DATA MINING

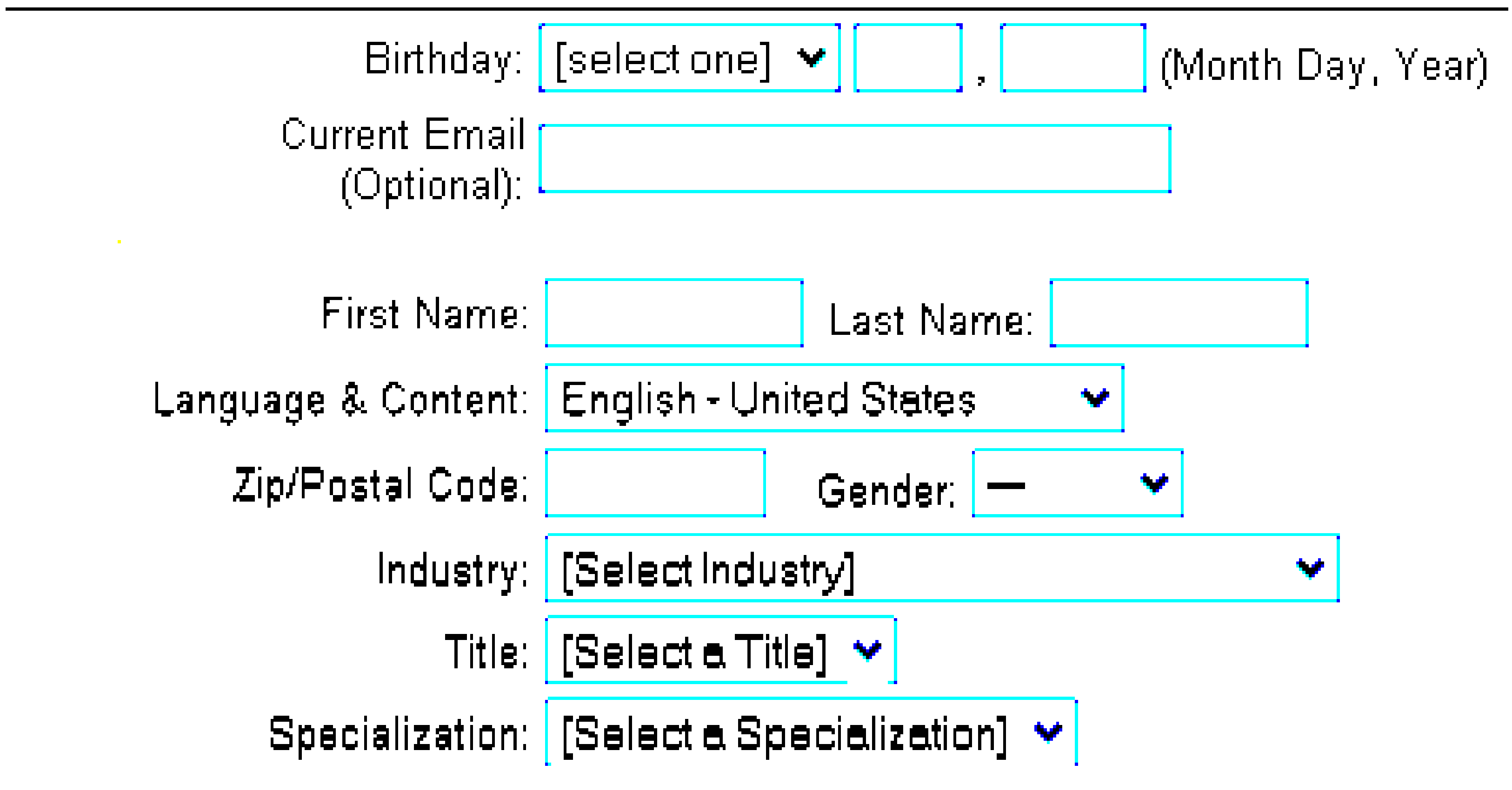
E0 261

Jayant Haritsa

Computer Science and Automation

Indian Institute of Science

A Typical Web-Service Form (e.g. Amazon.com)



Why collect this information?

