جامعة الطائف
TAIF UNIVERSITY

DATA MINING

Assoc. Prof. Dr. Salha Alzahrani

College of Computers and Information Technology
Taif University
Saudi Arabia

s.zahrani@tu.edu.sa

Prediction | Association Rule Mining- Part: I

Recap of Previous Lecture

Content of This Lecture

Summary & Checklist

Recap of Lecture 7

- Introduction to Association Rule Mining
- Measures of Rule Interestingness
- Basic measures of rules interestingness: Confidence, Support and Completeness
- Example
- Measures of Rule Interestingness: discriminability
- The Piatetsky-Shapiro Criteria and the RI Measure
- Rule Interestingness Measures Applied to the chess Dataset
- Association Rule Mining Tasks

Prediction | Association Rule Mining- Part: II

Recap of Previous Lecture

Content of This Lecture

Summary & Checklist

Content of Lecture 8

- Association Rule Mining- Part: II: Introduction
- Transactions and Itemsets
- Support for an Itemset
- Association Rules
- Generating Association Rules: Apriori Algorithm
- Generating Association Rules: Apriori-gen Algorithm
- Generating association rules using Apriori algorithm: Example
- Handout (7): Generating Association Rules for Market Basked Using Apriori Algorithm
- Summary & Checklist.

- We concern with a special form of **Association Rule Mining (ARM)**, which is known as ***Market Basket Analysis***.
- Here we are interested in any **rules that relate the purchases made by customers in a shop**, frequently a large store with many thousands of products.
- Other applications of the same kind include analysis of **items purchased by credit card**, **patients' medical records**, **crime data** and data from **satellites**.

Prediction | Transactions and Itemsets

- Assume that we have a database comprising ***n transactions*** (i.e. records), each of which is a set of ***items***.
- In **market basket analysis**, we can think of each transaction as corresponding to a group of purchases made by a customer.
- For example ***{milk, cheese, bread}***. So, milk, cheese, bread etc. are **items** and we call ***{milk, cheese, bread}*** an ***itemset***.
- We are interested in finding rules known as ***association rules*** that apply to the purchases made by customers, for example 'buying fish and sugar is often associated with buying milk and cheese', but only want rules that meet certain criteria for '**interestingness**'

Prediction | Transactions and Itemsets

- Assume that there are *m* possible items that can be bought and will use the letter *I* to denote the set of all possible items.
  - Example, if a database comprises 8 transactions (so ***n* = 8**) and there are **only 5 different items**, denoted by *a*, *b*, *c*, *d* and *e*, so we have ***m* = 5** and ***I* = {a, b, c, d, e}**,
  - The database that comprises the transactions is shown below.

- All **itemsets** are **subsets of *I*.**

- An itemset can have Items from 1 up to *m* members.

Transaction number	Transactions (itemsets)
1	{a, b, c}
2	{a, b, c, d, e}
3	{b}
4	{c, d, e}
5	{c}
6	{b, c, d}
7	{c, d, e}
8	{c, e}

Figure 13.1 A Database With Eight Transactions

Prediction | Support for an Itemset

- We will use the term ***count(S)*** of an **itemset *S*** to mean the number of transactions in the database matched by *S*.

- An itemset ***S matches* a transaction *T*** if *S* is a subset of *T*, i.e. all the items in *S* are also in *T*.
  - For example, itemset {bread, milk} matches the transaction {cheese, bread, fish, milk, juice}. If an itemset *S* = {bread, milk} has a support count of 12, written as count(*S*) = 12 or count({bread, milk}) = 12, it means that 12 of the transactions in the database contain both the items bread and milk.

- We define the ***support(S)*** of an itemset *S*, to be the proportion of itemsets in the database that are matched by *S*,

$$\textbf{support}(S) = \textbf{count}(S)/n$$

Prediction | Association Rules

The aim of **Association Rule Mining (ARM)** is to examine the contents of the database and find rules, known as ***association rules***, in the data.

