International Journal of Current Engineering and Technology
©2021 INPRESSCO®, All Rights Reserved

E-ISSN 2277 – 4106, P-ISSN 2347 – 5161
Available at http://inpressco.com/category/ijcet

Research Article

Mining Frequent Pattern on Big Data using Map Reducing Technique

Ms.Swarupa Kulkarni and Prof.Priyanka Kedar

Department of Computer Engineering, Dhole Patil College of Engineering Pune, India.

Received 10 Nov 2020, Accepted 10 Dec 2020, Available online 01 Feb 2021, **Special Issue-8 (Feb 2021)**

Abstract

In the existing system Conventional cluster ensemble approaches have several limitations. The must-connect limitation implies that two element vectors ought to be relegated to a similar group, while they can't interface imperatives implies that the two component vectors can't be doled out to a similar bunch. A large portion of the group troupe strategies can't accomplish acceptable outcomes on high dimensional datasets. Conventional example mining calculations are not reasonable for really huge information, displaying two fundamental difficulties to be settled: computational multifaceted nature and primary memory prerequisites. A progression of calculations dependent on the MapReduce system and the Hadoop open-source usage have been proposed here. All the proposed models depend on the notable Apriori calculation and the MapReduce system. The proposed calculations are isolated into three fundamental gatherings. Two calculations Apriori MapReduce (AprioriMR) and iterative AprioriMR (IAprioriMR) are appropriately intended to separate examples in huge datasets. These calculations remove any current thing set in information in any case their recurrence. Pruning the hunt space by methods for the antimonotone property. Two extra calculations space pruning AprioriMR (SPAprioriMR) and top AprioriMR (TopAprioriMR) are proposed with the point of finding any incessant example accessible in information. Maximal successive examples. A last calculation maximal AprioriMR (MaxAprioriMR) is additionally proposed for mining dense portrayals of successive examples, i.e., visit designs with no continuous supersets.

Keywords: *Big Data, Hadoop, Data Mining*

1. Introduction

Visit thing set mining (FIM) is an essential research subject in information mining. Be that as it may, the conventional FIM may find a lot of incessant however low-esteem thing sets and lose the data on important thing sets having low selling frequencies. Thus, it can't fulfill the prerequisite of clients who want to find thing sets with high utilities, for example, high benefits. A thing set is said to be visit if its help is no not exactly a given least help edge. Many examinations have been directed on this point. Nonetheless, a significant confinement of FIM is its suspicions that all things have a similar significance to the client (e.g., unit benefit or weight) and that things may not appear(quantity) more than once in every exchange. These presumptions frequently don't hold, in actuality. For instance, in exchange databases, things may have diverse unit benefits, and things in exchanges might be related with various buy amounts.[8] Additionally, in actuality, applications, retailers might be increasingly keen on discovering thing sets that return a high benefit as opposed to finding regular thing sets. Utility mining, which alludes to the disclosure of thing sets with utilities higher than a client determined least utility limit, is a significant undertaking and has a wide scope of uses, particularly in web based business. In any case, setting a fitting least utility limit is a troublesome issue. On the off chance that the base edge is set excessively low, such a large number of high utility thing sets will be created and it requires some investment to register, while setting the base limit too high would bring about too not many outcomes.

Setting suitable least utility edge by experimentation isn't exceptionally proficient. To accurately control the yield measure and find the thing sets with the most elevated utilities without setting the limits, a promising arrangement is to rethink the undertaking of mining HUIs as mining top-k high utility thing sets (top-k HUIs).[6] The thought is to let the clients determine k, i.e., the quantity of wanted thing sets, rather than indicating the base utility limit. Setting k is more agreeable than setting the edge since k speaks to the quantity of thing sets that the clients need to discover.

