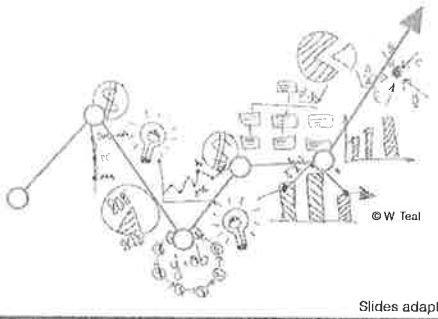
CSE 347/447: DATA MINING

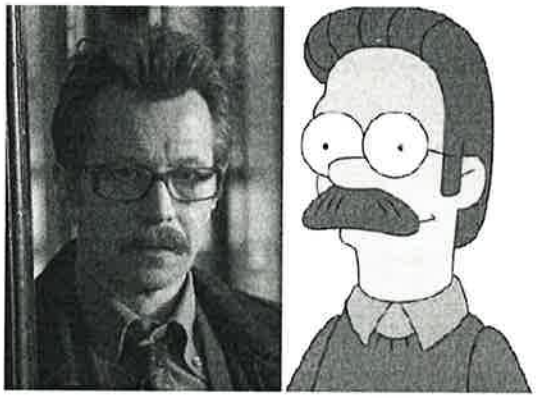
Lecture 4: Similarity



Slides adapted from TSK and LRU

Announcements

- Homework 1 is posted on CourseSite (due on Sep 15)
- According to the poll results, we will have multiple competitions for the course project



Definition

- Similarity 相似度
  - Numerical measure of how likely two objects are
  - Higher if two objects are more alike
  - Often scaled to [0, 1]
- Dissimilarity 相异度 distance
  - Numerical measure of how different two objects are
  - Lower if two objects are more alike
  - Minimum dissimilarity is typically 0 (i.e., identical objects)
- Proximity refers to both similarity or dissimilarity 邻近度.

Simple Attributes

- p and q are the attribute values of two data objects

Attribute Type	Dissimilarity	Similarity
Nominal	$d = \begin{cases} 0 & \text{if } p = q \\ 1 & \text{if } p \neq q \end{cases}$	$s = \begin{cases} 1 & \text{if } p = q \\ 0 & \text{if } p \neq q \end{cases}$
Ordinal	$d = \frac{\lvert p-q \rvert}{n-1}$ (values mapped to integers 0 to $n-1$, where n is the number of values)	$s = 1 - \frac{\lvert p-q \rvert}{n-1} = 1-d$
Interval or Ratio	$d = \lvert p - q \rvert$	$s = -d$, $s = \frac{1}{1+d}$ or $s = 1 - \frac{d - \min d}{\max d - \min d}$

Table 5.1. Similarity and dissimilarity for simple attributes

Euclidean Distance

- Defintion

$$dist = \sqrt{\sum_{k=1}^{n} (p_k - q_k)^2}$$

  - Where n is the number of dimensions (attributes) and p_k and q_k are respectively the k-th attributes of data objects p and q

- Standardization is necessary, if scales differ

Cosine Distance

- If p and q are two vectors, then their cosine similarity

$$\text{sim}(p, q) = \frac{p \cdot q}{||p||||q||}$$

  - where • indicates vector dot (inner) product and ||v|| is the L_2 norm of v
- Their cosine distance:
  - $\text{dist}(p, q) = 1 - \text{sim}(p, q)$

without considering magnitudes

失去数值意义. 改变数据特性.

Edit Distance

- Edit distance measures two strings x and y as the smallest number of insertion and deletions of single characters to convert x to y
  - x = *abcde*, y = acfdeg, dist(x, y) = 3
  - (1) delete *b*, (2) insert *f* after *c*, and (3) insert *g* after *e*
- Edit distance is computed using dynamic programming

Combining Similarities

- Overall similarity over different types of attributes

1. For the k^{th} attribute, compute a similarity, s_k, in the range $[0, 1]$.
2. Define an indicator variable, δ_k, for the k_{th} attribute as follows:

$$\delta_k = \begin{cases} 0 & \text{if the } k^{th} \text{ attribute is a binary asymmetric attribute and both objects have a value of 0, or if one of the objects has a missing values for the } k^{th} \text{ attribute} \\ 1 & \text{otherwise} \end{cases}$$

3. Compute the overall similarity between the two objects using the following formula:

$$similarity(p, q) = \frac{\sum_{k=1}^{n} \delta_k s_k}{\sum_{k=1}^{n} \delta_k}$$

当属性没有不同类型时:

$$d(p, q) = \frac{\sum_{k=1}^{n} \delta_k d_k}{\sum_{k=1}^{n} \delta_k}$$

A Common Metaphor

- **Many problems can be expressed as finding "similar" sets:**
  - **Find near-neighbors in high-dimensional space**
- Examples:
  - **Pages with similar words**
    - For duplicate detection, classification by topic
  - **Customers who purchased similar products**
    - Products with similar customer sets
  - **Users with similar behaviors**
    - Users who visited similar websites

Problem Definition

- **Given: High dimensional data points $x_1, x_2, \ldots$**
  - **For example:** Image is a long vector of pixel colors

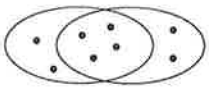
$$\begin{bmatrix} 1 & 2 & 1 \\ 0 & 2 & 1 \\ 0 & 1 & 0 \end{bmatrix} \rightarrow [1\,2\,1\,0\,2\,1\,0\,1\,0]$$

- **And some distance function $d(x_1, x_2)$**
  - Which quantifies the "distance" between x_1 and x_2
- **Goal:** Find **all pairs of data points** (x_i, x_j) that are within some distance threshold $d(x_i, x_j) \leq s$
- **Note:** Naïve solution would take $O(N^2)$ ⊗ where N is the number of data points
- **MAGIC: This can be done in $O(N)$!! How?**

Similarity Metric

- **Document D_1 is a set of its k-shingles $C_1=S(D_1)$**
- Equivalently, each document is a 0/1 vector in the space of *k*-shingles
  - Each unique shingle is a dimension
  - Vectors are very sparse
- **A natural similarity measure is the Jaccard similarity:**

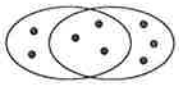
  $sim(D_1, D_2) = |C_1 \cap C_2| / |C_1 \cup C_2|$



Motivation for MinHash/LSH

- **Suppose we need to find near-duplicate documents among $N = 1$ million documents**
- Naïvely, we would have to compute **pairwise** Jaccard similarities for **every pair of docs**
  - $N(N-1)/2 \approx 5*10^{11}$ comparisons
  - At 10^5 secs/day and 10^6 comparisons/sec, it would take **5 days**
- For $N = 10$ million, it takes more than a year...

Bit Vector Encoding



- Many similarity problems can be formalized as **finding subsets that have significant intersection**
- Encode sets using 0/1 (bit, boolean) vectors
  - One dimension per element in the universal set
- Interpret set intersection as bitwise **AND**, and set union as bitwise **OR**
- Example: C_1 = 10111; C_2 = 10011
  - Size of intersection = **3**; size of union = **4**,
  - **Jaccard similarity** (not distance) = **3/4**
  - **Distance: $d(C_1,C_2)$ = 1 – (Jaccard similarity) = 1/4**

From Sets to Boolean Matrices

- **Rows** = elements (shingles)
- **Columns** = sets (documents)
  - 1 in row ***e*** and column ***s*** if and only if ***e*** is a member of ***s***
  - Column similarity is the Jaccard similarity of the corresponding sets (rows with value *1)*
  - Typical matrix is sparse!
- **Each document is a column:**
  - Example: **$sim(C_1, C_2)$ = ?**
    - Size of intersection = 3; size of union = 6, Jaccard similarity (not distance) = 3/6
    - **$d(C_1,C_2)$ = 1 – (Jaccard similarity) = 3/6**

Shingles \ Documents				
	1	1	1	0
	1	1	0	1
	0	1	0	1
	0	0	0	1
	1	0	0	1
	1	1	1	0
	1	0	1	0

Outline: Finding Similar Columns

- **So far:**
  - Documents → Sets of shingles
  - Represent sets as boolean vectors in a matrix
- **Next goal: Find similar columns while computing small signatures**
  - **Similarity of columns == similarity of signatures**

Outline: Finding Similar Columns

- Naïve approach:
  - **1) Signatures of columns:** small summaries of columns
  - **2) Examine pairs of signatures** to find similar columns
    - **Essential:** Similarities of signatures and columns are related
  - **3) Optional:** Check that columns with similar signatures are really similar
- Warnings:
  - Comparing all pairs may take too much time: **Job for LSH**
    - These methods can produce false negatives, and even false positives (if the optional check is not made)

Example

Permutation π | Input matrix (Shingles x Documents) | Signature matrix M

Permutation π			Input matrix				Signature matrix M			
2	4	3	1	0	1	0	2	1	2	1
3	2	4	1	0	0	1	2	1	4	1
7	1	7	0	1	0	1	1	2	1	2
6	3	2	0	1	0	1				
1	6	6	0	1	0	1				
5	7	1	1	0	1	0				
4	5	5	1	0	1	0				

Similarities:

	1-3	2-4	1-2	3-4
Col/Col	0.75	0.75	0	0
Sig/Sig	0.67	1.00	0	0

MinHash Signature

- **Pick K=100 random permutations of the rows**
- Think of ***sig*(C)** as a column vector
- ***sig*(C)[i]** = according to the *i*-th permutation, the index of the first row that has a 1 in column *C*

$$sig(C)[i] = \min(\pi_i(C))$$

- **Note:** The sketch (signature) of document *C* is small **~100 bytes!**
- **We achieved our goal! We "compressed" long bit vectors into short signatures**

Implementation Tricks

- Permuting rows even once is prohibitive
- Row hashing!
  - Pick **K = 100** hash functions k_i
  - Ordering under k_i gives a random row permutation!
- **One-pass implementation**
  - For each column ***C*** and hash-func. k_i keep a "slot" for the min-hash value
  - Initialize all ***sig(C)[i]*** = ∞
  - **Scan rows looking for 1s**
    - Suppose row *j* has 1 in column ***C***
    - Then for each k_i:
      - If $k_i(j) < sig(C)[i]$, then $sig(C)[i] \leftarrow k_i(j)$

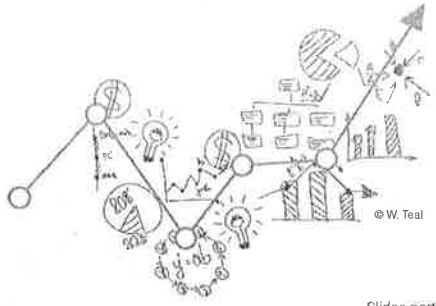
How to pick a random hash function h(x)?
Universal hashing:
$h_{a,b}(x)=((a\cdot x+b) \bmod p) \bmod N$
where:
a,b ... random integers
p ... prime number (p > N)

For Next Lecture

- Assigned readings:
  - Ch.3, Appendix C of TSK

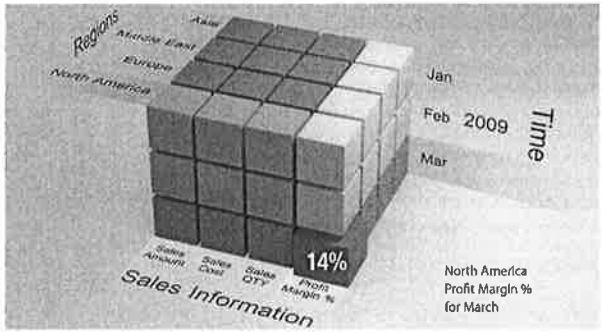
CSE 347/447: DATA MINING

Lecture 6: Classification - Decision Tree



Slides partially adapted from TSK

OLAP



Definition

联机分析处理.

- On-Line Analytical Processing (OLAP) was proposed by E. F. Codd, the father of the relational database.
- Relational databases put data into tables, while OLAP uses a multidimensional array representation.
  - Such representations of data previously existed in statistics and other fields
- There are a number of data analysis and data exploration operations that are easier with such a data representation.

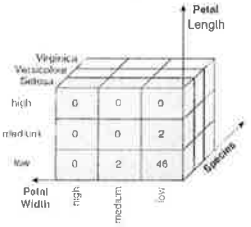
Example

- We show how the attributes, petal length, petal width, and species type can be converted to a multidimensional array
  - First, we discretized the petal width and length to have categorical values: *low*, *medium*, and *high*
  - We get the following table - note the count attribute

Petal Length	Petal Width	Species Type	Count
low	low	Setosa	46
low	medium	Setosa	2
medium	low	Setosa	2
medium	medium	Versicolour	43
medium	high	Versicolour	3
medium	high	Virginica	3
high	medium	Versicolour	2
high	medium	Virginica	3
high	high	Versicolour	2
high	high	Virginica	44

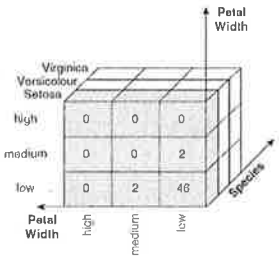
Example

- Each unique tuple of petal width, petal length, and species type identifies one element of the array.
- This element is assigned the corresponding count value.
- The figure illustrates the result.
- All non-specified tuples are 0.



