International Journal of Engineering Technology and Management Sciences[IJETMS]

Website: ijetms.in Issue:6, Volume No.4, September-2020 DOI: 10.46647/ijetms.2020.v04i06.001

Evaluation of Frequent Itemset Mining Algorithms -priori and FP Growth

[1]Jismy Joseph,[2]Dr.G. Kesavaraj

[1]PhD Research Scholar, [2]Professor and head
Department of Computer Science, Vivekanandha College of Arts and
Science for Women (Autonomous), Elayampalayam, Thiruchengode, Tamil Nadu, India

[1]jismyjoseph2018@gmail.com
[2]Dr.Kesavaraj@vicas.org

Abstract- **Nowadays the Frequentitemset mining (FIM) is an essential task for retrieving frequently occurring patterns, correlation, events or association in a transactional database. Understanding of such frequent patterns helps to take substantial decisions in decisive situations. Multiple algorithms are proposed for finding such patterns, however the time and space complexity of these algorithms rapidly increases with number of items in a dataset. So it is necessary to analyze the efficiency of these algorithms by using different datasets. The aim of this paper is to evaluate theperformance of frequent itemset mining algorithms, Apriori and Frequent Pattern (FP) growth by comparing their features. This study shows that the FP-growth algorithm is more efficient than the Apriori algorithm for generating rules and frequent pattern mining.**

Keywords- **Frequent itemset mining, Apriorialgorithm, FP-growth algorithm, Weka, Association Rule.**

I. INTRODUCTION

These days the size of databasesincreasesrapidly, this leads to invention of different tools to extract information automatically from the large database. Many researchers used data mining or knowledge discovery in database (KDD) to extract innovative and frequent pattern from large databases or transactional databases.

Frequent itemset mining has been applied in a great number of fields, including intrusion detection, Market basket analysis and credit card fraud prevention to discover unpredicted association among itemset in transactional and relational database.Frequent pattern Mining or association mining shows which items appear together in a transaction or relation.

Frequent patterns are patterns such as subsequences, itemsets or substructures that appears in a data set frequently. Finding such frequent pattern plays an essential role in mining association, correlations, and many other relationships among the data. It also helps in data classification, clustering and other data mining tasks as well. Thus, frequent pattern mining has becomes an important data mining task and a focused theme in data mining research [1].

II. LITERATURE REVIEW

Much research work has been done to compare different algorithms used for frequent itemset. In the paper ‘Survey on Frequent Item set Mining Algorithms’ [2] Pramod S and O.P. Vyas conducted a survey on frquent itemset mining algorithms. They used Adult and Mushroom dataset for evaluating the performance of different algorithm. This study comprises features like different support values, size of transactions and different datasets. They concluded that SaM algorithm performed better in all data set.

In [3] M.S Mythili, A.R Mohamed Shanavas conducted a Performance Evaluation of Apriori and FP-Growth Algorithms by comparing the capabilities of these algorithms. The study show that FP-growth algorithm is more efficient than the Apriori algorithm.

In [4] Mr.Rahul Shukla, Dr. Anil kumar Solanki conducted a performance Evaluation for Frequent Pattern mining Algorithm. In the paper they compared the performance of Apriori and ECLAT Algorithm on medical data and they concluded that the Eclat approach is more efficient for mining frequent patterns in a large database.

In the paper 'Algorithms for frequent itemset mining: a literature review' Chin-Hoong Chee, JafreezalJaafar, Izzatdin Abdul Aziz, MohdHilmi Hasan & William Yeoh reviewedthe strengths and weaknesses of the important and recent algorithms in Frequent Pattern Mining and they identified two major problem with frequent pattern mining. They mentioned that frequent hidden pattern mining needs more time and memory space [5].

In the paper[6]' Frequent Itemsets Mining for Big Data: A Comparative Analysis'DanieleApiletti,ElenaBaralis,TaniaCerquitelli,PaoloGarza,FabioPulvirenti and LucaVenturini conducted a theoretical and experimental comparative analyses of Hadoop- and Spark-based scalable algorithms to find frequent pattern from big data.

