Analysis of Decision Tree-A Survey

Ms. Priti Phalak
Computer Engineering
TCET
Mumbai, India.

Mr. Kiran Bhandari
Dy. HOD of Computer Engineering
TCET,
Mumbai

Dr. Rekha Sharma
HOD of Computer Engineering
TCET
Mumbai, India.

***Abstract*—Decision tree is a fast and efficient classification prediction algorithm in data mining which is most widely used for decision making. It is one of the most popular data-mining techniques for knowledge discovery. It systematically analyses the information contained in a large data source to extract valuable rules and relationships. It recursively partitions a data set of records using depth-first greedy approach or breadth-first approach, until all the data items belong to a particular class are identified. Decision trees are categorized as a supervised method that trying to find the relationship between input attributes and target attributes which represent the relationship in structure as a model. It first compare the attribute values of the object, and select optimal attribute as the root node, then depend on attribute to determine the leaf node of the tree branch from the root down.**

Most decision tree classifiers perform classification in two phases: Tree-growing (or building) and Tree-pruning. The tree building is done in top-down manner. During this phase the tree is recursively partitioned till all the data items belong to the same class label. In the tree pruning phase the full grown tree is cut back to prevent over fitting and improve the accuracy of the tree in bottom up fashion. It is used to improve the prediction and classification accuracy of the algorithm by minimizing the over-fitting. There are various techniques for pruning decision trees such as Cost Complexity Pruning, Reduced Error Pruning, Pessimistic Pruning, Minimum Error Pruning, Error-Based Pruning and for decision tree induction algorithms such as ID3, C4.5, CART.

***Index Terms* —Decision Tree, Classification, Pruning, Pruning methods, Splitting Criteria.**

I. INTRODUCTION

Decision tree is one of the most popular and efficient technique in data mining. This technique has been established and well-explored by many researchers. A decision tree is a flow-chart-like tree structure, where each internal node is denoted by rectangles, and leaf nodes are denoted by ovals. All internal nodes have two or more child nodes. All internal nodes contain splits, which test the value of an expression of the attributes. Arcs from an internal node to its children are labeled with distinct outcomes of the test. Each leaf node has a class label associated with it. Decision trees are a reliable and effective decision making technique that provide high classification accuracy with a simple representation of gathered knowledge. Classification is the process of categorization of data for its most effective and efficient use. Decision tree algorithms have characteristics as follows:

(1) Structure of the decision tree is simple and easy to understand.
(2) Decision tree is more appropriate for the case of a large amount of training set.
(3) Decision tree has high accuracy.

Classification is a supervised learning technique in data mining where training data is given to classifier that builds classification rules .Classification of data objects is a data mining and knowledge management technique used in grouping similar data objects together. There are many classification algorithms available in literature such as SVM, Naive Bayes, decision tree induction algorithm (ID3, C4.5 and CART) etc., but decision tree is the most commonly used because of its ease of execution and easier to understand compared to other classification algorithms. Classification of data is one of the important tasks in data mining. Decision tree is one of the most widely used technique, and is very popular among researchers because of their simplicity, intelligibility, ease of implementation, decision tree construction faster and produce better accuracy then other classification algorithms.

Decision tree can be represented by two types of structures: Tree structure (hierarchical structure) and Rules (if-then statement). If decision tree is complicated, tree structure and rules might be wasted. Some decision tree algorithms may produce a large structure of tree size and it is difficult to understand. Further it leads to misclassification of data which often occurs in learning process. For such a complex tree, pruning procedures must be developed to facilitate the interpretation. Pruning methods have been introduced to reduce the complexity of tree structure without decreasing the accuracy of classification. Pruning is a method that reduces the size of the tree by removing parts of the tree that are not meaningful to avoid unnecessary complexity and to avoid over-fitting of the dataset. Pruning decision trees is a fundamental step in optimizing the computational efficiency as well as classification accuracy of such a model. The complexity of tree is clearly controlled by the pruning methods.
The main objective of this seminar is to study top-down algorithmic framework for decision tree induction, decision tree construction process and pruning process, types of decision tree induction algorithm such as C4.5, ID3, CHAID,

QUEST and CART etc. along with various pruning methods (Cost-Complexity Pruning, Reduced Error Pruning, Minimum Error Pruning, Pessimistic Pruning, Error-Based Pruning, Optimal Pruning, Minimum Description Length Pruning), also comparative study of decision tree induction algorithms such as C4.5, ID3 and CART etc. and to study the application of decision tree induction algorithm.
Seminar report is organized as Follows:
Section 1: Introduction of decision trees, motivation and objective of seminar report.
Section 2: Types of decision tree induction and different types of pruning methods.
Chapter 3: Comparative study of decision tree induction algorithms.
Chapter 4: Explain the application of decision tree algorithm.
Chapter 5: Comparative analysis of decision tree induction algorithm and comparative analysis of pruning methods.
Chapter 6: Conclusion.