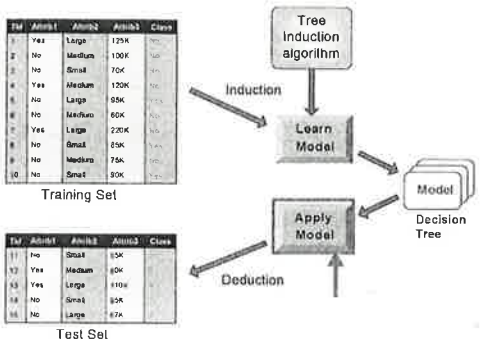
Operation: Slicing and Dicing

- Slicing is selecting a group of cells from the entire multidimensional array by specifying a specific value for one or more dimensions.
- Dicing involves selecting a subset of cells by specifying a range of attribute values.
  - This is equivalent to defining a subarray from the complete array.
- In practice, both operations can also be accompanied by aggregation over some dimensions.



1.

Decision Tree Classification



confusion matrix:

		Predicted Class	
		1	0
Actual Class	1	f_{11}	f_{10}
	0	f_{01}	f_{00}

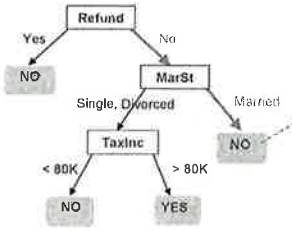
$$\text{Accuracy} = \frac{f_{11}+f_{00}}{f_{11}+f_{00}+f_{10}+f_{01}}$$

$$\text{Error rate} = \frac{f_{10}+f_{01}}{f_{11}+f_{00}+f_{10}+f_{01}}$$

Apply Model to Data

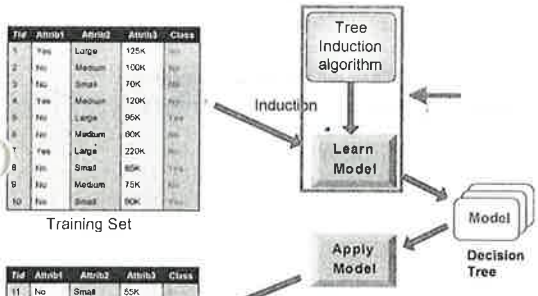
Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



Assign Cheat to "No"

Decision Tree Classification



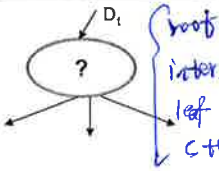
- Hunt's Algorithm (one of the earliest)
- CART
- ID3, C4.5
- SLIQ,SPRINT

Structure of Hunt's Algorithm

- Let D_t be the set of training records that reach a node t
- General Procedure:
  - If D_t contains records that belong the same class y_t, then t is a leaf node labeled as y_t
  - If D_t is an empty set, then t is a leaf node labeled by the default class, y_d
  - If D_t contains records that belong to more than one class, use an attribute test to split the data into smaller subsets. Recursively apply the procedure to each subset.

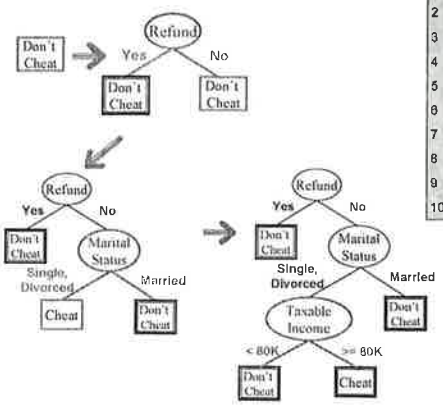
递归 调用

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



root node
internal node
leaf node
(terminal node)

Example



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

Tree Induction

- Greedy strategy. 贪心策略 to get suboptimal.
  - Split the records based on an attribute test that optimizes certain criterion.
- Issues
  - Determine how to split the records 属性测试条件.
    - How to specify the attribute test condition?
    - How to determine the best split?
  - Determine when to stop splitting

Optimal Splitting

- Greedy approach:
  - Nodes with homogeneous class distribution are preferred
- Need a measure of node impurity:

C0: 5 C1: 5	C0: 9 C1: 1
Non-homogeneous, High degree of impurity	Homogeneous, Low degree of impurity

- Gini Index
- Entropy
- Misclassification error

$$\text{Gain } \Delta = I(\text{parent}) - \sum_{j=1}^{k} \frac{N(v_j)}{N} I(v_j)$$

NOTE: $I(\cdot)$: the impurity of node.

N : # of total records at parent node

k : # of attribute values.

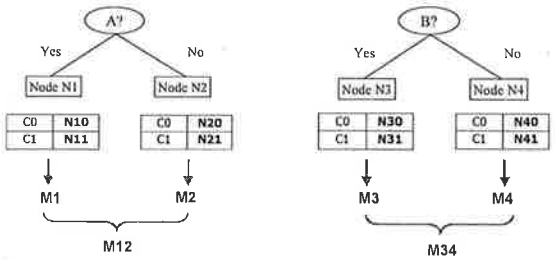
v_j : the child node.

Optimal Splitting

Before Splitting:

C0	N00
C1	N01

→ M0



Gain = M0 – M12 vs M0 – M34

if $I(\cdot)$ is Entropy,

Δ_{info} : information gain.

GINI

- Gini Index for a given node t :

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

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

- 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	

P(C1) = 1/6 P(C2) = 5/6

Gini = 1 – $(1/6)^2$ – $(5/6)^2$ = 0.278

Splitting based on GINI

- Used in CART, SLIQ, SPRINT.
- When a node p is split into k partitions (children), the quality of split is computed as,

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

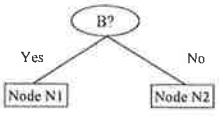
where, n_i = number of records at child i,

n = number of records at node p.

Smaller is better.

GINI: Binary Attribute

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



	Parent
C1	6
C2	6
Gini = 0.500	

	N1	N2
C1	5	1
C2	2	4
Gini=0.333		

Gini(N1)
=1 - $(5/7)^2$ - $(2/7)^2$
= 0.408

Gini(N2)
=1 - $(1/5)^2$ - $(4/5)^2$
= 0.320

Gini(Children)
=7/12*0.408 +5/12*0.320
=0.371

GINI: Categorical Attribute

- 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)

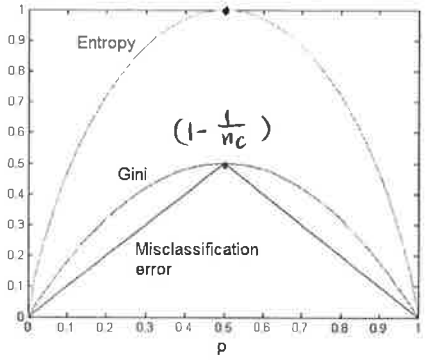
	CarType	
	{Sports, Luxury}	{Family}
C1	3	1
C2	2	4
Gini	0.400	

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

Comparison

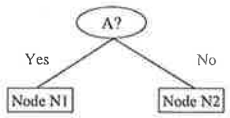
For a 2-class problem:

$(\log_2 n_c)$



$(1 - \frac{1}{n_c})$

Misclassification Error vs GINI



	Parent
C1	7
C2	3
Gini = 0.42	

Gini(N1)
= 1 – (3/3)² – (0/3)²
= 0

Gini(N2)
= 1 – (4/7)² – (3/7)²
= 0.489

	N1	N2
C1	3	4
C2	0	3
Gini=0.361		

Gini(Children)
= 3/10 * 0
+ 7/10 * 0.489
= 0.342

Gini improves !!

$gain(E) = \frac{3}{10} - (\frac{3}{10} \times 0 + \frac{7}{10} \times \frac{3}{7}) = 0$

$gain(Gini) = 0.42 - 0.342 > 0$

Tree Induction

- Greedy strategy.
  - Split the records based on an attribute test that optimizes certain criterion.
- Issues
  - Determine how to split the records
    - How to specify the attribute test condition?
    - How to determine the best split?
  - Determine when to stop splitting

Stopping Criteria

- Stop expanding a node when all the records belong to the same class
- Stop expanding a node when all the records have similar attribute values
- Early termination (4.4.5)

Advantages of Decision Tree

- Inexpensive to construct
- Extremely fast at classifying unknown records
- Easy to interpret for small-sized trees
- Accuracy is comparable to other classification techniques for many simple data sets

For Next Lecture

- Assigned reading
  - Ch.4.4 - 4.6 of TSK

CSE 347/447: DATA MINING

Lecture 7: Model Selection and Evaluation

Slides partially adapted from TSK

Announcement

- HW1 due tonight
- HW2 will be posted later today
- All questions regarding HW should be directed to Piazza

Review:

GINI: Continuous Attribute

- 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

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

GINI: Continous Attribute

- 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 →	65	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

Entropy

- Entropy at a given node t:

$$Entropy(t) = -\sum_j p(j \mid t) \log p(j \mid t)$$

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

– Measures homogeneity of a node.
  - Maximum ($\log n_c$) when records are equally distributed among all classes implying least information
  - Minimum (0.0) when all records belong to one class, implying most information

-- Entropy based computations are similar to the GINI index computations

C1	1
C2	5

P(C1) = 1/6 P(C2) = 5/6

Entropy = – (1/6) $\log_2$ (1/6) – (5/6) $\log_2$ (1/6) = 0.65

Splitting Based on Entropy

- Information Gain:

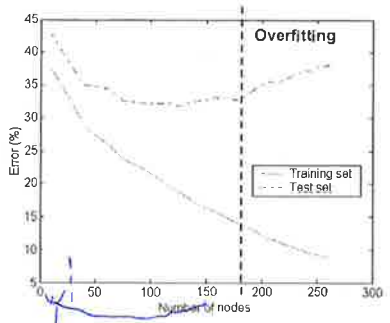
$$GAIN_{split} = Entropy(p) - \left(\sum_{i=1}^{k} \frac{n_i}{n} Entropy(i) \right)$$

Parent Node, p is split into k partitions;
n_i is number of records in partition i

– Measures Reduction in Entropy achieved because of the split. Choose the split that achieves most reduction (maximizes GAIN)
– Used in ID3 and C4.5
-- Disadvantage: Tends to prefer splits that result in large number of partitions, each being small but pure.

Under/Overfitting

Under/Overfitting

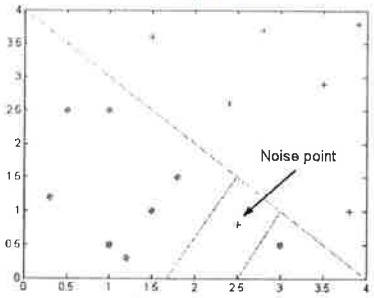


Underfitting: when model is too simple, both training and test errors are large

underfitting.

Overfitting due to Noise

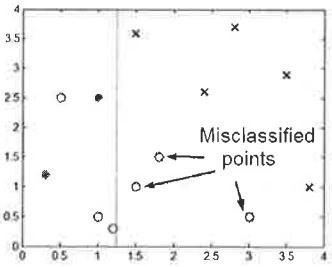
Reason 1.



Decision boundary is distorted by noise point

... due to Insufficient Data

Reason 2.



Lack of data points in the lower half of the diagram makes it difficult to predict correctly the class labels of that region

- Insufficient number of training records in the region causes the decision tree to predict the test examples using other training records that are irrelevant to the classification task

Notes on Overfitting

- Overfitting results in decision trees that are more complex than necessary
- Training error no longer provides a good estimate of how well the tree will perform on previously unseen records
- Need new ways for estimating errors

model 在未知记录上的期望误差

② plus: $e'(T) = \frac{\sum_{i=1}^{k} [e(t_i) + \Omega(t_i)]}{\sum_{i=1}^{k} n(t_i)}$

$= \frac{e(T) + \Omega(T)}{N_t}$ better $= e(T) + \frac{\Omega(T)}{N_t}$

Estimating Generalization Error

- Re-substitution errors: error on training (Σ e(t)) apparent error.
- Generalization errors: error on testing (Σ e'(t))
- Methods for estimating generalization errors:
  1. Optimistic approach: e'(t) = e(t)
  2. Pessimistic approach: add penalty term.
     - For each leaf node: e'(t) = (e(t)+0.5)
     - Total errors: e'(T) = e(T) + N × 0.5 (N: number of leaf nodes)
     - For a tree with 30 leaf nodes and 10 errors on training (out of 1000 instances):
       Training error = 10/1000 = 1%
       Generalization error = (10 + 30×0.5)/1000 = 2.5%
  3. Reduced error pruning (REP):
     - uses validation data set to estimate generalization error

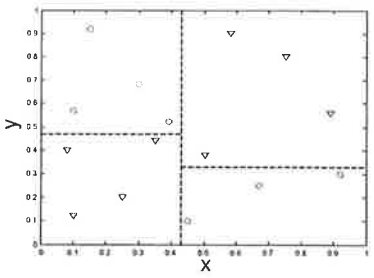
the 3rd set.

