CSE 601
Association Analysis Report

Submitted by
Paritosh Kumar Velalam (pvelalam) (50295537)
Sampreeth Boddi Reddy (sboddire) (50298591)

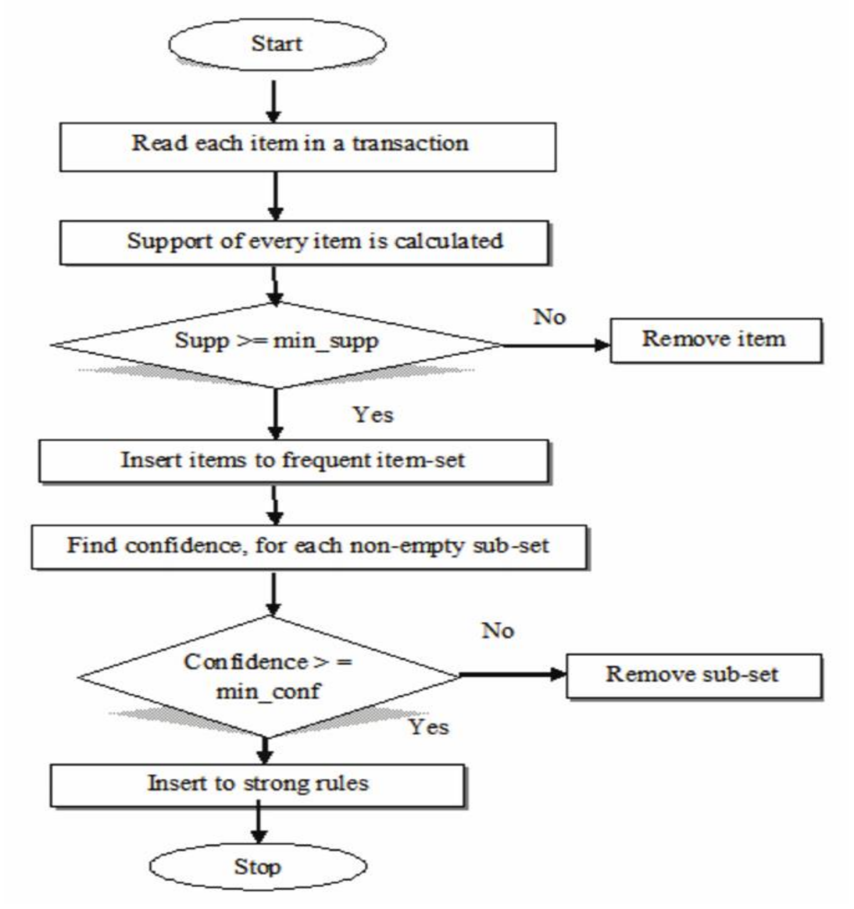
Apriori Principle:

Apriori algorithm is useful in mining frequent item sets and relevant association rules. It's used to find groups of items that occur together frequently in a dataset. Usually, this algorithm works on a database containing a large number of transactions.

The algorithm uses a simple concept: "If an item set is frequent then all of its subsets will be also frequent". Therefore, by using anti-monotone we property we can conclude that "If an item set is infrequent then all of its supersets will be also infrequent".

When we create combinations of data items, then for each level of item set lattice we can check the support count. If any of the nodes of the item set lattice has a support below the given threshold, then we can eliminate all the supersets of that node.

Flow Chart of Apriori Algorithm:


Start
Read each item in a transaction
Support of every item is calculated
Supp >= min_supp
No
Remove item
Yes
Insert items to frequent item-set
Find confidence, for each non-empty sub-set
Confidence > = min_conf
No
Remove sub-set
Yes
Insert to strong rules
Stop

Flow chart of apriori algorithm for rules generation

Steps followed to implement Apriori Algorithm:

1. Read the data from the given dataset and add features of each disease to set and maintain list of sets for all the records from input file

2. Read all the attributes and append the attributes whose support is greater than given support to a frequent item set

3. Then we feed this 1-length item set to generate the 2-length item set by using union operator and checking minimum support condition. Again two length frequent item sets are merged and sets of length three are selected as frequent item sets. In this way all frequent item sets up to k length are generated.

