Jurnal Teknologi

Full Paper

MINING DENSE DATA: ASSOCIATION RULE DISCOVERY ON BENCHMARK CASE STUDY

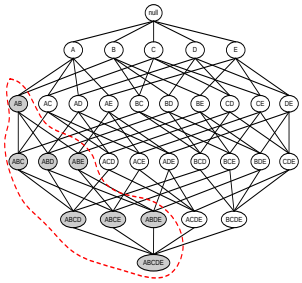
Wan Aezwani Wan Abu Bakar[a*], Md. Yazid Md. Saman[a], Zailani Abdullah[a], Masita@Masila Abd Jalil[a], Tutut Herawan[b]

[a]School of Informatics and Applied Mathematics, Universiti Malaysia Terengganu, 21030 Kuala Terengganu, Terengganu, Malaysia
[b]Department of Information Systems, Faculty of Computer Science and Information Technology, University of Malaya, Lembah Pantai, 50603 Kuala Lumpur, Malaysia

Article history
Received
15 June 2015
Received in revised form
1 October 2015
Accepted
13 October 2015

*Corresponding author
beny2194@yahoo.com

Graphical abstract



Abstract

Data Mining (DM), is the process of discovering knowledge and previously unknown pattern from large amount of data. The association rule mining has been in trend where a new pattern analysis can be discovered to project for an important prediction about any issues. In this article, we present comparison result between Apriori and FP-Growth algorithm in generating association rules based on a benchmark data from frequent itemset mining data repository. Experimentation with the two (2) algorithms are done in Rapid Miner 5.3.007 and the performance result is shown as a comparison. The results obtained confirmed and verified the results from the previous works done.

Keywords : Data Mining (DM), Association Rule Mining (ARM), Rapid Miner (RM), frequent itemset, interestingness measure

Abstrak

Perlombongan Data (DM), adalah proses mencari ilmu dan corak sebelum ini tidak diketahui dari jumlah data yang besar. Perlombongan peraturan persatuan telah trend di mana analisis corak baru boleh didapati projek untuk sebuah ramalan penting tentang sebarang isu. Dalam artikel ini, kami membentangkan hasil perbandingan antara apriori dan algoritma FP- Growth menjana peraturan persatuan berdasarkan data penanda aras dari itemset kerap repositori data perlombongan. Uji kaji dengan dua (2) algoritma dilakukan di Rapid Miner 5.3.007 dan keputusan prestasi ditunjukkan sebagai perbandingan. Keputusan yang diperolehi disahkan dan disahkan hasil dari kerja-kerja yang dilakukan sebelum ini .

Kata kunci: Perlombongan Data (DM), Persatuan Peraturan Mining (ARM) , Rapid Miner (RM), itemset kerap, langkah interestingness

© 2016 Penerbit UTM Press. All rights reserved

1.0 INTRODUCTION

Data mining is the research area where the huge dataset in database and data repository is being scoured and mined to find novel and useful pattern [1]. Association analysis is one of the four (4) core data mining tasks besides cluster analysis, predictive modeling and anomaly detection [2]. The task of association rule mining is to discover if there exist the frequent itemset or pattern in database and if any, an interesting relationships between these frequent itemsets can reveal a new pattern analysis for the

future decision making. Finding frequent itemsets or patterns as shown in Figure 1 by [3] is a big challenge and has a strong and long-standing tradition in data mining. It is a fundamental part of many data mining applications including market basket analysis, web link analysis, genome analysis and molecular fragment mining.

The idea of mining association rule originates from the analysis of market basket data [4]. Example of simple rule is *A customer who buys bread and butter will also tend to buy milk with probability s% and c%.* The applicability of such rule to business problems makes the association rule to become a popular mining method.

