Web usage knowledge based Web-Page Recommendation system

Ms. Sonule, Prashika Abasaheb[1] Prof. Tanveer I. Bagban[2]

[1] *Student, Department of Computer Science and Engineering, D.K.T.E. Society's Textile and Engineering Institute, Shivaji University, Maharashtra, India.*
[2] *Associate Professor, Department of Information Technology, D.K.T.E. Society's Textile and Engineering Institute, Shivaji University, Maharashtra, India*

ABSTRACT

Due to the vast release of World Wide Web (WWW), which serves as widely distributed, global information service for every type of information such as advertisements, news, education and many other information services, regularly, Web users struggle to find pages. As the amount of information on the Web is increasing rapidly, it has become more difficult to discover applicable and beneficial information on behalf of Web users. To overcome these problems, Web usage knowledge can be obtained by Web usage mining, to learn usage patterns in order to better serve Web page recommendations. This paper, presents system for Web-page recommendation system, based on Web usage knowledge and its performance analysis, i.e. experimental evaluation conducted in terms of precision and satisfaction.

Keywords: *Web-page Recommendation System, Web mining, Web usage mining*.

1. INTRODUCTION

The irregular progress and rise in data on (www), i.e. World Wide Web, with the improvement of inventive electronic devices, has made information of Web progressively vital in everybody's life. As internet technologies are growing rapidly, so web has become huge storage of information and grow with high and quick rate of change, therefore, websites are also familiarized quickly with new inventions.

Web page recommendations are one of those newer inventions related to websites. Web page recommendations predict and recommend pages, in order to help user. Regularly, Web users finds difficulty in searching required page. Even they find it uneasy if pages in website are not well designed. User has to search every page wasting time in searching pages. These problems are faced by web users in getting related page they want which is very time-consuming. So, there is a need of Web-page recommendation systems.

The reason behindhand above problems can be huge amount of volatile growth of information, which is noisy and even irrelevant. Taking into considerations, the above mentioned problems, it seems that there was a need of cleaning and construct or structure that irrelevant data. This construction of data and reformatting it before processing is called as data preprocessing. This is done before applying any Web mining techniques.

Web usage mining helps to solve above cited problems. An application from data mining techniques i.e., Web usage mining, notices and identifies usage knowledge of website [1]. Web Usage Mining is a part of web mining which contracts and compresses with the extraction of knowledge from log files [1, 2]. This is implemented in this work and will be explained further in details.

Thus, this paper, presents system for Web-page recommendation based on Web usage knowledge and its performance analysis, i.e. experimental evaluation conducted in terms of precision and satisfaction. Here paper is prearranged as below: Section 2 illustrates System architecture of Web Usage Knowledge based Web page recommendation system. Section 3, elaborates performance analysis, i.e. experimental evaluation. Section 4, gives the conclusion. And finally, section 5 consists of references.

2. SYSTEM ARCHITECTURE

Figure 4.1 shows the architecture of the system for Web-page recommendation. As shown in figure 4.1., below, each module is explained as below:

- DATA PREPROCESSING:

It collects web pages accessed by users to construct data in order to make it available in extracting patterns. Several data preprocessing techniques like data cleaning have been used in improving other phases of web usage mining like Pattern discovery and Pattern analysis [3, 4]. Data preprocessing required for our work is explained in details in the next section.

- WEB USAGE KNOWLEDGE MODEL:

Web usage knowledge can be discovered from Web log files using a well-known, technique Web usage mining technique. We employ an advanced Web usage mining technique, namely Apriori, to discover the Web usage knowledge [5].

- WEB PAGE RECOMMENDATION BASED ON WEB USAGE KNOWLEDGE:

Efficiently provide better web-page recommendation through semantic enrichment. The aim of a recommender system is to determine which Web pages are more likely to be accessed by the user in the future.

