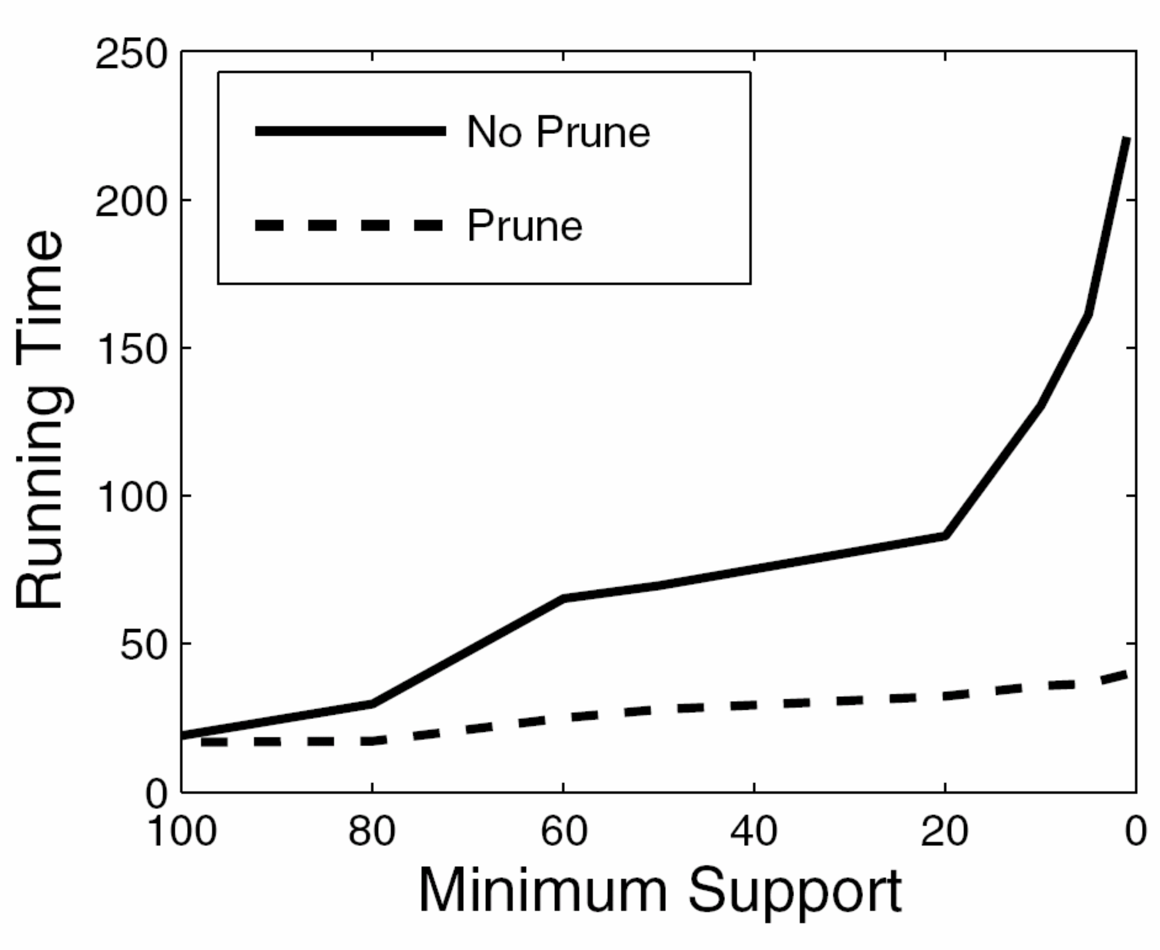
Datasets	Harmony	PatClass	DDPMine
adult	81.90	84.24	84.82
chess	43.00	91.68	91.85
crx	82.46	85.06	84.93
hypo	95.24	99.24	99.24
mushroom	99.94	99.97	100.00
sick	93.88	97.49	98.36
sonar	77.44	90.86	88.74
waveform	87.28	91.22	91.83
Average	82.643	92.470	92.471

Accuracy Comparison

Efficiency: Runtime



Branch-and-Bound Search: Runtime



Mining Most Significant Graph with Leap Search [Yan et al., SIGMOD'08]

Given a graph dataset D and an objective function $F(g)$, find a graph pattern g^*, s.t.

$$g^* = arg\ max_g\ F(g).$$

Objective functions

(1) Contrast: p/q,

(2) G-test: $p \cdot ln\frac{p}{q} + (1-p) \cdot ln\frac{1-p}{1-q}$,

(3) Information Gain: $H(C) - H(C|X)$

(4) Cosine

(5) many others.

Upper-Bound

Idea: derive an upper bound, $\widehat{F}(g)$, s.t., $\widehat{F}(g)$ is monotonic to freq(g).

$$G_t(p,q) = p \cdot ln\frac{p}{q} + (1-p) \cdot ln\frac{1-p}{1-q},$$

$$\begin{aligned} \frac{\partial G_t}{\partial q} &= \frac{q-p}{(1-q)q}, \\ \frac{\partial G_t}{\partial p} &= ln\frac{p(1-q)}{q(1-p)}. \end{aligned}$$

Since $\frac{p(1-q)}{q(1-p)} < 1$ when $p < q$, hence,

$$\text{if } p > q, \frac{\partial G_t}{\partial p} > 0, \frac{\partial G_t}{\partial q} < 0, \tag{1}$$

$$\text{if } p < q, \frac{\partial G_t}{\partial p} < 0, \frac{\partial G_t}{\partial q} > 0. \tag{2}$$

Upper-Bound: Anti-Monotonic

$$\text{if } p > q, \frac{\partial G_t}{\partial p} > 0, \frac{\partial G_t}{\partial q} < 0, \quad (1)$$

$$\text{if } p < q, \frac{\partial G_t}{\partial p} < 0, \frac{\partial G_t}{\partial q} > 0. \quad (2)$$

Rule of Thumb :
If the frequency difference of a graph pattern in the positive dataset and the negative dataset increases, the pattern becomes more interesting

small number

$$F(g) = F(p, q) < \max(\underline{F(p, \epsilon)}, \underline{F(\epsilon, q)}).$$

