ISSN 2250-0987
Parita Parikh et al, UNIASCIT, Vol 2 (1), 2012, 170-172

Comparative Study of Association Rule Mining Algorithms

Parita Parikh Dinesh Waghela
Computer Engineering Department Computer Engineering Department
PIET, GTU PIET, GTU
Gujarat, India Gujarat, India

Abstract - Association rule mining, one of the most important techniques of data mining. It aims to extract interesting correlations, frequent patterns, associations or casual structures among sets of items in the transaction databases or other data repositories. Association rules are widely used in various areas such as telecommunication networks, market and risk management, inventory control etc. This paper represents comparative study of association rule mining algorithm.

Keywords–Association rule mining, Apriori, AprioriTid, AprioriHybrid

I. INTRODUCTION

Association rule mining is to find out association rules that satisfy the predefined minimum support and confidence from a given database.

The problem is usually decomposed into two sub problems. One is to find those itemsets whose occurrences exceed a predefined threshold in the database; those itemsets are called frequent or large itemsets. The second problem is to generate association rules from those large itemsets with the constraints of minimal confidence. Support and confidence are important measures for association rules.

Generally, an association rules mining algorithm contains the following steps:

i. The set of candidate k-itemsets is generated by 1-extensions of the large (k -1) itemsets generated in the previous iteration.

ii. Supports for the candidate k-itemsets are generated by a pass over the database.

iii. Itemsets that do not have the minimum support are discarded and the remaining itemsets are called large k-itemsets.

II. APRIORI ALGORITHM

The first pass of the Apriori algorithm simply counts item occurrences to determine the large 1-itemsets. A subsequent pass, say pass k, consists of two phases. First, the large itemsets Lk-1 found in the (k-1)th pass are used to generate the candidate itemsets Ck, then the database is scanned and the support of candidates in Ck is counted. For fast counting, we need to efficiently determine the candidates in Ck that are contained in a given transaction. Where Lk represents Set of large k-itemsets and Ck represents Set of candidate k-itemsets.

Join Step: Ck is generated by joining Lk-1 with itself.

Prune Step: Any (k-1)-itemset that is not frequent cannot be a subset of a frequent k-itemset

Algorithm

L1 = {large 1-itemsets};
for (k = 2; Lk-1 != Φ; k++) do begin
Ck = apriori-gen (Lk-1);
for all transactions t Є D do begin
Ct = subset (Ck , t);
for all candidates c Є Ct do
c.count++;
end
Lk = {c Є C | j c.count >= minsup}
End
Answer = Uk Lk;

The apriori-gen function takes as argument Lk-1, the set of all large (k - 1)-itemsets. It returns a superset of the set of all large k-itemsets. The function works as follows. First, in the join step, we join Lk-1 with Lk-1:

insert into Ck select p.item1, p.item2, ..., p.item k-1, q.item k-1 from Lk-1 p, Lk-1 q where p.item1 = q.item1, . . ., p.item k-2 = q.item k- 2,p.item k-1 < q.item k-1;

Next, in the prune step, we delete all itemsets c Є Ck such that some (k-1)-subset of c is not in Lk-1:

for all itemsets c Є Ck do
for all (k-1)-subsets s of c do
if (s !Є Lk-1) then
delete c from Ck;

Limitations of Apriori algorithms are:

1) Algorithm must spend a lot of time to deal with huge candidate item sets.

2) It must repeatedly scan the transaction database to carry out pattern matching for the candidate item sets.

III. AprioriTID ALGORITHM

The AprioriTid algorithm also uses the apriori-gen function to determine the candidate itemsets before the pass begins. The interesting feature of this algorithm is that the database D is not used for counting support after the first pass.

Algorithm

L1 = {large 1-itemsets};
D1 = database D;
for (k = 2; Lk-1 != Φ ; k++) do begin
Ck = apriori-gen (Lk-1);
Dk = Φ;
for all entries tЄ2 Dk-1 do begin
Ct = { c Є Ck | (c - c[k]) Є t:set-of-itemsets ^ (c - c[k-1]) Є t.set-of-itemsetsg;
for all candidates c Є Ct do
c.count++;
if (Ct != Φ) then Dk += < t.TID,Ct >;
end
Lk = {c Є Ck | c.count >= minsup}
End
Answer = Uk Lk;

IV. AprioriHybrid ALGORITHM

It is not necessary to use the same algorithm in all the passes over data.

