Knowledge Discovery in Databases II

Winter Term 2015/2016

Optional Lecture:
Pattern Mining & High-D Data Mining

Lectures : Prof. Dr. Peer Kröger, Yifeng Lu
Tutorials: Yifeng Lu

http://www.dbs.ifi.lmu.de/cms/Knowledge_Discovery_in_Databases_II_(KDD_II)

Outline

- Frequent Itemset Mining
  - **Recap**
  - Relationship with subspace clustering
- Rare pattern mining
  - Relationship with subspace outlier detection

- Sequential Pattern Mining
  - Recap
  - Relationship with high dimensional data mining

Recap: Frequent Itemset Mining (KDD 1)

Frequent Itemset Mining: Finding frequent patterns, associations, correlations, or causal structures among sets of items or objects in transaction databases, relational databases, and other information repositories.

- Given:
  - A set of items $I = \{i_1, i_2, \dots, i_m\}$
  - A database of transactions D, where a transaction $T \subseteq I$ is a set of items
- Task 1: find all subsets of items that occur together in many transactions.
  - E.g.: 85% of transactions contain the itemset {milk, bread, butter}
- Task 2: find all rules that correlate the presence of one set of items with that of another set of items in the transaction database.
  - E.g.: 98% of people buying tires and auto accessories also get automotive service done
- Applications: Basket data analysis, cross-marketing, recommendation systems, etc.

Recap: Frequent Itemset Mining (KDD1)

- Transaction database

  D= {{butter, bread, milk, sugar};
  {butter, flour, milk, sugar};
  {butter, eggs, milk, salt};
  {eggs};
  {butter, flour, milk, salt, sugar}}

  NOTE: no quantity

- Question of interest:
  - Which items are bought together frequently?
- Applications
  - Improved store layout
  - Cross marketing
  - Focused attached mailings / add-on sales
  - ** ⇒ Maintenance Agreement*
    (What the store should do to boost Maintenance Agreement sales)
  - *Home Electronics ⇒ ** (What other products should the store stock up?)

items	frequency
{butter}	4
{milk}	4
{butter, milk}	4
{sugar}	3
{butter, sugar}	3
{milk, sugar}	3
{butter, milk, sugar}	3
{eggs}	2
…	

Recap: Naïve Algorithm - BFS

- Naïve Algorithm
  - count the frequency of all possible subsets of I in the database

  → *too expensive* since there are 2^m such itemsets for $|I| = m$ items

- The *Apriori* principle (anti-monotonicity):

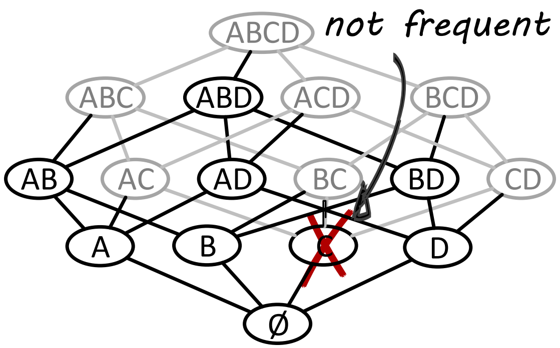
  Any non-empty subset of a frequent itemset is frequent, too!

  $\mathrm{A} \subseteq \mathrm{I}$ with $\mathrm{support(A)} \geq \mathrm{minSup} \Rightarrow \forall \mathrm{A}' \subset \mathrm{A} \wedge \mathrm{A}' \neq \emptyset: \mathrm{support(A')} \geq \mathrm{minSup}$

  Any superset of a non-frequent itemset is non-frequent, too!

  $\mathrm{A} \subseteq \mathrm{I}$ with $\mathrm{support(A)} < \mathrm{minSup} \Rightarrow \forall \mathrm{A}' \supset \mathrm{A}: \mathrm{support(A')} < \mathrm{minSup}$

- Method based on the apriori principle
  - First count the 1-itemsets, then the 2-itemsets, then the 3-itemsets, and so on
  - When counting (*k*+1)-itemsets, only consider those (*k*+1)-itemsets where all subsets of length *k* have been determined as frequent in the previous step



Recap: Naïve Algorithm - BFS

database D

TID	items
100	1 3 4 6
200	2 3 5
300	1 2 3 5
400	1 5 6

scan D

C_1

itemset	count
{1}	3
{2}	2
{3}	3
{4}	1
{5}	3
{6}	2

itemset	count
{1}	3
{2}	2
{3}	3
{5}	3
{6}	2

$L_1 \bowtie L_1$

C_2

itemset
{1 2}
{1 3}
{1 5}
{1 6}
{2 3}
{2 5}
{2 6}
{3 5}
{3 6}
{5 6}

prune C_1

C_2

itemset
{1 2}
{1 3}
{1 5}
{1 6}
{2 3}
{2 5}
{2 6}
{3 5}
{3 6}
{5 6}

scan D

C_2

itemset	count
{1 2}	1
{1 3}	2
{1 5}	2
{1 6}	2
{2 3}	2
{2 5}	2
{2 6}	0
{3 5}	2
{3 6}	1
{5 6}	1

