LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Database Systems Group • Prof. Dr. Thomas Seidl

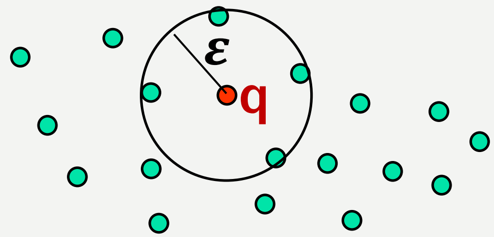
QA-Session

Knowledge Discovery in Databases I

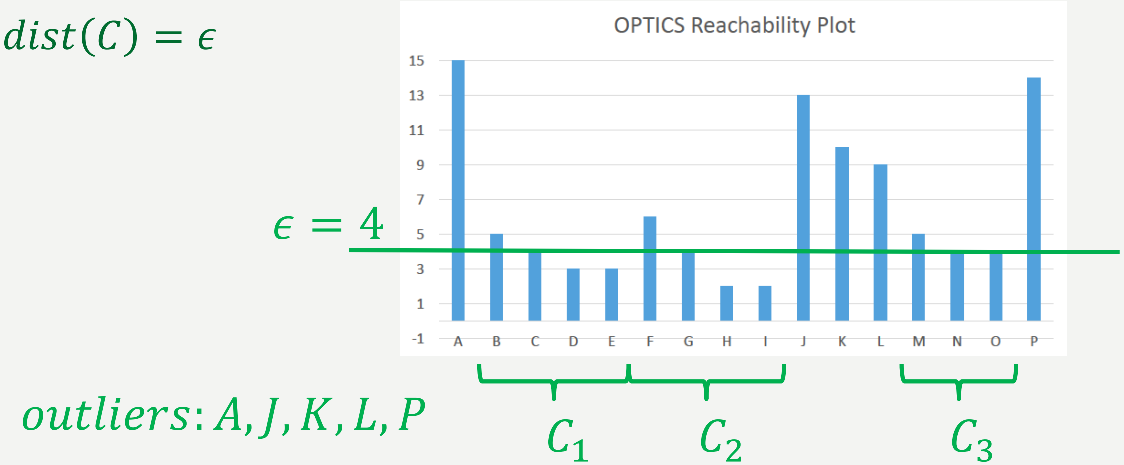
SS 2016

1. Considering the ϵ-neighborhood of an object, in which cases does the object itself count towards the $minPts$?
   - This decision is subject to convention
   - Either you consider the object itself part of the ϵ-neighborhood, or not
   - What is important is consitency, i.e. when executing DBSCAN or OPTICS, commit to one of the two possibilities

$MinPts = 5$ → **q** is a core object?



2. When extracting a clustering from an OPTICS reachability plot, what is the convention regarding cluster affiliation, if a reachability distance corresponds exactly to the threshold?
   - In order to be included into the previous cluster, the rechability distance of the next object needs to be *smaller or equal* than the threshold
   - This ensures, that we extract *density-based* clusters
   - Cf. Exercise 8-2: $r - dist(C) = \epsilon$



3. Soft margin SVM: How do I find the Lagrange multipliers α_i? What is the intuitive meaning behind them?

- The vector $\alpha = (\alpha_1, \dots, \alpha_n)$ is the solution vector of the dual problem
- It can be obtained by solving the dual problem (using a numerical solver)

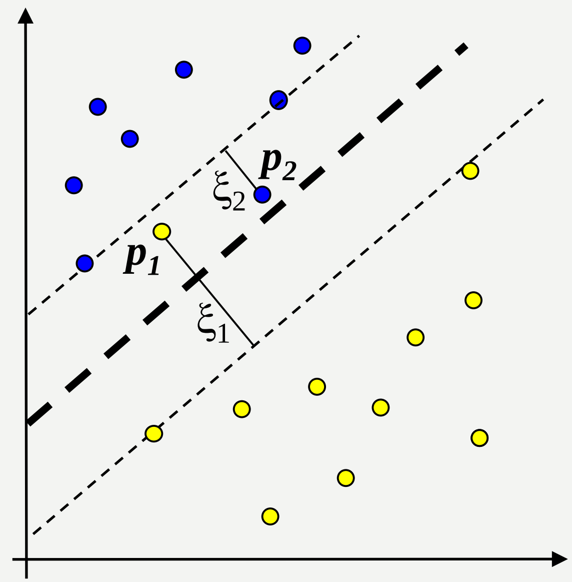
Dual Optimization Problem:

