CMU SCS

Carnegie Mellon Univ.
Dept. of Computer Science
15-415/615 – DB Applications

Lecture # 24: Data Warehousing / Data Mining
(R&G, ch 25 and 26)

CMU SCS

Data mining - detailed outline

- Problem
- Getting the data: Data Warehouses, DataCubes, OLAP
- Supervised learning: decision trees
- Unsupervised learning
  - association rules
  - (clustering)

CMU SCS

Problem

Given: multiple data sources
Find: patterns (classifiers, rules, clusters, outliers...)

PGH

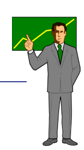
NY

sales(p-id, c-id, date, $price)

???

customers(c-id, age, income, ...)

SF



CMU SCS

Data Ware-housing

- First step: collect the data, in a single place (= Data Warehouse)
- How?
- How often?
- How about discrepancies / non-homegeneities?

CMU SCS

Data Ware-housing

- First step: collect the data, in a single place (= Data Warehouse)
- How? A: Triggers/Materialized views
- How often? A: [Art!]
- How about discrepancies / non-homegeneities? A: Wrappers/Mediators

CMU SCS

Data Ware-housing

- Step 2: collect counts. (DataCubes/OLAP) Eg.:

OLAP

Problem: "is it true that shirts in large sizes sell better in dark colors?"

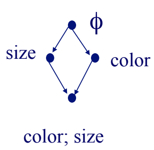
sales

ci-d	p-id	Size	Color	$
C10	Shirt	L	Blue	30
~~C10~~	~~Pants~~	~~XL~~	~~Red~~	~~50~~
C20	Shirt	XL	White	20
…				

C / S	S	M	L	TOT
Red	20	3	5	28
Blue	3	3	8	14
Gray	0	0	5	5
TOT	23	6	18	47

DataCubes

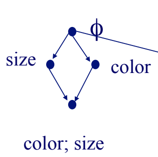
'color', 'size': DIMENSIONS
'count': MEASURE



C / S	S	M	L	TOT
Red	20	3	5	28
Blue	3	3	8	14
Gray	0	0	5	5
TOT	23	6	18	47

DataCubes

'color', 'size': DIMENSIONS
'count': MEASURE

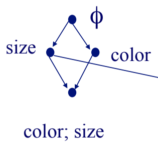


C / S	S	M	L	TOT
Red	20	3	5	28
Blue	3	3	8	14
Gray	0	0	5	5
TOT	23	6	18	47

CMU SCS

DataCubes

'color', 'size': DIMENSIONS
'count': MEASURE

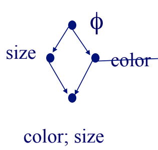


C / S	S	M	L	TOT
Red	20	3	5	28
Blue	3	3	8	14
Gray	0	0	5	5
TOT	23	6	18	47

CMU SCS

DataCubes

'color', 'size': DIMENSIONS
'count': MEASURE

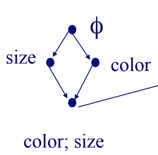


C / S	S	M	L	TOT
Red	20	3	5	28
Blue	3	3	8	14
Gray	0	0	5	5
TOT	23	6	18	47

CMU SCS

DataCubes

'color', 'size': DIMENSIONS
'count': MEASURE

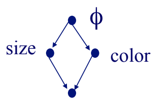


C / S	S	M	L	TOT
Red	20	3	5	28
Blue	3	3	8	14
Gray	0	0	5	5
TOT	23	6	18	47

CMU SCS

DataCubes

'color', 'size': DIMENSIONS
'count': MEASURE



color; size

C / S	S	M	L	TOT
Red	20	3	5	28
Blue	3	3	8	14
Gray	0	0	5	5
TOT	23	6	18	47

DataCube

CMU SCS

DataCubes

SQL query to generate DataCube:

- Naively (and painfully:)

```
select size, color, count(*)
from sales where p-id = 'shirt'
group by size, color

select size, count(*)
from sales where p-id = 'shirt'
group by size
...
```


CMU SCS

DataCubes

SQL query to generate DataCube:

- with 'cube by' keyword:

select size, color, count(*)
from sales
where p-id = 'shirt'
cube by size, color

CMU SCS

DataCubes

DataCube issues:

Q1: How to store them (and/or materialize portions on demand)

Q2: Which operations to allow

CMU SCS

DataCubes

DataCube issues:

Q1: How to store them (and/or materialize portions on demand) A: ROLAP/MOLAP

Q2: Which operations to allow A: roll-up, drill down, slice, dice

[More details: book by Han+Kamber]

CMU SCS

DataCubes

Q1: How to store a dataCube?

