Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X
IMPACT FACTOR: 7.056

IJCSMC, Vol. 10, Issue. 7, July 2021, pg.20 – 27

An Enhanced Apriori with Interestingness of Patterns using cSupport and rSupport

Sudhir Tirumalasetty[1]; A. Aruna[2]; A. Padmini[3]; D. Vijaya Sagaru[4]; A. Tejeswini[5]
[1]Department of Computer Science & Engineering, Vasireddy Venkatadri Institute of Technology, Guntur, India
[2]Department of Computer Science & Engineering, Vasireddy Venkatadri Institute of Technology, Guntur, India
[3]Department of Computer Science & Engineering, Vasireddy Venkatadri Institute of Technology, Guntur, India
[4]Department of Computer Science & Engineering, Vasireddy Venkatadri Institute of Technology, Guntur, India
[5]Department of Computer Science & Engineering, Vasireddy Venkatadri Institute of Technology, Guntur, India
[1] sudhir.t@vvit.net; [2] arunaardala@gmail.com; [3] ammireddypaddu27@gmail.com;
[4] vijaysagardakuri123@gmail.com; [5] ambatiteja062@gmail.com
DOI: 10.47760/ijcsmc.2021.v10i07.003

Abstract— Data mining is wide spreading its applications in several areas. There are different tasks in mining which provides solutions for wide variety of problems in order to discover knowledge. Among those tasks association mining plays a pivotal role for identifying frequent patterns. Among the available association mining algorithms Apriori algorithm is one of the most prevalent and dominant algorithm which is used to discover frequent patterns. An enhancement to Apriori algorithm is done i.e. Apriori$_2$ which minimized the number of scans. In this research Apriori$_2$ is modified by including rSupport or cSupport. Also includes the comparison of these variants of APRIORI along with the proposed.

Keywords— Apriori, Apriori$_2$, cSupport, rSupport

I. INTRODUCTION

Data set has cutting-edge distinctive expansion in its volume with time. This remarkable expansion in information came about with a point of finding information which is utilized to help dynamic framework. Information mining is the critical advance in the information disclosure measure. The undertakings of information mining are by and large partitioned in two classifications: Predictive and Descriptive. The objective of the prescient errands is to anticipate the worth of a specific characteristic dependent on the upsides of different traits and the objective of unmistakable undertakings, is to mine beforehand obscure and valuable data from enormous information bases. The objectives of these assignments in information mining are acquired by certain procedures. They are: bunching, grouping, affiliation rule mining, successive example disclosure and investigation. The advances of information mining frameworks have wide spread its extent lately for some, dynamic frameworks like deals examination, medical services, online business, producing, and so on

Among the different procedures utilized in finding information, affiliation mining is perhaps the most focal information mining's usefulness. This principally includes in separating affiliation rules [16]. These principles

are utilized in distinguishing incessant examples [17]. The upsides of these standards are finding obscure connections and creating results which gives premise to dynamic and forecast in regions like medical care, banking, fabricating, media communications and so on [16].

Existing affiliation mining calculations has not many blemishes. They are: (I) The entire information base should be filtered for more number of times despite the fact that couple of examples are fascinating. This outcomes in wastage of time. (ii) The guidelines produced by these affiliation mining methods are huge and are hard to comprehend. (iii) Defining backing and certainty esteems isn't clear. These qualities are characterized tentatively. In general fostering an ideal affiliation mining calculation is a provocative undertaking.

In this paper the overall Apriori calculation and a further developed rendition of Apriori calculation [18] are analyzed which brought about advancement of another new further developed variant of Apriori calculation. This new further developed adaptation limits the quantity of data set outputs.

The remainder of the paper is coordinated as follows. Segment 2 explains about affiliation mining. Segment 3 thinks about the overall Apriori calculation and existing further developed variant of Apriori calculation [18]. Area 4 presents another upgraded adaptation of Apriori calculation. Segment 5 thinks about the aftereffects of existing general Apriori calculation, existing further developed variants of Apriori calculation [18] and new further developed adaptation of Apirori calculation. At long last, end and the future extent of this new further developed rendition of Apriori calculation.