Maximize $L(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \cdot \alpha_j \cdot y_i \cdot y_j \cdot \langle \mathbf{x}_i, \mathbf{x}_j \rangle$

subject to $\sum_{i=1}^{n} \alpha_i \cdot y_i = 0$ and $0 \le \alpha_i \le C$

- Intuition: It can be shown that $\alpha_i > 0$ iff the corresponding vector p_i lies either exactly on the margin, inside the margin, or on the wrong side of the hyperplane
- These vectors are called *support vectors*
- They represent the „difficult" instances of the learning problem

$\alpha_i = 0$: p_i is not a support vector
$\alpha_i = C$: p_i is a support vector with $\xi_i > 0$
$0 < \alpha_i < C$: p_i is a support vector with $\xi_i = 0$



4. Same question for the kernel SVM.

- The same holds for the kernelized soft margin SVM
- Recall: $\phi: \mathcal{X} \rightarrow \mathcal{H}$, $\kappa(x, x') = \langle \phi(x), \phi(x') \rangle$ for all $x, x' \in \mathcal{X}$

Dual Optimization Problem with Lagrange multipliers (Wolfe dual):

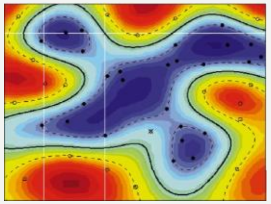
Maximize $L(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \cdot \alpha_j \cdot y_i \cdot y_j \cdot \langle \mathbf{x}_i, \mathbf{x}_j \rangle$

HERE: $\sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \cdot \alpha_j \cdot y_i \cdot y_j \cdot \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle$

subject to $\sum_{i=1}^{n} \alpha_i \cdot y_i = 0$ and $0 \leq \alpha_i \leq C$

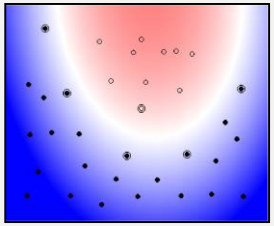
However:

- The maximum margin hyperplane is defined in the feature space $\mathcal{H}$
- In the original space, the decision boundary corresponds to a level set of a linear combination of kernel functions



Decision rule:

$$h(x) = sign\left(\sum_{x_i \in SV} \alpha_i \cdot y_i \cdot K(\mathbf{x}_i, \mathbf{x}) + b\right)$$



- Again, the support vectors are those x_i, for which $\alpha_i > 0$
- The decision boundary depends only on the support vectors

5. How do I calculate the normal vector w and offset parameter b of the maximum margin hyperplane?

- In Exercise 10-1, the location of the maximum margin hyperplane was clear
- In general, w and b can be obtained by solving the primal problem
- However, in practice one usually solves the dual problem
- In this case, the parameters can be recovered as

$w = \sum_i \alpha_i y_i x_i$ and

$b = y_j - \sum_i y_i \alpha_i x_i^T x_j$ where x_j is any support vector

Decision rule:

$$h(x) = sign\left(\sum_{x_i \in SV} \alpha_i \cdot y_i \cdot \langle \mathbf{x}_i, \mathbf{x} \rangle + b\right)$$

- Note: The solution depends only on the support vectors! -> Efficiency

6. FP-Growth example.

- Database: (a,g), (b,c,g), (e,g,), (d,g,), (d,f,g), (d,g), (a,g), (a,g), (a,e), (a,g), (a,f,h), (a,f), (a,d), (d,f,g)
- minSup = 10% -> An itemset needs to appear in at least 2 transactions to be considered frequent

Frequent itemset mining with FP-Growth:

- Count frequencies of single items: a:8, b:1, c:1, d:5, e:2, f:4, g:10, h:1
- Drop infrequent items, sort frequent items in the order of descending support (header table)

