ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 5, Issue 02, February 2018

Mining of Frequent Patterns using Session ID with K-Nearest Neighbor Algorithm

D.Gandhimathi[1], N.Anbazhagan[2]
Department of Computer Science
Alagappa University, Karaikudi, India
[1]Mathi.mathi65@gmail.com
* Department of Mathematics
Alagappa University, Karaikudi, India
[2] Anbazhagan_n@yahoo.co.in

Abstract: Web mining is one of the essential areas in data mining. Frequent patterns mining is considered efficient task in web data mining. In this paper, the frequent page patterns considered as results by web user activities on website. First; the session records are going to log structure. The Alpha Numeric session Id ($\mathcal{U}si$) is generated for every page by web user's clicks. Session Id is named as sequence of pages. So, the session id identified as string for mining the frequent patterns. Second, comparison is making of first session Id with previous and so on. Third, K-Nearest Neighbor algorithm is used to classify unknown data to known according to Euclidean Distance. Fourth, we check whether count is equivalent to minimum support or not. If it is matched to count value it must be increased. In frequent list, the combinations are realized by using recursive method and produce results.

Keywords: Web usage mining, Session identification, K-Nearest Neighbor, Classification, Frequent pattern mining.

I. INTRODUCTION

Web mining is mainly used for mining associated patterns from web data. Web usage mining is called as web-log mining is the process of mining web page patterns from logs and other web usage data. Every individual session is identified by the Alpha Numeric Session-Id ($\mathcal{U}_{si}$), after that the results will be stored in the log source file. An important thing is, there is no need to scan the whole large dataset for mining the frequent patterns. Second, every session Id is compared with the previous session Id data. If those pages are available in the previous set then the values are increased otherwise not increased. Third, The K-Nearest Neighbor algorithm is used to classify the unknown patterns to known samples. The classification is significant because we needn't process all of the data whichever is accessed by user. Forth, when each page's count value is greater than the minimum support count value those will be accumulated as pages of frequent list. In the frequent list, the combinations are performed then it will be compared with every individual session records. Purpose of the frequent pattern mining is what are the pages are highly associated with each other than the others. In this paper, we discuss the performance evaluation task of the work and the related results are shown in the consecutive figures.

II. RELATED WORK

C.Dimopoulos, C.Makris and et al present the problem of web page usage prediction in a web site by modeling users' navigation activity and web page content with weighted suffix trees were discussed in [4]. Z.Pabarskaite and A.Raudys recognize a number of web log mining sub topics including specific tasks such as data cleaning, user and session identification are presented in [13]. D.Dong presents the approach based on data cube stresses on turning web logs into structuralized data cube it can understand multi angle comprehensive mining and analysis in addition to introduce a variety of mature data mining technologies are explained in [5]. G.Fang, etal present The algorithm turns session pattern of online user into binary, then uses up and down search strategy to double create candidate frequent itemsets are explained in [7]. Ritika Agarwal and et al. have attempted to provide a new viewpoint to the KNN classifier with the eye of a modified apriori algorithm those are discussed in [1].

A.Kousar Nikhath , K.Subrahmanyam and et al presents two various processes first, the training processes use a earlier categorized set of documents to train the system. Second, the classifier uses the training 'model' to categorize new incoming documents are explained in [12].

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 5, Issue 02, February 2018

P.G.Srinivas, P.K.Reddy and etal presents a model of coverage patterns (CPs) and approaches for mining CPs from transactional databases were discussed in [16]. R.Cooley, B.Mobasher and J.Srivastava present several data preparation techniques in order pattern instead of confidence all detailed works are explained in [3]. Poornalatha G, P.Raghavendra present in order to cluster the data, similarity measure is essential to attain the distance between any two possible sessions are handled in [14]. T.T.Aye mainly focuses on data preprocessing of the first phase of web usage mining with activities like field extraction and data cleaning algorithms. Field extraction algorithm performs the process of separating fields from the single line of the log file all are discussed in [2]. A.Guerbas, O.Addam and et al proposes a refined time-out based heuristic for session identification. Second, the researchers are suggesting the usage of a particular density based algorithm for navigational pattern discovery. Finally, a new approach for efficient online prediction is also suggested all are handled in [8].

