CHAPTER 2

LITERATURE REVIEW

Knowledge discovery in databases is referred as data mining. As a consequence of the increase in the size of databases, data mining has become a very important research area. Many techniques from different areas like machine learning, statistics, neural networks and genetic algorithms are used in data mining to extract implicit information from the huge amount of data. Prediction, classification, identification and optimization are some of the goals of data mining. Association rules, sequential patterns, clusters and classification are the different forms of knowledge discovered by data mining.


The useful information is extracted from the Web access logs by successfully utilizing the different forms of data mining techniques (Mobasher et al 1999, Nasraoui et al 1999, Perkowitz and Etzioni 1998). The clients or data items that have similar characteristics can be grouped together using one of the data mining techniques called clustering. Clustered data are one of the most important types that involve a wide range of applications from real time personalization to link prediction. It can facilitate the development of future marketing strategies, such as automated return mail,
present advertisements to clients falling within a certain cluster, or
dynamically change a particular site for a client on a return visit, based on
past classification of that client. The key problem lies in how effectively
clusters of Web pages or users with common interest are discovered.

There is a lot of difference between clustering analysis to mine the
web and traditional clustering. This difference is because of the difference
between web usage data clustering and classic clustering. So, to perform
clustering analysis based on web usage data, some specialized techniques
need to be introduced. Some approaches to clustering analysis have been
developed for mining the Web access logs. Perkowitz and Etzioni (1998)
discussed adaptive web sites that learn from user access patterns. The clusters
of related but unlinked pages are found using the page co-occurrence
frequencies with the help of the PageGather (Perkowitz and Etzioni 1998)
algorithm. A graph is created with nodes which are pages and whose edge
weights are page co-occurrence frequencies. Cliques or connected
components in this graph are used to find clusters. Easier navigation is
possible by creating new index pages based on the algorithm. Mobasher et al
(1999) proposed a technique for capturing common user profiles based on
association rule discovery and usage-based clustering. This technique directly
computes overlapping clusters of URL references, based on their co-
ocurrence patterns across user transactions.

Hyperedges are found by the apriori algorithm as frequent itemsets
using which a hypergraph is obtained. The average of all the confidences of
association rules in this frequent itemset gives the weight of a hyperedge. The
hypergraph partitioning algorithm is used to obtain clusters by applying the
algorithm on hypergraph. Nasraoui et al (1999) defined the similarity measure
between sessions using a modified cosine angle and the similarity measure
that takes the hierarchical structure of URL into consideration. Relation Fuzzy
C-Maximal Density Estimator (RFC-MDE) algorithm is used for clustering the sessions based on the pair-wise dissimilarities between sessions. Session clusters are made based on all the user sessions using the above approaches. The difference between this approach and other clustering algorithms is that it finds user clusters. The common access pattern is found for each group of users by separating the users into different groups.

A general solution for cluster-based service to achieve high performance is provided by socket cloning. Socket cloning is a new network support mechanism for cluster-based web servers and aims at resolving the aforementioned problems. The web server software is allowed to move opened sockets between the cluster nodes efficiently using this approach. The load balancing and high performance can be achieved by processing a request in the web server node with the document in its cache without the need to transfer the cached copy (Sit et al 2002). The apriori algorithm (Agarwal and Shafer 1996) was shown to be superior among all the earlier extant solutions (Holsheimer et al 1995, Park et al 1995). The itemset lattice is pruned by Apriori algorithm based on the downward closure property. Downward closure property of an itemset is the property that all subsets of a frequent itemset must themselves be frequent. So, to construct the candidate (K+1) itemsets, only the frequent K-itemsets are used. Support information is found by generating and counting each node explicitly (Agarwal and Srikant 1994).

