3 Cluster Analysis and Recommendation Approaches

The ability to extract useful knowledge hidden in the Web data and to act on that knowledge is an important strategic asset in today’s competitive world. Thus, the emergence of the Web has put forth a number of challenges for knowledge discovery and analysis. Web usage mining can be considered as a special case of Knowledge Discovery in Databases (KDD) [SCDT00, MCS00]. In view of this, the application of data mining techniques such as clustering has been utilised, in this research, in the pattern discovery phase to extract ‘interesting’ patterns and useful knowledge. The discovery of usage patterns will enable organizations to cater to the dynamic needs/interests of Web users and create a personalized experience for visitors by providing appropriate recommendations. This chapter discusses in detail the various steps involved in cluster analysis and provides an overview of the different types of clustering with a special focus on probabilistic clustering. This chapter also extends the use of the resulting discovered patterns to one of the various applications of Web usage mining which includes the recommendation of Web pages. Various approaches to recommendation have been discussed and advocates the use of data mining approach for the purpose of recommendation.

3.1 Pattern Discovery from Web Usage Data

Subsequent to data preprocessing phase is the pattern discovery phase. The resultant user transaction/session file obtained after preprocessing and post-processing (if necessary), is suitable for a variety of analysis of the Web usage data. Different knowledge discovery techniques can be applied to the user transaction file in the pattern discovery phase to obtain useful patterns. One of the approaches to achieve this is to adopt data mining techniques. The various data mining techniques are:
Clustering, Association rule mining, Sequential pattern mining, Classification and Prediction [SCDT00]. From a practical perspective, clustering plays an outstanding role in diverse applications such as scientific data exploration, information retrieval and text mining, spatial database applications, Web analysis, CRM, marketing, medical diagnostics and computational biology [Bert02, PPPS03]. This research focuses on clustering technique in Web usage mining and concentrates on the applications of Web usage mining with respect to recommendations.

3.2 Overview of Cluster Analysis

Basically, classification systems are supervised or unsupervised based on the existence of a finite number of discrete supervised classes (otherwise called class labels) or unsupervised classes respectively. Cluster analysis is a set of methodologies for automatic classification of samples into a number of more or less homogeneous subgroups based on a measure of similarity so that samples within a group are similar and samples belonging to different groups are dissimilar. Clustering or exploratory data analysis is one of the steps in cluster analysis and is a category of unsupervised classification due to the existence of unlabelled data [IMF99, ELL01]. Cluster analysis is a repetitive process which includes a feedback pathway and consists of four steps [XW05] as shown in figure 3.1.

1. Feature selection or extraction: Certain features or attributes may be redundant or irrelevant and may not be important for learning models. Feature selection is a technique of identifying a subset of relevant features from the original set of features. In general, feature selection is an important method for improving the efficiency and accuracy of learning models. Hence, selection of a subset of the original features often leads to a better clustering result. Feature selection not only reduces the high dimensionality of the feature space but also provides
Figure 3.1: Steps involved in Cluster analysis

a better data understanding which improves the clustering result [DL00]. There are many standard data analysis software systems used for feature selection such as MATLAB, SciLab and R language. There are also software systems tailored specifically to the feature selection task such as Weka, RapidMiner and Orange. Feature extraction is a special form of dimensionality reduction which utilizes some transformations on the input data to generate a reduced representation set of features from the original ones [JMF99]. Feature selection and extraction play a vital role to the effectiveness of clustering applications [MDLN02].

2. Clustering algorithm selection: Samples are grouped together on the basis of similarity. Similarity measures constitute a major part of clustering and directly affects the formation of the resulting clusters [JMF99]. The term ‘similarity’ means that the value of $s(x, x')$ is large when $x$ and $x'$ are two similar samples and the value of $s(x, x')$ is small when $x$ and $x'$ are not similar. In some cases, instead of the similarity measure, the dissimilarity between two samples
using a distance measure defined on the feature space may be calculated. Some of the well-known distance measures are the Euclidean distance, Manhattan distance and Minkowski distance. Once the similarity or distance measure is chosen, a criterion function is defined. Clustering algorithms produce clusters by optimizing a criterion function. A number of clustering algorithms have been developed to solve different problems in various fields. However, there is no clustering algorithm that can be universally used to solve all problems. Therefore, it is important to carefully investigate the characteristics of the problem at hand, in order to select or design an appropriate clustering strategy.