R.Tang and Y.Zou present an approach that routinely identifies service composition patterns from a variety of applications using execution logs that were explained in [17]. S.Dua, E.Cho and S.S.Iyengar presents novel techniques to mine the sets of all predictive access sequences from semi-structured web access logs. The predictive access routines discovered by AllFreSeq are useful for understanding and improving website domain tree all are discussed in [6]. H.F.Li presents a new online mining algorithm, known as Top-DSW (top-k path traversal patterns of stream Damped Sliding Window), to find out the set of top-k path traversal patterns from streaming maximal forward references, where "k" is the desired number of path traversal patterns to be mined in [10]. S.Sisodia, M.Pathak, etal presents a frequent sequential traversal pattern mining algorithm with weights constraint all are discussed in [15].

D.G.Lorentzen presents differences in type of data can also be seen, with webometrics more focused on analyses of the structure of the web and web mining more focused on web content and usage, even though both fields have been embracing the possibilities of user generated content are explained in [11]. Y.Tao, T.Hong and Y.Su presents the cornerstone research results on IBD, that help open up the scope of WUM applications or decision support for knowledge discovery by jumping out the existing frame of algorithm as well as the conventional web log records all are manipulated in [18]. Y-F.Huang and J-M.Hsu presents an access sequence miner to extract popular surfing 2-sequences with their conditional probabilities from the proxy log, and stored them in the rule table are explained in [9].

III. SCOPE OF RESEARCH

Most of the existing methodologies are applied to discover the frequent patterns from large dataset. But, in this proposed work, we have used Alpha Numeric session Id to find out the accurate results. Mainly find out the frequent patterns are important for improving the website quality in web usage mining.

IV. PROPOSED WORK

Mainly, click stream data can be considered as a source of information, because it incorporates navigational details of the internet user on the website. The main reason for examining click streams is to extract associated data about searching activities of users on website. However, it is a complicated task to evaluate from the unstructured data and many diverse formats depending on the web server. Logs can be mined as .txt files from the web server. Here, highest time utility is required to do examine of all surfing activities of the internet user. So, Alpha Numeric Session Id is used in this paper by us to get rid the most frequent pattern results. Commonly, the file is processed for web usage mining works. But, in this paper the pages and count values are processed whenever a page is clicked by the user that's count value is opened. Again, click on different page it has a new count.

In the proposed methodology, it has distinctive formation when it is compared with the existing procedures. We focus on the four various components such as: at first, each session is recognized with different Alpha Numeric Session-Id ($\mathcal{U}_{si}$). The session oriented results will be accumulated in the log source file. Second, every session Id is compared with the prior collection of the session Ids. Suppose, if the match is true then the values are incremented by one. Third, K-Nearest Neighbor method is used to produce the frequent group results based on classes. Fourth, when its count value reaches the minimum support count value it will be stored as a page of frequent list. In the frequent list, the permutations are performed by the recursive method then it will be compared with every individual session. User's closing behavior of the browser is progressed on the figure 1. Every *RSE* are identified by the user navigational behavior after that the earlier one is changed to *PSE*. This changeover is most applicable one in this methodology than the other existing works.

A) Session Identification using Session Id($\mathcal{U}_{si}$)

Session Identification is the prominent process of partitioning the user activity record of each user into sessions, each representing a single visit to the website. Web

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 5, Issue 02, February 2018

sites don't have the advantage of extra authentication information from users and without mechanisms such as embedded session ids must rely on heuristics methods for session identification. The goal of session identification is to reconstruct from the click stream data, the actual sequence of actions performed by one user during one visit to the site. Here, session is identified by the unique session identifier namely Session ID ($\mathcal{U}_{si}$). This session identifier has numeric values and alphabetical also. For example, a single session has rp1, rp2, rp3 but the session id is a string like "1s2a3k". According to this method, we should know about the session has three various pages. The alphabets "s a k" are inserted at the middle of the every unique visited pages in every single session.

