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

Data Warehousing / Data Mining
(R&G, ch 25 and 26)
*C. Faloutsos and A. Pavlo*

---

**Data mining - detailed outline**

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

---

**Problem**

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

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

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

PGH

---

**Data Ware-housing**

First step: collect the data, in a single place (= Data Warehouse)

How?
How often?
How about discrepancies / non-homegeneities?
Data Warehousing

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

Data Warehouse

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

OLAP

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

```
<table>
<thead>
<tr>
<th>sales</th>
<th>ci-d</th>
<th>p-id</th>
<th>size</th>
<th>Color</th>
<th>$</th>
</tr>
</thead>
<tbody>
<tr>
<td>C10</td>
<td>Shirt</td>
<td>L</td>
<td>Blue</td>
<td>20</td>
<td>30</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Red</td>
<td>20</td>
</tr>
<tr>
<td>-</td>
<td>Pants</td>
<td>XL</td>
<td>Red</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>-</td>
<td></td>
<td></td>
<td></td>
<td>Blue</td>
<td>3</td>
</tr>
<tr>
<td>-</td>
<td></td>
<td></td>
<td></td>
<td>Gray</td>
<td>20</td>
</tr>
<tr>
<td>C20</td>
<td>Shirt</td>
<td>XL</td>
<td>White</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Red</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Blue</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Gray</td>
<td>0</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

DataCubes

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

