Web User Session Cluster Discovery Based on *k*-Means and *k*-Medoids Techniques

Zahid Ahmed Ansari
Department of CSE, P.A. College of Engineering
Mangalore, India
zahid_cs@pace.edu.in

Abstract— The explosive growth of World Wide Web (WWW) has necessitated the development of Web personalization systems in order to understand the user preferences to dynamically serve customized content to individual users. To reveal information about user preferences from Web usage data, Web Usage Mining (WUM) techniques are extensively being applied to the Web log data. Clustering techniques are widely used in WUM to capture similar interests and trends among users accessing a Web site. Clustering aims to divide a data set into groups or clusters where inter-cluster similarities are minimized while the intra cluster similarities are maximized. This paper describes the discovery of user session clusters using *k*-Means and *k*-Medoids clustering techniques. These techniques are implemented and tested against the Web user navigational data. Performance and validity results of each technique are presented and compared.

Keywords-web usage mining; k-means clustering; k-medoids clustering

I. INTRODUCTION

Web Usage Mining [1] is described as the automatic discovery and analysis of patterns in web logs and associated data collected as a result of user interactions with Web resources on one or more Web sites. The goal of Web usage mining is to capture, model, and analyse the behavioural patterns and profiles of users interacting with a Web site. The discovered patterns are usually represented as collections of URLs that are frequently accessed by groups of users with common interests. Web usage mining has been used in a variety of applications such as i) Web Personalization systems [2], ii) Adaptive Web Sites [3][4], iii) Business Intelligence [5], iv) System Improvement to understand the web traffic behaviour which can be utilized to decide strategies for web caching [6], load balancing and data distribution [7], iv) Fraud detection: detection of unusual accesses to the secured data [8], etc.

Clustering techniques are widely used in WUM to capture similar interests and trends among users accessing a Web site. Clustering aims to divide a data set into groups or clusters where inter-cluster similarities are minimized while the intra cluster similarities are maximized. Details of various clustering techniques can be found in survey articles [9]-[11]. The ultimate goal of clustering is to assign data points to a finite system of k clusters. Union of these clusters is equal to a full dataset with the possible exception of outliers. Clustering groups the data objects based only on the information found in the data which describes the data objects and the relationships between them.

Some of the main categories of the clustering methods are [12]: i) *Partitioning* methods, that create k partitions of a given data set, each representing a cluster. Typical partitioning methods include k-means, k-medoids etc. In *k-means* algorithm each cluster is represented by the mean value of the data points in the cluster called centroid of the cluster. On the other hand and in *k-medoids* algorithm, each cluster is represented by one of the data point located near the center of the cluster called medoid of the cluster. Leader clustering is also a partitioning based clustering techniques which generates the clusters based on an initially specified dissimilarity measure, ii) *Hierarchical* methods create a hierarchical decomposition of the given set of data objects. A hierarchical method can be classified as being either agglomerative or divisive, based on how the hierarchical decomposition is formed. iii) *Density- based* methods form the clusters based on the notion of density. They can discover the clusters of arbitrary shapes. These methods continue growing the given cluster as long as the number of objects or data points in the "neighborhood" exceeds some threshold. DBSCAN is a typical density-based method that grows clusters according to a density-based connectivity analysis. iv) *Grid-based* methods quantize the data object space into a finite number of cells that form a grid structure. All the clustering operations are performed on the grid structure. v) *Model-based* methods, that discover the best fit between data points given a mathematical model. Mathematical model is usually specified as a probability distribution. Clustering techniques have been widely used for mining web usage patterns from web log data [27][28][29][30][31].

The remainder of the paper is organized as follows. Section II presents a overview of web usage mining using clustering techniques and the underlying concepts. Section III presents each of the *k*-Means and *k*-Medoids clustering techniques in detail along with the underlying mathematical formulations. Section IV describes the experimental results of each technique, followed by a comparison of the results. A brief conclusion is described in Section V.

