International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Impact Factor (2012): 3.358

Comparative Study of Web Content Mining Techniques for HTML and XML Contents

Rupinder Kaur[1], Kamaljit Kaur[2]

[1]SGGSW University, Department of Computer Science and Engineering,
Fatehgarh Sahib, Punjab, India 140406

[2]Assistant Professor, Department of Computer Science and Engineering, SGGSW University,
Fatehgarh Sahib, Punjab, India 140406

Abstract: ***World Wide Web is the rapidly grown source of information. Data on the web is available in many forms which are – structured data, unstructured data and semi- structured data. Also it is growing on daily basis. It is becoming difficult to the user to get the relevant data from the web. Data Mining is the subject of computer science which is used to mine useful information from very large amount of data. Web mining is the application of data mining, which implements various techniques of data mining to get the relevant information from the web. Web developers have now started to develop Web pages on emerging Web Technologies like XML, Flash etc. XML was designed to describe data and to focus on what the data is. XML also plays the role of a meta- language and allows authors to create customized markup language for different types of documents, making it a standard data format for online data exchange. To date, famous algorithms like Apriori and FP- Growth algorithms are used to fetch the web data for XML contents and for HTML contents numerous techniques have been proposed. In this paper, a hybrid approach is used to fetch HTML as well as XML contents from a web page. In the hybrid approach, Apriori algorithm is used to remove the unimportant information from the contents and Decision tree is used to fetch the contents from a web page. This hybrid approach is compared with the previous technique implementing FP-Growth algorithm for HTML and XML contents. At the end, results are shown using graphs.***

Keywords: Web Mining, XML, Apriori, Decision Tree, FP- Growth algorithm.

1. Introduction

Web is the largest source of information. As the number of documents grows, searching for information is turning into a cumbersome and time consuming operation. Data Mining is the field of computer science which is used to extract information from the large amount of data. Web Mining is the application of data mining which is used to generate patterns from the web. Patterns must be such that they are easily understandable, useful and novel. Various techniques of data mining are used to extract the information from the web. Not only data mining but also other tools from fields of artificial intelligence, machine learning, natural language processing can also be used efficiently to fetch web data. It is very wide area of research for the researchers because of the growing use of the web. Web Mining on the basis of type of data to be explore can be divided into three main categories-

1.1 Web Usage Mining

It is the mining of the user preferences while user is navigating through the websites. This is done by applying the mining process on the log files repository. [6] By web usage mining, commercial websites take advantage of knowing the usage pattern of customer, their behaviour and frequency of their visits.

1.2 Web Structure Mining

The Web page structure consists of a Web page as a node and hyperlinks as edges connecting to other pages. [6] In other words, it works in the form of graphs. It focuses on the connectivity of the web site to other sites that are called as hyperlinks.

1.3 Web content Mining

Web content mining is the process of mining of contents of a web page. Contents of a web page may include free text, images, audio, video, animations and semi-structured records in the form of html and xml contents like hyperlinks, imagelinks etc. which are either embedded in the web page or having links to other pages.

Most of the data on the web is in unstructured form i.e. in the form of free text, images, audio, video and semi-structured form like HTML and XML etc. Since HTML has many limitations like limited tags, not case sensitive and designed to display data only, Web developers has started to develop Web pages on emerging Web Technologies like XML, Flash etc. [4] XML was designed to describe data and to focus on what the data is. XML also plays the role of a meta-language and allows document authors to create customized mark-up language for limitless different types of documents, making it a standard data format for online data exchange. This growing use has raised need for better tools and techniques to perform mining on XML too. In the proposed paper, a hybrid approach is used to fetch XML contents from XML file and HTML contents too.

Rest of the paper is organised as follows. Section 2 presents the previous technique related to the topic. Section 3 covers the hybrid approach. Section 4 gives results of the work and comparison. Section 5 concludes the paper.