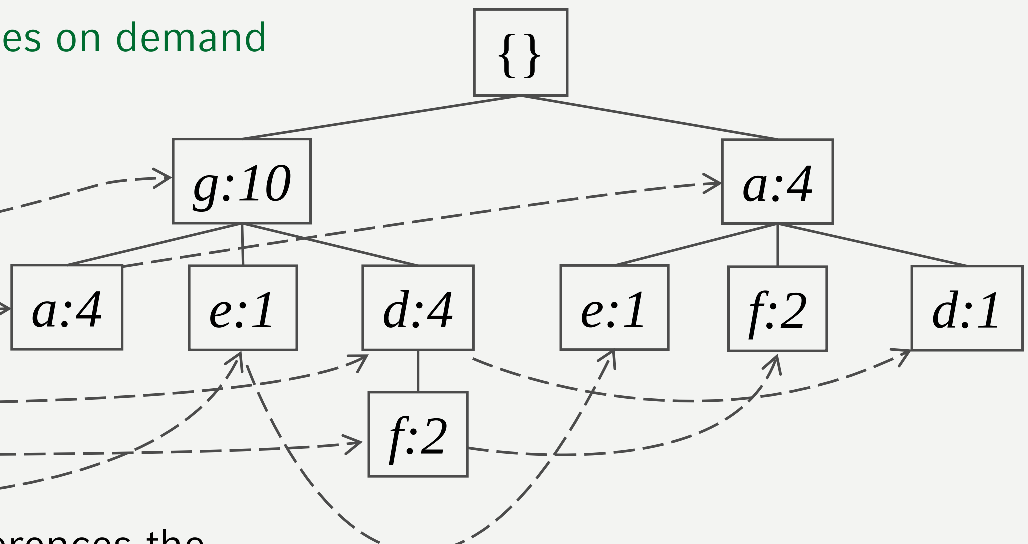
item	frequency
g	10
a	8
d	5
f	4
e	2

FP-Growth

- Sort items within transactions in descending order of their frequencies:
  (g,a)x4, (g), (g,e), (g,d)x2, (g,d,f)x2, (a,e), (a,f)x2, (a,d)
- Construct initial FP-tree:
  - For each transaction add a path from the root in sorted order
  - Increment frequency of nodes along the path
  - Add new nodes on demand

header table:

item	frequency
g	10
a	8
d	5
f	4
e	2

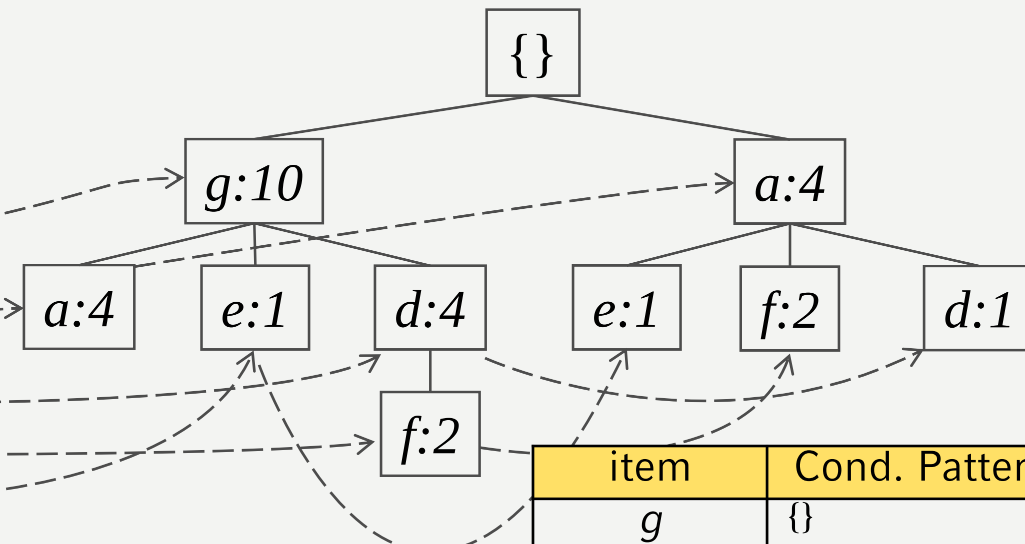


header table references the occurrences of the frequent items in the FP-tree

- Construct conditional pattern base:
  - Iterate over all items in the header table
  - For each item, follow the links and find all prefix path with counts written in the corresponding item nodes

header table:

item	frequency
g	10
a	8
d	5
f	4
e	2

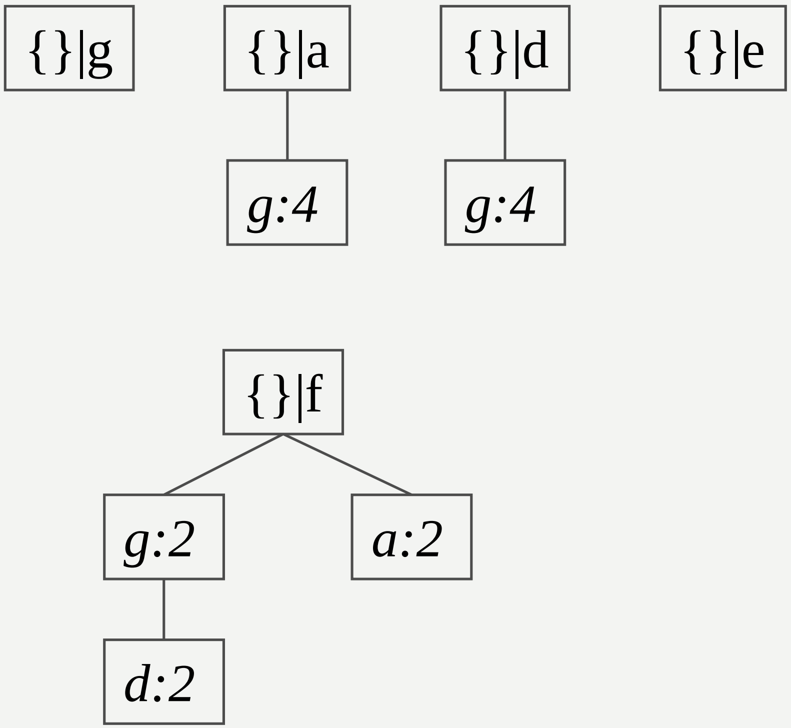


item	Cond. Pattern base
g	{}
a	g:4
d	g:4, a:1
f	gd:2, a:2
e	g:1, a:1

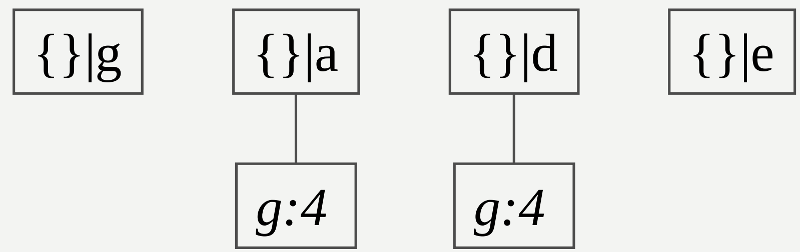
FP-Growth

- Interpret each conditional pattern base as a set of transactions:
  - Drop infrequent items, keep the order sorted
  - Construct conditional FP-tree for each item just as before

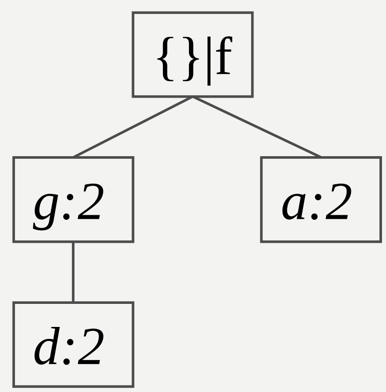
item	Cond. Pattern base
g	{}
a	g:4
d	g:4, a:1
f	gd:2, a:2
e	g:1, a:1



- Recursively mine conditional FP-trees for frequent itemsets:
  - If the tree contains only a single path, simply enumerate all the itemsets (enumerate all combinations of sub-paths)

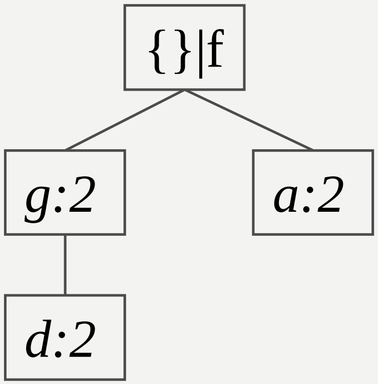


Frequent itemsets: (g) (a), (g,a) (d), (g,d) (e)



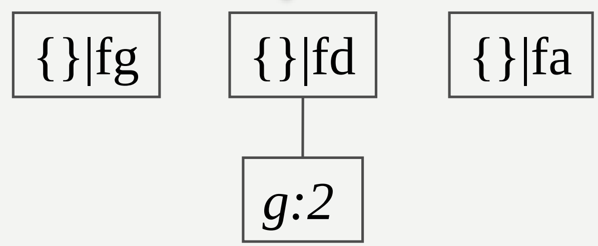
Recursive step

- Recursively mine the conditional FP-tree for f:



conditional pattern base for f:

item	Cond. Pattern base
g	{}
d	g:2
a	{}



Frequent itemsets: (f) (f,g) (f,d), (f,d,g) (f,a)

Final set of frequent itemsets: (g), (a), (a,g), (d), (d,g), (e), (f), (fg), (f,d), (f,d,g), (f,a)

Exam

- Time: 27.07.2016, 16:00-18:00
- Place: LMU main building B201 and A240
- Exam questions will be stated in English, answers may be given in either English or German
- Except for some selected topics (see next slide), all contents discussed in the lecture and exercises are potentially relevant for the exam
- There will be no code in the exam, you don‘t need any Python skills
- No further resources (e.g. calculators) will be allowed
- There will be no second exam
- News will be posted on the course website!

The following topics do *not* need to be prepared for the exam. They are still important however.

- In Frequent Itemset Mining Chapter:
  - Hierarchical/Quantitative Association Rules
- In Sequential Pattern Mining Chapter:
  - Process Mining
- In Clustering Chapter:
  - EM formulas
  - Further topics
- Further topics presented in the last lecture