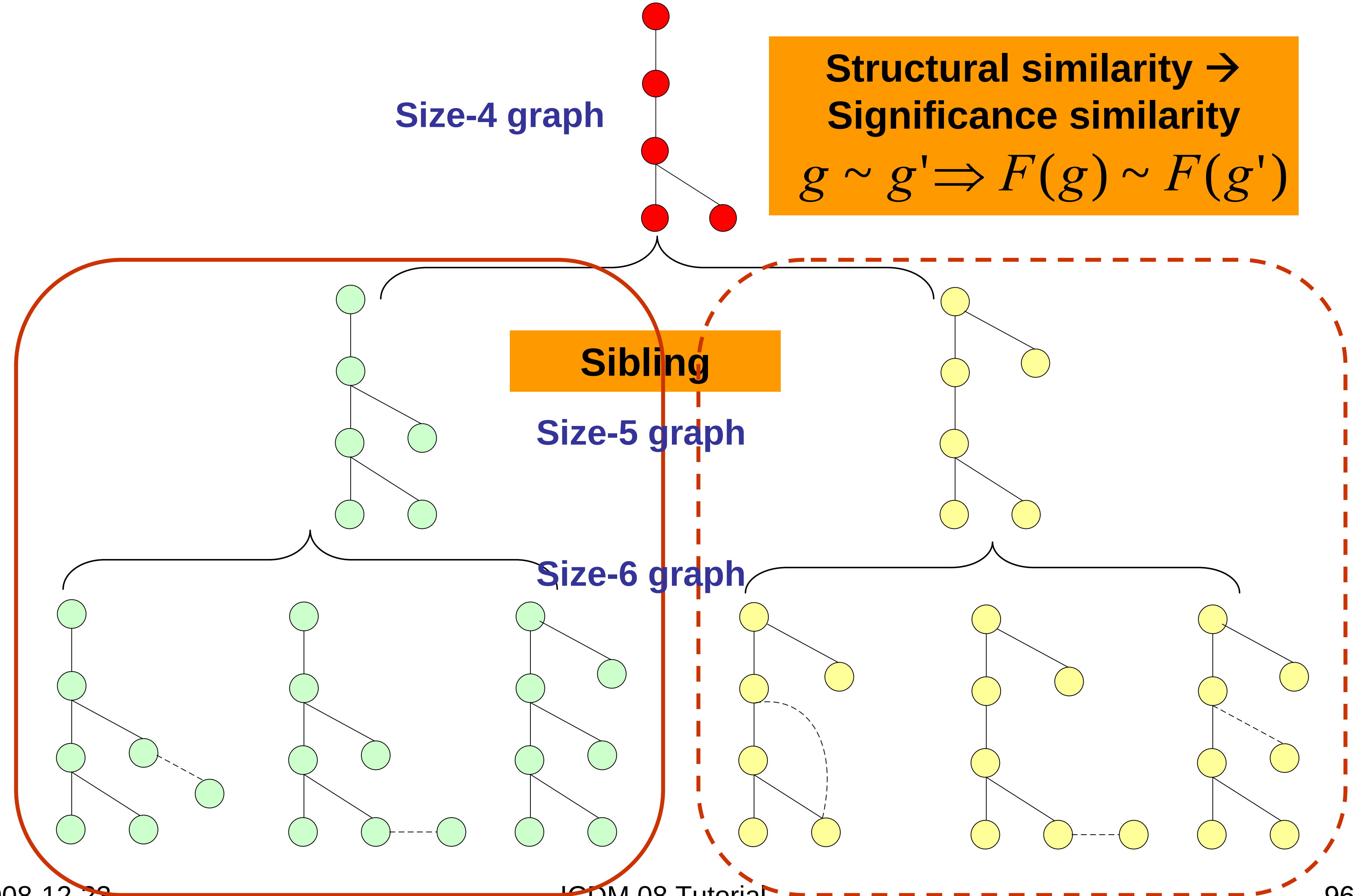
Monotonic to p Monotonic to q

We can recycle the existing graph mining algorithms to accommodate non-monotonic functions.

Structural Similarity

Structural similarity → Significance similarity

$$g \sim g' \Rightarrow F(g) \sim F(g')$$



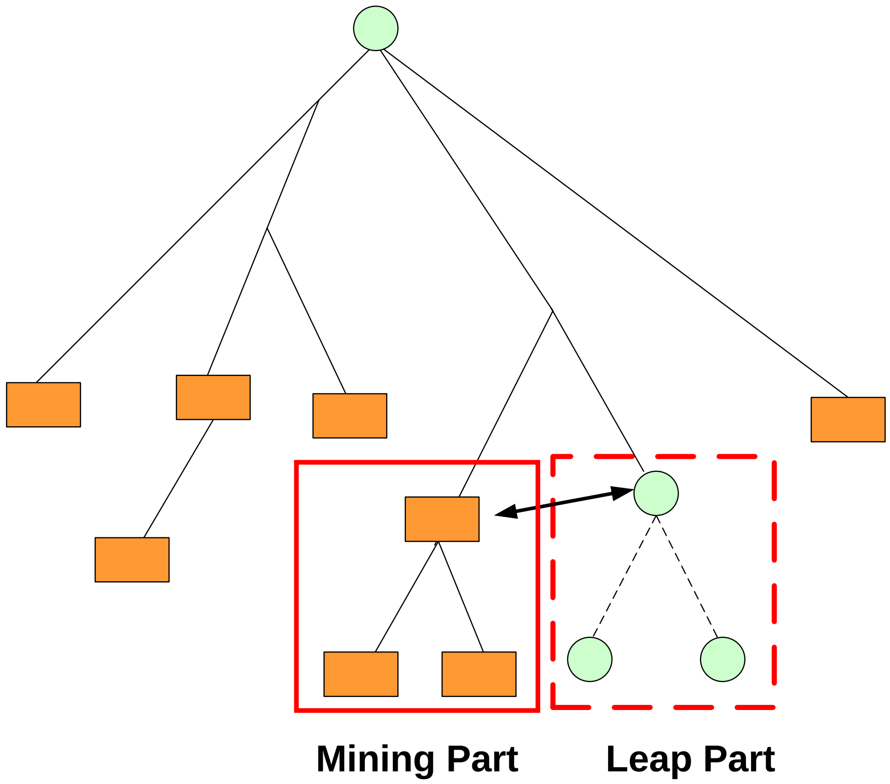
Structural Leap Search

Leap on g' subtree if

$$\frac{2\Delta_{+}(g,g')}{\sup_{+}(g)+\sup_{+}(g')} \leq \sigma$$

$$\frac{2\Delta_{-}(g,g')}{\sup_{-}(g)+\sup_{-}(g')} \leq \sigma$$

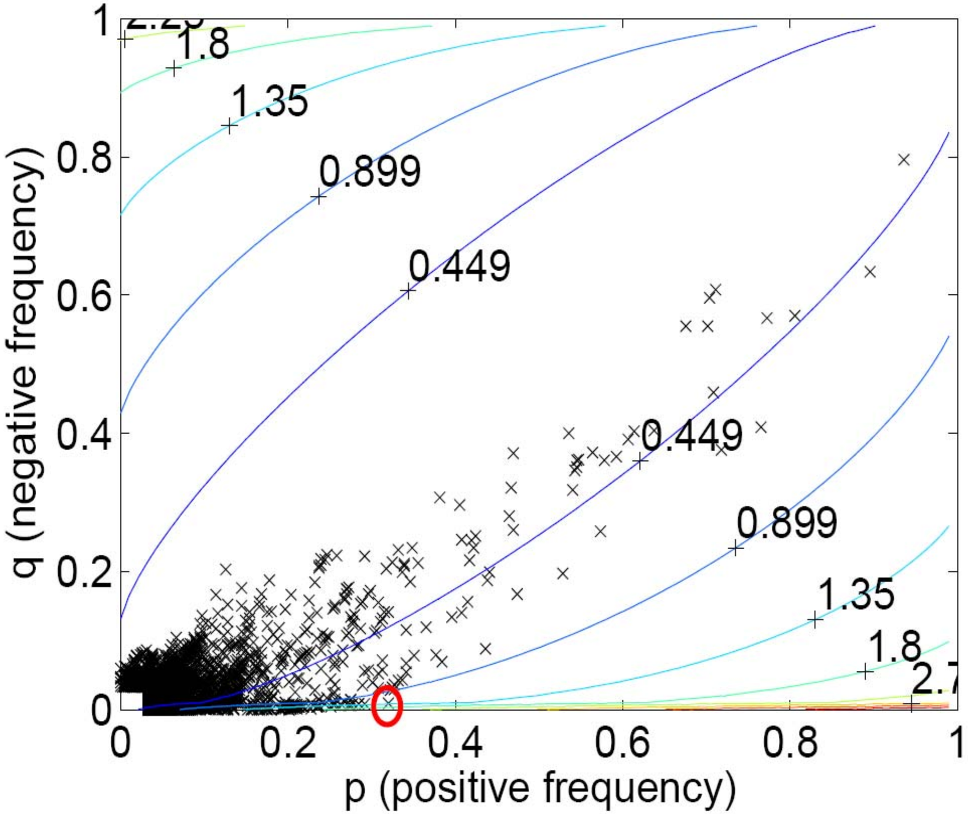
σ **:** leap length, tolerance of structure/frequency dissimilarity