- For example, we might notice that when items *c* and *d* are bought item *e* is often bought too. We can write this as the rule

$$cd \rightarrow e$$
$$\{c, d\} \rightarrow \{e\}$$

- The arrow is read as **'implies'** to think of rules in terms of ***prediction***: if we know that *c* and *d* were bought we can predict that *e* was also bought.

Prediction | Association Rules

- The rule ***cd → e*** is satisfied for transactions 2, 4 and 7, but not for transaction 6, i.e. it is **satisfied in 75% of cases**.

- For basket analysis it might be interpreted as 'if bread and milk are bought, then cheese is bought too in 75% of cases'.

- Items *c*, *d* and *e* in transactions 2, 4, and 7 can also be used to find other rules such as

  c → ed ***e → cd***

  ce →d ***de → c***

Transaction number	Transactions (itemsets)
1	{a, b, c}
2	{a, b, c, d, e}
3	{b}
4	{c, d, e}
5	{c}
6	{b, c, d}
7	{c, d, e}
8	{c, e}

Figure 13.1 A Database With Eight Transactions

Prediction | Association Rules

- The set of items appearing on the left- and right-hand sides of a given rule as ***L*** and ***R***, respectively, and the rule ***L → R***.
- ***L*** and ***R*** are called
  - ***antecedent*** and ***consequent*** , or
  - ***body*** and ***head***.

- *L* and *R* must each have at least one member and the two sets must be ***disjoint***, i.e. have no common members.

- The *union* ***L ∪ R*** of the sets *L* and *R* is the set of items that occur in either *L* or *R*. The number of items in the itemset *L ∪ R*, is called the ***cardinality*** **of** ***L ∪ R*** (must be at least two).

Prediction | Association Rules

- For the rule ***cd → e***, we have
  L = {c, d}, ***R = {e}*** and ***L ∪ R = {c, d, e}***.
  count(*L*) = 4 and **count(*L ∪ R*) = 3**.

- As there are **n=8** transactions in the database we can calculate
  support(*L*) = count(*L*)/8 = 4/8 and
  support(*L ∪ R*) = count(*L ∪ R*)/8 = 3/8

- A large number of rules can be generated from even quite a small database and we are generally only interested in those that satisfy given criteria for *interestingness*.

- There are many interestingness measures, but the two most commonly used are ***support*** and ***confidence***.

Support

The *support* for a rule $L \rightarrow R$ is the proportion of the database to which the rule successfully applies

i.e. the proportion of transactions in which the items in L and the items in R occur together.

This value is just the support for itemset $L \cup R$, so we have

support($L \rightarrow R$) = support($L \cup R$).

Confidence

The confidence of a rule can be calculated either by

$$\text{confidence}(L \rightarrow R) = \text{count}(L \cup R) / \text{count}(L)$$

or by

$$\text{confidence}(L \rightarrow R) = \text{support}(L \cup R) / \text{support}(L)$$

We reject any rule if the support is below a minimum threshold value called ***minsup***, typically **0.01** (i.e. 1%) and also reject all rules if the confidence below a minimum threshold value called ***minconf***, typically **0.8** (i.e. 80%).

For the rule $cd \rightarrow e$, the confidence is count($\{c, d, e\}$)/count($\{c, d\}$), which is 3/4 = 0.75.

Prediction | Generating Association Rules: Apriori Algorithm

- This account is based on the very influential ***Apriori* algorithm** by Agrawal and Srikant [16], which showed how association rules could be generated in a realistic timescale, at least for relatively small databases.

- The method relies on the following very important result.

Theorem 1
If an itemset is supported, all of its (non-empty) subsets are also supported.
Proof
Removing one or more of the items from an itemset cannot reduce and will often increase the number of transactions that it matches. Hence the support for a subset of an itemset must be at least as great as that for the original itemset. It follows that any (non-empty) subset of a supported itemset must also be supported.

Prediction | Generating Association Rules: Apriori Algorithm

- If we write the set containing all the supported itemsets with cardinality ***k*** as L_k then a second important result is

Theorem 2

If $L_k = \emptyset$ (the empty set) then L_{k+1}, L_{k+2} etc. must also be empty.

