CSE 601: Data Mining and Bioinformatics

Project 1.2

Association Analysis

SUBMITTED BY:
DEBANJAN PAUL (50208716)
JAY BAKSHI (50206954)
SHIVAM GUPTA (50206323)

Objective:

This project is divided in two parts:

Part 1: Frequent Item Set Generation

The aim of this project is to generate sets of data items which has support above a given minimum value. We achieve this by using Apriori algorithm.

Part 2: Association Rule Generation

The aim of this project is to generate association rules from the given frequent item sets. The rules are generated according to a given query (which needs to be in a specific format as described in readme.txt).

Apriori Algorithm:

The apriori algorithm is an efficient way to extract combinations of item sets which has support above a threshold. The algorithm uses a simple concept: "If an itemset 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".
Therefore, when we create combinations of data items then we do not need to compute support for all of the combinations. Instead, when we create combinations of data items, then for each level of itemset lattice we can check the support count. If any of the nodes of the itemset lattice has a support below the given threshold, then we can eliminate all the supersets of that node.

Flow of Apriori and Association Rule Algorithm:

Packages used: Pandas (0.20.1), itertools

Here are the steps that we followed to implement Apriori:

1. Import the given file and modify the item names according to the desired output.
2. Set the minimum support and confidence values for the given frequent itemset.
3. Frequent itemset of length 1 are generated from the columns of the given dataset.
4. Generate all possible combinations of the data items using the itertools library.
5. Iterate through the combinations. Prune the candidate itemsets, if they have a support below the min support.

Here are the steps that we followed to implement Association Rule Algorithm:

1. Read the query from the command line and parse the query using the substring functions.

2. Call the corresponding template module and perform the requested operations mentioned in the query.

3. For the template 1, we can perform the following set of operations on either of RULE, BODY, or HEAD:
Template Query: asso_rule.template1("RULE/BODY/HEAD", "ANY/NONE/1", “item”)
a) ANY: Fetch all the frequent itemsets which have the given item.
b) NONE: Fetch all the frequent itemsets which do not have given itemset.
c) 1: Fetch all the frequent itemsets which have exactly one of the any of given items.

4. For the template 2, we can perform the following set of operations on either RULE, BODY or HEAD:
Template Query: asso_rule.template2("RULE/BODY/HEAD", item_count)
a) Fetch all the frequent itemsets which have total number of items >= item_count

5. For template 3, we can perform following operations on either RULE, BODY or HEAD:
Template Queries: asso_rule.template3("1or2", "BODY", "ANY", ['G10_Down'], "HEAD", 2)
There will be two sets of operations for the template 3 queries. Perform either of ‘AND’ or ‘OR’ operations on the results of those two parts according to the given query.

6. Return the result of the query in a dataframe.

RESULTS:

Support = 30%, Confidence = 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
Total frequent itemsets: 12879

Support = 40%, Confidence = 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
Total frequent itemsets: 1077
2528 rules generated.

Support = 60%, Confidence = 60%
number of length-1 frequent itemsets: 34
number of length-2 frequent itemsets: 2
Total frequent itemsets: 36
4 rules generated.

Support = 70%, Confidence = 70%
number of length-1 frequent itemsets: 7
Total frequent itemsets: 7
0 rules generated.

Support = 50%, Confidence = 70%
Number of length-1 frequent itemsets: 109
Number of length-2 frequent itemsets: 63
Number of length-3 frequent itemsets: 2
Total frequent itemsets: 174
117 rules generated.

Template 1:

1) *Query : asso_rule.template1("RULE", "ANY", ['G59_Up'])*
   Result: 26 rows selected.

2) *Query: asso_rule.template1("RULE", "NONE", ['G59_Up'])*
   Result: 91 rows selected.

3) Query: asso_rule.template1("RULE", 1, ['G59_Up', 'G10_Down'])
   Result: 39 rows selected.

4) *Query: asso_rule.template1("BODY", "ANY", ['G59_Up'])*
   Result: 9 rows selected.

5) *Query: asso_rule.template1("BODY", "NONE", ['G59_Up'])*
   Result: 108 rows selected

6) *Query: asso_rule.template1("BODY", 1, ['G59_Up', 'G10_Down'])*
   Result: 17 rows selected

7) *Query: asso_rule.template1("HEAD", "ANY", ['G59_Up'])*
   Result: 17 rows selected

8) *Query: asso_rule.template1("HEAD", "NONE", ['G59_Up'])*
   Result: 100 rows selected

9) *Query: asso_rule.template1("HEAD", 1, ['G59_Up', 'G10_Down'])*
   Result: 24 rows selected

Template 2:

1) *Query: asso_rule.template2("RULE", 3)*
   Result: 9 rows selected
2) *Query: asso_rule.template2("BODY", 2)*
   Result: 6 rows selected
3) *Query: asso_rule.template2("HEAD", 1)*
   Result: 117 rows selected

Template 3:

1) *Query: asso_rule.template3("1or1", "BODY", "ANY", ['G10_Down'], "HEAD", 1, ['G59_Up'])*
   Result: 24 rows selected
2) *Query: asso_rule.template3("1and1", "BODY", "ANY", ['G10_Down'], "HEAD", 1, ['G59_Up'])*
   Result: 1 rows selected
3) *Query: asso_rule.template3("1or2", "BODY", "ANY", ['G10_Down'], "HEAD", 2)*
   Result: 11 rows selected
4) *Query: asso_rule.template3("1and2", "BODY", "ANY", ['G10_Down'], "HEAD", 2)*
   Result: 0 rows selected
5) *Query: asso_rule.template3("2or2", "BODY", 1, "HEAD", 2)*
   Result: 117 rows selected
6) *Query: asso_rule.template3("2and2", "BODY", 1, "HEAD", 2)*
   Result: 3 rows selected