Finding Contextual Term from Index of Documents using Apriori Algorithm

Taruna Kumari
M.Tech(IT)
YMCA University of Science and Technology
Faridabad,Haryana

Annu Saini
M.Tech(C.S)
Banasthali University
Rajasthan

ABSTRACT
Data mining is a multidisciplinary field that has mainly introduced to process large database and discover useful information that could help in decision making. World Wide Web is a vast resource of interlinked hypertext documents which are accessed via the Internet. Web mining, an extension of data mining, employs techniques of data mining to documents on the internet. Association rule mining is a major technique in the area of data mining. Association rule mining finds frequent itemsets from a set of transactional databases. Apriori algorithm is one of the earliest algorithm of association rule mining. Apriori employs an iterative approach known as levelwise search.. The Apriori Algorithm for mining frequent itemsets for boolean association rules can be applied to the index file of a search engine to find contextually related terms i.e the terms which occur in number of documents together. Ranking, a major component of a search engine ranks the web pages based on some criteria. The ranking algorithms consider the keywords entered in the query for rank purpose. Considering contextually related terms with query term can improve the ranking system. In this paper it is shown how apriori algorithm can be applied on index of web documents to find contextually related terms.

General Terms
Ranking ,apriori algorithm.

Keywords
Contextual term , data mining, association rule mining, apriori algorithm.

1. INTRODUCTION
Data Mining is a way of obtaining undetected patterns or facts from massive amount of data in a database. Data Mining is also known as knowledge discovery in databases (KDD). Data mining is more in demand because it helps to reduce cost and increases the revenues[1]. It is also a process for merging together statistical analysis, machine learning and databases to extract hidden rules and relationships[2]. The various applications of data mining are customer retention, market analysis, and production control and fraud detection. Data Mining is designed for different databases such as object-relational databases, relational databases, data warehouses and multimedia databases. Data mining methods can be categorized into classification, clustering, association rule mining, sequential pattern discovery, regression etc. Amongst these methods, association rule mining is very important which results in generating strong association rules.

Association rules was first proposed by R.Agrawal which aims at finding frequent itemsets from a set of transactional databases. The various algorithms in associations rule mining are Apriori, FP-Growth, Direct hashing and pruning (DHP), Apriori Tid. This apriori algorithm has many applications. Conventional query dependent page ranking algorithm[6] use term occurrence frequency and occurrence position of the given query terms for computing page rank. This rank value can be improved by considering synonyms or inferred terms for ranking that may contain the related information with respect to given query even without considering the actual keywords in the query. For example the query – "computer" does not contain the keywords laptop, desktop, keyboard mouse, CPU etc. But they are related to the query. These terms are called contextual terms. These contextual terms can be find by applying apriori algorithm on the index of documents.

2. APRIORI ALGORITHM
Apriori algorithm is the fundamental algorithm of association rule mining proposed by R.Agrawal and S.Srikant in 1994 [3]. Apriori Algorithm is an algorithm for frequent itemsets mining and association rule learning over transactional databases. Apriori is designed to operate on databases containing transactions. As is common in association rule mining, given a set of itemsets, the algorithm attempts to find subsets which are common to at least a minimum number C of the itemsets. Apriori uses a "bottom up" approach, where frequent subsets are extended one item at a time (a step known as candidate generation), and groups of candidates are tested against the data. The algorithm terminates when no further successful extensions are found.

Apriori employs an iterative approach known as levelwise search. In Apriori, (k+1) itemsets are generated from k-itemsets. First, scan the database for count of each candidate and compare candidate support count with minimum support count to generate set of frequent 1-itemsets. The set is denoted as L1. Then, L1 is used to find L2, set of frequent 2-itemsets, which is further used to find L3 and so on, until no more frequent k-itemsets can be found [4]. After finding set of frequent k-itemsets, it is easy to generate strong association rules. The process of finding each Lk requires the database to be scanned completely once. An important Apriori property is used to improve efficiency of levelwise generation of frequent itemsets which tells that any subset of frequent itemset must be frequent.

The quest to mine frequent patterns appears in many domains. The prototypical application[7] is market basket analysis, i.e., to mine the sets of items that are frequent bought together, at a supermarket by analyzing the customer shopping carts (the so-called "market baskets"). Once we mine the frequent sets, they allow us to extract association rules among the item sets, where we make some statement about how likely are two sets of items to co-occur or to conditionally occur. For example, in the weblog scenario frequent sets allow us to extract rules like, "Users who visit the sets of pages main, laptops and rebates also visit the pages shopping-cart and checkout", indicating, perhaps, that the special rebate offer is resulting in

more laptop sales. In the case of market baskets, we can find rules like, "Customers who buy Milk and Cereal also tend to buy Bananas", which may prompt a grocery store to co-locate bananas in the cereal aisle.

