Association Analysis Short Review

cse352

Artificial Intelligence

Professor Anita Wasilewska
Computer Science Department
Stony Brook University

The Apriori Algorithm: Basics

The Apriori Algorithm

It is an influential algorithm for mining
frequent itemsets
and using them for creating
association rules

Key Concepts:

- Frequent Itemsets
- Apriori Property

The Apriori Algorithm: Basics

Key Concepts:

Frequent Itemset

is the set of itemset which has minimum support

which also has the following

Apriori Property:

“ all subsets of frequent itemset must be frequent”

- Join Operation
- To find C_k , a set of candidate k-itemsets is generated by **joining** L_{k-1} with itself.

The Apriori Algorithm in a Nutshell

- Apriori Algorithm **finds** the frequent itemsets
  i.e. sets of items that have minimum support

  and follows the

  Apriori Principle:

  all subsets of a frequent itemset must be frequent itemsets

  i.e. {*A, B*} is a frequent itemset only of both {*A*} and {*B*} are frequent itemsets

The Apriori Algorithm in a Nutshell

- Apriori Algorithm

  The algorithm Iteratively **finds** frequent itemsets with cardinality from 1 to *k* (k-itemset)

- As the next step in the Apriori Process we use the frequent itemsets to **generate** **association rules**

The Apriori Algorithm : Pseudo code

- Join Step: C_k is generated by **joining** L_{k-1} with itself
- Prune Step: Any (k-1)-itemset that is not frequent cannot be a subset of a frequent k-itemset

- Pseudo-code:

C_k: Candidate itemset of size k
L_k : frequent itemset of size k

L_1 = {frequent items};
for (k = 1; L_k !=$\varnothing$; k++) **do begin**
C_{k+1} = candidates generated from L_k;
for each transaction t in database do
increment the count of all candidates in C_{k+1} that are contained in t
L_{k+1} = candidates in C_{k+1} with min_support
end
return $\cup_k L_k$;

The Apriori Algorithm: Example

TID	List of Items
T100	I1, I2, I5
T100	I2, I4
T100	I2, I3
T100	I1, I2, I4
T100	I1, I3
T100	I2, I3
T100	I1, I3
T100	I1, I2 ,I3, I5
T100	I1, I2, I3

- Consider a database, D , consisting of 9 transactions.
- Suppose min. support count required is 2 (i.e. min_sup = 2/9 = 22 %)
- Let minimum confidence required is 70%.
- We have to first find out the **frequent itemset** using Apriori algorithm.
- Then, **Association rules** will be generated using min. support & min. confidence.

Step 1: Generating 1-itemset Frequent Pattern

Scan D for count of each candidate

Itemset	Sup.Count
{I1}	6
{I2}	7
{I3}	6
{I4}	2
{I5}	2

C_1

Compare candidate support count with minimum support count

Itemset	Sup.Count
{I1}	6
{I2}	7
{I3}	6
{I4}	2
{I5}	2

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 candidates.

Step 2: Generating 2-itemset Frequent Pattern

- To **discover** the set of frequent 2-itemsets, L_2 , the algorithm uses $\mathbf{L_1}$ ***Join*** $\mathbf{L_1}$ to generate a candidate set of 2-itemsets, C_2

- **Next**, the transactions in D are scanned and the support count for each candidate itemset in C_2 is accumulated (as shown in the middle table)

Step 2: Generating 2-itemset Frequent Pattern

- 2-itemsets, L_2 , is then **determined**, consisting of those candidate 2-itemsets in C_2 having minimum support

- **Note:** We haven't used Apriori Property because all 1-itemsets were frequent

Step 2: Generating 2-itemset Frequent Pattern

Generate C_2 candidates from L_1

Itemset
{I1, I2}
{I1, I3}
{I1, I4}
{I1, I5}
{I2, I3}
{I2, I4}
{I2, I5}
{I3, I4}
{I3, I5}
{I4, I5}

C_2

Scan D for count of each candidate

Itemset	Sup. Count
{I1, I2}	4
{I1, I3}	4
{I1, I4}	1
{I1, I5}	2
{I2, I3}	4
{I2, I4}	2
{I2, I5}	2
{I3, I4}	0
{I3, I5}	1
{I4, I5}	0

C_2

Compare candidate support count with minimum support count

Itemset	Sup Count
{I1, I2}	4
{I1, I3}	4
{I1, I5}	2
{I2, I3}	4
{I2, I4}	2
{I2, I5}	2

L_2

Step 3: Generating 3-itemset Frequent Pattern

L_2 *Join* L_2 and **prune to generate** C_3 candidates

Itemset
{I1, I2, I3}
{I1, I2, I5}

C_3

Scan D for count of each candidate

Itemset	Sup. Count
{I1, I2, I3}	2
{I1, I2, I5}	2

C_3

Compare candidate support count with min support count

Itemset	Sup Count
{I1, I2, I3}	2
{I1, I2, I5}	2

L_3

- In order to find C_3, we first compute L_2 *Join* L_2
- C_3 = L2 *Join* L2 = {{I1, I2, I3}, {I1, I2, I5}, {I1, I3, I5}, {I2, I3, I4}, {I2, I3, I5}, {I2, I4, I5}}.
- Now, Join step is **complete** and **Prune step** will be used to **reduce** the size of C_3
- **Prune step** uses Apriori Property helps to avoid heavy computation due to large C_k.