Given: Initialize Unique Session Identifier $\mathcal{U}_{si} = 0$, Previous Set Elements within Session-Id (PSE) and Running Set Elements within Session-I d (RSE)

1. $\mathcal{U}_{si} = 0$
2. If the user clicks on different pages
3. Each page has $\mathcal{U}_{si}$
4. If the user closes the browser
5. The (RSE) are identified
6. Else
7. The (RSE) are processed
8. If the user clicks on different pages again after closing of browser
9. The (RSE) are identified, at that time the before one of the (RSE) that is known as the (PSE)
10. End if
11. End if

Fig. 1. Each user session based on their session identifier $\mathcal{U}_{si}$

Initially, we recognize every distinctive session data by this Alpha Numeric Session Id is as a string ahead of it reaches to the log file. Figure 1 shows every distinctive session is identified before the browser is closed. After the closing of browser by the internet user, if the user looks at the pages another time of the website then it will be considered as a fresh session's pages in this study. And also after the closing behavior of the browser the individual session id is created. In this proposed work, unique ($\mathcal{U}_{si}$) is structured for every distinct page-click by the user. If the similar page is clicked by the same user again in that particular same session then it will not be considered as a count. Because, that page-click has its unique session id and that has been recorded previously. According to this manner, the replication is excluded that means the prior clicked page is not come again in further on that identical session.

1. *String Representation is used for every (session) Identification:*

We consider every session as a set and every session id as a string. In this paper, we have considered every session as every classical (crisp) set. Because, every session has number of accessed pages and the session id string for every individual session. This string is alpha numeric characters. The classical (crisp) set theory describes sets as the "collection of objects called elements of a set". These are referred as crisp naturally because they only inform whether an element is a member or not. A classical set is defined by crisp boundaries. In this paper, the classical set is used because of whenever the browser is closed by the online visitor such kind of set is regard as the classical set naturally. Here, a finite set has a first element, second element, until it attains its k^{th} element. It didn't keep going forever on the number of elements in the finite set. So, we have considered as finite set for every single session elements.

A finite set is one in which it is possible to list and count all the users of the set. Every finite set is denoted by,

$$S = \{\text{Accessed-Pages by Users}\} \quad (1)$$

For example, we consider first session pages only;

$$S = \{rp1, rp2, rp3\} \quad (2)$$

The number of elements in a finite set S is denoted by,

$$n(S) = 3 \quad (3)$$

Let's consider the session $S_1 = \{1, 2, 3\}$ and session $S_2 = \{2,4\}$. Both sets are finite, but their cardinality is different. Set S_1 has a cardinality of 3 as opposed to set S_2 has a cardinality of 2. Cardinality is used to express finite sets of numbers. Look at how this relation compares between these two sets. The size is mathematically expressed as cardinality. But, whenever the user clicks on pages it is appeared as alpha numeric characters. These alphabetic are available within the S_1 in equation 1. The purpose of inserting alphabets within every session string is we are able to identify every individual visited page. According to the alpha numeric session id we can identify every unique page within session. From table I has contains diverse finite set of sessions. Table I shows seven various sessions for obtaining the count of every individual page-click. Here, each session also has the alpha numeric session id. Table I represents every unique session and its unique number of online page clicks. Here, we have considered every unique session's pages as finite set of pages. The first row of the table I reveals session-1 pages that are known as single finite set. If that page is present on the session the count is initialized by 1 otherwise null. In the second row, wherever the page is

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 5, Issue 02, February 2018

present it will be increased by 1 otherwise all are null. In the second session, two page clicks are accessed by the online user (rp2:2, rp4:1). One prominent task is running here that is the first session has (rp1:1, rp2:1, rp3:1) and the remaining are (rp4:0, rp5:0). Session-2 contains only (rp2:2, rp4:1), here rp2 has been presented in the earlier session also. Next, we consider page rp4 but the same page click is not available in the earlier session so it has the value "zero". Here, we assume the sample set of online page clicks as follows:

TABLE I: THE SAMPLE SET OF ONLINE-PAGE CLICKS

Sessions	*Accessed-Pages by Users*
S_1	*rp1:1,rp2:1,rp3:1*
S_2	*rp2:2,rp4:1*
S_3	*rp1:2,rp2:3,rp4:2*
S_4	*rp3:2,rp5:1*
S_5	*rp4:3,rp5:2*
S_6	*rp1:3,rp3:3,rp4:4,rp5:3*
S_7	*rp1:4,rp4:5*

We consider next session (rp1, rp2, rp4) but one essential thing is how we can decide the value of rp1 because the page-click is not present in the previous session. So, we check whether the page is available in the previous session or not. This difficulty is not in the rp2 page because it has the earlier visit by the user and also rp4. From this, we are able to know every session value is changed when the entrance of next session. The same method is continued throughout the whole process.

B) Comparisons between the Previous Session Id Elements (PSE) and Running Session Id Elements (RSE)

Two various strings are processed here; such as 1.Running String (*RS*) and 2.Previous String (*PS*). The Running string has all of the clicked pages that mean every accessed page is computed on the running string. The pages are available in every unique alpha numeric session id. So, we are able to process the accessing pages by the alpha numeric session id. After that, the string is called as previous string that string is before the running string. Click stream data can be obtained from page clicks directly it is referred to as hits. These log details of internet visitors must be accumulated as particular file. Various log files were generated and that can be divided as access logs, error logs, distinct logs and others. Website owners have entire control over these log source files. In this paper, every (*RS*) has number of page requests with distinctive ($\mathcal{U}_{si}$). This session id is generated automatically when the internet user accesses the page.

Here, every running string is compared with the previous string (*PS*). For example, If the (*PS*) has (rp1:3, 0, rp3:3, rp4:4, rp5:3), it represents this session only encloses pages 1 and 3 through 5. This has the alphabets also. The comparison is running between the running session string and the previous string. The results are (rp1:4, 0, 0, rp4:5, 0) this means that the running string has rp1 and rp4. From this the resulting pages are 1 and 5 then both are incremented probably in this mechanism and all the other clicks are null. There is no valid accessing on those pages by the internet users on the website.

TABLE II: COMPARISON BETWEEN TWO STRINGS

PSE	***RSE***
	rp1:1,rp2:1,rp3:1
rp1:1,rp2:1,rp3:1	*rp2:2,rp4:1*
rp2:2,rp4:1	*rp1:2,rp2:3,rp4:2*
rp1:2,rp2:3,rp4:2	*rp3:2,rp5:1*
rp3:2,rp5:1	*rp4:3,rp5:2*
rp4:3,rp5:2	*rp1:3,rp3:3,rp4:4,rp5:3*
rp1:3,rp3:3,rp4:4,rp5:3	*rp1:4,rp4:5*

Every internet user is searching on the website with arbitrary manner so that are not a predictable one by us. Whenever, the access is made on pages that will be collected as pages for individual session. Table II shows that, first finite set of session is known as Running String.

It has three different page values such as {rp1:1, rp2:1, rp3:1}. The finite set is defined by user's closing behavior of the browser. Then, this first string of session is considered as Previous String Elements. Then, we took next simultaneous string to process as Running String of Elements. Again, this will be recognized as Previous String Elements. The comparison is made by set S_1 {rp1:1, rp2:1, rp3:1} and set S_2 {rp2:2, rp4:1}. According to the table II every ***PSE*** elements are mapped with the consequent ***RSE*** elements. After the comparison process is done, we can get individual pages and their count values also.

C) K-Nearest Neighbor Algorithm for classifying the data

The K-NN algorithm is belongs to the family of instance-based and lazy learning algorithms. After the classification, we must get the necessary group of records so that process is made on one class of classified records only.