L_2

itemset	count
{1 3}	2
{1 5}	2
{1 6}	2
{2 3}	2
{2 5}	2
{3 5}	2

$L_2 \bowtie L_2$

C_3

itemset
{1 3 5}
{1 3 6}
{1 5 6}
{2 3 5}

prune C_2

C_3

itemset
{1 3 5}
{1 3 6} ✗
{1 5 6} ✗
{2 3 5}

scan D

C_3

itemset	count
{1 3 5}	1
{2 3 5}	2

L_3

itemset	count
{2 3 5}	2

$L_3 \bowtie L_3$

C_4 is empty

Recap: Advanced Algorithm - DFS

- Idea: Divide and Conqure
- Recursively breaking down the problem into sub-problems of the same or related type
  - Breaking down a large database into smaller database
  - Mining frequent pattern on small database
  - Summing up the result

- Consider frequent patterns in previous section:

itemset	count
{1}	3
{2}	2
{3}	3
{5}	3
{6}	2

itemset	count
{1 3}	2
{1 5}	2
{1 6}	2
{2 3}	2
{2 5}	2
{3 5}	2

itemset	count
{2 3 5}	2

Recap: Advanced Algorithm - DFS

- All patterns can be divided into different sets:
  - {Contain 1}, {Contain 2 | no 1}, {Contain 3 | no 1,2}, …
  - i.e. $\{\{1\},\{1\ 3\},\{1\ 5\},\{1\ 6\}\},\{\{2\},\{2\ 3\},\{2\ 5\},\{2\ 3\ 5\}\},\{\{3\},\{3\ 5\}\}$, …

- Same strategy could also be applied on database:
  - Subset contain 1
  - Subset contain 2, no 1
  - Subset contain 3, no 1,2
  - …
- Each subdatabase is responsible for generating a set of frequent patterns
- Combine all frequent patterns will give the full frequent pattern set
  - Could be applied recursively on subset

Recap: Example

- Assume items in each transaction is ordered, e.g.: alphabet order

minSup=0.5

TID	items
100	1 3 4 6
200	2 3 5
300	1 2 3 5
400	1 5 6

- Delete infrequent items

TID	items
100	1 3 6
200	2 3 5
300	1 2 3 5
400	1 5 6

- Generate all single frequent items:
  - {1}, {2}, {3}, {5}, {6}

Recap: Example

- Each frequent item results in a sub-dataset

{1}

TID	items
100	3 6
300	2 3 5
400	5 6

{2}

TID	items
200	3 5
300	3 5

{3}

TID	items
100	6
200	5
300	5

{5}

TID	items
200	{}
300	{}
400	6

{6}

TID	items
100	{}
400	{}

- For each subsets, repeat the process above

{1}

TID	items
100	3 6
300	2 3 5
400	5 6

Delete infrequent →

{1}

TID	items
100	3 6
300	3 5
400	5 6

Frequent items → {3}, {5}, {6} *Sub-subset* → …

Frequent pattern:
{1 3}, {1 5}, {1 6}

Recap: Association Rule Mining

- Question of interest:
  - If milk and sugar are bought, will the customer always buy butter as well? $milk, sugar \Rightarrow butter$?
  - In this case, what would be the probability of buying butter?

- *Association rule:* An association rule is an implication of the form $X \Rightarrow Y$ where $X, Y \subseteq I$ are two itemsets with $X \cap Y = \emptyset$.

- $confidence(X \Rightarrow Y) = P(Y|X) = \frac{|\{T \in D | X \cup Y \subseteq T\}|}{|\{T \in D | X \subseteq T\}|} = \frac{support(X \cup Y)}{support(X)}$
  "conditional probability that a transaction in D containing the itemset X also contains itemset Y"