C / S	S	M	L	TOT
Red	20	3	5	28
Blue	3	3	8	14
Gray	0	0	5	5
TOT	23	6	18	47

CMU SCS

DataCubes

Q1: How to store a dataCube?

A1: Relational (R-OLAP)

Color	Size	count
'all'	'all'	47
Blue	'all'	14
Blue	M	3
…		

C / S	S	M	L	TOT
Red	20	3	5	28
Blue	3	3	8	14
Gray	0	0	5	5
TOT	23	6	18	47

Faloutsos & Pavlo CMU SCS 15-415/615 19

CMU SCS

DataCubes

Q1: How to store a dataCube?

A2: Multi-dimensional (M-OLAP)

A3: Hybrid (H-OLAP)

C / S	S	M	L	TOT
Red	20	3	5	28
Blue	3	3	8	14
Gray	0	0	5	5
TOT	23	6	18	47

Faloutsos & Pavlo CMU SCS 15-415/615 20

CMU SCS

DataCubes

Pros/Cons:

ROLAP strong points: (DSS, Metacube)

Faloutsos & Pavlo CMU SCS 15-415/615 21

DataCubes

Pros/Cons:

ROLAP strong points: (DSS, Metacube)

- use existing RDBMS technology
- scale up better with dimensionality

DataCubes

Pros/Cons:

MOLAP strong points: (EssBase/hyperion.com)

- faster indexing

(careful with: high-dimensionality; sparseness)

HOLAP: (MS SQL server OLAP services)

- detail data in ROLAP; summaries in MOLAP

DataCubes

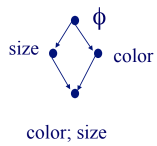
Q1: How to store a dataCube

Q2: What operations should we support?

CMU SCS

DataCubes

Q2: What operations should we support?



C / S	S	M	L	TOT
Red	20	3	5	28
Blue	3	3	8	14
Gray	0	0	5	5
TOT	23	6	18	47

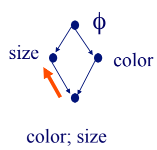
Faloutsos & Pavlo CMU SCS 15-415/615 25

CMU SCS

DataCubes

Q2: What operations should we support?

Roll-up



C / S	S	M	L	TOT
Red	20	3	5	28
Blue	3	3	8	14
Gray	0	0	5	5
TOT	23	6	18	47

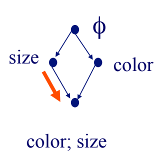
Faloutsos & Pavlo CMU SCS 15-415/615 26

CMU SCS

DataCubes

Q2: What operations should we support?

Drill-down



C / S	S	M	L	TOT
Red	20	3	5	28
Blue	3	3	8	14
Gray	0	0	5	5
TOT	23	6	18	47

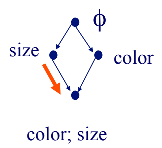
Faloutsos & Pavlo CMU SCS 15-415/615 27

CMU SCS

DataCubes

Q2: What operations should we support?

Slice



C / S	S	M	L	TOT
Red	20	3	5	28
Blue	3	3	8	14
Gray	0	0	5	5
TOT	23	6	18	47

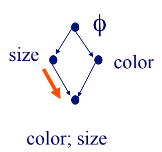
Faloutsos & Pavlo CMU SCS 15-415/615 28

CMU SCS

DataCubes

Q2: What operations should we support?

Dice



C / S	S	M	L	TOT
Red	20	3	5	28
Blue	3	3	8	14
Gray	0	0	5	5
TOT	23	6	18	47

Faloutsos & Pavlo CMU SCS 15-415/615 29

CMU SCS

DataCubes

Q2: What operations should we support?