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 5, Issue 02, February 2018

A data is classified by a majority vote of its neighbors, with the data being assigned to the class most common amongst its K-nearest neighbors measured by a distance function. If K = 1, then the data/pattern is assigned to the class of its nearest neighbor. All the data correspond to pattern points in the n-dimensional space. Here, the nearest neighbors are defined by terms of Euclidean Distance. The algorithm assigns to a point X the class for the majority of its K-nearest neighbors from a sample pattern set $rp_i \in \{rp_1,\ldots rp_{l'}\}$ of known classification, where $l' \geq l$ and each of these data belongs to one of the l classes. A nearest-neighbor (NN) classification rule assigns an unlabeled pattern X to the class of its nearest neighbor, where $rp_i \in \{rp_1,\ldots rp_{l'}\}$ is a nearest neighbor to X if the distance,

$$D(rp_i,X)=\min_j\{D(rpj,X)\}, \text{ for } j = 1, 2, \ldots, l' \quad (4)$$

Here, we consider another example from the user clicks; table III represents set of sessions only not alphabets and visited pages by the user. As per session-1's result, the request page 1 and also 5 are visited two times. In session-2 and 6, the request page 5 is accessed by the online user frequently. Further, the session-5 contains the page 4 and session-6 contains page 3 more than once. Finally, session-7 has the page 2 twice. In some sessions, some pages are accessed by visitors again and again so that the frequency of that page will definitely increase. The K-Nearest Neighbor algorithm is used to classify the unknown patterns. The classification is very important because we needn't process all of the data whichever is accessed by user minimum number of times. By the use of class label of the classification algorithm, which class is required for the further progress we are able to utilize such class records only.

TABLE III: SESSIONS WITH VISITED PAGES

Sessions	*Pages visited by users*
S_1	*rp1,rp2,rp3,rp5,rp6,rp1,rp5*
S_2	*rp2,rp3,rp4,rp5,rp6,rp5*
S_3	*rp1,rp2,rp4,rp5,rp6*
S_4	*rp1,rp2,rp3,rp4,rp5,rp6*
S_5	*rp4,rp5,rp4,rp6*
S_6	*rp3,rp4,rp5,rp6,rp3,rp5*
S_7	*rp1,rp2,rp3,rp4,rp6,rp2*

First of all we have to decide the X_1 and X_2 values for this paper that is the X_1 represents how many sessions have contained every page as well as X_2 represents total count of every page. Two different class labels are used namely C_1 (Most-Frequent) and C_2 (Frequent). C_1 have gathered pages those have more number of count values totally and also depends on sessions. At that same time C_2 aggregated some requested pages with minimum number of counts when compared with the C_1.

The steps for KNN algorithm are:

1. Determine the parameter K that means the number of nearest neighbors: A simple approach to select K is, set

$$K= n\hat{}\left(\frac{1}{2}\right) \quad (5)$$

   Where n is the number of instances in the feature pace. By using the equation 5 we selected the k value.
2. Calculate the distance between the query patterns with all of the other pattern points. The query pattern point is (6,7). Euclidean distance is calculated with every point from the query point.

$$D(X,q)=\sqrt{(X_1 - q_1)^2 + (X_2 - q_2)^2} \quad (6)$$

   The equation 6 is used to calculate the distance between every data point (X) with the query point (q).
3. Sort the distance and determine the Nearest Neighbors based on the "K"th minimum distance. After getting the results, ranking is performed based on the minimum distance. The result will be (7,7), (5,6), (5,6), (6,9), (4,5). So, (7,7) is almost nearest with the query data and also (5,6), (5,6) are very close to the query point.
4. Collect the category "Y" of the nearest neighbors. According to the value of K=3 so, the point (6,7) is gathered by the class C_2 because of the class C_1 has one point (7,7) only. But, the class C_2 has two points (5,6) and (5,6).
5. According to the majority of the category of nearest neighbors, we conclude 2 frequent categories and 1 most-frequent category of class. So, the query point (q) can be classified into the class C_2.

After getting the classification results, we have to get the minimum number of combinations only. Several combinations are available but we need highest combinative pages only. For example; one thousand and more related request pages are obtained by this classification method. So, when the classification is done use the minimum support value to get the most frequent results from the class. We considered every session as a set and it goes to the classification part. Then, the classified records are processed and compared every alpha numeric session Id elements with the previous alpha numeric session Id elements. After that we consider only the visited pages not alphabets.