II. Web Usage Mining Using Clustering

A number of clustering algorithms have been used in Web usage mining where the data items are user sessions consisting of sequence of page URLs accessed and interest scores on each URL page based on the characteristics of user behaviour such as time elapsed on a page or the bytes downloaded [2]. In this context, clustering can be used in two ways, either to cluster users or to cluster items. In user-based clustering, users are grouped together based on the similarity of their web page navigational patterns. In item based clustering, items are clustered based on the similarity of the interest scores for these items across all users. Mobasher et. al. [13], [14] have used both user-based clustering as well as item-based clustering in a personalization framework based on Web usage mining.

A typical user-based clustering starts with the matrix representing the user sessions or user profiles and partitions this multi-dimensional space into k groups of profiles that are close to each other based on a measure of distance or similarity among the vectors (such as Euclidean or Manhattan distance). Clusters obtained in this way can represent user segments based on their common navigational behaviour or interest shown in various URL items. In order to determine similarity between a target user and a user segment represented by the user session clusters, the centroid vector corresponding to each cluster is computed which is the representation of that user segment. To make a recommendation for a target user u and target URL item i, a neighbourhood of user segments that have a interest scores for i and whose aggregate profile is most similar to u are selected. This neighbourhood represents the set of user segments of which the target user is most likely to be a member. Given that the aggregate profile of a user segment that contains the average interest scores for each item within the segment, a prediction can be made for item i using k-nearest-neighbor approach [15].

We map the user sessions as vectors of URL references in a *n*-dimensional space. Let $U = \{u_1, u_2, \ldots, u_n\}$ be a set of *n* unique URLs appearing in the preprocessed log and let $S = \{s_1, s_2, \ldots, s_m\}$ be a set of *m* user sessions discovered by preprocessing the web log data, where each user session $s_\mathrm{i} \in S$ can be represented as $s = \{w_{u_1}, w_{u_2}, \ldots, w_{u_m}\}$. Each w_{u_i} may be either a binary or non-binary value depending on whether it represents presence and absence of the URL in the session or some other feature of the URL. If w_{u_i} represents presence of absence of the URL in the session, then each user session is represented as a bit vector where

$$w_{u_i} = \begin{cases} 1; \text{ if } u_i \in s; \\ 0; \text{otherwise} \end{cases} \quad (1)$$

Instead of binary weights, feature weights can also be used to represent a user session. These feature weights may be based on frequency of occurrence of a URL reference within the user session, the time a user spends on a particular page or the number of bytes downloaded by the user from a page.

III. k-Means and k-medoids based Clustering Techniques

A. k-Means Clustering Algorithm:

The *k*-Means clustering or Hard *c*-Means clustering algorithm [16] is one of the most commonly used methods for partitioning the data. Given a set of *m* data points $X = \{x_i \mid i = 1 \cdots m\}$, where each data point is a *n*-dimensional vector, *k*-means clustering algorithm aims to partition the *m* data points into *k* clusters ($k \le m$) $\boldsymbol{C} = \{c_1, c_2, \ldots, c_k\}$ so as to minimize an *objective function* (or a cost function) *J*(*V*, *X*) of dissimilarity [17], which is the within-cluster sum of squares. In most cases the dissimilarity measure is chosen as the Euclidean distance. The objective function is an indicator of the distance of the *n* data points from their respective cluster centers. The objective function *J*, based on the Euclidean distance between a data point vector x_i in cluster *j* and the corresponding cluster center $v_{j,}$ is defined in (2).

$$J(X,V) = \sum_{j=1}^{k} J_i(x_i, v_j) = \sum_{j=1}^{k} \left(\sum_{i=1}^{m} u_{ij}.d^2(x_i, v_j) \right), \quad (2)$$

where, $J_i(x_i, v_j) = \sum_{i=1}^{m} u_{ij}.d^2(x_i, v_j)$,

is the objective function within cluster c_i,

$u_{ij} = 1$, if $x_i \in c_j$ and 0 otherwise.

