15-826: Multimedia Databases and Data Mining

Lecture #30: Data Mining - assoc. rules

C. Faloutsos

Outline

Goal: ‘Find similar / interesting things’
• Intro to DB
• Indexing - similarity search
• Data Mining
  – …
  – Association Rules

Must-read Material

• Rakesh Agrawal, Tomasz Imielinski and Arun Swami Mining Association Rules Between Sets of Items in Large Databases Proc. ACM SIGMOD, Washington, DC, May 1993, pp. 207-216

Association rules - outline

• Main idea [Agrawal+SIGMOD93]
  • performance improvements
  • Variations / Applications
  • Follow-up concepts
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%

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

In general, for a given rule
  \[ I_j, I_k, \ldots, I_m \rightarrow I_x | c \]
  ‘c’ = ‘confidence’ (how often people by Ix, given that they have bought Ij, ... Im
  ‘s’ = support: how often people buy Ij, ... Im, Ix

Closely related concept: “large itemset”
  \[ I_j, I_k, \ldots, I_m, I_x \]
  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

100
200
2

first pass

min-sup:10

---

Association rules - idea

A, B
A, C
B, C

first pass

min-sup:10
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 \( i \))

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
(see example 6.1 in [Han+Kamber])

Association rules - outline

• Main idea [Agrawal+SIGMOD93]
  performance improvements
• Variations / Applications
• Follow-up concepts

Association rules - improvements

• Use the independence assumption, to second-guess large itemsets a few steps ahead
• eliminate ‘market baskets’, that don’t contain any more large itemsets
• Partitioning (eg., for parallelism): find ‘local large itemsets’, and merge.
• Sampling
• report only ‘maximal large itemsets’ (dfn?)
• FP-tree (seems to be the fastest)
Association rules - improvements

- FP-tree: no candidate itemset generation - only two passes over dataset
- Main idea: build a TRIE in main memory
  Specifically:
  - first pass, to find counts of each item - sort items in decreasing count order
  - second pass: build the TRIE, and update its counts

(eg., let A,B, C, D be the items in frequency order:)

\begin{itemize}
  \item \{\\}\{32\}
  \item A\{10\} 32 records
  \item B\{4\} 10 of them have A
  \item C\{1\} 4 have AB
  \item D\{2\} 2 have AC
  \item has C
\end{itemize}

Association rules - outline

- Main idea [Agrawal+SIGMOD93]
- performance improvements
- Variations / Applications
- Follow-up concepts
Association rules - variations

1) Multi-level rules: given concept hierarchy
   - ‘bread’, ‘milk’, ‘butter’ -> foods;
   - ‘aspirin’, ‘tylenol’ -> pharmacy
   Look for rules across any level of the hierarchy, eg
   - ‘aspirin’ -> foods
   (similarly, rules across dimensions, like ‘product’,
    ‘time’, ‘branch’):
    - ‘bread’, ‘12noon’, ‘PGH-branch’ -> ‘milk’

Association rules - variations

2) Sequential patterns:
   - ‘car’, ‘now’ -> ‘tires’, ‘2 months later’
   Also: given a stream of (time-stamped) events:
   A A B A C A B A C ......
   Find rules like
   B, A -> C
   [Mannila+KDD97]

Association rules - variations

3) Spatial rules, eg:
   - ‘house close to lake’ -> ‘expensive’

Association rules - variations

4) Quantitative rules, eg:
   - ‘age between 20 and 30’, ‘chol. level <150’ ->
   - ‘weight > 150lb’
   I.e., given numerical attributes, how to find rules?
Association rules - variations

4) Quantitative rules
Solution:
- bucketize the (numerical) attributes
- find (binary) rules
- stitch appropriate buckets together:

Association rules - follow-up concepts

Associations rules vs. correlation.
Motivation: if milk, bread
is a ‘large itemset’, does this means that there is a positive correlation between ‘milk’ and ‘bread’ sales?

Association rules - outline

- Main idea [Agrawal+SIGMOD93]
- performance improvements
- Variations / Applications
  - Follow-up concepts

Association rules - follow-up concepts

Associations rules vs. correlation.
Motivation: if milk, bread
is a ‘large itemset’, does this means that there is a positive correlation between ‘milk’ and ‘bread’ sales?

NO!!
‘milk’ and ‘bread’ 
ANTI-correlated, 
yet milk+bread: frequent
Association rules - follow-up concepts

What to do, then?

A: report only pairs of items that are indeed correlated - ie, they pass the Chi-square test
The idea can be extended to 3-, 4- etc itemsets (but becomes more expensive to check)
See [Han+Kamber, §6.5], or [Brin+, SIGMOD97]

Association rules - Conclusions

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