D) Getting results using support count value

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 5, Issue 02, February 2018

The Most-Frequent class records are processed here. An item set is a collection of items. Generally, Minimum-support is used to cut the search space and to limit the number of rules generated. Putting of solitary min-sup count value for item set/pattern does not work in real life application. In various applications, some patterns come into view very frequently in the data, while other patterns seldom appear. There are two problems here such as: 1.If the min-sup is set too high, one loses rules that have frequent patterns or rare patterns in the data. 2. If we set min-sup very low, in order to find rules those have both frequent and rare items easily. Traditionally, in the data mining the purpose of the minimum support count value is to produce the accurate frequency based results. Based on the consideration of the support value, the frequency based results will be accumulated on the set. Here, we consider the support value (s) is 3 that is denoted by s=3. When, the frequency count is greater than or equal to the support value that will be gathered as the frequent page.

$$\text{Frequent page Count} \geq \text{Support Value } (s) \qquad (7)$$

The equation (7) represents check the frequent count value is greater than or equal to the support value. After that, it will be included to the frequent list otherwise not. Then we can get the accurate frequent pages as associated manner. Several combinations are created by this method in this work so we have to identify the accurate number of combinations only. The Most-Frequent class results are processed and use the support count to fin-out the frequent page results.

The permutations are performed for the frequent class results:

All of the class data are already within the Most-Frequent list of patterns but the support value is necessary to check the particular data must be a valid one or not.

```
Given: Initialize Results (R)
1. Get Combinations
2.        For each value (v) in the text
3. For each string value (r) within the result R
4.          Add both values to R
5.          Add "v" value to R
6.  End for
7.  End for
8.  Return R
```

Fig. 2 The collective combinations from the frequent list of requested pages

So, we have to use an accurate support count value to get the frequent list of request pages. Several existing frequent pattern mining algorithms are used to get the accurate list of permutation results. Such algorithms are Apriori, frequent pattern tree structures and others. But, these traditional number of permutation algorithms have some effective drawbacks. We have to overcome those drawbacks by using the simple permutation based recursive algorithm which is characterized by recurrence or repetition, in particular.

Take all those frequent records and do the recursive algorithm to obtain such kind of combinations effectively. Let's first try sorting the output by the length of the string. That is every data point is placed by single and combined with another for example; (a,b,c) here, two combinational results will be (ab, ac and bc). Figure 2 represents the collective combinations from the frequent list of requested pages. The reason behind for getting such types of combinations is, we are able to get the associative pattern results as frequent. In the web usage mining process, whatever pages are accessed simultaneously by the online users. Most of the visitors have visited the pages associatively. So, those pages are associative pages then it is very useful for improving the quality of those pages. This recursive algorithm must use to get the associated request pages.

V. EXPERIMENTAL RESULTS

All of the experimental results are given in this section. The proposed methodology is finished in four main tasks such as first; sessions are identified by using the alpha numeric session-Id is created for every session. Next, the comparison is made between both strings. Then classify the data because of numerous accessed pages are available. But, we are not able to take course of action for the whole number of records. After that, we check the count value is greater than the support value. Combinations are performed by Recursive method. All of the experimental tasks are conducted in the PHP environment. In this paper we have given the methodological work and the performance evaluation results in the PHP platform. We can identify the better accuracy results by this proposed methodology. This is mainly used to increase the quality of results. The system information is: Processor Intel(R) Pentium (R) CPU B960 @ 2.20 GH_z 2.20 GH_z, the System Type is 32-bit Windows 7 and Memory Capacity is 4.00 GB. The most accurate results of the frequent-1, 2 and 3 pages are presented in the figure 3. The figure 3 shows single count, double count and triple count values of pages in (a),(b) and (c) respectively. Only we have given the most accurate combinative results in the figure 3. So, the final results are giving here. The resulting pages are accessed by more number of internet users. The performance of this methodology outperforms are better than

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 5, Issue 02, February 2018

the other existing methods. The reason behind the associated frequent page results is we are able to know if an internet user accesses one page (rp2), the same user also accesses the another page (rp4). We are able to know one thing from here that is the pages rp2 and rp4 are accessed repeatedly by the online visitor. Those pages are considered as frequent list of pages. The use of this is we can improve the quality of the page. Improving the quality of the page is used to give more attention to that page.