g : a discovered graph

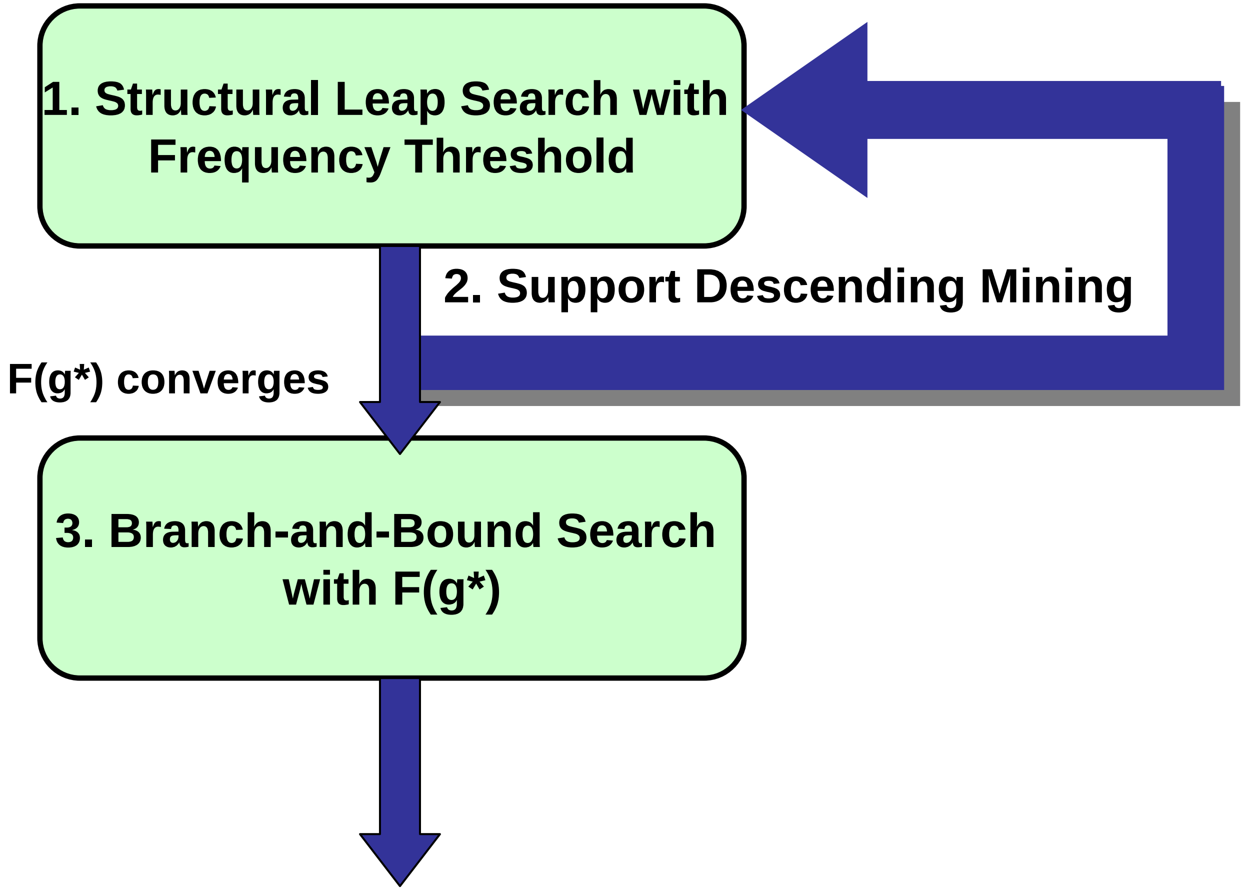
g': a sibling of g

Frequency Association



Association between pattern's frequency and objective scores
Start with a high frequency threshold, gradually decrease it

LEAP Algorithm



Branch-and-Bound vs. LEAP

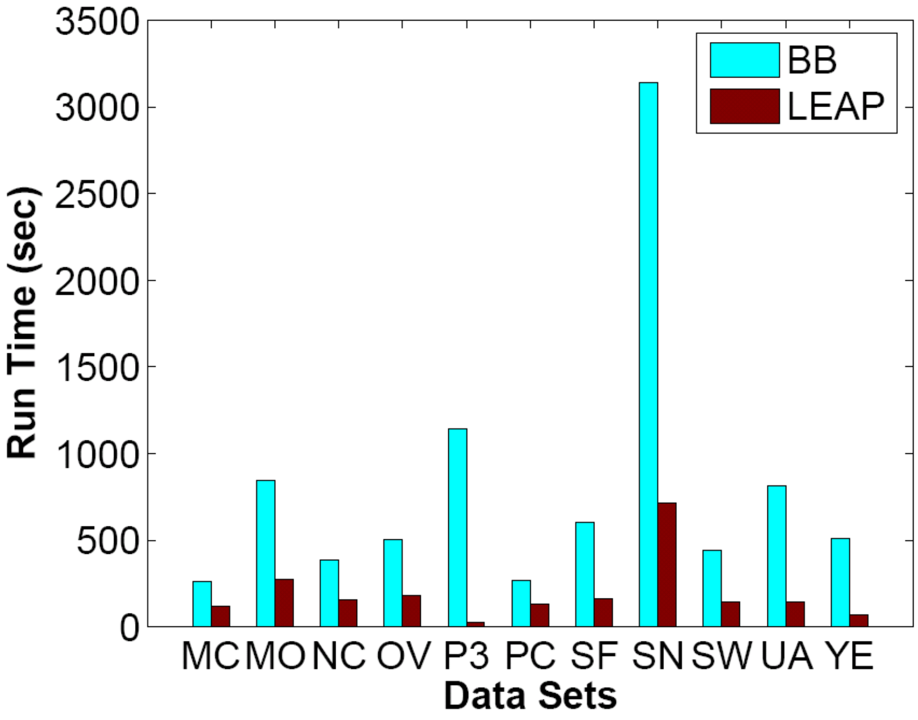
	Branch-and-Bound	LEAP
Pruning base	Parent-child bound ("vertical") strict pruning	Sibling similarity ("horizontal") approximate pruning
Feature Optimality	Guaranteed	Near optimal
Efficiency	Good	Better