Among the current issues in information mining affiliation mining is prevalent. Finding successive examples (rules) is common in affiliation mining. These guidelines assume a vital part for dynamic frameworks and are an arising region in research [1]. These standards gives answer for issues in regions like banking, advertising, medical care, telecom, text information bases [2], web [3] and data sets containing satisfactory pictures [4].

Till dated wide number of affiliation mining calculations were presented [5, 6, 7, 8, 9, 10]. These calculations are gathered into two gatherings dependent on their methodology. They are:

a. Candidate age approach

   Ex: Apriori [6]

b. Pattern development approach

   Ex: FP Growth [9, 10]

Between these two gatherings, the primary gathering created numerous affiliation mining calculations. Among those, Apriori calculation is the main calculation. This Apriori calculation is upgraded by numerous researchers brought about evolution of streamlined Apriori like calculations [11, 12, 13, 14, 15]. To find successive examples these Apriori like calculations follow iterative methodology.

II. EXISTING ALGORITHMS

A. The General Apriori Algorithm

The general Apriori algorithm is:

T: Transactional data base
Ck: Candidate item set of size k
Lk: Frequent item set of size k
s: Support

```
Apriori(T, s)
    L1 ← { large 1-item set that appear in more than or equal to s transactions }
    k ← 2
    While Lk-1 ≠ ϕ
        Ck ← Join(Lk-1)
        For each transaction t in T
            For each candidate c in Ck
                If(c c t) then
                 count[c]←count[c]+1
                End If
            End For
        End For
        Lk = ϕ
```



```
            For each candidate c in Ck //Prune
                    If (count[c] >= s) then
                       Lk ← Lk U {c}
                    End If
            End For
            k ← k + 1
        End While
        Return Lk
    End Apriori
```

In everyday Apriori calculation, for every up-and-comer in Ck, recurrence is determined by checking conditional information base. In the wake of computing frequencies for all applicants in a Ck these frequencies are contrasted and backing, s and avoid up-and-comers with frequencies not as much as s. This outcomes in age of Lk.

The overall Apriori calculation has a few defects:

- The conditional information base is filtered more than once. This is on the grounds that each up-and-comer of applicant set (Ck) produced after Join activity should be checked in all exchanges of conditional information base for the presence of up-and-comer.
- If there are sufficient exchanges then the genera Apriori calculation isn't able.

B. Enhanced Apriori Algorithm – Apriori$_1$

Mohammed Al-Maolegi et. al. fostered a further developed variant of Apriori calculation, Apriori$_1$ [18] pointed in decreasing the rehashed sweeps of conditional information base. This further developed calculation Apriori$_1$ is:

```
T: Transactional data base
Ck: Candidate item set of size k
Lk: Frequent item set of size k
s: Support

Apriori1(T, s)
    L1 ← { large 1-item set that appear in more than or equal to s transactions }
    k ← 2
    While Lk-1 ≠ ϕ
            Ck ← Join(Lk-1)
            For each candidate c in Ck
               Ix = Get_Item_Min_Support(c, L1)
               Tid = Get_Transaction_Ids(Ix)
                    For each transaction t in Tid
                            If(c c t) then
                              count[c]←count[c]+1
                            End If
                    End For
            End For
            Lk = ϕ
            For each candidate c in Ck //Prune
                    If (count[c] >= s) then
                       Lk ← Lk U {c}
                    End If
            End For
            k ← k + 1
    End While
    Return Lk
End Apriori1
```

In this further developed form of Apriori, for every applicant c in Ck, thing (Ix) with least help among the things in c is gotten and the exchanges that contain that thing (Ix) are gathered (Tid). Then, in every exchange t in Tid the presence of c is checked and the recurrence of c is determined as opposed to filtering the whole

conditional data set. Later the frequencies of c in Ck are contrasted and backing, s and prohibit the competitors with frequencies not as much as s. This outcomes in age of Lk.

The benefit of this further developed Apriori calculation is:

- The whole value-based data set isn't filtered for ascertaining the recurrence of c in Ck.

This further developed form of Apriori calculation has an imperfection:

- All exchanges with exchange ids in Tid are checked for presence of c despite the fact that couple of exchanges contain c.