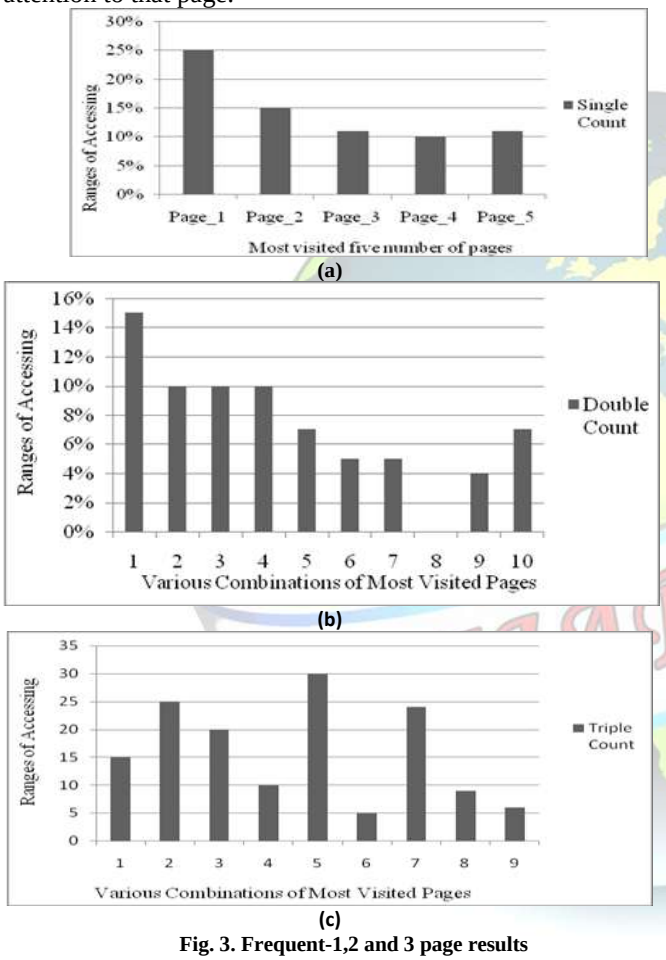

Fig. 3. Frequent-1,2 and 3 page results

VI. CONCLUSION

Most of the web usage mining methodologies are available to give the frequent pattern of records from user's navigational activities. In this paper, a new Alpha Numeric Session-Id (*SI*) based task is completed to create the frequent results without searching the log source file. The session identification is different with the click stream related tasks. Further, it will be accumulated on log source file. Here, the time of the session identification is based on the access of every page. This type of identification is used to circumvent the duplicate access of the same page that was accessed by the same session visitor. After that, the comparison is completed with the previous one. If the value is null in the previous session record for that page at that time it will get the before one from the earlier session to process. The entire surfing of log source file is never required. Use of K-Nearest Neighbor algorithm to obtain the most possible number of data points/request pages. Finally, combinations of the frequent page results are performed based on the support value and accurate results will be gathered using recursive method. The permutations and combinations are performed to finalize the associative results.

REFERENCES

[1]. Agarwal, R., Kochar, Dr.B., and Srivasta, D., May (2012). A Novel and Efficient KNN using Modified Apriori Algorithm. *International Journal of Scientific and Research Publications*, 2(5) 1-5.

[2]. Aye, T., March (2011). Web Log Cleaning for Mining of Web Usage Patterns. Computer Research and Development (ICCRD), 3rd International Conference on IEEE, 490-494, doi: 10.1109/ICCRD.2011.5764181.

[3]. Cooley, R., Mobasher, B., Srivastava, J., February (1999). Data Preparation for Mining World Wide Web Browsing Patterns. Knowledge and Information System, 1(1), 5-32, doi: 10.1007/BF03325089.