Step 3: Generating 3-itemset Frequent Pattern

- Apriori property says that all subsets of a frequent itemset must also be frequent

- C_3 = L2 *Join* L2 = {{I1, I2, I3}, {I1, I2, I5}, {I1, I3, I5}, {I2, I3, I4}, {I2, I3, I5}, {I2, I4, I5}}

- We determine now which of candidates in C_3 **can** and which **can not** possibly be frequent
- Take {I1, I2, I3}
- The 2-item subsets of it are {I1, I2}, {I1, I3}, {I2, I3}

  All of them are members of L_2

  We **keep** {I1, I2, I3} in C_3

Step 3: Generating 3-itemset Frequent Pattern

- Lets take {I2, I3, I5}
- The 2-item subsets are {I2, I3}, {I2, I5}, {I3,I5}
- But {I3, I5} is not a member of L_2 and hence it is not frequent **violating** Apriori Property
- Thus we **remove** {I2, I3, I5} from C_3

All 2-item subsets of {I1, I2, I5} members of L_2

Therefore C_3 = {{I1, I2, I3}, {I1, I2, I5}}

- Now, the transactions in D are scanned in order to determine L_3, consisting of those candidates 3-itemsets in C_3 having **minimum support** and we get that

- L_3 = {{I1, I2, I3}, {I1, I2, I5}}

Step 4: Generating 4-itemset Frequent Pattern

- The algorithm uses L_3 *Join* L_3 to generate a candidate set of 4-itemsets, C_4
- C_4 = L3 *Join* L3 = {{I1, I2, I3, I5}}
- This itemset {{I1, I2, I3, I5}} is **pruned** since its subset {{I2, I3, I5}} is **not frequent.**
- Thus, $C_4 = \varphi$ and algorithm **terminates**
- What' s Next ?

  Obtained **frequent itemsets** are to be used to generate strong association rules

(where strong association rules are rules that satisfy both minimum support and minimum confidence)

Step 5: Generating Association Rules from Frequent Itemsets

- Procedure:
  - For each frequent itemset **I**, generate the set of all nonempty subsets of **I**
  - For every nonempty subset ***S*** of **I**,
  - output the rule **S → I - S**
  - if support_count(**I**) **/** support_count(**S**) **>=** min_conf
  - where min_conf is minimum confidence threshold.

- Example

  We obtained the set od all frequent itemsets

L = {{I1}, {I2}, {I3}, {I4}, {I5}, {I1,I2}, {I1,I3}, {I1,I5}, {I2,I3}, {I2,I4}, {I2,I5}, {I1,I2,I3}, {I1,I2,I5}}

- Lets take for example **I** = {I1,I2,I5}

Step 5: Generating Association Rules from Frequent Itemsets

- Lets take **I** = {I1,I2,I5}
  - Its all nonempty subsets are {I1,I2}, {I1,I5}, {I2,I5}, {I1}, {I2}, {I5}

  Let minimum confidence threshold be, say 70%
- The resulting association rules are shown below, each listed with its confidence.
  - R1: I1 ^ I2 → I5
    - Confidence = sc{I1,I2,I5}/sc{I1,I2} = 2/4 = 50%
    - R1 is Rejected.
  - R2: I1 ^ I5 → I2
    - Confidence = sc{I1,I2,I5}/sc{I1,I5} = 2/2 = 100%
    - R2 is Selected.
  - R3: I2 ^ I5 → I1
    - Confidence = sc{I1,I2,I5}/sc{I2,I5} = 2/2 = 100%
    - R3 is Selected.

Step 5: Generating Association Rules from Frequent Itemsets

- R4: I1 → I2 ^ I5
  - Confidence = sc{I1,I2,I5}/sc{I1} = 2/6 = 33%
  - R4 is rejected.
- R5: I2 → I1 ^ I5
  - Confidence = sc{I1,I2,I5}/{I2} = 2/7 = 29%
  - R5 is rejected.
- R6: I5 → I1 ^ I2
  - Confidence = sc{I1,I2,I5}/ {I5} = 2/2 = 100%
  - R6 is Selected

- We have found three **strong** association rules