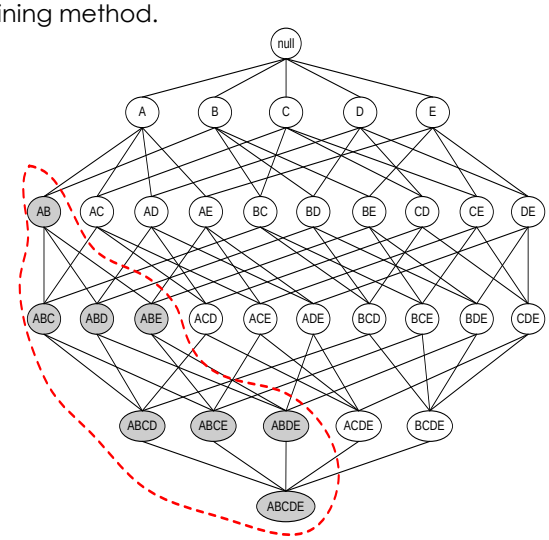


Figure 1 Frequent itemset and its subset

Following is the formal definition of the problem [4]. Let I = {i_1, i_2,...i_m} be the set of items. D is a set of transaction where each transaction T is a set of items such that $T \subseteq I$. An association rule is an implication of the form $X \subseteq Y$ where X represent the antecedent part of the rule and Y represents the consequent part of the rule where $X \subseteq I$, $Y \subseteq I$ and $X \cap Y = \emptyset$. The itemset that satisfies minimum support is called frequent itemset. The rule $X \Rightarrow Y$ holds in the transaction set D with confidence c if c% of transactions in D that contain X also contain Y. The rule $X \Rightarrow Y$ has support s in the transaction set D if s% of transaction in D contain $X \cup Y$. The illustration of support-confidence framework is given as below:

a) The support of rule $X \Rightarrow Y$ is the fraction of transactions in D containing both X and Y.

$$Support\ (X \Rightarrow Y) = \frac{X \cup Y}{|D|}$$

$where\ |D| is\ the\ total\ number\ of\ records\ in\ database$

b) The confidence of rule $X \Rightarrow Y$ is the fraction of transactions in D containing X that also contain Y.

$$Confidence\ (X \Rightarrow Y) = \frac{supp\ (X \cup Y)}{supp\ (X)}$$

The rules satisfy both a minimum support threshold (min_supp) and minimum confidence threshold (min_conf) are called strong rule where min_supp and min_conf are user specified values.

2.0 RELATED WORKS

Since the introduction of frequent itemset mining by [4], it has received a major attention among researchers and various efficient and sophisticated algorithms have been proposed to do frequent itemset mining. Among the best-known algorithms are Apriori and FP-Growth with their different searching strategies.

The Apriori algorithm [4, 5] uses a breadth-first search and the downward closure property, in which any superset of an infrequent itemset is infrequent, to prune the search tree. Apriori usually adopts a horizontal layout to represent the transaction database and the frequency of an itemset is computed by counting its occurrence in each transaction. Apriori uses a "bottom up" approach, where frequent subsets extend one item at a time (a step known as *candidate generation*, and groups of candidates tested against the data). The algorithm terminates when no further successful extensions are found. The key idea is such that the apriori property (downward closure property) states that any subsets of a frequent itemset are also frequent itemsets. The best known algorithm that involve two steps:

- *Step 1 : Find all itemsets that have minimum support (frequent itemsets, also called large itemsets).*
- *Step 2 : Use frequent itemsets to generate rules.*

The FP-Growth [6] uses a depth-first search and employs a divide-and-conquer strategy with an FP-tree data structure to achieve a condensed representation of the transaction database. It has become one of the fastest algorithms for frequent pattern mining. In large databases, it's not possible to hold the FP-tree in the main memory. A strategy to cope with this problem is to firstly partition the database into a set of smaller databases (called projected databases), and then construct an FP-tree from each of these smaller databases. The steps are as follows:

- *Step 1 : Scan DB once, find frequent 1-itemset (single item pattern)*
- *Step 2 : Sort frequent items in frequency descending order, f-list*
- *Step 3 : Scan DB again, construct FP-tree*

A comparison work done in [7] discovers the performance of the FP-Growth algorithm is not influenced by the support factor while the performance of Apriori algorithm decreases with support factor.

In evaluating the performance of association rule mining algorithms, an experiment by [8], FP-Growth algorithm requires less processing time regardless of the number of instances used as compared to Apriori.

The performance of Apriori and FP-Growth algorithms are interpreted using statistical representation in [9] where the author uses One Sample Kolmogorov-Smirnov test to determine the distribution of three min_support values chosen.