2. Previous Techniques

D. Jayalatchumy et al. [12] have compared various algorithms that have been used for HTML contents like

images, audio, video etc. Also pros and cons of each algorithm is given.

Nassem et al. [5] constructs an FP-tree structure and mines frequent patterns by traversing the constructed FP-tree. In addition, some features have been suggested that need to be added into XQuery in order to make the implementation of the First Frequent Pattern growth more efficient. In future work of paper it was planned to implement other standard data mining algorithms which can be expressed in XQuery to improve the performance of the results. This method is advantageous because, it doesn't generate any candidate items. It is disadvantageous because, it suffers from the issues of special and temporal locality issues.

FP-growth algorithm constructs an FP-tree structure and mines frequent patterns by traversing the constructed FP-tree. The FP-tree structure is an extended prefix-tree structure involving crucial condensed information of frequent patterns.

(a) FP-tree structure
The FP-tree structure has sufficient information to mine complete frequent patterns. It consists of a prefix tree of frequent 1-itemset and a frequent-item header table. Each node in the prefix-tree has three fields: *item-name*, *count*, and *node-link*. *item-name* is the name of the item. *count* is the number of transactions that consist of the frequent 1-items on the path from root to this node. *node-link* is the link to the next same item name node in the FP-tree. Each entry in the frequent-item header table has two fields: *item-name* and *head of node-link*. *head of node-link* is the link to the first same item-name node in the prefix-tree.

(b) Construction of FP-tree
FP-growth has to scan the TDB twice to construct an FP-tree. The first scan of TDB retrieves a set of frequent items from the TDB. Then, the retrieved frequent items are ordered by descending order of their supports. The ordered list is called an F-list. In the second scan, a tree T whose root node R labeled with "null" is created. Then, the following steps are applied to every transaction in the TDB. Here, let a transaction represent [p\P] where p is the first item of the transaction and P is the remaining items. In each transaction, infrequent items are discarded. Then, only the frequent items are sorted by the same order of F-list.

Call insert_tree (p\P, R) to construct an FP-tree.

The function insert_tree (p\P, R) appends a transaction [p\P] to the root node R of the tree T. Pseudo code of the function insert_tree (p\P, R). This FP-tree is constructed from Table 2 with min-sup=2. In Figure 1, every node is represented by

(item-name: count).
Links to next same item-name node are represented by dotted arrows.
function insert_tree (p|P, R) {
let N be a direct child node of R, such that N's
item-name = p's item-name.
if (R has a direct child node N) {
increment N's count by 1. }
else{ create a new node M linked under the R . set M 's item-name equal to p . set M 's count equal to 1. }
call insert_tree (P, N). }

Table 1: Sample data

Transaction ID	Items	Frequent Items
100	A,B,C	A,B,E
200	B,D	B,D
300	B,C	B,C
400	A,B,D	A,B,D
500	B,C	B,C
600	A,B,C,E	A,B,C,E
700	A,B,C	A,B,C

B	7
A	4
C	4
D	2
E	2

(c) FP-growth
FP-growth mines frequent patterns from an FP-tree. To generate complete frequent patterns, FP-growth traverses all the node-links from "head of node-links" in the FP-tree's header table. For node C, FP-growth mines a frequent pattern (C:4) by traversing C's node-links through node (C:2) to node (C:2). Then, it extracts C's prefix paths; <B:7, A:4> and <B:7>. To study which items appear together with C, the transformed path <B:2,A:2> is extracted from <B:7,A:4> because the support value of C is 2. Similarly, we have <B:2>, the set of these paths {(B:2,A:2),(B:2)} is called C 's conditional pattern base. FP-growth then constructs C's conditional FP-tree containing only the paths in C's conditional pattern base. As only B is an item occurring more than *min_sup* appearing in C 's conditional pattern base, C 's conditional FP-tree leads to only one branch (B:7). Hence, only one frequent pattern (BC:4) is mined. The final frequent patterns including item C are (C:4) and (BC:4). [5]

Mishra et al. [9] developed the improved FP-tree. The FP-tree structure has sufficient information to mine complete frequent patterns. It consists of a prefix tree of frequent 1-itemset and a frequent-item header table. FP-growth has to scan the *TDB* twice to construct an FP-tree. The first scan of *TDB* retrieves a set of frequent items from the *TDB*. Then, the retrieved frequent items are ordered by descending order of their supports. The ordered list is called an F-list. In the second scan, a tree *T* whose root node *R* labelled with "null" is created. It will increases the mining efficiency and also takes less memory.

