EAST WEST UNIVERSITY

Project Report

Course Title: Data Mining

Course Code: CSE477

Section: 1

Submitted to:

Jesan Ahammed Ovi

Senior Lecturer

Department of Computer Science and Engineering

East West University

GROUP MEMBERS:

Sk. Amir Hamza	2017-1-60-091
Avia Anwar	2017-1-60-154
Tanim Hasan Mahmud	2017-1-60-130
Md. Abuhorayra	2017-2-60-077
Swakshar Debnath	2017-2-60-034

Comparison Of Fp-growth and Apriori Algorithm On Chess and Mushroom Dataset

Chess Dataset

The dataset contains 3,196 transactions and the total number of items is 75. Each row shows a transaction. In rows, the items are separated by a space between them. There were no odd values or null values in the dataset. Our task was to apply Apriori and Fp -growth algorithms to this dataset. We prepared the data by separating the items with commas. So that we can apply both apriori and Fp-growth algorithms to it.

Apriori Algorithm on Chess Dataset

At the time of applying Apriori algorithm, the minimum support threshold was 60% and we increased it by 5% with each iteration up to 100%. At first, we took the support threshold of 50% which caused a runtime error. The memory was overflowing. Cause the platform we were using couldn't handle the patterns generated by 50% threshold.

In the graph, we have considered the minimum threshold in X-axis and time complexity in Y-axis. We can see in Figure 01, the computation time is very high in the first couple of iterations. But as we keep increasing the minimum threshold, the run time decreases drastically. This is because more and more patterns get eliminated as we keep increasing the minimum threshold. And the fewer the number of patterns is, the less time Apriori will take to scan the database. So the run time gets lower when we increase the threshold.

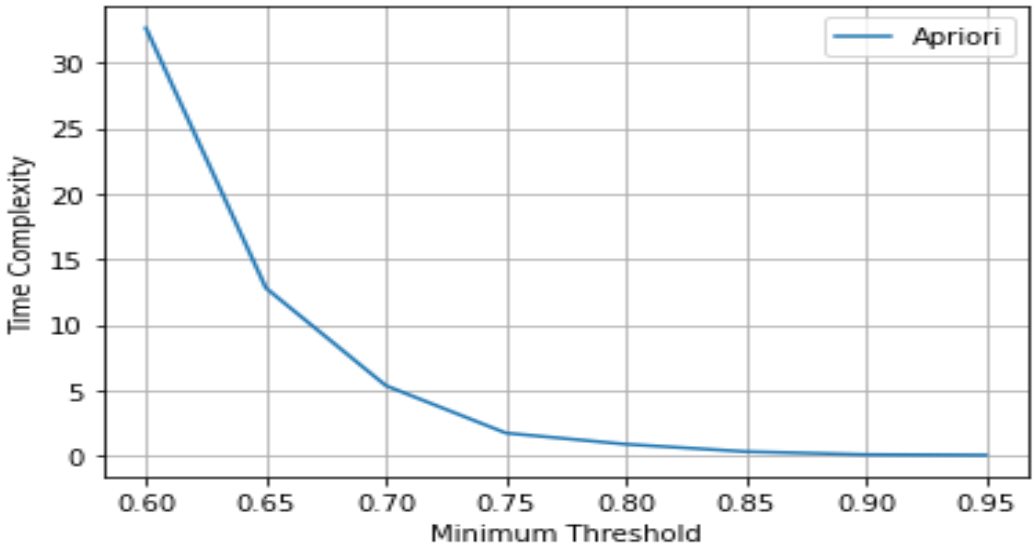

Figure 01: Line graph illustrating time complexity of the Apriori algorithm

Fp-Growth Algorithm on Chess Dataset

In this part, we have used the same support threshold and increased it with the same value, and performed Fp-growth. But the time complexity is very less and with every iteration, it decreases.