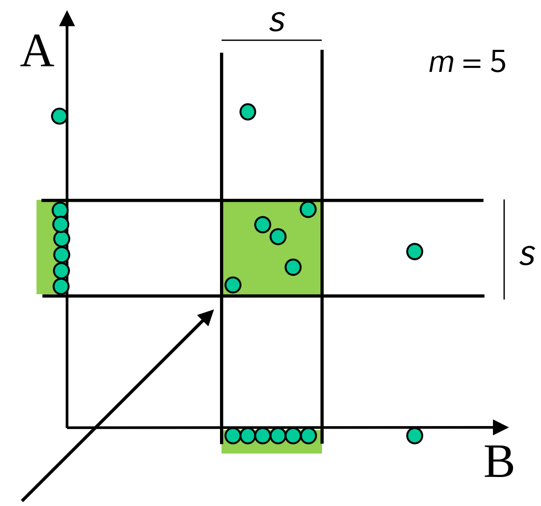
- $corr_{A,B} = \frac{P(A \cup B)}{P(A)P(B)} = \frac{P(B|A)}{P(B)} = \frac{conf(A \Rightarrow B)}{supp(B)} = corr_{B,A}$

Outline

- Frequent Itemset Mining
  - Recap
  - **Relationship with subspace clustering**
- Rare pattern mining
  - Relationship with subspace outlier detection

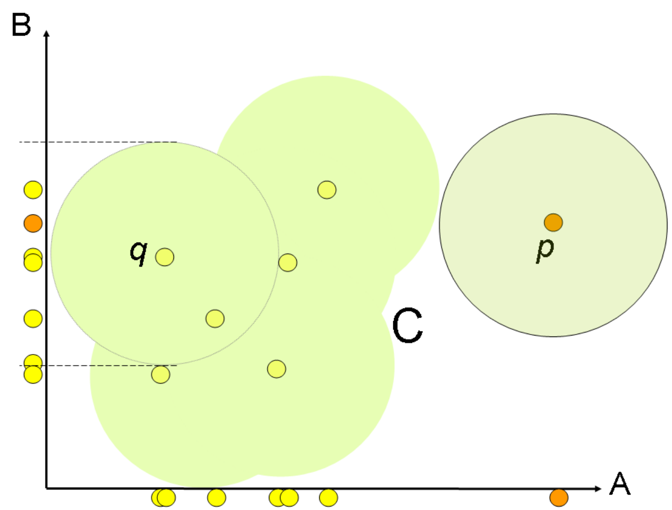
- Sequential Pattern Mining
  - Recap
  - Relationship with high dimensional data mining

FIM and High-D Subspace Clustering



- Find clusters in all subspaces:
  - First: search for subspaces
  - Second: find clusters in the subspace
- Monotonicity Property (Apriori) applied
- Frequent Itemset Mining as High-D Subspace Clustering:
  - Items as entries:

Tid	A	B	C	D
1	1	0	1	1
2	0	1	1	0



  - MinSup as “density threshold”

FIM and High-D Subspace Clustering

- Main steps of subspace clustering in our lecture:
  - Generate all 1-D clusters
  - Generate $(k + 1)$-D clusters form k-D clusters
    - Generate $(k + 1)$-dimensional candidate subspaces *Cand* from S_k
    - Test candidates and generate $(k + 1)$-dimensional clusters
- Breadth First Search in dimensional space
  - Apriori algorithm (Naïve algorithm) in FIM
  - Inefficient with candidate generation step
- Depth First Search based algorithm is possible for subspace clustering

FIM with Numerical Variables

- FIM vs. Subspace Clustering => Binary (Categorical) vs. Numerical
- More advanced FIM: High Utility Itemset Mining

transaction database with quantities

Trans.	items
T_0	a(1), b(5), c(1), d(3), (e,1)
T_1	b(4), c(3), d(3), e(1)
T_2	a(1), c(1), d(1)
T_3	a(2), c(6), e(2)
T_4	b(2), c(2), e(1)

unit profit table

item	unit profit
a	5 $
b	2 $
c	1 $
d	2 $
e	3 $

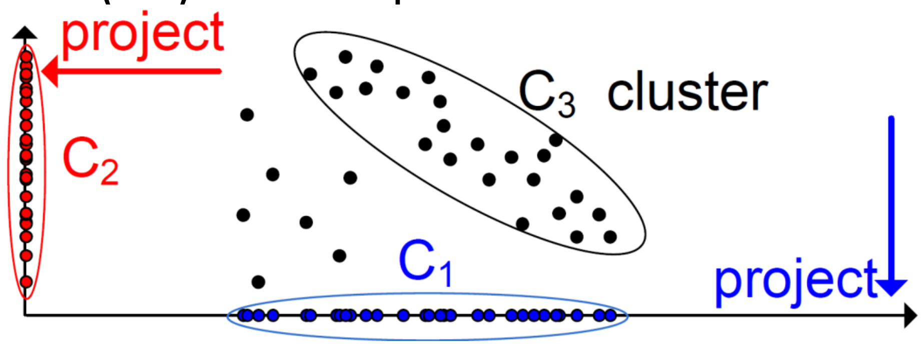
High utility itemsets

{a,c} : 28$
{a,c,e}: 31 $
{a,b,c,d,e}: 25 $
{b,c} : 28 $
{b,c,d}: 34 $
{b,c,d,e}: 40 $
{b,c,e} : 37 $
{b,d} : 30 $
{b,d,e} : 36 $
{b,e} : 31 $
{c, e}: 27$

  - Number of items => Value of each attribute
  - Unit profit => Dimension weight
- High Utility Itemset Mining => Weighted Subspace Clustering?