In [7] Ravi Ranjan and Aditi Sharmathey compared Hadoop, Spark, Flink by using Apriori and Fp-Growth on different dataset. They suggested that Flink is performing better in the field of big data.

III. FREQUENT PATTERN MINING(FPM) AND ALGORITHMS

In data mining tasks, the frequent item sets plays an important role for finding frequent pattern or association from various kinds of databases like relational database, transactional database etc. FPM plays an essential role for clustering, classifying and identifying outliers a set of data. Apart from this, FPM has many applications like spatiotemporal data analysis, biological data analysis, and software bug detection [8].

FPM is used to predict the occurrence of a specific item based on the occurrence of other items in the transaction. The terminologies used in frequent pattern mining are support and confidence. Support specifies how frequently an itemset appears in the dataset and the confidence describes how often the rule has been found to be true. The support and confidence are defined as:

$$\text{Support } (A \Rightarrow B) = P\ (A \cup B)$$

$$\text{Confidence } (A \Rightarrow B) = \frac{\text{Support } (AUB)}{\text{Support } (A)}$$

The rule is considered as strong if it satisfies a minimum support threshold and a minimum confidence threshold. The methods used in frequent item set mining are

- Finding Frequent itemsets using candidate generation
- Mining Frequent itemsets without candidate generation
- Mining Frequent itemsets using Vertical Data format.

A. *APRIORI Algorithm*

R.Ararwal and R.Srikant proposed this algorithm in 1994 for mining frequent itemsets for Boolean association rule [1]. It uses candidate generation for finding frequent itemsets. Aprioriis an iterative approach, where k-itemsets are used to find (k=1)-itemsets. This algorithm uses Apriori property to reduce the search space. It is a two-step process called join and prune.

Join step - To find L_k, a set of k-itemset is generated by joing L_{k-1} with itself.

Prune step - Scans the count of each item. If it less than minimum support then it does not considered as frequent.

Apriori Algorithm Pseudocode

```
Procedure Apriori(D, minSupport)    // D – database, minsupport- Minimum support
{
L1 = {frequent items};
For (k= 2; Lk-1! =∅; k++)
{
        Ck= candidates generated from Lk-1
        For each transaction t in database do
        {
                Increment all candidates
                Lk = candidates in Ck with
                minSupport
        }
}
ReturnCkLk;
}
```

B. *FP-Growth Algorithm*

The main disadvantages of Apriori algorithm are it generates huge number of candidate sets and this algorithm repeatedly scans the database and checks a large number of candidates by pattern matching. Hence it is very costly. To overcome these disadvantages, the next method for generating frequent itemset without using candidate generation is FP-Growth. FP-Growth uses a divide and conquer approach. It first compresses the database into FP-Tree and then divides these compressed database into a set of conditional database.

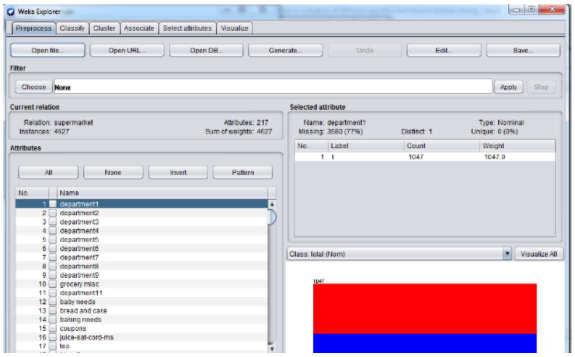
FP-Growth Algorithm

Step1: Scan the database to get set of frequent itemsetand their support count.
Step2: Sort the frequent itemset in descending order using support count.
Step3: Construct FP-tree. Initially it creates the root of the tree and labelled as 'Null'.
Step4: Construct the FP-conditional tree for each item (or itemset)
Step5: Determine the frequent patterns.