Apriori property called downward property proposed in association rule mining (Agarwal and Srikant 1994) is very important based on which most of the basic and earlier algorithms for sequential pattern mining are developed. A series of apriori-like algorithms have been proposed based on this heuristic: AprioriAll, AprioriSome, DynamicSome in (Agarwal and Srikant 1995), GSP (Srikant and Agarwal 1996) and SPADE (Zaki 2001) which is a lattice based algorithm, MEMISP (Lin and Lee 2002) is a memory
indexing based approach, while SPIRIT (Garofalakis et al 1999) integrates constraints by using regular expression. I/O is minimized by scanning the database twice in the partition algorithm (Savasere et al 1995). The first scan of the database is for generating a set of potential frequent itemsets and the second scan is for gathering their support. I/O overhead can also be minimized if the work is carried out on a small sample of the database (Toivonen 1996, Zaki et al 1997). A number of parallel algorithms have also been proposed (Agarwal and Shafer 1996, Zaki et al 1996, Zaki et al 1997). Some new algorithms are proposed in the later stage which scan the database only once for fast association mining (Zaki et al 1997).

MaxMiner (Bayardo 1998) is another algorithm for finding the maximal elements. The search space is quickly narrowed by using efficient pruning techniques. The effectiveness of superset-frequency pruning is increased by employing the item reordering heuristic technique. In this algorithm, breadth-first traversal is used to traverse the search space and lookaheads are used to prune out the branches of the tree (Bayardo 1998). The lookaheads involve superset pruning using apriori in reverse. Actually, lookaheads` performance is better with a depth first traversal when compared to breadth first traversal. In order to reduce the number of scans over the database, the MaxMiner (Bayardo 1998) uses the breadth first traversal. The number of scans over the database using Apriori algorithm and MaxMiner is the same as this algorithm also uses only the original horizontal format of the database.

Long itemsets are determined by DepthProject (Agarwal et al 2000) algorithm by performing the mixed depth-first traversal of the tree along with the variations of the superset pruning. Here, a dynamic reordering of children nodes is used instead of a pure depth first traversal in DepthProject to reduce the search space by trimming the infrequent items out of each node’s tail. The
size of the database is reduced by improved counting method and a projection mechanism. This is one of the advantages of DepthProject. It required post-pruning to eliminate non-maximal patterns and returns a superset of the Maximal Frequent itemsets (MFI). MaxClique and MaxEclat (Zaki 2000) used the vertical format representation of the database. These algorithms divide the subset lattice into smaller pieces referred as “cliques” and mine these cliques in bottom-up apriori fashion. However, both the algorithms rely on a pre-processing step that limits future mining flexibility.

Pincer-Search also assumes that a pre-processing step has taken place before the algorithm is executed (Lin and Kedem 1998). The data format used in the Pincer-search is a horizontal data format. The candidates are constructed in the bottom-up approach like Apriori. A candidate set of maximal patterns is maintained by starting the top-down search in parallel. Eliminating the non-maximal sets at the early stage helps in reducing the number of scans on the database. In general, the overhead of maintaining the maximal candidate set which is a superset of the maximal patterns can be very high. The VIPER algorithm, a method based on a vertical layout, can sometimes outperform even the optimal method using a horizontal layout (Shenoy et al 2000). During the execution of the algorithm, the intermediate data are stored using the vertical bit vector with compression while counting is performed using a vertical tid-list approach. The drawback of the VIPER algorithm is that it cannot be used to generate the MFI set when the patterns are very long. But it returns the entire set FI. Holsheimer (1995) and Savasere et al (1995) presented the other vertical mining methods for finding frequent itemsets (FI). Ganti et al (2000) explored the benefits of using the vertical tid-list. Dunkel et al (1999) discussed the performance issues based on the different database representations. He has also given the analysis report and the impact of the representation on performance.
Dynamic weight is used for each item in frequent pattern mining (Chowdhury FA et al 2008). As the weight (price or significance) changes with time, the authors used variable weight in mining frequent patterns. The authors proposed DWFPM (dynamic weight frequent pattern mining) to address the variation of weight for each item dynamically. To avoid level-wise candidate set generation-and-test methodology, pattern growth mining technique is used. The property of one database scan enables it for use in stream data mining. The authors also proved that the DWFPM is efficient with the performance analysis and also the proposed algorithm is scalable for WFP mining. Kun-Ming et al (2010) proposed two parallel mining algorithms Tidset-based Parallel FPtree (TPFP-tree) and Balanced Tidset-based Parallel FP-tree (BTP-tree) for frequent pattern mining on PC Clusters and multi-cluster grids. Tidset is used to select the transaction effectively when compared to the database scan. The loading is balanced according to the computing ability of the processors by the grid system in BTP-tree. When the database size is increased, results show that the execution time required for TPFP-tree is less than PFP-tree on PC cluster. Moreover, the BTP-tree shortened the execution time significantly and had a better load balance capability than both the TPFP-tree and PFP-tree on a multi-cluster grid.