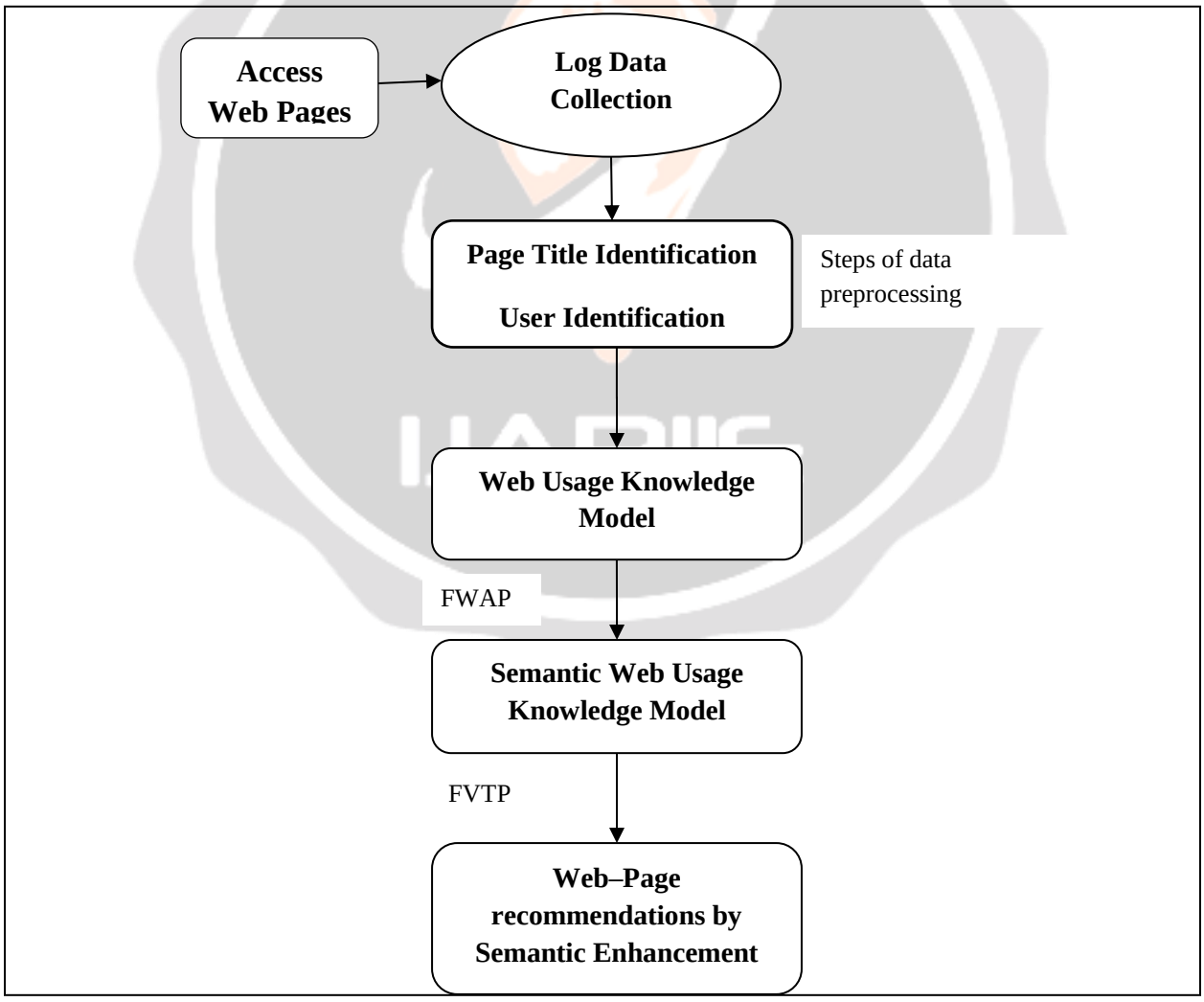

Figure 1: Architecture of the system for Web-page recommendation

DEFINITION:

For data preprocessing there is a need of collecting data which is nothing but, 'preprocessed data collection' in this work. No specific algorithm or formula is used for data preprocessing. For this work data will be in the form of log data. This log data is implemented as in 'standard log data' definition given below:

STANDARD LOG DATA DEFINITION: Let P be a set of 'literals', called pages or clicks, and U be a set of 'users'. A log is a set of triples $\{< u_1,p_1,t_1 >, \ldots\ldots\ldots < u_n,p_n,t_n > \}$ where $u_i \epsilon U$, $p_i \epsilon P$, and $t_i \epsilon$ timestamp.

In this work, to get log data according this definition, special website www.dktes.com is used such that log data can be created which will be in preprocessed form to reduce further complexity and such that semantic enhancement can be given to web page recommendations. To get log data as in definition, i.e. user identification; set of pages accessed or hits, clicks i.e. path completion and timestamp i.e. session identification are implemented as further.

Changes are made at <a href…> tag of every page of website; where action is taken to where to go, when user hits that page. Code is added at <a href …> tag, for example as given below: <a href="../SaveLink.jsp?val= overview.html"...>. Here, SaveLink.jsp is a 'JavaServer Pages script file' for a code, program also called as a software agent, to create a web log file, which is a web log data as defined in above standard definition of web log data. Where 'val' is a string attribute defined in a SaveLink.jsp program; overview.html is stored in val. Here, overview.html is taken as an example. Same code is added at <a href …> tag of each and every page of a website, such that whenever user hits that page, user will be identified and will be added to the log data in a log file. Here log.txt is taken as a log file which is created at E:\\log.txt. Work is followed as steps given below fig 1.1 Log data creation algorithm.