$d^2(x_i, v_j)$ is the disatnce between x_i and v_j

Euclidian distance between various data points and cluster centers can be calculated using (3).

$$d^2(x_i, v_j) = \left\| \sum_{k=1}^{n} x_k^i - v_k^j \right\|^2 \tag{3}$$

where, n is the number of dimensions of each data point

x_k^i is the value of k^{th} dimensions of x_i

v_k^j is the value of k^{th} dimensions of v_j

The k-means clustering first initializes the cluster centers randomly. Then each data point x_i is assigned to some cluster v_j which has the minimum distance with this data point. Once all the data points have been assigned to clusters, cluster centers are updated by taking the weighted average of all data points in that cluster. This recalculation of cluster centers results in better cluster center set. The process is continued until there is no change in cluster centers.

The partitioned clusters are defined by a m k binary membership matrix $\boldsymbol{U}$, where the element u_{ij} is 1, if the ith data point x_i belongs to the cluster j, and 0 otherwise. Once the cluster centers $V = \{v_1, v_2, \ldots, v_k\}$, are fixed, the membership function u_{ij} that minimizes (2) can be derived as follows:

$$u_{ij} = \begin{cases} 1; \text{ if } d^2(x_i, v_j) \le d^2(x_i, v_{j*}) \;\; j \ne j*, \forall \; j* = 1, \cdots, k \\ 0; \text{ otherwise} \end{cases} \tag{4}$$

The equation (4) specifies that assign each data point x_i to the cluster c_j with the closest cluster center v_j. Once the membership matrix U=[u_{ij}] is fixed, the optimal center v_j that minimizes (2) is the mean of all the data point vectors in cluster j :

$$v_j = \frac{1}{\left|c_j\right|} \sum_{i, x_i \in c_j}^{m} x_i \tag{5}$$

where,

$\left|c_j\right|$, is the size of cluster c_j and also $\left|c_j\right| = \sum_{i=1}^{m} u_{ij}$

```
Algorithm: k-Means clustering  algorithm for partitioning,
           where each cluster's center is represented by the
           mean value of the data points in that cluster.

Input:     k, the number of clusters and Set of m data points
           X={x1, …, xm}.

Output:    Set of k centroids, V={v1, …, vk}, corresponding to the
           clusters C={c1, …, ck}, and membership matrix
           U=[uij].

Steps:
1)  Initialize the k centroids V={v1, …, vk}, by randomly
    selecting k data points from X.
2)  repeat
      i)   Determine the membership matrix U using (8), by
           assigning each data point xi to the closest cluster cj.
      ii)  Compute the objective function J(X,V) using (6). Stop
           if it below a certain threshold ε.
      iii) Recompute the centroid of each cluster using (9).
3)  until Centroids do not change
```

Figure 1. *k*-Means Clustering Algorithm

Given an initial set of k means or cluster centers, $V = \{v_1, v_2, \ldots, v_k\}$, the algorithm proceeds by alternating between two steps: i) Assignment step: Assign each data point to the cluster with the closest cluster center. ii) Update step: Update the cluster center as the mean of all the data points in that cluster. The input to the algorithm is a set of m data points $X = \{x_i \mid i = 1 \cdots m\}$, where each data point is a n-dimensional vector, it then determines the cluster centers v_j and the membership matrix $\boldsymbol{U}$ iteratively as explained in algorithm show in Fig. 1.

The *k*-means algorithm provides locally optimal solutions with respect to the sum of squared errors represented by the error objective function. Since it is a fast iterative algorithm, it has been applied to a variety of areas [18]-[20]. The attractiveness of the k-means lies in its simplicity and flexibility. However, it suffers from major shortcomings that have been a cause for it not being implemented on large datasets. The most important among these are i) *k*-Means scales poorly with respect to the time it takes for large number of points; ii) The algorithm might converge to a solution that is a local minimum of the objective function. The main disadvantage of this algorithm lies in its sensitivity to initial positions of the cluster centroids [21]. Since the performance of the k-Means algorithm depends on the initial positions of the cluster centeroids, it is recommended to execute the algorithm multiple times, each with a different set of initial centroids.