IV. DATASETS AND RESULTS

We have used supermarket data set and vote datasets from 'storm.cis.fordham.edu' for comparing two frequent pattern mining algorithms. The first data set is 'Super Market Data Set (SMDS)' [9], which contains 4627 instances and 217attributes. The second one is 'Vote Data Set (VDS)' [9], which consists of 435 instances and 17 attributes. In this comparative study, twofrequent itemset mining algorithms are used. They are Apriori and FP growth.

Initially, the datasets are preprocessed and afterthat the algorithms are applied. The following figures shows the results obtained from weka after preprocessing. Fig.1 shows the result obtained after preprocessing the dataset 'SMDS.



Filter: None

Relation: supermarket
Instances: 4627
Attributes: 217
Sum of weights: 4627

Name: department1
Missing: 3580 (77%)
Distinct: 1
Type: Nominal
Unique: 0 (0%)

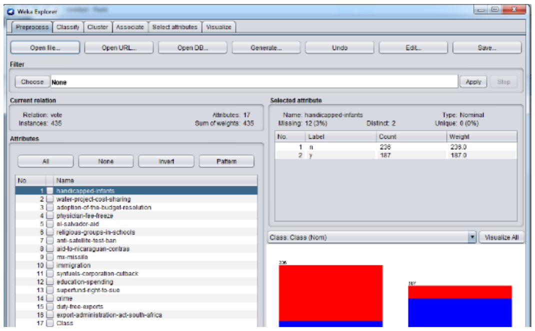
No.	Label	Count	Weight
1	t	1047	1047.0

No.	Name
1	department1
2	department2
3	department3
4	department4
5	department5
6	department6
7	department7
8	department8
9	department9
10	grocery misc
11	department11
12	baby needs
13	bread and cake
14	baking needs
15	coupons
16	juice-sat-cord-ms
17	tea

Class: total (Nom)

FIG.1. PREPROCESSING RESULT – SMDS

Fig.2 shows the result obtained after preprocessing the dataset 'VDS'.



Filter: None

Relation: vote
Instances: 435
Attributes: 17
Sum of weights: 435

Name: handicapped-infants
Missing: 12 (3%)
Distinct: 2
Type: Nominal
Unique: 0 (0%)

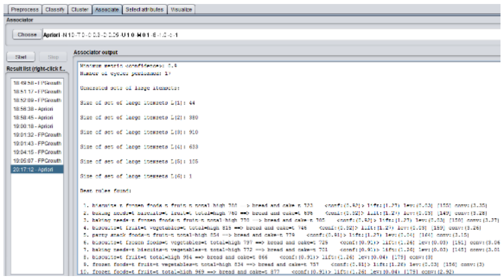
No.	Label	Count	Weight
1	n	236	236.0
2	y	187	187.0

No.	Name
1	handicapped-infants
2	water-project-cost-sharing
3	adoption-of-the-budget-resolution
4	physician-fee-freeze
5	el-salvador-aid
6	religious-groups-in-schools
7	anti-satellite-test-ban
8	aid-to-nicaraguan-contras
9	mx-missile
10	immigration
11	synfuels-corporation-cutback
12	education-spending
13	superfund-right-to-sue
14	crime
15	duty-free-exports
16	export-administration-act-south-africa
17	Class

Class: Class (Nom)

FIG.II. PREPROCESSING RESULT – VDS

The following figures shows the output obtained from Apriori and FP growth algorithms.Fig3 shows the rules generated by Apriori algorithm in SMDC dataset. Ten rules are generated based on support and confidence.