2. Review of literature

In this work, Mining class affiliation rules (CARs) with the thing set requirement is worried about the disclosure of standards, which contain a lot of explicit things in the standard precursor and a class mark in the standard ensuing. This assignment is ordinarily experienced in mining medicinal information. For instance, while characterizing which segment of the populace is at high hazard for the HIV disease, disease transmission specialists frequently focus on tenets which incorporate statistic data, for example, sexual orientation, age, and conjugal status in the standard forerunner, and HIV-Positive in the standard ensuing. There are two gullible methodologies to take care of this issue, in particular pre-preparing and post-handling. The post-handling techniques need to produce and think about an enormous number of competitor CARs while the execution of the pre-preparing strategies rely upon the quantity of records sifted through. In this manner, such methodologies are tedious. This examination proposes an effective strategy for mining CARs with the item set requirement dependent on a cross section structure and the contrast between two arrangements of article identifiers (diffset) [1]

In this work, proposes the Apriori Algorithm based continuous direction design mining calculation to productively and successfully handle the direction database exchange. Before that the direction dataset is extricated from a content document and is transported in to an Oracle database in the wake of doing the underlying information cleaning process. Introductory recurrence include is done Oracle database utilizing its programming highlight. At that point the information is written in the working framework at that point further handling is done to locate the incessant direction design. Favorable position of this technique is later emphases are a lot quicker than the underlying cycles of the calculation. The outcomes gotten by this technique are progressively exact and solid. This calculation utilizes extensive arrange set property. Every cycle in this calculation can be parallelized with the goal that execution time can be decreased. Progressively over this calculation is anything but difficult to actualize. Burden of this strategy are, it utilizes a create, prune and test approach produces hopeful facilitate sets (1organize, 2-arrange, 3-arrange,…), to check the created grouping of directions are as of now produced or not, and tests in the event that they are visit by filtering the database and tallying their help each time. Age of applicant arrange sets is costly (in both reality). Since age and pruning steps are in memory occupant, it needs more RAM. Another burden is it needs n+1 database checks, n is the length of the directions in the longest pattern. [2]

In this work most existing calculations mine incessant examples from customary exchange databases that contain exact information. In these databases, clients unquestionably know whether a thing (or an occasion) is available in, or is missing from, an exchange in the databases. Nonetheless, there are some genuine circumstances in which one needs to manage indeterminate information. In such information clients are unverifiable about the nearness or nonappearance of a few things or occasions. For instance, a doctor may very speculate (yet can't ensure) that a patient experiences an explicit malady. The vulnerability of such doubt can be communicated as far as existential likelihood. Since there are some genuine circumstances in which information are dubious, proficient calculations for mining indeterminate information are sought after. Two calculations have been proposed for mining regular examples from indeterminate information. The past two calculations pursue the flat information portrayal. In this paper we considered the issue of mining successive itemsets from existential dubious information utilizing the Tidset vertical information portrayal. We presented the U-Eclat calculation, which is a changed variant of the Eclat calculation, to take a shot at such datasets. An execution examine is directed to feature the effectiveness of the proposed calculation likewise a near report between the proposed calculation and the notable calculation UFdevelopment is led and demonstrated that the proposed calculation outflanks the UF-growth.[3]

In this work, authors have proposed new effective example mining calculations to work in enormous information. All the proposed models depend on the notable Apriori calculation. This calculation has been additionally proposed for blending consolidated portrayals of regular examples. Pruning the hunt space by methods for hostile to monotone property. Two extra calculations have been proposed with the point of finding any successive example accessible in information. In Future, We will utilize the Top – K Ranking Algorithm to locate the best k visit designs from the given dataset. Positioning capacities are assessed by an assortment of methods; one of the least difficult is deciding the accuracy of the principal k top-positioned results for some settled k; Frequently, calculation of positioning capacities can be streamlined by exploiting the perception that just the overall request of scores matters, not their supreme esteem; thus terms or factors that are autonomous of the highlights might be expelled, and terms or factors that are free of the element might be pre-computed and put away with the dataset.[4]

In this ,a novel G3P algorithm for mining SD was presented and described in depth. The aim of this algorithm is to harness the features of G3P to solve the requirements of SD, i.e., rules that have a clear and flexible structure and obtain interesting subgroups according to a series of quality measures, which could be classified as complexity, generality, precision, and interest. One of the main features of this algorithm is its ability to discover comprehensible subgroups, i.e., a few set of rules having few variables, in an evolutionary way and without the need for tuning many parameters.[5]

3. Existing system

Conventional cluster ensemble approaches have several limitations: They do not consider how to make use of prior knowledge given by experts, which is represented by pair wise constraints. Pair shrewd requirements are frequently characterized as the must-interface imperatives and the can't connect limitations. The must-connect requirement implies that two component vectors ought to be doled out to a similar bunch, while the can't interface limitations implies that two element vectors can't be allotted to a similar group. The majority of the bunch outfit strategies can't accomplish acceptable outcomes on high dimensional datasets. Not all the gathering individuals add to the outcome.