3. Hybrid Technique

In this hybrid approach, two well-known data mining algorithms are used. Our approach is applied in two steps-
Step 1: In the first step Decision tree is implemented to fetch the data from the web.
Step 2: In the second step well-known Apriori algorithm is used to remove redundancy and unimportant data. These algorithms are explained as under:-

Step 1: Decision Tree

A decision tree is used for decision making purpose. Decision tree has root and branch node. From the root node, users split each node recursively based on decision tree learning algorithm. The final result of decision tree consists of branches and each branch represents a possible scenario of decision and its consequences. [13]

Pseudocode of a Decision Tree Induction Algorithm [16]

Input:

- Data partition, D, which is a set of training tuples and their associated class labels.
- attribute_list, the set of candidate attributes.
- Attribute selection method, a procedure to determine the splitting criterion that best partitions the data tuples into individual classes. This criterion includes a splitting_attribute and either a splitting point or splitting subset.

Output:

A Decision Tree

Method:

1. create a node N;
2. if tuples in D are all of the same class, C then return N as leaf node labeled with class C;
3. if attribute_list is empty then return N as leaf node with labeled with majority class in D;|| majority voting
4. apply attribute_selection_method (D, attribute_list) to find the best splitting_criterion;
5. label node N with splitting_criterion;
6. if splitting_attribute is discrete-valued and multiway splits allowed then // no restricted to binary trees
7. attribute_list = splitting attribute; // remove splitting attribute
8. for each outcome j of splitting criterion // partition the tuples and grow subtrees for each partition
   let Dj be the set of data tuples in D satisfying outcome j; // a partition
   If
   Dj is empty then attach a leaf labeled with the majority class in D to node N;
   Else
   attach the node returned by Generate decision tree (Dj, attribute list) to node N;
   end for
9. return N;

Step 2: Apriori Algorithm

As the name implies, this algorithm uses prior knowledge about frequent itemset properties. It employs an iterative approach where k-itemsets are used to explore (k +1)-itemsets. To improve the efficiency of the generation of frequent itemsets, it uses an important property called the Apriori property which says that all nonempty subsets of a frequent itemset must also be frequent. The Apriori algorithm first computes the frequent 1- itemsets, L1. To find frequent 2-itemsets L2, a set of candidate 2-itemsets, C2, is generated by joining L1 with itself. [17]

Pseudocode of Apriori algorithm [17]

Procedure Apriori (T, minSupport)// T is the database and minSupport is the minimum Support.

1. L1= {frequent items};
2. for (k=2; L_{k-1}!=Ø; k++) {
3. C_k = candidate generated from L_{k-1} // that is product of L_{k-1} * L_{k-1} and eliminating any k-1 size itemset that is frequent
4. for each transaction t in database do {
   # increment the count of all candidates in C_k that are contained in t
5. L_k = candidates in C_k with minSupport
   } // end for each
   } // end for
6. Return $U_k L_k$;

}

In this approach, Apriori algorithm is used to remove the unimportant data from the web pages. In the HTML contents, function of Apriori algorithm is to remove the redundancy of hyperlinks from any URL and in the XML contents, its working is to remove the unimportant elements like tags and meta-tags from the XML file.