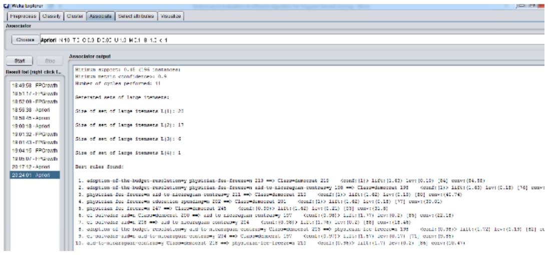
Apriori -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1 -S -1.0 -c -1

18:49:58 - FPGrowth
18:51:17 - FPGrowth
18:52:09 - FPGrowth
18:56:38 - Apriori
18:58:45 - Apriori
19:00:18 - Apriori
19:01:32 - FPGrowth
19:01:43 - FPGrowth
19:04:15 - FPGrowth
19:05:07 - FPGrowth
20:17:12 - Apriori

```
Minimum metric <confidence>: 0.9
Number of cycles performed: 17

Generated sets of large itemsets:

Size of set of large itemsets L(1): 44

Size of set of large itemsets L(2): 380

Size of set of large itemsets L(3): 910

Size of set of large itemsets L(4): 633

Size of set of large itemsets L(5): 105

Size of set of large itemsets L(6): 1

Best rules found:

 1. biscuits=t frozen foods=t fruit=t total=high 788 ==> bread and cake=t 723    <conf:(0.92)> lift:(1.27) lev:(0.03) [155] conv:(3.35)
 2. baking needs=t biscuits=t fruit=t total=high 760 ==> bread and cake=t 696    <conf:(0.92)> lift:(1.27) lev:(0.03) [149] conv:(3.28)
 3. baking needs=t frozen foods=t fruit=t total=high 770 ==> bread and cake=t 705    <conf:(0.92)> lift:(1.27) lev:(0.03) [150] conv:(3.27)
 4. biscuits=t fruit=t vegetables=t total=high 815 ==> bread and cake=t 746    <conf:(0.92)> lift:(1.27) lev:(0.03) [159] conv:(3.26)
 5. party snack foods=t fruit=t total=high 854 ==> bread and cake=t 779    <conf:(0.91)> lift:(1.27) lev:(0.04) [164] conv:(3.15)
 6. biscuits=t frozen foods=t vegetables=t total=high 797 ==> bread and cake=t 725    <conf:(0.91)> lift:(1.26) lev:(0.03) [151] conv:(3.06
 7. baking needs=t biscuits=t vegetables=t total=high 772 ==> bread and cake=t 701    <conf:(0.91)> lift:(1.26) lev:(0.03) [145] conv:(3.01
 8. biscuits=t fruit=t total=high 954 ==> bread and cake=t 866    <conf:(0.91)> lift:(1.26) lev:(0.04) [179] conv:(3)
 9. frozen foods=t fruit=t vegetables=t total=high 834 ==> bread and cake=t 757    <conf:(0.91)> lift:(1.26) lev:(0.03) [156] conv:(3)
10. frozen foods=t fruit=t total=high 969 ==> bread and cake=t 877    <conf:(0.91)> lift:(1.26) lev:(0.04) [179] conv:(2.92)
```

FIG.III. THE RULES GENERATED BY APRIORI ALGORITHM IN SMDC DATASET

Fig.4 shows the rules generated by Apriori algorithm in VDS. Ten rules are generated based on support and confidence.



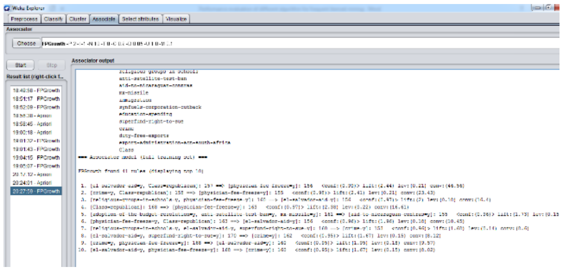
Apriori -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1 -S -1.0 -c -1