3. **Cluster validation**: Cluster validity analysis is the evaluation of the result of a clustering method. Given a data set, a clustering algorithm produces clusters irrespective of whether the structure exists or not. Moreover, different algorithms usually lead to different number of clusters. Hence, evaluation standards and criteria are important for the effectiveness of the clustering results [JMF99]. These standards should justify regarding the number of clusters hidden in the data, whether meaningful clusters are obtained and the selection of some algorithm instead of another. Generally, three categories of validation criteria exist which provide some insights on the quality of clustering solutions namely *external indices, internal indices* and *relative indices* [JD88].

4. **Cluster interpretation**: The outcome of the clustering algorithm helps users to understand the patterns extracted from the original data for various Web-based applications.

### 3.3 A Classification of Major Clustering Algorithms

A vast collection of clustering algorithms are available in the literature. Clustering algorithms are broadly classified based on the properties of the clusters [JD88, ELL01]:

• Hierarchical clustering
  – Agglomerative Algorithms
  – Divisive Algorithms

• Partitioning clustering
  – K-Means algorithm
  – K-Medoids algorithm

• Probabilistic Clustering
  – Expectation Maximization algorithm

• Density-Based Partitioning

• Grid-Based Methods

• Other Clustering Techniques

Most clustering algorithms are basically classified into Hierarchical clustering, Partitional clustering and Probabilistic clustering. An overview of Hierarchical and Partitional clustering has been discussed. This research work concentrates on Probabilistic model-based clustering using EM algorithm.

3.3.1 Hierarchical Clustering

Hierarchical clustering organize data in the form of a nested series of partitions. The input to the system is \((X, s)\) where \(X\) is a set of samples and \(s\) is a measure of similarity. These partitions can be arranged in the form of a dendogram or tree-structure. Every cluster node contains child clusters, facilitating data analysis at different levels of granularity. Hierarchical clustering methods are categorized into
agglomerative (bottom-up) and divisive (top-down). An **agglomerative clustering** starts with each object as an initial cluster and recursively merges two or more most appropriate clusters until all objects are in one large cluster. A **divisive clustering** starts with the entire set of samples $X$ as an initial cluster and recursively divides it into a partition of subsets and repeatedly divides each subset into smaller sets. The process continues until a stopping criterion (the number of clusters, $k$) is achieved. Since computation is expensive for divisive algorithms [ELL01], agglomerative algorithms are more frequently used in real-world applications than divisive algorithms.

At every stage of hierarchical clustering, the splitting or merging is chosen so as to optimize a criterion function. Conventional agglomerative hierarchical methods use heuristic criteria, such as single link (nearest neighbour) or complete link (farthest neighbour). The two criteria differ only in distance computation. In single link, the distance between two clusters is the minimum of the distances between all pairs of samples from the two clusters. In the complete-link algorithm, the distance between two clusters is the maximum of all distances between all pairs of samples from the two clusters.

The advantages of hierarchical clustering include:

- Flexibility regarding the level of granularity
- Ease of handling of similarity or distance computation
- Applicability to any attribute types

The disadvantages of hierarchical clustering are as follows:

- Lack of robustness
- Sensitive to noise and outliers
- Computational costs high for large-scale data sets
3.3.2 Partitional Clustering

While hierarchical algorithms build clusters gradually, partitioning algorithms learn clusters directly. Eventually, clusters are discovered by iteratively relocating points between subsets. Every partitional clustering algorithm obtains partitions which minimizes the within-cluster variation or maximizes the between-cluster variation. Partitional clustering assigns a set of objects into a pre-defined number of clusters \((k)\) without any hierarchical structure. One of the important factors in partitional clustering is the criterion function. The partitional techniques usually produce clusters by optimizing a criterion function defined either locally (on a subset of the patterns) or globally (defined over all the patterns). The sum of square-error function is one of the most widely used criterion. The general objective is to obtain the partition that minimizes the total square-error for a fixed number of clusters.