- Serves as the input to Data Mining Tools
- Data Mining:
  - Hot area of computer science research linking Database Management Systems (DBMS) with AI (machine learning) and Statistics
  - Automated and efficient extraction of "interesting" statistical patterns (or models) from enormous disk-resident archival databases
    - Petabyte (10^{15} bytes) databases are a reality today!
  - *"like prospectors searching for gold in a mine, we are trying to discover nuggets of crucial information from mountains of raw data"*

Interesting Patterns

- Associations: Capture object attribute relationships
  - E.g. If student "state = AP", high likelihood of having taken "Ramaiah coaching classes"
- Clustering: Group similar objects together
  - E.g. Google groups web-pages with similar answers
    - *"In order to show you the most relevant results, we have omitted some entries very similar to the 97 already displayed."*
- Classification: Assign objects to categories
  - E.g. Vehicle insurance categories (low-risk, medium-risk and high-risk) are based on owner's age, vehicle color
- Deviations: Detect "abnormal" behavior
  - E.g. Doping in Olympics ! Match-fixing !

Data Mining: Applications

- Business Strategy (marketing, advertising,….)
- Fraud Detection (credit card, telephone)
- Gene and Protein Sequencing
- Medical Prognosis and Diagnosis
- Sports ! (NBA stats - IBM Advanced Scout)
- ....

Among top 5 technologies of the decade [Gartner]

Benefits of Data Mining

- Better business strategies

  "*Action movies* released in *July* usually succeed at the box office"

- Improved customer services

  [amazon.com]:
  "People who bought *Macbeth* were also inclined to read *The Count of Monte Cristo*"

DATA MINING vs. DBMS (1)

List the names of employees who retired in 1996

Search process

QUERYING

People who buy milk and coffee usually buy sugar as well

Discovery process

QUARRYING

DATABASE

Data Mining is the study of efficient techniques for quarrying

Data Mining vs. DBMS (2)

Operational Data Processing

↓

Historical Data Processing

Association Rules

Association Rules

- Co-occurence of events:
  - On supermarket purchases, indicates which items are typically bought together

80 percent of customers purchasing coffee also purchased milk. $Coffee \Rightarrow Milk \ \ (0.8)$

To ensure statistical significance, need to also compute the "support" – coffee and milk are purchased together by 60 percent of customers. $Coffee \Rightarrow Milk \ \ (0.8, 0.6)$

Problem Formulation

- Given

Coffee	Sugar	Milk	Bread
Y	N	Y	N
N	Y	Y	Y
Y	N	Y	Y
Y	Y	Y	Y
...	...	...	...

Find all rules $X \Rightarrow Y$ (c,s) where c > min_confidence
s > min_support

$c = P(Y/X)$ $s = P(X \cup Y)$ X and Y are disjoint

Problem Breakup

- 1) Find all itemsets *I* such that the support of *I* is greater than minimum support specified by user

  example: min_support = 60%
    coffee(75%), milk(100%), bread (75%)
    coffee-milk (75%)

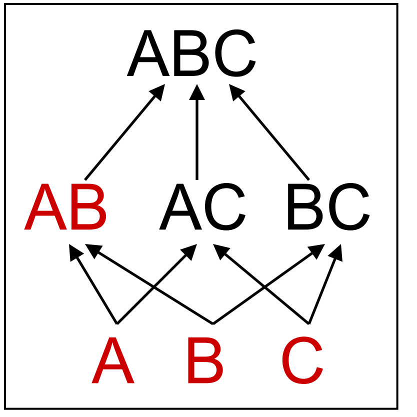
  – Called “large” or “frequent” itemsets
  – Hard to compute

Problem Breakup (contd)

- 2) Use the frequent itemsets to generate rules
  - simple to compute
  - for every frequent itemset f, find all subsets of f
  - for every subset s, output rule $s \rightarrow (f - s)$ if support (f) / support (s) > min_conf

Simple Solution

- The set of all itemsets is a lattice.