Input: Accessed Web-page links.
Output: log.txt (Page accessed, user id, time).
Process:

1. Begin
2. For each accessed link.
    1. Initialize Page title p_i, where $p_i \epsilon P$, $P = \{p_i,\ldots..p_n\}$.
    2. Get User Id u_i, $u_i \epsilon U$, $U = \{u_i,\ldots\ldots u_n\}$.
    3. Get time_stamp t_i.
3. Create log file ("E:\\log.txt").
4. Add page title, user id and timestamp to "E:\\log.txt".
5. Close file.
6. Go to next Web-page link accessed ;
7. End for
8. End

1.1 Log data creation algorithm

USER IDENTIFICATION:

In website, registration is given for authentication purpose; such that only registered user can access this website. This registration information is stored in table named as 'registration' in 'wpr' schema. This information is email, first name, last name, username and password. Using this username and password user is authenticated always and also helps for user identification.

When programming the site, it is very helpful to be able to associate some data with each visitor. For this purpose, "session" can be used in JSP. A session is an object associated with a visitor. Data can be put in the session and retrieved from it, much like a hash table. A different set of data is kept for each visitor.

For user identification separate code on each page is included when user visits that page. Changes are made at <a href…> tag; where action is taken to where to go, when user hits that page. Code is added at <a href …> tag, for example as given below: <a href="../SaveLink.jsp?val= overview.html"...>. Here, SaveLink.jsp is a 'JavaServer Pages script file' for a code, i.e., implementation of a log data creation algorithm given above, program also called as a software agent, to create a web log file, which is a web log data as defined in above standard definition of web log data. Where 'val' is a string attribute defined in a SaveLink.jsp program; overview.html is stored in 'val'. Here, overview.html is taken as an example.

Same code is added at <a href …> tag of each and every page of a website, such that whenever user hits that page user will be identified. User registration and login details are stored in separate table named as 'registration' in

'wpr' schema. Due to this, User id can be extracted from table named 'registration' using session.getAttribute (userid).toString ().

PATH COMPLETION:

This will include page or set of pages accessed or visited by user. Same Procedure is done as user identification. Changes are made at <a href…> tag; where action is taken to where to go, when user hits that page. Code is added at <a href …> tag, for example as given below: <a href="../SaveLink.jsp?val= overview.html"...>. Here, SaveLink.jsp is a 'JavaServer Pages script file' for a code, i.e., implementation of a log data creation algorithm given above, program also called as a software agent, to create a web log file, which is a web log data as defined in above standard definition of web log data. Where 'val' is a string attribute defined in a SaveLink.jsp program; overview.html is stored in 'val'; 'val' is key attribute which stores the path or name of that file. Here, overview.html is taken as an example. Same code is added at <a href …> tag of each and every page of a website, such that whenever user hits that page, path or name of that page will be identified and will be added to the log data in a log file. Here log.txt is taken as a log file which is created at E:\\log.txt.

SESSION IDENTIFICATION:

A sequence or series of web pages, that user browses in a single access is called as session identification. This displays the timestamp when user actually hits that particular page. This indicates only web pages that user browses in a single access. For example it displays the date and time as dd/mm/yy and time in hr: min. Session Identification is used just for our knowledge i.e when last user has accessed which page.

To get current time standard class Calender is used in SaveLink.jsp to access time of when that page was accessed as shown in program above; by creating object for a Calendar class Calendar c=Calendar.getInstance().

Same Procedure is implemented as user identification and path completion. Changes are made at <a href…> tag; where action is taken to where to go, when user hits that page. Code is added at <a href …> tag, for example as given below: <a href="../SaveLink.jsp?val= overview.html"...>. Here, SaveLink.jsp is a 'JavaServer Pages script file' for a code, program also called as a software agent, to create a web log file, which is a web log data as defined in above standard definition of web log data. Where 'val' is a string attribute defined in a SaveLink.jsp program; overview.html is stored in val; 'val' is key attribute which stores the path or name of that file. Here, overview.html is taken as an example. Same code is added at <a href …> tag of each and every page of a website, such that whenever user hits that page, path or name of that page will be identified and will be added to the log data in a log file. Here log.txt is taken as a log file which is created at E:\\log.txt.

In order to make better Web-page recommendations, semantic Web usage knowledge is needed, which can be obtained by integrating the domain knowledge model with Web usage knowledge model that can be discovered from Web log files using a Web usage mining technique. So the domain knowledge model and Web usage knowledge model will be combined to get semantic knowledge of Web usage of a website [5]. Therefore, it is necessary to develop Web usage knowledge model.

WORK DONE IMPLEMENTATION:

Web usage knowledge Model:

Web usage knowledge can be discovered from Web log files using a Web usage mining technique. Web usage mining technique, namely Apriori is employed here; to discover the Web usage knowledge, which will be in the form of Frequent Web Access Patterns (FWAP), i.e. patterns of frequently visited Web-pages.

Before applying any web usage mining technique, it is necessary to discover the set of web pages accessed sequence by each user. Discovery of Web Usage Knowledge will be as given in below definition 1. Figure 1, below shows the discovery of Web Usage Knowledge.