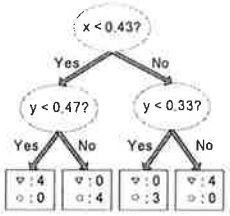
Expressiveness

disadvantages.

- Decision tree provides expressive representation for learning discrete-valued function
  - But they do not generalize well to certain types of Boolean functions
    - Example: parity function:
      - Class = 1 if there is an even number of Boolean attributes with truth value = True
      - Class = 0 if there is an odd number of Boolean attributes with truth value = True
    - For accurate modeling, must have a complete tree
- Not expressive enough for modeling continuous variables
  - Particularly when test condition involves only a single attribute at-a-time

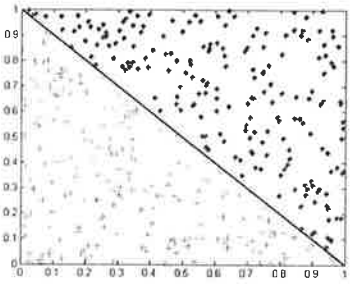
Decision Boundary

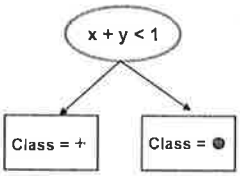




- Border line between two neighboring regions of different classes is known as decision boundary
- Decision boundary is parallel to axes because test condition involves a single attribute at-a-time

Oblique Decision Tree





- Test condition may involve multiple attributes
- More expressive representation
- Finding optimal test condition is computationally expensive

Model Evaluation

Model Evaluation

- Metrics for Performance Evaluation
  - How to evaluate the performance of a model?
- Methods for Performance Evaluation
  - How to obtain reliable estimates?
- Methods for Model Comparison
  - How to compare the relative performance among competing models?

Metrics

- Focus on the predictive capability of a model
  - Rather than how fast it takes to classify or build models, scalability, etc.
- Confusion Matrix:

		PREDICTED CLASS	
		Class=Yes	Class=No
ACTUAL CLASS	Class=Yes	a	b
	Class=No	c	d

a: TP (true positive)
b: FN (false negative)
c: FP (false positive)
d: TN (true negative)

For Next Lecture

- Assigned Reading: Ch.5.2-5.3 of TSK

FN: Type II error

FP: Type I error.

① Sensitivity (true positive rate, TPR).
= hit rate, recall.

$$TPR = \frac{TP}{P} = \frac{TP}{TP+FN}.$$

② Specificity (true negative rate, SPC).

$$SPC = \frac{TN}{N} = \frac{TN}{FP+TN}.$$

③ precision (positive predictive value, PPV)

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

④. negative predictive value (NPV)

$$NPV = \frac{TN}{TN+FN}.$$

⑤. fall-out (false positive rate, FPR).

$$FPR = \frac{FP}{N} = \frac{FP}{FP+TN} = 1-SPC$$

⑥. false discovery rate (FDR)

$$FDR = \frac{FP}{FP+TP} = 1-PPV.$$

⑦. miss rate (false negative rate, FNR).

$$FNR = \frac{FN}{P} = \frac{FN}{TP+FN} = 1-TPR.$$

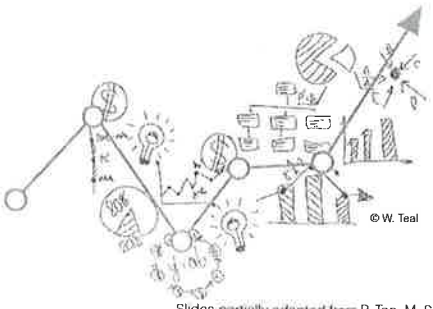
accuracy (Acc).

$$Acc = \frac{TP+TN}{P+N}.$$

F-measure (F) $= \frac{2TP}{2TP+FP+FN}$.

CSE 347/447: DATA MINING

Lecture 8: KNN and Bayesian Classifiers



Slides partially adapted from P. Tan, M. Steinbach, and V. Kumar

Review

Cost vs Accuracy

Count	PREDICTED CLASS		
		Class=Yes	Class=No
ACTUAL CLASS	Class=Yes	a	b
	Class=No	c	d

Cost	PREDICTED CLASS		
		Class=Yes	Class=No
ACTUAL CLASS	Class=Yes	p	q
	Class=No	q	p

Accuracy is proportional to cost if
1. C(Yes|No)=C(No|Yes) = q
2. C(Yes|Yes)=C(No|No) = p

N = a + b + c + d

Accuracy = (a + d)/N

Cost = p (a + d) + q (b + c)
= p (a + d) + q (N – a – d)
= q N – (q – p)(a + d)
= N [q – (q-p) × Accuracy]

Cost-Sensitive Measure

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

- Precision is biased towards C(Yes|Yes) & C(Yes|No)

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

- Recall is biased towards C(Yes|Yes) & C(No|Yes)

$$\text{F-measure (F)} = \frac{2rp}{r+p} = \frac{2a}{2a+b+c}$$

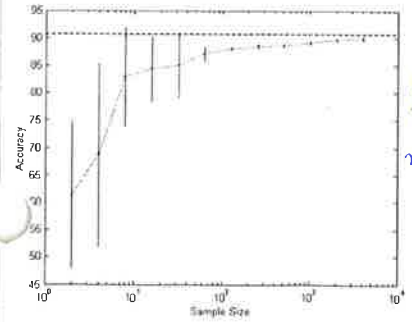
- F-measure is biased towards all except C(No|No)

Model Evaluation

- Metrics for Performance Evaluation
  - How to evaluate the performance of a model?
- Methods for Performance Evaluation
  - How to obtain reliable estimates?
- Methods for Model Comparison
  - How to compare the relative performance among competing models?

Learning Curve

- Performance of a model may depend on other factors besides the learning algorithm:
  - Class distribution
  - Cost of misclassification
  - Size of training and test sets



- Learning curve shows how accuracy changes with varying sample size
- Requires a sampling schedule for creating learning curve:
  - Arithmetic sampling (Langley, et al)
  - Geometric sampling (Provost et al)

Effect of small sample size:
- Bias in the estimate
- Variance of estimate

Methods of Estimation

- Holdout → random subsampling (iterate holdout)
  - Reserve 2/3 for training and 1/3 for testing $acc_{sub} = \sum_{i=1}^{k} acc_i / k$.
- Cross validation
  - Partition data into k disjoint subsets ← two-fold cross-validation
  - k-fold: train on k-1 partitions, test on the remaining one
  - Leave-one-out: k=n ← high var.
- Stratified sampling
  - oversampling vs undersampling

entire samples.

$$acc_{boot} = \frac{1}{b}\sum_{i=1}^{b}(0.632\,\varepsilon_i + 0.368\,acc_s)$$

bootstrap. 有放回 N次 取N.

概念: $1-(1-1/N)^N \approx 1-e^{-1} = 0.632$.

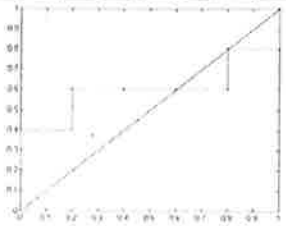
.632 bootstrap.

1

Constructing ROC

Class	+	-	+	-	-	-	+	-	+	+	
Threshold >=	0.25	0.43	0.53	0.76	0.85	0.95	0.95	0.87	0.93	0.96	1.00
TP	5	4	4	3	3	3	3	2	2	1	0
FP	5	5	4	4	3	2	1	1	0	0	0
TN	0	0	1	1	2	3	4	4	5	5	5
FN	0	1	1	2	2	2	2	3	3	4	5
TPR	1	0.8	0.8	0.6	0.6	0.6	0.6	0.4	0.4	0.2	0
FPR	1	1	0.8	0.8	0.6	0.4	0.2	0.2	0	0	0

ROC Curve:



Test of Significance

- Given two models:
  - Model M1: accuracy = 85%, tested on 30 instances
  - Model M2: accuracy = 75%, tested on 5000 instances
- Can we say M1 is better than M2?
  - How much confidence can we place on accuracy of M1 and M2?
  - Can the difference in performance measure be explained as a result of random fluctuations in the test set?

Confidence Interval

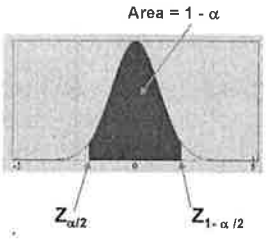
- Prediction can be regarded as a Bernoulli trial
  - A Bernoulli trial has 2 possible outcomes
  - Possible outcomes for prediction: correct or wrong
  - Collection of Bernoulli trials has a Binomial distribution:
    - x ~ Bin(N, p) x: number of correct predictions
    - e.g: Toss a fair coin 50 times, how many heads would turn up?
      Expected number of heads = N×p = 50 × 0.5 = 25
- Given x (# of correct predictions) or equivalently, acc=x/N, and N (# of test instances),

Can we predict p (true accuracy of model)?

Confidence Interval

- For large test sets (N > 30),
  - acc has a normal distribution with mean p and variance p(1-p)/N

$$P(Z_{\alpha/2} < \frac{acc - p}{\sqrt{p(1-p)/N}} < Z_{1-\alpha/2}) = 1-\alpha$$



- Confidence Interval for p:

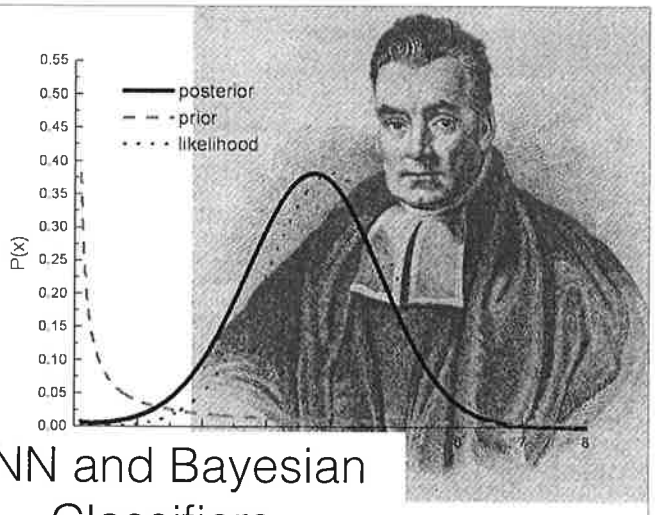
$$p = \frac{2 \times N \times acc + Z^2_{\alpha/2} \pm \sqrt{Z^2_{\alpha/2} + 4 \times N \times acc - 4 \times N \times acc^2}}{2(N + Z^2_{\alpha/2})}$$

Confidence Interval

- Consider a model that produces an accuracy of 80% when evaluated on 100 test instances:
  - N=100, acc = 0.8
  - Let 1-α = 0.95 (95% confidence)
  - From probability table, $Z_{\alpha/2}$=1.96

1-α	Z
0.99	2.58
0.98	2.33
0.95	1.96
0.90	1.65

N	50	100	500	1000	5000
p(lower)	0.670	0.711	0.763	0.774	0.789
p(upper)	0.888	0.866	0.833	0.824	0.811



KNN and Bayesian Classifiers

Nearest Neighbor Classification

- Compute distance between two points:
  - Euclidean distance

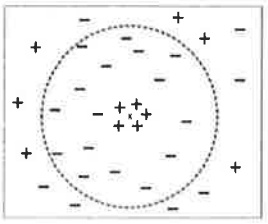
$$d(p,q) = \sqrt{\sum_i (p_i - q_i)^2}$$

- Determine the class from nearest neighbor list
  - take the majority vote of class labels among the k-nearest neighbors $y' = \arg\max_v \sum_{(x_i, y_i) \in D_z} I(v = y_i)$
  - Weigh the vote according to distance
    - weight factor, $w = 1/d^2$

$$y' = \arg\max_v \sum_{(x_i, y_i) \in D_z} w_i \times I(v = y_i)$$

Nearest Neighbor Classification

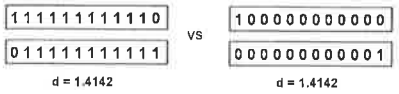
- Choosing the value of k:
  - If k is too small, sensitive to noise points
  - If k is too large, neighborhood may include points from other classes



Nearest Neighbor Classification

- Scaling issues 缩放比例.
  - Attributes may have to be scaled to prevent distance measures from being dominated by one of the attributes
  - Example:
    - height of a person may vary from 1.5m to 1.8m
    - weight of a person may vary from 90lb to 300lb
- Problem with Euclidean measure:
  - High dimensional data
    - curse of dimensionality
  - Can produce counter-intuitive results

Normalization!



Nearest Neighbor Classification

- k-NN classifiers are lazy learners
  - It does not build models explicitly
  - Unlike eager learners such as decision tree induction and rule-based systems
  - Classifying unknown records are relatively expensive

2.

Bayesian Classifier

- A probabilistic framework for solving classification problems
- Conditional Probability:

$$P(C \mid A) = \frac{P(A,C)}{P(A)}$$

$$P(A \mid C) = \frac{P(A,C)}{P(C)}$$

- Bayes theorem:

$$P(C \mid A) = \frac{P(A \mid C)P(C)}{P(A)}$$

Example of Bayes Theorem

posterior probability
prior probability.

