ISSN 2347 - 3983
Volume 7, No.8 August 2019
International Journal of Emerging Trends in Engineering Research
Available Online at http://www.warse.org/IJETER/static/pdf/file/ijeter02782019.pdf
https://doi.org/10.30534/ijeter/2019/02782019


# Mining Negative Frequent regular Itemsets from Data Streams

**NVS Pavan Kumar[1], Dr.JKR Sastry[2], Dr. K Raja Sekhara Rao[3]**
[1]Koneru Lakshmaiah Education Foundation, Vaddeswaram, Andhra Pradesh, India, nvspavankumar@kluniversity,in
[2]Koneru Lakshmaiah Education Foundation, Vaddeswaram, Andhra Pradesh, India, drsastry@kluniversity,in
3Usharama institute of Engineering and Technology, Andhra Pradesh, India, krr_it@yahoo.co.in

## ABSTRACT

Many Application in modern days requires capturing continuous data generated from remote sensors. Devices etc. and the data flows in streams, and the data is processed, and the processed results are temporarily stored. The original data is seldom stored. A lot amount of s Knowledge hidden in the data flowing in streams. Data patterns hidden in the data when mined will discover the hidden knowledge. Most of the research focussed on ming frequent items and positive associations. The regularity of occurrence of items is also important in addition to the frequency of occurrence of the itemsets. While positive associations are good, negative associations also revel very interesting findings which will help in taking important decisions.

In this approach, novel algorithms, along with its implementation presented that will help ming regular, frequent, and negative itemsets from the data streams.

**Key words:** Frequent Itemsets, regular Itemsets, Negative Associations, datastreams

## 1 INTRODUCTION

Data is stored either in standalone, distributed, incremental databases, or in data streams. The data collected from sensors consciously and generally sent as Streams, which needs to be collected and processed in line with the flow of the data. The size of the instream data is quite large as the data is sensed and moved with rapid speed. The instream data can be scanned only once compared to the databases as the instream data never stored in its raw form.

Static database, incremental databases, and distributed databases mined for identifying the interesting patterns from which either the positive or negative associations mined. Data can be stored into a single database or stored at different databases in a distributed manner. Data can be captured and stored and maintained for a period forming into a historical database used for interesting mining patterns. The historical database is either stored at a single location or many distributed locations. A database has grown by adding more data incrementally. In the modern-day data is captured from sensors and other devices which move in streams when needs to proceed in line with the data flow. In the case of data streams, the data not stored historically and only the processed results temporarily stored for further analysis and decision making

In all the cases the of non-stream data, the database is stored permanently and can be scanned several times whereas the streamed data scanned only once. The data that is flowing in stream needs to be processed in line to find the patterns which are regular and frequent. Once the regular and frequent patterns identified, one can find negative associations which can be further evaluated and presented to the end-user for visualization.

Finding patterns is one of the auspicious features of data mining. Many researchers invented fabulous methods, algorithms to find frequent patterns as well as regular patterns. More emphasis is required to find regular patterns compared to frequent patterns which bank on traditional support count and confidence framework. These methods suffer from outliers and require additional means and measures to eliminate them.

The FP-Tree and RP tree algorithms are more reliable and effective compared to the earlier algorithms. Many developments took place for finding sequential patterns, irregular patterns, and rare patterns, and so on. One can observe that not much of an emphasis made on finding negative patterns which are very few. Also, in real-time scenarios, there is great scope for the importance of negative patterns; for example, penguins are birds but do not fly.

An item set that happens within several transactions is called patterns. The patterns as such will be huge in number processing of which will take lots of time. The number of patterns also keeps increasing as the database increases. The frequent patterns are more important. Frequent patterns can

found by selecting the patterns that meet the minimum threshold value dictated by the user. The support value is an interesting measure selected by the user. Frequency of a pattern is the number of times that patter appears within the database. There is no period fixed in this case. Frequent patterns may be sporadic meaning there may not be any regularity which defines the occurrence behavior of patterns.

Thus there is a problem that frequent patterns may not be regular, and regular patterns may not be frequent. Sometimes there is a necessity to consider both of them. Some of the frequent patterns may bear a maximal property and precludes the patterns that have items which are a subset of the maximal patterns. Thus considering the maximal patterns will reduce the number of patterns making it essay and simple to evaluate and present them in the form required by the end-user. The association between the patterns mined is equally important. Many times, positive associations between the patterns considered concerning either regular, regular-frequent, regular-frequent-maximal.

However, some of the patterns may have a negative association which means one pattern contradicts the other which is the case when one deals with either drug having a different chemical composition or weather forecasting where one whether pattern contradicts another leading to wrong conclusions and decisions. Negative associations are some times more important than the positive associations and therefore needs clear investigations of the same.

The way the patterns are fund differs a lot based on the nature of the data that must be processed. Not streamed data can be processed several times as they are permeant stored, whereas streamed data has to be processed as the data flows and generally will not get stored. The actions taken recognized as the data flows and the actions initiated as and when they are recognized. Data streams are dynamic with numerous variables and complex objects. Sliding window protocol is very much handy in such an environment

In this paper, the issue of finding the negative associations considering that the data is flowing in streams considered and the algorithm is presented to find the negative associations. A method that considers the data as a sliding window presented in this thesis.

## 2. PROBLEM DEFINITION

### Notation

Let TDB the transaction database containing the transaction of the form T= $\{t_1, t_2, t_3 \ldots .t_m\}$. Let m=|TDB| be the size of the transaction database.

Hence for any given task, 0<k<=m, let n be the total number of items available in the TDB such that $I=\{i_1,i_2,i_3 \ldots i_n\}$ be the set of items where for any given $i_j$, j lies between 1 and n, i.e., 1<=j<=n.

All m number of transactions of TDB is in the form ($t_{id}$, S).

Tid stands for transaction ID and $S = \{i_a,i_b,i_c \ldots i_h\}$ is the set of items present in that transaction and 1<=a,b,c,h<=n. Let $P=\{p_1,p_2, p_3, \ldots . p_x\}$be the set of periods.

Let VDB is vertical database consists of item wise transactions of the form V= $\{i_{id}, TS\}$.
id is the item ID, $TS=\{t_q,t_r,t_s \ldots t_t\}$ is the set of transactions. $I_{id}$ belongs to me, and TS is a subset of T.

A period is a numeric value calculated as the difference between two consecutive occurrences of an item (ex. r-q, s-r…). $P_{max}$ is the maximum value of periods for a given item.

Positive regular itemset $PS=\{i_u,i_v,\ldots i_w\}$ is defined as the item set for which $P_{max} <$ □$_{max\_reg}$ ( user defined regularity threshold) where $i_u,i_v,i_w$ belongs to I.

Let NR = {PS1, PS2} be the negative regular item set where PS1 and PS2 be two positive regular item sets, and PS1 □ PS2 ={□}.

Thus the problem is to find regular, frequent, maximal item sets that are negatively associated when the data flows in regular and continuous streams.

## 3. RELATED WORK

Discovering association rules is a complex problem when the database is too large. Agrawal, R et al., [1] have presented two algorithms Apriori and AproiriTid that are different from many other algorithms. The algorithms proposed by them differ in the way candidate items sets are counted and selected. The candidate itemsets are counted and selected using only large itemsets found in the previous iteration. In this case, the database not used from 2$^{nd}$ iteration onwards. They have also shown how the algorithms have been combined to provide a hybrid solution known as AprioriHybrid. The algorithms have been developed based on the principle that an itemset of K items selected by joining the itemsets of K-1 itemsets. The algorithms proposed by them have been providing to be much faster.