18:49:58 - FPGrowth
18:51:17 - FPGrowth
18:52:09 - FPGrowth
18:56:38 - Apriori
18:58:45 - Apriori
19:00:18 - Apriori
19:01:32 - FPGrowth
19:01:43 - FPGrowth
19:04:15 - FPGrowth
19:05:07 - FPGrowth
20:17:12 - Apriori
20:24:01 - Apriori

```
Minimum support: 0.45 (196 instances)
Minimum metric <confidence>: 0.9
Number of cycles performed: 11

Generated sets of large itemsets:

Size of set of large itemsets L(1): 20

Size of set of large itemsets L(2): 17

Size of set of large itemsets L(3): 6

Size of set of large itemsets L(4): 1

Best rules found:

 1. adoption-of-the-budget-resolution=y physician-fee-freeze=n 219 ==> Class=democrat 219    <conf:(1)> lift:(1.63) lev:(0.19) [84] conv:(84.58)
 2. adoption-of-the-budget-resolution=y physician-fee-freeze=n aid-to-nicaraguan-contras=y 198 ==> Class=democrat 198    <conf:(1)> lift:(1.63) lev:(0.18) [76] conv:
 3. physician-fee-freeze=n aid-to-nicaraguan-contras=y 211 ==> Class=democrat 210    <conf:(1)> lift:(1.62) lev:(0.18) [80] conv:(40.74)
 4. physician-fee-freeze=n education-spending=n 202 ==> Class=democrat 201    <conf:(1)> lift:(1.62) lev:(0.18) [77] conv:(39.01)
 5. physician-fee-freeze=n 247 ==> Class=democrat 245    <conf:(0.99)> lift:(1.62) lev:(0.21) [93] conv:(31.8)
 6. el-salvador-aid=n Class=democrat 200 ==> aid-to-nicaraguan-contras=y 197    <conf:(0.98)> lift:(1.77) lev:(0.2) [85] conv:(22.18)
 7. el-salvador-aid=n 208 ==> aid-to-nicaraguan-contras=y 204    <conf:(0.98)> lift:(1.76) lev:(0.2) [88] conv:(18.46)
 8. adoption-of-the-budget-resolution=y aid-to-nicaraguan-contras=y Class=democrat 203 ==> physician-fee-freeze=n 198    <conf:(0.98)> lift:(1.72) lev:(0.19) [82] cc
 9. el-salvador-aid=n aid-to-nicaraguan-contras=y 204 ==> Class=democrat 197    <conf:(0.97)> lift:(1.57) lev:(0.17) [71] conv:(9.85)
10. aid-to-nicaraguan-contras=y Class=democrat 218 ==> physician-fee-freeze=n 210    <conf:(0.96)> lift:(1.7) lev:(0.2) [86] conv:(10.47)
```

FIG.IV. THE RULES GENERATED BY APRIORI ALGORITHM IN VDS

The rule generated by FP-Growth using vote data set is shown in Fig 5. FP Growth found 41 rules in vote dataset and only the top 10 rules are displayed in the following figure.

