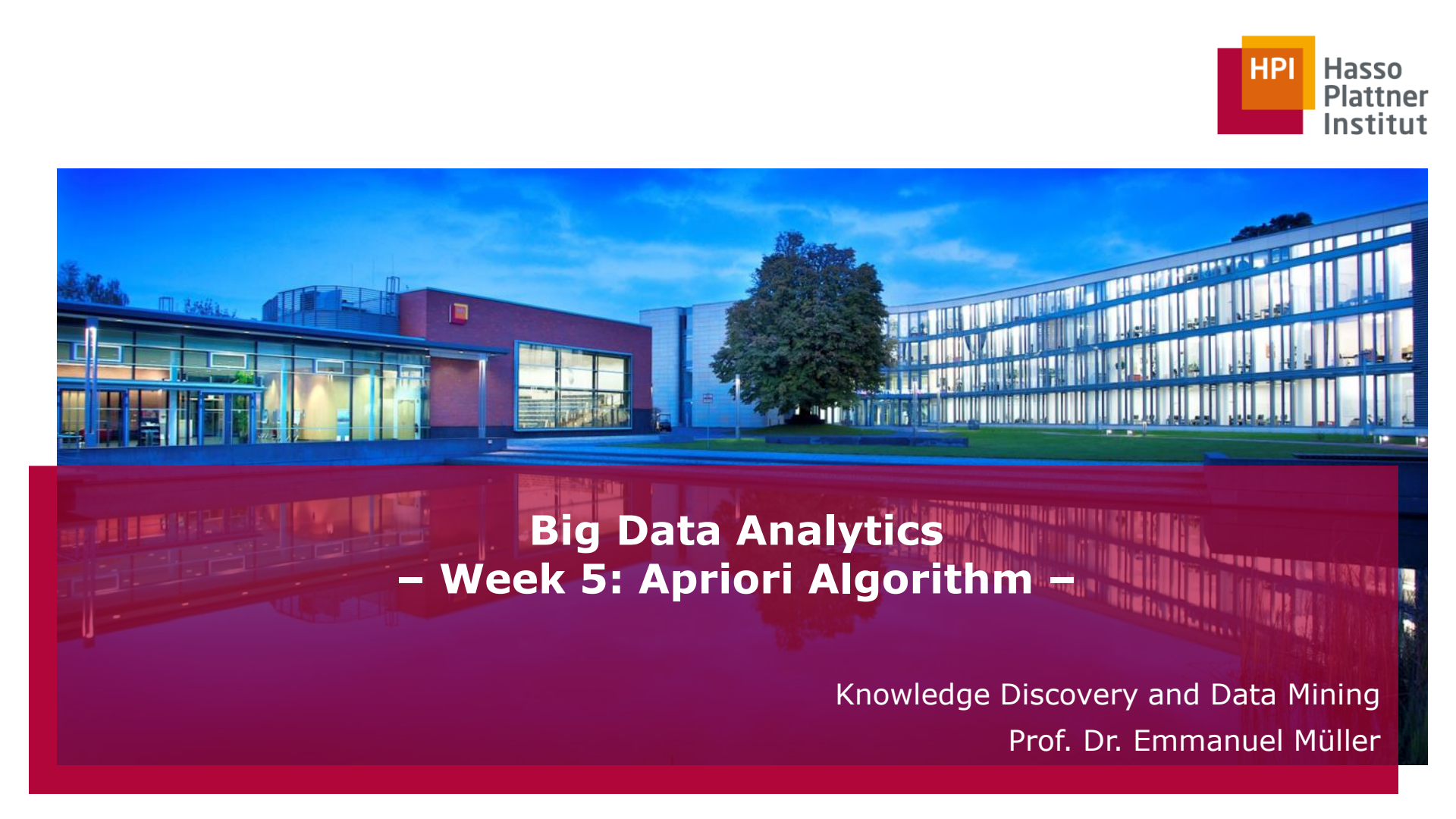




Big Data Analytics – Week 5: Apriori Algorithm –

Knowledge Discovery and Data Mining
Prof. Dr. Emmanuel Müller

The Apriori Algorithm

```

variable  $C_k$ : candidate itemsets of size  $k$ 
variable  $L_k$ : frequent itemsets of size  $k$ 
 $L_1 = \{\text{frequent items}\}$ 
for ( $k = 1$ ;  $L_k \neq \emptyset$ ;  $k++$ ) do begin
    produce candidates {
        // JOIN STEP: join  $L_k$  with itself to produce  $C_{k+1}$ 
        // PRUNE STEP: discard  $(k+1)$ -itemsets from  $C_{k+1}$ 
        // that contain non-frequent  $k$ -itemsets as subsets
         $C_{k+1} = \text{candidates generated from } L_k$ 
        for each transaction  $t$  in database do
            prove candidates {
                Increment the count of all candidates in  $C_{k+1}$ 
                that are contained in  $t$ 
                 $L_{k+1} = \text{candidates in } C_{k+1} \text{ with min\_support}$ 
            }
    }
return  $\cup_k L_k$ 

```

Generating Candidates (Prune Step)

Step 2: Pruning: $L_{k+1} = \{X \in C_{k+1} \mid \text{support}(X) \geq \text{minSup}\}$

- Naïve: Check support of every itemset in $C_{k+1} \rightarrow$ inefficient for huge C_{k+1}
- Instead, apply Apriori principle first:
Remove candidate $(k+1)$ -itemsets which contain a non-frequent k -subset s , i.e., $s \notin L_k$

```
forall itemsets  $c$  in  $C_{k+1}$  do  
    forall  $k$ -subsets  $s$  of  $c$  do  
        if ( $s$  is not in  $L_k$ ) then delete  $c$  from  $C_{k+1}$ 
```

Example 1

- $L_3 = \{(ACF), (ACG), (AFG), (AFH), (CFG)\}$
- Candidates after the join step: $\{(ACFG), (AFGH)\}$
- In the pruning step: delete (AFGH) because $(FGH) \notin L_3$,
i.e., (FGH) is not a frequent 3-itemset; **also** $(AGH) \notin L_3$