Many patterns discovering methods presented in the literature suffer due to the generation of too many irrelevant patterns as the algorithms do not use the existing prior domain knowledge. Balaji Padmanabhan et al., [2] have proposed a method to discover unexpected patterns by utilizing the domain knowledge that the users have, The knowledge that the decision-makers regarded as set expectations and beliefs represented as rules. These rules used as seeds for undertaking the searching for patterns.

Two kinds of approaches are in use for association rule mining that includes either a horizontal or vertical approach. The vertical mining algorithms have been proved to be very

effective. The algorithm uses a fast frequency counting method through the intersection operations on transactions and pruning the unwanted patterns. These algorithms, however, suffer badly when the intermediate results are too many requiring heavy memory. Mohammed J. Zaki et al., [3] have presented a method called Di-Sets, which maintains the differences in the transactions Ids of the candidate patterns used for finding the frequent patterns. The size of the memory required is cut-down drastically.

A model developed can represent the behavior of any process. The deviations of actual behavior from the model are called anomalies. A model representing normal behavior is first constructed with the help of the observed sample considering normally occurring patterns. The model thus can represent either positive or the negative patterns, which in turn detect the kind or detection to be either positive or negative. Fernando Esponda et al., [4] have presented a framework used for computing the trade-off between the positive and negative detection schemes, especially in determining the number of detectors required to maximize the coverage of all the patterns. In reality, the number of patterns is too many that it becomes to difficult the positive and negative patterns. Techniques such as partial matching specify the allowable and unallowable patterns. Choice of a matching rule, however, dictates the trade-off in determining the positive and negative patterns. Fernando Esponda et al., [4] have introduced a new matching rule called "chunks" and the generalization of matching rules done through a method called "crossover closure." Application of these two techniques leads to determining more precise discrimination between positive and negative discrimination.

W. Xindong et al., [5] have presented an efficient method that can be used for mining the positive and negative associations existing in large databases. The scope of Traditional associations rules is extended to include associations such as (A: B ) (: AB) (: A: B) that indicate negative associations. They have also considered a pruning strategy and interesting measure which scales well to large databases.

Positive associational rules generated considering the items that are the enumerations of a transactional database. Negative association rules also are generated, considering both positive and negative associations that exist among the enumerated data items existing in the database. Negative association rules are important as they reveal contradicting items and the data items that complement each other. Classification models can be built using the classifiers generated with the help of negative association rules. Many applications can be built using negative associations. It is necessary to examine a large database for mining negative association rules. Very few algorithms presented in the literature mine the negative association rules, despite the usefulness of those rules. A. Maria-Luiza et al. [6] have presented an algorithm that considers the sliding correlation coefficient threshold as an interesting measure. The algorithms can be used to discover strong negative association rules with strong negative correlations between the antecedents and consequents. The algorithm extends the support-confidence framework.

Many algorithms have presented in the literature for finding negative association rules considering a different perspective. But none of the solutions are satisfactory. C. Cornell et al., [7] have critically examined each of the method/algorithm presented in the literature and then proceeded to present an Apriori-based algorithm which is quite effective to find both valid positive and negative associations that comply with support confidence framework. The upward closure property inbuilt into the algorithm for supporting the generation negative association rules quite effectively.

Many algorithms presented in the literature for mining frequent data sets from static data databases. However, with the advent of internet data flows in streams, especially the data moved from remote sensors into cloud-based storage. Mining is to be done in line with the movement of data in streams dynamically. A quick response to the continuous request has to be provided to satisfy the user requirements. Chih-Hsiang Lin et al., [8] have used time-sensitive sliding windowing techniques for frequent mining itemsets from the data stream. Their algorithm is based on the storage of all frequent itemsets in the memory and also table organized in the memory in which the count of expired data items stored. The number of expired data items stored in the table can be minimized based on available storage space.

A single confidence threshold generally used for finding both positive and negative associations rules of the forms $A \Rightarrow B$, $A \Rightarrow \neg B$, $\neg A \Rightarrow B$, and $\neg A \Rightarrow \neg B$. In this approach, the user is left with a dilemma to either proceed with positive or negative associations. X. Dong et al., [9] have used different confidence thresholds for dealing with four different types of associations one positive and three negative associations. The study of the relationships among the four different confidences revealed that four confidence intervals are to be considered to focus on the four types of associations. Consideration of four confidence helps in generation of misleading rules. X. Dong et al.,[9] have presented the way the chi-squared test used for mining the association rules. They have proposed an algorithm called PNARMC based on chi-squared test considering four confidence thresholds.

The maximal property of data item sets when used will reduce the number of patterns among which the existing associations found. Chuanyao Yang et al., [10] have proposed an algorithm for mining maximal frequent patterns based on the construction of the projected sum tree. The frequent itemsets in a compact manner stored in a tree-like structure. The tree constructed by them is an ordered tree, including the root, intermediate, and child nodes. The algorithm avoids the generation of conditional FP tree

dynamically and recursively. It takes advantage of computational results for developing the ordered tree. The Algorithm proposed by them outperforms when compared to FPmax Algorithm.

P. Kazienko [11] has presented a method for sequential mining patterns, concluded as negative sequential patterns. He has presented that certain types of itemsets do not occur after the occurrence of regular, frequent sequences. They have presented a SPAWN algorithm for sequential mining patterns with negative conclusions.

In the case of some patterns, frequency alone is not a sufficient criterion. Besides, the regularity of the pattern considered, as such kinds of patterns is important for building special kind of applications. Any pattern that occurs at a regular time interval stated by the user is called a regular pattern. There have been some findings in the past related to periodic patterns existing in the time series and sequential data. However, no method is available in the literature for discovering the patterns that occur regularly in a transactional database. Tanbeer, S. K et al., [12] have presented a new concept for mining regular data itemsets from transactional databases. They have proposed an efficient tree-based structure called RP-Tree (Regular pattern tree) that enables mining of regular patterns considering the user-defined regularity threshold.

In the case of most of the existing mining algorithms, the equal weight given to every pattern existing in the database. No semantic significance is attached to any of the existing patterns. Ahmed, C. F et al.,[13] have presented the importance of attaching weights to the frequent patterns. Many algorithms exist in the literature, for determining frequent weighted patterns. These algorithms scan over the entire database for finding the frequent weighted patterns. This approach, however, is not useful, especially when the data is huge and the data flows in streams. The existing algorithms cannot use the recent knowledge of data flowing in streams as considerable processing done for adding the new records to the database. The existing algorithms are weak that the database has to be scanned several times, finding the resultant weighted patterns. Ahmed, C. F et al., [13] proposed an algorithm WFP-MDS (Weighted frequent pattern mining over data streams) using a sliding window technique. The algorithm scans of the data stream, only once, important knowledge required for discovering the pattern is extracted.

