Behavior Analysis of Different Decision Tree Algorithms

Anshu Katare, Dr. Vijay Anant Athavale
1Research Scholar, Department of Computer Science & Engineering, PAHER University Udaipur, Rajasthan
2Director, Dev Raj Group’s Technical Campus, Ferozepur, Punjab
1anshu.tiwaree@gmail.com, 2vijay.athavale@gmail.com

Abstract: Industries and other organization are now changing their working from manual to automatic working environment. In the same way the business decisions and complex analysis of the working patterns are evaluated using the machines or computer intelligence. Due to this a large amount of data is required and at the same time an effective and efficient data mining algorithm is required to enhance their decisions and produces accurate pattern analysis. These data is also turned into knowledge using some data mining process. In this paper we include the review and our study work over different decision tree and data model. Additionally here we provide the basic properties that affect the performance of any decision tree algorithm design.

Keywords- Decision Tree, Data Mining, Properties, Effects.

I. INTRODUCTION
Useful information extraction from the data, data source required data mining and their different approaches. Decision tree is a data model that returns results in most transparent way. Decision trees are a mathematical model based on weighted graph. Decisions trees are commonly used in operations research, specifically in decision analysis, to help in identify a strategy most likely to reach a goal. To demonstrate the working of any data model we provide here a basic example for processing using any machine learning algorithm.

Suppose we have a series with some variables and someone want to formulize this series we can achieve this using simple mathematical formula. The above given formula is satisfying all the same kind of series. Now if our problem is increased by adding some more series with one series and here we required some formulation that satisfies all the series. We required writing one another formula that satisfies the entire problem.
1+2+3+........n= n (n+1)/2

In this paper we are going to work with data mining and analyzing its behavior and effects of the data over decision trees, we work with different techniques and compare them to get the performance of both techniques. In this paper we are providing the different effecting parameters for the decision tree.

Decision trees have various advantages:
- Simple to understand and construct. It’s easy to understand decision tree models after a short explanation.
- Requires little data training. Other techniques often require data normalization, sample data need to be created and blank values to be removed.
- Able to handle both numerical and nominal data. Other methods are usually specialized in analysing datasets that have only one type of variable. Ex: relative rules can be used only with nominal variables while neural networks can be used only with numerical variables.
- Uses a white box prototype. If a given condition is visible in a model the justification for the circumstance is easily described by Boolean logic. Example of a black box model is an artificial neural network since the description for the results is complex to understand.
- Probable to validate a representation using statistical tests. That makes it possible to explanation for the reliability of the model.
- Robust. Performs well even if its assumptions are somewhat violated by the true model from which the data were generated.
- Performs well with large data in a little time. Huge quantity of data can be analysed using standard computing resources.

Disadvantages of Decision Tree
- Decision-tree learning algorithms are based on heuristic algorithms such as the greedy algorithm where locally optimal decisions are made at each node. This kind of algorithms cannot promise to return the globally optimal decision tree.
- Decision-tree learners can create complex trees that do not generalize the data. This is called over fitting. Mechanisms such as pruning are necessary to avoid this problem.

There are concepts that are hard to learn because decision trees do not precise them easily, like XOR, parity or multiplexer complications. In this type of cases, the decision tree becomes too large. Approaches to solve the problem involve either changing the presentation of the problem domain (also known as propositionalisation) or using learning algorithms based on more expressive representations (such as statistical relational learning or inductive logic programming).
**C5.0 Improvements from C4.5 algorithm**

- Speed - C5.0 is significantly faster than C4.5 (several orders of magnitude)
- Memory usage- C5.0 is more memory efficient than C4.5. C5.0 commonly uses an order of magnitude less memory than C4.5 during rule set construction.
- Accuracy- the C5.0 rules sets have noticeably lower error rates on unseen cases. Sometimes the C4.5 and C5.0 rule sets have the same predictive accuracy, but the C5.0 rule set is smaller.
- Smaller decision trees - C5.0 gets similar results to C4.5 with considerably smaller decision trees.
- Support for boosting - Boosting improves the trees and gives them more accuracy.
- Weighting - C5.0 allows you to weight different attributes and misclassification types.

### Problem of Current System

Issues in data mining with decision trees. There are some issues in decision trees which include:

- Determining how deeply to grow the decision tree
- Handling continuous attributes
- Choosing an appropriate attribute selection measure
- Handling training data with missing attribute values
- Handing attributes with differing costs
- Improve computational efficiency

Comparative study on ID3, C4.5 and C5.0 is done in the below section. C4.5 is the successor algorithm of ID3. C5.0 is the successor algorithm of C4.5. C5.0 algorithm has many features like:

- C5.0 algorithm can respond on noise and missing data.
- C5.0 provides boosting.
- A large decision tree may be difficult to read and comprehend.
- C5.0 provides the option of viewing the large decision tree as a set of rules which is easy to understand.
- Over fitting is solved by the C5.0 and Reduce error pruning technique.
- C5.0 can also predict which attributes are relevant in classification and which are not. This technique, known as Winnowing is especially useful while dealing with high dimensional datasets.