4. Results and Comparison

As we have implemented both the techniques which are FP-Growth tree algorithm and Hybrid approach, results which came out are as follows. All the results are dynamic in nature i.e. they vary with the internet speed and speed of the system which is used to take the results. Sample results are taken from the top popular URLs like twitter, eBay, facebook, homeshop18 etc. and out of them graphs have been drawn. Parameters which are calculated for HTML contents are execution time, precision, recall, f- measure, g-measure and parameters for XML contents are execution time. These results are compared for both techniques with the help of graphs.

4.1 Execution Time

It is the time difference when the project first starts its execution and time when hyperlinks and imagelinks are fetched. It is calculated in milliseconds.

Table 2: Execution Time

ID	*URLs*	*Time of Existing Technique (ms)*	*Time of Hybrid Approach (ms)*
1	https://twitter.com/	14,000	9664
2	http://www.ebay.in/	34,000	7295
3	http://www.homeshop18.com/	15,000	7690
4	http://tanishq.co.in/Home	38,000	9253
5	http://rontalk.com/	55,000	8651
6	http://www.jabong.com/	10,000	9916
7	https://www.facebook.com/	12,000	7908
8	http://www.123greetings.com/	12,000	8342
9	http://www.naturewallpaper.net/	46,000	7007
10	http://www.zigwheels.com/	19,000	8694

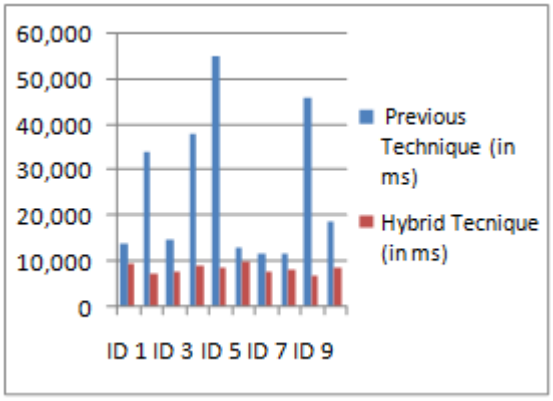

Figure 1: Graph showing comparison of Execution Time for Previous and Hybrid Technique

4.2 Precision

It is defined as the fraction of retrieved instances that are relevant. [18]

Table 3: Precision

ID	*URLs*	*Precision of Previous Technique*	*Precision of Hybrid Approach*
1	https://twitter.com/	0.0617	0.0847
2	http://www.ebay.in/	0.0813	0.205
3	http://www.homeshop18.com/	0.3714	0.5652
4	http://tanishq.co.in/Home	0.1182	0.1466
5	http://rontalk.com/	0.15	0.1941
6	http://www.jabong.com/	0.1917	0.2256
7	https://www.facebook.com/	0.111	0.1923
8	http://www.123greetings.com/	0.0585	0.0915
9	http://www.naturewallpaper.net	0.3333	0.3548
10	http://www.zigwheels.com/	0.2950	0.4043

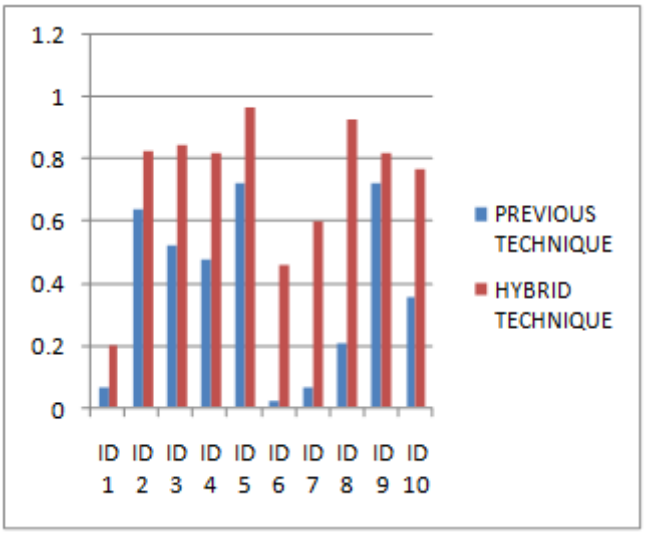

Figure 2: Graph showing comparison of Precision for Previous and Hybrid Technique