B. K-Medoids Clustering Algorthm:

k-Medoid is a classical partitioning technique of clustering that clusters the data set of *m* data points into *k* clusters. It attempts to minimize the squared error, which is the distance between data points within a cluster and a point designated as the center of that cluster. In contrast to the *k*-means algorithm, *k*-Medoids algorithm selects data points as cluster centers (or medoids). A medoid is a data point of a cluster, whose average dissimilarity to all the other data points in the cluster is minimal i.e. it is a most centrally located data point in the cluster [20],[22].

Given a set of *m* data points $X = \{x_i \mid i = 1 \cdots m\}$, where each data point is a *n*-dimensional vector, *k*-mdoids clustering algorithm aims to partition the *m* data points into *k* clusters ($k \leq m$) $\boldsymbol{C} = \{c_1, c_2, \ldots, c_k\}$ so as to minimize an objective function representing the sum of the dissimilarities between each of the data points and its corresponding cluster medoid. Let $\boldsymbol{M} = \{m_1, m_2, \ldots, m_k\}$ be the set of medoids corresponding to *C*. The objective function *J(X, M)* is defined in (7)

$$J(X,M) = \sum_{j=1}^{k} \left(\sum_{i=1}^{m} u_{ij} . d^2(x_i, m_j) \right), \qquad (7)$$

where,

x_i is the i^{th} data point

m_j is the medoid of cluster c_j

$u_{ij} = 1$, if $x_i \in c_j$ and 0 otherwise.

$d^2(x_i, m_j)$ is the Euclidean disatnce between x_i and m_j

$$d^2(x_i, m_j) = \left\| \sum_{k=1}^{n} x_k^i - m_k^j \right\|^2 \qquad (8)$$

where, n is the number of dimensions of each data point

x_k^i is the value of k^{th} dimensions of x_i

m_k^j is the value of k^{th} dimensions of m_j

The partitioned clusters are defined by a *m* *k* binary membership matrix ***U***, where the element u_{ij} is 1, if the *i*th data point x_i belongs to the cluster *j*, and 0 otherwise. Once the cluster medoids $M = \{m_1, m_2, \ldots, m_k\}$, are fixed, the membership function u_{ij} that minimizes (7) can be derived as follows:

$$u_{ij} = \begin{cases} 1; \text{ if } d^2(x_i, m_j) \leq d^2(x_i, m_{j*}) \ j \neq j*, \forall \ j* = 1, \cdots, k \\ 0; \text{otherwise} \end{cases} \qquad (9)$$

The equation (9) specifies that assign each data point x_i to the cluster medoid m_j. Once the membership matrix U=[u_{ij}] is fixed, the new cluster medoids m_j that minimizes (7) can be found using (10)

$$m_j = \arg\min_{xi \in c_j} \sum_{x_l \in c_j} d(x_i, x_l) \qquad (10)$$

The basic strategy of *k*-Medoids clustering algorithms is to discover *k* clusters in *m* objects by first arbitrarily selecting a representative data point (the Medoid) as the center for each cluster. Each remaining data point is clustered with the medoid to which it is the most similar. The algorithm takes the input parameter *k*, the number of clusters to be partitioned among a set of *m* objects. The most common realization of *k*-medoid clustering algorithm is described in Fig 2.

It is more robust to noise and outliers as compared to *k*-means because a medoid is less influenced by outliers or other extreme values than a mean. It minimizes the sum of pair-wise dissimilarities instead of a sum of squared Euclidean distances as in case of k-means. Both methods require the user to specify *k*, the number of clusters.

```
Algorithm: $k$-Medoids Clustering
Input:  Set of $m$ data points $X=\{x_1, \ldots, x_m\}$.
Output: Set of $k$ medoids, $M=\{m_1, \ldots, m_k\}$, corresponding to
        the clusters $C=\{c_1, \ldots, c_k\}$, and membership matrix
        $U=[u_{ij}]$ that minimizes the sum of the dissimilarities
        of all the data points to their nearest medoid.
Steps:
4)  Initialize the $k$ medoids $V=\{v_1, \ldots, v_k\}$, by randomly
    selecting $k$ data points from $X$.
5)  repeat
        iv)  Determine the membership matrix $U$ using (9), by
             assigning each data point $x_i$ to the closest cluster $c_j$.
        v)   Compute the objective function $J(X,M)$ using (7). Stop
             if it below a certain threshold $\varepsilon$.
        vi)  Re-compute the medoid of each cluster using (10).
6)  until Medoids do not change
```