NCI Anti-Cancer Screen Datasets

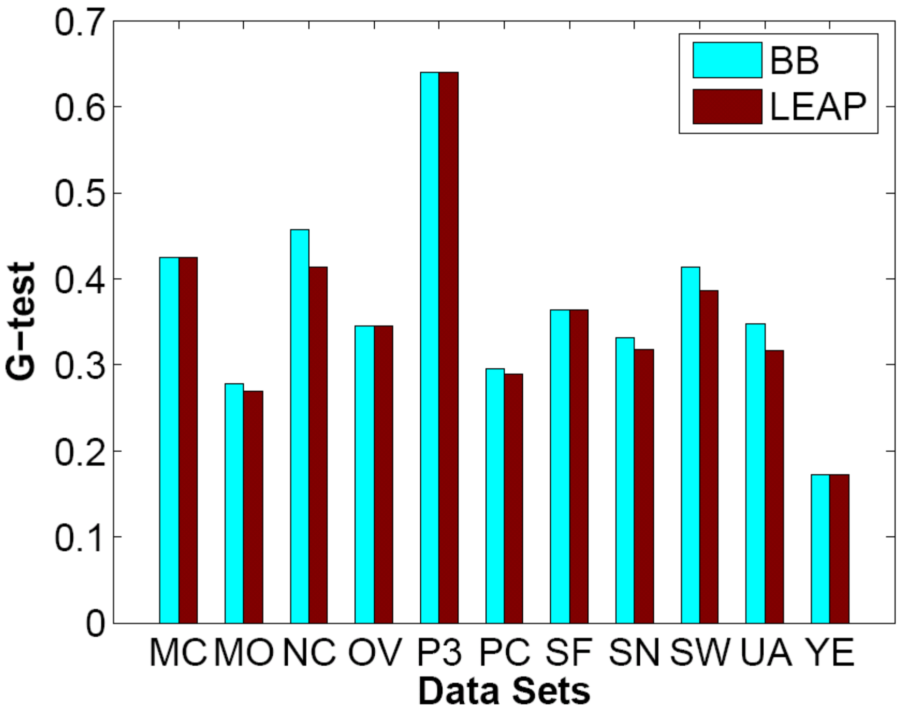
Name	Assay ID	Size	Tumor Description
MCF-7	83	27,770	Breast
MOLT-4	123	39,765	Leukemia
NCI-H23	1	40,353	Non-Small Cell Lung
OVCAR-8	109	40,516	Ovarian
P388	330	41,472	Leukemia
PC-3	41	27,509	Prostate
SF-295	47	40,271	Central Nerve System
SN12C	145	40,004	Renal
SW-620	81	40,532	Colon
UACC257	33	39,988	Melanoma
YEAST	167	79,601	Yeast anti-cancer

Data Description

Efficiency Tests



Search Efficiency

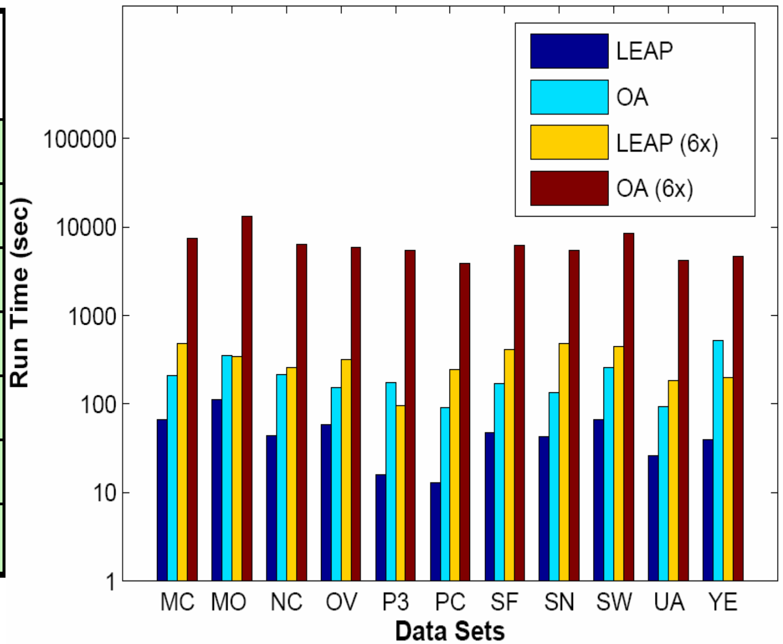


Search Quality: G-test

Mining Quality: Graph Classification

Name	OA Kernel*	LEAP	OA Kernel (6x)	LEAP (6x)
MCF-7	0.68	0.67	0.75	0.76
MOLT-4	0.65	0.66	0.69	0.72
NCI-H23	0.79	0.76	0.77	0.79
OVCAR-8	0.67	0.72	0.79	0.78
P388	0.79	0.82	0.81	0.81
PC-3	0.66	0.69	0.79	0.76
Average	0.70	0.72	0.75	0.77

AUC



Runtime

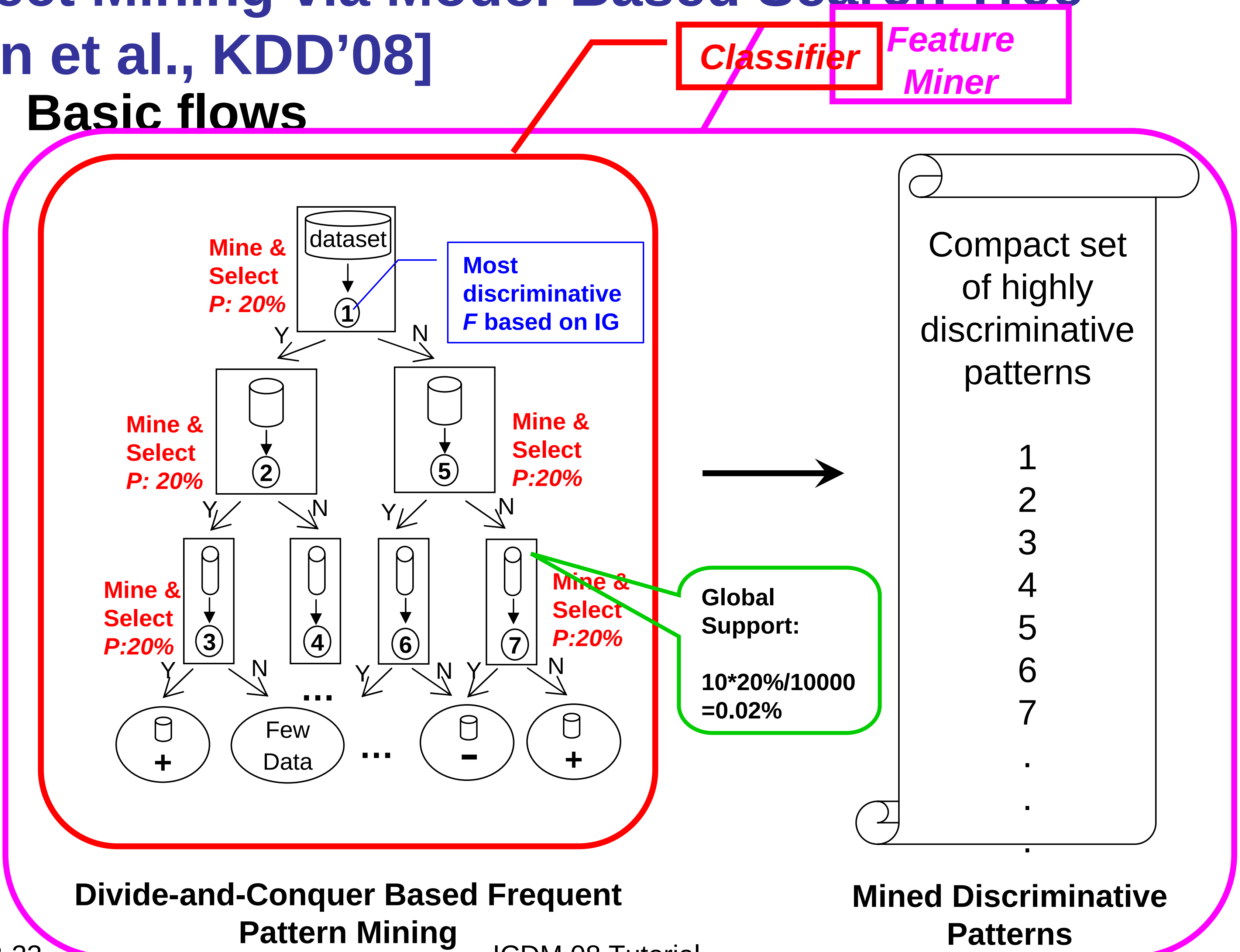
* **OA Kernel**: Optimal Assignment Kernel [Frohlich et al., ICML'05]

LEAP: LEAP search

OA Kernel $O(n^2m^3)$ scalability problem!