3.0 RESEARCH METHODOLOGY

3.1 Experimentation Platform and Datasets

All experiments are performed on a DELL Inspiron 620, Intel ® Pentium ® CPU G630 @ 2.70 GHz with 4GB RAM in a Win 7 64-bit platform. The tool used is Rapid Miner (RM) 5.3.007. The raw benchmark data are retrieved from http://fimi.ua.ac.be/data/ in a *.dat file format. For the ease of use in RM, we convert to Comma Separated Value (CSV) format. For experimentation purposes, the dataset is first 'cleaned', where in RM itself, we have to perform data transformation in order to be processed through the specified algorithm. When importing data into RM, we have to specify what parameter to be set as ID, label, or attributes. There are five (5) datasets include chess, connect, mushroom, pumb_star and T40I10D100K. Table 1 shows the characteristics of datasets.

Table 1 Database characteristics

Database	Size (KB)	Average Length (attribute)	Records (transaction)
chess	335	37	3196
connect	9039	43	67558
mushroom	558	23	8125
Pumb_star	11028	50	49047
T40I10D100K	15116	32	100001

3.2 RM Development and Results

Figure 2 illustrates the processes involved in deploying apriori algorithm. The W-Apriori process is an extension of Weka-Apriori into the RM tool. First, the benchmark data (in csv) is retrieved by calling *retrieve()* process. Then data transformation has to be constructed where *descretizebyfrequency()* process is called. This operator converts the selected numerical attributes into nominal attributes by discretizing the numerical attribute into a user-specified number of bins. Bins of equal frequency are automatically generated, the range of different bins may vary. Then data is converted from *nominaltonumerical()* to *numericalto polynominal()*. The process *nominaltonumerical()* is to change the nominal attributes to numerical attributes while the process *numericaltopolynominal()* is to change the numerical attributes to polynominal attributes, that is allowed in Apriori algorithm. Then we call the Weka extension, *W-Apriori()* to generate the best rules. The parameter is set to be a default value.

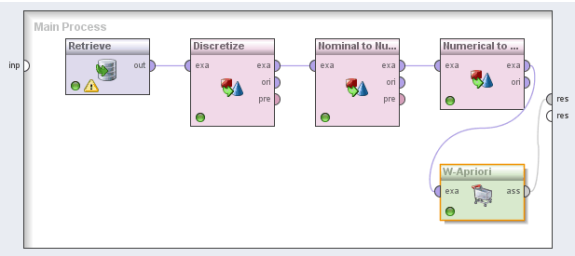


Figure 2 Rapid Miner processes by W-Apriori algorithm

Figure 3 depicts the processes involved in deploying the Weka extension W-FPGrowth algorithm. The root process starts with retrieving the csv dataset. Then the *discretizeby frequency()* is selected to change the real attributes to nominal. Next, the *NominaltoBinominal()* process is called to change the nominal attributes to binominal attributes which is allowed in FPGrowth algorithm and lastly *W-FPGrowth()* process is called to find frequent pattern and generate the rules.

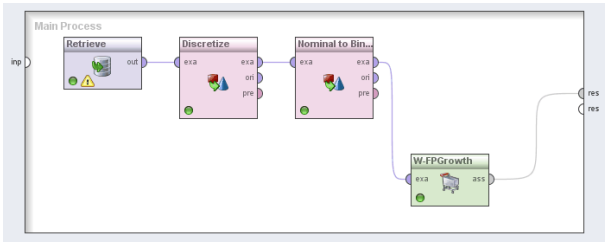


Figure 3 Rapid Miner processes by W-FPGrowth algorithm

The performance of the two algorithms (Apriori and FP-Growth) is measured in terms of total execution time and total generated rules. The running time is subjected to factors such as different search method in both algorithms and also the size of dataset itself. Table 2, 3 and 4 depict the summary of results obtained prior to running of the datasets through Apriori and FP-Growth algorithm within 3 specified confidence thresholds. The upper bound of min_supp is set to 1.0 and lower bound of min_supp is set to 0.1.

Table 2 Result obtained when min_conf is 0.9

Data (*.csv)	W-Apriori		W-FPGrowth	
	Time (s)	Total Rule	Time (s)	Total Rule
Chess	1	10	0	29
Connect	24	10	9	16
Mushroom	1	10	0	16
Pumb_star	20	10	6	18660
T40I10D100K	20	10	10	32

Table 3 Result obtained when min_conf is 0.5

Data (*.csv)	W-Apriori		W-FPGrowth	
	Time (s)	Total Rule	Time (s)	Total Rule
Chess	1	10	0	16
Connect	23	10	9	12
Mushroom	1	10	0	18
Pumb_star	20	10	7	18660
T40I10D100K	21	10	11	32