4.3 Recall

It is the fraction of relevant instances that are retrieved. [18]

Table 4: Recall

ID	*URLs*	*Recall of Previous Technique*	*Recall of Hybrid Approach*
1	https://twitter.com/	0.0666	0.2
2	http://www.ebay.in/	0.64	0.825
3	http://www.homeshop18.com/	0.5217	0.8461
4	http://tanishq.co.in/Home	0.4761	0.8181
5	http://rontalk.com/	0.7209	0.9696
6	http://www.jabong.com/	0.0212	0.4594
7	https://www.facebook.com/	0.0666	0.6
8	http://www.123greetings.com/	0.2083	0.9285
9	http://www.naturewallpaper.net/	0.7209	0.8181
10	http://www.zigwheels.com/	0.3563	0.7702

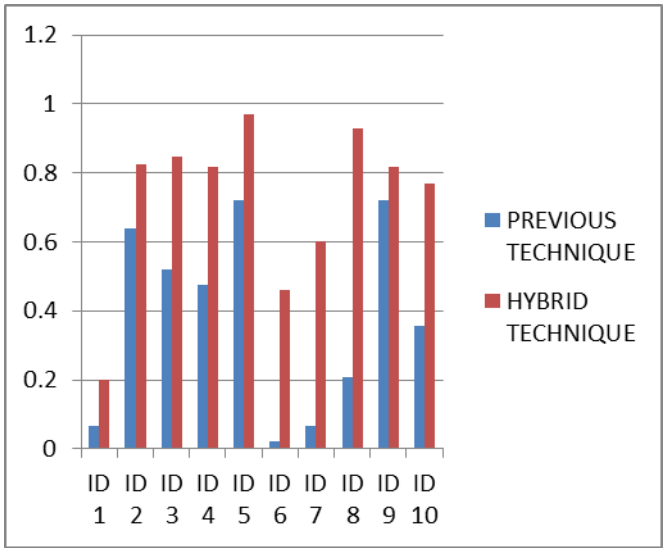

Figure 3 Graph showing comparison of Recall for Previous and Hybrid Technique

4.4 F-Measure

It is defined as the harmonic mean of precision and recall.
F-Measure= (2*Precision*Recall)/ (Precision + Recall)

Table 5: F-Measure

ID	*URLs*	*F-Measure of Previous Technique*	*F-Measure of Hybrid Approach*
1	https://twitter.com/	0.0641	0.1190
2	http://www.ebay.in/	0.1442	0.3285
3	http://www.homeshop18.com/	0.4339	0.6777
4	http://tanishq.co.in/Home	0.1894	0.495
5	http://rontalk.com/	0.2483	0.3234
6	http://www.jabong.com/	0.0383	0.3026
7	https://www.facebook.com/	0.0833	0.2912
8	http://www.123greetings.com/	0.0914	0.1665
9	http://www.naturewallpaper.net/	0.4588	0.495
10	http://www.zigwheels.com/	0.3227	0.5303

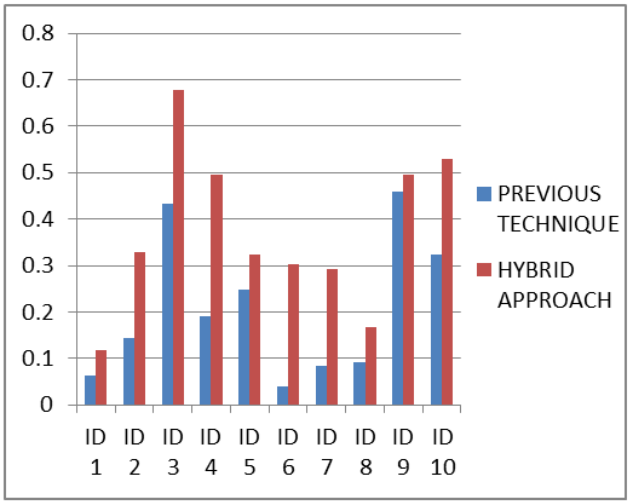

Figure 4: Graph showing comparison of F-Measure for Previous and Hybrid Technique

4.5 G-Measure

It is defined as the geometric mean of precision and recall.