DEFINITION 1 Frequent Web Access Patterns (FWAP):

Let D be a set of Web-pages Access Sequence (WAS), T be a set of different domain terms in the titles of the Web-pages. Let P = $\{P_1, P_2, P_3, \ldots, P_n\}$ be a set of FWAP, where each pattern P_i, i=[1, 2, ….,n] contains a sequence of Web-pages, n is the number of the patterns, and $P_i = d_{i1}, d_{i2}, \ldots d_{im}$, where $d_{ik} \in D$, k = [1….m], m is the number of Web-pages in the pattern.

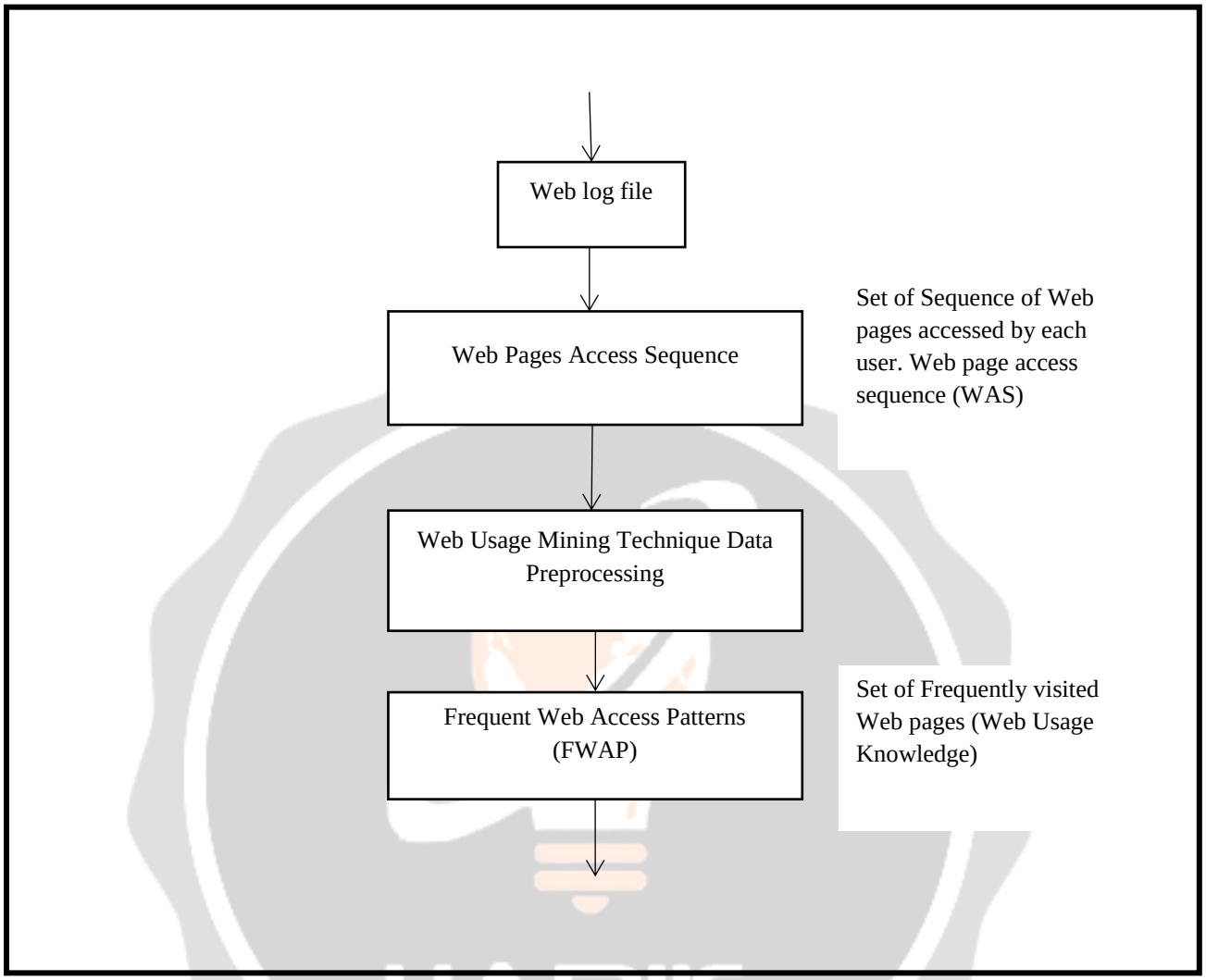

Figure 2: Discovery of Web Usage Knowledge

Web-pages Access Sequence (WAS):

As shown in figure 1, above, before applying any web usage mining technique, it is necessary to discover the set of sequential web pages accessed by each user. This set will contain Web pages accessed, sequentially one after another by each user and usernames of who accesses those pages. These usernames and pages accessed by each particular user will be extracted from log file 'log.txt', created during data preprocessing module. Logsequence.java is implemented as below, to extract usernames and pages accessed by that particular user from log file named as 'log.txt'. Input: Web-pages accessed by particular user, extracted from log file log.txt, created in data preprocessing module. Output: Sequence of Web pages accessed by each particular user will be stored in text file named as Logdata.txt.