Every organization has gone to the state of storing data related to every activity conducted by them with the advent of many data collection devices. The data stored is large that it is difficult to mine such data to discover useful information. Huge data also means that it is possible to discover more realistic and useful patterns. Sequential patterns reveal more interesting knowledge discovered from the data. The existing sequential mining methods are concerned with finding positive behavior of the patterns discovered that help in predicting the next event that will happen after a sequence of patterns. Fahad Anwar et al., [14] have presented a method that mines negative sequential patterns that contradict each other. The method proposed by them discover events/event sets which do not follow any pattern

User preferences discovered through mining text documents. The process as such is complicated as the text contains too many patterns, terms, structure, and also contains too much of a noise. Term-based approaches have been implemented immensely for finding patterns from text-based documents. The term-based approaches have suffered due to the reasons of polysemy and synonymy. Many experiments proved that pattern-based mining did not yield any good result compared to term-based approaches for mining user preferences. Yuefeng Li et al., [15] however, have presented a method that mines both positive and negative patterns in text documents which represent higher-level features leading to the determination of low-level features based on their specificity and distribution in higher-level features.

Most of the data is flowing in streams due to migration of most of the application on to the internet. Discovering interesting from the streamed data is complex and challenging. Support metric based pattern mining the streamed data has been quite successful. However, the occurrence frequency of a pattern found that it is not a proper measure for discovering meaningful patterns. Temporal regularity, on the other hand, is proved to be most appropriate when it comes to online mining the data streams for supporting applications such as stock market. A pattern can be said to be regular if its occurrence happens within the time interval defined by the user. None of the algorithms presented that address mining regular pattern from stream data generated out of online applications. Syed Khairuzzaman Tanbeer et al., [16] have presented a tree-based structure called Regular Pattern Stream tree (RPS-tree), and an efficient mining technique for discovering interesting patterns out of streamed data. They have used a sliding window method; the stream data captured in RPS tree. An efficient tree updating mechanism has also used for updating the tree with the latest data.

The regularity of occurrence of an itemset is interesting compared to the frequency of occurrence of an item set, especially when data flows through online applications such as Gene data analysis, stock market data analysis, etc. Many methods presented in the literature for finding the patterns based on the regularity of the itemsets existing in a data stream. However, no method is in existence that considers the vertical data format and also restricting the database into a single SCAN. G. Vijay Kumar et al.,[17] have presented a method that uses a VRDP table for generating a complete set of regular patterns based on the user-defined threshold from a transactional database.

From the literature, one can find interesting patterns discovered by using the frequency and regularity as the guiding measures. A pattern is said to be the regular, frequent pattern if the itemset occurs frequently and also occurs with the time specified by the user. G. Vijay Kumar et al., [18] have presented an algorithm called RFPDS (Regular, frequent patterns from data streams) that uses sliding window technique using vertical data format, and also that satisfies downward closure property.

Shirin Mirabedini et al.,[19] have presented an overview covering all the methods presented which are related to frequent pattern mining from data streams. Frequent item mining is one of the important method used for carrying either clustering or classification of the data items.

The requirement of mining data streams for discovering interesting patterns has been on the rise as the volume of data flowing across the network has been on the rise. Frequent itemset mining is required most of the times, considering both static and streaming data. The frequently occurring patterns may indicate many interesting like market trends, scientific phenomenon, etc. Pattern finding has been the building block for machine learning-based tasks such as association rule induction etc. In the several past scans of the data have been made to find the interesting patterns. In the case of streamed data, only one scan is possible so that the pattern-finding completed in a single scan of the data stream. The length of the stream is indeterminate and therefore no specific closure for such a data. An initial set of data is collected and stored in a temporary storage area before the streamed data is processed. VE Lee et al., [20] have presented a structured review of online frequent pattern mining techniques. They have classified the methods according to the type of patterns, data, and the time window.

**Juni Yang et al.,[21]** have presented an effective algorithm called DSM-Miner used for mining maximal frequent patterns. In the method they have included, Transaction sliding window method that uses the number of transactions used in each processing phase. A concept called decaying is used to differentiate between old and new transactions. They have proposed constructing SWM-Tree (Sliding window maximum frequent pattern tree) for maintaining the frequent patterns. The root of the SWM-tree is using as the root of an enumerated tree used for searching.

Many algorithms approach presented in the literature for mining frequent items sets suitable for different applications. Not much emphasis is given for regular and frequent itemsets considering the negative associations [22] [23] [24] [25] [26] [27][28][29][30][31]

## 4. COMPARATIVE ANALYSIS

Comparison of exiting algorithms done that mines negative associations considering regular, frequent, and maximal item sets to assess the adequacy of those algorithms and also considering streamed databases. Table 1 shows the comparison. From the table, one can see that none of the existing algorithms are dealing with the most important aspects of the negative associations that include regularity, positive/negative associations, and frequency and interestingness measures considering the streamed databases.

## 5. INVESTIGATIONS AND FINDINGS

### 5.1 Architectural design of Experimental Data Stream

IBM supplied 100,000 records related to sales transactions. Out of these 90,000 (Ninety Thousand), records placed into a flat-file and 10,000 (ten thousand) records placed into another flat file. Both the flat files stored on the same server. From this one, can mean that a static database of 90,000 records and an incremental database of 10,000 records considered. Here database increment is considered as 10,000 records

An Algorithm developed and implemented which can process the negatively associated patterns considering both the data files. The architecture of the distributed processing implemented for determining overall patterns shown in Figure 1.

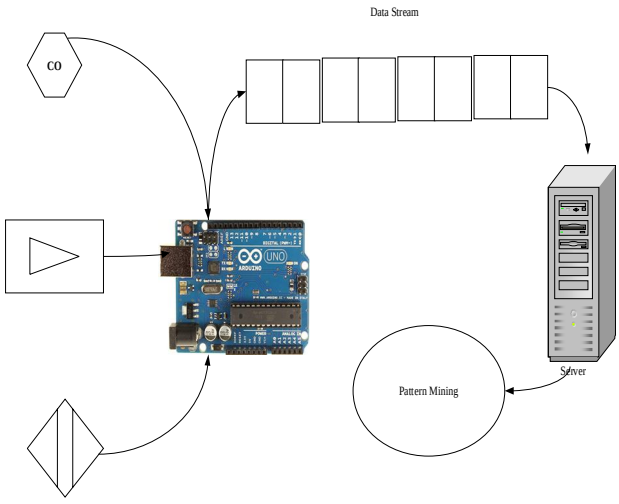


**Figure 1:** Processing In-stream Data

### 5.2 Algorithmic Approaches for finding negative associations from streamed database considering the regularity and frequency of the item sets

This algorithm uses a sliding windowing concept for generating negative patterns. The data canned as it is received and the initial transaction file of 1000 records is

either written into memory or disk. A concurrent process keeps reading the data stored in the transaction file concurrently and update the Horizontal table, which has the details of every item and the transactions in which the item sold. The pattern generation algorithm looks for the transaction data from the instream data and writes the transaction data into a memory-based buffer. Sufficient size of the buffer is allocated at least having the capacity to hold 1000 transactions. The buffer scanned in a round-robin fashion and the horizontal table updated as per the details contained in the transaction table

**Algorithm**

**Process - A**

1. Read first 1000 transactions from instream data and write them into memory buffer as shown in Table 2.

    1. Set the pointer to the beginning of the buffer
    2. Update the Vertical Tab based on the contents of the Transactions, considering each transaction at a time. The content of the vertical tab shown in Table 3.
    3. Place the pointer to the beginning of the buffer
    4. Read the transaction from data instream and write the data into the buffer at the location pointed by the buffer pointer
    5. Increment the Buffer pointer
    6. If the buffer pointer reached the end buffer them makes the buffer pointer equated to the beginning of the buffer.

**Process - B**

7. Repeat the following process in a concurrent process

    Consider the current Item in the vertical format table

    Select next item and prune it if it is not regular and frequent and go to the next item (self-Loop).