The drawbacks of partitional clustering algorithm include:

- Initial choice of the number of clusters which affects the clustering result
- Optimal \(k\) value not known
- Lack of scalability
- Sensitive to noise and outliers
- Applicability of only numerical attribute types

The clustering process partitions data into an appropriate number of clusters. Although for some applications, users can determine the number of clusters \((k)\) in terms of their expertise, under many circumstances, the value of \(k\) is unknown and needs to be estimated exclusively from the data themselves. The quality of the resulting clusters is largely dependent on the estimation of \(k\). A division with too many clusters complicates
Chapter 3. Cluster Analysis and Recommendation Approaches

the result, and therefore, makes it hard to interpret and analyze, while a division with too few clusters causes the loss of information and misleads the final decision. The problem of determining the number of clusters was referred to as “the fundamental problem of cluster validity” [JD88].

One of the approaches to estimate the approximate $k$ is the optimization of some criterion functions under probabilistic mixture-model framework.

3.3.3 Probabilistic Clustering

One approach to data partitioning, such as the probabilistic approach, is to represent a cluster mathematically by a parametric probability distribution. In other words, each distribution represents a cluster. The entire data is a mixture of these distributions. The problem is to estimate the parameters of the probability distributions in order to optimize the fit. Each cluster follows a normal or Gaussian distribution with the parameters such as mean and standard deviation. Traditional approaches to this problem involve obtaining, iteratively, a Maximum Likelihood Estimate (MLE) of the parameter vectors. Probabilistic algorithms make use of probabilities to decide clusters as compared to distances in hierarchical clustering. An advantage of probabilistic methods is the interpretability of the constructed clusters.

Probabilistic clustering has some important features:

- Since the mixture model has clear probabilistic foundation, the determination of the most suitable number of clusters becomes a more traceable task
- It can be modified to handle recodes of complex structure
- It can be stopped and resumed with consecutive batches of data, since clusters have representation totally different from sets of points
• At any stage of iterative process the intermediate mixture model can be used to assign cases (on-line property)

• It results in easily interpretable cluster system

Expectation Maximization Algorithm

The EM algorithm, which is an extension of the k-means partitioning algorithm, is a two-step iterative refinement/optimization algorithm that can be applied to the problem of parameter estimation [DLR77]. Unlike k-means algorithm in which an object is assigned to a cluster based on the cluster mean, EM assigns each object to a cluster according to a weight representing the probability of membership. Therefore, new means are computed based on weighted measures. Partitions are determined by the EM algorithm for Maximum Likelihood. The EM procedure begins with an initial estimate of the parameter vector and iteratively recomputes the parameters (or clusters) based on the Expectation step and Maximization step. The recomputed patterns are then used to update the parameter estimates.

• **Expectation step**: Assign each object \( x_i \) to cluster \( C_k \) with the probability as shown in equation 3.1:

\[
P(x_i \in C_k) = p(C_k|x_i) = \frac{p(C_k)p(x_i|C_k)}{p(x_i)}
\]  

(3.1)

where \( p(x_i|C_k) = N(m_k, E_k(x_i)) \) follows the normal (i.e. Gaussian) distribution around mean, \( m_k \), with expectation, \( E_k \). This step calculates the probability of cluster membership of object \( x_i \), for each of the clusters. These probabilities are the “expected” cluster memberships for object \( x_i \).

• **Maximization step**: Use the probability estimates from above to re-estimate (or
refine) the model parameters as shown in equation 3.2

\[
m_k = \frac{1}{n} \sum_{i=1}^{n} \frac{x_i P(x_i \in C_k)}{\sum_j P(x_i \in C_j)}
\]  (3.2)

This step is the “maximization” of the likelihood of the distributions given the data. This results in finding parameters that maximize log-likelihood. The process continues until log-likelihood convergence is achieved. Log-likelihood serves as the objective function. The value of \( k \) that maximizes the defined criterion is regarded as optimal. To summarize, EM assigns a probability distribution to each instance which indicates the probability of it belonging to each of the clusters. EM can decide the number of clusters to create by cross-validation which randomly divides data into training and test sets \( M \) times according to a certain fraction or it can be decided apriori how many clusters to generate. The cross-validation performed to determine the number of clusters is done in the following steps:

1. The number of clusters is initially set to 1
2. The training set is split randomly into 10 folds
3. EM is performed 10 times using the 10 folds in the cross-validation way
4. The log likelihood is averaged over all the 10 results
5. If log likelihood has increased, the number of clusters is increased by 1 and the program continues at step 2

The EM algorithm for clustering has a few limitations [FR98]:

- The rate of convergence can be very slow. This does not appear to be a problem in practice for well-separated mixtures when started with reasonable values.
• The number of conditional probabilities associated with each observation is equal to the number of components in the mixture and as a result, the EM algorithm for clustering may not be practical for models with very large numbers of components.