Similarly it can be used to find the terms which are co occurring in various documents of index file of a search engine and these co occurring terms are called context terms. For example if the keywords laptop, desktop, keyboard, mouse are co occurring with the term computer in some minimum number of documents then we can say these terms are contextually related to the term computer.

Advantages of Apriori algorithm[5]:

1) Apriori algorithm is easy to understand.

2) It is simple to implement.

3) It uses large itemset property.

4) It is easily parallelized.

Disadvantages of Apriori algorithm:

1) It requires many database scans.

2) It is less efficient and accurate.

3) It takes more time and consumes more memory.

2.1 Working of Apriori algorithm

Let D be the index of web documents. Support S[5] , is the occurrence frequency of a keyword in a document. Frequent k-term set is the set of k terms which co occurs in some minimum no. of documents.

1. Scan the index of web documents D, to get the support S of each 1-keyword (term) set, compare S with min_sup, and get a set of frequent 1- term sets, L1.

2. Use Lk-1 join Lk-1 to generate a set of candidate k-term set.

3. Scan the index database to get the support S of each candidate k-term set in the final set, compare S with min-sup, and get a set of frequent k- term set , Lk

4. If candidate set is empty then stop else go to step 2.

2.2 Apriori algorithm: example

Consider an example database consisting 6 documents shown below in Table 1.Suppose minimum support count required is 2.

Table 1. Example dataset of documents

Document	Keywords
Doc1	Computer, computer generation, desktop, laptop, CPU
Doc2	Computer, computer generation, desktop, laptop, super computer
Doc3	Computer, desktop, keyboard, mouse, printer, monitor
Doc4	Computer, mouse, printer, keyboard, laptop
Doc5	Computer, hp, printer, laptop, desktop
Doc6	Computer, laptop, desktop, dell

Step 1→ Generating 1-termset frequent pattern

Scan the database D for count of each candidate to generate C1

Term	Support count
Computer	6
Computer generation	2
Desktop	5
Laptop	5
CPU	1
Super computer	1
Keyboard	2
Mouse	2
Printer	3
Monitor	1
Hp	1
Dell	1

C1

Compare candidate support count with min_sup count to generate L1.

Term	Support count
Computer	6
Computer generation	2
Desktop	5
Laptop	5
Keyboard	2
Mouse	2
Printer	3

L1

Step 2→ Generating 2-termset frequent pattern

Generate candidate set C2 from L1.

Term
{computer, computer generation}
{computer, desktop}
{computer, laptop}
{computer, keyboard}
{computer, mouse}
{computer, printer}
{computer generation, desktop}
{computer generation, laptop}
{computer generation, keyboard}
{computer generation, mouse}
{computer generation, printer}
{desktop, laptop}
{desktop, keyboard}
{desktop, mouse}
{desktop, printer}
{laptop, keyboard}
{laptop, mouse}
{laptop, printer}
{keyboard, mouse}
{keyboard, printer}
{mouse, printer}

Scan the database D for count of each candidate C2.

Term	Support count
{computer, computer generation}	2
{computer, desktop}	5
{computer, laptop}	5
{computer, keyboard}	2
{computer, mouse}	2
{computer, printer}	3
{computer generation, desktop}	2
{computer generation, laptop}	2
{computer generation, keyboard}	0
{computer generation, mouse}	0
{computer generation, printer}	0
{desktop, laptop}	4
{desktop, keyboard}	1
{desktop, mouse}	1
{desktop, printer}	2
{laptop, keyboard}	1
{laptop, mouse}	1
{laptop, printer}	2
{keyboard, mouse}	2
{keyboard, printer}	2
{mouse, printer}	2

C2

Compare candidate support count with min_sup count and generate L2.

Term	Support count
{computer, computer generation}	2
{computer, desktop}	5
{computer, laptop}	5
{computer, keyboard}	2
{computer, mouse}	2
{computer, printer}	3
{computer generation, desktop}	2
{computer generation, laptop}	2
{desktop, laptop}	4
{desktop, printer}	2

{laptop, printer}	2
{keyboard, mouse}	2
{keyboard, printer}	2
{mouse, printer}	2

L2

Step 3→ Generating 3-termset frequent pattern

Generate candidate set C3 from L2.

Term
{computer, computer generation, desktop}
{computer, computer generation, laptop}
{computer, desktop, laptop}
{computer, desktop, printer}
{computer, laptop, printer}
{computer, keyboard, mouse}
{computer, keyboard, printer}
{computer, mouse, printer}

C3

Scan the database D for count of each candidate C3.