C. Apriori$_2$ [19]

The Apriori$_2$ algorithm is:

```
T: Transactional data base
Ck: Candidate item set of size k
Lk: Frequent item set of size k
s: Support
Apriori2(T, s)
        L1 ← { large 1-item set that appear in more than or equal to s transactions }
        k ← 2
        While Lk-1 ≠ ϕ
            Ck ← Join(Lk-1)
            For each candidate c in Ck
                Lk = ϕ
                Tid = Get_Common_Transaction_Ids(c, L1)
                If ( |Tid| >= s) then
                        Lk ← Lk U {c}
                End If
             End For
             k ← k + 1
        End While
        Return Lk
End Apriori2
```

Apriori$_2$ combines both join and prune operations of Apriori and Apriori1. The proposed algorithm obtains common transaction ids for all items in c of Ck as a group (Tid). The count of group of transaction ids (|Tid|) defines the frequency of c. If this frequency is greater than and equal to support, s then c of Ck is included in Lk. The number of statements in proposed algorithm Apriori$_2$ is less when compared with Apriori and Apriori$_1$.

None of the above algorithms specify directly about the interestingness of pattern being involved in the dataset. If this interestingness can be determined then decision making for better profitability of an organization can be improved i.e. considering patterns of highest weight. This is achieved by including rSupport [20] and cSupport [20] in Apriori$_2$ which is portrayed in proposed algorithm, Apriori$_3$

III. PROPOSED ALGORITHM

A. The Apriori$_3$ algorithm is:

```
T: Transactional data base
Ck: Candidate item set of size k
Lk: Frequent item set of size k
s: Support
Apriori3(T, s)
        L1 ← { large 1-item set that appear in more than or equal to s transactions }
        k ← 2
        While Lk-1 ≠ ϕ
```



```
            Ck ← Join(Lk-1)
            For each candidate c in Ck
                Lk = ϕ
                Tid = Get_Common_Transaction_Ids(c, L1)
                If ( |Tid| >= s) then
                        Lk ← Lk U {c}
                        Calculate_rSupport(c)
                        Calculate_cSupport(c)
                End If
            End For
            k ← k + 1
        End While
        Return Lk
End Apriori3

calculate_rSupport(trans,pattern):
    rSupport = 0
    n = len(pattern)
    originalPattern = n*(n+1)//2
    for each_trans in trans:
        lis = Transaction_Data[each_trans]
        lis.sort(reverse = True)
        patternPosition = 0
        for item in pattern:
            patternPosition += lis.index(item)+1
        rSupport += (originalPattern/patternPosition)*(originalPattern/patternPosition)
    return rSupport

calculate_cSupport(trans,pattern):
    pattern = sorted(list(pattern))
    cSupport = 0
    originalPattern = len(pattern)
    for each_trans in trans:
        lis = Transaction_Data[each_trans]
        lis.sort()
        if(pattern[0] in lis and pattern[-1] in lis):
            patternLength = lis.index(pattern[-1])-lis.index(pattern[0])+1
        cSupport += originalPattern/patternLength
    return cSupport
```

B. cSupport and rSupport

Definition:

$$cSup(\mathcal{P}) = \sum_{\mathcal{X} \in \mathbb{X}_{\mathcal{P}}} W_c(\mathcal{X}, \mathcal{P})$$

$$W_c(\mathcal{X}, \mathcal{P}) = \text{avg}_{I \in \mathbb{I}_{\mathcal{X}}(\mathcal{P})} \frac{\ell(\mathcal{P})}{\ell(I)}$$

$$rSup(\mathcal{P}) = \sum_{\mathcal{X} \in \mathbb{X}_{\mathcal{P}}} W_r(\mathcal{X}, \mathcal{P})$$

Where

$\mathbb{X}$	A database of event sequences
$\mathbb{E}$	The set of all possible events
$\mathcal{X} \in \mathbb{X}$	Event sequences
$a, b, \ldots, x \in \mathbb{E}$	Individual events
$\mathcal{P}$	A pattern (also a sequence of events)
$\mathbb{I}(\mathcal{P})$	The set of all instances of $\mathcal{P}$
$\mathbb{I}_{\mathcal{X}}(\mathcal{P})$	The set of all instances of $\mathcal{P}$ lying in $\mathcal{X}$
$I \in \mathbb{I}(\mathcal{P})$	A pattern instance of $\mathcal{P}$
$\ell(\mathcal{X})$, $\ell(\mathcal{P})$, or $\ell(I)$	The length of $\mathcal{X}$, $\mathcal{P}$, or I
$L(\mathcal{P})$ or $L(I)$	The extendable length of $\mathcal{P}$ or I
$i, j \in [1, \ell(\mathcal{X})]$	Index numbers for events in $\mathcal{X}$
$\delta(i, j)$	The distance from x_i to x_j in a sequence
$W(\mathcal{X}, \mathcal{P})$	The weight of $\mathcal{X}$ w.r.t. $\mathcal{P}$
$sup(\mathcal{P}), cSup(\mathcal{P}), rSup(\mathcal{P})$	Interestingness measures of $\mathcal{P}$