In sequential pattern mining, the generalized sequential pattern (GSP) algorithm (Srikant and Agarwal 1996) mines sequential patterns based on an apriori-like approach. It generates and tests all candidate subsequences with multiple scans of the original sequence database. FreeSpan (Han et al 2000) is developed to overcome this problem. FreeSpan (Han et al 2000) utilizes an initial projection growth-based approach. The main idea is to use frequent items to recursively project sequence databases into a set of fewer projected databases and to grow subsequence fragments in each projected database. FreeSpan outperforms the apriori-based GSP algorithm. However, FreeSpan
may generate any substring combination in a sequence, and the projection in FreeSpan keeps all the sequences in the original sequence database without length reduction. PrefixSpan (Pei et al 2001, Pei et al 2004), a more efficient pattern growth algorithm, improves the mining process. The main idea of PrefixSpan is to examine only the prefix subsequences and to project only their corresponding suffix subsequences into projected databases. Sequential patterns are grown by exploring only local frequent patterns in each projected database.

In the sequential pattern discovery, a vertical id-list data format was implemented and frequent sequence enumeration is performed by a simple join on id lists using equivalence classes (SPADE) algorithm (Zaki 2001). The extension of vertical format based frequent pattern mining can also be considered as the SPADE algorithm. Depth first traversal of the search space is combined with a vertical bitmap representation to store each sequence in the sequential pattern mining (SPAM) algorithm (Ayres et al 2002). Efficient sequential pattern mining algorithms (Chiu et al 2004) such as constraint-based sequential pattern mining (Garofalakis et al 1999, Seno and Karypis 2002), approximate sequential pattern mining with a weighted sequence structure (Kum et al 2003), sequential pattern mining without using support thresholds (Tzvetkov et al 2003), and closed sequential pattern mining (Wang and Han 2004, Yan et al 2003) have been developed. These approaches may mine patterns efficiently and reduce the number of patterns. In most of the previous sequential pattern mining algorithms, sequential patterns and items within sequential patterns have been treated uniformly, but real sequences differ in their importance.
The weighted sequential pattern mining (WSpan) algorithm (Yun and Leggett 2006) and weighted frequent pattern mining (Yun and Leggett 2005a, Yun and Leggett 2005b, Yun 2007) have been suggested. In WSpan, different weights are associated with different items within a sequence in the sequence database. When weights are associated with the items in the sequence, the most important anti-monotone property (Agarwal and Srikant 1995) fails. This is one of the drawbacks of the WSpan. In other words, super pattern may be weighted frequently even though a sequential pattern is weighted infrequently. This is because super patterns of the sequential pattern, with a low weight, can get a high weight after adding other items or itemsets with higher weights. With the prefix projected sequential pattern growth method (Pei et al 2001, Pei et al 2004), WSpan uses approximate weighted support within normalized weights to prune weighted infrequent sequential patterns but maintains the anti-monotone property. Even if WSpan can effectively identify weighted frequent sequential patterns, it cannot detect sequential correlated patterns with support/weight affinity. Using the framework of WSpan, it is imperative to study the problem of mining sequential affinity patterns with similar weight and/or support levels. One efficient strategy is to integrate w-confidence/s-confidence into the sequential pattern mining algorithm and to prune uninteresting patterns with weak affinity. The traditional methods are not effective in case of Business to Business (B2B) applications as they generate many uninteresting and meaningless patterns and a long computational time. To overcome this problem, an algorithm has been proposed to discover frequent sequential patterns which satisfy compactness, repetition and regency (Ya-Han Hu et al 2009). The authors proved that the proposed algorithm is effective and efficient in extracting useful patterns in the B2B environment.
The rough set perspective algorithm is proposed to the problem of constraint driven mining of sequential pattern (Jigyasa B et al 2009). Indiscernibility relation from theory of rough sets is used to partition the search space of sequential patterns. Also, the authors proposed an algorithm that allows pre-visualization of patterns and imposition of various types of constraints in the mining task. The authors proved that the proposed algorithm C-Rough Set Partitioning is at least 10 times faster than the SPRINT that is based on various regular expression constraints. Most of the current work on the traditional apriori algorithm (Agarwal et al 1993) makes use of the “large – support” metric framework. However, these works still view items as having equal weights though trying to distinguish them using various methods. Wang et al (2000) proposed an efficient mining methodology for Weighted Association Rules (WAR). The idea is inspired by the fact that a numerical attribute can be assigned for every item which in turn judges the weight of the item in a particular weight domain. For example, soda[4,6] → snack [3,5] is a targeted weighted association rule meaning that if a customer purchases soda in the quantity between 4 and 6 bottles, he is likely to purchase 3 to 5 bags of snacks.