- Given:
  - A doctor knows that meningitis causes stiff neck 50% of the time
  - Prior probability of any patient having meningitis is 1/50,000
  - Prior probability of any patient having stiff neck is 1/20
- If a patient has stiff neck, what's the probability he/she has meningitis?

$$P(M \mid S) = \frac{P(S \mid M)P(M)}{P(S)} = \frac{0.5 \times 1/50000}{1/20} = 0.0002$$

Example

Given a Test Record:

$$X = (\text{Refund} = \text{No}, \text{Married}, \text{Income} = 120\text{K})$$

naive Bayes Classifier:

P(Refund=Yes|No) = 3/7
P(Refund=No|No) = 4/7
P(Refund=Yes|Yes) = 0
P(Refund=No|Yes) = 1
P(Marital Status=Single|No) = 2/7
P(Marital Status=Divorced|No)=1/7
P(Marital Status=Married|No) = 4/7
P(Marital Status=Single|Yes) = 2/7
P(Marital Status=Divorced|Yes)=1/7
P(Marital Status=Married|Yes) = 0

For taxable income:
If class=No: sample mean=110
sample variance=2975
If class=Yes: sample mean=90
sample variance=25

- P(X|Class=No) = P(Refund=No|Class=No) × P(Married| Class=No) × P(Income=120K| Class=No) = 4/7 × 4/7 × 0.0072 = 0.0024
- P(X|Class=Yes) = P(Refund=No| Class=Yes) × P(Married| Class=Yes) × P(Income=120K| Class=Yes) = 1 × 0 × 1.2 × 10^{-9} = 0

Since P(X|No)P(No) > P(X|Yes)P(Yes)
Therefore P(No|X) > P(Yes|X)
=> Class = No

Naive Bayes Classifier

Problem:
- If one of the conditional probability is zero, then the entire expression becomes zero

Solve:
- Probability estimation:

$$\text{Original}: P(A_i \mid C) = \frac{N_{ic}}{N}$$

$$\text{Laplace}: P(A_i \mid C) = \frac{N_{ic}+1}{N_c + c}$$

$$\text{m- estimate}: P(A_i \mid C) = \frac{N_{ic} + mp}{N_c + m}$$

c: number of classes
p: prior probability
m: parameter

because : $n = 0 \rightarrow P(A_i|C) = p$

Example

Name	Give Birth	Can Fly	Live in Water	Have Legs	Class
human	yes	no	no	yes	mammals
python	no	no	no	no	non-mammals
salmon	no	no	yes	no	non-mammals
whale	yes	no	yes	no	mammals
frog	no	no	sometimes	yes	non-mammals
komodo	no	no	no	yes	non-mammals
bat	yes	yes	no	yes	mammals
pigeon	no	yes	no	yes	non-mammals
cat	yes	no	no	yes	mammals
leopard shark	yes	no	yes	no	non-mammals
turtle	no	no	sometimes	yes	non-mammals
penguin	no	no	sometimes	yes	non-mammals
porcupine	yes	no	no	yes	mammals
eel	no	no	yes	no	non-mammals
salamander	no	no	sometimes	yes	non-mammals
gila monster	no	no	no	yes	non-mammals
platypus	no	no	no	yes	mammals
owl	no	yes	no	yes	non-mammals
dolphin	yes	no	yes	no	mammals
eagle	no	yes	no	yes	non-mammals

A: attributes
M: mammals
N: non-mammals

$$P(A \mid M) = \frac{6}{7} \times \frac{6}{7} \times \frac{2}{7} \times \frac{2}{7} = 0.06$$

$$P(A \mid N) = \frac{1}{13} \times \frac{10}{13} \times \frac{3}{13} \times \frac{4}{13} = 0.0042$$

$$P(A \mid M)P(M) = 0.06 \times \frac{7}{20} = 0.021$$

$$P(A \mid N)P(N) = 0.004 \times \frac{13}{20} = 0.0027$$

Give Birth	Can Fly	Live in Water	Have Legs	Class
yes	no	yes	no	?

P(A|M)P(M) > P(A|N)P(N)
=> Mammals

Summary

- Robust to isolated noise points
- Handle missing values by ignoring the instance during probability estimate calculations
- Robust to irrelevant attributes
- Independence assumption may not hold for some attributes
  - Use other techniques such as Bayesian Belief Networks (BBN)

For Next Lecture

- Assigned Reading
  - Ch.5.5 of TSK

CSE 347/447: DATA MINING

Lecture 10: Ensemble Methods

Slides partially adapted from P. Tan, M. Steinbach, and V. Kumar and Xiaojin Zhu

$$\mathcal{L}(w,b,\lambda) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{m}\lambda_i\left[y^{(i)}(w^T x^{(i)}+b)-1\right]$$

$$= \frac{1}{2}w^T w - \sum_{i=1}^{m}\lambda_i y^{(i)} w^T x^{(i)} - \sum_{i=1}^{m}\lambda_i y^{(i)} b + \sum_{i=1}^{m}\lambda_i$$

$$= \frac{1}{2}w^T \sum_{i=1}^{m}\lambda_i y^{(i)} x^{(i)} - w^T \sum_{i=1}^{m}\lambda_i y^{(i)} x^{(i)} - \sum_{i=1}^{m}\lambda_i y^{(i)} b + \sum_{i=1}^{m}\lambda_i$$

$$= -\frac{1}{2}w^T \sum_{i=1}^{m}\lambda_i y^{(i)} x^{(i)} - b\sum_{i=1}^{m}\lambda_i y^{(i)} + \sum_{i=1}^{m}\lambda_i$$

SVM

$$= -\frac{1}{2}\left(\sum_{i=1}^{m}\lambda_i y^{(i)} x^{(i)}\right)^T\left(\sum_{i=1}^{m}\lambda_i y^{(i)} x^{(i)}\right) - b\sum_{i=1}^{m}\lambda_i y^{(i)} + \sum_{i=1}^{m}\lambda_i$$

$$= -\frac{1}{2}\sum_{i=1}^{m}\lambda_i y^{(i)} (x^{(i)})^T \sum_{j=1}^{m}\lambda_j y^{(j)} x^{(j)} - b\sum_{i=1}^{m}\lambda_i y^{(i)} + \sum_{i=1}^{m}\lambda_i$$

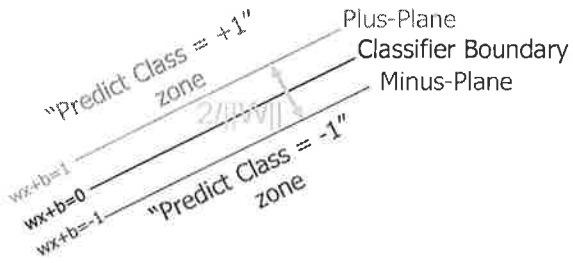
$$= -\frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\lambda_i y^{(i)} (x^{(i)})^T \lambda_j y^{(j)} x^{(j)} - b\sum_{i=1}^{m}\lambda_i y^{(i)} + \sum_{i=1}^{m}\lambda_i$$

$$= \sum_{i=1}^{m}\lambda_i - \frac{1}{2}\sum_{i=1,j=1}^{m}\lambda_i\lambda_j y^{(i)} y^{(j)} (x^{(i)})^T x^{(j)}.$$

only $x^{(i)}$'s are vector.

Linearly Separable Case

$\min_{W,b}$ ½ W'W
Subject to Yi (W'Xi + b) >= 1, for all i



Biting the bullet*

- Here's the original QP formula
  $\min_{W,b}$ ½ W'W
  Subject to Yi (W'Xi + b) >= 1, for all i (N constraints)
- Remember Lagrange multipliers?
  L = ½ W'W – $\sum$ai [Yi (W'Xi + b) – 1]
  Subject to ai >= 0, for all i

  We have them here because those are inequality constraints
- We want the gradient of L to vanish with respect to W, b, a. Try this and you'll get

  $W = \sum a_i Y_i X_i$

  $\sum a_i Y_i = 0$

  Put them back into the Lagrangian L

$$\mathcal{L}_P = \frac{1}{2}w^T w - \sum\lambda_i\left[y_i(w^T x_i + b) - 1\right]$$

$$\frac{\partial \mathcal{L}_P}{\partial w} = 0 \Rightarrow w = \sum_{i=1}^{N}\lambda_i y_i^{(i)} x_i^{(i)}$$

$$\frac{\partial \mathcal{L}_P}{\partial b} = 0 \Rightarrow \sum_{i=1}^{N}\lambda_i y_i^{(i)} = 0.$$

Biting the bullet*

$$\mathcal{L}_D = \sum_{i=1}^{N}\lambda_i - \frac{1}{2}\sum_{i,j}\lambda_i\lambda_j y_i y_j x_i x_j.$$

$\max_{\{ai\}} \sum ai - ½ \sum_{i,j} ai\ aj\ Yi\ Yj\ \mathbf{Xi'}\ \mathbf{Xj}$

Subject to

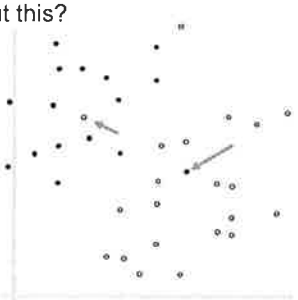
ai >= 0, for all I

$\sum ai\ Yi = 0$

- This is an equivalent QP problem (the dual)
- Before we optimize W (d variables), now we optimize a (N variables): which is better?
- X only appears in the inner product

Non-Separable Case

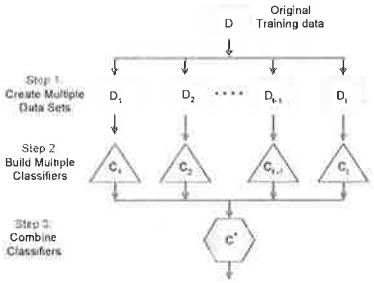
- What about this?



Can we insist on Yi (W'Xi + b) >= 1, for all i?

General Idea

- Construct a set of classifiers from the training data
- Predict class label of previously unseen records by aggregating predictions made by multiple classifiers



Why Does It Work?

- Suppose there are 25 base classifiers
  - Each classifier has error rate, $\varepsilon = 0.35$
  - Assume classifiers are independent
  - Probability that the ensemble classifier makes a wrong prediction:

$$\sum_{i=13}^{25} \binom{25}{i} \varepsilon^i (1-\varepsilon)^{25-i} = 0.06$$

Example of Ensemble Methods

- How to generate an ensemble of classifiers?
  - Bagging 装袋
  - Boosting 提升

Bagging

- Sampling with replacement

Original Data	1	2	3	4	5	6	7	8	9	10
Bagging (Round 1)	7	8	10	8	2	5	10	10	5	9
Bagging (Round 2)	1	4	9	1	2	3	2	7	3	2
Bagging (Round 3)	1	8	5	10	5	5	9	6	3	7

自助取样 - 均匀、重复、有放回.

- Build classifier on each bootstrap sample
- Each sample has probability $1-(1-\frac{1}{n})^n$ of being selected

Random - Forest

Boosting

- An iterative procedure to adaptively change distribution of training data by focusing more on previously misclassified records

AdaBoost.

闭权重.