IV. EXAMPLE

To follow out the proposed calculation consider the value-based data set with Transactions (Ti) and Items (Ii) displayed in Table I.

TABLE I
TRANSACTIONAL DATASET

Tid	Items
T1	I1, I2, I5
T2	I2, I4
T3	I2, I4
T4	I1, I2, I4
T5	I1, I3
T6	I2, I3
T7	I1, I3
T8	I1, I2, I3, I5
T9	I1, I2, I3

Result obtained for Apriori$_3$ over the dataset shown in Table I is shown in Fig. 1., which shows the interestingness of patterns of varying sizes with two measures cSupport and rSupport.

```
Pattern of length  1 :
Pattern                         cSupport            rSupport
('I1',)                         6.0                 0.8958333333333333
('I2',)                         7.0                 1.6111111111111112
('I3',)                         5.0                 4.25
('I4',)                         3.0                 3.0
('I5',)                         2.0                 2.0
Pattern of length  2 :
 Pattern                         cSupport             rSupport
('I5', 'I1')                    1.1666666666666665  0.9225
('I5', 'I2')                    1.6666666666666665  1.5625
('I3', 'I1')                    3.333333333333333   2.8125
('I3', 'I2')                    3.0                 2.36
('I4', 'I2')                    3.0                 3.0
('I1', 'I2')                    4.0                 1.2636734693877552
Pattern of length  3 :
 Pattern                         cSupport             rSupport
('I1', 'I2', 'I3')              2.0                 1.4444444444444444
('I1', 'I2', 'I5')              1.75                1.5625
Pattern of length  4 :
 Pattern                         cSupport             rSupport
Frequency Pattern
{('I1', 'I2', 'I5'): (2, ['T8', 'T1']), ('I1', 'I2', 'I3'): (2, ['T8', 'T9'])}
```

Fig. 1 Interestingness of patterns of varying sizes using Apriori$_3$

V. COMPARISONS OF APRIORI, APRORI$_1$, APRIORI$_2$ AND APRIORI$_3$

As mentioned interestingness of pattern being involved in dataset is not presented directly in Apriori, Apriori$_1$ and Apriori$_2$. This is achieved in Apriori$_3$. Apriori$_3$ portrays the interestingness of pattern in percentage using cSupport and rSupport. Also this algorithm involves with minimum number of disk access as in Apriori$_2$. This comparison is shown in Table II.

TABLE III
COMPARISONS OF APRIORI, APRORI$_1$, APRIORI$_2$ AND APRIORI$_3$

Pattern Size	**Apriori**		**Apriori$_1$**		**Apriori$_2$**		**Apriori$_3$**	
	No. of Scans	**Portraying Interestingness of Patterns**	**No. of Scans**	**Portraying Interestingness of Patterns**	**No. of Scans**	**Portraying Interestingness of Patterns**	**No. of Scans**	**Portraying Interestingness of Patterns**
Frequent 1-item set	45	No	45	No	45	No	45	Yes
Frequent 2-item set	54	No	25	No	0	No	0	Yes
Frequent 3-item set	36	No	14	No	0	No	0	Yes
Total no. of scans	135		84		45		45	

VI. CONCLUSIONS

The algorithm proposed in this paper Apriori$_3$ is evident that it portrays the interestingness of patterns with two different measures. This proposed algorithm is best suitable for decision making systems for apt decision while making decisions with an aim of increasing profitability.

REFERENCES

[1] J. Han and M. Kamber, "Data Mining: Concepts and Techniques", Morgan Kaufmann Publishers, 2000.

[2] J. D. Holt and S. M. Chung, "Efficient Mining of Association Rules in Text Databases" CIKM'99, Kansas City, USA, pp. 234242,Nov. 1999.