Figure 2. *k*-Medoids Clustering Algorithm

IV. Experimental Results

In order to discover the clusters that exist in user accesses sessions of a web site, we carried out a number of experiments using various clustering techniques. The Web access logs were taken from the P.A. College of Engineering, Mangalore web site, at URL http://www.pace.edu.in. The site hosts a variety of information, including departments, faculty members, research areas, and course information. The Web access logs covered a period of one month, from February 1, 2011 to February 8, 2011. There were 12744 logged requests in total.

A. Preprocessing the Web Log Data:

After performing the cleaning operation the output file contained 12744 entries. Total numbers of unique users identified are 16 and the number of user sessions discovered are 206. Table I depicts the results of cleaning and user identification and user session identification steps of preprocessing. Further details of our preprocessing approaches can be found from our previous work [23].

TABLE I
Results of Cleaning and User Identification

Items	Count
Initial No of Log Entries	12744
Log Entries after Cleaning	11995
No. of site ULRs accessed	116
No of Users Identified	16
No. of User Sessions Identified	206

B. Clustering of User Navigational Sessions:

Once the user sessions are discovered, user session data is presented to k-Means and k-Medoids clustering algorithms in order to discover session clusters that represent similar URL access patterns. Since the above clustering algorithms result in different clusters it is important to perform an evaluation of the results to assess their quality. We evaluated our results based on DB index and C Index which are two quality measures to evaluate the quality of the discovered clusters. These validity measures a described below:

Davies-Bouldin Validity Index: This index attempts to minimize the average distance between each cluster and the one most similar to it. It is defined as:

$$DB = \frac{1}{k}\sum_{i=1}^{k}\max_{1\le j\le k,\, j\ne i}\left(\frac{diam\left(c_i\right)+diam\left(c_j\right)}{dis\left(c_i, c_j\right)}\right) \qquad (11)$$

An optimal value of the k is the one that minimizes this index.

C Index: It is defined as [25]:

$$C = \frac{S - S_{\min}}{S_{\max} - S_{\min}}, \qquad (12)$$

Here S is the sum of distances over all pairs of objects form the same cluster. Let m be the number of those pairs and S_{min} is the sum of the m smallest distances if all pairs of objects are considered. Similarly S_{max} is the sum of the m largest distances out of all pairs. The interval of the C-index values is [0, 1] and this value should be minimized. The results of application of various clustering algorithms are presented in the following subsections.

1) *k-Means Algorithm:*

We conducted multiple runs of k-Means algorithm by selecting the input parameter *k* (number of clusters) ranging from k = 2, …, 67. (The value 67 for the number of clusters is one third of total number of the discovered user sessions). For each of these runs we computed the value of the clustering error function (J) using (2) which represents the sum of the squared error. We also computed the execution timings, Dunn's index, DB index and C index for all of the above runs. Table II describes the results after the application of *k*-Means algorithm.

TABLE II
K-MEANS CLUSTERING RESULTS

Clusters	SSE (*J*)	DB Index	C Index	Execution Time(ms)
10	583.54	1.3395	0.1229	49
20	443.06	1.3456	0.1060	110
30	357.24	1.2228	0.0769	142
40	284.08	1.1045	0.0610	164
50	279.29	1.1345	0.0651	278
60	260.64	0.8846	0.0783	188

One of the problems associated with the k-Means algorithm is that it may produce empty clusters depending on the initial centroids chosen. Graph in Fig. 3 Describes the number of empty clusters generated for different values of *k*. M.K. Pakhira [26] has proposed a modified k-means algorithm to avoid the empty clusters. K-medoids algoritm also rectifies this problem.