## II. BACKGROUND

### A. Related Work

Data mining is the method that makes efforts to determine hidden patterns in large data sets. It consumes methods of artificial intelligence, machine learning, statistics, and database systems. The goal of the data mining process is to extract information from a data set and transform it into an understandable structure for further use. There are various domains where we can apply these structure to get classify data, making decisions, and data analysis.

Decision tree can be used to visually and explicitly represent decisions and decision making. In data mining, a decision tree describes data but not decisions; rather the resulting classification tree can be an input for decision making. Decision trees used in data mining are of two main types: Classification tree analysis is when the predicted outcome is the class to which the data belongs and Regression tree analysis is when the predicted outcome can be considered a real number.

### 1. Data format: the data is raw in nature and found in unformatted way. But to work with the data model required to format data first, this process also called the data pre-processing. Data pre-processing includes the different phases to achieve a well formatted and arranged data. Moreover, after processing the data can be categorized into three main parts:

- Data set with only numerical values
- Data set with nominal values
- Data set with both nominal and numerical values.

For the experimental purpose we use the data. We use the ARFF data format that is available online for experiments of machine learning. ARFF also abbreviated as attribute relationship file format.

The Header of the ARFF file contains the name of the relation, a list of the attributes (the columns in the data), and their types.

% 1. Title: Iris Plants Database
% 2. Sources:
% (a) Creator: R.A. Fisher
% (b) Donor: Michael Marshall
%(MARSHALL%PLU@io.arc.nasa.gov)
% (c) Date: July, 1988
%
@RELATION iris
@ATTRIBUTE sepallength NUMERIC
@ATTRIBUTE sepalwidth NUMERIC
@ATTRIBUTE petallength NUMERIC
@ATTRIBUTE petalwidth NUMERIC
@ATTRIBUTE class {Iris-setosa,Iris-versicolor,Iris-virginica}

The **Data** of the ARFF file looks like the following:

@DATA
5.1,3.5,1.4,0.2,Iris-setosa
4.9,3.0,1.4,0.2,Iris-setosa
4.7,3.2,1.3,0.2,Iris-setosa
4.6,3.1,1.5,0.2,Iris-setosa
5.0,3.6,1.4,0.2,Iris-setosa
5.4,3.9,1.7,0.4,Iris-setosa
4.6,3.4,1.4,0.3,Iris-setosa
5.0,3.4,1.5,0.2,Iris-setosa
2. Over fitting: As we know in constructing decision trees we use training data set. We do this because we want to capture some general underlying functions or trends in that data, usually to be used in prediction. As we are not interested in capturing all the exact nuances and extremities of the training data. These are normally the result of errors or peculiarities that we are not likely to come across again. It is important that we can use our decision tree model to predict or generalise over future instances that we might obtain. Over fitting occurs when our decision tree characterises too much detail, or noise in our training data. More formally this can be stated as:

Two hypotheses, H1 and H2 over some data exist with the following relationship:

Training set errors(H1) < Training set errors(H2)

AND

Testing set errors(H1) > Testing set errors(H2)

As well as noise in the training data, this can happen when we don’t have much trained data and are trying to extrapolate an underlying hypothesis from it.

We want our decision tree to generalize well, but unfortunately if we build a decision tree until all the training data has been classified perfectly and all leaf nodes are reached, then chances are that it ‘we’ll have a lot of misclassifications when we try and use it. There are methods that we can use to avoid over fitting such as "pruning"

3. Pruning: Over fitting is a significant practical difficulty for decision tree models and many other predictive models. Over fitting happens when the learning algorithm continues to develop hypotheses that reduce training set error at the cost of an increased test set error. There are several approaches for avoiding over fitting in building decision trees.

- Pre-pruning that stop growing the tree earlier, before it perfectly classifies the training set.
- Post-pruning that allows the tree to perfectly classify the training set, and then post prune the tree.

The important step of tree pruning is to define a criterion be used to determine the correct final tree size using one of the following methods:

1. Use a distinct dataset from the training set (called validation set), to evaluate the effect of post-pruning nodes from the tree.
2. Build the tree by using the training set, then apply a statistical test to estimate whether pruning or expanding a particular node is likely to produce an improvement beyond the training set.
   - Error estimation
   - Significance testing
3. Minimum Description Length principle : Use an explicit measure of the complexity for encoding the training set and the decision tree, stopping growth of the tree when this encoding size (size(tree) + size(misclassifications(tree)) is minimized.