WAR uses a two-fold approach where the frequent itemsets are generated through standard association rule mining algorithms without considering weight. Post-processing is then applied on the frequent itemsets during rule-generation to derive the maximum WARs. WAR doesn’t interfere with the process of generating frequent itemset. Rather, it focuses on how weighted association rules can be generated by examining the weighting factors of the items included in generated frequent itemsets. Therefore, it is important to classify this type of weighted association rule mining methods as a technique of post-processing or maintaining association rules. Han et al (2002) proposed a solution where a concept hierarchy was used and
association rules were classified into multiple conceptual levels of granularity. This idea inspired the work in (Bing Liu et al 1999) where the existing association rule model is extended to allow users to specify multiple threshold supports. In the extended model, the threshold support is expressed in terms of minimum item supports (MIS) of the items that appear in the rule. The main feature of this technique is that the user can specify a different threshold item support for each item, similar to the scenario of assigning weights to items. This technique can discover rare item rules without causing frequent items to generate too many unnecessary rules. Liu’s model also breaks the “downward closure property”. The problem is solved by using a “sorted closure property” where the items in the item space are sorted in ascending order of their MIS values.

Agarwal et al (2001) considered the problem of mining association rules on a shared-nothing multiprocessor. Three algorithms, which explore the trade-off between computation, communication, memory usage, synchronization, and the use of problem-specific information, are presented. Methods for finding the maximal elements include All-MFS (Gunopulos et al 2003), which works by iteratively attempting to extend a working pattern until failure. A randomized version of the algorithm that uses vertical bit-vectors was studied, but it does not guarantee that every maximal pattern will be returned. In contrast, GenMax maintains only the current known maximal patterns for pruning. It integrates pruning with mining and returns the exact MFI. FPgrowth (Han et al 2000) uses the novel frequent pattern tree (FP-tree) structure, which is a compressed representation of all the transactions in the database. It uses a recursive divide-and-conquer and database projection approach to mine long patterns. Nevertheless, since it enumerates all frequent patterns but becomes impractical when pattern length is long.
Mafia (Burdick et al 2001) is the most recent method for mining the MFI. It uses three pruning strategies to remove non-maximal sets. The first is the look-ahead pruning first used in MaxMiner. The second is to check if a new set is subsumed by an existing maximal set. The last technique checks if \( t(X) \subseteq t(Y) \). If so, \( X \) is considered together with \( Y \) for extension. Mafia uses vertical bit-vector data format, and compression and projection of bitmaps to improve performance. It mines a superset of the MFI, and requires a post pruning step to eliminate non-maximal patterns. The problem of non-redundant rule generation is addressed in (Zaki 2000), provided that closed sets are available; yet an algorithm to efficiently mine the closed sets was not described in that paper. There have been several recent algorithms proposed for this task.