4. We save all length frequent item sets in the dictionary for generating rules.

5. Now iterate through all the item sets in the dictionary and generate combinations for each frequent item set and find confidence of each rule.

6. Save all the rules whose confidence is greater than given minimum confidence to pandas data frame

7. Now according to the query input retrieve the rules from data frame and print them.

Results obtained for different support values:

1. Support is set to be 30%

Number of length-1 frequent item sets	196
Number of length-2 frequent item sets	5340
Number of length-3 frequent item sets	5287
Number of length-4 frequent item sets	1518
Number of length-5 frequent item sets	438
Number of length-6 frequent item sets	88
Number of length-7 frequent item sets	11
Number of length-8 frequent item sets	1
Number of all-length frequent item sets	12879

2. Support is set to be 40%

Number of length-1 frequent item sets	167
Number of length-2 frequent item sets	753
Number of length-3 frequent item sets	149
Number of length-4 frequent item sets	7
Number of length-5 frequent item sets	1
Number of all-length frequent item sets	1077

3. Support is set to be 50%

Number of length-1 frequent item sets	109
Number of length-2 frequent item sets	63
Number of length-3 frequent item sets	2
Number of all-length frequent item sets	174

4. Support is set to be 60%

Number of length-1 frequent item sets	34
Number of length-2 frequent item sets	2
Number of all-length frequent item sets	36

5. Support is set to be 70%

Number of length-1 frequent item sets	7
Number of all-length frequent item sets	7

Results obtained for different Queries:

1. Template 1 queries:

QUERY	COUNT
(result11, cnt) = asso_rule.template1("RULE", "ANY", ['G59_Up'])	26
(result12, cnt) = asso_rule.template1("RULE", "NONE", ['G59_Up'])	91
(result13, cnt) = asso_rule.template1("RULE", 1, ['G59_Up', 'G10_Down'])	39
(result14, cnt) = asso_rule.template1("HEAD", "ANY", ['G59_Up'])	9

(result15, cnt) = asso_rule.template1("HEAD", "NONE", ['G59_Up'])	108
(result16, cnt) = asso_rule.template1("HEAD", 1, ['G59_Up', 'G10_Down'])	17
(result17, cnt) = asso_rule.template1("BODY", "ANY", ['G59_Up'])	17
(result18, cnt) = asso_rule.template1("BODY", "NONE", ['G59_Up'])	100
(result19, cnt) = asso_rule.template1("BODY", 1, ['G59_Up', 'G10_Down'])	24

2. Template 2 queries:

QUERY	COUNT
(result21, cnt) = asso_rule.template2("RULE", 3)	9
(result22, cnt) = asso_rule.template2("HEAD", 2)	6
(result23, cnt) = asso_rule.template2("BODY", 1)	117

3. Template 3 queries:

QUERY	COUNT
(result31, cnt) = asso_rule.template3("1or1", "HEAD", "ANY", ['G10_Down'], "BODY", 1, ['G59_Up'])	24
(result32, cnt) = asso_rule.template3("1and1", "HEAD", "ANY", ['G10_Down'], "BODY", 1, ['G59_Up'])	1
(result33, cnt) = asso_rule.template3("1or2", "HEAD", "ANY", ['G10_Down'], "BODY", 2)	11
(result34, cnt) = asso_rule.template3("1and2", "HEAD", "ANY", ['G10_Down'], "BODY", 2)	0
(result35, cnt) = asso_rule.template3("2or2", "HEAD", 1, "BODY", 2)	117
(result36, cnt) = asso_rule.template3("2and2", "HEAD", 1, "BODY", 2)	3