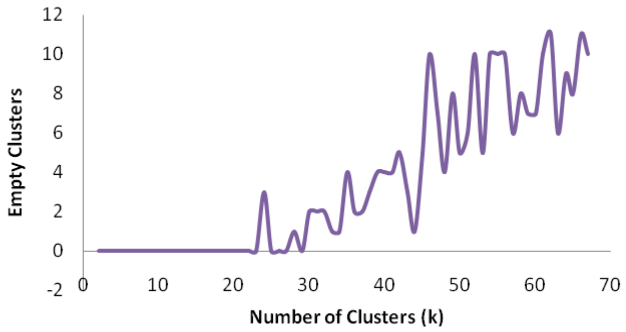

Figure 3. No. of Empty Clusters Vs. No. of Initial Clusters *k*

2) *k-Medoids Algorithm:*

We conducted the multiple runs of *k*-Medoids algorithm by selecting the input parameter *k* (number of clusters) ranging from k = 2, …, 67. (The value 67 for the number of clusters is one third of total number of the discovered user sessions). For each of these runs we computed the value of the clustering error function (J) using (7), which represents the sum of the squared error. We also computed the execution timings, Dunn's index and DB index and C index for all of the above runs. Table III describes the results after the application of *k*-Means clustering algorithm.

TABLE III
K-MEDOIDS CLUSTERING RESULTS

Clusters	Error (*J*)	DB Index	C Index	Execution Time(ms)
10	613.73	1.4426	0.1622	7
20	512.81	1.4689	0.1543	7
30	352.88	1.2018	0.05	5
40	315.63	0.9413	0.23572	6
50	257.83	2.35	0.03	7
60	254.13	2.85	0.06	9

We compared the k-Means and k-Medoids algorithms based on clustering error (*J* as defined in equations (2) and (7)), cluster validity using C index and the execution time.

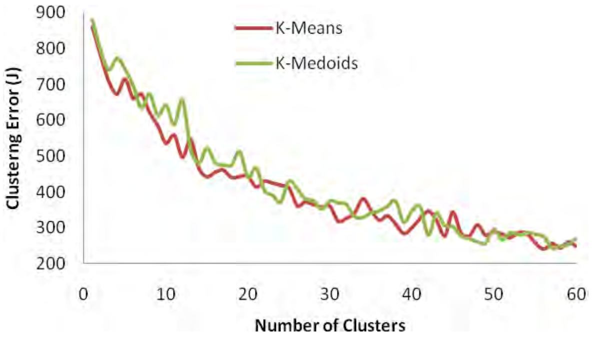

Figure 4. Clustering Error (*J*) Vs. No. of Clusters *k*

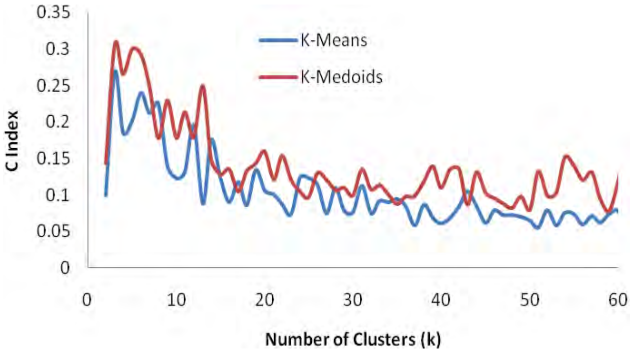

Figure 5. C Index Vs. No. of Clusters *k*

Our results (Fig. 4) show that the k-Means algorithm minimizes the clustering error (*J*) slightly better than the k-Medoids algorithm. C index values in graph plot of Fig. 5 indicates that the clusters of *k*-Means algorithm have better validity index than that of *k*-Medoids algorithm. On the other the execution timings of *k*-Medoids algorithms are faster than the that of *k*-Means algorithm as show in Fig.6.