Association Rule Mining and High-D Subspace Clustering

- Association Rule Mining tells the relationship across dimensions
- Not all frequent itemset but those with high confidence, etc. are more interesting
- Subspace Clustering
  - Clusters in arbitrary subsets of dimensions.
  - Exponential number of possible subspaces.
  - Inefficient: O(2^D) cluster operations



  - High dimensional clusters appear in lower dimensional projections
  - Highly redundant information!

Non-redundant Subspace Clustering

Basic Ideas and Challenges:

- Exclude redundant information (similar clusters)
- How to define redundancy?
- How to use redundancy for pruning?

Overview of approaches:

- INSCY: excludes lower dimensional redundant projections[1]
- RESCU: global optimization to include only relevant clusters[2]
- OSCLU: allows to detect multiple, non-redundant views on the data[3]
- StatPC: includes statistically descriptive clusters[4]

[1]Assent I., Krieger R., Müller E., Seidl T.: INSCY: Indexing Subspace Clusters with In-Process-Removal of Redundancy, ICDM, 2008
[2]Müller E., Assent I., Günnemann S., Krieger R., Seidl T.: Relevant Subspace Clustering: Mining the Most Interesting Non-Redundant Concepts in High Dimensional data, ICDM, 2009
[3]S. Günnemann, E. Müller, I. Färber, and T. Seidl, Detection of Orthogonal Concepts in Subspaces of High Dimensional Data, CIKM, 2009
[4]Moise, G. and Sander, J.: Finding non-redundant, statistically significant regions in high dimensional data: a novel approach to projected and subspace clustering, KDD, 2008

INSCY: Redundancy of Subspace Clusters

Redundancy Definition

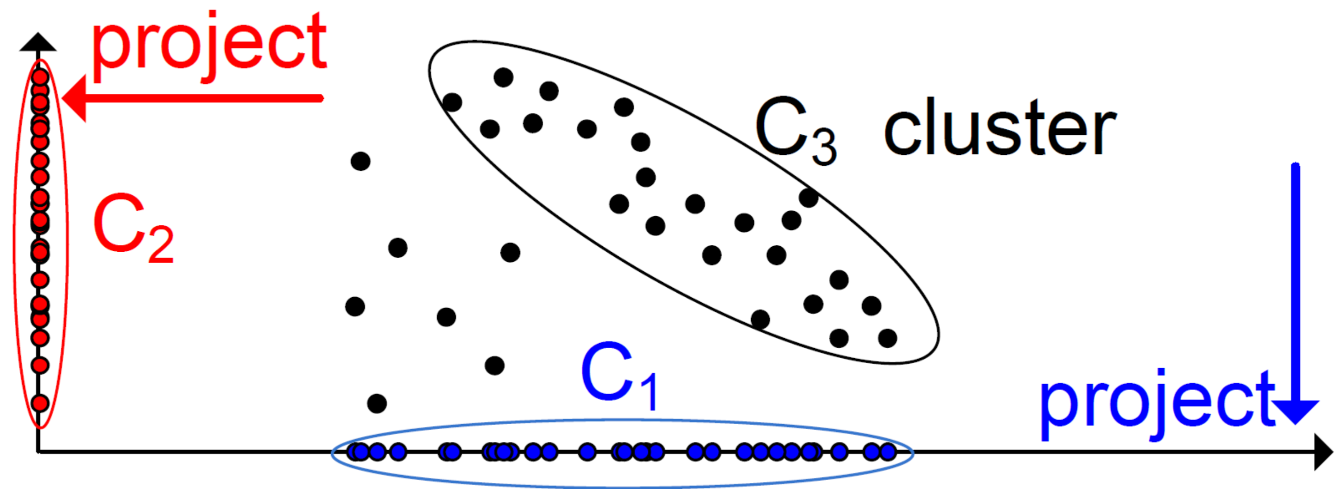
- A cluster $C = (O, S)$ is redundant if

  $\exists C'(O', S'): \quad S' \supset S \quad \wedge \quad O' \subseteq O \quad \wedge \quad |O'| \geq |O| \cdot R$

- The redundant cluster **C** in subspace **S** is covered to a degree of redundancy R by a cluster C' $|O'| \geq R \cdot |O|$ in a higher-dimensional subspace $\boldsymbol{S'} \supset \boldsymbol{S}$