Table 4 Result obtained when min_conf is 0.1

Data (*.csv)	W-Apriori		W-FPGrowth	
	Time (s)	Total Rule	Time (s)	Total Rule
Chess	0	10	0	20
Connect	22	10	10	36
Mushroom	1	10	0	18
Pumb_star	22	10	8	18660
T40I10D100K	22	10	12	32

Figure 4 to figure 9 illustrate the graphs of the results obtained. The six (6) figures representing three (3) different values of min_conf (i.e. 0.9, 0.5 and 0.1). In Figure 4 and Figure 5, when min_conf is set to 0.9, W-FPGrowth outperforms W-Apriori in lesser execution time but shows more number of rules generated with all five (5) datasets tested whereas W-Apriori depicts more time execution but with only ten (10) total rules generated.

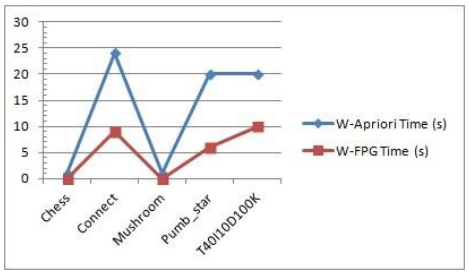


Figure 4 W-Apriori vs. W-FPGrowth: Execution time when min_conf = 0.9

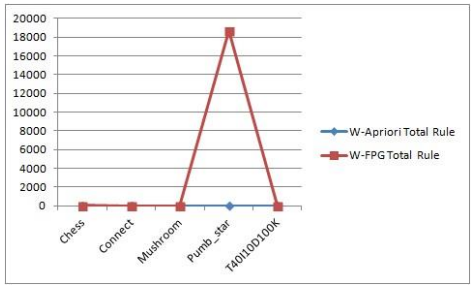


Figure 5 W-Apriori vs. W-FPGrowth: Rules generated when min_conf = 0.9

In Figure 6 and Figure 7, the results visualized quite similar trends with previous figures wherein the min_conf is set to 0.5, then all datasets reveal W-FPGrowth still outperforms W-Apriori with lesser execution time but more number of rules generated. The graphs in Figure 8 and Figure 9 illuminate similar patterns when compared to previous figures even when the min_conf is set to 0.1 respectively.

