Data Mining

Project 1: Dimensionality Reduction & Association Analysis

Team No. 39

Arvind Srinivass Ramanathan arvindsr 50205659
Naveen Muralidhar Prakash naveenmu 50208032
Senthil Kumar Laguduva Yadindra Kumar laguduva 50207553

Apriori Algorithm

- The file *associationruletestdata.txt* is read and each gene expression of all records is independently numbered and stored into *setDatabase* as a set for performing comparisons.
- The dataset is parsed to generate all unique gene expressions and the number of occurrences of them are stored in a dictionary *itemCount.* A minimum threshold support value is picked and the gene expressions that are over the value are the required 1-frequent itemsets.
- Using the 1-frequent itemsets, we generate all possible combinations of two gene expressions. Combinations below the support value are eliminated. Thus, the 2-frequent itemsets are generated.
- 3-frequent itemsets are generated by combining 2-frequent itemsets having 1 common gene expression among them.
- Similarly, by looping through, upto K frequent itemsets are obtained by combining the K-1 frequent itemsets such that they have K-2 common gene expressions.
- This process stops when no more K-frequent itemsets are generated.
- At each step the resulting n-frequent itemsets are pruned and these itemsets are not used to generate subsequent n-frequent itemsets.

Part - 1
Results obtained by different support values

Support is set to be 30%
number of length-1 frequent itemsets: 196
number of length-2 frequent itemsets: 5340
number of length-3 frequent itemsets: 5287
number of length-4 frequent itemsets: 1518
number of length-5 frequent itemsets: 438
number of length-6 frequent itemsets: 88
number of length-7 frequent itemsets: 11
number of length-8 frequent itemsets: 1
number of length-9 frequent itemsets: 0
number of all length frequent itemsets: 12879

Support is set to be 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 length-6 frequent itemsets: 0
number of all length frequent itemsets: 1077

Support is set to be 50%
number of length-1 frequent itemsets: 109
number of length-2 frequent itemsets: 63
number of length-3 frequent itemsets: 2
number of length-4 frequent itemsets: 0
number of all length frequent itemsets: 174

Support is set to be 60%
number of length-1 frequent itemsets: 34
number of length-2 frequent itemsets: 2
number of length-3 frequent itemsets: 0
number of all length frequent itemsets: 36

Support is set to be 70%
number of length-1 frequent itemsets: 7
number of length-2 frequent itemsets: 0
number of all length frequent itemsets: 7

Part – 2

The total number of rules generated for support at 30% and confidence at 70% is **31759**
The total number of rules generated for support at 40% and confidence at 70% is **1137**
The total number of rules generated for support at 50% and confidence at 70% is **117**.
The total number of rules generated for support at 60% and confidence at 70% is **4**
The total number of rules generated for support at 70% and confidence at 70% is **0**

Answer to template queries

(result11, cnt) = asso_rule.template1("RULE", "ANY", ['G59_Up'])
The number of rules that match the query is **26**

(result12, cnt) = asso_rule.template1("RULE", "NONE", ['G59_Up'])
The number of rules that match the query is **91**

(result13, cnt) = asso_rule.template1("RULE", 1, ['G59_Up','G10_Down'])
The number of rules that match the query is **39**

(result14, cnt) = asso_rule.template1("BODY", "ANY", ['G59_Up'])
The number of rules that match the query is **9**

(result15, cnt) = asso_rule.template1("BODY", "NONE", ['G59_Up'])
The number of rules that match the query is **108**

(result16, cnt) = asso_rule.template1("BODY", 1, ['G59_Up', 'G10_Down'])
The number of rules that match the query is **17**

(result17, cnt) = asso_rule.template1("HEAD", "ANY", ['G59_Up'])
The number of rules that match the query is **17**

(result18, cnt) = asso_rule.template1("HEAD", "NONE", ['G59_Up'])
The number of rules that match the query is **100**

(result19, cnt) = asso_rule.template1("HEAD", 1, ['G59_Up','G10_Down'])
The number of rules that match the query is **24**

(result21, cnt) = asso_rule.template2("RULE", 3)
The number of rules that match the query is **9**

(result22, cnt) = asso_rule.template2("BODY", 2)
The number of rules that match the query is **6**

(result23, cnt) = asso_rule.template2("HEAD", 1)
The number of rules that match the query is **117**

(result31, cnt) = asso_rule.template3("1or1", "BODY", "ANY", ['G10_Down'], "HEAD", 1, ['G59_Up'])
The number of rules that match the query is **24**

(result32, cnt) = asso_rule.template3("1and1", "BODY", "ANY", ['G10_Down'], "HEAD", 1, ['G59_Up'])
The number of rules that match the query is **1**

(result33, cnt) = asso_rule.template3("1or2", "BODY", "ANY", ['G10_Down'], "HEAD", 2)
The number of rules that match the query is **11**

(result34, cnt) = asso_rule.template3("1and2", "BODY", "ANY", ['G10_Down'], "HEAD", 2)
The number of rules that match the query is **0**

(result35, cnt) = asso_rule.template3("2or2", "BODY", 1, "HEAD", 2)
The number of rules that match the query is **117**

(result36, cnt) = asso_rule.template3("2and2", "BODY", 1, "HEAD", 2)
The number of rules that match the query is **3**