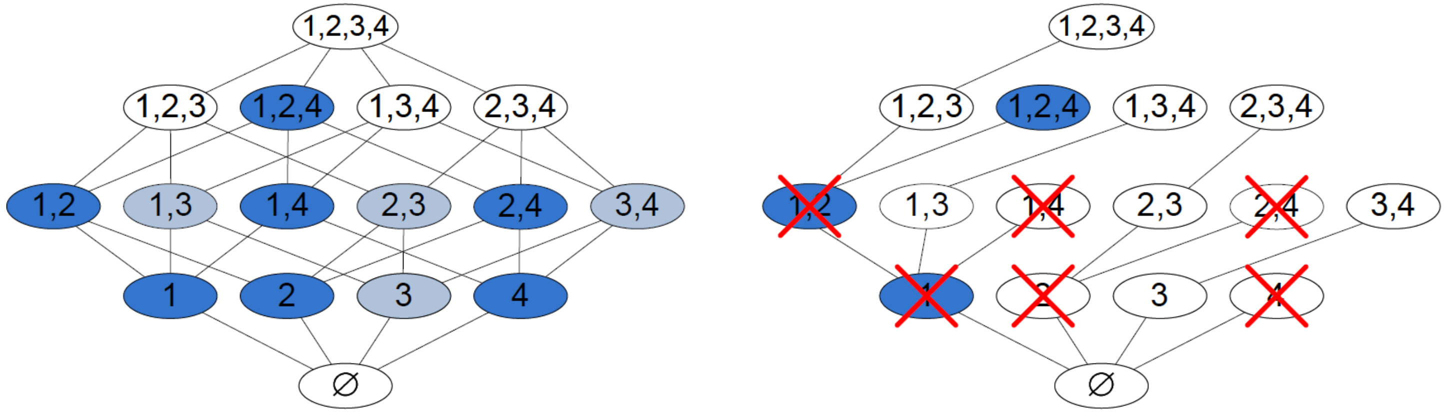
Notice: $R = \frac{|O'|}{|O|}$ => The same as the definition of confidence!

- Higher dimensional clusters are preferred =>



INSCY: Depth First Search

- **Depth-First Processing** enables in-process pruning of redundant clusters.



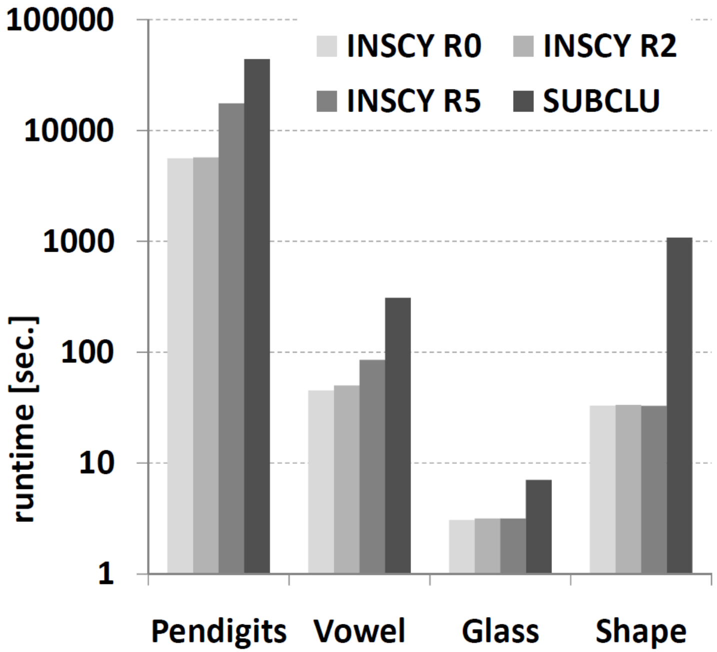
- Lower dimensional projections of clusters can be efficiently pruned.

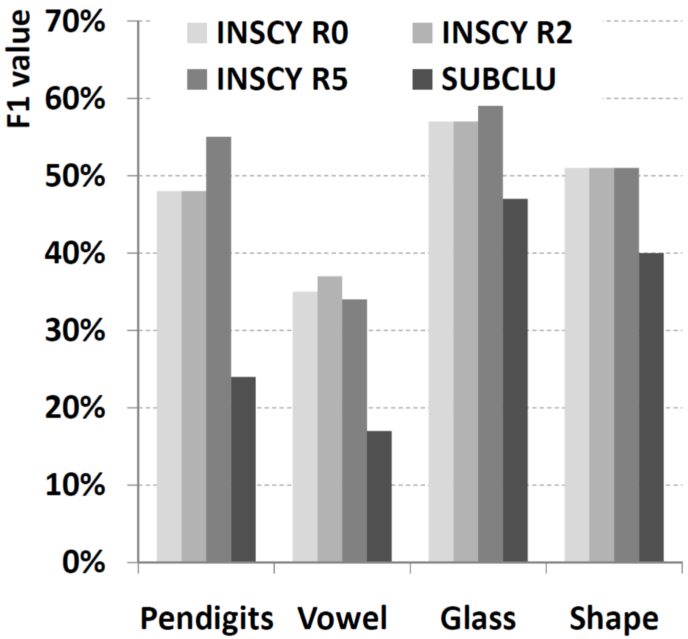
→ Expensive data base scans can be reduced.