- Roll-up
- Drill-down
- Slice
- Dice
- (Pivot/rotate; drill-across; drill-through
- top N
- moving averages, etc)

Faloutsos & Pavlo CMU SCS 15-415/615 30

CMU SCS

D/W - OLAP - Conclusions

- D/W: copy (summarized) data + analyze
- OLAP - concepts:
  - DataCube
  - R/M/H-OLAP servers
  - 'dimensions'; 'measures'

Faloutsos & Pavlo CMU SCS 15-415/615 31

CMU SCS

Outline

- Problem
- Getting the data: Data Warehouses, DataCubes, OLAP
- Supervised learning: decision trees
- Unsupervised learning
  - association rules
  - (clustering)

Faloutsos & Pavlo CMU SCS 15-415/615 32

CMU SCS

Decision trees - Problem

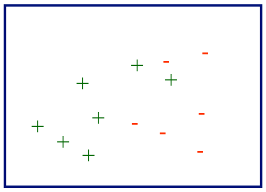
Age	Chol-level	Gender	…	CLASS-ID
30	150	M		+
				…
				-
				??

Faloutsos & Pavlo CMU SCS 15-415/615 33

Decision trees

- Pictorially, we have

num. attr#2
(eg., chol-level)

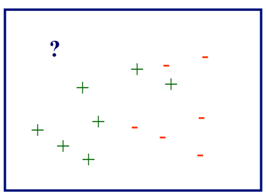


num. attr#1 (eg., 'age')

Decision trees

- and we want to label '?'

num. attr#2
(eg., chol-level)

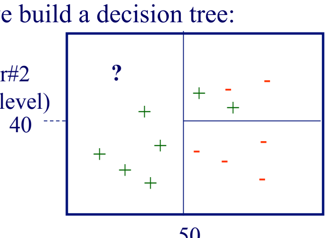


num. attr#1 (eg., 'age')

Decision trees

- so we build a decision tree:

num. attr#2
(eg., chol-level)

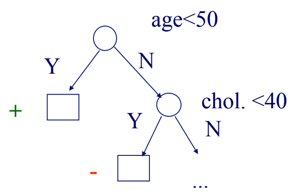


num. attr#1 (eg., 'age')

CMU SCS

Decision trees

- so we build a decision tree:



Faloutsos & Pavlo CMU SCS 15-415/615 37

CMU SCS

Outline

- Problem
- Getting the data: Data Warehouses, DataCubes, OLAP
- Supervised learning: decision trees
  - problem
  - approach
  - scalability enhancements
- Unsupervised learning
  - association rules
  - (clustering)

Faloutsos & Pavlo CMU SCS 15-415/615 38

CMU SCS

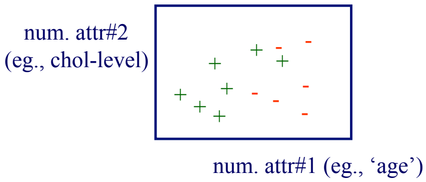
Decision trees

- Typically, two steps:
  - tree building
  - tree pruning (for over-training/over-fitting)

Faloutsos & Pavlo CMU SCS 15-415/615 39

Tree building

- How?



Tree building

- How?
- A: Partition, recursively - pseudocode:

Partition (Dataset S)
  if all points in S have same label
  then return
  evaluate splits along each attribute A
  pick best split, to divide S into S1 and S2
  Partition(S1); Partition(S2)

Not In Exam=N.I.E.

Tree building

- Q1: how to introduce splits along attribute A_i
- Q2: how to evaluate a split?

N.I.E.

Tree building

- Q1: how to introduce splits along attribute A_i
- A1:
  - for num. attributes:
    - binary split, or
    - multiple split
  - for categorical attributes:
    - compute all subsets (expensive!), or
    - use a greedy algo

N.I.E.

Tree building

- Q1: how to introduce splits along attribute A_i
- Q2: how to evaluate a split?

N.I.E.

Tree building

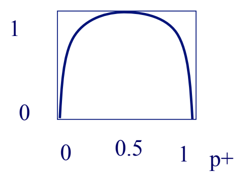
- Q1: how to introduce splits along attribute A_i
- Q2: how to evaluate a split?
- A: by how close to uniform each subset is - ie., we need a measure of uniformity:

N.I.E.