Close (Pasquier et al 1999) is an Apriori-like algorithm that directly mines frequent closed itemsets. There are two main steps in Close. The first is to use bottom-up search to identify generators, the smallest frequent itemset that determines a closed itemset. All generators are found using a simple modification of Apriori. After finding the frequent sets at level \( k \), Close compares the support of each set with its subsets at the previous level. If the support of an itemset matches the support of any of its subsets, the itemset cannot be a generator and is thus pruned. The second step in Close is to compute the closure of all the generators found in the first step. To compute the closure of an itemset, it is necessary to perform an intersection of all transactions where it occurs as a subset.

The closures for all generators can be computed in just one database scan, provided all generators fit in memory. Nevertheless computing closures in this way is an expensive operation. The authors of Close recently developed Pascal (Bastide et al 2000), an improved algorithm for mining closed frequent sets. They introduced the notion of key patterns and showed...
that other frequent patterns could be inferred from the key patterns without access to the database. They showed that Pascal, even though it finds both frequent and closed sets, is typically twice as fast as Close, and ten times as fast as Apriori. Since Pascal enumerates all patterns, it is only practical when pattern length is short. The Closure algorithm (Cristofor et al 2000) is also based on a bottom-up search. It performs only marginally better than apriori. Thus CHARM should outperform it easily. Recently two new algorithms for finding frequent closed itemsets have been proposed. Closet (Pei et al 2000) uses a novel frequent pattern tree (FP-tree) structure, which is a compressed representation of all the transactions in the database. It uses a recursive divide-and-conquer and database projection approach to mine long patterns.

It will be shown later that CHARM outperforms Closet by orders of magnitude as support is lowered. Mafia (Burdick et al 2001) is primarily intended for maximal pattern mining, but it has an option to mine the closed sets as well. It relies on efficient compressed and projected vertical bitmap based frequency computation. At higher supports, both Mafia and CHARM exhibit similar performance, but as one lowers the support the gap widens exponentially. CHARM can deliver improvements ten times over Mafia for low supports.

There have been several efficient algorithms for mining maximal frequent itemsets such as MaxMiner (Bayardo 1998), DepthProject (Agarwal et al 2000), Mafia (Burdick et al 2001), and GenMax (Gouda and Zakil 2001). It is not practical to first mine maximal patterns and then to check if each subset is closed. This is because it is important to check $2^l$ subsets, where $l$ is length of the longest pattern. Zaki and Hsiao (1999) tested a modified version of MaxMiner to discover closed sets in a post-processing step. Then it is found to be too slow for all except short patterns.
A decision tree (Han and Kamber 2001, Quinlan 1993) is a flow-chart-like tree structure, which is constructed by a recursive divide-and-conquer algorithm. In a decision tree, each internal node denotes a test on an attribute; each branch represents an outcome of the test; and each leaf node has an associated target class. The top-most node in a tree is called root and each path from the root to a leaf node represents a rule.

In the current field of the multi-relational classification, the most common one is Inductive Logic Programming (ILP). The well known ILP systems are FOIL (Quinlan and Cameron 1993), Golem (Muggleton and Feng 1990), Progol (Muggleton 1995), TILDE (Blockeel et al 1998), and CrossMine (Yin et al 2004). First-order inductive learning (FOIL) takes CN2 algorithm (Clark and Niblett 1989) as the basis and applies top-down and general-to-specific search to establish numerous rules. Each rule will include as many positive rules as possible. On the contrary, Golem adopts bottom-up and specific-to-general search and uses the techniques of RLGG (Relative Least General Generalization) to undertake generalization among a number of specific rules.