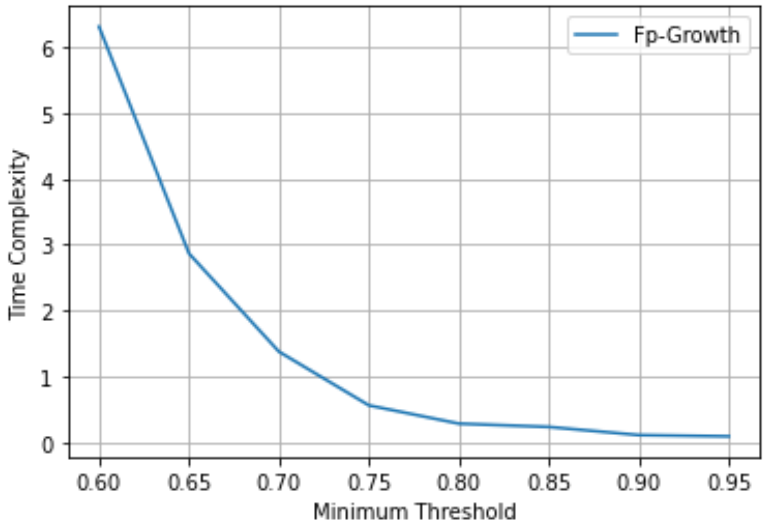

Figure 02: Line graph illustrating time complexity of the Fp-growth algorithm

Comparing Apriori and Fp-growth on Chess

After running two algorithms, we compared their performance according to their time complexity under each threshold. We summarize and compare their results in the following table.

Threshold	Apriori Algorithm(in secs)	Fp-growth Algorithm(in secs)
60%	32.65s	6.31s
65%	12.8s	2.86s
70%	5.34s	1.38s
75%	1.73s	0.55s
80%	0.88s	0.28s
85%	0.31s	0.23s
90%	0.09s	0.11s
95%	0.05s	0.09s

Table 01. The time complexity of the Apriori and Fp Growth algorithms on the chess dataset is based on the minimum threshold

We have plotted a line graph combining both of these algorithms in Figure 03. So that we can visualize the differences easily.

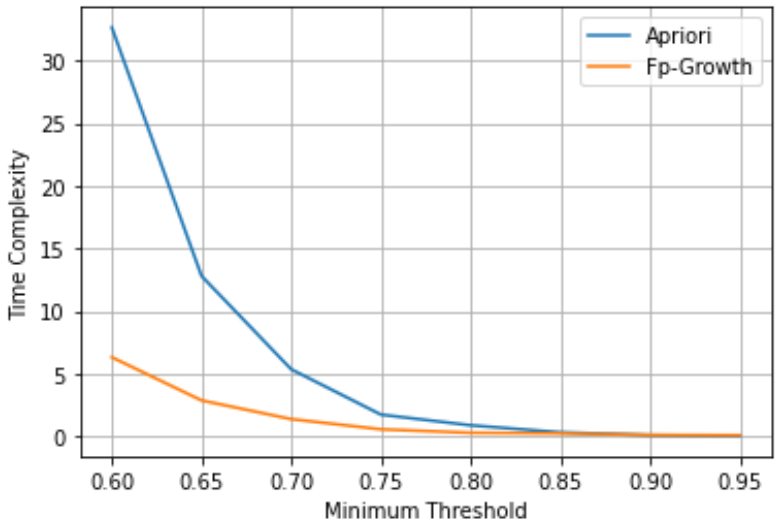

Figure 03: Comparing time complexity of Apriori and Fp-growth Algorithm on Chess Dataset

A bar chart in Figure 04 clearly describes the domination of Fp-growth over Apriori for the chess dataset.