  - Initially, all N records are assigned equal weights
  - Records that are wrongly classified will have their weights increased
  - Records that are classified correctly will have their weights decreased

Original Data	1	2	3	4	5	6	7	8	9	10
Boosting (Round 1)	7	3	2	8	7	9	4	10	6	3
Boosting (Round 2)	5	4	9	4	2	5	1	7	4	2
Boosting (Round 3)	(4)	(4)	8	10	(4)	5	(4)	6	3	(4)

- Example 4 is hard to classify
- Its weight is increased, therefore it is more likely to be chosen again in subsequent rounds

CSE 347/447: DATA MINING

Lecture 12: Clustering I

Slides partially adapted from P. Tan, M. Steinbach, and V. Kumar

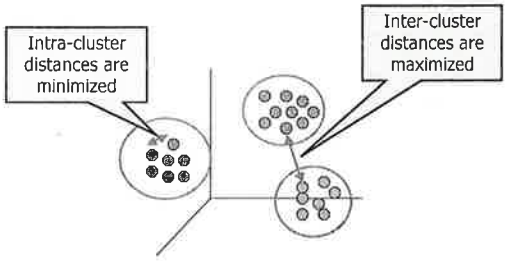
Announcements

- Homework: more frequent, less heavier
- Project on the way!
- Instructor office hours: 3-5 PM Wed (start from next week)
- No in-class break

K-means Clustering

Definition

Finding groups of objects such that the objects in a group will be similar (or related) to one another and different from (or unrelated to) the objects in other groups



Applications

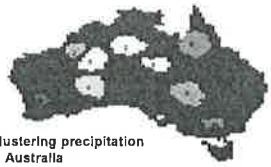
Understanding

- Group related documents for browsing, group genes and proteins that have similar functionality, or group stocks with similar price fluctuations

	Discovered Clusters	Industry Group
1	Applied-Mat-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,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

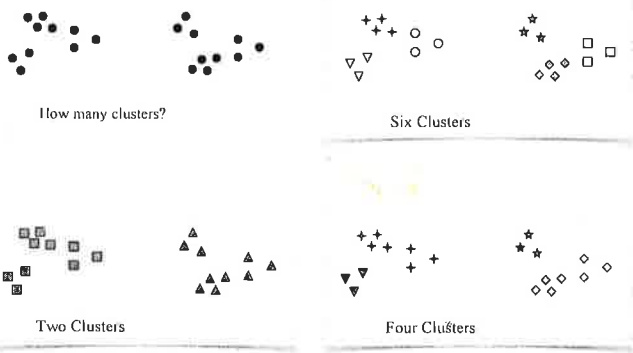
Summarization

- Reduce the size of large data sets

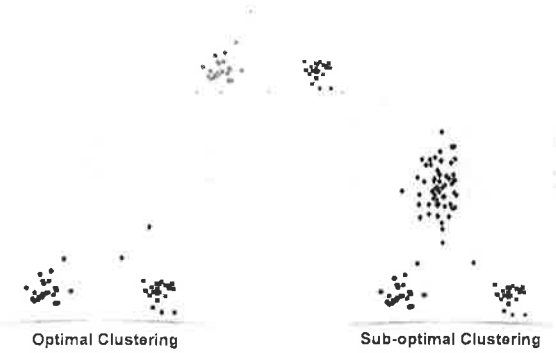


Clustering precipitation in Australia

Notion of a Cluster can be Ambiguous



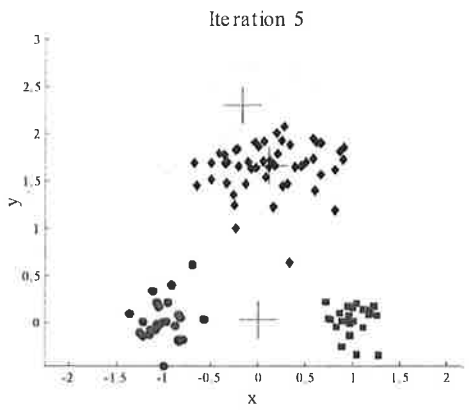
Two Different Clusterings



Optimal Clustering

Sub-optimal Clustering

Importance of Initial Centroids



Problem with Selecting Initial Centroids

If there are K 'real' clusters then the chance of selecting one centroid from each cluster is small.

- Chance is relatively small when K is large
- If clusters are the same size, n, then

$$P = \frac{\text{number of ways to select one centroid from each cluster}}{\text{number of ways to select } K \text{ centroids}} = \frac{K!n^K}{(Kn)^K} = \frac{K!}{K^K}$$

- For example, if K = 10, then probability = $10!/10^{10} = 0.00036$
- Sometimes the initial centroids will readjust themselves in 'right' way, and sometimes they don't

Solutions to Initial Centroids Problem

because of cost

useful in small sample and small K.

- Multiple runs
  - Helps, but probability is not on your side
- Sample and use hierarchical clustering to determine initial centroids
- Select more than k initial centroids and then select among these initial centroids
  - Select most widely separated
- Bisecting K-means
  - Not as susceptible to initialization issues

Post-processing

To make total SSE smaller.

- Eliminate small clusters that may represent outliers
- ① Split 'loose' clusters, i.e., clusters with relatively high SSE
- ② Merge clusters that are 'close' and that have relatively low SSE
- Can use these steps during the clustering process

③. disperse.

④. Introduce a new centroid. Usually the forest point from centroids.

①④ K↑; ②③ K↓

Bisecting K-means

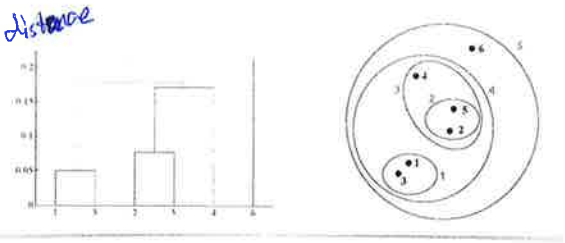
Bisecting K-means algorithm

- Variant of K-means that can produce a partitional or a hierarchical clustering

Initialize the list of clusters to contain the cluster containing all points.
repeat
  Select a cluster from the list of clusters
  for $i = 1$ to *number_of_iterations* do
    Bisect the selected cluster using basic K-means
  end for
  Add the two clusters from the bisection with the lowest SSE to the list of clusters.
until Until the list of clusters contains K clusters

Definition

- Produces a set of nested clusters organized as a hierarchical tree
- Can be visualized as a dendrogram 樹狀圖.
  - A tree like diagram that records the sequences of merges or splits



Strengths

- Do not have to assume any particular number of clusters
  - Any desired number of clusters can be obtained by 'cutting' the dendogram at the proper level
- They may correspond to meaningful taxonomies
  - Example in biological sciences (e.g., animal kingdom, phylogeny reconstruction, ...)

Algorithms

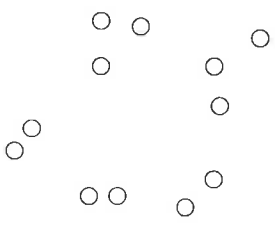
- Two main types of hierarchical clustering
  - Agglomerative: 凝聚
    - Start with the points as individual clusters
    - At each step, merge the closest pair of clusters until only one cluster (or k clusters) left
  - Divisive: 分裂
    - Start with one, all-inclusive cluster
    - At each step, split a cluster until each cluster contains a point (or there are k clusters)
- Traditional hierarchical algorithms use a similarity or distance matrix
  - Merge or split one cluster at a time

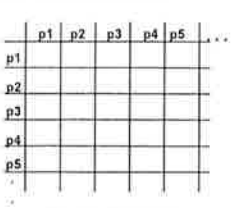
Agglomerative Clustering

- More popular hierarchical clustering technique
- Basic algorithm is straightforward
  1. Compute the proximity matrix
  2. Let each data point be a cluster
  3. **Repeat**
  4. Merge the two closest clusters
  5. Update the proximity matrix
  6. **Until** only a single cluster remains
- Key operation is the computation of the proximity of two clusters
  - Different approaches to defining the distance between clusters distinguish the different algorithms

Starting Situation

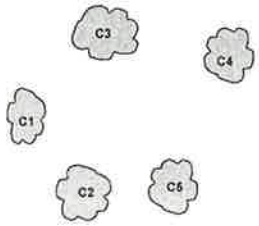
- Start with clusters of individual points and a proximity matrix

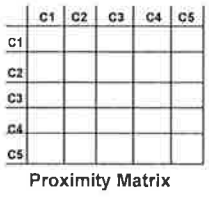


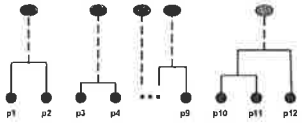


Intermediate Situation

- After some merging steps, we have some clusters

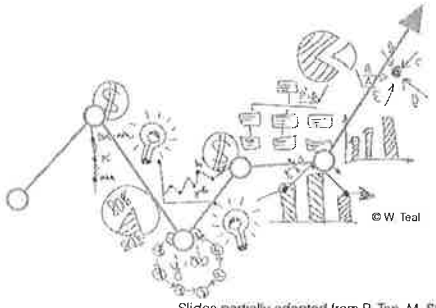






CSE 347/447: DATA MINING

Lecture 13: Clustering II



Slides partially adapted from P. Tan, M. Steinbach, and V. Kumar

Announcements

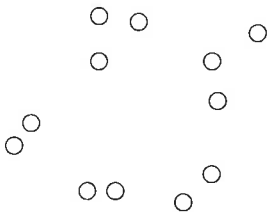
- The first project launched
- Homework 3 out today
- Instructor office hours: 3-5 PM Wed (start from this week)

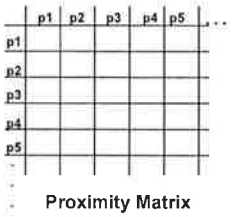
Hierarchical Clustering

Review

Starting Situation

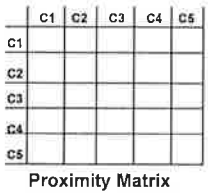
- Start with clusters of individual points and a proximity matrix

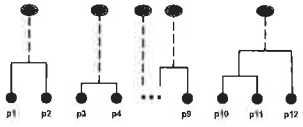




Intermediate Situation

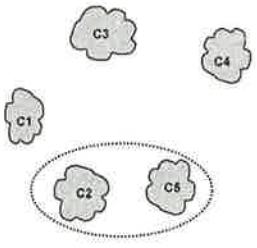
- After some merging steps, we have some clusters

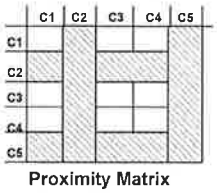


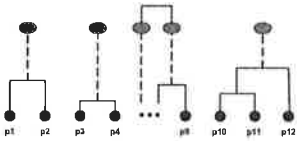


Intermediate Situation

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





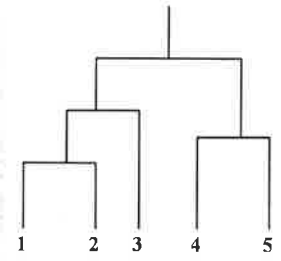


Group Average

Proximity of two clusters is the average of pairwise proximity between points in the two clusters.

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

	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



$$\text{proximity}(C_i, C_j) = \frac{\sum_{x \in C_i,\ y \in C_j} \text{proximity}(x, y)}{m_i \cdot m_j}$$

Strengths and Limitations

Compromise between Single and Complete Link

Strengths
- ✓ Less susceptible to noise and outliers

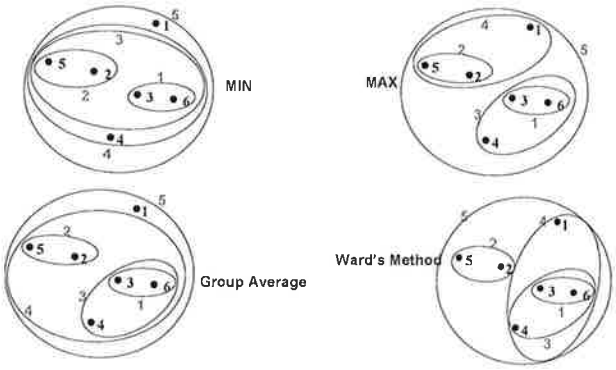
Limitations
- ✗ Biased towards globular clusters

Ward's Method

min(ΔSSE)

- Similarity of two clusters is based on the increase in squared error when two clusters are merged
  - Similar to group average if distance between points is distance squared
- ✓ Less susceptible to noise and outliers
- ✗ Biased towards globular clusters
- ✓ Hierarchical analogue of K-means
  - Can be used to initialize K-means

Hierarchical Clustering: Comparison



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
  - Complexity can be reduced to $O(N^2 \log(N))$ time for some approaches

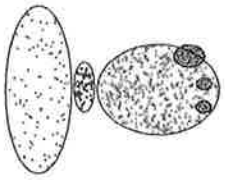
★ Lance-Williams 公式 of proximity

Hierarchical Clustering: Limitations

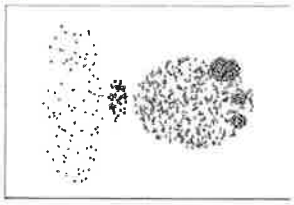
- Once a decision is made to combine two clusters it cannot be undone
- No objective function is directly minimized
- Different schemes have problems with one or more of the following:
  - Sensitivity to noise and outliers
  - Difficulty handling different sized clusters and convex shapes
  - Breaking large clusters → during ③ ④

2 solution { weighted by # of points in clusters
unweighted

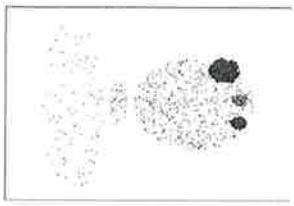
When It Does Not



Original Points



(MinPts=4, Eps=9.75).