[4]. Dimopoulos, C., Makris, C., Panagis, Y., Theodoridis, E., Tsakalidis, A., April (2010). A web page usage prediction scheme using sequence indexing and clustering techniques. Data and Knowledge Engineering, 69(4), 371-382, doi: 10.1016/j.datak.2009.04.010.

[5]. Dong, D., May (2009). Exploration on Web Usage Mining and Its Applications. Intelligent Systems and Applications, ISA 2009, International Workshop on IEEE, 1-4, doi: 10.1109/IWISA.2009.5072860.

[6]. Dua, S., Cho, E., Iyengar, S., March (2000). Discovery of Web Frequent Patterns and User Characteristics from Web Access Logs: A Framework for Dynamic Web Personalization. Application-Specific Systems and Software Engineering Technology, Proceedings 3rd Symposium on IEEE, 3-8, doi: 10.1109/ASSET.2000.888025.

[7]. Fang, G., Wang, J., Ying, H., Xiong, J., December (2009). A double algorithm of Web usage mining based on Sequence Number. Information Engineering and Computer Science 2009, International Conference on ICIECS 2009, 1-4, doi: 10.1109/ICIECS.2009.5363879.

[8]. Guerbas, A., Addam, O., Zaarour, O., Nagi, M., Elhajj, A., Ridley, M., Alhajj, R., September (2013) Effective web log mining and online navigational pattern prediction. Knowledge-Based Systems, 49, 50–62, doi:10.1016/j.knosys.2013.04.014.

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 5, Issue 02, February 2018

[9]. Huang, Y-F., Hsu, J-M., February (2008). Mining web logs to improve hit ratios of pre fetching and caching. Knowledge-Based Systems, 21(1), 62-69, doi:10.1016/j.knosys.2006.11.004.

[10]. Li, H.F., October (2009). Mining top-k maximal reference sequences from streaming web click-sequences with a damped sliding window. Expert Systems with Applications, 36(8), 11304–11311, doi:10.1016/j.eswa.2009.03.045.

[11]. Lorentzen, D.G., January (2014). Webometrics benefitting from web mining? An investigation of methods and applications of two research fields. Scientometrics, 99(2), 409–445, doi: 10.1007/s11192-013-1227-x.

[12]. Nikhath, A.K., Subrahmanyam, K., and Vasavi, R., (2016). Building a K-Nearest Neighbor Classifier for Text Categorization. International Journal of Computer Science and Information Technologies, 7(1), 254-256.

[13]. Pabarskaite, Z., Raudys, A., February (2007). A process of knowledge discovery from web log data: Systematization and critical review. Journal of Intelligent Information Systems, 28(1), 79-104, doi: 10.1007/s10844-006-0004-1.

[14]. Poornalatha, G., Raghavendra, P., August (2011). Alignment Based Similarity distance Measure for Better Web Sessions Clustering. Procedia Computer Science, 2nd International Conference on Ambient Systems, Networks and Technologies (ANT), 5, 450-457, doi:10.1016/j.procs.2011.07.058.

[15]. Sisodia, M.S., Pathak, M., Verma, B., Nigam, R.K., Design and Implementation of an algorithm for finding frequent sequential traversal patterns from web logs based on weight constraint. Second International Conference on Emerging Trends in Engineering and Technology, ICETET-09, doi: 10.1109/ICETET.2009.70.

[16]. Srinivas, P.G., Reddy, P.K., Trinath, A.V., Bhargav, S., Uday Kiran, R., May (2014). Mining coverage patterns from transactional databases. Journal of Intelligent Information Systems, 45(3), 423-439, doi: 10.1007/s10844-014-0318-3.

[17]. Tang, R., and Zou, Y., September (2010). An Approach for Mining Web Service Composition Patterns from Execution Logs. Web Systems Evolution (WSE) 12th International Symposium on IEEE, 53-62, doi: 10.1109/ICWS.2010.35.

[18]. Tao, Y., Hong, T., Su, Y., April (2008). Web usage mining with intentional browsing data. Expert Systems with Applications, 34(3), pp: 1893-1904, doi:10.1016/j.eswa.2007.02.017.