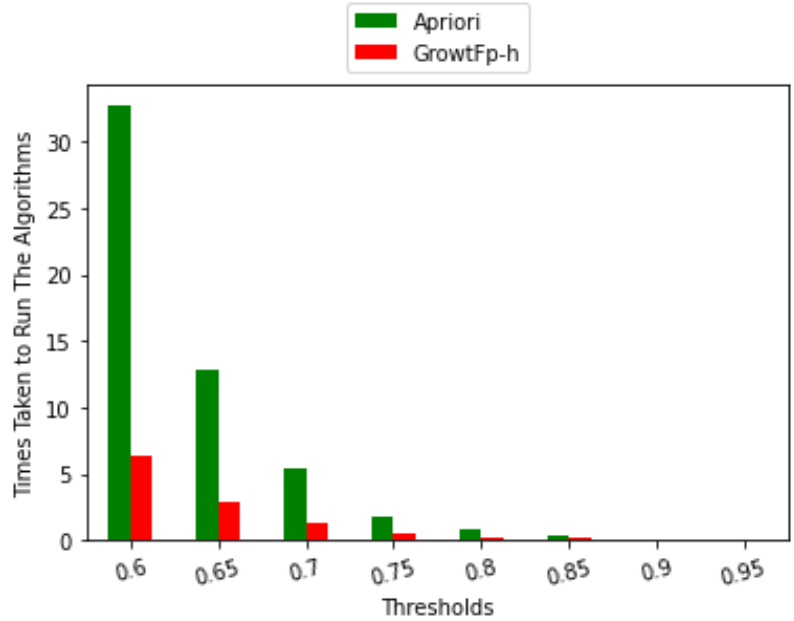

Figure 04: Visualization of comparison using bar chart

Here we can see a proper visualization of the time complexity of both Apriori and Fp-growth. The combination of Figure 02 and Figure 01 is shown in Figure 03. Figure 04 bar chart was used to visualize the comparison. Here we can observe that the Fp-growth algorithm is more efficient than the Apriori algorithm. For the chess dataset, Fp-growth works almost 6 times better than Apriori. In Fp Growth the database is scanned only 2 times to create trees and patterns. While the Apriori algorithm requires multiple scans to generate candidate sets. Therefore, there is a run time difference between the two algorithms, as we observed in the chess dataset above. The time complexity is very similar, ranging from 85% to 95%. This is due to the higher threshold. We got the less frequent pattern itemsets. So the apriori algorithm performed better. Still, the Fp-growth has given better results.

Mushroom Dataset

The dataset contains 8,124 transactions and the total number of items is 119. Each row shows a transaction.In rows the items are separated by a space between them. There were no odd values or null values in the dataset.Our task was to apply Apriori and Fp-growth algorithms on this dataset .We prepared the data by separating the items with commas. So that we can apply both Apriori and Fp-growth algorithms on it. And Fp-growth performs better in this dataset.

Apriori Algorithm on Mushroom Dataset

At the time of applying Apriori algorithm, the minimum support threshold was 30% and we increased it with 5% with each iteration up to 100%.
In the graph, we have considered the minimum threshold in X-axis and time complexity in Y-axis. We can see in Figure 05, the computation time is very high in the first couple of iterations. But as we keep increasing the minimum threshold, the run time decreases. This is because more and more patterns get eliminated as we keep increasing the minimum threshold. And the fewer the number of patterns is, the less time Apriori will take to scan the database.

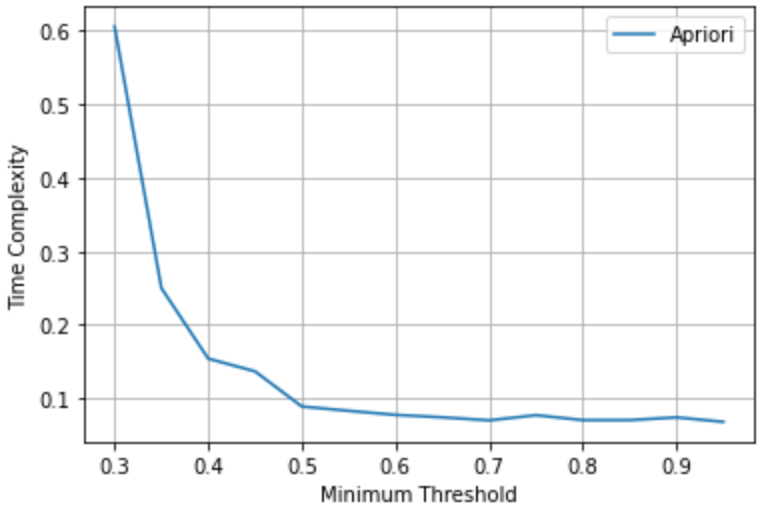

Figure 05: The time complexity of the apriori algorithm on Mushroom dataset

Fp-Growth Algorithm on Mushroom Dataset

In this part, we have used the same support threshold and increased with the same value.But the time complexity is higher compared to Apriori algorithm from 40%.