G-Measure=√ (Precision*Recall)

Table 6: G-Measure

ID	*URLs*	*G-Measure of Previous Technique*	*G-Measure of Hybrid Approach*
1	https://twitter.com/	0.0641	0.1301
2	http://www.ebay.in/	0.2281	0.4113
3	http://www.homeshop18.com/	0.4402	0.6915
4	http://tanishq.co.in/Home	0.2373	0.5388
5	http://rontalk.com/	0.3288	0.4338
6	http://www.jabong.com/	0.0638	0.3219
7	https://www.facebook.com/	0.08606	0.3396
8	http://www.123greetings.com/	0.1104	0.2914
9	http://www.naturewallpaper.net/	0.4902	0.5388
10	http://www.zigwheels.com/	0.3242	0.5580

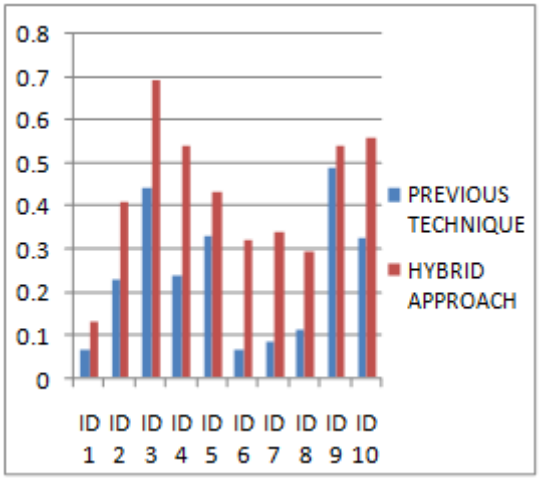

Figure 5: Graph showing comparison of G-Measure for Previous and Hybrid Technique

4.6 XML Contents

XML contents are fetched from XML files and their execution time is computed in milliseconds.

Table 7: XML Files

ID	Previous Technique (ms)	Hybrid Approach (ms)
1	3680	3000
2	3623	3000

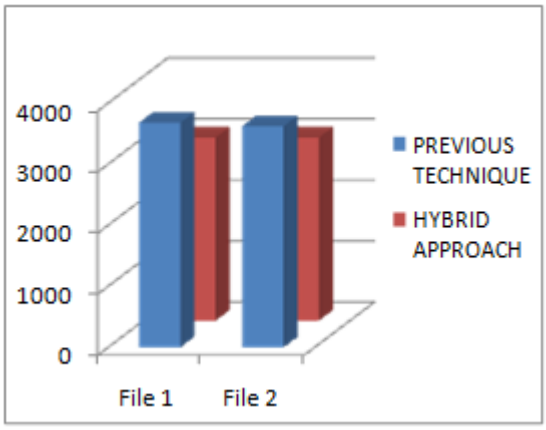

Figure 6: Graph showing comparison of Execution Time for Previous and Hybrid Approach for XML Contents

5. Conclusion

In this paper, a comparative study of two techniques is made. First technique is the well-known FP-Tree algorithm implemented to fetch both the HTML and XML contents. Second approach is the hybrid approach which is the combination of Apriori and Decision Tree algorithms. From the Result and Comparison section, it is made clear that hybrid approach is better than the previous one. Execution time for fetching the contents in both the HTML and XML contents is less in the hybrid approach whereas relevancy which is measured in the form of precision and recall is better with the hybrid approach. Other parameters like F-measure and G-measure which depends on precision and recall are then obviously better than the previous technique.