   Tree height: The height of the tree represents the number of decisions (comparisons) made to sort the particular data represented by the leaf. The longest path from root to leaf (the height) gives the number of comparisons in the worst case the average of the lengths of the paths from the root to all the leaves gives the average number of comparisons.

   1. Of all possible decision trees for sorting items by comparison, which has a longest path which is the shortest?.
   - Gives lower bound on the average sorting time for comparison sorts.
   2. Which decision tree has the smallest average path length?
   - Gives lower bound on the average sorting time for comparison sorts.

4. Missing Attribute Values: The problem of reasoning with missing attribute values is known in machine learning and a lot of work has been already done for interpretation of the issues related to this problem as well as methods for reasoning with missing attribute values. However, there is not a single satisfactory solution to the problems related to reasoning over incomplete data in the considered sense. In relational databases the nature of missing values was established and for more than a decade also the industrial standards fulfil the proposed logical framework and semantically meaning of the null values. Such an approach is not yet available in data mining at all and particularly, in the rough set theory and practice. Furthermore, it seems to be almost infeasible to discover one theoretical framework for dealing with missing attribute values and their role in induction learning that will fit in all aspects of Machine Learning. The findings in area of missing attribute values are rather loosely connected or even exclusive and do not form any coherent guidelines that would be applicable to a wide range of data mining problems.

B.Pros and cons of the decision tree: Decision tree classifiers algorithm is interesting and advantageous by the following reasons.

1. Complex decision can be estimated by the union of simpler local decision regions at various levels of the tree.
2. In a tree classifier a sample is tested against only certain subsets of classes, thus unnecessary computational overhead is removed.
3. Only one subset of features is used for discriminating among all classes.
4. Large numbers of features and classes usually needs to estimate either high-dimensional distributions (possibly multimodal) or certain parameters of class distributions. In so doing, one usually faces the problem of "high-dimensionality." This problem may be avoided in a Decision Tree Classifier by using a smaller number of
features at each internal node without excessive degradation in the performance.
As it is the domain of interest this data structure also having some problems.
1. If any data set contains a large number of classes can cause the number of terminals to be much larger than the number of actual classes and thus increase the search time and memory space requirements.
2. Errors may accumulate from level to level in a large tree.
3. There may be difficulties involved in designing an optimal DTC. The performance of a DTC strongly depends on how well the tree is designed.

C. Optimal decision tree construction: The problem of designing a truly optimal DTC seems to be a very difficult problem. In fact it has been shown by Hyafil and Rivest [3] that the problem of constructing optimal binary trees, optimal in the sense of minimizing the expected number of tests required to classify an unknown sample is an NP-complete problem and thus very unlikely of non-polynomial time complexity. It is conjectured that more general problems, i.e. the problem with a general cost function or minimizing the maximum number of tests (instead of average) to classify an unknown sample would also be NP-complete. They also conjecture that no sufficient algorithm exists (on the supposition that P ≠ NP) and thereby supply motivation for finding efficient heuristics for constructing near-optimal decision trees.

The various heuristic methods for construction of DTC can roughly be divided into four categories:
- Bottom-Up approaches
- Top-Down approaches
- The Hybrid approach and
- Tree Growing-Pruning approaches.

In this section of the work we analyses and provide main challenges involve in the decision tree design. In the next section of this paper we provide the basic idea of designing a decision tree algorithm.

III. DESIGN OPTIMAL DECISION TREE

To design a decision tree model we required to work with the following given stages. The four basic approaches for design of DTC's examined in this review are:
1. Bottom-up approach where one starts with the information classes and continues combining classes until one is left with a node containing all the classes.
2. Top-down approach where starting from the root node, using a splitting rule, classes are divided until a stopping criterion is met. The main issues in this approach are: a) choice of splitting criterion; b) stopping rules; c) labelling the terminal nodes
3. Hybrid approach where one use a bottom-up procedure to direct and assist a top-down procedure.
4. Tree growing-pruning approach where in order to avoid some difficulties in choosing a stopping rule, one grows the tree to its maximum size where the terminal nodes are pure or almost pure, and then selectively prunes the tree.

IV. CONCLUSION

In this paper our main goal is to analyze and make review of traditional methods of data mining and their popular methods. Thus in future we use and design a new decision tree algorithm with the help of data preprocessing for data mining. And evaluate their performance, after evaluation we simulate there results using a real time application.

REFERENCES