    If the next item is regular and frequent, get the intersection of the transactions of the current item and next item

    If the intersection is null, then enter the current and next items into a negative set array

    If the intersection is not null, then get the common elements and see if the count of elements is > regularity threshold decided by the user

    If the common elements satisfy the regularity and frequency constraint, then add the common elements into a vertical table as a new row as they are regular and frequent.

    If the common elements do not satisfy the regularity and frequency constraint, then ignore them

    If all the elements in the vertical table are exhausted, then convert the next item which is next to the current item as a current item and then LOOP

    If all the elements in the vertical table are not exhausted, then move to the next item and LOOP

### 5.3 Experimentation for Mining Negative patterns based on regularity and frequency considering Incremental Databases

The proposed algorithm applied on the IBM supplied data set, and the following results achieved

Step-1

Read the In-stream data and recognize the transactions contained in the data and write the transactions into a transaction buffer until the buffer is filled up with 1000 records. The first 12 transactions written to the buffer shown in Table 2, which shows the list of transactions and the items contained in those transactions. These transactions stored as a Flat file at the server.

**Step-2**

A concurrent process converts the Transactions into a vertical format, as shown in Table 3, which shows the list of items and the transactions in which the item appears**.** The process determines the frequency and regularity and updates the contents in the Vertical format table.

Step-3

**Table 2:** Transaction Table sample data

| TrId | Item Set |
|---|---|
| **1** | **I1 I2 I3 I4 I5 I9 I10 I14** |
| **2** | **I4 I5 I6 I10 I15** |
| **3** | **I2 I3 I7 I13 I14 I15** |
| **4** | **I5 I8 I10 I11 I15** |

| TrId | Item Set |
|---|---|
| **5** | **I1 I3 I5 I6 I9** |
| **6** | **I4 I5 I6 I15** |
| **7** | **I2 I3 I7 I11 I12 I13** |
| **8** | **I5 I8 I11 I12 I14 I15** |
| **9** | **I1 I3 I5 I8 I9** |

| **10** | **I4 I5 I6 I10 I15** |
|---|---|
| **TrId** | **Item Set** |
| **11** | **I2 I3 I7 I8 I13 I14 I15** |
| **12** | **I5 I8 I11 I15** |
| **13** | **I1 I3 I5 I9 I11** |
| **14** | **I4 I5 I6 I14 I15** |
| **15** | **I2 I3 I6 I7 I12 I13** |
| **16** | **I5 I8 I11 I12 I14 I15** |
| **17** | **I1 I3 I5 I6 I9 I10** |
| **18** | **I4 I5 I6 I12 I14 I15** |
| **19** | **I2 I3 I4 I7 I13** |
| **20** | **I5 I8 I11 I12 I15** |
| **21** | **I1 I3 I5 I9 I14** |

**Table 3:** Inverted table – Sample data

| Item Code | TrId | Frequency |
|---|---|---|
| I1 | 1 5 9 13 17 21 | 6 |
| I2 | 1 3 7 11 15 19 | 6 |
| I3 | 1 3 5 7 9 11 13 15 17 19 21 | 11 |
| I4 | 1 2 6 10 14 18 19 | 7 |
| I5 | 1 2 4 5 6 8 9 10 12 13 14 16 17 18 20 21 | 16 |
| I6 | 2 5 6 10 14 15 17 18 | 8 |
| I7 | 3 7 11 15 19 | 5 |
| I8 | 4 8 9 11 12 16 20 | 7 |
| I9 | 1 5 9 13 17 21 | 6 |
| I10 | 1 2 4 10 17 | 5 |
| I11 | 4 7 8 12 13 16 20 | 7 |
| I12 | 7 8 15 16 18 20 | 6 |
| I13 | 3 7 11 15 19 | 5 |
| I14 | 1 3 8 11 14 16 18 21 | 8 |
| I15 | 2 3 4 6 8 10 12 14 16 18 20 | 11 |

Scan the data stream to get the next transaction. As the transaction traced, write the transaction into the transaction buffer. The concurrent process shall read the transaction and update the vertical format table. This process repeated until some pre-fixed deadline reached.

**Step 4**

Find the first regular and frequent item by pruning all the previous items whose regularity is < User has given Maximum Regularity threshold ($\square_{min_reg).\ A}$nd the frequency is > the user given frequency. Here regularity implies the relative occurrence of the Item, computed as the distance between two successive transactions. Considering the ($\lambda_{min_reg}$) = 6. The First regular Item is called Previous-Item. Frequency means the number of transactions in which the item is figuring. The list of items that will be leftover in the vertical format table shown in Table 4.

**Table 4:** Pruning the Items in the vertical table until the first regular Item traced

| Item Code | TrId | Regularity (Periods) | Maximum Regularity of the Item | Frequency of the Item |
|---|---|---|---|---|
| I1 | 1 5 9 13 17 21 | 1 4 4 4 4 | 4 | 6 |
| I2 | 1 3 7 11 15 19 | 1 4 4 4 4 2 | 4 | 6 |
| I3 | 1 3 5 7 9 11 13 15 17 19 21 | 1 2 2 2 2 2 2 2 2 2 | 2 | 11 |
| I4 | 1 2 6 10 14 18 19 | 1 1 4 4 4 4 1 2 | 4 | 7 |
| I5 | 1 2 4 5 6 8 9 10 12 13 14 16 17 18 20 21 | 1 1 2 1 1 2 1 1 2 1 1 2 1 1 2 1 | 2 | 16 |
| I6 | 2 5 6 10 14 15 17 18 | 2 3 1 4 4 1 2 1 3 | 4 | 8 |
| I8 | 4 8 9 11 12 16 20 | 4 4 1 2 1 4 4 1 | 4 | 7 |
| I9 | 1 5 9 13 17 21 | 1 4 4 4 4 4 | 4 | 6 |
| I11 | 4 7 8 12 13 16 20 | 4 3 1 4 1 3 4 1 | 4 | 7 |
| I15 | 2 3 4 6 8 10 12 14 | 2 1 1 2 2 2 2 2 | 2 | 11 |

**Step-5**

Consider each item starting from Previous-item and repeat the following procedure.

1. Consider the next item and let that be current-item
2. Find if the current-item is regular and frequent. If the current-item is not regular and frequent, then prune it.
3. If the current- item is regular and frequent, find Intersection of the transactions of the current item with the previous-item.
4. If the intersection is null, then add the Item set into the negative item-set list.
5. If the intersection is not null, find the regularity and frequency considering the common elements.
6. If the regularity is < ($\lambda_{min\_reg),}$ then add the previous item and the current item set along with its related transaction as an additional record to the vertical database since they are positively associated.
7. If the next item is not the lost entry in the vertical table, Make Current Item as the next Item and loop.

8. If the next item is the last item in the Vertical table, then Previous Item = Previous Item +1 and then Loop.

After this step completed, the pruned items shown in Table 5 and the negatively associated items shown in Table 6 and positively associated items shown in Table 7.