Progol is based on the AQ algorithm (Michalski 1969) which integrates the searching methods in FOIL and Golem. However, higher calculating cost will be caused when the database is enlarged as the common disadvantage for traditional ILP approaches. In order to reduce the time cost, TOP-down induction of first order logical decision trees (TILDE) is proposed. TILDE is a binary logical decision tree and based on the C4.5 algorithm. Experiments also show that its accuracy is definitely better than the traditional ILP approaches (Bostrom 1995). However, TILDE is inapplicable for the large-scale databases. To solve the scalability problem, based on FOIL, CrossMine algorithm (Yin et al 2004) is proposed. In order to reduce the requirement of memory, CrossMine virtually joins target relation and non-
target relation together. It propagates the primary key and the target class in target relation to all non-target relations. Accordingly, there are two additional columns “IDs” and “class labels” in each non-target relation. By this propagation approach, CrossMine can calculate the foil gain in each relation without joining all the relations into an individual table. Therefore the memory cost can be considerably reduced. Owing to the well consideration of connections among all relations, its accuracy performs much better than FOIL.

The proposed techniques aiming at the class imbalanced problem so far could be classified into three categories as follows (Barandela et al 2003). Over-sampling and under-sampling are two main techniques in this category. Over-sampling could be further classified into random over-sampling and focused over-sampling (Aha et al 1991). Random over-sampling approach over-samples the minority class at random until it matches the size of the majority class. Focused over-sampling approach over-samples the minority class only with data close to the boundaries between the minority class and the majority class. Similarly, under-sampling could also be classified into random under-sampling and focused under-sampling (Dehmeshki et al 2003, Derouin et al 1991, Lewis and Catlett 1997). The former approach removes the majority class at random until it contains as many examples as the minority class, while the latter one removes the majority examples lying further away. The main idea of focused under-sampling is to remove the noise or outlier data and to reduce the size of majority class by sampling. The combination of over-sampling and under-sampling has also been proposed (Cohen et al 2006, Zhou and Liu 2006). However, over-sampling will increase the training set size and therefore enlarge the computational burden and the impact of noise data. Under-sampling has been proved to be ineffective since it results in excluding some useful information (Barandela et al 2003, Japkowicz and Stephen 2002).
Cost-Modifying approach (Pazzani et al 1994, Zadrozny and Elkan 2001) reduces the relative misclassification cost of the majority class (or increasing that of the minority class) to make it correspond to the size of the minority class. However, it is hard for a user to assign a proper cost when he/she is unfamiliar with the domain knowledge (Japkowicz and Stephen 2002). This approach is more attractive and has been proved to be more effective than the above two approaches (Japkowicz and Stephen 2002). The main idea of this technique is to develop an approach that is insensitive to the imbalance problem. The proposed techniques includes example weighting, rule removing, attribute correlation analysis, etc. (Anto et al 2000, Ezawa et al 1996, Fawcett and Provost 1996, Kubat et al 1998, Lawrence et al 1998). Among the proposed imbalance-insensitive approaches, some of them are limited to specific dataset and some take a lot of training time due to the natural property of neural network. Comparatively, SHRINK (Kubat et al 1998) is applicable to most applications with numeric attribute data and eliminates the disadvantage described above. SHRINK was developed by the principle of BRUTE (Riddle et al 1994), it searches for the most accuracy for not only the minority, but also the majority simultaneously by the use of g-mean to reach the best partition. However, its shortcoming is to only take care of numeric attribute and search for the best interval in a single one to each attribute. Once if the positive rules are distributed in two extremities of attribute values, SHRINK will have a poor accuracy. Context Based Sequential Pattern mining algorithms (Radoslaw 2007) introduces a novel Context Mapping algorithm used for the context pattern mining. This algorithm takes into account non nominal attributes and similarity of sequence's elements. Another algorithm called Fast Sequential Pattern Enumeration (FSPE) (Jie et al 2009), which scans the database only once to mine sequential patterns without the need to predetermine the minimum support threshold.