One of the major phases in cluster analysis is the evaluation of the clustering result which may be characterized as a “good” clustering result or a “poor” clustering result. In general, all clustering algorithms produce clusters. Nevertheless, some clustering algorithms produce “better” clusters than others. One such assessment of a clustering algorithm result is the cluster validity analysis. Validity assessments are objective and are performed to determine whether the output is meaningful.

3.4 Evaluation Measures of Cluster Validity

The evaluation measures of cluster validity have been discussed in detail by Tan et al. and are classified into three types [TSK06]:

1. **Unsupervised:** Unsupervised measures are often called *internal* indices because they use only information present in the data set. They measure the goodness of a clustering structure with respect to internal information and not dependent on prior knowledge or external information. An example of this is SSE (Sum of Squared Error). Unsupervised measures of cluster validity are further divided into two classes:

   • **Cluster cohesion:** It determines how closely related the objects within a cluster are.

   • **Cluster separation:** It evaluates how distinct or well-separated a cluster is from other clusters.
2. **Supervised**: Supervised measures are often called *external* indices because they use information which are not present in the data set. They are dependent on external information or prior information on the data. They measure the extent to which the clustering structure discovered by a clustering algorithm matches some external structure.

- **Entropy**: The extent to which a cluster contains objects of a single class. For each cluster, the class distribution $p_{ij}$ of the data is calculated initially. The probability that a member of cluster $i$ belongs to class $j$ is computed using equation 3.3

$$p_{ij} = \frac{m_{ij}}{m_i}$$

(3.3)

where $m_i$ is the number of objects in cluster $i$ and $m_{ij}$ is the number of objects of class $j$ in cluster $i$. Using this class distribution, the entropy of each cluster $i$ is calculated using equation 3.4

$$e_i = -\sum_{j=1}^{L} p_{ij} \log_2 p_{ij}$$

(3.4)

where $L$ is the number of classes. The total entropy for a set of clusters is calculated as the sum of the entropies of each cluster weighted by the size of each cluster given by equation 3.5

$$e = \sum_{i=1}^{K} \frac{m_i}{m} e_i$$

(3.5)

where $K$ is the number of clusters and $m$ is the total number of data points.

- **Purity**: This measures the extent to which a cluster contains objects of a
single class. The purity of a cluster $i$ is calculated using equation 3.6

$$p_i = \max_j p_{ij}$$  \hspace{1cm} (3.6)$$

The overall purity of a clustering is given by equation 3.7

$$purity = \sum_{i=1}^{K} \frac{m_j}{m} p_i$$  \hspace{1cm} (3.7)$$

- **Precision:** The fraction of a cluster that consists of objects of a specified class. The precision of cluster $i$ with respect to class $j$ is given by equation 3.8

$$precision(i, j) = p_{ij}$$  \hspace{1cm} (3.8)$$

- **Recall:** The extent to which a cluster contains all objects of a specified class. The recall of cluster $i$ with respect to class $j$ is given by equation 3.9

$$recall(i, j) = \frac{m_{ij}}{m_j}$$  \hspace{1cm} (3.9)$$

where $m_j$ is the number of objects in cluster $j$.

- **F-measure:** A combination of both precision and recall that measures the extent to which a cluster contains only objects of a particular class and all objects of that class. The F-measure of cluster $i$ with respect to class $j$ is given by the equation 3.10

$$F(i, j) = \frac{(2 \times precision(i, j) \times recall(i,j))}{precision(i, j) + recall(i, j)}$$  \hspace{1cm} (3.10)$$

3. **Relative:** A relative cluster evaluation measure is a supervised or unsupervised that is used for the purpose of comparison. This measure compares different clustering structures to determine the clustering algorithm which reveals the
Chapter 3. **Cluster Analysis and Recommendation Approaches**

characteristics of the objects to a greater extent.