- INSCY additionally introduces an index structure to further reduce the number of data base scans

INSCY

- INSCY outperforms SUBCLU in terms of efficiency and accuracy





Summary

- Concepts in FIM have a good mapping to concepts in High-D subspace clustering
  - FIM searches the possible dense subspaces
  - High dimensional clustering do clustering based on the result of FIM

    or

  - FIM is a special case of high dimensional clustering
- Question: What about High-D projection clustering / correlation clustering?

Outline

- Frequent Itemset Mining
  - Recap
  - Relationship with subspace clustering
- **Rare pattern mining**
  - **Relationship with subspace outlier detection**

- Sequential Pattern Mining
  - Recap
  - Relationship with high dimensional data mining

Rare Pattern Mining and Subspace Outlier Detection

- Outlier detection always come together with clustering

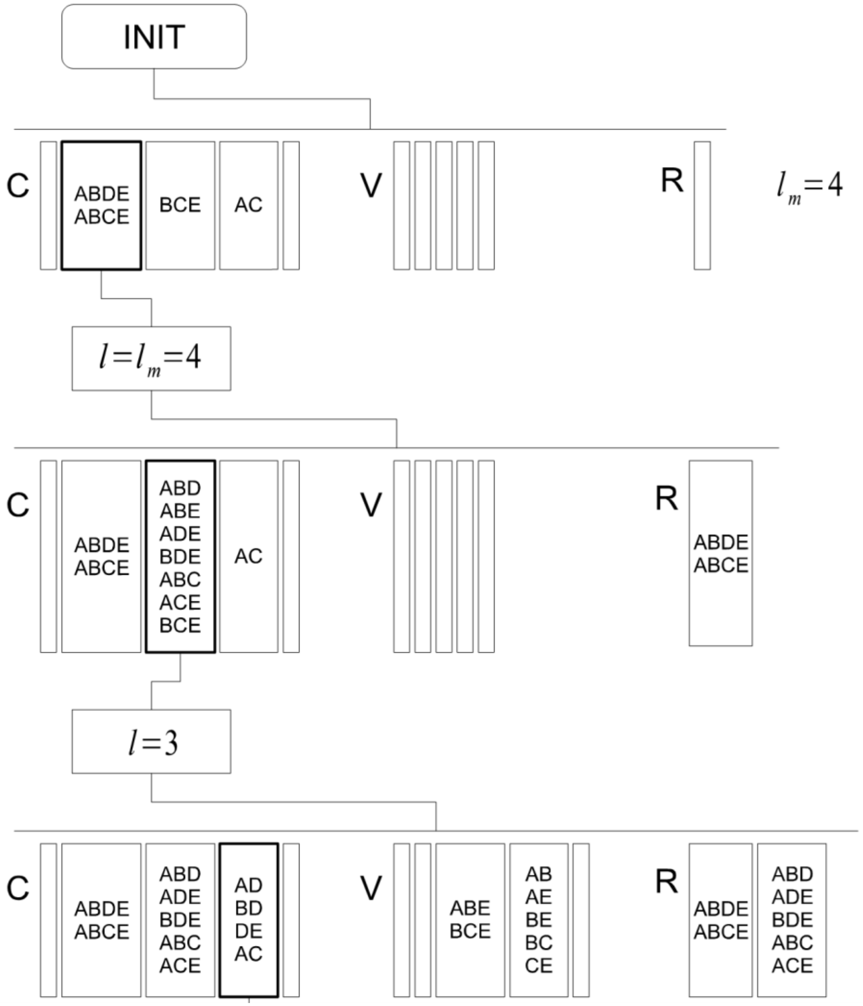
  Frequent Itemset Mining ⇔ High Dimensional **Subspace** Clustering

  Rare Itemset Mining ⇔ High Dimensional **Subspace** Outlier Detection
- As you can image, high dimensional outlier detection also includes two parts:
  - Finding subspaces (Rare Itemset Mining)
  - Finding outliers in subspaces
- Overview of Rare Itemset Mining Approaches:
  - Arima[1]
  - Rarity[2]
  - RP-Tree[3]

