A Comparative Study Of Multi-Relational Decision Tree Learning Algorithm

Vaibhav Tripathy

Abstract: This paper provides a comparative study of the working and implementation of multi relational decision tree learning algorithm with that of MRDTL-2, which works on the theory initially proposed by Knobbe et al. This paper also outlines some of the shortcomings of MRDTL viz. calculation speed, accuracy and most importantly handling of missing values. We had used some of the real world data sets from various data mining competition and performed a graphical comparison for the aforementioned two approaches. Results from the experiments signify that MRDTL-2 is convincingly more efficient approach than its predecessor.

Keywords: MRDTL, Shortcomings of MRDTL, MRDTL-2, Efficiency of MRDTL-2, Graphical Analysis

1. Introduction

The massive advancement in the field of digital storage, huge throughput data acquisition, and communication technologies have shaped it achievable to gather and hold very enormous amounts of data in many scientific and commercial domains. Substantial amount of this data resides in relational databases. Even though when the data repository is not a relational database, these are often viewed conveniently as heterogeneous data sources as if they are a collection of relations [1] which are then utilized for the purpose of extracting, inferring and organizing information from multiple sources. Hence, this topic of relational learning from relational databases started getting to acquire significant considerations in the literature [2],[3],[5],[6],[7],[8],[9],[10],[11],[12],[13]. Knobbe et al. [4] proposed and outlined a general framework for multi-relational data mining which uses structured query language (SQL) to retrieve the information required for developing classifiers (e.g., decision trees and graphs) from multi-relational data. Based on this very framework, [14] proposed a multi- relational decision tree learning algorithm (MRDTL). Experiments reported by this demonstrated that decision trees framed by employing MRDTL have much accurate results which are comparable to those obtained using other algorithms on several multi-relational data sets.

2. Previous approaches to Relational Learning

Several techniques which were proposed previously for relational data mining are discussed below:

- **Inductive Logic Programming (ILP)** evolved from Induction which is used as a major technique to develop models through machine learning algorithm and Logic Programming which works as programming paradigm which uses first order logic to represent relations. ILP is a wide field which got evolved from its foremost focus on developing algorithms for the processing of logic programs from sources and background knowledge (i.e. inferring or obtaining knowledge for some domain) to more latest considerations for association, clustering, classification, regression, and analysis [10]. Because of its flexible and expressive ways of defining background knowledge and examples, the field acknowledges both single-table representation and multiple-table representation of the data. Considering other learning approaches, ILP has been one of the first and most elaborated ones. Though the usage of ILP in relational data mining has been limited, because of the reason of the differences in input specification and non supportability of language in different ILP engines. So in a way to deal with the logic formalism and to integrate different input specifications for different ILP engines, proposed Unified Modelling Language (UML).[15]

- **Bayesian Logic Programming** is first order extension of Bayesian networks is introduced as a simplification and reformulation of PLPs, but also as a common framework for the previous mentioned approaches. This kind of In BLPs, the qualitative part of the Bayesian net is presented by a set of Bayesian definite clauses. The difference between this type of clauses and classical clauses is that every node in a BLP depicts a random variable. Set of random variables corresponds to the least Herbrand model of this logical program, i.e. the group of all ground nodes that are logically entailed by it. The parents of a some random variable n are all facts directly influencing n.[8]

- **Multi Relational Data Mining** as a term was initially used by [4] in a way to describe a novel approach for relational learning and knowledge discovery from relational databases and data consisting of complex/structured objects. In multi-relational data

*Vaibhav Tripathy* completed his graduation in the field of ‘Information Technology’ and is currently pursuing master degree program in ‘Software Engineering’ from Suresh Gyan Vihar University, India.

E-mail: er.vaibhavtripathy@live.com

---

**INTERNATIONAL JOURNAL OF SCIENTIFIC & TECHNOLOGY RESEARCH VOLUME 2, ISSUE 8, AUGUST 2013**

**ISSN 2277-8616**
mining framework, the data model consists of many tables, each describing features of particular objects, only one view of the objects is central to the analysis. The user can select the kind of objects to be analyzed by selecting one of the tables as target table. The point of importance is that each record in the target table will refer to a single object in the database. Once the target table has been chosen, a particular descriptive attribute from that table can be selected for classification or regression purposes, this is termed as the target attribute within the target table.