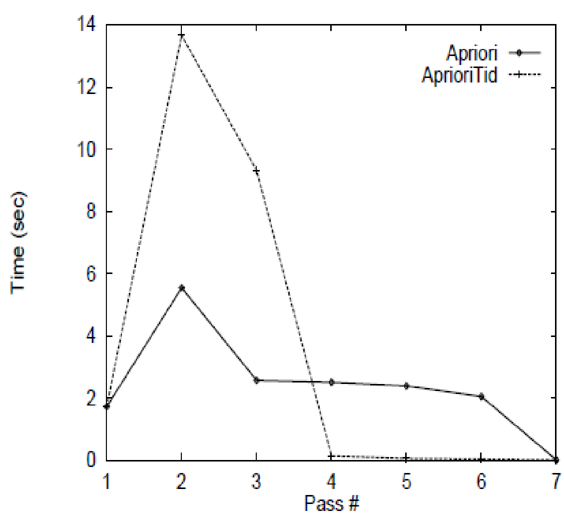

Figure 1 Per pass execution times of Apriori and AprioriTid

Figure 1 shows the execution times for Apriori and AprioriTid for different passes. In the earlier passes, Apriori does better than AprioriTid. However, AprioriTid beats Apriori in later passes, the reason for which is as follows. Apriori and AprioriTid use the same candidate generation procedure and therefore count the same itemsets. In the later passes, the number of candidate itemsets reduces However; Apriori still examines every transaction in the database. On the other hand, rather than scanning the database, AprioriTid scans Ck for obtaining support counts, and the size of Ck has become smaller than the size of the database.Based on these observations AprioriHybrid algorithm has been designed. This uses Apriori in the initial passes and switches to AprioriTid in the later passes.

Characteristics	Apriori	AprioriTid	Apriori Hybrid
Data Support	**Limited**	Often support large	Very Large
Speed in initial phase	High	Slow	High
Speed in later phases	Slow	High	High
Accuracy	Less	More accurate than Apiori	More accurate than AprioriTid

Table 1 Comparison of Apriori, AprioriTid and AprioriHybrid

V. CONCLUSION

This paper represents comparison of three association rule mining algorithms: Apriori, AprioriTid and AprioriHybrid. The AprioriTid and AprioriHybrid have been proposed to solve the problem of apriori algorithm. From the comparison we conclude that the AprioriHybrid is better than Apriori and AprioriTid, because it reduced overall speed and improve the accuracy.

REFERENCES

[1] R. Agrawal and R. Srikant. Fast algorithms for mining association rules in large databases. Research Report RJ 9839, IBM Almaden Research Center, San Jose, California, June 1994.

[2] R. Agrawal, T. Imielinski, and A. Swami Database mining: A performance perspective.IEEE Transactions on Knowledge and Data Engineering, 5(6):914{925, December 1993. Special Issue on Learning and Discovery in Knowledge Based Databases.

[3] J. Han, Y. Cai, and N. Cercone. Knowledge discovery in databases: An attribute oriented approach. In Proc. of the VLDB Conference, pages 547{559, Vancouver, British Columbia, Canada, 1992.

[4] M. Holsheimer and A. Siebes. Data mining: The search for knowledge in databases. Technical Report CS-R9406, CWI, Netherlands, 1994.

[5] M. Houtsma and A. Swami. Set-oriented mining of association rules. Research Report RJ 9567, IBM Almaden Research Center, San Jose, California, October 1993.

[6] R. Krishnamurthy and T. Imielinski. Practitioner problems in need of database research: Research directions in knowledge discovery. SIGMOD RECORD, 20(3):76{78, September 1991.

[7] P. Langley, H. Simon, G. Bradshaw, and J. Zytkow. Scienti_c Discovery: Computational Explorations of the Creative Process. MIT Press, 1987.

[8] H. Mannila and K.-J. Raiha. Dependency inference. In Proc. of the VLDB Conference, pages 155{158, Brighton, England, 1987.

[9] H. Mannila, H. Toivonen, and A. I. Verkamo. Efficient algorithms for discovering association rules. In KDD-94: AAAI Workshop on Knowledge Discovery in Databases, July 1994.