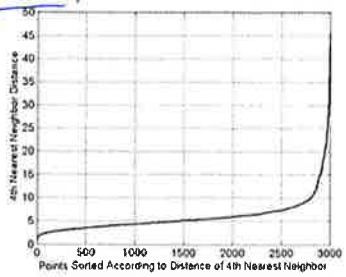


(MinPts=4, Eps=9.92)

- Varying densities
- High-dimensional data

Determining EPS and MinPts

- Idea is that for points in a cluster, their k^{th} nearest neighbors are at roughly the same distance
- Noise points have the k^{th} nearest neighbor at farther distance
- So, plot sorted distance of every point to its k^{th} nearest neighbor

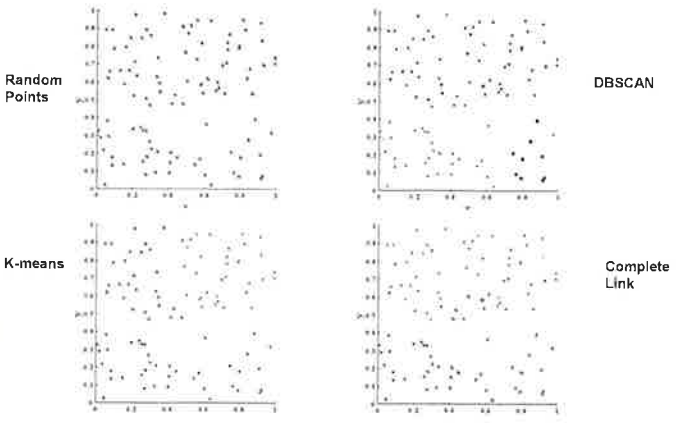


Cluster Evaluation

Cluster Validity

- For supervised classification we have a variety of measures to evaluate how good our model is
  - Accuracy, precision, recall
- For cluster analysis, the analogous question is how to evaluate the "goodness" of the resulting clusters?
- But "clusters are in the eye of the beholder"!
- Then why do we want to evaluate them?
  - To avoid finding patterns in noise
  - To compare clustering algorithms
  - To compare two sets of clusters
  - To compare two clusters

Clustering of Random Data



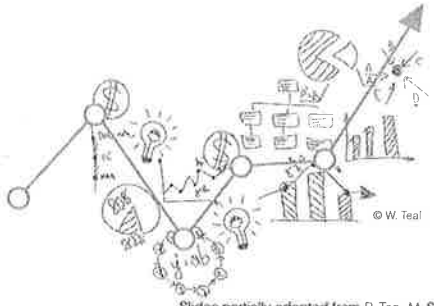
Different Aspects of Validity

聚类趋势

1. Determining the clustering tendency of a set of data, i.e., distinguishing whether non-random structure actually exists in the data.
2. Comparing the results of a cluster analysis to externally known results, e.g., to externally given class labels.
3. Evaluating how well the results of a cluster analysis fit the data *without* reference to external information.
   - Use only the data
4. Comparing the results of two different sets of cluster analyses to determine which is better.
5. Determining the 'correct' number of clusters.

CSE 347/447: DATA MINING

Lecture 14: Association Analysis I



Slides partially adapted from P. Tan, M. Steinbach, and V. Kumar.

Wrap-up of Clustering

Measures of Cluster Validity

- External Index: Used to measure the extent to which cluster labels match externally supplied class labels.
  - Entropy
- Internal Index: Used to measure the goodness of a clustering structure *without* respect to external information.
  - Sum of Squared Error (SSE)
- Relative Index: Used to compare two different clusterings or clusters.
  - Often an external or internal index is used for this function, e.g., SSE or entropy

Cohesion and Separation

- Cluster Cohesion: Measures how closely related are objects in a cluster
  - Example: SSE
- Cluster Separation: Measure how distinct or well-separated a cluster is from other clusters
- Example: Squared Error
  - Cohesion is measured by the within cluster sum of squares (SSE)

$$WSS = \sum_i \sum_{x \in C_i} (x - m_i)^2$$

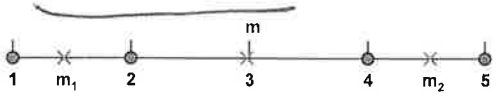
  - Separation is measured by the between cluster sum of squares

$$BSS = \sum |C_i| (m - m_i)^2$$

  - Where $|C_i|$ is the size of cluster i

Example: WSS and BSS

- Example: SSE
  - BSS + WSS = constant



K=1 cluster:

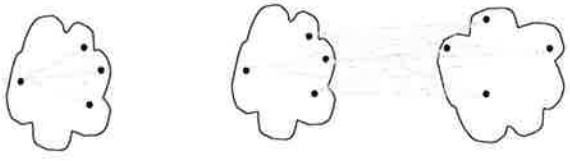
$$WSS = (1-3)^2 + (2-3)^2 + (4-3)^2 + (5-3)^2 = 10$$
$$BSS = 4 \times (3-3)^2 = 0$$
$$Total = 10 + 0 = 10$$

K=2 clusters:

$$WSS = (1-1.5)^2 + (2-1.5)^2 + (4-4.5)^2 + (5-4.5)^2 = 1$$
$$BSS = 2 \times (3-1.5)^2 + 2 \times (4.5-3)^2 = 9$$
$$Total = 1 + 9 = 10$$

Cohesion and Separation

- A proximity graph based approach can also be used for cohesion and separation.
  - Cluster cohesion is the sum of the weight of all links within a cluster.
  - Cluster separation is the sum of the weights between nodes in the cluster and nodes outside the cluster.



cohesion separation

Concept: Association Rule

- **Association Rule**
  - An implication expression of the form $X \rightarrow Y$, where X and Y are itemsets
  - Example: {Milk, Diaper} → {Beer}

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

- **Rule Evaluation Metrics**
  - Support (s)
    - Fraction of transactions that contain both X and Y
  - Confidence (c)
    - Measures how often items in Y appear in transactions that contain X

Example:

$\{Milk, Diaper\} \Rightarrow Beer$

$$s = \frac{\sigma(Milk, Diaper, Beer)}{|T|} = \frac{2}{5} = 0.4$$

$$c = \frac{\sigma(Milk, Diaper, Beer)}{\sigma(Milk, Diaper)} = \frac{2}{3} = 0.67$$

$s(X \rightarrow Y) = \frac{\sigma(X \cup Y)}{N}$

$c(X \rightarrow Y) = \frac{\sigma(X \cup Y)}{\sigma(X)}$.

antecedent ↓ consequent. 不必然有因果关系.

Formal Definition

- Given a set of transactions T, the goal of association rule mining is to find all rules having
  - support ≥ *minsup* threshold
  - confidence ≥ *minconf* threshold
- Brute-force approach:
  - List all possible association rules
  - Compute the support and confidence for each rule
  - Prune rules that fail the *minsup* and *minconf* thresholds

⇒ Computationally prohibitive!

$R = 3^d + 1 - 2^{d+1}$.

d 项.

Examples and Observations

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

Example of Rules:

{Milk,Diaper} → {Beer} (s=0.4, c=0.67)
{Milk,Beer} → {Diaper} (s=0.4, c=1.0)
{Diaper,Beer} → {Milk} (s=0.4, c=0.67)
{Beer} → {Milk,Diaper} (s=0.4, c=0.67)
{Diaper} → {Milk,Beer} (s=0.4, c=0.5)
{Milk} → {Diaper,Beer} (s=0.4, c=0.5)

Observations:

- All the above rules are binary partitions of the same itemset: {Milk, Diaper, Beer}
- Rules originating from the same itemset have identical support but can have different confidence
- Thus, we may decouple the support and confidence requirements

Two-Step Approach

1. Frequent Itemset Generation
   - Generate all itemsets whose support ≥ minsup
2. Rule Generation
   - Generate high confidence rules from each frequent itemset, where each rule is a binary partitioning of a frequent itemset

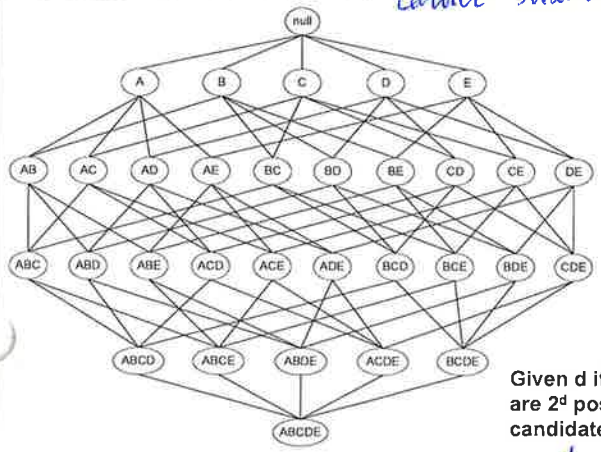
generate strong rule.

Frequent itemset generation is still computationally expensive

step 1.

Frequent Itemset Generation

Lattice structure.

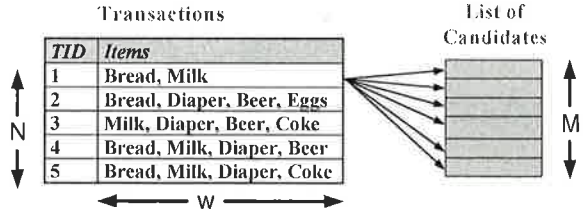


Given d items, there are 2^d possible candidate itemsets

$2^d - 1$. without $\emptyset$.

Brute-Force Approach

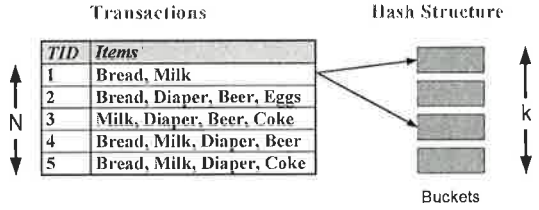
- Each itemset in the lattice is a candidate frequent itemset
- Count the support of each candidate by scanning the database



- Match each transaction against every candidate
- Complexity ~ O(NMw) => Expensive since M = 2^d !!!

Reducing Number of Comparisons

- Candidate counting:
  - Scan the database of transactions to determine the support of each candidate itemset
  - To reduce the number of comparisons, store the candidates in a hash structure
    - Instead of matching each transaction against every candidate, match it against candidates contained in the hashed buckets



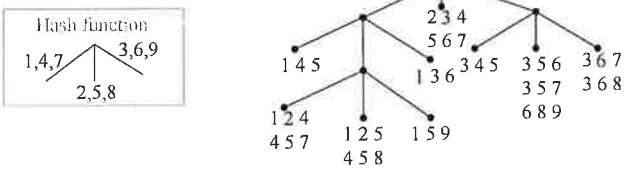
Generating Hash Tree

Suppose you have 15 candidate itemsets of length 3:

{1 4 5}, {1 2 4}, {4 5 7}, {1 2 5}, {4 5 8}, {1 5 9}, {1 3 6}, {2 3 4}, {5 6 7}, {3 4 5}, {3 5 6}, {3 5 7}, {6 8 9}, {3 6 7}, {3 6 8}

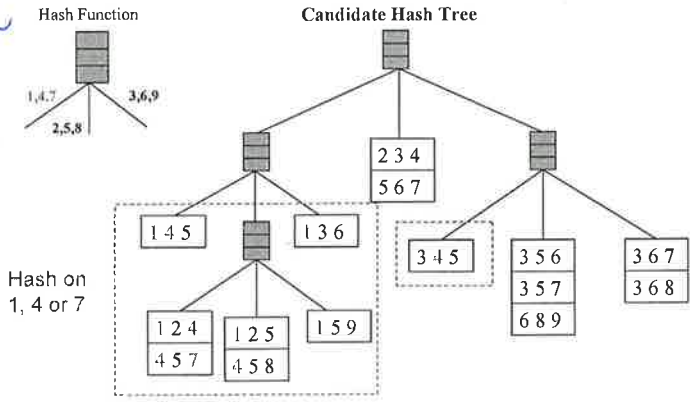
You need:

- **Hash function** $h(p) = p \bmod k$.
- **Max leaf size: max number of itemsets stored in a leaf node (if number of candidate itemsets exceeds max leaf size, split the node)**



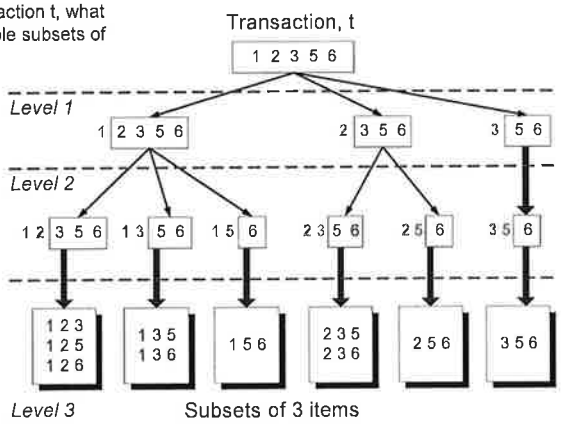
good example.

Rule Discovery: Hash Tree

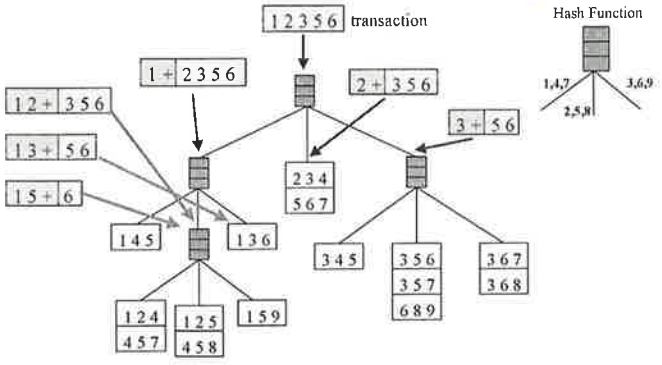


Subset Operation

Given a transaction t, what are the possible subsets of size 3?



Using Hash Tree

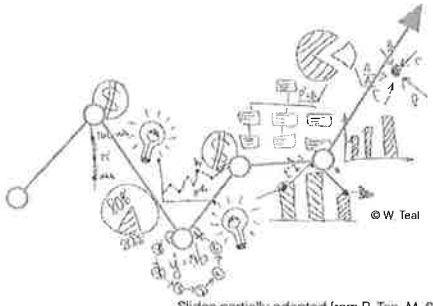


For Next Lecture

- Reading Ch. 6.5 ~ 6.8 of TSK

CSE 347/447: DATA MINING

Lecture 15: Association Analysis II



Slides partially adapted from P. Tan, M. Steinbach, and V. Kumar

Apriori Approach (cont.)