[1]Szathmary, L., Napoli, A., & Valtchev, P. (2007). Towards rare itemset mining. In *Proceedings - International Conference on Tools with Artificial Intelligence, ICTAI* (Vol. 1, pp. 305–312). https://doi.org/10.1109/ICTAI.2007.30
[2]Troiano, L., Scibelli, G., & Birtolo, C. (2009). A fast algorithm for mining rare itemsets. In *ISDA 2009 - 9th International Conference on Intelligent Systems Design and Applications* (pp. 1149–1155). https://doi.org/10.1109/ISDA.2009.55
[3]Tsang, Sidney, Yun Sing Koh, and Gillian Dobbie. "RP-Tree: rare pattern tree mining." *International Conference on Data Warehousing and Knowledge Discovery*. Springer Berlin Heidelberg, 2011.

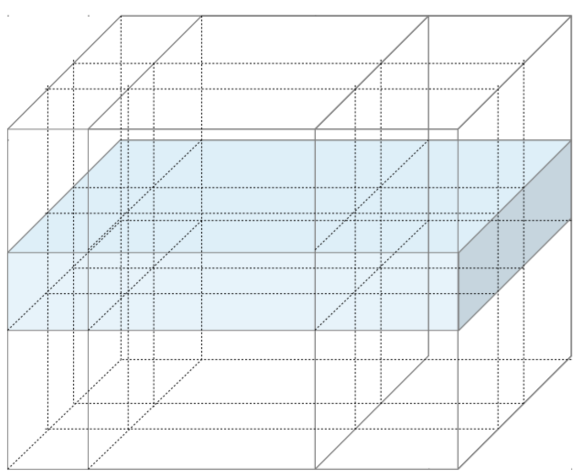
Rarity

- Inverse of Apriori Algorithm ($\leq minSup$)



Subspace Outlier Detection

- First subspace outlier detection algorithm[1] is similar with CLIQUE
  - resembles a grid-based subspace clustering approach but not searching dense but sparse grid cells
  - report objects contained within sparse grid cells as outliers
  - evolutionary search for those grid cells (Apriori-like search not possible, complete search not feasible)



- divide data space in φ equi-depth cells
- each 1-dim. hyper-cuboid contains f = N/φ objects
- expected number of objects in k-dim. hyper-cuboid: $N \cdot f^k$
- standard deviation: $\sqrt{N \cdot f^k (1 - f^k)}$
- "sparse" grid cells: contain unexpectedly few data objects

[1]Aggarwal, Charu C., and Philip S. Yu. "Outlier detection for high dimensional data." ACM Sigmod Record. Vol. 30. No. 2. ACM, 2001.

Summary

- Key words mentioned up to now

  Frequent Itemset Mining ⇔ Subspace Clustering

  Association Rule Mining ⇔ Non-redundant Subspace Clustering

  Rare Pattern Mining ⇔ Subspace Outlier Detection

- More related algorithms can be found in ELKI: http://elki.dbs.ifi.lmu.de/

Outline

- Frequent Itemset Mining
  - Recap
  - Relationship with subspace clustering
- Rare pattern mining
  - Relationship with subspace outlier detection

- **Sequential Pattern Mining**
  - **Recap**
  - **Relationship with high dimensional data mining**

Recap: Frequent Sequential Pattern Mining (KDD1)

DATABASE SYSTEMS GROUP

LMU

- Both can be applied on similar dataset
  - Each customer has a customer id and aligned with transactions.
  - Each transaction has a transaction id and belongs to one customer.
  - Based on the transaction id, each customer also aligned to a transaction sequence.

Cid	Tid	Item
1	1	{butter}
	2	{milk}
	3	{sugar}
2	4	{butter, sugar}
	5	{milk, sugar}
	6	{butter, milk, sugar}
	7	{eggs}
3	8	{sugar}
	9	{butter, milk}
	10	{eggs}
	11	{milk}

Cid	Item
1	{butter} ,{milk}, {sugar}
2	{butter, sugar}, {milk, sugar}, {butter, milk, sugar}, {eggs}
3	{sugar}, {butter, milk}, {eggs}, {milk}

Frequent itemset mining

- No temporal importance in the order of items happening together

items	frequency
{butter}	4
{milk}	5
{butter, milk}	2
…	

sequences	frequency
{butter}	4
{butter, milk}	2
{butter},{milk}	4
{milk},{butter}	1
{butter},{butter,milk}	1
…	

Recap: Sequential Pattern Mining

- Breadth-first search based
  - GSP (*Generalized Sequential Pattern*) algorithm[1]
  - SPADE[2]
  - …
- Depth-first search based
  - PrefixSpan[3]
  - SPAM[4]
  - …

[1]Sirkant & Aggarwal: *Mining sequential patterns: Generalizations and performance improvements.* EDBT 1996
[2]Zaki M J. *SPADE: An efficient algorithm for mining frequent sequences[J].* Machine learning, 2001, 42(1-2): 31-60.
[3]Pei at. al.: *Mining sequential patterns by pattern-growth: PrefixSpan approach.* TKDE 2004
[4]Ayres, Jay, et al: *Sequential pattern mining using a bitmap representation.* SIGKDD 2002.