4. Problem statement

The pattern mining is one of the important tasks to extract meaningful and useful information from raw data. This task aims to extract item-sets that represent any type of homogeneity and regularity in data. Traditional pattern mining algorithms are not suitable for truly big data, presenting two main challenges to be solved: computational complexity and main memory requirements. a series of algorithms based on the MapReduce framework and the Hadoop opensource implementation have been proposed.

5. Proposed system

In this system proposal of new efficient pattern mining algorithms to work in big data is given. All of them rely on the MapReduce framework and the Hadoop open-source implementation [2]. Two of these algorithms (AprioriMR and IAprioriMR) enable any existing pattern to be discovered. Two additional algorithms (SPAprioriMR and TopAprioriMR) use a pruning strategy for mining frequent patterns. Finally, an algorithm for mining MaxAprioriMR is also proposed.

6. System architecture

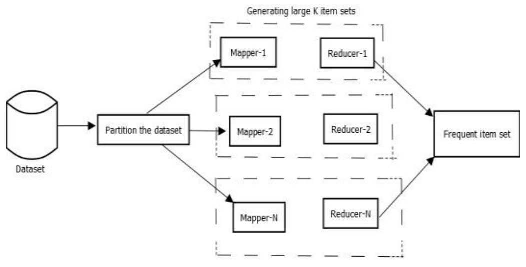


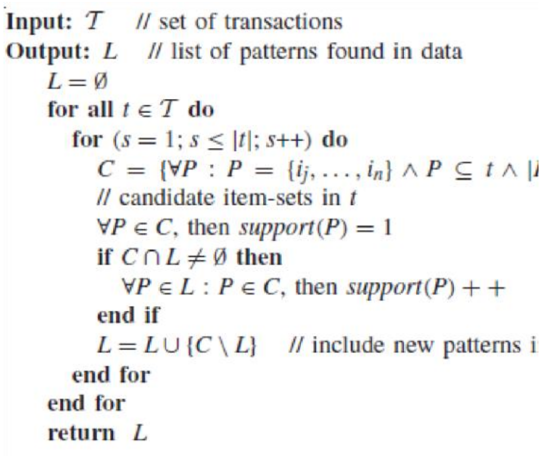
Figure1. System architecture

Methodology

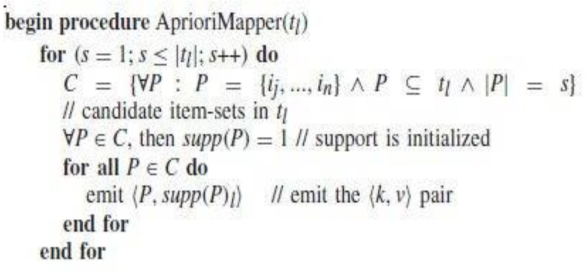
1. Pre-Evaluation of Min_Util: Though TKU provides a way to mine top-k HUIs, min_util is set to 0 before the construction of the UP-Tree. This results in the construction of a full UP-Tree in memory, which degrades the performance of the mining task. If min_util could be raised before the construction of the UP-Tree and prune more unpromising items in transactions, the number of nodes maintained in memory could be reduced and the mining algorithm could achieve better performance. Based on this idea, we propose a strategy named PE (Preevaluation Step) to raise min_util during the first scan of the database.
2. Construction Of UP-Tree: A UP-Tree can be constructed by scanning the original database twice. In the first scan, the transaction utility of each transaction and TWU of each item are computed. During the second database scan, transactions are reorganized and then inserted into the UP-Tree. 3. Generating PTKHUIs: TKU algorithm uses UP tree to generate the potential top k high utility item sets. Parallelly, it raises the value of minimum utility threshold dynamically during the generation of PTKHUIs.
4. Identifying Top-K: HUIs From PTKHUIs After identifying PTKHUIs, TKU calculates the exact utility of PTKHUIs by scanning the original database once again, to identify the top-k HUIs.
5. Construction of Utility List Structure: In the TKO algorithm, each item (set) is associated with a utilitylist. The utility lists of items are called initial utilitylists, which can be constructed by scanning the database twice. In the first database scan, the TWU and utility values of items are calculated. During the second database scan, items in each transaction are sorted in order of TWU values and the utility-list of each item is constructed.
6. Finding Top-K HUIs: In the TKO algorithm, initially, a list of top-k high utility item sets is generated .Parallelly, it raises the value of minimum utility threshold dynamically and updates the list of top-k high utility item sets.