Factors Affecting Complexity

- Choice of minimum support threshold
  - lowering support threshold results in more frequent itemsets
  - this may increase number of candidates and max length of frequent itemsets
- Dimensionality (number of items) of the data set
  - more space is needed to store support count of each item
  - if number of frequent items also increases, both computation and I/O costs may also increase
- Size of database
  - since Apriori makes multiple passes, run time of algorithm may increase with number of transactions
- Average transaction width
  - transaction width increases with denser data sets
  - This may increase max length of frequent itemsets and traversals of hash tree (number of subsets in a transaction increases with its width)

Compact Representation

冗余的.

- Some itemsets are redundant because they have identical support as their supersets

TID	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10
1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1

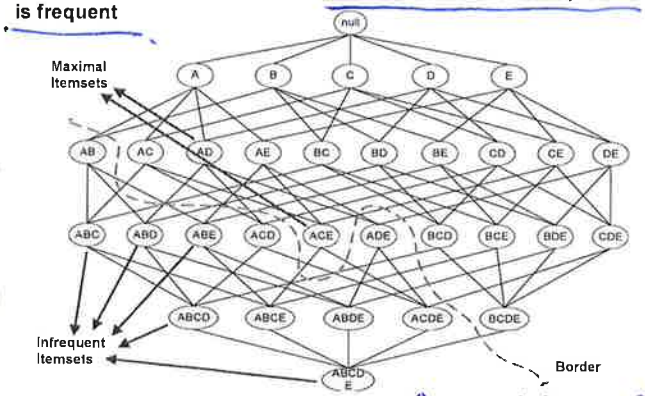
- Number of frequent itemsets $= 3 \times \sum_{k=1}^{10} \binom{10}{k}$

只对三个闭频繁项集分析即可.

eg. 支持度阈值 20%.

Maximal Frequent Itemset

An itemset is maximal frequent if none of its immediate supersets is frequent



smallest set of itemsets from which all frequent itemsets can be derived.

Closed Itemset

- An itemset is closed if none of its immediate supersets has the same support as the itemset

TID	Items
1	{A,B}
2	{B,C,D}
3	{A,B,C,D}
4	{A,B,D}
5	{A,B,C,D}

Itemset	Support
{A}	4
{B}	5
{C}	3
{D}	4
{A,B}	4
{A,C}	2
{A,D}	3
{B,C}	3
{B,D}	4
{C,D}	3

Itemset	Support
{A,B,C}	2
{A,B,D}	3
{A,C,D}	2
{B,C,D}	3
{A,B,C,D}	2

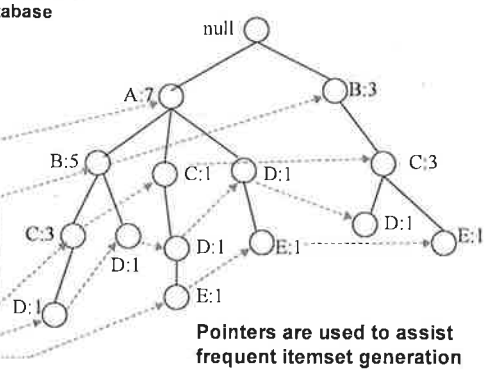
FP-Tree Construction

TID	Items
1	{A,B}
2	{B,C,D}
3	{A,C,D,E}
4	{A,D,E}
5	{A,B,C}
6	{A,B,C,D}
7	{B,C}
8	{A,B,C}
9	{A,B,D}
10	{B,C,E}

Transaction Database

Header table

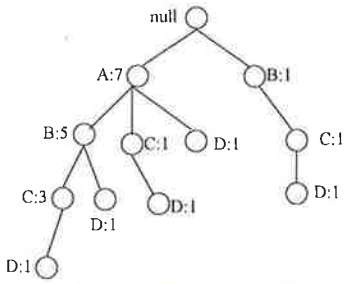
Item	Pointer
A	
B	
C	
D	
E	



Pointers are used to assist frequent itemset generation

FP-Growth

前缀路径 prefix path.



Conditional Pattern base for D:
P = {(A:1,B:1,C:1),
(A:1,B:1),
(A:1,C:1),
(A:1),
(B:1,C:1)}

循环

Recursively apply FP-growth on P

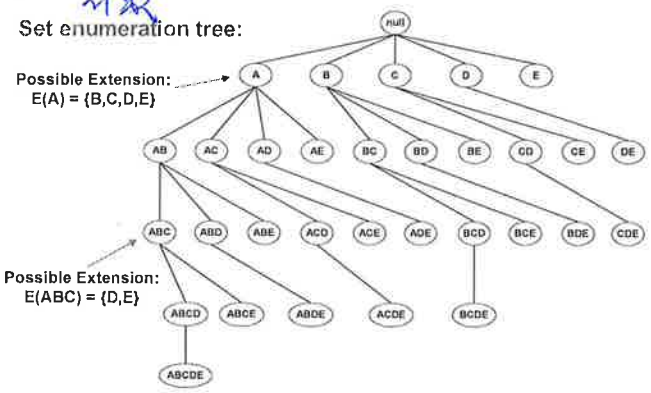
Frequent Itemsets found (with sup > 1):
AD, BD, CD, ACD, BCD

更新条件FP树时，prefix path上的点的支持度会变.

Tree Projection

计数

Set enumeration tree:



Tree Projection

- Items are listed in lexicographic order
- Each node P stores the following information:
  - Itemset for node P
  - List of possible lexicographic extensions of P: E(P)
  - Pointer to projected database of its ancestor node
  - Bitvector containing information about which transactions in the projected database contain the itemset

Projected Database

Original Database:

TID	Items
1	{A,B}
2	{B,C,D}
3	{A,C,D,E}
4	{A,D,E}
5	{A,B,C}
6	{A,B,C,D}
7	{B,C}
8	{A,B,C}
9	{A,B,D}
10	{B,C,E}

Projected Database for node A:

TID	Items
1	{B}
2	{}
3	{C,D,E}
4	{D,E}
5	{B,C}
6	{B,C,D}
7	{}
8	{B,C}
9	{B,D}
10	{}

For each transaction T, projected transaction at node A is $T \cap E(A)$

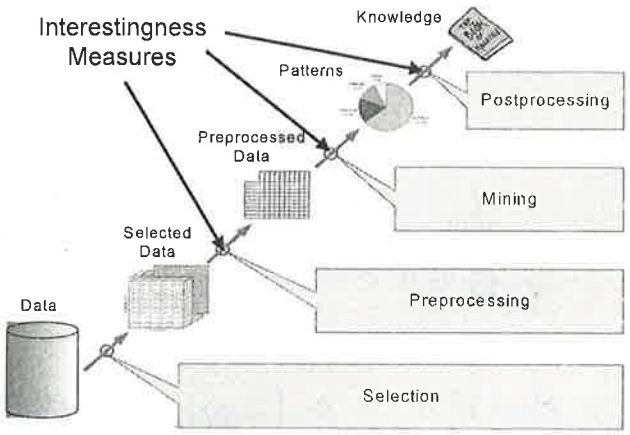
step 2.

Rule Generation

Pattern Evaluation

- Association rule algorithms tend to produce too many rules
  - many of them are uninteresting or redundant
  - Redundant if {A,B,C} → {D} and {A,B} → {D} have same support & confidence
- Interestingness measures can be used to prune/rank the derived patterns
- In the original formulation of association rules, support & confidence are the only measures used

"Interestingness"



规则
忽略了后件中项集的支持度.

Interestingness Measure

Given a rule X → Y, information needed to compute rule interestingness can be obtained from a contingency table

列联表.

Contingency table for X → Y

	Y	$\overline{Y}$	
X	f_{11}	f_{10}	f_{1+}
$\overline{X}$	f_{01}	f_{00}	f_{0+}
	f_{+1}	f_{+0}	|T|

f_{11}: support of X and Y
f_{10}: support of X and $\overline{Y}$
f_{01}: support of $\overline{X}$ and Y
f_{00}: support of $\overline{X}$ and $\overline{Y}$

Used to define various measures
- support, confidence, lift, Gini, J-measure, etc.

Drawback of Confidence

	Coffee	$\overline{\text{Coffee}}$	
Tea	15	5	20
$\overline{\text{Tea}}$	75	5	80
	90	10	100

Association Rule: Tea → Coffee

Confidence= P(Coffee|Tea) = 0.75
but P(Coffee) = 0.9
⇒ Although confidence is high, rule is misleading
⇒ P(Coffee|$\overline{\text{Tea}}$) = 0.9375

$I(A,B) \begin{cases} =1 & \text{独立.} \\ >1 & \text{正相关} \\ <1 & \text{负相关} \end{cases}$

Statistical-Based Measure

Measures that take into account statistical dependence

① $Lift = \frac{P(Y|X)}{P(Y)}$ $= \frac{C(A\to B)}{S(B)}$ 提升度 $= \frac{S(A,B)}{S(A)\times S(B)} = \frac{N f_{11}}{f_{1+}\times f_{+1}}$

② $Interest = \frac{P(X,Y)}{P(X)P(Y)}$ interest factor 兴趣因子.

③ $PS = P(X,Y) - P(X)P(Y)$

④ $\phi - coefficient = \frac{P(X,Y)-P(X)P(Y)}{\sqrt{P(X)[1-P(X)]P(Y)[1-P(Y)]}}$

$\phi = \frac{f_{11}f_{00} - f_{01}f_{10}}{\sqrt{f_{1+}f_{+1}f_{0+}f_{+0}}} \in [-1, 1]$.
P N.

Example: Lift/Interest

	Coffee	$\overline{\text{Coffee}}$	
Tea	15	5	20
$\overline{\text{Tea}}$	75	5	80
	90	10	100

Association Rule: Tea → Coffee

Confidence= P(Coffee|Tea) = 0.75
but P(Coffee) = 0.9
⇒ Lift = 0.75/0.9= 0.8333 (< 1, therefore is negatively associated)

⑤ $IS(AB) = \sqrt{I(A,B)\times S(A,B)} = \frac{S(A,B)}{\sqrt{S(A)S(B)}}$
$= cosine(\vec{A},\vec{B})$.

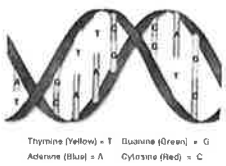
drawback 类似 cos-distance. 3.

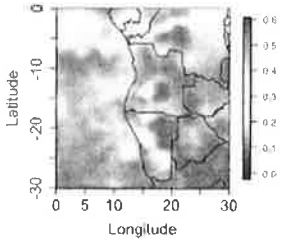
CSE 347/447: DATA MINING

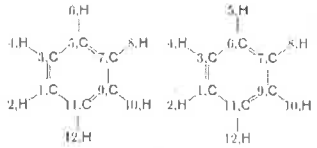
Lecture 16: Association Analysis III

Slides partially adapted from P. Tan, M. Steinbach, and V. Kumar.

