Outline

Project Update
Data Mining: Answers without Queries
- Patterns and statistics
- Finding frequent item sets
- Classification and regression trees

Project Update
One week left
- My office hours tomorrow 4-6
- Yangzhe’s office hours 7-9
- Wednesday: consultation with Vlad in lieu of recitation

Make sure you have something working by Wednesday!

Project Update
Hand in Monday by 6.
- Email to mds.
- URL of working system
- Zip/Tar file of code
- Suggested tour

Project Update
Useful tool: sessions
HttpSession s = request.getSession();
Object o = s.getAttribute("attribute");
s.setAttribute("another", o);

(Sessions will get lost when server restarts.)

Data Mining

SQL is about answering specific questions
What if you don’t know question to ask?
- What's interesting about this data?
- What's going on here?
- What happens a lot?

Data mining!
Limits of Data Mining

Randomness
– Some things just happen for no reason
– In large data sets, you may see this a lot

Sparse data
– Beware of breaking up data
– The amount of data available decreases exponentially in number of constraints

Human in the Loop

Selecting data to explore
Cleaning data
– Minimizing noise, outliers, discrepancies in format, organizing data into new and better tables
Evaluating results
– Understanding what’s happening
– Explaining it to the boss

Finding frequent item sets

Problem of associations
– What items go together in a table
– Example: market basket
  • What items tended to be bought together?

Factoids

In 75% of transactions a pen and ink are purchased together
In 25% of transactions milk and juice are purchased together…

Finding frequent item sets

Sample table:

<table>
<thead>
<tr>
<th>transact</th>
<th>item</th>
<th>transact</th>
<th>item</th>
</tr>
</thead>
<tbody>
<tr>
<td>111</td>
<td>pen</td>
<td>113</td>
<td>pen</td>
</tr>
<tr>
<td>111</td>
<td>ink</td>
<td>113</td>
<td>milk</td>
</tr>
<tr>
<td>111</td>
<td>milk</td>
<td>114</td>
<td>pen</td>
</tr>
<tr>
<td>111</td>
<td>juice</td>
<td>114</td>
<td>ink</td>
</tr>
<tr>
<td>112</td>
<td>pen</td>
<td>114</td>
<td>juice</td>
</tr>
<tr>
<td>112</td>
<td>ink</td>
<td>114</td>
<td>water</td>
</tr>
</tbody>
</table>
**Definition**

Itemset: a set of items
Support: the fractions of transactions that contain all the items in the itemset
Frequent itemsets: all itemsets whose support exceeds some threshold

**Example**

Frequent itemsets at 70%
- \{pen\}, \{ink\}, \{milk\}, \{pen,ink\}, \{pen,milk\}

**Efficient Algorithm**

Key property
- Every subset of a frequent itemset is also a frequent itemset

**Algorithm step 1**

Identify the frequent itemsets with one item

```
select item from table
group by item
having count(*) > threshold
```

**Algorithm step 2**

Iteratively
Try to build larger frequent itemsets out of the ones you’ve found already
**Algorithm step 2**

For each new frequent itemset $I_k$ with $k$ items
generate all itemsets $I_{k+1}$ with $k+1$ items
Scan all the transactions once
  Check if the new itemsets are frequent
Set $k = k + 1$

---

**Algorithm step 3**

Stop when no new frequent itemsets are identified

---

**Finding frequent item sets**

Sample table:

<table>
<thead>
<tr>
<th>transact</th>
<th>item 1</th>
<th>transact</th>
<th>item 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>111</td>
<td>pen</td>
<td>113</td>
<td>pen</td>
</tr>
<tr>
<td>111</td>
<td>ink</td>
<td>113</td>
<td>milk</td>
</tr>
<tr>
<td>111</td>
<td>milk</td>
<td>114</td>
<td>pen</td>
</tr>
<tr>
<td>111</td>
<td>juice</td>
<td>114</td>
<td>ink</td>
</tr>
<tr>
<td>112</td>
<td>pen</td>
<td>114</td>
<td>juice</td>
</tr>
<tr>
<td>112</td>
<td>ink</td>
<td>114</td>
<td>water</td>
</tr>
</tbody>
</table>

---

**Mining for association rules**

Association rules
- $\text{LHS} \Rightarrow \text{RHS}$
- $\text{LHS}$ is a set of items
- $\text{RHS}$ is a set of items

Example
$\{\text{pen}\} \Rightarrow \{\text{ink}\}$
If a pen is purchased in a xact, it is likely that ink is also purchased in that xact.