Direct Mining via Model-Based Search Tree [Fan et al., KDD'08]

- **Basic flows**



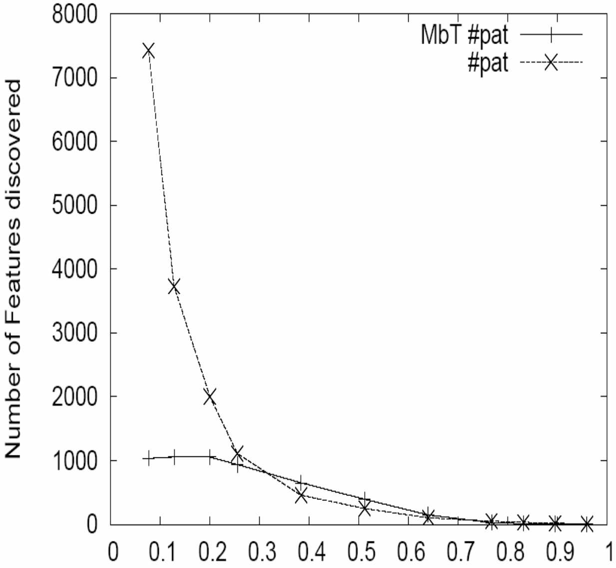
Analyses (I)

1. Scalability of pattern enumeration

- Upper bound: $O\left(s^{s(1-p)}\right)$

- "Scale down" ratio:

$$\simeq s^{s(1-p)}/s^{s} = \frac{1}{s^{sp}}$$



(a) Number of itemsets mined with varying supports

2. Bound on number of returned features

$$O\left(2^{\log_m(s)-1}-1\right) < O\left(s/2-1\right) = O(s) = O(n)$$

Analyses (II)

3. Subspace pattern selection

$$IG(\mathcal{C}|X) = H(\mathcal{C}) - H(\mathcal{C}|X)$$
$$= -\sum_{c\in\{0,1\}} P(c)\log P(c) + \sum_{x\in\{0,1\}} P(x) \sum_{c\in\{0,1\}} P(c|x)\log P(c|x)$$

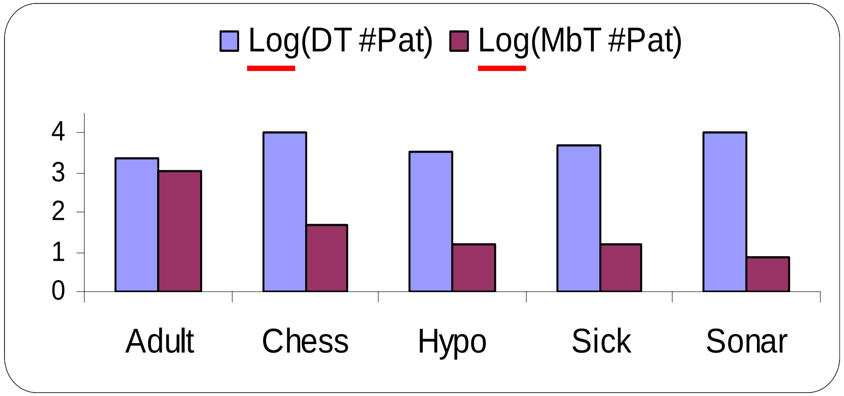
- Original set: $\frac{P_0}{C_0} \sim \frac{P_1}{C_1}$
- Subset: $\frac{P_0}{C_0} \ll \frac{P_1}{C_1}$ or vice versa

4. Non-overfitting

5. Optimality under exhaustive search

Experimental Study: Itemset Mining (I)