Beyond Shopping Baskets





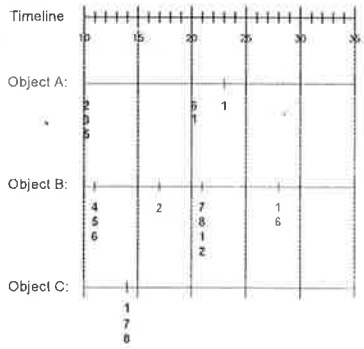


Sequence

Sequence Data

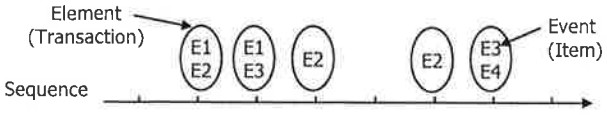
Sequence Database:

Object	Timestamp	Events
A	10	2, 3, 5
A	20	6, 1
A	23	1
B	11	4, 5, 6
B	17	2
B	21	7, 8, 1, 2
B	28	1, 6
C	14	1, 8, 7



Examples

Sequence Database	Sequence	Element (Transaction)	Event (Item)
Customer	Purchase history of a given customer	A set of items bought by a customer at time t	Books, diary products, CDs, etc
Web Data	Browsing activity of a particular Web visitor	A collection of files viewed by a Web visitor after a single mouse click	Home page, index page, contact info, etc
Event data	History of events generated by a given sensor	Events triggered by a sensor at time t	Types of alarms generated by sensors
Genome sequences	DNA sequence of a particular species	An element of the DNA sequence	Bases A,T,G,C



Examples

- Web sequence:

  < {Homepage} {Electronics} {Digital Cameras} {Canon Digital Camera} {Shopping Cart} {Order Confirmation} {Return to Shopping} >

- Sequence of initiating events causing the nuclear accident at 3-mile Island:

  (http://stellar-one.com/nuclear/staff_reports/summary_SOE_the_initiating_event.htm)

  < {clogged resin} {outlet valve closure} {loss of feedwater} {condenser polisher outlet valve shut} {booster pumps trip} {main waterpump trips} {main turbine trips} {reactor pressure increases}>

- Sequence of books checked out at a library:

  <{Fellowship of the Ring} {The Two Towers} {Return of the King}>

Candidate Generation

- Base case (k=2):
  - Merging two frequent 1-sequences $<\{i_1\}>$ and $<\{i_2\}>$ will produce two candidate 2-sequences: $<\{i_1\}\ \{i_2\}>$ and $<\{i_1\ i_2\}>$
- General case (k>2):
  - A frequent *(k-1)*-sequence w_1 is merged with another frequent *(k-1)*-sequence w_2 to produce a candidate *k*-sequence if the subsequence obtained by removing the first event in w_1 is the same as the subsequence obtained by removing the last event in w_2
    - The resulting candidate after merging is given by the sequence w_1 extended with the last event of w_2.
      - If the last two events in w_2 belong to the same element, then the last event in w_2 becomes part of the last element in w_1
      - Otherwise, the last event in w_2 becomes a separate element appended to the end of w_1

Candidate Generation Example

- Merging the sequences
  w_1=<{1} {2 3} {4}> and w_2 =<{2 3} {4 5}>
  will produce the candidate sequence < {1} {2 3} {4 5}> because the last two events in w_2 (4 and 5) belong to the same element
- Merging the sequences
  w_1=<{1} {2 3} {4}> and w_2 =<{2 3} {4} {5}>
  will produce the candidate sequence < {1} {2 3} {4} {5}> because the last two events in w_2 (4 and 5) do not belong to the same element
- We do not have to merge the sequences
  w_1 =<{1} {2 6} {4}> and w_2 =<{1} {2} {4 5}>
  to produce the candidate < {1} {2 6} {4 5}> because if the latter is a viable candidate, then it can be obtained by merging w_1 with < {1} {2 6} {5}>

GSP Example

Frequent 3-sequences

< {1} {2} {3} >
< {1} {2 5} >
< {1} {5} {3} >
< {2} {3} {4} >
< {2 5} {3} >
< {3} {4} {5} >
< {5} {3 4} >

→ Candidate Generation

< {1} {2} {3} {4} >
< {1} {2 5} {3} >
< {1} {5} {3 4} >
< {2} {3} {4} {5} >
< {2 5} {3 4} >

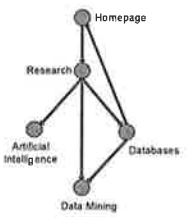
→ Candidate Pruning

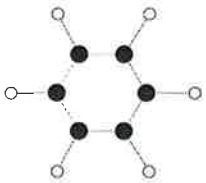
< {1} {2 5} {3} >

Graph

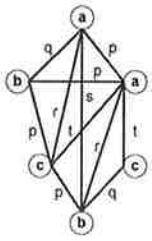
Frequent Subgraph Mining

- Extend association rule mining to finding frequent subgraphs
- Useful for Web Mining, computational chemistry, bioinformatics, spatial data sets, etc

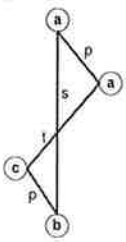




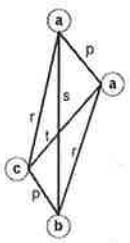
Graph Definition



(a) Labeled Graph



(b) Subgraph



(c) Induced Subgraph

Apriori-Like Algorithm

- Find frequent 1-subgraphs
- Repeat
  - Candidate generation
    - Use frequent (*k-1*)-subgraphs to generate candidate *k*-subgraph
  - Candidate pruning
    - Prune candidate subgraphs that contain infrequent (*k-1*)-subgraphs
  - Support counting
    - Count the support of each remaining candidate
  - Eliminate candidate *k*-subgraphs that are infrequent

For Next Lecture

- Reading Ch. 10 of TSK

CSE 347/447: DATA MINING

Lecture 17: Anomaly Detection

Slides partially adapted from P. Tan, M. Steinbach, and V. Kumar.

Candidate Generation Example

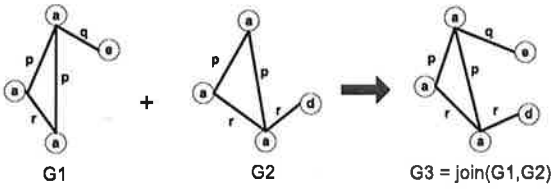
- Merging the sequences
  w_1=<{1} {2 3} {4}> and w_2 =<{2 3} {4 5}>
  will produce the candidate sequence < {1} {2 3} {4 5}> because the last two events in w_2 (4 and 5) belong to the same element
- Merging the sequences
  w_1=<{1} {2 3} {4}> and w_2 =<{2 3} {4} {5}>
  will produce the candidate sequence < {1} {2 3} {4} {5}> because the last two events in w_2 (4 and 5) do not belong to the same element
- We do not have to merge the sequences
  w_1 =<{1} {2 6} {4}> and w_2 =<{1} {2} {4 5}>
  to produce the candidate < {1} {2 6} {4 5}> because if the latter is a viable candidate, then it can be obtained by merging w_1 with
  < {1} {2 6} {5}>

Frequent Subgraph

Frequent Subgraph

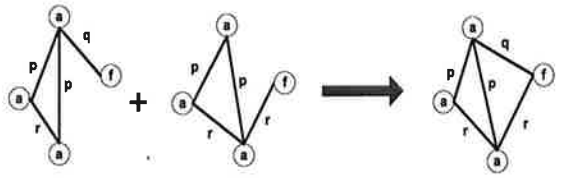
- Support:
  - number of graphs that contain a particular subgraph
- Apriori principle still holds
- Level-wise (Apriori-like) approach:
  - Vertex growing:
    - k is the number of vertices
  - Edge growing:
    - k is the number of edges

Vertex Growing



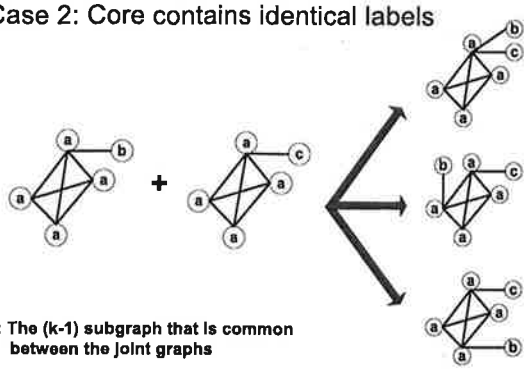
$$M_{G1} = \begin{pmatrix} 0 & p & p & q \\ p & 0 & r & 0 \\ p & r & 0 & 0 \\ q & 0 & 0 & 0 \end{pmatrix} \quad M_{G2} = \begin{pmatrix} 0 & p & p & 0 \\ p & 0 & r & 0 \\ p & r & 0 & r \\ 0 & 0 & r & 0 \end{pmatrix} \quad M_{G3} = \begin{pmatrix} 0 & p & p & 0 & q \\ p & 0 & r & 0 & 0 \\ p & r & 0 & r & 0 \\ 0 & 0 & r & 0 & 0 \\ q & 0 & 0 & 0 & 0 \end{pmatrix}$$

Edge Growth



Multiplicity of Cand. (Edge Growing)

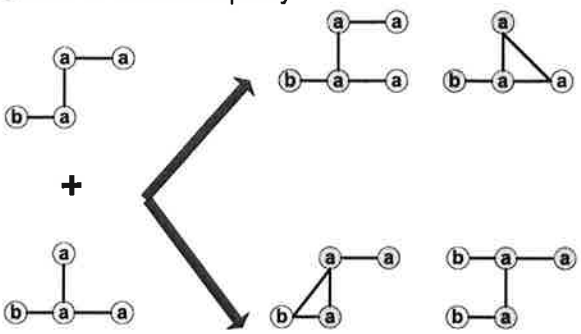
- Case 2: Core contains identical labels



Core: The (k-1) subgraph that is common between the joint graphs

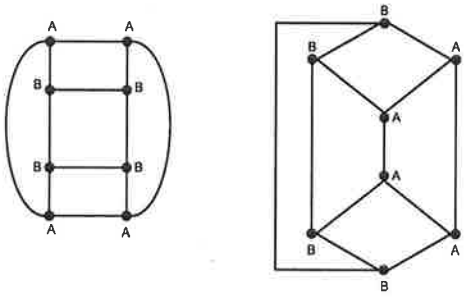
Multiplicity of Cand. (Edge Growing)

- Case 3: Core multiplicity



Graph Isomorphism

A graph is isomorphic if it is topologically equivalent to another graph

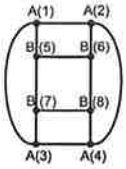


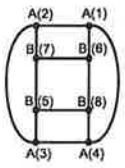
Graph Isomorphism

Test for graph isomorphism is needed:

- During candidate generation step, to determine whether a candidate has been generated
- During candidate pruning step, to check whether its (*k-1*)-subgraphs are frequent
- During candidate counting, to check whether a candidate is contained within another graph

Adjacent Matrix





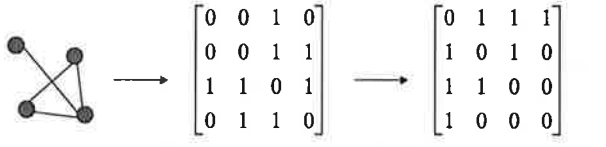
	A(1)	A(2)	A(3)	A(4)	B(5)	B(6)	B(7)	B(8)
A(1)	1	1	1	0	1	0	0	0
A(2)	1	1	0	1	0	1	0	0
A(3)	1	0	1	1	0	0	1	0
A(4)	0	1	1	1	0	0	0	1
B(5)	1	0	0	0	1	1	1	0
B(6)	0	1	0	0	1	1	0	1
B(7)	0	0	1	0	1	0	1	1
B(8)	0	0	0	1	0	1	1	1

	A(1)	A(2)	A(3)	A(4)	B(5)	B(6)	B(7)	B(8)
A(1)	1	1	0	1	0	1	0	0
A(2)	1	1	1	0	0	0	1	0
A(3)	0	1	1	1	1	0	0	0
A(4)	1	0	1	1	0	0	0	1
B(5)	0	0	1	0	1	0	1	1
B(6)	1	0	0	0	0	1	1	1
B(7)	0	1	0	0	1	1	1	0
B(8)	0	0	0	1	1	1	0	1

Graph Isomorphism

Use canonical labeling to handle isomorphism

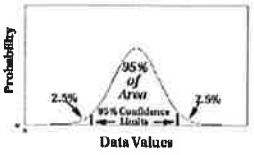
- Map each graph into an ordered string representation (known as its code) such that two isomorphic graphs will be mapped to the same canonical encoding
- Example:
  - Lexicographically largest adjacency matrix



String: 0010001111010110 **Canonical: 0111101011001000**

del Based Approaches

- Assume a parametric model describing the distribution of the data (e.g., normal distribution)
- Apply a statistical test that depends on
  - Data distribution
  - Parameter of distribution (e.g., mean, variance)
  - Number of expected outliers (confidence limit)



Likelihood Approach

- Assume the data set D contains samples from a mixture of two probability distributions:
  - M (majority distribution)
  - A (anomalous distribution)
- General Approach:
  - Initially, assume all the data points belong to M
  - Let $L_t(D)$ be the log likelihood of D at time t
  - For each point x_t that belongs to M, move it to A
    - Let L_{t+1} (D) be the new log likelihood.
    - Compute the difference, $\Delta = L_t(D) - L_{t+1}(D)$
    - If $\Delta > c$ (some threshold), then x_t is declared as an anomaly and moved permanently from M to A

Likelihood Approach

how to find outliers.

- Data distribution, $D = (1 - \lambda) M + \lambda A$
- M is a probability distribution estimated from data
  - Can be based on any modeling method (naïve Bayes, maximum entropy, etc)
- A is initially assumed to be uniform distribution
- Likelihood at time t:

$$L_t(D) = \prod_{i=1}^{N} P_D(x_i) = \left((1-\lambda)^{|M_t|} \prod_{x_i \in M_t} P_{M_t}(x_i)\right)\left(\lambda^{|A_t|} \prod_{x_i \in A_t} P_{A_t}(x_i)\right)$$

$$LL_t(D) = |M_t| \log(1-\lambda) + \sum_{x_i \in M_t} \log P_{M_t}(x_i) + |A_t| \log \lambda + \sum_{x_i \in A_t} \log P_{A_t}(x_i)$$

Drawbacks of Statistical Approaches

- Most of the tests are for a single attribute
- In many cases, data distribution may not be known
- For high dimensional data, it may be difficult to estimate the true distribution

Proximity Based Approaches

- Data is represented as a vector of features
- Three major approaches
  - Nearest-neighbor based
  - Density based
  - Clustering based

Nearest Neighbor Based Approach

anti-KNN.

- Compute the distance between every pair of data points
- There are various ways to define outliers:
  - Data points for which there are fewer than *p* neighboring points within a distance *D*
  - The top n data points whose distance to the kth nearest neighbor is greatest
  - The top n data points whose average distance to the k nearest neighbors is greatest