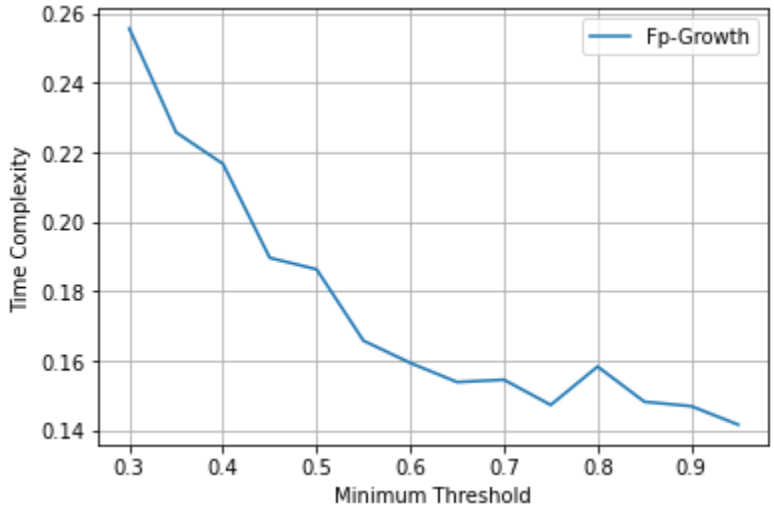

Figure 06: The time complexity of the Fp-Growth algorithm on Mushroom dataset

Comparing Apriori and Fp-growth on Mushroom Dataset

After running these two algorithms, we compared their performance according to their time complexity under each threshold. We summarize and compare their results in the following table.

Threshold	Apriori Algorithm(in secs)	Fp-growth Algorithm(in secs)
30%	0.60s	0.25s
35%	0.24s	0.22s
40%	0.15s	0.21s
45%	0.14s	0.18s
50%	0.08s	0.18s
55%	0.08s	0.16s
60%	0.07s	0.15s
65%	0.07s	0.15s
70%	0.07s	0.15s
75%	0.07s	0.14s
80%	0.07s	0.15s
85%	0.07s	0.14s
90%	0.07s	0.14s
95%	0.06s	0.14s

Table 02. The time complexity of the Apriori and Fp Growth algorithms on the Mushroom dataset is based on the minimum threshold

We can see from Table 02 that Fp-growth algorithm's performance is better than Apriori algorithm up to 35% minimum threshold. From Figure 07 it can be easily observed that at the beginning, that is from threshold 30% to 35% Fp-growth algorithm performs better. Whereas, apriori has runtime of 0.60s at 30% threshold and 0.24s at 35% threshold. For better visualization, we have plotted the combined Line Graph of both algorithms for Mushroom dataset in Figure 07.

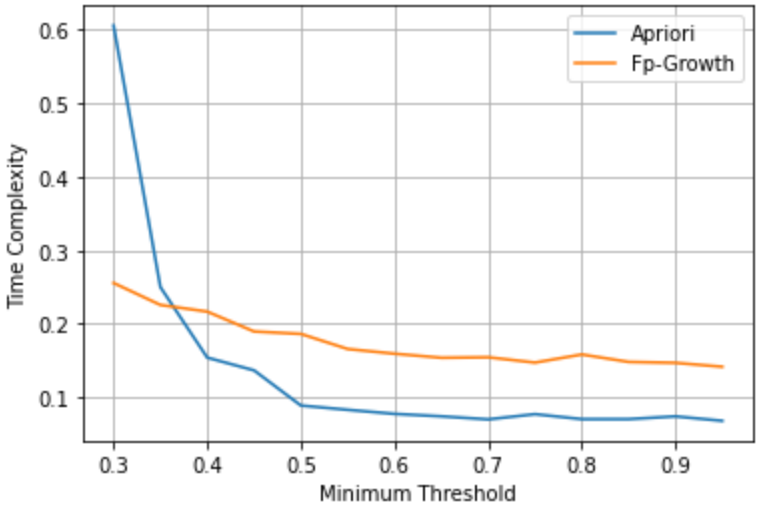

Figure 07: Comparing time complexity of Apriori and Fp-growth Algorithm on Mushroom Dataset