Tree building

entropy: H(p+, p-)

Any other measure?

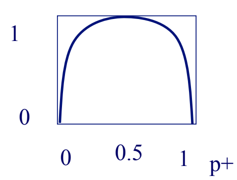


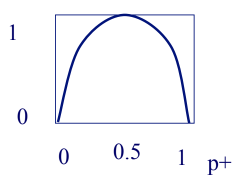
N.I.E.

Tree building

entropy: $H(p_+, p_-)$

'gini' index: $1-p_+^2 - p_-^2$





N.I.E.

Tree building

entropy: $H(p_+, p_-)$

'gini' index: $1-p_+^2 - p_-^2$

(How about multiple labels?)

CMU SCS

N.I.E.

Tree building

Intuition:

- entropy: #bits to encode the class label
- gini: classification error, if we randomly guess '+' with prob. p_+

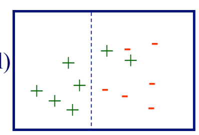
CMU SCS

N.I.E.

Tree building

Thus, we choose the split that reduces entropy/classification-error the most: Eg.:

num. attr#2 (eg., chol-level)



num. attr#1 (eg., 'age')

CMU SCS

N.I.E.

Tree building

- Before split: we need

  $(n_+ + n_-) * H(p_+, p_-) = (7+6) * H(7/13, 6/13)$

  bits total, to encode all the class labels
- After the split we need:

  0 bits for the first half and

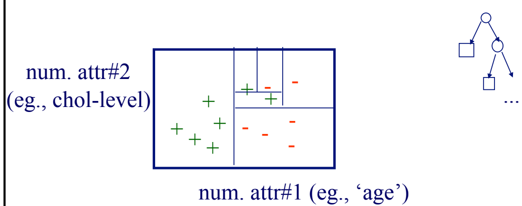
  (2+6) * H(2/8, 6/8) bits for the second half

CMU SCS

N.I.E.

Tree pruning

- What for?



CMU SCS

N.I.E.

Tree pruning

Shortcut for scalability: DYNAMIC pruning:

- stop expanding the tree, if a node is 'reasonably' homogeneous
  - ad hoc threshold [Agrawal+, vldb92]
  - (Minimum Description Language (MDL) criterion (SLIQ) [Mehta+, edbt96])

CMU SCS

N.I.E.

Tree pruning

- Q: How to do it?
- A1: use a 'training' and a 'testing' set - prune nodes that improve classification in the 'testing' set. (Drawbacks?)
- (A2: or, rely on MDL (= Minimum Description Language))

N.I.E.

Outline

- Problem
- Getting the data: Data Warehouses, DataCubes, OLAP
- Supervised learning: decision trees
  - problem
  - approach
  - scalability enhancements
- Unsupervised learning
  - association rules
  - (clustering)

N.I.E.

Scalability enhancements

- Interval Classifier [Agrawal+,vldb92]: dynamic pruning
- SLIQ: dynamic pruning with MDL; vertical partitioning of the file (but label column has to fit in core)
- SPRINT: even more clever partitioning

Conclusions for classifiers

- Classification through trees
- Building phase - splitting policies
- Pruning phase (to avoid over-fitting)
- For scalability:
  - dynamic pruning
  - clever data partitioning

CMU SCS

Outline

- Problem
- Getting the data: Data Warehouses, DataCubes, OLAP
- Supervised learning: decision trees
  - problem
  - approach
  - scalability enhancements
- Unsupervised learning
  - association rules
  - (clustering)

CMU SCS

Association rules - idea

[Agrawal+SIGMOD93]

- Consider 'market basket' case:
  (milk, bread)
  (milk)
  (milk, chocolate)
  (milk, bread)
- Find 'interesting things', eg., rules of the form:
  milk, bread -> chocolate | 90%

CMU SCS

Association rules - idea

In general, for a given rule
  Ij, Ik, ... Im -> Ix | c
'c' = 'confidence' (how often people by Ix, given that they have bought Ij, ... Im
's' = support: how often people buy Ij, ... Im, Ix

Association rules - idea

Problem definition:

- given
  - a set of 'market baskets' (=binary matrix, of N rows/ baskets and M columns/products)
  - min-support 's' and
  - min-confidence 'c'
- find
  - all the rules with higher support and confidence

Association rules - idea

Closely related concept: "large itemset"

Ij, Ik, ... Im, Ix

is a 'large itemset', if it appears more than 'min-support' times

Observation: once we have a 'large itemset', we can find out the qualifying rules easily (how?)