Proof

If any supported itemsets of cardinality $k+1$ or larger exist, they will have subsets of cardinality k and it follows from Theorem 1 that all of these must be supported. However we know that there are no supported itemsets of cardinality k as L_k is empty. Hence there are no supported subsets of cardinality $k+1$ or larger and L_{k+1}, L_{k+2} etc. must all be empty.

Prediction | Generating Association Rules: Apriori Algorithm

- The *Apriori* algorithm for generating all the supported itemsets of cardinality at least 2.

Create L_1 = set of supported itemsets of cardinality one
Set k to 2
while $(L_{k-1} \neq \emptyset)$ {
    Create C_k from L_{k-1}
    Prune all the itemsets in C_k that are not
        supported, to create L_k
    Increase k by 1
}
The set of all supported itemsets is $L_1 \cup L_2 \cup \cdots \cup L_k$

Figure 13.2 The Apriori Algorithm (adapted from [16])

Prediction | Generating Association Rules: Apriori Algorithm

- To start the process we construct C_1, the set of all itemsets comprising just one item
- Count the number of transactions that match each itemset.
- Divide each of these counts by the number of transactions in the database gives the value of **support** for each single-element itemset.
- We discard all those with **support < *minsup*** to give L_1.

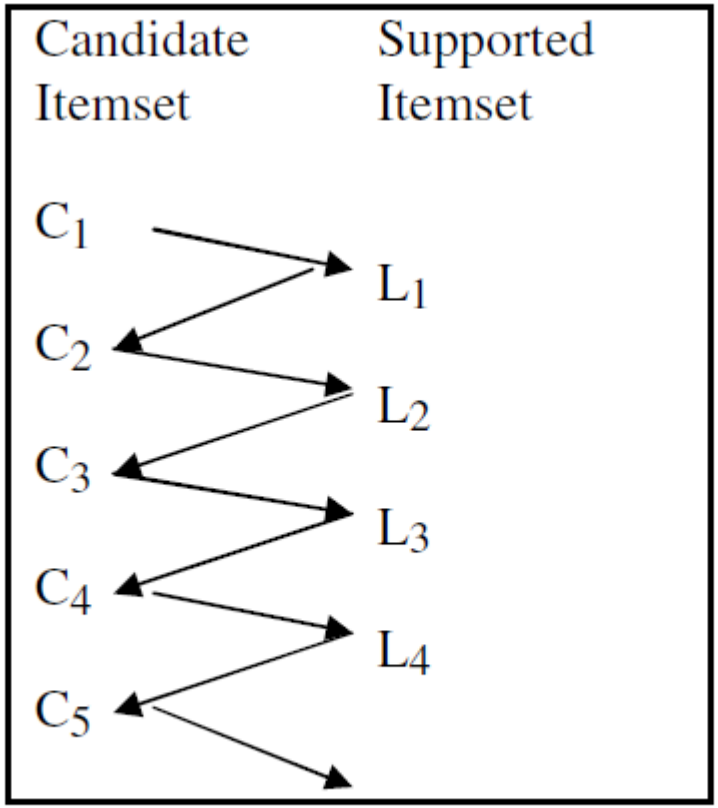


Figure 13.3 Diagram Illustrating the Apriori Algorithm

Prediction | Generating Association Rules: Apriori-gen Algorithm

- Agrawal and Srikant's paper also gives an algorithm *Apriori-gen* which takes L_{k-1} and generates C_k in two stages below

(Generates C_k from L_{k-1})
Join Step
Compare each member of L_{k-1}, say A, with every other member, say B, in turn. If the first $k-2$ items in A and B (i.e. all but the rightmost elements of the two itemsets) are identical, place set $A \cup B$ into C_k.
Prune Step
For each member c of C_k in turn {
Examine all subsets of c with $k-1$ elements
Delete c from C_k if any of the subsets is not a member of L_{k-1}
}

Figure 13.4 The Apriori-gen Algorithm (adapted from [16])

Prediction | The Apriori Algorithm: Example

- Consider a database, D consisting of 8 transactions.
- Suppose that

  minsup = 0.25

  minconf = 0.8 (i.e. 80%)

- We have to first find out the frequent itemset using Apriori algorithm.