But in figure 07 we can see that Apriori outperforms Fp-growth algorithm form 40% of threshold. But we know that, the Fp-growth algorithm performs better than the Apriori algorithm in general. Apriori algorithm works better with a big dataset, while the FPGrowth Algorithm works better with small datasets [1,2]. In our experiment the Chess dataset it has 3,196 transactions and 75 unique items. And the Mushroom dataset outnumbers the chess dataset with its 8,124 transactions and 119 unique items.

FP Growth algorithm constructs conditional pattern tree and mines conditional patterns from the database which satisfies the minimum support. Its runtime is dependent on the compaction factor from the dataset [2]. If the conditional FP-Tree of the result has many branches, then the algorithm performance will be drop drastically because the algorithm has to process a lot of conditional FP-Tree. The complexity of FP-Growth is very dependent on the pathfinding in FP-Tree for every item in the header table, which is dependent on how deep and complex the tree is [2]. We generated a bar chart in Figure 08 which again illustrates that the Fp-growth performed better at the beginning but from 40% Apriori algorithm showed a better result.

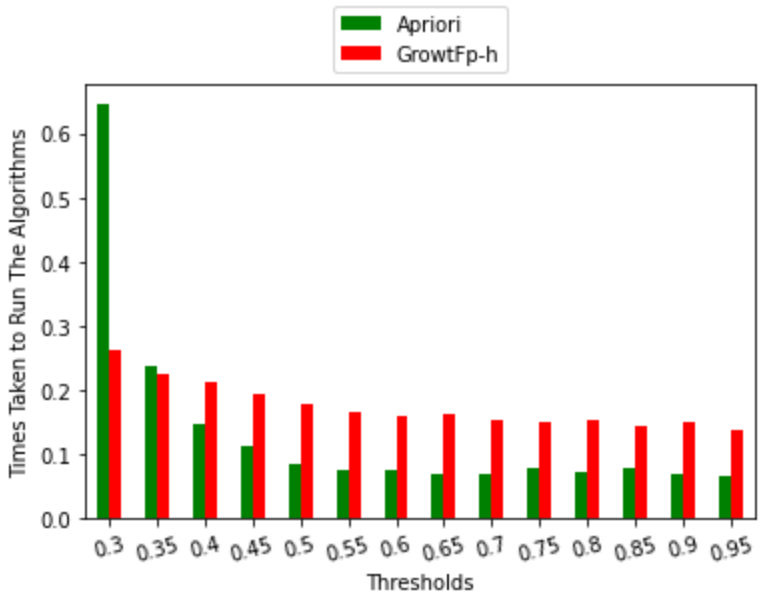

Figure 08: Visualization of comparison using bar chart

Conclusion

We applied both Apriori and Fp-growth algorithm on Chess and Mushroom datasets. Chess dataset is smaller in size while Mushroom's dataset is a bit larger. We saw that Fp-growth algorithm showed better result with chess dataset and Apriori with Mushroom. So, we can conclude by saying that Apriori performs better for larger datasets. Because to generate itemsets, Apriori Algorithm needs to scan repeatedly which makes this algorithm not right for finding a small number of rules. And Fp-growth does better for smaller datasets.

REFERENCES

[1] Kavitha, M., & Selvi, M. S. T. T., Comparative Study on Apriori Algorithm and Fp Growth Algorithm with Pros and Cons, 4(4), 161–164. (2016)

[2] WICAKSONO D, JAMBAK MI, SAPUTRA DM. The Comparison of Apriori Algorithm with
Preprocessing and FP-Growth Algorithm for Finding Frequent Data Pattern in Association Rule. Sriwijaya International Conference on Information Technology and Its Applications (SICONIAN 2019)
2020 May 6 (pp. 315-319). Atlantis Press.