7. Algoritm

a. Original Apriori Algorithm



```
Input: $\mathcal{T}$    // set of transactions
Output: $L$    // list of patterns found in data
  $L = \emptyset$
  for all $t \in \mathcal{T}$ do
    for $(s = 1; s \leq |t|; s{+}{+})$ do
      $C = \{\forall P : P = \{i_j, \ldots, i_n\} \wedge P \subseteq t \wedge |$
      // candidate item-sets in $t$
      $\forall P \in C$, then $support(P) = 1$
      if $C \cap L \neq \emptyset$ then
        $\forall P \in L : P \in C$, then $support(P) + +$
      end if
      $L = L \cup \{C \setminus L\}$    // include new patterns i
    end for
  end for
  return $L$
```



```
begin procedure AprioriMapper($t_l$)
  for $(s = 1; s \leq |t_l|; s{+}{+})$ do
    $C = \{\forall P : P = \{i_j, \ldots, i_n\} \wedge P \subseteq t_l \wedge |P| = s\}$
    // candidate item-sets in $t_l$
    $\forall P \in C$, then $supp(P) = 1$ // support is initialized
    for all $P \in C$ do
      emit $\langle P, supp(P)_l \rangle$    // emit the $\langle k, v \rangle$ pair
    end for
  end for
```

b. Apriori MR Algorithm

PSEUDO CODE

- Join step: is generated by joining with itself
- Prune Step: any (k-1) item set that is not frequent cannot be a subset of a frequent kitem set

Pseudo-code

```
$C_k$: Candidate item set of size k
        $L_k$ : Frequent item set of size k
    $L_1$= {frequent items};
For (k=1; $L_k$!=Φ;k++) do begin
$C_{k+1}$ = candidates generated from $L_k$;
For each transaction $t$ in database do
    Increment the count of all candidates in $C_{k+1}$
    Those are contained in $t$
$L_{k+1}$ = candidates in $C_{k+1}$ with min_support
  End
Return $U_k$ $L_k$;
```

8. System requirements

A. Software Requirement

1) Operating System : Windows7 AND ABOVE
2) Application Server : Tomcat5.0/6.X, Glassfish
3) Front End : HTML, Java, Jsp
4) Scripts : JavaScript

9. Experimental Analysis

Input Dataset:

Dataset	#Trans.	Avg. Length of Trans.	#Items	Type
Foodmart	4,141	4.4	1,559	Sparse
Retail	88,162	10.3	16,470	Sparse
Chainstore	1,112,949	7.2	46,086	Large
Mushroom	8,124	23	119	Dense
Chess	3,196	37	76	Dense
Accident	340,183	33.8	468	Dense
T12I8D100K	100,000	12	1,000	Sparse

5) Server side Script : Java Server Pages. 6) Database : My sql 5.5 7) Database Connectivity : JDBC.

B. Hardware Requirement

1) Processor : Intel
2) CPU Speed : 1.1 GHz or Higher RAM : 2 GB or Higher

3) Hard Disk : 100 GB or Higher

TKU with min_util=0 v/s TKU with min_util non zero
TKO with min_util=0 v/s TKO with min_util non zero
TKU with min_util=0 v/s TKO with min_util=0 TKU with non zero min_util vs TKO with non zero min_util

Example
If TKU with min_util=0 =>10 seconds
TKU with min_util=non-zero =>75 seconds

It can be shown as

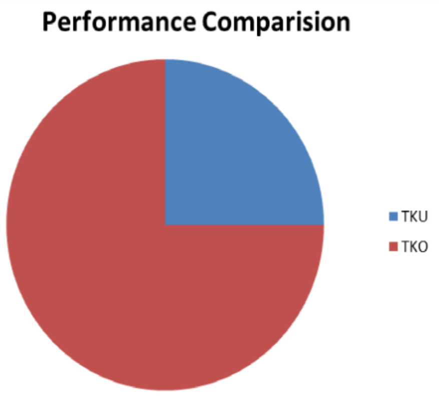


Figure 2. : Performance Comparison Graph

TKU's time consider as 100%(time which is grater is considered as 100%)
So here TKO is 25% better than TKU because 75 is 75% of 10 i.e. (100%)

