Association Report

CSE 601

Project1

Kun Lin 50146291

Weijin Zhu 50146866

Apriori Algorithm:

Apriori algorithm is used in Mining association rule and frequent set generation. One key apriori principle is if an itemset is frequent, then all of its subsets must be frequent, on the other hand any superset of an infrequent itemset must be infrequent. With that in mind, we are able to reduce the number of candidates set, eliminating it's superset if the candidate's support count does meet the threshold.

Flow of the association rule generation:

1. **Generating frequent itemsets:** Let k = 1, use generate_set() method generate size 1 candidate itemsets. Then we pass the candidate sets into scan_and_generate_frequent() that will eliminate any candidate that does not meet the min_sup count threshold. From there onward, to find k+1 length candidate sets we simply use k-1 frequent sets where each two k sets have to share k-1 common genes. For example if k=2, and we're trying to generate k+1, length of 3 candidate sets, the two k length frequent sets have to share 1 common gene then we can union the gene and form length 3 candidates. repeat the step until no further k+1 candidate set could be generated.
2. **Generating association rule:** After we generated frequent sets of various lengths. For each frequent itemset, we will call rule_one_one_freq_set() to recursively find the possible combinations of association rules that meet this single frequent set that meets the minimum confidence threshold. If an association rule is found to not satisfy the min_confidence, we will stop the recursive call that generates more rules with the same itemset but more items on the right (by Apriori monotone property). Finally append all the association rules together to form one over all association rules.

Part1 Result:

Support is set to 30%
number of length-1 frequent itemsets: 194
number of length-2 frequent itemsets: 5323

number of length-3 frequent itemsets: 5251
number of length-4 frequent itemsets: 1463
number of length-5 frequent itemsets: 388
number of length-6 frequent itemsets: 61
number of length-7 frequent itemsets: 3
number of all lengths frequent itemsets: 12683

Support is set to 40%
number of length-1 frequent itemsets: 167
number of length-2 frequent itemsets: 753
number of length-3 frequent itemsets: 149
number of length-4 frequent itemsets: 7
number of length-5 frequent itemsets: 1
number of all lengths frequent itemsets: 1077

Support is set to 50%
number of length-1 frequent itemsets: 109
number of length-2 frequent itemsets: 63
number of length-3 frequent itemsets: 2
number of all lengths frequent itemsets: 174

Support is set to 60%
number of length-1 frequent itemsets: 34
number of length-2 frequent itemsets: 2
number of all lengths frequent itemsets: 36

Support is set to 70%
number of length-1 frequent itemsets: 7
number of all lengths frequent itemsets: 7

Part2 Result:

*note: all the template calls return results in the form of head,body,count. where head and body are lists that store the rules of head and body with corresponding indexes that satisfy the query. and count is the number of over satisfying rules.

Template 1 query results:

(result11,cnt) = asso_rule.template1("RULE", "ANY". ['G59_Up'])
- cnt = 26

(result12,cnt) = asso_rule.template1("RULE", "NONE". ['G59_Up'])
- cnt = 91

(result13,cnt) = asso_rule.template1("RULE", 1, ['G59_Up', 'G10_Down'])
- cnt = 39

(result14,cnt) = asso_rule.template1("HEAD", "ANY". ['G59_Up'])
- cnt = 9

(result15,cnt) = asso_rule.template1("HEAD", "NONE". ['G59_Up'])
- cnt = 108

(result16,cnt) = asso_rule.template1("HEAD", 1, ['G59_Up', 'G10_Down'])
- cnt = 17

(result17,cnt) = asso_rule.template1("BODY", "ANY", ['G59_Up'])
- cnt = 17

(result18,cnt) = asso_rule.template1("BODY", "NONE". ['G59_Up'])
- cnt = 100

(result19,cnt) = asso_rule.template1("BODY", 1, ['G59_Up', 'G10_Down'])
- cnt = 24

Template 2 query results:

(result21, cnt) = asso_rule.template2("RULE", 3)
- cnt = 9

(result22, cnt) = asso_rule.template2("HEAD", 2)
- cnt = 6

(result23, cnt) = asso_rule.template2("BODY", 1)
- cnt =117

Template 3 query results:

(result31, cnt) = asso_rule.template2("1or1", "HEAD", "ANY", ['G10_Down'], "BODY", 1, ['G59_Up'])
- cnt = 24

(result32, cnt) = asso_rule.template2(“1and1”, “HEAD”, “ANY”, [‘G10_Down’], “BODY”, 1, [‘G59_Up’])

- cnt = 1

(result33, cnt) = asso_rule.template2(“1or2”, “HEAD”, “ANY”, [‘G10_Down’], “BODY”, 2)

- cnt = 11

(result34, cnt) = asso_rule.template2(“1and2”, “HEAD”, “ANY”, [‘G10_Down’], “BODY”, 2)

- cnt = 0

(result35, cnt) = asso_rule.template2(“2or2”, “HEAD”, 1, “BODY”, 2)

- cnt = 117

(result36, cnt) = asso_rule.template2(“2and2”, “HEAD”, 1, “BODY”, 2)

- cnt = 3