Recap: PrefixSpan

- The *PrefixSpan* algorithm computes the support for only the individual items in the projected databased D_s
- Then performs recursive projections on the frequent items in a depth-first manner
- Initialization: $D_R \leftarrow D, \boldsymbol{R} \leftarrow \emptyset, \mathcal{F} \leftarrow \emptyset$

- $PrefixSpan(D_R, \boldsymbol{R}, minSup, \mathcal{F})$

  For each $s \in \Sigma$ such that $\sup(s, D_R) \geq minSup$ do
  - $\boldsymbol{R_s} = \boldsymbol{R} + s$ // append s to the end of $\boldsymbol{R}$
  - $\mathcal{F} \leftarrow \mathcal{F} \cup \{(\boldsymbol{R_s}, \sup(s, D_R))\}$ // calculate the support of s for each $\boldsymbol{R_s}$within D_R
  - $D_s \leftarrow \emptyset$ // create projected data for s
  - For each $\boldsymbol{S_i} \in D_R$ do
    - $\boldsymbol{S_i'} \leftarrow$ projection of $\boldsymbol{S_i}$ w.r.t. item s
    - Remove an infrequent symbols from $\boldsymbol{S_i'}$
    - If $\boldsymbol{S_i'} \neq \emptyset \; then \;\; D_s = D_s \cup \boldsymbol{S_i'}$
  - If $D_s \neq \emptyset$ then $PrefixSpan(D_s, \boldsymbol{R_s}, minSup, \mathcal{F})$

Recap: Example

Id	Sequence
S_1	$CAGAAGT$
S_2	$TGACAG$
S_3	$GAAGT$

$D_\emptyset$	
S_1	$CAGAAGT$
S_2	$TGACAG$
S_3	$GAAGT$
$A(3), \cancel{C(2)}, G(3), T(3)$	

D_A	
S_1	$GAAGT$
S_2	AG
S_3	AGT
$A(3), G(3), \cancel{T(2)}$	

D_T	
S_2	$GAAG$
$\cancel{A(1)}, \cancel{G(1)}$	

D_G	
S_1	$AAGT$
S_2	AAG
S_3	$AAGT$
$A(3), G(3), \cancel{T(2)}$	

D_{AA}	
S_1	AG
S_2	G
S_3	G
$\cancel{A(1)}, G(3)$	

D_{AG}	
S_1	AAG
$\cancel{A(1)}, \cancel{G(1)}$	

D_{GA}	
S_1	AG
S_2	AG
S_3	AG
$A(3), G(3)$	

D_{GG}
$\emptyset$

Recap: Example

Id	Sequence
S_1	$CAGAAGT$
S_2	$TGACAG$
S_3	$GAAGT$

D_{A}	
S_1	$GAAGT$
S_2	AG
S_3	AGT
$A(3), G(3), \cancel{T(2)}$	

D_{AA}	
S_1	AG
S_2	G
S_3	G
$\cancel{A(1)}, G(3)$	

D_{AAG}
$\emptyset$

D_{AG}	
S_1	AAG
$\cancel{A(1)}, \cancel{G(1)}$	

D_{G}	
S_1	$AAGT$
S_2	AAG
S_3	$AAGT$
$A(3), G(3), \cancel{T(2)}$	

D_{GA}	
S_1	AG
S_2	AG
S_3	AG
$A(3), G(3)$	

D_{GAA}	
S_1	G
S_2	G
S_3	G
$G(3)$	

D_{GAG}
$\emptyset$

D_{GG}
$\emptyset$

D_{T}	
S_2	$GAAG$
$\cancel{A(1)}, \cancel{G(1)}$	

SPM and High Dimensional Data Mining

- In each SPM, each item can exist multiple times
  - More complicate in high dimensional view: same dimension might happened multiple times
- Sequence with temporal information: trace

$$A \xrightarrow{5.6} B \xrightarrow{2.1} C \qquad [A(1.1), B(6.7), C(8.8)]$$

- Existing algorithms introduce heuristics:
  - No noise or noise will not affect the order of events
  - Thus, SPM like algorithm can be applied to find “subspace” first
  - Then, clustering based on the temporal information