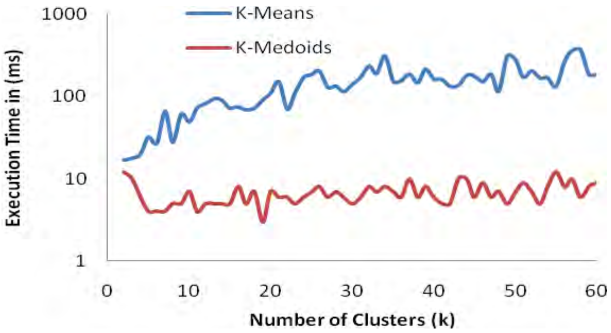

Figure 6. Execution Time in milliseconds Vs. No. of Clusters *k*

V. CONCLUSIONS

In this paper we have presented our framework for web usage data clustering for users' navigational sessions using k-Means and k-Medoids clustering algorithms. We provided a detailed overview of these techniques. We also described the formulation model and algorithmic details related to the implementation of these clustering algorithms in order to discover the user sessions clusters. From the results presented in the previous section, we conclude the following points.

- K-means clustering produces fairly higher accuracy and lower clustering error as compared with k-medoids clustering algorithm.
- K-means algorithm may result in the formation of empty cluster while it is not the case with k-medoids algorithm.
- Our result shows that k-medoids algorithm gives reasonably better time performance than that of the k-means algorithm. The reason behind this is we are using a large data set. The k-Medoids algorithm requires to compute the distance between every pair of data objects only once and uses this distance at every stage of iteration. On the other for an optimal solution k-Means algorithm performs multiple runs and computes the distance between every data object and it's corresponding cluster center.

REFERENCES

[1] J. Srivastava, R. Cooley, M. Deshpande, and P.N. Tan. Web usage mining: Discovery and applications of usage patterns from web data. SIGKDD explorations, 1(2):12–23, 2000.
[2] B. Mobasher. Data mining for web personalization. Lecture Notes in Computer Science, 4321:90, 2007.
[3] Etzioni O. Perkowitz, M. Adaptive web sites: Automatically synthesizing web pages. In Proceedings of the 15th National Conference on Artificial Intelligence, Madison, WI (July1998) 727-732, 1998.
[4] Etzioni O. Perkowitz, M. Adaptive web sites. Communications of ACM, 43:152–158, 2000.
[5] Ajith Abraham. Business intelligence from web usage mining. Journal of Information & Knowledge Management, 2(4):375–390, 2003.
[6] Edith Cohen, Balachander Krishnamurthy, and Jennifer Rexford. Improving end-to-end performance of the web using server volumes and proxy filters. SIGCOMM Comput. Commun. Rev., 28:241–253, October 1998.
[7] Alexandros Nanopoulos, Dimitrios Katsaros, and Yannis Manolopoulos. Exploiting web log mining for web cache enhancement. In WEBKDD 2001 Mining Web Log Data Across All Customers Touch Points, volume 2356 of Lecture Notes in Computer Science, pages 235–241. Springer Berlin / Heidelberg, 2002.
[8] G. Vigna, W. Robertson, Vishal Kher, and R.A. Kemmerer. A stateful intrusion detection system for world-wide web servers. In Computer Security Applications Conference, 2003. Proceedings. 19th Annual, pages 34–43, 2003.
[9] P. Berkhin, "Survey of clustering data mining techniques," Springer, 2002.
[10] B. Pavel, "A survey of clustering data mining techniques," in Grouping Multidimensional Data. Springer Berlin Heidelberg, 2006, pp. 25–71.
[11] R. Xu and I. Wunsch, D., "Survey of clustering algorithms," Neural Networks, IEEE Transactions on, vol. 16, no. 3, pp. 645–678, May 2005.
[12] M. K. Jiawei Han, Data Mining: Concepts and Techniques. Academic Press, Morgan Kaufmarm Publishers, 2001.
[13] B. Mobasher, H. Dai, T. Luo, and M. Nakagawa. Discovery and evaluation of aggregate usage profiles for web personalization. Data Mining and Knowledge Discovery, 6(1):61–82s, 2002.
[14] Bamshad Mobasher, Robert Cooley, and Jaideep Srivastava. Automatic personalization based on web usage mining. Commun. ACM, 43:142– 151, August 2000.
[15] Jonathan L. Herlocker, Joseph A. Konstan, Al Borchers, and John Riedl. An algorithmic framework for performing collaborative filtering. In Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval, SIGIR '99, pages 230–237, New York, NY, USA, 1999. ACM.
[16] J. A. Hartigan and M. A. Wong, "A k-means clustering algorithm, Applied Statistics, 28:100--108, 1979.