Therefore, based on the goals of the analyst and applications, various algorithms can be applied for cluster analysis. Probabilistic clustering has been proposed as a basis for cluster analysis. Cluster analysis includes a series of steps, ranging from preprocessing and algorithm development to validity and evaluation of the clusters. Each of them is tightly related to each other and poses great challenges to the scientific disciplines:

- Usually, algorithms are designed with certain assumptions and support some type of biases. In this context, it is not precise to pronounce which clustering algorithm is the best even though a few comparisons are possible. These comparisons are mostly based on some specific applications, under certain conditions, and the results may be quite different if the conditions change.

- At the preprocessing and post-processing phase, feature selection/extraction and cluster validation are equally important as the clustering algorithms. Choosing appropriate and meaningful features can greatly reduce the burden of subsequent change in designs.

### 3.5 Recommendation Approaches

The ultimate goal of any Web-based system is to cater to the needs or interests of a particular user or a segment of users. Hence, in the context of the Web, one of the applications of Web usage mining includes automatic recommendation/personalization. Web recommendation can be described as any action that makes the Web experience of a user customized to the user’s taste or preferences. The tremendous growth in the number and the complexity of information resources and services on the Web has made Web recommendation an indispensable tool for both Web-based organizations and for the end users. The ability of a recommendation system is to recommend
items/pages that the current user requires based on previous usage profiles created. Recommendation systems have become an important research area since 1990 [SM95] which have been used in diverse applications such as recommendation of books and CDs [LSY03], Web pages [MDLN02, MCS00], movies [n09] and news.

Conventional approaches to automatic personalization/recommendation consist of rule-based filtering systems, content-based and collaborative filtering. Each approach differs by the precise nature of data collected for the derivation of user profiles and also by the type of algorithmic approach used to obtain personalized content. In general, recommendation process is sub-divided into two phases such as the data collection phase and learning phase. In the first phase, the information related to the user’s interests are obtained. Based on this, the learning phase is involved in the construction of user profiles. Learning from data can be categorized into two types: memory based (also called as lazy) learning and model based (or eager) learning. This classification depends on whether the learning is done online while the system is performing the recommendation tasks or offline using training data.

Memory-based systems, that perform the learning online, memorize the data which are generalized during recommendations. As a result of this, they are prone to scalability issues. Model-based approaches, that perform the learning phase offline, generally, are likely to scale better than memory based systems. On the other hand, as more data is collected, memory based systems are more capable of reorienting themselves when user’s interests change as compared to model based techniques since the model must either be incremental or be rebuilt when new data is added.

### 3.5.1 Rule-Based Recommendation Systems

Rule-based recommendation systems depend on decision rules that are generated manually or automatically by Web site administrators based on demographic, psychographic
or other personal characteristics of users. These rules control the content provided to a user whose profile matches one or more rule conditions which can be utilised for recommendation. However, this type of recommendation is dependant on the proper construction of the rules with respect to the specific characteristics of the domain. The user profiles are generally obtained through explicit interactions with users. Research in this area, has also focused on machine learning techniques for classifying users based on their demographic attributes, and derive decision rules automatically that can be used for recommendation.

The major drawback of rule-based recommendation techniques lies in the type of input which is usually a subjective description of users or their interests provided by the users themselves, and thus is prone to bias which may affect the user profiles.

3.5.2 Content-Based Recommendation Systems

In content-based recommendation systems, the information about each item $i$ is represented in the form of a feature vector $x_i$. For Web documents, the content descriptions are textual features extracted from Web pages. These systems often rely on well-known document modelling techniques in information retrieval and information filtering research. The items are represented as weighted term vectors based on tf-idf [Sal88] term-weighting model.