Input: log.txt
Output: WAS in Logdata.txt
Process:

1. Begin
2. Read log.txt
3. **for** each line from log.txt.
4. Tokenize Strings from each line.
5. Put usernames and pages accessed by that particular user in hash table.
6. Extract usernames and pages accessed from Hash table.
7. **end for**
8. Write extracted usernames and Sequence of Web-pages accessed by that particular user to text file to Logdata.txt.
9. Close file.
10. End

Algorithm: Web Access Sequence

In this work, Web usage knowledge, is in the form of Frequent Web Access Patterns (FWAP), i.e. patterns of frequently visited Web-pages. An itemset is a frequent itemset only if it occurs in atleast specific percent of user transactions, this specific percentage will be minimum support value. This can be implemented by using Apriori algorithm implemented as illustrated below:

APRIORI ALGORITHM:

It is a classic algorithm used in data mining for learning association rules. For example, suppose you have records of web pages of a website, accessed by different users as follows:

Input: Logdata.txt ---- Web Access Sequence (WAS)
Output: FWAP.txt ---- Frequent Web Access Patterns (FWAP).
Process: numItems, numTransactions, minSup /*parameters used for number of items, number of transactions and setting minimum support respectively*/

1. **Begin**
2. Assign id to each page title and put it in hashtable.
3. Read Logdata.txt file
4. **for** each line from Logdata.txt do
5. count numTransactions //(number of users who accesses pages)
6. count numItemsets // (number of sequences of web pages accessed by each user)
7. **end for**
8. **for** each Item from numItems **do**
9. Add each itemset candidates to arraylist //Create itemsets of size1.
10. **end for**
11. **while** itemset size > 0
12. minSup = (support/numTransactions)
13. **if** (numItemset ≥ minSup) **then**
14. create itemsets of size 2
15. create itemsets of size 3

/* by taking only pages accessed by users greater than or equal to minimum support */

16. **end if**
17. create new itemsets from previous ones
18. **end while**
19. Trigger action if a frequent item set has been found

20. Write frequent itemset in specific text file in particular folder. Get Frequent Web Access Patterns (FWAP).
21. Close files.
22. End

Algorithm: Apriori

Page prediction model

It is necessary to predict pages based on frequent web accessed patterns, so page prediction algorithm is applied on frequent web access patterns which are results obtained by applying Web Usage Mining Technique. The results of page prediction will be helpful to get Web-page recommendations. The Page Prediction Algorithm is as below.

- Input: FWAP.txt
- Output: PPM.txt

- Process:

1. Begin
2. Initialize M=0. //Where M = pages to be recommended.
3. Read FWAP.txt
4. Read each line of FWAP i.e. each sequence of Frequent Web Access Pattern
5. Split each page titles from sequence and store them in String array.
6. Initialize cNode objects with NodeName = t_i; t_{i+1}; t_{i-1} and Occur=1 if they are not found in M.
7. Initialize a cOutlink object with Name = $t_i\_t_{i+1}$ and Occur =1 if it is not found in M.
8. Increase t_i, Occur and $t_i\_t_{i+1}$. Occur if they are found in M.
9. $t_i\_t_{i+1}$, linkTo= t_{i+1}
10. t_i, outLink=$t_i\_t_{i+1}$
11. t_i, inLink=t_{i-1}
12. Update all objects into M
13. Repeat from step 4 to 10 for each $p_i \in F$; /*Where, p_i is a page that belongs to set F i.e. set of frequent web access patterns. */
14. Update transition probabilities in the cOutLink objects.
15. Repeat steps 4 to 12 for each F = $t_{1.....}t_m \in F$. Where $t_{1.....}t_m$ is a sequence from set of sequences i.e. F.
16. Return M
17. End.

Algorithm : Page prediction

Web usage based Web-page recommendation:

For effective Web-page recommendations PPM.txt file is used resulting from Page Prediction Model explained above, for Web-page recommendations. It predicts pages with the help of page prediction algorithm.

- Input: FWAP.txt, FVTP.txt, PPM.txt
- Output: Recommended Web-pages;
  Recommended Domain terms of recommended Web-pages.
- Process:

1. Begin
2. Read PPM.txt
3. Read each line of PPM.txt and store in String.
4. Split each strings having '_' and store in a String Array //String[] arr = str.split("_");
5. Create an array list //ArrayList s = new ArrayList();
6. Extract predicted page from String array :
   a. if array length ≥2,
   b. if selected item form jComboBox is at position 0 i.e. arr[0]

Add string from position 1 to ArrayList i.e. arr[1] to s.add(arr[1]);
c. end if
d. end if
7. Read FWAP.txt
8. Read FVTP.txt
9. Read each line of FWAP.txt
10. Read each line of FVTP.txt
11. Split all frequent Web-page titles from FWAP.txt and store in String array.
12. for all terms in String array of frequent viewed terms.
13. Split all frequent viewed terms from FVTP.txt and store in String array.
14. Assign each frequent Web-page titles from String array to a string variable.
15. Assign each frequent viewed term from String array to a string variable.
16. Put string variables from step 13 and step 14 to hash table object.
17. end for
18. Set empty string before appending to add new strings of newly selected page to jTextArea.
19. Append recommended next page title or page titles of selected page from jComboBox to jTextArea1 as below. Here recommended next page is in ArrayList object s from step 5.
//jTextArea1.append(s.get(i).toString() +"\r\n");
20. Append conceptual recommendations which are terms also called as domains of recommended page, from hash table to jTextArea as below:
//jTextArea2.append(ht1.get(s.get(i).toString())+"\r\n");
21. Close files.
22. End.