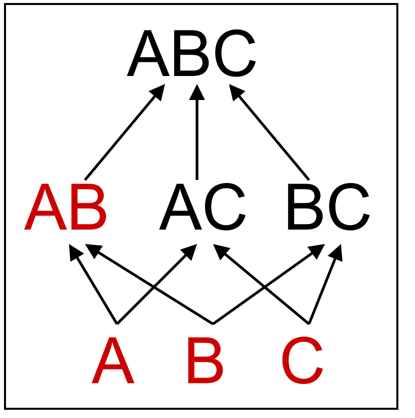


- Do one scan of the database, incrementing the count for each itemset present in each tuple.
- Problem:
  – Number of counters is 2^m ($m = |\mathcal{I}|$)
    - m can be in thousands (KDD cup data had 150000 !)
  – wasteful since most itemsets will be infrequent

Apriori Algorithm

Procedure

- Multiple scans over the complete database
- In scan i,
  - Read database row by row
  - Count occurrences of "candidate" itemsets of length i (counters stored in special data structure)
  - At end of scan, determine the frequent itemsets of length i : F_i
  - Determine "candidate" itemsets (length i+1) for the next scan



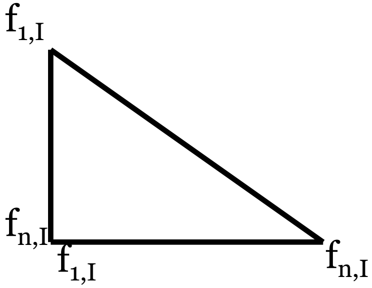
- Return $\cup F_i$

First Pass

- Read database row by row
- Count occurrences of <u>all</u> individual items (called "1-itemsets")
  - counters stored in a 1-D array
- At end of scan, determine F_1
- Determine "candidate" itemsets (length 2) for the next scan
  - $F_1 * F_1$

Second Pass

- Read database row by row
- Count occurrences of candidate 2-itemsets
  - counters stored in a lower 2-D triangular array



- At end of scan, determine F_2
- Determine "candidate" itemsets (length 3) for the next scan
  - AprioriGen

AprioriGen

- To generate candidates C_k
  - Join $F_{k-1} * F_{k-1}$

    insert into C_k

    Select $p.i_1, p.i_2, ..., p.i_{k-1}, q.i_{k-1}$

    from F_{k-1} p, F_{k-1} q

    where $p.i_1 = q.i_1, ..., p.i_{k-2} = q.i_{k-2}$ and $p.i_{k-1} < q.i_{k-1}$
  - Prune

    for all itemsets $c \in C_k$ do

    for all $(k-1)$-subsets s of c do

    if $s \notin F_{k-1}$ then

    delete c from C_k

Example

- F_2 = AB, AC, CD, CE, DE

  Join: ABC, CDE

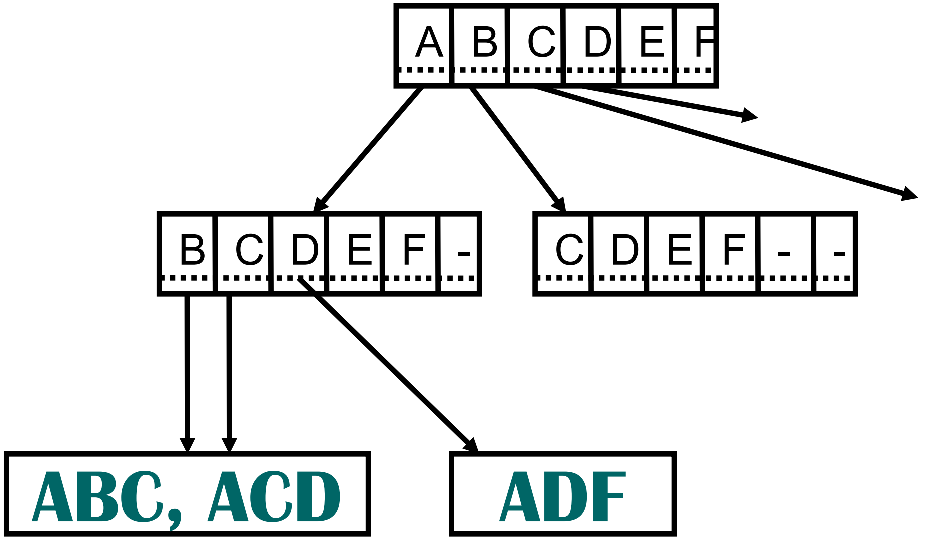
  Prune: Remove ABC since BC is not in F_2