[3]] B. Mobasher, N. Jain, E.H. Han, and J. Srivastava, "Web Mining: Pattern Discovery from World Wide Web Transactions" Department of Computer Science, University of Minnesota, Technical Report TR96-050, (March, 1996).

[4] C. Ordonez, and E. Omiecinski, "Discovering Association Rules Based on Image Content" IEEE Advances in Digital Libraries (ADL'99), 1999.

[5] R. Agrawal, T. Imielinski, and A. Swami. "Mining association rules between sets of items in large databases". In Proceedings of the ACM SIGMOD International Conference on Management of Data (ACM SIGMOD '93), pages 207216, Washington, USA, May 1993.

[6] R. Agrawal, H. Mannila, R. Srikant, H. Toivonen, and A. I. Verkamo. "Fast discovery of association rules. In Advances in Knowledge Discovery and Data Mining", pages 307328. AAAI Press, 1996.

[7] R. Bayardo and R. Agrawal. "Mining the most interesting rules". In Proceedings of the 5th International Conference on Knowledge Discovery and Data Mining (KDD '99), pages 145154, San Diego, California, USA, August 1999.

[8] J. Hipp, U. Güntzer, and U. Grimmer. "Integrating association rule mining algorithms with relational database systems". In Proceedings of the 3rd International Conference on Enterprise Information Systems (ICEIS 2001), pages 130137, Setúbal, Portugal, July 710 2001.

[9] R. Ng, L. S. Lakshmanan, J. Han, and T. Mah. "Exploratory mining via constrained frequent set queries". In Proceedings of the 1999 ACM-SIGMOD International Conference on Management of Data (SIGMOD '99), pages 556558, Philadelphia, PA, USA, June 1999.

[10] Y. Guizhen: "The complexity of mining maximal frequent itemsets and maximal frequent patterns", Proceedings of the 2004 ACM SIGKDD international conference on Knowledge discovery and data mining , pages:343353,August 2004, Seattle, WA, USA.

[11] L. Klemetinen, H. Mannila, P. Ronkainen, et al. (1994) "Finding interesting rules from large sets of discovered association rules". Third International Conference on Information and Knowledge Management pp. 401407.Gaithersburg, USA.

[12] J. S. Park, M.S. Chen, and P.S. Yu. "An Effective HashBased Algorithm for Mining Association Rules". Proceedings of the 1995 ACM SIGMOD International Conference on Management of Data, San Jose, CA, USA, 1995, 175186.
[13] H. Toivonen, "Sampling large databases for association rules". 22nd International Conference on Very Large Data Bases pp. 134–145. 1996.
[14] P. Kotásek and J. Zendulka, "Comparison of Three Mining Algorithms for Association Rules". Proc. of 34th Spring Int. Conf. on Modelling and Simulation of Systems (MOSIS'2000), Workshop Proceedings Information Systems Modelling (ISM'2000), pp. 8590. Rožnov pod Radhoštěm, CZ, MARQ. 2000.
[15] J. Han, J. Pei, and Y. Yin, "Mining frequent patterns Candidate generation". In Proc. 2000 ACMSIGMOD Int. Management of Data (SIGMOD'00), Dallas, TX. 2000.
[16] F. H. AL-Zawaidah, Y. H. Jbara, and A. L. Marwan, "An Improved Algorithm for Mining Association Rules in Large Databases," Vol. 1, No. 7, 311-316, 2011.
[17] J. Han, M. Kamber,"Data Mining: Concepts and Techniques", Morgan Kaufmann Publishers, Book, 2000.
[18] Mohammed Al-Maolegi1, Bassam Arkok2, International Journal on Natural Language Computing (IJNLC) Vol. 3, No.1, February 2014.
[19] Sudhir Tirumalasetty, Aruna Jadda and Sreenivasa Reddy Edara, "An Enhanced Apriori Algorithm for Discovering Frequent Patterns with Optimal Number of Scans", International Journal of Computer Science Issues, Volume 12, Issue 3, May 2015.
[20] Hayyku Kim and Dong Wan Choi, "Recency-based sequential pattern mining in multiple event sequences", Data Mining and Knowledge Discovery, Springer, September, 2020, https://doi.org/10.1007/s10618-020-00715-7.