3. Methodology

3.1 Multi-relational decision tree learning algorithm

Proposed by [16] is an extension of the logical decision tree induction algorithm called TILDE proposed by [17]. MRDTL broadens TILDE’s [18] approach in order to deal with records in relational databases. It uses first order logic clauses to represent decisions (nodes) in the tree. It creates decision trees whose patterns of nodes are multi-relational in nature i.e., selection graphs. Unlike propositional version of the decision tree algorithm [19], MRDTL adds decision nodes to the tree through a process of successive refinement as far as some termination criteria are encountered. Whenever some termination criterion is met, a leaf node corresponding to its class is introduced instead. The choosing of decision node to be added at every step is influenced by a suitable impurity measure (e.g., information gain). Hence, MRDTL commences with a single node from the tree root, which represents the set of all objects of interest in the relational database. This node tends to target table $T_0$ together with the specified target attribute. The general outline for the algorithm taken from [14] is shown below. In the algorithm the function optimal-refinement deals with every possible refinement that can be done to the current pattern $S$ with respect to the database $D$ and then selects, in a greedy approach the optimal refinement (i.e., optimal information gain). The feasible set of refinements to be made at a certain point during the process is ruled by the current selection graph, the database structure, and the multiplicity of the associations involved. $^3$ denotes the complement of the selection graph (i.e., it selects objects from the database which are not selected by S). The induction algorithm outlined below creates binary splits decision trees.

$$
\text{tree\_induction} (T: \text{tree}, D: \text{database}, S: \text{selection graph})
$$

\begin{align*}
\text{Input} & \quad \text{database } D, \text{selection graph } S \\
\text{Step 1} & \quad R_i = \text{optimal-refinement}(S) \\
\text{Step 2 if} & \quad \text{stopping criteria}(S) \\
\text{Step 3 return leaf} \\
\text{Step 4 else} \\
\text{Step 5} & \quad T_i := \text{tree\_induction} (D; R(S)) \\
\text{Step 6} & \quad T_r := \text{tree\_induction} (D; \overline{R} (S))
\end{align*}

Fig 1 General working structure of a decision tree learning algorithm

Based on the above mentioned algorithm, a flowchart is constructed below in fig. 2, which specifically defines the workflow paradigm and provides a general outline of its structure.

### 3.1.1 Refinements of Selection Graphs

As noted above, once the target table and the target attribute have been selected (i.e. the kind of objects central to the analysis have been completely defined) a number of possible refinements can be applied to the initial node representing $T_0$ in order to find a hypothesis to be consistent with the data in the training database. [4] Proposed some possible ways of refining selection graphs.

**Add condition (positive):** This refinement simply adds a condition to the intended selection node in $S$ without altering its structure. Assuming the graph given below fig. 3(a) taken from [14], the refinement to be made was “atom.element = ‘b’”. Previously to this operation the set of conditions for atom node comprised only the condition “atom.charge <= -0.392”. Hence after adding the discussed condition the resultant graph becomes as shown in 3(a).

**Add condition (negative):** This condition is the complementary version of operation for the previous one. If the node that is being refined does not represent the target table, then this refinement adds a new absent edge from the parent node of that selection node to a new closed node that is a duplicate copy of the selection node that is being refined fig 3(b). Its condition list and join list (depicted by the edges coming out from this

![Flowchart depicting the work pattern of the algorithm](image-url)
node) must be copied to the new closed node, and the first list must be elongated by adding the new condition not negated.

- **Mutual Exclusion:** The subsets associating with the patterns must be mutually exclusive which are derived from the same parent, hence two kind of refinement, (add a condition and add an edge plus a node) are brought in with their complementary operations.

- **Adding present edge and open node:** Under this refinement, an association is instantiated in the data model as a present edge along with its table represented as an open node and then these are added to selection graph.

- **Adding absent edge and closed node:** This is the complementary portion to the previous one. It instantiates an association in the data model as an absent edge along with its corresponding table depicted as a closed node and adds these to selection graph.

![Fig 3 Graph depicting the “add condition”](image)

- **Look-ahead refinement:** In some cases while refining a selection graph, employing optimal refinement may not result in any kind of information gain. Hence in such cases, if any modification to particular selection graph does not produce any improvement, then that path is discontinued and a leaf node is introduced instead, even if it introduces future possible conditions or edges that are relevant to the search process.