Algorithm: Web-page recommendation by Semantic Enhancement

3 EXPERIMENTAL RESULTS:

To estimate, the efficiency of the proposed models of knowledge representation and the recommendation strategies, performance of Web-page recommendation is tested using different datasets.

3.1 DATASETS:

The educational website (www.dktes.com) is used to run the experimental cases. The dataset is created by accessing web pages of website. Registration and login is given for users to access pages of website. So, users are identified by their user-ID and password. Pages are identified by their titles. So dataset consist of log records.

Each log record consists of User-ID, Page title accessed by that particular user, and time when that page was accessed. Work is on log record having 280 access Web-pages by particular users. Precision and Satisfaction are obtained by applying different minimum support values 0.2, 0.3. Thus, this is training data, and this dataset is suitable for the experiment.

Experimental evaluation is done on different datasets of respective log records for experimental results. These per dataset consist of log records having 250, 300, 350, 400 access Web-pages by particular users. Minimum support values 0.2, 0.3 are applied on these datasets to get two major performance metrics: *Precision* and *Satisfaction*, as explained above. Thus, these are our testing data or datasets. The experimental results are showing better performance in terms of precision and satisfaction. The comparisons of these experimental results are illustrated in next section below.

3.2 Methodology:

The performance of Web-page recommendation is measured in terms of two major performance metrics: *Precision* and *Satisfaction* [6, 7]. In order to calculate these two metrics, two definitions: Support and Web-page recommendation rules, are introduced as follows:

SUPPORT:

Definition 1 Support [2]:

Given a set Δ of Web Access Sequence WAS and a set $P = \{P_1, P_2, \ldots P_n\}$ of frequent (contiguous) Web access Sequences over a set Δ of WAS, the support of each $P_i \in P$ is defined as:

$\sigma(P_i) = |\{S \in \Delta: S \in P_i\}| / |\Delta|$, where S is a WAS.

Support value help in removing uncommon Web-pages and determining Frequent Web Access Pattern (FWAP) from Web Access Sequence (WAS). This is accomplished by setting a Minimum Support (*MinSup*) and using it as a threshold to check WAS. The Web Access Sequences obtained, whose support values are greater than, or equal to minimum support (*MinSup)* value are considered as Frequent Web Access Pattern.

3.3 WEB PAGE RECOMMENDATION RULES:

Definition 2 Web-page recommendation rules:

Let $S = s_1 s_{2\ldots.} s_k s_{k+1\ldots} s_n$ ($n \geq 2$) be a WAS. For each prefix sequence $Sprefix = s_1 s_{2\ldots} s_k$ ($k \leq n-1$), a Web-page recommendation rule is defined as a set of recommended Web-pages generated by a Web-page recommendation strategy, denoted as $RR=\{r_1, r_{2,\ldots.} r_m\}$, where $r_i (i=[1\ldots.M])$ is a recommended Web-page.

A Web-page recommendation rule is considered as correct, and/or satisfied, or empty based on the subsequent conditions:

1. If $s_{k+1} \in RR$, RR is correct.
2. If $\exists s_i \in RR$ ($k+1 \leq i \leq n$), RR is satisfied.
3. If M=0, RR is empty.

Given a set of recommendation rules, $R= \{RR_i, RR_{i+1,}\ldots..RR_{i+n}\}$, where $RR_i (1 \leq i \leq n)$ is a recommendation rule, and the total number of recommendation rules in R is $|R| = N$ including empty rules, the performance of Web-page recommendation strategies is measured in terms of two major performance metrics: Precision and Satisfaction. The precision is beneficial to determine how feasible a user will access one of the recommended Web-pages. Moreover, it is also required to deliberate if a user accesses one of the recommended Web-pages in the near future. The succeeding page accessed by a user may not be the goal page that user wants. Many times, a user has to access a few in-between pages before reaching the goal page. Therefore, the 'satisfaction' 's', is necessary to give the 'precision' 'p', that the recommended pages will be accessed in the near future as defined below [6, 7].