II. PRELIMINARIES

Technical terms and notations used in this paper are as follows:

Entropy:
Entropy is a measure of the number of random ways in which a system may be arranged. For a data set S containing n records the information entropy is defined as

$$Entropy(S) = -\sum P_1 \log_2 P_1$$

(Here P_1 is the proportion of S belonging to class 1.)

Gain
Gain or the expected information gain is the change in information entropy from a prior state to a state that takes some information. The information gain of example set S on attribute A is defined as

$$Gain(S,A) = Entropy(s) - \sum_{v \in values(A)} \frac{|S_v|}{|S|} Entropy\ (S_v)$$

where Value(A) is the set of all possible values of attribute A,
S_v is the subset of S for which attribute A has value v,
$|S_v|$ is the number of elements in S_v and
| S | is the number of elements in S.

Gini index
Gini index for a data set S is defined as $Gini(S) = 1 - \sum P_I$
and for a 2-split

$$gini_{split}(S) = \frac{n1}{n} gini(S_1) + \frac{n2}{n} gini(S_2)$$

Hunts algorithm
Hunts method algorithm for decision tree construction trains the data set with recursive partition using depth first greedy technique, till all the record data sets belong to the class label.

There are following types of attributes used in decision tree algorithm

- *Categorical* attribute is also called as discrete or nominal attributes. In Categorical Data, a set of data is sorted or divided into different categories, according to the attributes of the data. Categorical data are always nominal whereas nominal data need not be categorical. Nominal data means, a type of categorical data in which objects fall into ***unordered*** categories. For example, gender is a categorical variable having two categories (male and female). Generally, discrete data are counts.
- *Continuous* attributes Has real numbers as attribute values Examples: temperature, height, or weight. Practically, real values can only be measured and represented using a finite number of digits. Continuous attributes are typically represented as floating-point variables. Generally, continuous data come from measurements.

III. TYPE OF DECISION TREE

A. ID3

Iterative Dichotomiser 3 is a simple decision tree learning algorithm introduced by Quinlan Ross. It is serially implemented and based on Hunt's algorithm. The basic idea of ID3 algorithm is to construct the decision tree by employing a top-down, greedy search through the given sets to test each attribute at every tree node. It is a top-down approach starting with selecting the best attribute to test at the root of the tree. The selection of the best attribute in ID3 is based on an information theory approach or entropy. Entropy is a measure of the amount of uncertainty present in a set of data. When all data in a set belong to a single class, there is no uncertainty and hence, the entropy is zero. In general, the value of entropy falls between 0 and 1and reaches a maximum when the probabilities are all the same. ID3 does not apply any pruning procedure. It does not handle numeric attributes neither missing values. It does not give accurate result when there is noise. Thus an intensive pre-processing of data is carried out before building a decision tree model with ID3. Given a set S, containing two examples ('positive' and 'negative') of target concept, the entropy of a set S relative to this binary classification is defined as:

$$E(S) = -P_{(positive)} \log_2 P_{(positive)} \quad - P_{(negative)} \log_2 P_{(negative)})$$

Where P(positive) and P(negative) are the fraction of positive and negative examples in S.

Information gain calculates the expected reduction in entropy. Gain (S, X) of an attribute X, relative to a collection of examples S is,

$$\text{Gain(S, X)} = \text{E(S)} - \sum_{v \in values(x)} \frac{|Sv|}{|S|} \times \text{E(Sv)} \qquad (2)$$

where, values (X) = set of all possible values for attribute X.
Sv = subset of S for which attribute X has value v
(ie. Sv = {s∈S | X(s) = v})
E(S) = entropy of the original collection S.
ID3 select splitting attributes with the highest information gain. It can handle only nominal attributes. Modifications and

improvements on the ID3 algorithm culminated into the popular C4.5 algorithm.