 $\rightarrow C_4 = \{(ACFG)\} \rightarrow$ check the support to generate L_4

Generating Candidates (Join Step)

Requirements for set of all candidate $(k+1)$ -itemsets C_{k+1}

- **Completeness**: Must contain all frequent $(k+1)$ -itemsets (superset property $C_{k+1} \supseteq L_{k+1}$)
- **Selectiveness**: Significantly smaller than the set of all $(k+1)$ -subsets

Step 1: Joining ($C_{k+1} = L_k \bowtie L_k$)

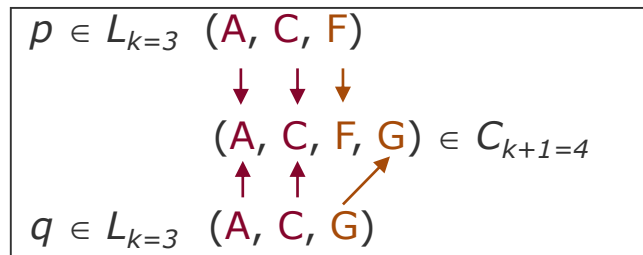
- Suppose the items are sorted by any order (e.g. lexicograph.)
- Consider frequent k -itemsets p and q
- p and q are joined if they share the same first $k-1$ items

insert into C_{k+1}

select $p.i_1, p.i_2, \dots, p.i_{k-1}, p.i_k, q.i_k$

from $L_k : p, L_k : q$

where $p.i_1=q.i_1, \dots, p.i_{k-1}=q.i_{k-1}, p.i_k < q.i_k$



Apriori Algorithm – Full Example

minSup=0.5
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

L ₁	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 C₁

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 C₂

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

$L_3 \bowtie L_3$

C₄ is empty

Is Apriori Fast Enough?

– Performance Bottlenecks –

The core of the Apriori algorithm:

- Use frequent $(k - 1)$ -itemsets to generate **candidate** frequent k -itemsets
- Use database scan and pattern matching to collect counts for the candidate itemsets

The bottleneck of *Apriori*: **candidate generation**

- Huge candidate sets:
 - 10^4 frequent 1-itemsets will generate 10^7 candidate 2-itemsets
 - To discover a frequent pattern of size 100, e.g., $\{a_1, a_2, \dots, a_{100}\}$, one needs to generate $2^{100} \approx 10^{30}$ candidates.
- Multiple scans of database:
 - Needs n or $n+1$ scans, n is the length of the longest pattern

→ Is it possible to mine the complete set of frequent itemsets without candidate generation?