Definition 3 Precision :

Let R_c be the sub-set of recommendation rules R, where $R= \{RR_i, RR_{i+1,} \ldots..RR_{i+n}\}$, which consists of all correct recommendation rules. The Web-page recommendation precision is defined as:

- Precision= $|R_i| / |R|$

Definition 4 Satisfaction:

Let R_s be the sub-set of recommendation rules R, where $R= \{RR_i, RR_{i+1,} \ldots..RR_{i+n}\}$, which consists of all satisfied recommendation rules. The satisfaction for Web-page recommendation is defined as:

- Satisfaction= $|R_s| / |R|$

3.4 EVALUATION OF PARAMETERS:

According to the observation, on the used dataset the size of exposed FWAP is too large to run the algorithms when MinSup value is lower than 0.3%, and the size of it becomes too lesser to test the algorithms when MinSup is higher than 1%. Hence, in these experimental cases, a number of MinSup values are chosen for each case ranging from 0.2% to 1.0% to control the quantities of FWAP that are discovered from the Web usage mining process, and to guarantee that the amount of generated information is enough to test the proposed models.

Work is done on different datasets of respective log records for experimental results. These per dataset consist of log records having 250, 300, 350, 400 access Web-pages by particular users. Minimum support values 0.2, 0.3 are applied on these datasets to get two major performance metrics: *Precision* and *Satisfaction*, as explained in section, above.

Set of 8 recommendation rules are obtained, for dataset having log records of 250 access Web-pages. For datasets with log records of 300, 350 and 400, respectively, set of 20 recommendation rules are obtained. Then correct recommendations and satisfied recommendations are counted from these set. Using this information Precision and Satisfaction are calculated according formulas defined in section above. The comparisons of these experimental results are represented in table 1.

The evaluation measure value (Precision) with minimum support (MinSup) values which are 0.2, 0.3 on different datasets (web-page access records) are depicted in figure 1 below, where the horizontal axis represents different datasets (web-page access records), and the vertical axis on the left part represents Precision with minimum support values (MinSup).

The evaluation measure value (Satisfaction) with minimum support (MinSup) values which are 0.2, 0.3 on different web-page access records are depicted in figure 2 below, where the horizontal axis represents different

datasets (web-page access records), and the vertical axis on the left part represents Satisfaction with minimum support values (MinSup).

Datasets (web-page access records)	Support	Precision	Satisfaction
250	0.2	0.75	0.5
300	0.3	0.25	0.25
350	0.2	0.45	0.45
400	0.3	0.35	0.3