Monotonicity Property

- Any subset of a frequent itemset must itself be frequent.
  - basic foundation of all association rule mining algorithms
  - implies build frequent itemsets bottom-up

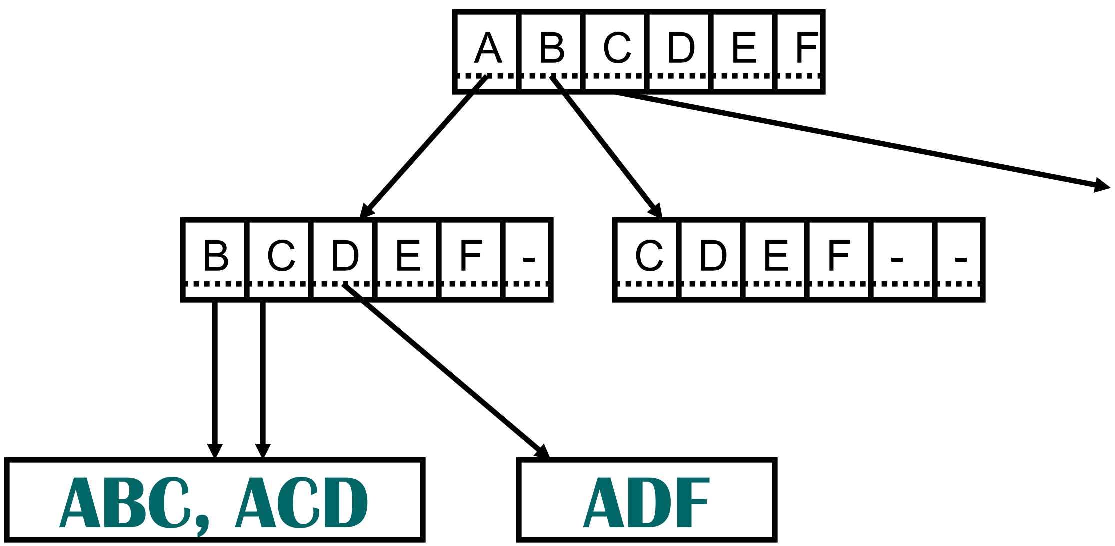
Third (and following) passes

- Same as before except that candidate itemsets are stored in a hash-tree.
- Non-leaf nodes contain a hash table, with each entry in the bucket pointing to another node at next level.
- Leaf nodes contain a list of itemsets.
- In Pass k, the maximum height of the tree is equal to k .



Hash-Tree

- Transaction ACDF in Pass 3:
  - Subsets to be checked are ACD, ACF, ADF, CDF



Subset Search

- To check all candidate subsets of transaction T:
  - if at leaf, find which itemsets there are in T
  - if at an internal node that has been reached by hashing on item i, hash on each item after i in turn, and continue recursively for each such successor.

Summary

- Apriori is considered the "classical" association rule mining algorithm
- Used in IBM's IntelliMiner product
- Number of passes proportional to longest frequent itemset
- Works for sparse matrices

END DATA MINING

E0 261