- Then, association rules will be generated using min. support & min. confidence.

Transaction number	Transactions (itemsets)
1	{a, b, c}
2	{a, b, c, d, e}
3	{b}
4	{c, d, e}
5	{c}
6	{b, c, d}
7	{c, d, e}
8	{c, e}

Figure 13.1 A Database With Eight Transactions

Prediction | Generating association rules using Apriori algorithm: Example

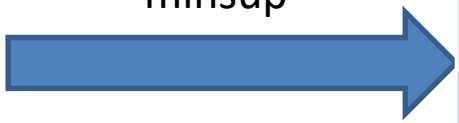
Step 1 : Generating 1-itemset Frequent Pattern

Scan D for count of each candidate

Itemset	Support
{a}	2/8 =0.25
{b}	4/8=0.5
{c}	7/8=0.88
{d}	4/8=0.5
{e}	4/8=0.5

C_1

Compare candidate support count with minsup



Itemset	Support
{a}	2/8 =0.25
{b}	4/8=0.5
{c}	7/8=0.88
{d}	4/8=0.5
{e}	4/8=0.5

L_1

- The set of frequent 1-itemsets, L 1, consists of the candidate 1-itemsets satisfying minimum support.
- In the first iteration of the algorithm, each item is a member of the set of candidate.

Step 2: Generating 2-itemset Frequent Pattern

Generate C_2 candidates from L_1

Itemset	Support
{a,b}	2/8 =0.25
{a,c}	2/8=0.25
{a,d}	1/8=0.13
{a,e}	1/8=0.13
{b,c}	3/8=0.38
{b,d}	2/8=0.25
{b,e}	1/8=0.13
{c,d}	4/8=0.5
{c,e}	4/8=0.5
{d,e}	3/8=0.5

C_2

Compare candidate support count with minsup

Itemset	Support
{a,b}	2/8 =0.25
{a,c}	2/8=0.25
{b,c}	3/8=0.38
{b,d}	2/8=0.25
{c,d}	4/8=0.5
{c,e}	4/8=0.5
{d,e}	3/8=0.5

L_2

Step 3: Generating 3-itemset Frequent Pattern: Join Step

- The generation of the set of candidate 3-itemsets, C_3, involves use of the Apriori-gen algorithm.

- In order to find C_3, we compute
- $C_3 = L_2$ Join L_2
- C_3 ={{a,b,c}, {a,b,d}, {a,c,d}, {a,c,e}, {b,c,d}, {b,c,e}, {b,d,e}, {c,d,e}}.

- Now, **join step** is complete and **prune step** will be used to reduce the size of C_3 .

Itemset
{a,b}
{a,c}
{b,c}
{b,d}
{c,d}
{c,e}
{d,e}

L_2

Step 3: Generating 3-itemset Frequent Pattern: Prune Step

Based on the Apriori that all subsets of a frequent itemset must also be frequent, How ?

{a,b,c} → 2-itemsets : {a,b}, {a,c}, {b,c} → All occurs in L_2 → OK
{a,b,d} →2-itemsets : {a,b}, {a,d}, {b,d} → {a,d} does not occur in L_2→ NOT OK
{a,c,d} → 2-itemsets : {a,c}, {a,d}, {c,d} → {a,d} does not occur in L_2→ NOT OK
{a,c,e} → 2-itemsets : {a,c}, {a,e}, {c,e} → {a,e} does not occur in L_2→ NOT OK
{b,c,d} → 2-itemsets : {b,c}, {b,d}, {c,d} → All occur in L_2→ OK
{b,c,e} → 2-itemsets : {b,c}, {b,e}, {c,e} → {b,e} does not occur in L_2→ NOT OK
{b,d,e} → 2-itemsets : {b,d}, {b,e}, {d,e} → {b,e} does not occur in L_2→ NOT OK
{c,d,e} → 2-itemsets : {c,d}, {c,e}, {d,e} → All occur in L_2→ OK