```
<table>
<thead>
<tr>
<th>size</th>
<th>color</th>
<th>C/S</th>
<th>M</th>
<th>L</th>
<th>TOT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Red</td>
<td></td>
<td>20</td>
<td>3</td>
<td>5</td>
<td>28</td>
</tr>
<tr>
<td>Blue</td>
<td></td>
<td>3</td>
<td>3</td>
<td>8</td>
<td>14</td>
</tr>
<tr>
<td>Gray</td>
<td></td>
<td>0</td>
<td>0</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>TOT</td>
<td></td>
<td>23</td>
<td>6</td>
<td>18</td>
<td>47</td>
</tr>
</tbody>
</table>
```
DataCubes

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

<table>
<thead>
<tr>
<th>C / S</th>
<th>S</th>
<th>M</th>
<th>L</th>
<th>TOT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Red</td>
<td>20</td>
<td>3</td>
<td>5</td>
<td>28</td>
</tr>
<tr>
<td>Blue</td>
<td>5</td>
<td>3</td>
<td>8</td>
<td>14</td>
</tr>
<tr>
<td>Gray</td>
<td>0</td>
<td>0</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>TOT</td>
<td>23</td>
<td>6</td>
<td>18</td>
<td>47</td>
</tr>
</tbody>
</table>

DataCubes

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

<table>
<thead>
<tr>
<th>C / S</th>
<th>S</th>
<th>M</th>
<th>L</th>
<th>TOT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Red</td>
<td>20</td>
<td>3</td>
<td>5</td>
<td>28</td>
</tr>
<tr>
<td>Blue</td>
<td>3</td>
<td>3</td>
<td>8</td>
<td>14</td>
</tr>
<tr>
<td>Gray</td>
<td>0</td>
<td>0</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>TOT</td>
<td>23</td>
<td>6</td>
<td>18</td>
<td>47</td>
</tr>
</tbody>
</table>

DataCubes

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

<table>
<thead>
<tr>
<th>C / S</th>
<th>S</th>
<th>M</th>
<th>L</th>
<th>TOT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Red</td>
<td>20</td>
<td>3</td>
<td>5</td>
<td>28</td>
</tr>
<tr>
<td>Blue</td>
<td>3</td>
<td>3</td>
<td>8</td>
<td>14</td>
</tr>
<tr>
<td>Gray</td>
<td>0</td>
<td>0</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>TOT</td>
<td>23</td>
<td>6</td>
<td>18</td>
<td>47</td>
</tr>
</tbody>
</table>

DataCubes

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

<table>
<thead>
<tr>
<th>C / S</th>
<th>S</th>
<th>M</th>
<th>L</th>
<th>TOT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Red</td>
<td>20</td>
<td>3</td>
<td>5</td>
<td>28</td>
</tr>
<tr>
<td>Blue</td>
<td>3</td>
<td>3</td>
<td>8</td>
<td>14</td>
</tr>
<tr>
<td>Gray</td>
<td>0</td>
<td>0</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>TOT</td>
<td>23</td>
<td>6</td>
<td>18</td>
<td>47</td>
</tr>
</tbody>
</table>
DataCubes

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

<table>
<thead>
<tr>
<th>C/S</th>
<th>S</th>
<th>M</th>
<th>L</th>
<th>TOT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Red</td>
<td>20</td>
<td>3</td>
<td>5</td>
<td>28</td>
</tr>
<tr>
<td>Blue</td>
<td>3</td>
<td>3</td>
<td>8</td>
<td>14</td>
</tr>
<tr>
<td>Gray</td>
<td>0</td>
<td>0</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>TOT</td>
<td>23</td>
<td>6</td>
<td>18</td>
<td>47</td>
</tr>
</tbody>
</table>

DataCube

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

DataCubes

SQL query to generate DataCube:
• with ‘cube by’ keyword:
  select size, color, count(*)
  from sales
  where p-id = 'shirt'
  cube by size, color

DataCube issues:
Q1: How to store them (and/or materialize portions on demand)
Q2: Which operations to allow
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]

DataCubes

Q1: How to store a dataCube?

<table>
<thead>
<tr>
<th>Color</th>
<th>S</th>
<th>M</th>
<th>L</th>
<th>TOT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Red</td>
<td>20</td>
<td>3</td>
<td>5</td>
<td>28</td>
</tr>
<tr>
<td>Blue</td>
<td>3</td>
<td>3</td>
<td>8</td>
<td>14</td>
</tr>
<tr>
<td>Gray</td>
<td>0</td>
<td>0</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>TOT</td>
<td>23</td>
<td>6</td>
<td>18</td>
<td>47</td>
</tr>
</tbody>
</table>

DataCubes

Q1: How to store a dataCube?

A1: Relational (R-OLAP)

<table>
<thead>
<tr>
<th>Color</th>
<th>Size</th>
<th>count</th>
</tr>
</thead>
<tbody>
<tr>
<td>'all'</td>
<td>'all'</td>
<td>47</td>
</tr>
<tr>
<td>Blue</td>
<td>'all'</td>
<td>14</td>
</tr>
<tr>
<td>Blue</td>
<td>M</td>
<td>3</td>
</tr>
</tbody>
</table>

...
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

Q1: How to store a dataCube
Q2: What operations should we support?
Q2: What operations should we support?

**DataCubes**

**Roll-up**

<table>
<thead>
<tr>
<th>C / S</th>
<th>S</th>
<th>M</th>
<th>L</th>
<th>TOT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Red</td>
<td>20</td>
<td>3</td>
<td>5</td>
<td>28</td>
</tr>
<tr>
<td>Blue</td>
<td>3</td>
<td>3</td>
<td>8</td>
<td>14</td>
</tr>
<tr>
<td>Gray</td>
<td>0</td>
<td>0</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td><strong>TOT</strong></td>
<td>23</td>
<td>6</td>
<td>18</td>
<td>47</td>
</tr>
</tbody>
</table>

**Drill-down**

<table>
<thead>
<tr>
<th>C / S</th>
<th>S</th>
<th>M</th>
<th>L</th>
<th>TOT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Red</td>
<td>20</td>
<td>3</td>
<td>5</td>
<td>28</td>
</tr>
<tr>
<td>Blue</td>
<td>3</td>
<td>3</td>
<td>8</td>
<td>14</td>
</tr>
<tr>
<td>Gray</td>
<td>0</td>
<td>0</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td><strong>TOT</strong></td>
<td>23</td>
<td>6</td>
<td>18</td>
<td>47</td>
</tr>
</tbody>
</table>

**Slice**

<table>
<thead>
<tr>
<th>C / S</th>
<th>S</th>
<th>M</th>
<th>L</th>
<th>TOT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Red</td>
<td>20</td>
<td>3</td>
<td>5</td>
<td>28</td>
</tr>
<tr>
<td>Blue</td>
<td>3</td>
<td>3</td>
<td>8</td>
<td>14</td>
</tr>
<tr>
<td>Gray</td>
<td>0</td>
<td>0</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td><strong>TOT</strong></td>
<td>23</td>
<td>6</td>
<td>18</td>
<td>47</td>
</tr>
</tbody>
</table>
DataCubes

Q2: What operations should we support?

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

<table>
<thead>
<tr>
<th></th>
<th>Red</th>
<th>Blue</th>
<th>Gray</th>
<th>TOT</th>
</tr>
</thead>
<tbody>
<tr>
<td>C/S</td>
<td>20</td>
<td>3</td>
<td>0</td>
<td>23</td>
</tr>
<tr>
<td>S</td>
<td>3</td>
<td>2</td>
<td>5</td>
<td>10</td>
</tr>
<tr>
<td>M</td>
<td>5</td>
<td>3</td>
<td>0</td>
<td>8</td>
</tr>
<tr>
<td>L</td>
<td>5</td>
<td>8</td>
<td>14</td>
<td>27</td>
</tr>
<tr>
<td>TOT</td>
<td>23</td>
<td>6</td>
<td>18</td>
<td>47</td>
</tr>
</tbody>
</table>

D/W - OLAP - Conclusions

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

Outline

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

<table>
<thead>
<tr>
<th>Age</th>
<th>Chol-level</th>
<th>Gender</th>
<th>...</th>
<th>CLASS-ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>30</td>
<td>150</td>
<td>M</td>
<td></td>
<td>+</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>?</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Pictorially, we have

- num. attr#2 (eg., chol-level)
- num. attr#1 (eg., ‘age’)

and we want to label ‘?’

- num. attr#2 (eg., chol-level)
- num. attr#1 (eg., ‘age’)

so we build a decision tree:

- num. attr#2 (eg., chol-level)
- num. attr#1 (eg., ‘age’)

50
Decision trees

- so we build a decision tree:

```
+------------------
<p>| age&lt;50           |
|                  |
| Y                |
|                  |
| chol. &lt;40        |
| Y                |
|                  |
| N                |</p>
<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>
```

Outline

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

Decision trees

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

Tree building

- How?

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

```
+  +  -  -
+  +  -  -
```

```
num. attr#1 (eg., ‘age’)
```
Tree building

• How?
• A: Partition, recursively - pseudocode:
  Particle (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)

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

Q2: how to evaluate a split?
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 - i.e., we need a measure of uniformity:

\[
\begin{array}{c|c|c|c}
+ & + & + \\
+ & + & - \\
+ & - & - \\
- & - & -
\end{array}
\]

entropy: $H(p^+, p^-)$

Any other measure?

'gini' index: $1 - p^+ - p^-$

‘gini’ index: $1 - p^+ - p^-$

(How about multiple labels?)
Tree building

Intuition:
- entropy: #bits to encode the class label
- gini: classification error, if we randomly guess ‘+’ with prob. \( p \)

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

Before split: we need
\[
(n_+ + n_-) \times H(p_+, p_-) = (7+6) \times H(7/13, 6/13)
\]
bits total, to encode all the class labels

After the split we need:
- 0 bits for the first half and
- \((2+6) \times H(2/8, 6/8)\) bits for the second half

Tree pruning

- What for?
**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])

**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))

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

**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

Outline

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

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%

Association rules - idea

In general, for a given rule

\[ I_j, I_k, \ldots, I_m \rightarrow I_x | c \]

‘c’ = ‘confidence’ (how often people buy \( I_x \), given that they have bought \( I_j, \ldots, I_m \))

‘s’ = support: how often people buy \( I_j, \ldots, I_m, I_x \)
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, lx
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?

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
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 \( i \))

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
Association rules - Conclusions

Association rules: a great tool to find patterns
• easy to understand its output
• fine-tuned algorithms exist

Outline
• Problem
• Getting the data: Data Warehouses, DataCubes, OLAP
• Supervised learning: decision trees
  – problem
  – approach
  – scalability enhancements
• Unsupervised learning
  – association rules
  – clustering

Clustering
• Problem:
  – given N points in V dimensions,
  – group them

Clustering
• Problem:
  – given N points in V dimensions,
  – group them
Clustering

- Problem:
  - given N points in V dimensions,
  - group them

- MANY algorithms:
  - K-means, X-means, BIRCH, OPTICS

Easiest to describe: k-means
- User gives # clusters ‘k’
- Start with ‘k’ random seeds
- Assign each point to its nearest seed
- Move seed towards center, and repeat

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: k-means (& BIRCH, CURE, OPTICS)

Reading material

Additional references

- Jiawei Han and Micheline Kamber, *Data Mining*, Morgan Kaufman, 2001, chapters 2.2-2.3, 6.1-6.2, 7.3.5