6. Future Scope

In the future, Hybrid approach can be improved using other data mining algorithms. Also numbers of modifications are available for both Apriori and Decision Tree which can be implemented in this approach of fetching web data. This approach doesn't work well for low speed internet; work can be done to improve this limitation of the approach.

References

[1] O. Etzioni, "The world wide Web: Quagmire or gold mine." Communications of the ACM, Vol. 39 No. 11, pp. 65-68, Nov. 1996.

[2] R. Kosala, H. Blockeel "Web mining research: A survey," ACM SIGKDD Explorations, Vol. 2 No. 1, pp. 1-15, June 2000.

[3] Qin Ding and Gnanasekaran Sundarraj, "Association Rule Mining from XML Data", Conference on Data Mining | DMIN'06|.

[4] Krishna Murthy. A, Suresha, "XML: URL Data Set Creation for Future Web Mining Research Avenues",

International Journal of Computer Applications (0975 – 8887), 2011.

[5] Mohammad. Nassem, Prof. Sirish Mohan Dubey, "First Frequent Pattern-tree based XML pattern fragment growth method for Web Contents", International Journal of Advanced Research in Computer Science and Software Engineering, Volume 2, Issue 1, January 2012.

[6] Ganesh Dhar, Govind Murari Upadhyay, "Web Mining: Concepts and Decision Making Aid". Volume 2, Issue 7, July 2012.

[7] Darshna Navadiya, Roshni Patel, "Web Content Mining Techniques- A Comprehensive Survey", IJERT, Vol. 1 Issue 10, December- 2012.

[8] Amit Kumar Mishra, Hitesh Gupta, "A Recent Review on XML data mining and FFP", International Journal of Engineering Research, Volume No.2, Issue No.1, pp : 07-12.

[9] Amit Kumar Mishra, Hitesh Gupta, "A Novel FP-Tree Algorithm for Large XML Data Mining", International Journal of Engineering Sciences and Research Technology, [Mishra, 2(3): March, 2013].

[10] Abdelhakim Herrouz, Chabane Khentout Mahieddine Djoudi, "Overview of Web Content Mining Tools", The International Journal of Engineering And Science, Volume 2, Issue 6, pp. 106-110, June 2013.

[11] Buhwan Jeong, Daewon Lee, Jaewook Lee, and Hyunbo Cho, "Towards XML Mining: The Role of Kernel Methods".

[12] D. Jayalatchumy, Dr. P.Thambidurai, "Web Mining Research Issues and Futur Directions – A Survey", IOSR Journal of ComputerEngineering (IOSR-JCE), Volume 14, Issue 3 (Sep. - Oct. 2013), pp. 20-27.

[13] K. Nethra, J. Anitha G. Thilagavathi, "Web Content Extraction Using Hybrid Approach", ICTACT Journal On Soft Computing, JAN 2014, VOLUME: 04, ISSUE: 02.

[14] http://en.wikipedia.org/wiki/Web_content.

[15] http://webdesign.about.com/od/content/qt/what-is-web-content.htm.

[16] http://www.tutorialspoint.com/data_mining/dm_dti.htm.

[17] http://en.wikipedia.org/wiki/Apriori_algorithm.

[18] http://en.wikipedia.org/wiki/Precision_and_recall.

Author Profile

Rupinder Kaur is currently completing Master of Technology from Sri Guru Granth Sahib World University, Fatehgarh Sahib. She has received the Bachelor of Technology degree in Information Technology from Baba Banda Singh Bahadur Engineering Technology, Fatehgarh Sahib in the year of 2008. She is currently writing thesis on "Web Mining".

Kamaljit Kaur is currently pursuing PhD in Data and Web Mining. She obtained her Master of Technology and Bachelor of Technology degrees in CSE. She has over 10 years of experience. Currently she is working as Assistant Professor at SGGSW University, Fatehgarh Sahib and guiding various M.Tech thesis in area of database security and data mining.