Itemset
{a,b}
{a,c}
{b,c}
{b,d}
{c,d}
{c,e}
{d,e}

L_2

Step 3: Generating 3-itemset Frequent Pattern: C_3

Generate C_3 candidates from L_2

Itemset	Support
{a,b,c}	2/8 =0.25
{b,c,d}	2/8=0.25
{c,d,e}	3/8=0.38

C_3

Compare candidate support count with minsup

Itemset	Support
{a,b,c}	2/8 =0.25
{b,c,d}	2/8=0.25
{c,d,e}	3/8=0.38

L_3

Step 4: Generating 4-itemset Frequent Pattern

- The algorithm uses L_3 Join L_3 to generate a candidate set of 4-itemsets, C4. Although the join results in {{a, b, c, d}, {b,c,d,e}, but they are not found in L_3 .
- Thus, C4= ϕ , and algorithm terminates, having found all of the frequent items. This completes our Apriori Algorithm.

What's Next ?

- These frequent itemsets will be used to generate **valuable/strong/interesting association rules** (where these rules should satisfy both minimum support & minimum confidence).

Step 5: Generating Association Rules for Supported (Frequent) Itemset

- Generate all possible L sides, then generate R using the unused items in L.
- For itemset *{a,b,c}* there are 6 possible rules that can be generated, as listed below.

Itemset
{a,b,c}
{b,c,d}
{c,d,e}

L_3

Rule $L \rightarrow R$	count($L \cup R$)	count(L)	confidence ($L \rightarrow R$)	Satisfy minconf?
a→bc	2	2	1	Yes
b→ac	2	4	0.5	NO
c→ab	2	7	0.29	NO
ab→c	2	2	1	Yes
ac→b	2	2	1	Yes
bc→a	2	3	0.67	No

Prediction | The Apriori Algorithm: Example

- For itemset {b,c,d} there are 6 possible rules that can be generated, as listed below.

Rule $L \rightarrow R$	count($L \cup R$)	count(L)	confidence ($L \rightarrow R$)	Satisfy minconf?
b→cd	2	4	0.5	NO
c→bd	2	7	0.28	NO
d→bc	2	4	0.5	NO
bc→d	2	3	0.67	NO
bd→c	2	2	0.5	NO
cd→b	2	4	0.5	NO

Itemset
{a,b,c}
{b,c,d}
{c,d,e}

L_3

Prediction | The Apriori Algorithm: Example

- For itemset *{c, d, e}* there are 6 possible rules that can be generated, as listed below.

Rule $L \rightarrow R$	count($L \cup R$)	count(L)	confidence ($L \rightarrow R$)	Satisfy minconf?
c→de	3	7	0.43	NO
d→ce	3	4	0.75	NO
e→cd	3	4	0.75	NO
cd→e	3	4	0.75	NO
ce→d	3	4	0.75	NO
de→c	3	3	1	Yes

Itemset
{a,b,c}
{b,c,d}
{c,d,e}

L_3

Prediction | Handout (7): Generating Association Rules for Market Basked Using Apriori Algorithm

Recap of Previous Lecture
Content of This Lecture
Summary & Checklist

Summary & Checklist

- [x] Association Rule Mining- Part: II: Introduction
- [x] Transactions and Itemsets
- [x] Support for an Itemset
- [x] Association Rules
- [x] Generating Association Rules: Apriori Algorithm
- [x] Generating Association Rules: Apriori-gen Algorithm
- [x] Generating association rules using Apriori algorithm: Example
- [x] Handout (7): Generating Association Rules for Market Basked Using Apriori Algorithm

Reminder | Next Lecture !

Next Lecture…

Clustering (Ch. 14)

- *Be ready!*
- *Prepare for Quiz (2).*
- *Download & print the lecture notes before your class.*

Thank You !

✉ s.zahrani@tu.edu.sa