[17] Jang, J.-S. R., Sun, C.-T., Mizutani, E., “Neuro- Fuzzy and Soft Computing – A Computational Approach to Learning and Machine Intelligence,” Prentice Hall.

[18] R.O. Duda, P.E. Hart, Pattern Classification and Scene Analysis, Wiley, New York, 1973.

[19] A.K. Jain, R.C. Bubes, Algorithm for Clustering Data, Prentice-Hall, Englewood Cli s, NJ, 1988.

[20] L. Kaufman, P.J. Rousseeuw, Finding Groups in Data. An Introduction to Cluster Analysis, Wiley, New York, 1990.

[21] P.S. Bradley and Usama M. Fayyad: Refining initial points for k-means clustering. In Proceedings Fifteenth International Conference on Machine Learning, pages 91-99, San Francisco, CA, 1998, Morgan Kaufmann.

[22] Jain, A.K., M.N. Murty and P.J. Flynn, Data Clustering: A Review, ACM Computing Surveys, Vol. 31, No. 3, pp. 264- 323, Sep. 1999.

[23] Zahid Ansari, A. Vinaya Babu, Waseem Ahmed and Mohammad Fazle Azeem, “A Fuzzy Set Theoretic Approach to Discover User Sessions from Web Navigational Data”, in International Conference on IEEE Recent Advances in Intelligent Computational Systems, Trivandrum Sep. 22-24 2011, pp. 879-884.

[24] D.L. Davies, D.W. Bouldin. A cluster separation measure. 1979. IEEE Trans. Pattern Anal. Machine Intell. 1 (4). 224-227.

[25] Hubert, L. and Schultz, J. Quadratic assignment as a general data-analysis strategy. British Journal of Mathematical and Statistical Psychology, 29, 190-241, 1976.

[26] M.K. Pakhira, A Modified k-means Algorithm to Avoid Empty Clusters, International Journal of Recent Trends in Engineering Vol 1, No. 1. 2009.

[27] Zahid Ansari , Mohammad Fazle Azeem, A. Vinaya Babu and Waseem Ahmed. “A Fuzzy Clustering Based Approach for Mining Usage Profiles from Web Log Data” International Journal of Computer Science and Information Security, pp. 70-79 Vol. 9, No. 6, June 2011.

[28] Zahid Ansari, Waseem Ahmed , M.F. Azeem and A.Vinaya Babu. “Discovery of Web Usage Profiles Using Various Clustering Techniques”. International Journal of Computer Information Systems, pp. 18-27 Vol. 1, No. 3, July 2011.

[29] Zahid Ansari, A. Vinaya Babu, Waseem Ahmed and Mohammed Fazle Azeem. “A Comparative Study of Mining Web Usage Patterns Using Variants of k-Means Clustering Algorithm”. International Journal of Computer Science and Information Technologies, pp. 1407-1413 Vol. 2 No. 4, July 2011.

[30] Zahid Ansari, A.Vinaya Babu, M.F. Azeem and Waseem Ahmed. “Quantitative Evaluation of Performance and Validity Indices for Clustering the Web Navigational Sessions” World of Computer Science and Information Technology Journal pp. 217-226, Vol. 1, No. 5, June 2011.

[31] Zahid Ansari, M.F.Azeem, A. Vinaya Babu and Waseem Ahmed. “A Fuzzy Approach for Feature Evaluation and Dimensionality Reduction to Improve the Quality of Web Usage Mining Results”. International Journal on Advanced Science Engineering and Information Technology , pp. 67-73 Vol. 2 No. 6, 2012.