**Table 5:** List of the Pruned item list

| Item Code | TrId | Regularity (Periods) | Maximum Regularity of the Item | Frequency of the item set |
|---|---|---|---|---|
| I7 | 3 7 11 15 19 | 3 4 4 4 4 2 | 4 | 5 |
| I10 | 1 2 4 10 17 | 1 1 2 6 7 4 | 7 | 5 |
| I12 | 7 8 15 16 18 20 | 7 1 7 1 2 2 1 | 7 | 6 |
| I13 | 3 7 11 15 19 | 3 4 4 4 4 2 | 4 | 5 |
| I14 | 1 3 8 11 14 16 18 21 | 1 2 5 3 3 2 2 3 | 5 | 8 |

**Table 6: :**Negatively Associated Item set With Maximum Regularity Threshold 4 and Minimum Support Count 6

| Item Set-1 | Item-set 2 |
|---|---|
| 1 | 15 |
| 4 | 8 |
| 4 | 11 |
| 4 | 8, 11 |
| 6 | 8 |
| 6 | 8 |
| 6 | 11 |
| 6 | 8, 11 |
| 8 | 4,6 |
| 9 | 15 |
| 11 | 4,6 |
| 15 | 1,9 |
| 4,6 | 8,11 |

**Table 7: :**Positively Associated Item set

| Itemset | Trids | Periods | Max Regularity | Support Count |
|---|---|---|---|---|
| 1 | 1 5 9 13 17 21 | 4 4 4 4 4 | 4 | 6 |
| 2 | 1 3 7 11 15 19 | 2 4 4 4 4 | 4 | 6 |
| 3 | 1 3 5 7 9 11 13 15 | 2 2 2 2 2 2 2 | 2 | 11 |
| 4 | 1 2 6 10 14 18 19 | 1 4 4 4 4 1 | 4 | 7 |
| 5 | 1 2 4 5 6 8 9 10 12 | 1 2 1 1 2 1 1 2 | 2 | 16 |
| 6 | 2 5 6 10 14 15 17 | 3 1 4 4 1 2 1 | 4 | 8 |
| 7 | 3 7 11 15 19 | 4 4 4 4 | 4 | 5 |
| 8 | 4 8 9 11 12 16 20 | 4 1 2 1 4 4 | 4 | 7 |
| 9 | 1 5 9 13 17 21 | 4 4 4 4 4 | 4 | 6 |
| 11 | 4 7 8 12 13 16 20 | 3 1 4 1 3 4 | 4 | 7 |
| 13 | 3 7 11 15 19 | 4 4 4 4 | 4 | 5 |
| 15 | 2 3 4 6 8 10 12 14 | 1 1 2 2 2 2 2 | 2 | 11 |
| 1,3 | 1 5 9 13 17 21 | 4 4 4 4 4 | 4 | 6 |
| 1,5 | 1 5 9 13 17 21 | 4 4 4 4 4 | 4 | 6 |
| 1,9 | 1 5 9 13 17 21 | 4 4 4 4 4 | 4 | 6 |
| 2,3 | 1 3 7 11 15 19 | 2 4 4 4 4 | 4 | 6 |
| 2,7 | 3 7 11 15 19 | 4 4 4 4 | 4 | 5 |
| 2,13 | 3 7 11 15 19 | 4 4 4 4 | 4 | 5 |
| 3,5 | 1 5 9 13 17 21 | 4 4 4 4 4 | 4 | 6 |
| 3,7 | 3 7 11 15 19 | 4 4 4 4 | 4 | 5 |
| 3,9 | 1 5 9 13 17 21 | 4 4 4 4 4 | 4 | 6 |
| 3,13 | 3 7 11 15 19 | 4 4 4 4 | 4 | 5 |
| 4,5 | 1 2 6 10 14 18 | 1 4 4 4 4 | 4 | 6 |
| 4,6 | 2 6 10 14 18 | 4 4 4 4 | 4 | 5 |
| 4,15 | 2 6 10 14 18 | 4 4 4 4 | 4 | 5 |
| 5,6 | 2 5 6 10 14 17 18 | 3 1 4 4 3 1 | 4 | 7 |
| 5,8 | 4 8 9 12 16 20 | 4 1 3 4 4 | 4 | 6 |
| 5,9 | 1 5 9 13 17 21 | 4 4 4 4 4 | 4 | 6 |
| 5,11 | 4 8 12 13 16 20 | 4 4 1 3 4 | 4 | 6 |
| 5,15 | 2 4 6 8 10 12 14 16 | 2 2 2 2 2 2 2 | 2 | 10 |
| 6,15 | 2 6 10 14 18 | 4 4 4 4 | 4 | 5 |
| 7,13 | 3 7 11 15 19 | 4 4 4 4 | 4 | 5 |
| 8,11 | 4 8 12 16 20 | 4 4 4 4 | 4 | 5 |
| 8,15 | 4 8 12 16 20 | 4 4 4 4 | 4 | 5 |
| 1,3,5 | 1 5 9 13 17 21 | 4 4 4 4 4 | 4 | 6 |
| 1,3,9 | 1 5 9 13 17 21 | 4 4 4 4 4 | 4 | 6 |
| 2,3,7 | 3 7 11 15 19 | 4 4 4 4 | 4 | 5 |
| 2,3,1 | 3 7 11 15 19 | 4 4 4 4 | 4 | 5 |
| 11,1 | 4 8 12 16 20 | 4 4 4 4 | 4 | 5 |
| 1,3,5 | 1 5 9 13 17 21 | 4 4 4 4 4 | 4 | 6 |
| 1,5,9 | 1 5 9 13 17 21 | 4 4 4 4 4 | 4 | 6 |
| 3,5,9 | 1 5 9 13 17 21 | 4 4 4 4 4 | 4 | 6 |

| Itemset | Trids | Periods | Max Regularity | Support Count |
|---|---|---|---|---|
| 4,5,6 | 2 6 10 14 18 | 4 4 4 4 | 4 | 5 |
| 4,5,1 | 2 6 10 14 18 | 4 4 4 4 | 4 | 5 |
| 4,6,1 | 2 6 10 14 18 | 4 4 4 4 | 4 | 5 |
| 5,6,1 | 2 6 10 14 18 | 4 4 4 4 | 4 | 5 |
| 2,3,7 | 3 7 11 15 19 | 4 4 4 4 | 4 | 5 |
| 2,7,1 | 3 7 11 15 19 | 4 4 4 4 | 4 | 5 |
| 3,7,1 | 3 7 11 15 19 | 4 4 4 4 | 4 | 5 |
| 5,8,1 | 4 8 12 16 20 | 4 4 4 4 | 4 | 5 |
| 5,8,1 | 4 8 12 16 20 | 4 4 4 4 | 4 | 5 |
| 1,3,9 | 1 5 9 13 17 21 | 4 4 4 4 4 | 4 | 6 |
| 1,5,9 | 1 5 9 13 17 21 | 4 4 4 4 4 | 4 | 6 |
| 3,5,9 | 1 5 9 13 17 21 | 4 4 4 4 4 | 4 | 6 |
| 5,8,1 | 4 8 12 16 20 | 4 4 4 4 | 4 | 5 |
| 5,11, | 4 8 12 16 20 | 4 4 4 4 | 4 | 5 |
| 8,11, | 4 8 12 16 20 | 4 4 4 4 | 4 | 5 |
| 2,3,1 | 3 7 11 15 19 | 4 4 4 4 | 4 | 5 |
| 2,7,1 | 3 7 11 15 19 | 4 4 4 4 | 4 | 5 |
| 3,7,1 | 3 7 11 15 19 | 4 4 4 4 | 4 | 5 |
| 4,5,1 | 2 6 10 14 18 | 4 4 4 4 | 4 | 5 |
| 4,6,1 | 2 6 10 14 18 | 4 4 4 4 | 4 | 5 |
| 5,6,1 | 2 6 10 14 18 | 4 4 4 4 | 4 | 5 |
| 5,8,1 | 4 8 12 16 20 | 4 4 4 4 | 4 | 5 |
| 5,11, | 4 8 12 16 20 | 4 4 4 4 | 4 | 5 |
| 8,11, | 4 8 12 16 20 | 4 4 4 4 | 4 | 5 |
| 4,5,6 | 2 6 10 14 18 | 4 4 4 4 | 4 | 5 |
| 2,3,7 | 3 7 11 15 19 | 4 4 4 4 | 4 | 5 |
| 5,8,1 | 4 8 12 16 20 | 4 4 4 4 | 4 | 5 |
| 2,3,7 | 3 7 11 15 19 | 4 4 4 4 | 4 | 5 |
| 4,5,6 | 2 6 10 14 18 | 4 4 4 4 | 4 | 5 |
| 5,8,1 | 4 8 12 16 20 | 4 4 4 4 | 4 | 5 |
| 1,3,5 | 1 5 9 13 17 21 | 4 4 4 4 4 | 4 | 6 |