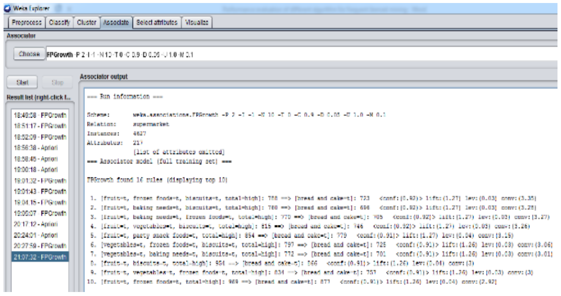


FPGrowth -P 2 -I -1 -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1

```
              religious-groups-in-schools
              anti-satellite-test-ban
              aid-to-nicaraguan-contras
              mx-missile
              immigration
              synfuels-corporation-cutback
              education-spending
              superfund-right-to-sue
              crime
              duty-free-exports
              export-administration-act-south-africa
              Class
=== Associator model (full training set) ===

FPGrowth found 41 rules (displaying top 10)

 1. [el-salvador-aid=y, Class=republican]: 157 ==> [physician-fee-freeze=y]: 156   <conf:(0.99)> lift:(2.44) lev:(0.21) conv:(46.56)
 2. [crime=y, Class=republican]: 158 ==> [physician-fee-freeze=y]: 155   <conf:(0.98)> lift:(2.41) lev:(0.21) conv:(23.43)
 3. [religious-groups-in-schools=y, physician-fee-freeze=y]: 160 ==> [el-salvador-aid=y]: 156   <conf:(0.97)> lift:(2) lev:(0.18) conv:(16.4)
 4. [Class=republican]: 168 ==> [physician-fee-freeze=y]: 163   <conf:(0.97)> lift:(2.38) lev:(0.22) conv:(16.61)
 5. [adoption-of-the-budget-resolution=y, anti-satellite-test-ban=y, mx-missile=y]: 161 ==> [aid-to-nicaraguan-contras=y]: 155   <conf:(0.96)> lift:(1.73) lev:(0.15
 6. [physician-fee-freeze=y, Class=republican]: 163 ==> [el-salvador-aid=y]: 156   <conf:(0.96)> lift:(1.96) lev:(0.18) conv:(10.45)
 7. [religious-groups-in-schools=y, el-salvador-aid=y, superfund-right-to-sue=y]: 160 ==> [crime=y]: 153   <conf:(0.96)> lift:(1.68) lev:(0.14) conv:(8.6)
 8. [el-salvador-aid=y, superfund-right-to-sue=y]: 170 ==> [crime=y]: 162   <conf:(0.95)> lift:(1.67) lev:(0.15) conv:(8.12)
 9. [crime=y, physician-fee-freeze=y]: 168 ==> [el-salvador-aid=y]: 160   <conf:(0.95)> lift:(1.95) lev:(0.18) conv:(9.57)
10. [el-salvador-aid=y, physician-fee-freeze=y]: 168 ==> [crime=y]: 160   <conf:(0.95)> lift:(1.67) lev:(0.15) conv:(8.02)
```

FIG.V. THE RULE GENERATED BY FP-GROWTH USING VOTE DATA SET

Fig 6 shows the rules generated by FP Growth in SMDC data set. FP Growth found 16 rules in vote data set and only the top 10 rules are displayed in the following figure.



FPGrowth -P 2 -I -1 -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1

```
=== Run information ===

Scheme:       weka.associations.FPGrowth -P 2 -I -1 -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1
Relation:     supermarket
Instances:    4627
Attributes:   217
              [list of attributes omitted]
=== Associator model (full training set) ===

FPGrowth found 16 rules (displaying top 10)

 1. [fruit=t, frozen foods=t, biscuits=t, total=high]: 788 ==> [bread and cake=t]: 723   <conf:(0.92)> lift:(1.27) lev:(0.03) conv:(3.35)
 2. [fruit=t, baking needs=t, biscuits=t, total=high]: 760 ==> [bread and cake=t]: 696   <conf:(0.92)> lift:(1.27) lev:(0.03) conv:(3.28)
 3. [fruit=t, baking needs=t, frozen foods=t, total=high]: 770 ==> [bread and cake=t]: 705   <conf:(0.92)> lift:(1.27) lev:(0.03) conv:(3.27)
 4. [fruit=t, vegetables=t, biscuits=t, total=high]: 815 ==> [bread and cake=t]: 746   <conf:(0.92)> lift:(1.27) lev:(0.03) conv:(3.26)
 5. [fruit=t, party snack foods=t, total=high]: 854 ==> [bread and cake=t]: 779   <conf:(0.91)> lift:(1.27) lev:(0.04) conv:(3.15)
 6. [vegetables=t, frozen foods=t, biscuits=t, total=high]: 797 ==> [bread and cake=t]: 725   <conf:(0.91)> lift:(1.26) lev:(0.03) conv:(3.06)
 7. [vegetables=t, baking needs=t, biscuits=t, total=high]: 772 ==> [bread and cake=t]: 701   <conf:(0.91)> lift:(1.26) lev:(0.03) conv:(3.01)
 8. [fruit=t, biscuits=t, total=high]: 954 ==> [bread and cake=t]: 866   <conf:(0.91)> lift:(1.26) lev:(0.04) conv:(3)
 9. [fruit=t, vegetables=t, frozen foods=t, total=high]: 834 ==> [bread and cake=t]: 757   <conf:(0.91)> lift:(1.26) lev:(0.03) conv:(3)
10. [fruit=t, frozen foods=t, total=high]: 969 ==> [bread and cake=t]: 877   <conf:(0.91)> lift:(1.26) lev:(0.04) conv:(2.92)
```

