10/2/2018

Apriori Algorithm Report

Submitted By:

Kautuk Desai	50247648
Sujay Vijay Purandare	50205931
Palwinder Singh	50247454

Apriori Algorithm

This algorithm is used for frequent item set mining and association rule generation on transactional databases. The key idea in this algorithm is that any subset of a frequent item set must also be frequent and any superset of an infrequent item set is also infrequent. This idea is used to prune non-frequent item sets in each set as per the given support.

Apriori Workflow

1. We have implemented the Apriori Workflow in Java by dividing it into following 3 tasks: Creating frequent item sets of different lengths: Function fillItemSupport(support) creates a list of frequent item sets of different length as per the given support. It starts with one length frequent item sets and iteratively creates n+1 frequent item set using function upgrade () where n is a length of each item set in the previous step. In each step, item sets whose support is less than given support are pruned using function pruneItems(support).

2. Creating rules from frequent item sets: After getting the list of frequent item sets of different lengths from previous operation, we call createAllRules() to generate rules. This function traverses on each frequent item set and generates permutations of items in that set using function createPermutations(frequentItemSet). Each of this permutation is then passed to function createRules(permutations) which creates rules by creating different head and bodies on the given permutation.

3. Pruning rules that doesn't match the given confidence: After generating the rules from previous operation, these rules are passed to the function pruneRules(confidence). This function traverses over each rule and calculates its confidence using formula confidence_rule = support of all items in rule/support of items in head. It then checks removes all the rules from list whose confidence_rule is less than the given confidence.

Results

Part 1 Result:

1)Support = 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 all lengths frequent itemsets= 12879

2)Support = 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

3)Support =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

4)Support =60%

- Number of length-1 frequent itemsets= 34
- Number of length-2 frequent itemsets= 2
- Number of all lengths frequent itemsets= 36

5)Support = 70%

- Number of length-1 frequent itemsets= 7
- Number of all lengths frequent itemsets= 7

Number of rules for different supports and confidence 70%

- Number of rules for support 30% and confidence 70% = 31759
- Number of rules for support 40% and confidence 70% = 1137
- Number of rules for support 50% and confidence 70% = 117
- Number of rules for support 60% and confidence 70% = 4
- Number of rules for support 70% and confidence 70% = 0

Number of rules for support 50% and confidence 70%

Sample Template 1 outputs

- (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

Sample Template 2 outputs

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

Sample Template 3 outputs

- (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