### 3.1.2 Shortcomings of MRDTL

There were several shortcomings which were found in literature while implementing this algorithm [36] in essence. Several experiments reported in literature have shown that decision trees which were constructed using MRDTL have comparable accuracies to that obtained using other algorithms on many multi-relational data sets [36]. However, MRDTL has two convincing limitations from the viewpoint of multi-relational data mining on large data sets.

- **Slow running time:** MRDTL [16] based on the (multi-relational data mining framework) employs selection graphs in order to query the databases and to obtain the information required for constructing the classifier. The experiments using MRDTL on data from KDD Cup 2001 [5] demonstrated that the queries directed by such selection graphs are a major cause of hindrance in terms of the running time of the algorithm.

- **Unable to handle missing attribute values:** In many multi-relational real-world databases, a significant portion of the data has one or more missing values. For instance, in gene localization task from KDD Cup 2001 [21], 70% of CLASS, 50% of COMPLEX and 50% of MOTIF attribute values are missing. The implementation of MRDTL [14] doesn't handle each missing value and consequently doesn't include any statistically well-managed techniques to deal with missing values. Hence, the accuracy of decision trees constructed using MRDTL is of major concern as these missing value attribute are pretty common in real multi-relational datasets. For e.g., the accuracy of MRDTL on the gene localization task was reported approximately 50% in the literature.

### 3.2 MRDTL-2

MRDTL-2 is a more efficient version of MRDTL which is proposed by [22]. It is based on the concepts proposed by [14] which in turn are based on the algorithm proposed by [16] and the logical decision tree induction algorithm called TILDE [2]. The working of MRDTL-2 is very similar to that MRDTL, but in addition to the framework it suggested that some of the results of computations that were performed in the phase of adding nodes and refinements in the decision tree, can be reutilized at lower levels in the phase of further refinement of that given tree. Previously in MRDTL while refining an existing selection graph a unnecessary repeated work is performed each time by re-retrieving those instances which are previously covered by that selection graph. This problem can be significantly simplified by storing those instances which are covered by the selection graph from previous iteration in a table to avoid re-retrieval from the database. Hence, in MRDTL-2 with each iteration of the algorithm, primary keys from all open, front nodes of the selection graph for all the objects covered by it along with its classification values are stored in a table. That resulting table of primary keys is referred to as sufficient table for selection graph $S$ and is denoted by $\Sigma_S$. This can be understood as storing the ‘skeletons’ of the objects covered by that selection graph. And as the table doesn't stores other attribute's information from the records except their primary keys, the resultant table comprises of primary keys wholly. In this way the tables that are needed to be joined becomes considerably less unlike in MRDTL in which the number of tables increases each time. It is this growth that was accountable for the evident performance decline of MRDTL as nodes get added to the decision tree. Hence, this mere change increases up the execution rate considerably.
3.2.1 Handling of Missing Values
MRDTL-2 incorporates a simple approach in order to deal with missing attribute values in the data. For each attribute in a table a Naive Bayes model is constructed based on the other attributes without including the class attribute. Missing attribute values are then ‘filled in’ with the most acceptable value which is predicted by the Naive Bayes for that corresponding attribute. Once the table are pre-processed in the database using this technique, MRDTL-2 then starts to build decision trees from the obtained tables which contain no missing attribute values.

4. Experimental Results
The experiments mainly focused on three data sets - the mutagenesis dataset which has been widely used in Inductive Logic Programming (ILP) research [20], the dataset for localization and function of protein/gene from KDD Cup 2001 [23] and the dataset for predicting thrombosis taken from PKDD 2001 Discovery Challenge [24].

- We compared and analysed the results graphically in context of accuracy factor with the best reported results in the literature which were obtained using MRDTL-2 algorithm [22] and concluded that MRDTL-2 is more accurate.

5. Conclusion and Discussion
The results of the comparison of MRDTL-2 performance with the best-known reported results for the same datasets from the literature shows clearly that MRDTL-2 outplays all the other approaches in the field of multi-relational data mining. The major part for its better performance is the ability of MRDTL-2 to handle missing attribute values which is quite concerning problem. Moreover it also provides a better approach which speeds up the execution process and reduces the running time of the algorithm. Hence, MRDTL-2 by overcoming aforementioned limitations outperforms all other previous approaches and proves to be a better approach.

6. Acknowledgment
I would like to express my sincere gratitude to Mr. Dev Krishna (Asst. Prof., Suresh Gyan Vihar University) for his valuable technical suggestions and guidance towards the completion of this paper.

References