Term	Support count
{computer, computer generation, desktop}	2
{computer, computer generation, laptop}	
{computer, desktop, laptop}	4
{computer, desktop, printer}	2
{computer, laptop, printer}	2
{computer, keyboard, mouse}	2
{computer, keyboard, printer}	2
{computer, mouse, printer}	2

C3 , L3

After comparing support count of C3 termset with min_support we get L3 as shown above.

Step 4→ Generating 4-termset frequent pattern

Generate C4 candidates from L3 and scan the database D, for count of each candidate

Term	Support count
{computer, computer generation, desktop, laptop}	2
{computer, desktop, laptop, printer}	2
{computer, keyboard, mouse, printer}	2

L4

Now it is not possible to generate C5 from L4. In this way by using apriori algorithm we can create contextual table.

3. EXPERIMENTAL RESULTS

Apriori algorithm is implemented using java as front end tool and mysql as back end tool to store index of web documents. A snapshot of resultant contextual table is shown in figure below:

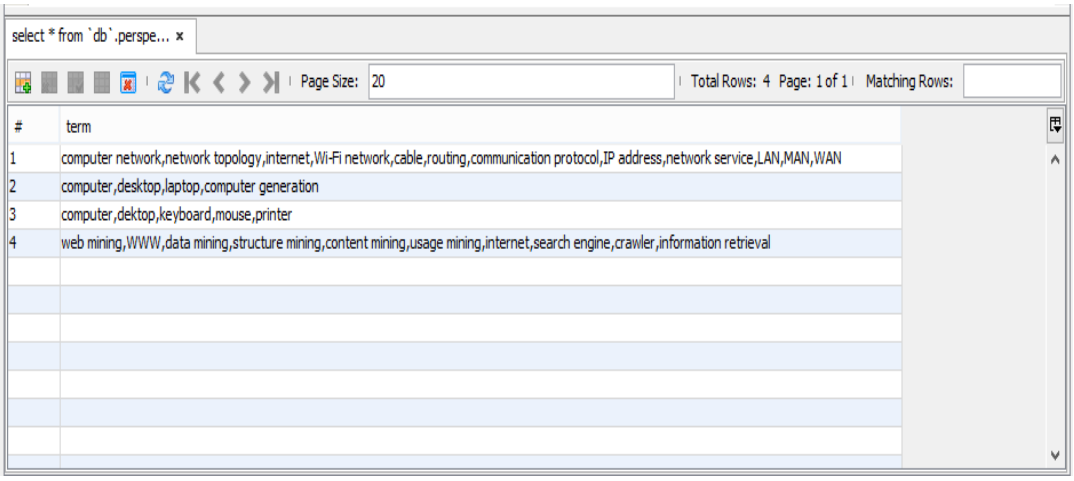

Fig 1.: Perspective table

4. CONCLUSION

Apriori algorithm is an innovative way to find association rules on large scale, allowing implication outcomes that consist of more than one item based on minimum support threshold.In this paper it is shown how this algorithm can be applied to find context terms from index of documents. These context terms can be used in page ranking algorithm to improve the rank factor of pages.

5. REFRENCES

[1] "Introduction to Data Mining and Knowledge Discovery" Third Edition by Two Crows Corporation ISBN: 1-892095-02-5.

[2] Agrawal, R.T., Imielinski, L., & Swami, A.N. (1993). , Mining association rules between sets of items in large databases. In International conference on. management of data (SIGMOD-93) (pp. 207–216).

[3] Mrs. R. Sumithra, Dr (Mrs). Sujni Paul, Using distributed apriori association rule and classical apriori mining algorithms for grid based knowledge discovery, 2010 Second International conference on Computing, Communication and Networking Technologies, 978-1-4244-6589-7/10/$26.00 ©2010 IEEE.

[4] Xue-gang hu, de-xing wang, xiao-ping liu, jun guo, Hao wang, the analysis on model of Association rules mining based on concept Lattice and apriori algorithm ,Proceedings of the Third International Conference on Machine Leaming and Cybernetics, Shanghai, 26-29 August 2004, 0-7603-8403-2/04/$20.00 @2W4 IEEE.

[5] Shilpi single,Arun Malik, Survey on various improved Apriori Algorithm, *International* Journal of Advanced Research in Computer and Communication Engineering Vol. 3, Issue 11, November 2014.

[6] Ashutosh Dixit, A.K.Sharma, Ashlesha Gupta, Perspective Terms based Mathematical Model for Page Ranking, ACM International Conference and workshop on Emerging trends in Technology, ICWET 2012,Feb 24-25,2012 TCET Kandawali(E), Mumbai.

[7] http://www2.cs.uregina.ca/~dbd/cs831/notes/itemsets/itemset_apriori.html