For each user $u$, a user profile vector $p_u$ is obtained from the textual features. The importance $s(u, i)$ of item $i$ for user $u$ is estimated based on the importance $s(u, i_j)$ assigned by user $u$ to items $i_j \in I$ (where ‘I’ is the set of items) that are “similar” to item $i$. In [Lan95, BS97], user profiles are computed using the Rocchio algorithm. The user profiles can then be used to recommend new items to a user $u$ that have a high degree of similarity with the user’s current preferences, by suggesting the item whose feature vector $x_i$ is most similar to the user profile vector $p_u$, for example,
using the cosine similarity [DK11, AM07] or using probabilistic approaches such as Bayesian classification [PB97]. For example, in the classification of Web pages, the Naïve Bayesian classifier is used to estimate the probability that page $p_j$ belongs to a pre-defined class $C_i$, given the set of keywords $k_{1,j}, ..., k_{n,j}$ on that page:

$$P(C_i|k_{1,j}, ..., k_{n,j})$$

Therefore, for each page $p_j$, the probability $P(C_i|k_{1,j}, ..., k_{n,j})$ is computed for each class $C_i$ and page $p_j$ is assigned to class $C_i$ having the highest probability.

The chief drawback of content-based filtering systems is to focus mainly on the choice of items since profiles are formed depending on the user’s previous rating of items. In addition to this, content-based filtering necessitates that items can be represented effectively using extracted textual features which may not always be possible due to the heterogeneous nature of Web data.

### 3.5.3 Collaborative Filtering Systems

Some of the shortcomings of the above approaches have been dealt with in the Collaborative approach. Collaborative recommender systems estimate the importance of items for a particular user based on the items previously rated by other users. For instance, to recommend Web pages in a Web site to user $u$, collaborative recommender system tries to identify other users who have similar tastes in the same Web pages and recommends only those Web pages that are preferred by the “peers” of user $u$. Unlike content-based recommendation methods, collaborative recommender systems estimate the importance $s(u, i)$ of item $i$ for user $u$ based on the importance $s(u_j, i)$ assigned to item $i$ by those users $u_j \in U$ (where ‘U’ is the set of users) who are “similar” to user $u$.

Normally, the primary technique employed to achieve this task is the standard
memory-based k-Nearest Neighbour (kNN) classification approach which compares a current user’s profile with the historical profiles of other users in order to find the top ‘k’ users who have similar tastes or interests [WR09, AM07, KJN08].

In the context of recommendation based on Web usage mining, kNN involves determining the similarity between the active session, \( \vec{s} \) and each transaction vector \( \vec{t}, t \in T \) (where ‘T’ is the set of transactions). The top ‘k’ most similar transactions to \( \vec{s} \) are considered to be the neighbourhood for the session ‘s’, denoted by \( NB(s) \).

\[
NB(s) = \{ \vec{t}_{s1}, \vec{t}_{s2}, ..., \vec{t}_{sk} \}
\]

There exists a variety of similarity measures in order to identify the nearest neighbours. In conventional collaborative filtering approaches, the Pearson correlation coefficient is generally used. However, this measure may not be suitable for clickstream data. Alternatively, the cosine coefficient, commonly used in information retrieval is used which measures the cosine of the angle between two vectors. Given the active session \( \vec{s} \) and a transaction \( \vec{t} \), the similarity between them is obtained by

\[
sim(\vec{t}, \vec{s}) = \frac{\vec{t} \cdot \vec{s}}{|\vec{t}| \times |\vec{s}|}
\]

Both the content-based and the collaborative approaches use the same cosine measure from information retrieval. However, in content-based recommender systems, it is used to measure the similarity between vectors of tf-idf weights, whereas, in collaborative systems, it measures the similarity between vectors of the actual user-specified ratings.

To recommend items that have not been visited by the user in the active session, a recommendation score is computed for each item/pageview \( p_i \in P \) (where ‘P’ is the set of pageviews) based on the neighbourhood for the active session. Initially, the
mean vector (centroid) of NB(s) is computed. The value of each pageview in the mean vector is computed by finding the ratio of the sum of the pageview’s weights across transactions to the total number of transactions in the neighbourhood. This is denoted by cent(NB(s)) [Mob04]. For each pageview $p$ in the neighbourhood centroid, the recommendation score is computed as a function of the similarity of the active session to the centroid vector and the weight of that item in this centroid [Mob04]. The function is denoted by $rec(\vec{s}, p)$.

$$rec(\vec{s}, p) = \sqrt{\text{weight}(p, NB(s)) \times \text{sim}(\vec{s}, \text{cent}(NB(s)))}$$