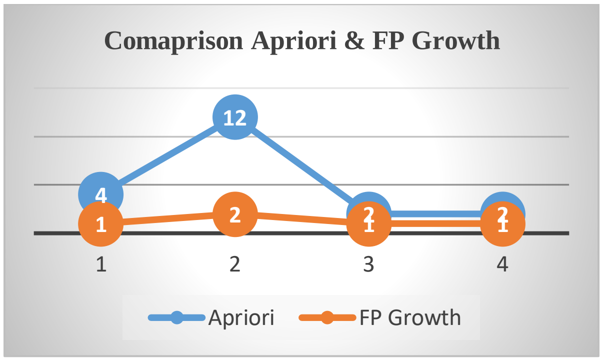
FIG. VI. THE RULES GENERATED BY FP GROWTH IN SMDC DATA SET

V. COMPARISON OF RESULTS

This section deals the comparison part of Apriori and FP Growth algorithms. The table1 shows the time needed to generates rules by using Apriori and FP Growth.Compared to Apriori the FP Growth algorithm takes less time to generate the rules in both datasets. In the supermarket dataset,Apriori algorithm took 4 seconds to produce the rules when minimum support is 0.15. But FP growth algorithm generated the rules within in a second. In the case of vote dataset FP Growth performed better than the Apriori algorithm.

Instances	**Attribute**	**Minimum support**	**Time Taken (in Sec)**	
			Apriori	**FP Growth**
SupperMarket (4627)	217	0.15	4	1
		0.2	12	2
Vote(435)	17	0.15	2	1
		0.2	2	1

TABLE 1- COMPARISON RESULTS



VI. CONCLUSION

In many data mining applications, association rule plays an important role for finding frequent pattern. In this study we observed that FP Growth algorithm is better than the Apriori algorithm. In both datasets the FP growth taken less time to generate the rule. FP-growth is more acceptable for larger databases.

References

[1] Jiawei Han and Micheline Kamber. Data Mining: Concepts and Techniques, Morgan Kaufmann Publishers, Second Edition, 2003.

[2] Pramod, S., & Vyas, O. P. (2010). Survey on frequent item set mining algorithms. *International journal of computer applications*, *1*(15), 86-91.

[3] Mythili, M. S., &Shanavas, A. M. (2013). Performance evaluation of apriori and fp-growth algorithms. *International Journal of Computer Applications*, *79*(10).

[4] Mr.Rahul Shukla, &Dr.AnilkumarSolanki(2015). Performance Evaluation for Frequent Pattern mining Algorithm. International Journal of Engineering Research and General Science Volume 3, Issue 5.

[5] Chee, C. H., Jaafar, J., Aziz, I. A., Hasan, M. H., &Yeoh, W. (2019). Algorithms for frequent itemset mining: a literature review. *Artificial Intelligence Review*, *52*(4), 2603-2621.

[6] Apiletti, D., Baralis, E., Cerquitelli, T., Garza, P., Pulvirenti, F., &Venturini, L. (2017). Frequent itemsets mining for big data: a comparative analysis. *Big Data Research*, *9*, 67-83.

[7] Ranjan, R., & Sharma, A. (2019). Evaluation of frequent itemset mining platforms using apriori and fp-growth algorithm. *International Journal of Information Systems & Management Science*, *2*(2).

[8] Aggarwal CC (2014) An introduction to Frequent Pattern Mining. In: Aggarwal CC, Han J (eds) Frequent Pattern Mining. Springer, Basel, pp 1–14.

[9] Weiss, G. M. (2020, September). *data-mining/datasets.html*. Retrieved from storm.cis.fordham.edu: https://storm.cis.fordham.edu/~gweiss/data-mining/datasets.html