### 4.4 Pseudo Code

```
Algorithm Bool Reg (ik, TId^Ik, □max_reg,m)
{
      // TId^Ik_first and TId^Ik_last are the first and last Transactions

      Ik ik_First
      =TrId^Ik_first-0;

      If  ( ik_First> □nax_reg ) return F; ik_Reg=ik_First;
      for p= TId^Ik_first +1 to TId^Ik_last

          {
              ik_Next= TId^Ik_p - TId^Ik_p-1; if(ik_Next)>ik_Reg
                then
              ik_Reg=ik_Next;
          }

      Ik_Next=m- TId^Ik_last;

      if(ik_Next)>ik_Reg then ik_Reg=ik_Next;
      if (ik_Reg }> □max_reg) return F;
      return T;
}

Algorithm SW_PRISM(Ik, Ij,n)

{

   Ikj= Ik∪Ij ;

   TId^Ikj = TId^Ik∪TId^Ij;

   if (Reg(Ikj, TId^Ikj, □max_reg)= = F) prune Ikj ;
   else

  {
     VDB= VDB ∪ { Ikj , TId^Ikj };

              n
              =
              n
              +
              1
              ;
   }

}

Algorithm SW_Result (Ik, Ij)

{

  if (Ikj □ VDB) return;
  if (TIdIk∩ TrIdIj) = = {□} ) Result=Result ∪Ikj ;
  else
  return;

}

Algorithm SW_NPRISM ( )

{

        // To find a first regular item for k=1 to n

        if ( Reg( Ik, TId^Ik, □max_reg,m )= = F) prune Ik
        else

        break;
```

```
// To find remaining regular items

for j= k+1 to n

{

  if ( Reg( Ij, TIdIj, □max_reg )= = F) prune Ij;
  else
  break;

{

SW_PRISM(Ik,Ij);

SW_Result(ik,ij);

}
```

## 5 ANALYSIS OF IBM DATA

The IBM supplied data analyzed for different sizes of the samples drawn in terms of 20,000, and 60,000. The data analyzed with different regularity percentages. The number of negative frequent regular patterns, for different maximum regularity and. The cross-sections of the 20,000 records selected from 1,00,000 IBM data shown in Table 8 and the vertical format of the raw data shown in **Table 9.**

**Table 8:** Sample IBM 20,000 record dataset

| TrId | Itemset |
|---|---|
| 1 | 25 52 164 240 274 328 368 448 538 561 630 687 730 775 825 834 |
| 2 | 39 120 124 205 401 581 704 814 825 834 |
| 3 | 35 249 674 712 733 759 854 950 |
| 4 | 39 422 449 704 825 857 895 937 954 964 |
| 5 | 15 229 262 283 294 352 381 708 738 766 853 883 966 978 |
| 6 | 26 104 143 320 569 620 798 |
| 7 | 7 185 214 350 529 658 682 782 809 849 883 947 970 979 |
| 8 | 227 390 |
| 9 | 71 192 208 272 279 280 300 333 496 529 530 597 618 674 675 720 855 914 932 |
| 10 | 183 193 217 256 276 277 374 474 483 496 512 529 626 653 706 878 939 |
| 11 | 161 175 177 424 490 571 597 623 766 795 853 910 960 |
| 12 | 125 130 327 698 699 839 |
| 13 | 392 461 569 801 862 |
| 14 | 27 78 104 177 733 775 781 845 900 921 938 |
| 15 | 101 147 229 350 411 461 572 579 657 675 778 803 842 903 |
| 16 | 71 208 217 266 279 290 458 478 523 614 766 853 888 944 969 |
| 17 | 43 70 176 204 227 334 369 480 513 703 708 835 874 895 |
| 18 | 25 52 278 730 |
| 19 | 151 432 504 830 890 |
| 20 | 71 73 118 274 310 327 388 419 449 469 484 706 722 795 810 844 846 918 |
| 21 | 130 274 432 528 967 |
| 22 | 188 307 326 381 403 523 526 722 774 788 789 834 950 975 |
| 23 | 89 116 198 201 333 395 653 720 846 |
| 24 | 70 171 227 289 462 538 541 623 674 701 805 946 964 |
| 25 | 143 192 317 471 487 631 638 640 678 735 780 865 888 935 |
| 26 | 17 242 471 758 763 837 956 |
| 27 | 52 145 161 283 375 385 676 721 731 790 792 885 |
| 28 | 182 229 276 529 |
| 29 | 43 522 565 617 859 |
| 30 | 12 296 350 354 401 548 684 740 774 775 782 841 937 |

**Table 9:** Sample IBM 20,000 record dataset in verticle format

| Item Number | Transaction ID |
|---|---|
| 1 | 55 136 152 187 227 236 414 557 595 659 745 775 887... |
| 2 | 96 162 179 313 341 578 915 1189 1269 1278 1404 147... |
| 3 | 737 1057 1108 1179 1530 1823 2024 2300 2415 2494 2... |
| 4 | 159 197 266 379 445 491 791 982 1082 1174 1223 126... |
| 5 | 35 59 143 165 292 360 388 393 471 635 693 702 871 ... |
| 6 | 56 161 219 255 322 323 349 361 723 734 736 750 802... |
| 7 | 7 178 252 397 754 769 868 970 1094 1162 1251 1275 ... |
| 8 | 91 121 130 215 257 324 326 390 438 439 448 455 464... |

| Item Number | Transaction ID |
|---|---|
| 10 | 44 147 224 288 354 366 381 396 498 507 529 595 623... |
| 11 | 248 261 482 672 779 838 1063 1360 1459 1568 1781 1... |
| 12 | 30 83 86 101 176 185 235 254 259 291 327 350 395 4... |
| 13 | 5497 5586 7010 8372 10015 13935 16400 17122 |
| 14 | 834 882 890 1215 1439 1968 2556 2661 2809 3211 378... |
| 15 | 5 421 876 911 1099 1278 1516 1544 2004 2040 2754 2... |
| 16 | 305 1247 1355 2095 2597 2746 2885 3198 4506 5016 5... |
| 17 | 26 77 78 82 102 186 188 416 483 509 620 660 682 73... |
| 18 | 86 144 347 380 472 717 931 1092 1123 1154 1249 136... |
| 19 | 495 1415 1986 2082 3825 3853 4378 4881 5102 5195 5... |
| 20 | 433 512 6779 8099 8852 15866 15966 16429 |
| 21 | 46 67 71 97 133 159 187 270 277 293 330 512 632 64... |
| 22 | 77 597 859 1160 1399 2740 2835 2896 2956 3232 3293... |
| 23 | 377 400 1016 1232 1248 1659 1891 2405 2857 4472 47... |
| 24 | 73 553 831 1995 2258 2919 3309 3336 4733 5348 6650... |
| 25 | 1 18 172 308 382 400 637 658 675 781 828 849 979 1... |
| 26 | 6 47 221 594 825 879 1303 1500 1644 1660 1718 1748... |
| 27 | 14 112 113 133 215 260 287 292 399 437 474 497 502... |
| 28 | 34 165 253 264 322 353 376 388 393 525 530 564 718... |
| 29 | 968 3154 3986 4680 5563 5584 5945 6788 7282 9120 9... |
| 31 | 39 53 114 194 221 237 241 322 325 327 403 426 471 ... |