Collaborative filtering has been known to be the most successful recommendation technique that has been used in a number of different applications such as recommending Web pages, movies, articles and products. However, despite their success, collaborative filtering techniques have their own drawbacks. One of the drawbacks is their lack of scalability. Basically, the identification or learning of ‘similar’ neighbours is performed as an online process compared to the model-based approaches in which the learning is performed off-line from training data. As the number of users and items increase, this approach may lead to undesirable long computation time for providing recommendations or dynamic content during user interaction. Another limitation of kNN-based techniques is related to sparsity of the dataset. The number of ratings already obtained is very small compared to the number of ratings that need to be predicted because typical collaborative filtering requires explicit non-binary user ratings for similar products. As a result, collaborative filtering based recommendations cannot accurately compute the neighbourhood and identify the products to recommend [Mob07a].

Optimization strategies have been proposed and employed to overcome these shortcomings. Some of the strategies include offline clustering of user sessions,
Chapter 3. **Cluster Analysis and Recommendation Approaches**

requiring the online, active user session to search only within a matching cluster. In addition, similarity indexing and dimensionality reduction may be performed to minimize real-time search costs and sparsity problems for effective results in pattern discovery thereby resulting in precise recommendations.

### 3.5.4 Data Mining Approach to Recommendation

Data mining approach to Web recommendation is different from the other traditional approaches due to several reasons. Web usage mining is not an explicit algorithm, but rather it performs the typical data mining cycle. In addition, it provides a great deal of flexibility for the recommendation tasks to be better incorporated with other existing applications. Furthermore, since data mining is focused on efficient probability-based pattern discovery algorithms, recommendation systems based on data mining tend to be more scalable than those based on customary approaches such as standard collaborative filtering. The quality of the recommendations has an important effect on the user’s future preferences.

The data mining approaches to recommendation can be regarded as extensions of collaborative filtering. In these approaches, the input to the pattern discovery algorithms is the navigational patterns of past users and aggregate usage models are generated. The usage models, in turn, can be used, in combination with the profile of an active user, to obtain matching profile(s) for the purpose of providing recommendations. Various researchers have suggested Web usage mining as an enabler to overcome the problems related to the traditional techniques since it will reduce the need for obtaining subjective user ratings or registration-based personal preferences [MCS00, MDL+00]. Recommendation systems based on Web usage mining rely on clickstream and navigational data automatically collected by Web and application servers and stored in server log file. Through examining such Web usage information,
it is possible to make a more accurate analysis of user’s interest or preference across all Web pages in the Web site(s). In this research work, recommendations of Web pages using data mining approach has been proposed.

The advantages and flexibilities afforded by the data mining approach to recommendation come precisely from the fact that recommendation is viewed as a holistic process rather than as individual algorithms or specific data types. The focus of this research study is primarily on Web usage mining where the goal is to leverage data collected as a result of user interactions with the Web in order to learn user models and to use these models for recommendation. The results of the mining phase are transformed into aggregate usage models, suitable for use in the recommendation phase. The recommendation engine considers the active user’s profile along with the discovered patterns to provide suitable recommendations.

Recommendation approaches, including those used in the pattern discovery phase of Web usage mining, employ a two stage process for recommendation generation. The first stage is carried out offline, where user transaction data collected during previous interactions is mined and explicit aggregate usage profiles are generated for use in future interactions. The second stage, is carried out online as a new visitor begins an interaction with the Web site.

3.6 Summary

The pattern discovery phase is a crucial offline phase in Web usage mining process. Various techniques can be employed to derive patterns from the user’s navigation data available in the session file. However, the use of unsupervised learning approach such as clustering through EM algorithm has been suggested as an ideal technique to obtain usage profiles. It has been learnt from the literature that the clustering process would definitely result in certain number of clusters, whether ‘good’ or ‘bad’. To prove the
effectiveness of clustering, measures have been adopted to evaluate the clusters.

The discovered usage patterns can be utilised for applications such as recommendation of Web pages to new, online users. This requires the matching of the user’s navigational patterns to the previously discovered profiles in order to assign the new user to the appropriate cluster and, henceforth, dynamically recommend pages of interest. Though various traditional recommendation approaches exist in the literature, this study has proposed the use data mining approach using clustering for recommendation of Web pages. The use of data mining techniques for recommendation systems will result in reasonable improvement in scalability and computational performance since most of the computational time is spent during the offline clustering phase.