Table 1. Experimental results

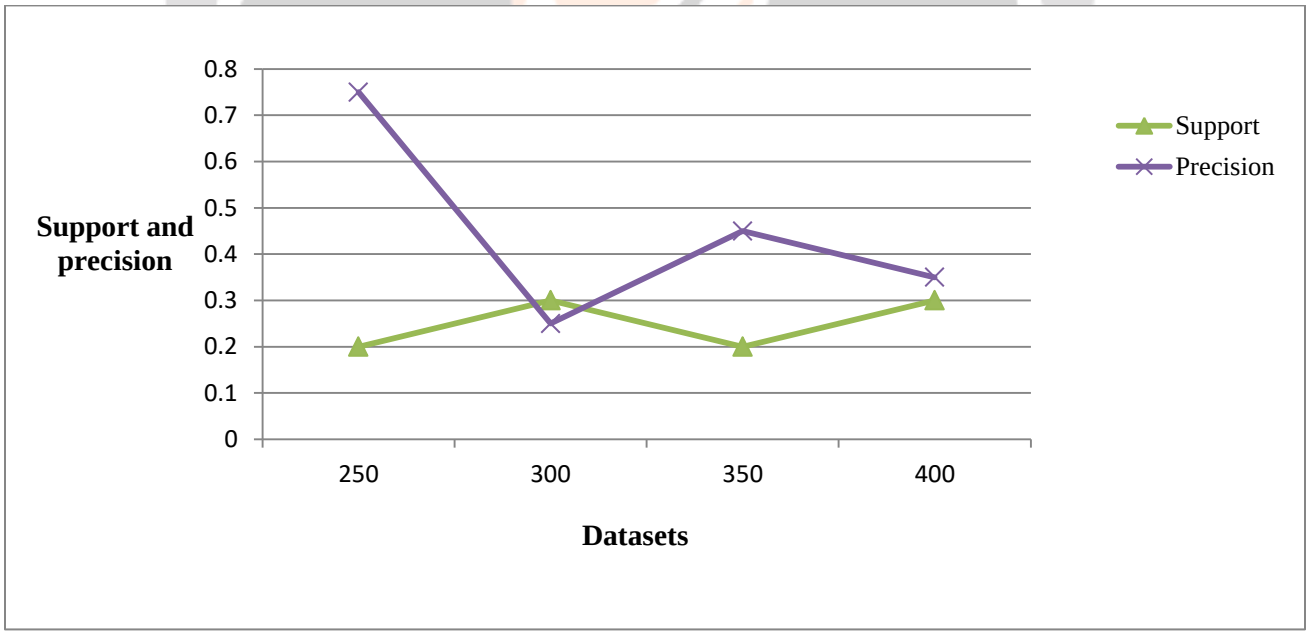

Fig 1. Experimental evaluation on different datasets to obtain precision

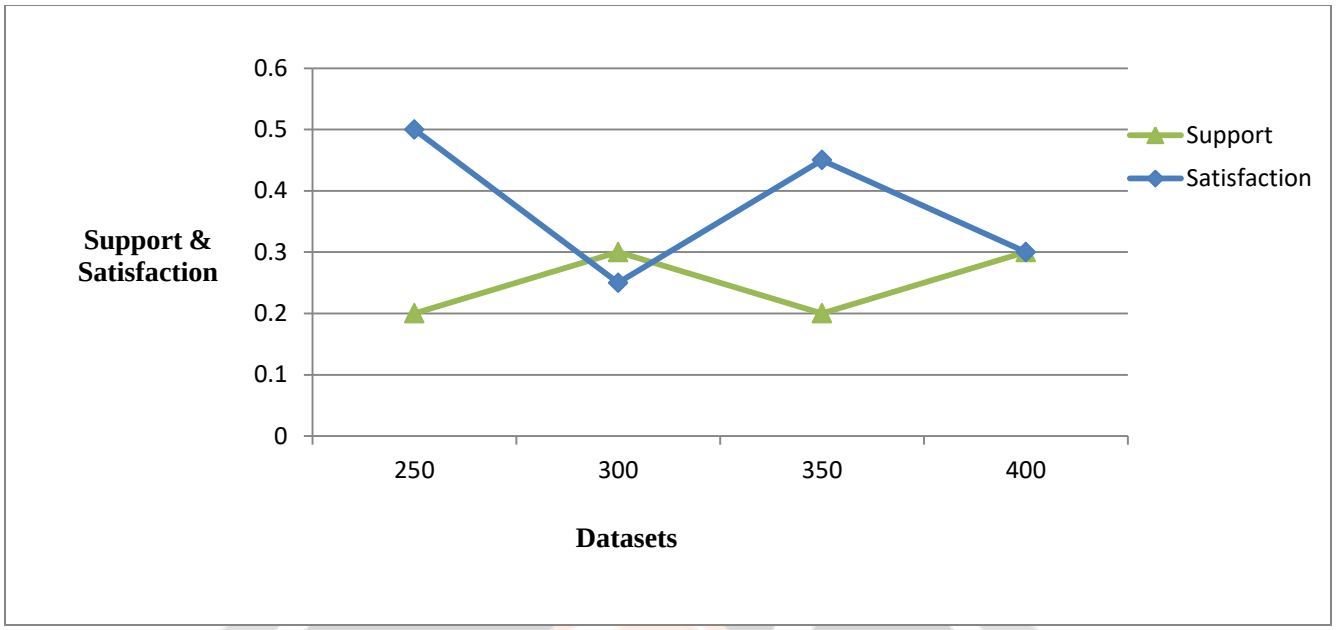

Fig 2. Satisfaction based on different MinSup value

4. CONCLUSION

This paper, demonstrated, Web-page recommendation system, based on web usage knowledge. It focuses on its architecture and implementation, where, Web usage knowledge model, uses Apriori algorithm. Web-page recommendation strategy, i.e. page prediction model, helps to predict next Web-pages. Finally, the experimental results are elaborated, which are, auspicious and indicative of usefulness of mentioned models.

5. REFERENCES

1. B. Liu, B. Mobasher, and O. Nasraoui, "Web usage mining", in Web Data Mining: Exploring Hyperlinks, Contents, and Usage Data, B. Liu, Ed. Berlin, Germany: Springer-Verlag, 2011, pp. 527–603.

2. G. Stumme, A. Hotho, and B. Berendt, "Usage mining for and on the Semantic Web", in Data Mining: Next Generation Challenges and Future Directions. Menlo Park, CA, USA: AAAI/MIT Press, 2004, pp. 461–480.

3. Vijayashri Losarwar et al., "Data Preprocessing in Web Usage Mining", International Conference on Artificial Intelligence and Embedded Systems, July 15-16, 2012, Singapore.

4. Sonule Prashika Abasaheb et al., "A Survey on Web page recommendation and Data Preprocessing", International Journal of Computer Engineering In Research Trends Volume 3, Issue 4, April-2016.

5. H. Dai and B. Mobasher, "Integrating semantic knowledge with web usage mining for personalization," in Web Mining: Applications and Techniques, A. Scime, Ed. Hershey, PA, USA: IGI Global, 2005, pp. 205–232.

6. B. Zhou, S. C. Hui, and A. C. M. Fong, "Efficient sequential access pattern mining for web recommendations," Int. J. Knowl.-Based Intell. Eng. Syst., vol. 10, no. 2, pp. 155–168, Mar. 2006.

7. Thi Thanh Sang Nguyen, Hai Yan Lu, and Jie Lu, "Web-Page Recommendation Based on Web Usage and Domain Knowledge", IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, VOL. 26, NO. 10, OCTOBER 2014.