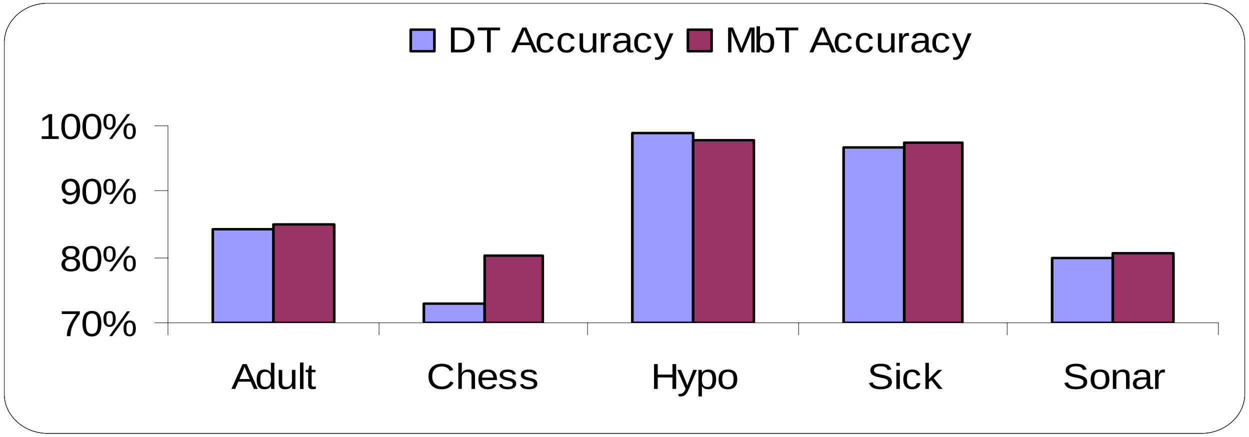
- **Scalability comparison**

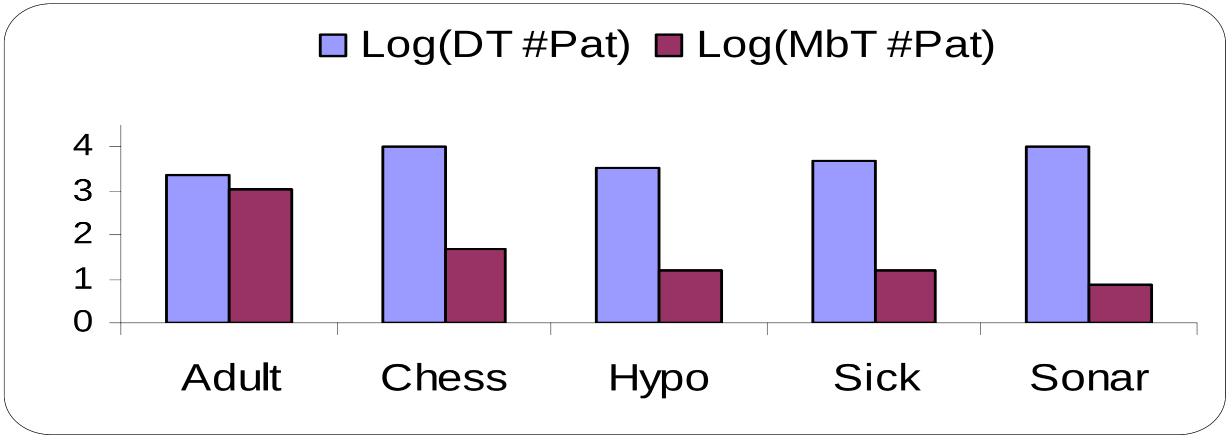


Datasets	MbT #Pat	#Pat using MbT sup	Ratio (MbT #Pat / #Pat using MbT sup)
Adult	1039.2	252809	0.41%
Chess	46.8	+∞	~0%
Hypo	14.8	423439	0.0035%
Sick	15.4	4818391	0.00032%
Sonar	7.4	95507	0.00775%

Experimental Study: Itemset Mining (II)

- **Accuracy of mined itemsets**





4 Wins
1 loss

much smaller number of patterns

Tutorial Outline

- Frequent Pattern Mining
- Classification Overview
- Associative Classification
- Substructure-Based Graph Classification
- Direct Mining of Discriminative Patterns
- Integration with Other Machine Learning Techniques
- Conclusions and Future Directions

Integrated with Other Machine Learning Techniques

- **Boosting**
  - Boosting an associative classifier [Sun, Wang and Wong, TKDE'06]
  - Graph classification with boosting [Kudo, Maeda and Matsumoto, NIPS'04]

- **Sampling and ensemble**
  - Data and feature ensemble for graph classification [Cheng et al., In preparation]

Boosting An Associative Classifier [Sun, Wang and Wong, TKDE'06]

- Apply AdaBoost to associative classification with low-order rules

- **Three weighting strategies for combining classifiers**
  - Classifier-based weighting (AdaBoost)

$$H(\underline{x}) = arg \max_{y_i, i=1..k} \left(\sum_{t=1}^{T} \alpha_t I[h_t(\underline{x}) = y_i] \right) \qquad \alpha = \frac{1}{2} ln \frac{1-\varepsilon}{\varepsilon}$$

  - Sample-based weighting (Evaluated to be the best)

$$H(\underline{x}) = arg \max_{y_i, i=1..k} \left(\sum_{t=1}^{T} r_{ti} I[h_t(\underline{x}) = y_i] \right) \qquad r_i = W(Y = y_i / Y \neq y_i | \underline{x}) = \log \frac{P(\underline{x}|Y = y_i)}{P(\underline{x}|Y \neq y_i)}$$

  - Hybrid weighting

$$H(\underline{x}) = arg \max_{y_i, i=1..k} \left(\sum_{t=1}^{T} \alpha_t r_{ti} I[h_t(\underline{x}) = y_i] \right)$$

Graph Classification with Boosting [Kudo, Maeda and Matsumoto, NIPS'04]

- **Decision stump** $< t, y >$
  - If a molecule x contains t, it is classified as y

$$h_{<t,y>}(\mathrm{x}) = \begin{cases} y & if\ t \subseteq \mathrm{x}, \\ -y & otherwise \end{cases}$$

- **Gain** $gain(< t, y >) = \sum_{i=1}^{n} y_i h_{<t,y>}(\mathrm{x}_i)$
  - **Find a decision stump (subgraph) which maximizes gain**

- **Boosting with weight vector** $\mathrm{d}^{(\mathrm{k})} = (d_1^{(k)}, \ldots, d_n^{(k)})$

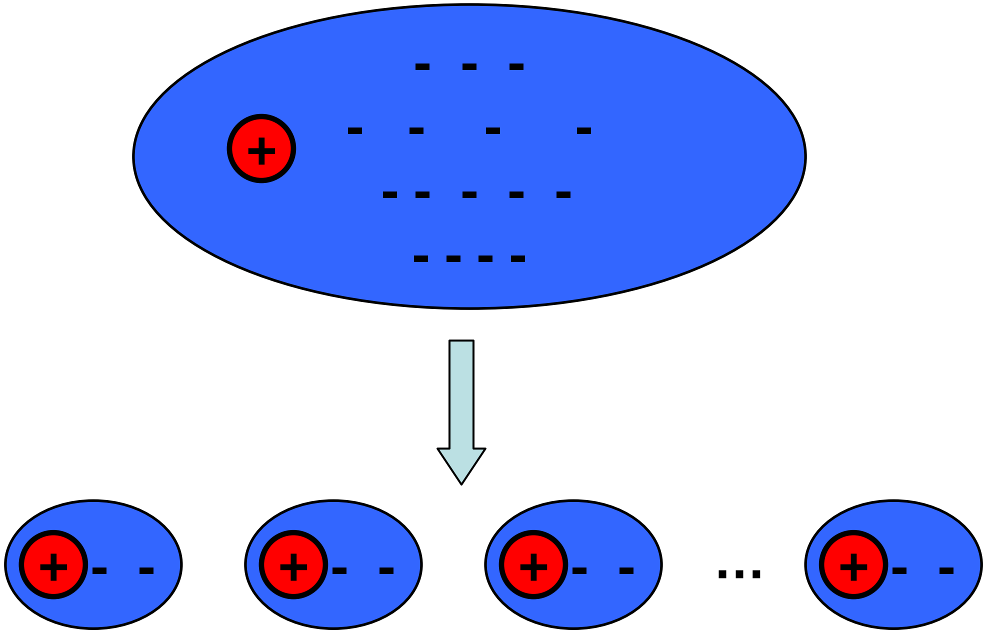
$$gain(< t, y >) = \sum_{i=1}^{n} y_i d_i^{(k)} h_{<t,y>}(\mathrm{x}_i)$$

Sampling and Ensemble [Cheng et al., In Preparation]

