CSE 601

Data Mining Project1

Association Analysis

Anjana Guruprasad (50205233)

Apoorva Hejib (50206516)

Shreya Ravi Hegde (50208485)

Overview

To implement apriori algorithm and generate all the frequent itemsets that satisfy the support threshold. And further, Generate association rules based on the query specified and the confidence threshold.

What is Apriori algorithm?

Apriori algorithm is an algorithm that is applied to transactional databases.It is used to mine frequent itemsets from the given data.

The generated itemsets are then used to determine association rules which highlight the general trends that are present in the database. It is used in domains such as e-commerce, market basket analysis and so on.

Measures used to qualify itemsets and rules generated

Support

The support of an itemset **X**, is the proportion of the number of transactions that the itemset **X** has appeared in to the total number of transactions.

$$supp(X) = \frac{\text{Number of transaction in which } X \text{ appears}}{\text{Total number of transactions}}$$

Confidence

An implication expression of the form X -> Y where X and Y are itemsets. The confidence is computed as the proportion of transactions which contain X and Y to the number of transactions that contain only X.

$$conf(X \longrightarrow Y) = \frac{supp(X \cup Y)}{supp(X)}$$

It signifies the likelihood of item Y being purchased when item X is purchased. It can give some important insights, but it also has a major drawback. It only takes into account the popularity of the itemset X and not the popularity of Y. If Y is equally popular as X then there will be a higher probability that a transaction containing X will also contain Y thus increasing the confidence.

How does Apriori algorithm work?

Apriori algorithm is based on the **anti - monotone** property of the *support measure*. It uses the assumption that:

-> All subsets of a frequent itemset must be frequent

-> Similarly, the superset of an infrequent itemset must be infrequent too

Implementation

● Parse the input document in order to retrieve all the fields in the required format.

● Count the occurrences of all length 1 itemsets and convert it into a dictionary of the format where key is the itemset and the value is the count.

● Set the support threshold and remove the keys that don't satisfy the threshold.

● From length 2 itemsets, generate all possible combination from the previous accepted length 1 itemsets.

● Count the occurrence of each combination.

● Remove the itemsets that don't satisfy the support threshold and add the remaining itemsets and their number of occurrences as the key and value of the dictionary.

● For length k (k>2) itemsets, generate all possible itemsets from the previous accepted length k-1 itemsets.

● Merge two itemsets of length k-1 only if the first k elements in the itemset are matching.

● Compare the support of each combination against the support threshold.

● Insert the accepted combinations as keys to the dictionary along with their number of occurrences as the value.

● Repeat the above 4 steps until no further frequent itemset can be generated.

Mining / Generating association rules

Create rules from each frequent itemset using the binary partition of frequent itemsets and look for the ones with high confidence.

-> For each frequent itemset "X", generate all nonempty subsets of X.

-> For every nonempty subset s of X, Output the rule ***s -> X - s***, If

Support_count (X) / Support_count (s) >= Confidence threshold

Implementation

● Consider only the frequent itemsets of length two and more from the apriori algorithm.

● For a frequent itemset of length k, generate all the subsets from length 1 to k-1, only ignoring the null subsets.

● For a frequent itemset X, consider all the subsets s, such that s -> X-s satisfies the confidence threshold.

● For each accepted rules s -> X-s , Consider (s) as the body and (X-s) as the head.

● Check the accepted rules against the query given, Print the rules that satisfy the conditions given

● Repeat the steps until no frequent itemsets are remaining

Pros:

-> It is easy to implement and understand.

-> It can be used on large itemsets.

Cons:

-> Sometimes, it may need to find a large number of candidate rules which can be computationally expensive.

-> Calculating support is expensive as it has to go through the entire database.

Frequent Itemsets Generated for various support thresholds:

Support = 30%

Length Of Itemset	*Number Of Frequent Itemsets*
1	196
2	5340
3	5287
4	1518
5	438

6	88
7	11
8	1

Support = 40%

Length Of Itemset	*Number Of Frequent Itemsets*
1	167
2	753
3	149
4	7
5	1

Support = 50%

Length Of Itemset	*Number Of Frequent Itemsets*
1	109
2	63
3	2

Support = 60%

Length Of Itemset	*Number Of Frequent Itemsets*
1	34

2	2

Support = 70%

Length Of Itemset	*Number Of Frequent Itemsets*
1	7

Association rules generated:

Support is set as 50% and confidence is set as 70%

Queries and the number of rules generated:

Query	***Number of rules generated***
asso_rule.template1("RULE", "ANY", ['G59_Up'])	26
asso_rule.template1("RULE", "NONE", ['G59_Up'])	91
asso_rule.template1("RULE", 1, ['G59_Up', 'G10_Down'])	39
asso_rule.template1("BODY", "ANY", ['G59_Up'])	9
asso_rule.template1("BODY", "NONE", ['G59_Up'])	108
asso_rule.template1("BODY", 1, ['G59_Up', 'G10_Down'])	17
asso_rule.template1("HEAD", "ANY", ['G59_Up'])	17
asso_rule.template1("HEAD", "NONE", ['G59_Up'])	100
asso_rule.template1("HEAD", 1, ['G59_Up', 'G10_Down'])	24
asso_rule.template2("RULE", 3)	9
asso_rule.template2("BODY", 2)	6
asso_rule.template2("HEAD", 1)	117
asso_rule.template3("1or1", "BODY", "ANY", ['G10_Down'], "HEAD", 1, ['G59_Up'])	24

asso_rule.template3("1and1", "BODY", "ANY", ['G10_Down'], "HEAD", 1, ['G59_Up'])	1
asso_rule.template3("1or2", "BODY", "ANY", ['G10_Down'], "HEAD", 2)	11
asso_rule.template3("1and2", "BODY", "ANY", ['G10_Down'], "HEAD", 2)	0
asso_rule.template3("2or2", "BODY", 1, "HEAD", 2)	117
asso_rule.template3("2and2", "BODY", 1, "HEAD", 2)	3

References:

-> Lecture notes

-> https://en.wikipedia.org/wiki/Apriori_algorithm

-> https://www.slideshare.net/INSOFE/apriori-algorithm-36054672