Comparisons between Algorithms

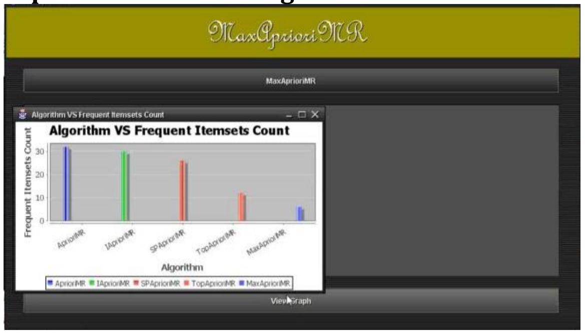


Fig 3: View graph

Table 1: Input and Output of the system

Condition	Input	Output
Load Dataset	Retail dataset	Dataset loaded successfully
AprioriMR	Transaction details	Hadoop Map reduce using Apriori
IApriori	Transaction details, chunk size	Frequent item sets
SPAprioriMR	Output of AprioriMR	Single items removed
TopAprioriMR	Transaction details with items, support, confidence	Frequent items with highest support values
MaxAprioriMR	Support, confidence, retail transaction	Maximum frequent item sets with highest support values
Min-Util	Transaction ids, product ids, product unit price, product utility	Min-util value to calculate Top-K high utility item sets

Conclusion

In this task, proposed new productive example mining calculations to work in enormous information. All the proposed models depend on the notable Apriori calculation and the MapReduce system. The proposed calculations are partitioned into three primary groups [7][9].
No pruning technique. Two calculations (AprioriMR and IAprioriMR) for mining any current example in information have been proposed.

1. Pruning the inquiry space by methods for hostile to monotone property. Two extra calculations (SPAprioriMR and TopAprioriMR) have been proposed with the point of finding any continuous example accessible in information.
2. Maximal continuous examples. A last calculation (MaxAprioriMR) has been additionally proposed for mining dense portrayals of regular examples.

References

[1]. Arthur.A.Shaw and N P Gopalan. Article: Frequent Pattern Mining of Trajectory Coordinates using Apriori Algorithm. *International Journal of Computer Applications* 22(9):1–7, May 2011

[2]. Abd-Elmegid, Laila & E. El-Sharkawi, Mohamed & El-Fangary, Laila & Helmy, Yehia. (2010). Vertical Mining of Frequent Patterns from Uncertain Data. Computer and

[3]. Information Science. 3. 10.5539/cis.v3n2p171.

[4]. Lakshminarayanan. "Frequent pattern mining on big data using Apriori algorithm." *International Journal of Advance Research, Ideas and Innovations in Technology* 3.5 (2018).

[5]. Nguyen, Dang & Nguyen, Loan & Vo, Bay & Pedrycz, Witold. (2016). Efficient Mining of Class Association Rules with the Itemset Constraint. Knowledge-Based Systems. 103. 73-88. 10.1016/j.knosys.2016.03.025.

[6]. J. M. Luna, J. R. Romero, C. Romero, and S. Ventura, "On the use of genetic programming for mining comprehensible rules in subgroup discovery," *IEEE Trans. Cybern.*, vol. 44, no. 12, pp. 2329–2341, Dec. 2014. [Online]. Available: http://dx.doi.org/10.1109/TCYB.2014.2306819

[7]. R. Agrawal, T. Imielinski, and A. Swami, "Database mining: A performance perspective," *IEEE Trans. Knowl. Data Eng.* vol. 5, no. 6, pp. 914–925, Dec. 1993.

[8]. J. Han, J. Pei, Y. Yin, and R. Mao, "Mining frequent patterns without candidate generation: A frequent-pattern tree approach," *Data Min. Know. Disc.*, vol. 8, no. 1, pp. 53–87, 2004.

[9]. S. Zhang, Z. Du, and J. T. L. Wang, "New techniques for mining frequent patterns in unordered trees," *IEEE Trans. Cybern.*, vol. 45, no. 6, pp. 1113–1125, Jun. 2015. [Online]. Available: http://dx.doi.org/10.1109/TCYB.2014.2345579

[10]. Nandini, G & Nynalasetti, Konadala Kameswara Rao. (2019). Utility Frequent Patterns Mining on Large Scale Data based on Apriori MapReduce Algorithm. International Journal of Research. 3. 19381-19387. 10.17762/ijrisat25815814.1903081-