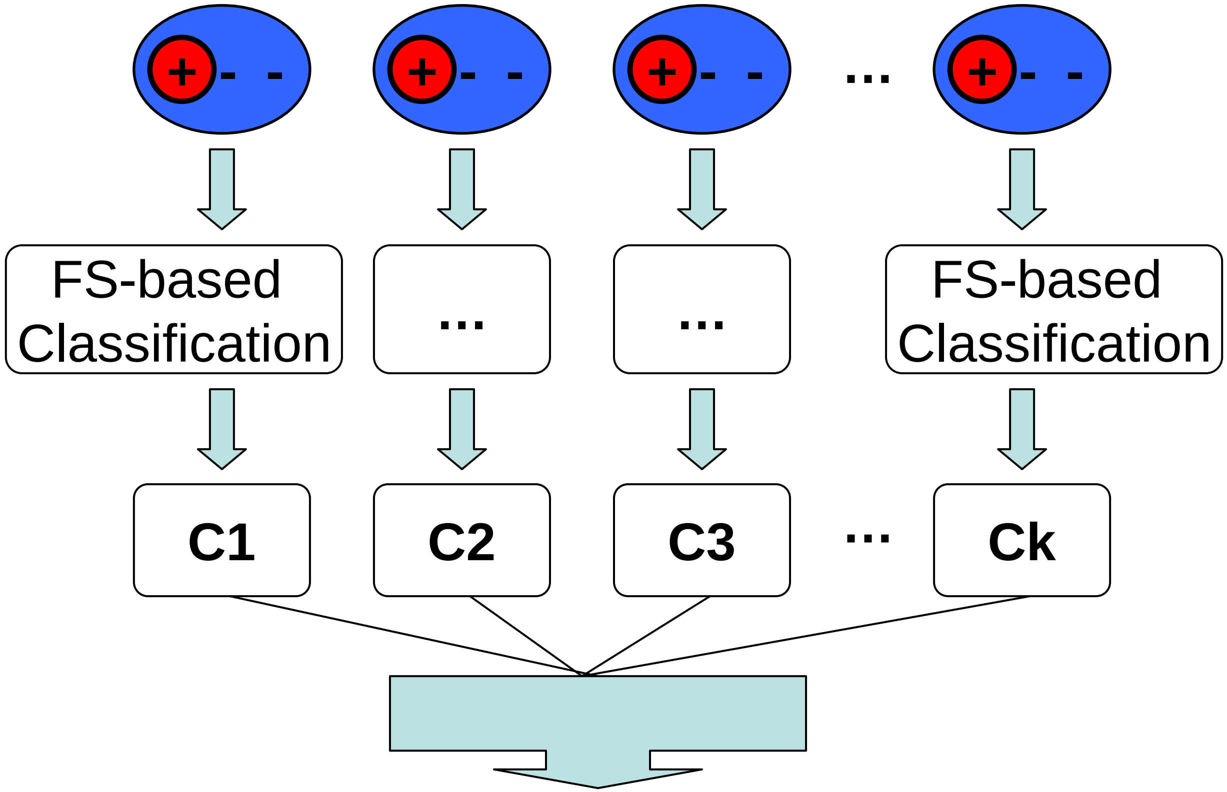
- **Many real graph datasets are extremely skewed**
  - Aids antiviral screen data: **1%** active samples
  - NCI anti-cancer data: **5%** active samples
- Traditional learning methods tend to be biased towards the majority class and ignore the minority class
- The cost of misclassifying minority examples is usually huge

Sampling

- Repeated samples of the positive class
- Under-samples of the negative class
- Re-balance the data distribution



Balanced Data Ensemble

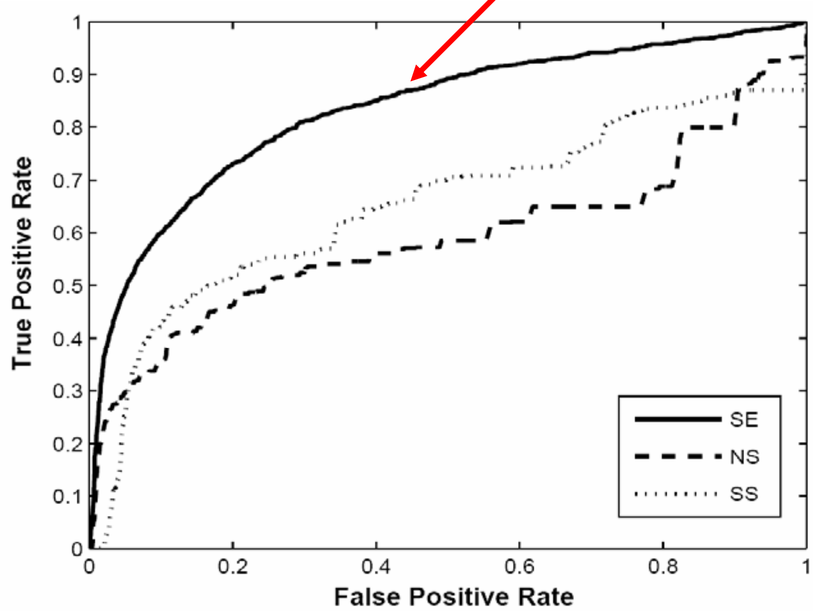


$$f^E(x) = \frac{1}{k}\sum_{i=1}^{k} f^i(x)$$

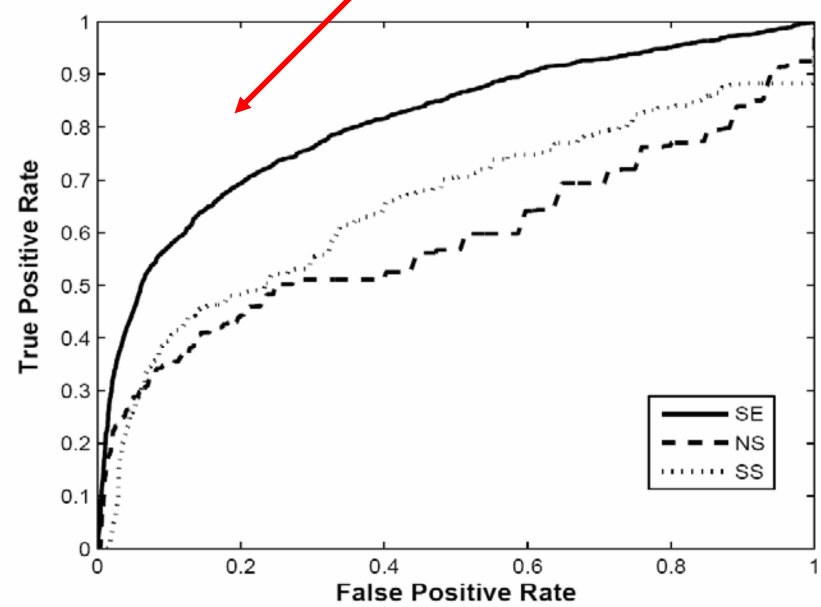
The error of each classifier is independent, could be reduced through ensemble.

ROC Curve

Sampling and ensemble



(a) ROC, NCI1



(b) ROC, NCI81

ROC50 Comparison

Datasets	SE	FS	GF
NCI1	**0.5318**	0.2630	0.3260
NCI109	**0.6149**	0.2380	0.3020
NCI123	**0.6059**	0.2400	0.2630
NCI145	**0.5716**	0.2650	0.3400
NCI167	**0.5059**	0.0540	0.0640
NCI33	**0.5815**	0.2510	0.3180
NCI330	**0.4847**	0.2420	0.3430
NCI41	**0.5809**	0.3000	0.3570
NCI47	**0.6002**	0.2430	0.3110
NCI81	**0.5406**	0.2390	0.2950
NCI83	**0.6113**	0.2670	0.3170
H1	**0.5878**	0.2280	0.2680
H2	0.6086	0.5810	**0.6510**

SE: Sampling + Ensemble

FS: Single model with frequent subgraphs

GF: Single model with graph fragments

Tutorial Outline

- ❑ Frequent Pattern Mining
- ❑ Classification Overview
- ❑ Associative Classification
- ❑ Substructure-Based Graph Classification
- ❑ Direct Mining of Discriminative Patterns
- ❑ Integration with Other Machine Learning Techniques
- ❑ Conclusions and Future Directions

Conclusions

- Frequent pattern is a discriminative feature in classifying both structured and unstructured data.
- Direct mining approach can find the most discriminative pattern with significant speedup.
- When integrated with boosting or ensemble, the performance of pattern-based classification can be further enhanced.