---

**Measures**

Support
- Percentage of xacts that have $\text{LHS} \cup \text{RHS}$

Confidence
- Percentage of $\text{LHS}$ xacts that also have $\text{RHS}$
- Support of $\text{(LHS} \cup \text{RHS}) / \text{Support of LHS}$

---

**Finding them**

First, find frequent itemsets
Create possible rules from frequent itemsets
- Keep those with high confidence
Example

{pen, milk}
Support is 75%
{pen} ⇒ {milk}
Confidence is 75%
{milk} ⇒ {pen}
Confidence is 100%

Statistical Perspective

Is L ⇒ R surprising?
– Statistical independence
– Support tells us:
  \[ P(L \land R) \]
  \[ P(L) \]
  \[ P(R) \]
– Not so interesting if
  \[ P(L \land R) = P(L) \times P(R) \]

Correlation and prediction

Want L ⇒ R to be associated with causality
Basic idea of causality:
Even if we intervene to change how value of L is determined
We still get the same correlation with R.

Correlation and Prediction

For example, with \( \{\text{pen}\} \Rightarrow \{\text{ink}\} \)
– If we change why people buy pens, we still want them to buy ink too.
– For example, we can lower the price of pens.

Problem

Things can go together for other reasons

CART

Classification and regression trees
Tree structured rules

Node either makes prediction
– E.g., classify into a particular class
Or looks at a variable/field
– Tests its value
– Discrete fields: test if equals specific case
– Numerical fields: test if > threshold
– Recurse

Example

Insurance info relation
age cartype highrisk
23 sedan false
30 sports false
36 sedan false
25 truck true
30 sedan false
23 truck true
30 truck false
25 sports true
18 sedan false

Example - visualization

<table>
<thead>
<tr>
<th>Age/cartype</th>
<th>sedan</th>
<th>sports</th>
<th>truck</th>
</tr>
</thead>
<tbody>
<tr>
<td>18</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>F</td>
<td>T</td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>T</td>
<td>T</td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>36</td>
<td>F</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Classification tree

Visualization in Space

<table>
<thead>
<tr>
<th>Age/cartype</th>
<th>sedan</th>
<th>sports</th>
<th>truck</th>
</tr>
</thead>
<tbody>
<tr>
<td>18</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>F</td>
<td>T</td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>T</td>
<td>T</td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>36</td>
<td>F</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Top-down greedy algorithm

BuildTree(data D)
Find the best split of D into D1 and D2
BuildTree(D1)
BuildTree(D2)
Visualization in Space

<table>
<thead>
<tr>
<th>Age/cartype</th>
<th>sedan</th>
<th>sports</th>
<th>truck</th>
</tr>
</thead>
<tbody>
<tr>
<td>18</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>23</td>
<td></td>
<td>T</td>
<td></td>
</tr>
<tr>
<td>25</td>
<td></td>
<td></td>
<td>T</td>
</tr>
<tr>
<td>30</td>
<td>F</td>
<td>F</td>
<td></td>
</tr>
<tr>
<td>36</td>
<td>F</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Possible splits

Supporting this with SQL

Attribute-value Class Sets (AVCs)

```
SELECT attribute, class, COUNT(*)
FROM     table
GROUP BY attribute, class
```

Supporting this with SQL

For example

```
SELECT age, highrisk, COUNT(*)
FROM InsuranceInfo
GROUP BY age, highrisk
```

Supporting this with SQL

Gives a “slice” through the database space

<table>
<thead>
<tr>
<th>Age</th>
<th>True</th>
<th>False</th>
</tr>
</thead>
<tbody>
<tr>
<td>18</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>23</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>25</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>30</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>36</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

Supporting this with SQL

Enough to find candidate split values
And determine how pure each set is
**Algorithm with SQL support**

BuildTree(data D)
- Scan the data and construct AVC group
- Use AVC group to split into D1 and D2
  - BuildTree(D1)
  - BuildTree(D2)

**CART and Statistics**

Sparse data
Overfitting