B. C4.5

C4.5 algorithm is an improved version of ID3, this algorithm uses Gain Ratio as a splitting criteria, instead of taking gain in ID3 algorithm for splitting criteria in tree growth phase. Hence C4.5 is an evolution of ID3. This algorithm handles both continuous and discrete attributes- In order to handle continuous attributes, C4.5 creates a threshold and then splits the list into those whose attribute value is above the threshold and those that are less than or equal to it. Like ID3 the data is sorted at every node of the tree in order to determine the best splitting attribute. The splitting ceases when the number of instances to be split is below a certain threshold. The main advantages of C4.5 is when building a decision tree, C4.5 can deal with datasets that have patterns with unknown attribute values. C4.5 can also deal with the case of attributes with continuous domains by discretization. This algorithm handles training data with attribute values by allowing attribute values to be marked as missing. Missing attribute values are simply not used in gain and entropy calculations. It has an enhanced method of tree pruning that reduces misclassification errors due to noise or too much detail in the training data set

In general, when a decision tree is built, missing data are simply ignored. The gain ratio is calculated by looking only at the other records, which have a value for that attribute. In order to classify a record with a missing attribute value, the attribute values for the other records can be used to predict the same.

If S is the set of training data denoting a concept with c classes, f(Cj, S) is the frequency of class Cj occurring in that set, then the expected information required to classify a given class in S is:

$$Info(S) = -\sum_{j=1}^{c} \frac{f(Cj,S)}{|S|} log_2 \left(\frac{f(Cj,S)}{|S|}\right) \quad (3)$$

when an attribute, A, with v values, has been selected as a test attribute, then the expected information needed to identify a class under that test is:

$$Info_A(S) = \sum_{i=1}^{v} \frac{|s_i|}{|S|} info(S_i) \quad (4)$$

where S1, S2, . . ., Sv is the subset of S all of whose instances possess value i for attribute A.

The information gain is the difference between the expected information needed to identify a class with and without the test on attribute A:

$$\text{Gain(A)} = \text{Info(S)} - \sum_{i=1}^{v} \frac{|S_i|}{|S|} \times Info(S_i) \quad (5)$$

The attribute giving the maximum information gain is selected as the current split. ID3 used information gain criterion (Equation.6) to select the test for partition. However, the gain criterion is biased towards the high frequency data. To restructure this problem, C4.5 normalizes the information gain by the amount of the potential information generated by dividing T into v subsets:

$$Splitinfo(A) = -\sum_{i=1}^{v} \frac{|S_i|}{|S|} log_2 \left(\frac{|S_i|}{|S|}\right) \quad (6)$$

C4.5 selects the test to partition the set of available cases is defined as:

$$gainratio(A) = \frac{gain(A)}{Splitinfo(A)}$$

C4.5 selects the test that maximizes gain ratio value. The difference between ID3 and C4.5 algorithm is that ID3 uses binary splits, whereas C4.5 algorithm uses multi-way splits. In order to reduce the size of the decision tree, C4.5 uses post-pruning technique; whereas an optimizer combines the generated rules to eliminate redundancies. The improved version of C4.5 is C5.0, which includes cross-validation and boosting capabilities.

C. Classification and Regression Trees (CART)

CART is a non-parametric decision tree algorithm. It produces either classification or regression trees, based on whether the response variable is categorical or continuous. This methodology is a binary recursive partitioning procedure, which always split the node into only two nodes. The partitioning procedure is repeated for every node of the data until it becomes the terminal node. There are three important steps in CART. (i)Tree growing process: It is based on the recursive partitioning algorithm to select the variables using splitting criterion. In CART, gini criterion is used for determining the best split. Let i(T) denote the impurity at node(T), then i(T) must be zero when node(T) is pure and a maximum when the categories are equally represented. The gini impurity for node T is defined as:

$$i(T) = [1 - \sum_{j} P^2(cj)] \quad (8)$$

where, P (cj) is the fraction of rows in node T with class cj. The reduction in impurity of node T is given by:

$$\Delta i(T) = i(t) - P_L i(T_L) - P_R i(T_R) \quad (9)$$

where, TL and TR - Left and Right child node of nodes, i(TL) and i(TR) are their impurities, and PL and PR are proportion of examples in the child node TL and TR.. Maximum reduction in impurity is chosen as the split point. The splitting process will continue until no further split is possible and the maximal tree is obtained. The process is stopped when there is only single case in each of the terminal nodes or all cases within each terminal node have the same distribution of predictor variables, making splitting impossible. (ii) Tree pruning: There are several reasons involved, which may lead to overfit the data. When a tree is overfitted, it will lead to inaccuracy in estimating prediction errors, which can be overcome by pruning. The type of pruning differs depending upon the application type, i.e., whether the decision tree is used for classification or for prediction or clustering. The popular pruning techniques include cost-complexity pruning, reduced error pruning, pessimistic error pruning, minimum error pruning and minimum description length, bootstrapping etc.