Thus, let's focus on how to find 'large itemsets'

Association rules - idea

Naive solution: scan database once; keep 2**|I| counters

Drawback?

Improvement?

CMU SCS

Association rules - idea

Naive solution: scan database once; keep 2**|I| counters

Drawback? 2**1000 is prohibitive...

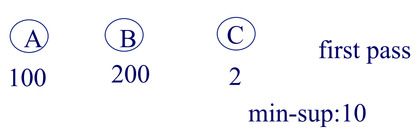
Improvement? scan the db |I| times, looking for 1-, 2-, etc itemsets

Eg., for |I|=3 items only (A, B, C), we have

Faloutsos & Pavlo CMU SCS 15-415/615 64

CMU SCS

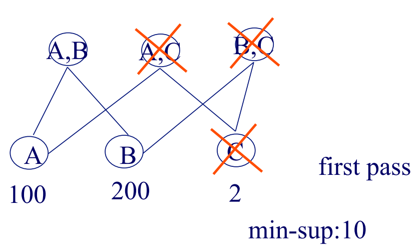
Association rules - idea



Faloutsos & Pavlo CMU SCS 15-415/615 65

CMU SCS

Association rules - idea



Faloutsos & Pavlo CMU SCS 15-415/615 66

CMU SCS

Association rules - idea

Anti-monotonicity property:

if an itemset fails to be 'large', so will every superset of it (hence all supersets can be pruned)

Sketch of the (famous!) 'a-priori' algorithm

Let *L(i-1)* be the set of large itemsets with *i-1* elements

Let *C(i)* be the set of candidate itemsets (of size *i*)

Faloutsos & Pavlo CMU SCS 15-415/615 67

CMU SCS

Association rules - idea

Compute L(1), by scanning the database.

repeat, for i=2,3...,

- '**join**' L(i-1) with itself, to generate C(i)
  - two itemset can be joined, if they agree on their first *i-2* elements
- **prune** the itemsets of C(i) (how?)
- scan the db, finding the counts of the C(i) itemsets - set this to be L(i)
- unless L(i) is empty, repeat the loop

Faloutsos & Pavlo CMU SCS 15-415/615 68

CMU SCS

Association rules - Conclusions

Association rules: a great tool to find patterns

- easy to understand its output
- fine-tuned algorithms exist

Faloutsos & Pavlo CMU SCS 15-415/615 69

CMU SCS

Overall Conclusions

- Data Mining = ``Big Data'' Analytics = Business Intelligence:
  - of **high** commercial, government and research interest
- DM = DB+ ML+ Stat+Sys

- Data warehousing / OLAP: to get the data
- Tree classifiers (SLIQ, SPRINT)
- Association Rules - 'a-priori' algorithm
- (clustering: BIRCH, CURE, OPTICS)

Faloutsos & Pavlo CMU SCS 15-415/615 70

CMU SCS

Reading material

- Agrawal, R., T. Imielinski, A. Swami, *'Mining Association Rules between Sets of Items in Large Databases'*, SIGMOD 1993.
- M. Mehta, R. Agrawal and J. Rissanen, `*SLIQ: A Fast Scalable Classifier for Data Mining*', Proc. of the Fifth Int'l Conference on Extending Database Technology (EDBT), Avignon, France, March 1996

Faloutsos & Pavlo CMU SCS 15-415/615 71

CMU SCS

Additional references

- Agrawal, R., S. Ghosh, et al. (Aug. 23-27, 1992). *An Interval Classifier for Database Mining Applications*. VLDB Conf. Proc., Vancouver, BC, Canada.
- Jiawei Han and Micheline Kamber, *Data Mining* , Morgan Kaufman, 2001, chapters 2.2-2.3, 6.1-6.2, 7.3.5

Faloutsos & Pavlo CMU SCS 15-415/615 72