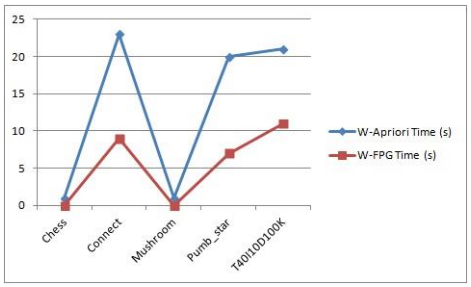


Figure 6 W-Apriori vs. W-FPGrowth: Execution time when min_conf = 0.5

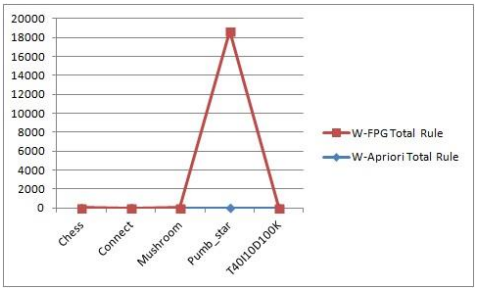


Figure 7 W-Apriori vs. W-FPGrowth: Rules generated when min_conf = 0.5

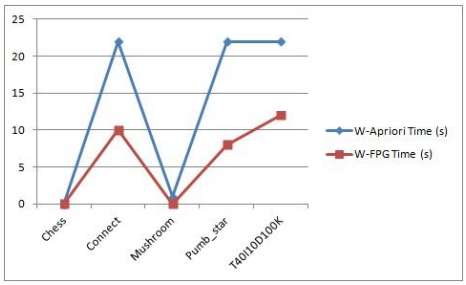


Figure 8 W-Apriori vs. W-FPGrowth: Execution time when min_conf = 0.1

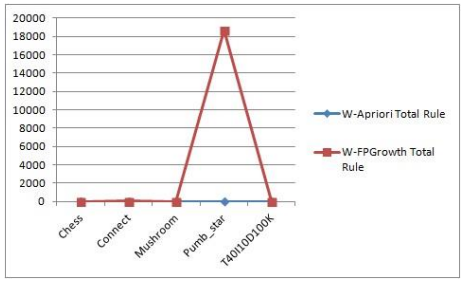


Figure 9 W-Apriori vs. W-FPGrowth: Rules generated when min_conf = 0.1

It realizes that the patterns plotted in those six (6) figures are almost similar. The graphs show that W-FPGrowth outperforms W-Apriori where more number of rules generated within lesser execution time. From the detailed result in RM, between W-Apriori and W-FPGrowth, there are almost similar attributes interpreted to be the antecedent and consequent. With W-FPGrowth, there are more attributes found to be the interesting rules as compared to W-Apriori. Nevertheless for any mining algorithm, it should find the same set of rules although their computational efficiencies and memory requirements may be different [6].

4.0 CONCLUSION

The experiment conducted in this paper shows a comparison results between Apriori algorithm and FPGrowth algorithm using the benchmark dense databases experiences with Rapid Miner 5.3.007. Regardless of how many rules the algorithms can generate, RM will only display the best top 10 rules. This illustration from the graphs show FPGrowth outperforms W-Apriori in terms of lesser execution time with more rules generated. The W-FPGrowth is found to be better algorithm when encountering the support-confidence framework. Originally, W-FPGrowth only performs two (2) passes over datasets and, the datasets are already "compressed" with the generation of FP-Tree by reducing irrelevant information. This is done through removing infrequent items. While for W-Apriori, it requires multiple database scans and a candidate generation approach by self-joining (by generating all possible candidate itemsets with up to 2^k - 2 candidates in total) before pruning (by removing those candidates that is not frequent). Therefore, it results in more execution time.

There are many other interestingness measure that can be imposed to the algorithm and see whether the performance result between Apriori and FPGrowth are still the same or otherwise. For the future analysis, there are few alternatives we might want to tackle either in the same interestingness measure with vertical data format approach (i.e. Eclat algorithm) [10] or with different interestingness measure but with similar ARM algorithms and compare the outcome.

Acknowledgement

We wish to thank Mustafa Man for his insightful comments and suggestions and credit also to MyPhD scholarship under MyBrain15 of Kementerian Pendidikan Malaysia (KPM) for the financial foundation of this work.

References

[1] Man, M., Rahim, M., Jusoh, J., and Zakaria, M. Z. 2011. Designing Multiple Types Of Spatial And Non Spatial Databases Integration Model Using Formal Specification Approach. *Software Engineering (MySEC), 2011 5th Malaysian Conference in. IEEE*. 20–24.

[2] Tan, P.-N., Steinbach, M., and Kumar, V. 2005. *Introduction to Data Mining*. First Edition. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc.

[3] Tuan, T. A. 2012. A Vertical Representation For Parallel Declat Algorithm In Frequent Itemset Mining. Master's thesis, Ritsumeikan University.

[4] Agrawal, R., Srikant *et al.,* R. 1994. Fast Algorithms For Mining Association Rules. *Proc. 20th Int. Conf. Very Large Data Bases, VLDB*. 1215: 487-499.

[5] Agrawal, R., Imieli´nski, T., and Swami, A. 1993. Mining Association Rules Between Sets Of Items In Large Databases, *SIGMOD Rec*. 22(2): 207-216. [Online]. Available: http://doi.acm.org/10.1145/170036.170072.

[6] Han, J., Kamber, M. and Pei, J. 2006. *Data Mining: Concepts And Techniques*. Morgan Kaufmann.

[7] Gy˝orödi, C., Gy˝orödi, R. and Holban, S. 2004. A Comparative Study Of Association Rules Mining Algorithms. *SACI 2004, 1st Romanian-Hungarian Joint Symposium on Applied Computational Intelligence*. 213-222.

[8] Vanitha, K. 2011. Evaluating The Performance Of Association Rule Mining Algorithms. *Journal of Global Research in Computer Science*. 2(6): 101-103.

[9] Hunyadi, D. 2011. Performance Comparison Of Apriori And Fp-Growth Algorithms In Generating Association Rules, *Proceedings of the European Computing Conference*. 376-381.

[10] Man, M., Jusuh, J. A., Rahim, M. S. M., Zakaria, M. Z. 2011. Formal Specification For Spatial Information Databases Integration Framework (SIDIF). *Telkomnika*. 9(1): 81-88.