Frequent regular items and Negative frequent regular item sets mined for a sample size of 20,000 records and varying the Maximum regularity (600,800,1000) and keeping Minimum frequency (350, 400, 450) and the number of time sets mined are shown in Table 10

The number of Frequent regular and negative frequent regular item sets mined are shown in Figure 2 Keeping Maximum regularity as 600 and the minim frequency set at three levels (350, 400, 450, One can see from the figure that Frequent regular items stabilizes as one moves from minimum support from 350 to 450 while the negative frequent regular itemsets steeply drops.

**Table 10:** Frequent Regular and negative frequent regular Itemsets for a sample size of 20,000

| TT | MR (maximum Regularity | MF (Minimum Frequency) | FR (Frequent Regular Item Sets | NFR (Negative Frequent Regular) Item sets |
|---|---|---|---|---|
| 20000 | 600 | 350 | 368 | 218 |
| | | 400 | 342 | 114 |
| | | 450 | 324 | 65 |
| | 800 | 350 | 421 | 602 |
| | | 400 | 377 | 196 |
| | | 450 | 346 | 97 |
| | 1000 | 350 | 449 | 666 |
| | | 400 | 380 | 211 |
| | | 450 | 347 | 98 |

The number of Frequent regular and negative frequent regular item sets mined are shown in Figure 3 Keeping Maximum regularity as 800 and the minim frequency set at three levels (350, 400, 450, One can see from the figure that Frequent regular items stabilizes as one moves from minimum support from 350 to 450 while the negative frequent regular itemsets steeply drops.

Figure 2 Behavior of Frequent regular items and negative frequent regular item keeping Maximum regularity = 800 and Regulalirty caring from 600 to 1000

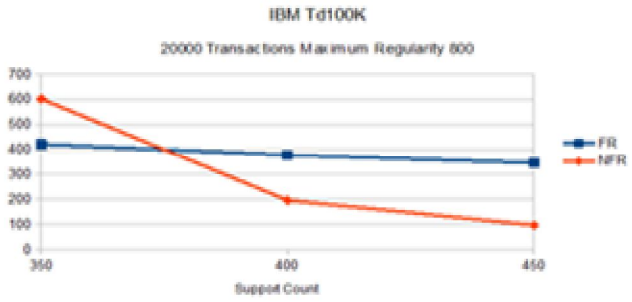


**Figure 3:** Behavior of Frequent regular items and negative frequent regular item keeping Maximum regularity = 800 and Regulalirty from 600 to 1000

The number of Frequent regular and negative frequent regular item sets mined are shown in Figure 4 Keeping Maximum regularity as 1000 and the minim frequency set at three levels (350, 400, 450, One can see from the figure that Frequent regular items stabilizes as one moves from

minimum support from 350 to 450 while the negative frequent regular itemsets steeply drops.

Figure 5 provides a comparative picture of variations infrequent regular and negative frequent regular item sets for different minimum support and maximum regularity for understanding the behavior of itemsets. From the figure, it one can see that the frequent regular items stabilize at maximum regularity of 1000 and a minimum frequency of 450.


**Figure 4:** Behavior of Frequent regular items and negative frequent regular item keeping Maximum regularity = 1000 and Regulalirty caring from 600 to 1000

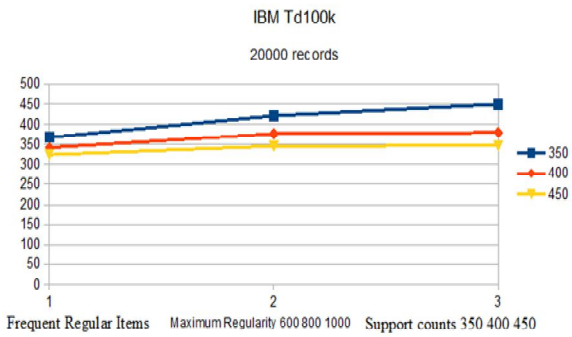


**Figure 5:** Behavior Frequent regular items concerning Maximum regularity and minimum frequency

## 6. CONCLUSION

Data constantly moves in a data stream. Data never stored as a result; only one scan of the data done as the data moves in a stream. Data is never static. Data as such is not stored any ware permanently when it comes to data streams.

Frequent patterns generated without much concern with the time during which the item set occurs. Frequency is just a count. The regularity of occurrence of a pattern is more important, which means the patterns that happen within a period is important. Item sets / frequent itemsets found from a fixed time frame and frequency. All the three dimensions of pattern finding ({regularity}, {regularity, Frequency}, {regularity, Frequency, Maximally}) investigated so that the patterns found from different dimensions can be investigated and used as per their applicability

A reference transaction data needs to be created and held into a buffer which can be used to update a vertical table, which is the primary source for finding the negative associations. An iterative process is needed to select the transactions and held into a buffer and also process them to update a vertical format table. The vertical format table also must be processed in an iterative loop to fine the regular, frequent, and maximal patterns and then move on to find the patterns that yield negative associations.

## REFERENCES

1. Agrawal, R., & Srikant, R. “Fast algorithms for mining association rules. In Proceedings of the 20th international conference on very large databases” pp. 487–499, September 1994.
2. Balaji Padmanabhan, AlexanderTuzhlin,” A Belief-Driven Method for Discovering Unexpected Patterns” in KDD-98 Proceedings 1998
3. Mohammed J. Zaki, Karam Gouda,” Fast Vertical Mining Using Diffsets,” SIGKDD ’03, August 2427, 2003
4. Fernando Esponda, Stephanie Forrest, and Paul, “Helman A Formal Framework for Positive and NegativeDetection Schemes,” IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS— PART B: CYBERNETICS, VOL. 34, NO. 1, FEBRUARY 2004
5. W. Xindong, Z. Chengqi and Z. Shichao, "Efficient mining of both positive and negative association rules," ACM Transactions on Information Systems (TOIS), vol. 22, pp. 381-405, 2004.
6. A.Maria-Luiza, and R. Z. Osmar, "Mining positive and negative association rules: an approach for confined rules," in Proceedings of the 8th European Conference on Principles and Practice of Knowledge Discovery in Databases, Pisa, Italy, 2004, pp. 27-38.
7. C. Cornells, Y. Peng, Z. Xing, and C. Guoqing, "Mining Positive and Negative Association Rules from Large Databases," in IEEE Conference on Cybernetics and Intelligent Systems, 2006, pp. 1-6.
8. Chih-Hsiang Lin, Ding-Ying Chiu, Yi-Hung Wu, Arbee L. P. Chen, “Mining Frequent Itemsets from Data Streams with a Time-Sensitive Sliding Window,” in Society for industrial and applied mathematics, 2006.
9. X. Dong, F. Sun, X. Han, R. Hou, “Study of positive and negative association rules based on multi-confidence and chi-squared test,” in ADMA06, in

LNCS, vol.4093, Springer-Verlag, Berlin–Heidelberg, 2006, pp.100–109.