Future Directions

- **Mining more complicated patterns**
  - Direct mining top-k significant patterns
  - Mining approximate patterns
- **Integration with other machine learning tasks**
  - Semi-supervised and unsupervised learning
  - Domain adaptive learning
- **Applications: Mining colossal discriminative patterns?**
  - Software bug detection and localization in large programs
  - Outlier detection in large networks
    - Money laundering in wired transfer network
    - Web spam in internet

References (1)

- R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan. **Automatic Subspace Clustering of High Dimensional Data for Data Mining Applications,** SIGMOD'98.
- R. Agrawal and R. Srikant. **Fast Algorithms for Mining Association Rules,** VLDB'94.
- C. Borgelt, and M.R. Berthold. **Mining Molecular Fragments: Finding Relevant Substructures of Molecules,** ICDM'02.
- C. Chen, X. Yan, P.S. Yu, J. Han, D. Zhang, and X. Gu, **Towards Graph Containment Search and Indexing,** VLDB'07.
- C. Cheng, A.W. Fu, and Y. Zhang. **Entropy-based Subspace Clustering for Mining Mumerical Data,** KDD'99.
- H. Cheng, X. Yan, and J. Han. **Seqindex: Indexing Sequences by Sequential Pattern Analysis,** SDM'05.
- H. Cheng, X. Yan, J. Han, and C.-W. Hsu, **Discriminative Frequent Pattern Analysis for Effective Classification**, ICDE'07.
- H. Cheng, X. Yan, J. Han, and P. S. Yu, **Direct Discriminative Pattern Mining for Effective Classification**, ICDE'08.
- H. Cheng, W. Fan, X. Yan, J. Gao, J. Han, and P. S. Yu, **Classification with Very Large Feature Sets and Skewed Distribution**, In Preparation.
- J. Cheng, Y. Ke, W. Ng, and A. Lu. **FG-Index: Towards Verification-Free Query Processing on Graph Databases,** SIGMOD'07.

References (2)

- G. Cong, K. Tan, A. Tung, and X. Xu. **Mining Top-k Covering Rule Groups for Gene Expression Data,** SIGMOD'05.
- M. Deshpande, M. Kuramochi, N. Wale, and G. Karypis. **Frequent Substructure-based Approaches for Classifying Chemical Compounds,** TKDE'05.
- G. Dong and J. Li. **Efficient Mining of Emerging Patterns: Discovering Trends and Differences,** KDD'99.
- G. Dong, X. Zhang, L. Wong, and J. Li. **CAEP: Classification by Aggregating Emerging Patterns,** DS'99
- R. O. Duda, P. E. Hart, and D. G. Stork. **Pattern Classification (2nd ed.),** John Wiley & Sons, 2001.
- W. Fan, K. Zhang, H. Cheng, J. Gao, X. Yan, J. Han, P. S. Yu, and O. Verscheure. **Direct Mining of Discriminative and Essential Graphical and Itemset Features via Model-based Search Tree**, KDD'08.
- J. Han and M. Kamber. **Data Mining: Concepts and Techniques (2nd ed.),** Morgan Kaufmann, 2006.
- J. Han, J. Pei, and Y. Yin. **Mining Frequent Patterns without Candidate Generation,** SIGMOD'00.
- T. Hastie, R. Tibshirani, and J. Friedman. **The Elements of Statistical Learning,** Springer, 2001.
- D. Heckerman, D. Geiger and D. M. Chickering. **Learning Bayesian Networks: The Combination of Knowledge and Statistical Data,** Machine Learning, 1995.

References (3)

- T. Horvath, T. Gartner, and S. Wrobel. **Cyclic Pattern Kernels for Predictive Graph Mining,** KDD'04.
- J. Huan, W. Wang, and J. Prins. **Efficient Mining of Frequent Subgraph in the Presence of Isomorphism,** ICDM'03.
- A. Inokuchi, T. Washio, and H. Motoda. **An Apriori-based Algorithm for Mining Frequent Substructures from Graph Data,** PKDD'00.
- T. Kudo, E. Maeda, and Y. Matsumoto. **An Application of Boosting to Graph Classification,** NIPS'04.
- M. Kuramochi and G. Karypis. **Frequent Subgraph Discovery,** ICDM'01.
- W. Li, J. Han, and J. Pei. **CMAR: Accurate and Efficient Classification based on Multiple Class-association Rules,** ICDM'01.
- B. Liu, W. Hsu, and Y. Ma. **Integrating Classification and Association Rule Mining,** KDD'98.
- H. Liu, J. Han, D. Xin, and Z. Shao. **Mining Frequent Patterns on Very High Dimensional Data: A Topdown Row Enumeration Approach,** SDM'06.
- S. Nijssen, and J. Kok. **A Quickstart in Frequent Structure Mining Can Make a Difference,** KDD'04.
- F. Pan, G. Cong, A. Tung, J. Yang, and M. Zaki. **CARPENTER: Finding Closed Patterns in Long Biological Datasets,** KDD'03

References (4)

- F. Pan, A. Tung, G. Cong G, and X. Xu. **COBBLER: Combining Column, and Row enumeration for Closed Pattern Discovery,** SSDBM'04.
- J. Pei, J. Han, B. Mortazavi-Asl, H. Pinto, Q. Chen, U. Dayal, and M-C. Hsu. **PrefixSpan: Mining Sequential Patterns Efficiently by Prefix-projected Pattern Growth,** ICDE'01.
- R. Srikant and R. Agrawal. **Mining Sequential Patterns: Generalizations and Performance Improvements,** EDBT'96.
- Y. Sun, Y. Wang, and A. K. C. Wong. **Boosting an Associative Classifier,** TKDE'06.
- P-N. Tan, V. Kumar, and J. Srivastava. **Selecting the Right Interestingness Measure for Association Patterns,** KDD'02.
- R. Ting and J. Bailey. **Mining Minimal Contrast Subgraph Patterns,** SDM'06.
- N. Wale and G. Karypis. **Comparison of Descriptor Spaces for Chemical Compound Retrieval and Classification,** ICDM'06.
- H. Wang, W. Wang, J. Yang, and P.S. Yu. **Clustering by Pattern Similarity in Large Data Sets,** SIGMOD'02.
- J. Wang and G. Karypis. **HARMONY: Efficiently Mining the Best Rules for Classification,** SDM'05.
- X. Yan, H. Cheng, J. Han, and P. S. Yu, **Mining Significant Graph Patterns by Scalable Leap Search**, SIGMOD'08.
- X. Yan and J. Han. **gSpan: Graph-based Substructure Pattern Mining,** ICDM'02.

References (5)

- X. Yan, P.S. Yu, and J. Han. **Graph Indexing: A Frequent Structure-based Approach,** SIGMOD’04.
- X. Yin and J. Han. **CPAR: Classification Based on Predictive Association Rules,** SDM’03.
- M.J. Zaki. **Scalable Algorithms for Association Mining,** TKDE’00.
- M.J. Zaki. **SPADE: An Efficient Algorithm for Mining Frequent Sequences,** Machine Learning’01.
- M.J. Zaki and C.J. Hsiao. **CHARM: An Efficient Algorithm for Closed Itemset mining,** SDM’02.
- F. Zhu, X. Yan, J. Han, P.S. Yu, and H. Cheng. **Mining Colossal Frequent Patterns by Core Pattern Fusion,** ICDE’07.

Questions?

hcheng@se.cuhk.edu.hk

http://www.se.cuhk.edu.hk/~hcheng