10. Chuan Yao Yang1 Yuqin Li1 Chenghong Zhang2 Yunfa Hu1 "A Novel Algorithm of Mining Maximal Frequent Pattern Based on Projection Sum Tree," IEEE transactions. 2007.
11. P. Kazienko, "Mining Sequential Patterns with Negative Conclusions," in Proceedings of the 10th international conference on Data Warehousing and Knowledge Discovery (DaWaK '08) Turin, Italy: Springer-Verlag, 2008, pp. 423-432.
12. Tanveer, S.K., Ahmed, C.F., Jeong, B.-S., Lee, Y.-K.: Mining Regular Patterns in Transactional Databases. IEICE Trans. on Inf. & Sys. E91-D(11), 2568–2577 (2008)
13. Ahmed, C. F., Tanveer, S. K., Jeong, B. S., & Lee, Y. K. "An efficient algorithm for sliding window-based weighted frequent pattern mining over data streams."IEICE Transactions, Vol. 92-D(7), pp. 1369–1381, 2009.
14. Fahad Anwar, Ilias Petrounias, Tim Morris, Vassilis Kodogiannis, " Discovery of events with negative behavior against given sequential patterns" -1-4244-5164-7/10/$26.00 ©2010 IEEE
15. Algarin, N. Zhong, "Mining positive and negative patterns for relevance feature discovery," in KDD'2010, 2010, pp.753–762.
16. Syed Khairuzzaman Tanbeer, Chowdhury Farhan Ahmed, and Byeong-Soo Jeong," Mining Regular Patterns in Data Streams," DASFAA 2010, Part I, LNCS 5981, pp. 399–413, 2010, Springer-Verlag Berlin Heidelberg 2010
17. G. Vijay Kumar, M. Sreedevi, NVS. Pavan Kumar, "Mining Regular Patterns in Transactional Databases using Vertical Format," International Journal of Advanced Research in Computer Science, 2 (5), Sept-Oct, 2011
18. G. Vijay Kumar, V. Valli Kumari," Sliding window technique to mine regular, frequent patterns in data streams using the vertical format," Computational Intelligence & Computing Research (ICCIC), 2012 IEEE International Conference.
19. Shirin Mirabedini, Mahdi Ahmadi Panah, Maryam Darbanian, "Frequent Pattern Mining in Data Streams," Journal of Engineering and Applied Sciences 12(4); 857-863, 2014
20. VE Lee, R. Jin, G. Agarwal, Frequent Pattern Mining in Data Streams**,** *Frequent Pattern Mining*, DOI 10.1007/978-3-319-07821-2_9, © Springer International Publishing Switzerland 2014
21. Juni Yang, Yanjun Wei, Fenfen Zhou, "An efficient algorithm for mining maximal frequent patterns over data streams," 7th International Conference on Intelligent Human-Machine Systems and Cybernetics IEEE 2015
22. Changala, R., Rajeswara Rao, D., Evaluation and analysis of discovered patterns using pattern classification methods in text mining, ARPN Journal of Engineering and Applied Sciences, 13(11), pp. 3706-3717,2018
23. Kolli, S., Sreedevi, M., Prototype analysis of different data mining classification and clustering approaches, ARPN Journal of Engineering and Applied Sciences,13(9), pp. 3129-3135,2018
24. Deshpande, L., Rao, M.N., Concept drift identification using classifier ensemble approach, International Journal of Electrical and Computer Engineering,8(1), pp. 19-25,2018
25. Changala, R., Rajeswara Rao, D. T., A survey on the development of pattern evolving model for discovery of patterns in text mining using data mining techniques, Journal of Theoretical and Applied Information Technology,95(16), pp. 3974-3981,2017
26. Wagner, S.S., Rajarajeswari, P.., Parallel frequent dataset mining and feature subset selection for high dimensional data on Hadoop using map-reduce, International Journal of Applied Engineering Research, 12(18), pp. 7783-7789,2017
27. Vijay Kumar, G., Krishna Chaitanya, T., Pratap, M., Mining popular patterns from the multidimensional database, Indian Journal of Science and Technology, 9(17),93106
28. Gangadhar, M.N.S., Sreedevi, M., Regular pattern mining on dynamic databases using vertical formate on given user regularity threshold, Journal of Theoretical and Applied Information Technology, 86(3), pp. 360-364, 2016
29. Greeshma, L., Pradeepini, G., Input split frequent pattern tree using MapReduce paradigm in Hadoop, Journal of Theoretical and Applied Information Technology, 84(2), pp. 260-271
30. Greeshma, L., Pradeepini, G., Mining Maximal Efficient Closed Itemsets Without Any Redundancy, Advances in Intelligent Systems and Computing, 433, pp. 339-347
31. Real-time Streaming Data Analysis using Spark, International Journal of Emerging Trends in Engineering Research, Volume 6, No.1, 2018, pp. 1-5
32. Career Guidance through TIC-TAC-TOE Game, International Journal of Emerging Trends in Engineering Research, Volume 7, No.6, 2019, pp. 25-31

Table 1 Comparative Analysis of Algorithms – Negative Associations – Regular and Frequent - using Data Streams

| Algorithm Serial Number | Main Author | Interestingness measures | | | | | Occurrence Behaviour | | | | | Type of Associations | | Mining technique |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Support | Confidence | Correlation | Multi support | Multi Correlation | Regularity | Irregularity / Rare | Frequent | Maximal | Natural | Positive Associations | Negative Associations | |
| 1 | Agarwal R | | | | | | | | | | √ | | | Apriori |
| 2 | Balaji Padmanabhan | | | | | | | | | | √ | | | |
| 3 | Mohammed J. Zaki | | | | | | | | | | √ | | | Di sets |
| 4 | Fernando Esponda | | | | | | | | | | √ | √ | √ | |
| 5 | W. Xindong | √ | | | | | | | √ | | | √ | √ | |
| 6 | A. Maria-Luiza | | | √ | | | | | | | | √ | √ | |
| 7 | C. Cornell | | | | | | | | | | | √ | √ | Apriori |
| 8 | **Chih-Hsiang Lin** | | | | | √ | | | | | | √ | √ | |
| 9 | X. Dong | | | | | √ | | | | | | √ | √ | |
| 10 | Chuanyao Yang | | | | | | | | | | | √ | √ | MFTree |
| 11 | P. Kazienko | | | | | | | | | | | | √ | SPAWN |
| 12 | Tanbeer, S. K | √ | | | | | √ | | | | | √ | √ | |
| 13 | Ahmed, C. F | √ | | √ | | | | | | | √ | | | WFP-MDS |
| 14 | **Fahad Anwar** | √ | | | | | | | | | | | √ | |
| 15 | Yuefeng Li | | | | | | | | | | | √ | √ | |
| 16 | Syed Khairuzzaman Tanbee | √ | | | | √ | | | | | | √ | √ | |
| 17 | G. Vijay Kumar | | | | | | √ | | | | | √ | | VRDP |
| 18 | G. Vijay Kumar | | | | | | √ | | √ | | | V | | RFPDS |
| 22 | Junrui Yang | √ | | | | | | | | | √ | | | DSM-Miner |