D. Supervised Learning In Quest (SLIQ)

SLIQ stands for Supervised Learning In Quest, it was one of the first scalable algorithms for decision tree induction. This can be implemented in serial and parallel pattern. It is not based on Hunt's algorithm for decision tree classification. It partitions a training data set recursively using breadth-first greedy strategy that is integrated with pre-sorting technique during the tree building phase. SLIQ uses a vertical data format, meaning all values of an attribute were stored as a list, which was sorted at the start of the algorithm. This means that the attributes need not be sorted repeatedly at each node as was the case in existing algorithms like CART and C4.5. With the pre-sorting technique sorting at decision tree nodes is eliminated and replaced with one-time sort, with the use of list data structure for each attribute to determine the best split point.

Calculation of gini index for each possible split point can be done efficiently by storing class distributions in histograms, one per class per node. However SLIQ uses a memory resident data structure called class list which stores the class labels of each record. This data structure limits the size of the datasets SLIQ can handle. In building a decision tree model SLIQ handles both numeric and categorical attributes. One of the disadvantages of SLIQ is that it uses a class list data structure that is memory resident thereby imposing memory restrictions on the data. It uses Minimum Description length Principle in pruning the tree after constructing it. MDL is an inexpensive technique in tree pruning that uses the least amount of coding in producing tree that are small in size using bottom –up technique.

SLIQ decision tree algorithm produces accurate decision trees that are significantly smaller than the trees produced using C4.5 and CART. At the same time, SLIQ executes nearly an order of magnitude faster than CART.

E. Scalable Parallelizable Induction of decision Tree algorithm (SPRINT)

SPRINT stands for scalable parallelizable induction of decision tree algorithm. It was introduced by Shafer et al in 1996. It presents a more memory efficient version of SLIQ. This algorithm associates the class label along with the record identifier for each value in the attribute lists. It is an enhancement of SLIQ as it can be implemented in both serial and parallel pattern for good data placement and load balancing. It is fast, scalable decision tree classifier. It is not based on Hunt's algorithm in constructing the decision tree, rather it partitions the training data set recursively using breadth-first greedy technique until each partition belong to the same leaf node or class. It can be implemented in both serial and parallel pattern for good data placement and load balancing.

It uses two data structure: attribute list and histogram which is not memory resident making sprint suitable for large data sets, thus it removes all the data memory restrictions on data. It handles both continuous and categorical attributes.

IV. PRUNING METHODS

Pruning is a method that reduces the size of the tree by removing parts of the tree that are not meaningful to avoid unnecessary complexity and to avoid over-fitting of the dataset. Pruning decision trees is a fundamental step in optimizing the computational efficiency as well as classification accuracy of such a model. Pruning is done after the complete creation of the tree. It aims to reduce classification errors caused by specialization in the training set. It makes the tree more general. There are various techniques for pruning decision trees

TABLE I. COMPARATIVE ANALYSIS OF DECISION TREE METHODS

Algorithms	**Measures**	**Attributes**	**Procedure**	**Pruning**
ID3	Entropy info-gain	Categorical	Top-down decision tree construction	Pre-pruning using a single pass algorithm
C4.5	Gain ratio	Continuous, Categorical, & Missing values	Top-down decision tree construction	Post-pruning using a single pass algorithm
CART	Gini diversity index	Numeric & Categorical	Constructs binary decision tree	Post pruning based on cost-complexity measure
SLIQ	Gini index	Numeric & Categorical	Decision tree construction in a breadth first manner	Post-pruning based on MDL principle
SPRINT	Gini index	Continuous & Categorical	Decision tree construction in a breadth first manner	Post-pruning based on MDL principle

A. Cost Complexity Pruning

Cost Complexity Pruning also known as weakest link pruning or error complexity Pruning, it proceeds in two stages. In the first stage, a sequence of trees T0, T1 . . . Tk are built on the training data where T0 is the original tree before pruning and Tk is the root tree. In second stage, one of these trees is chosen as the pruned tree, based on its generalization error estimation. Generalization error is defined as the misclassification rate over the distribution D.

If the given dataset is large enough the authors suggests to break it into training set and pruning set. The trees are constructed using the training set and evaluated on the pruning set. On the other hand, if the given dataset is not large enough they propose to use cross-validation methodology, despite the computational complexity implications.

B. Reduced-Error Pruning

Reduced Error Pruning is simple procedure for pruning decision trees. While traversing over the internal nodes from the bottom to the top, the procedure checks for each internal node, whether replacing it with the most frequent class does not reduce the tree's accuracy. In this case, the node is pruned. The procedure continues until any further pruning would decrease the accuracy. It uses pruning set in order to estimate the accuracy. It can be shown that this procedure ends with the smallest accurate sub tree with respect to a given pruning set.

C. Minimum Error Pruning (MEP)

Minimum Error Pruning has been proposed by Niblett and Bratko. It performs bottom-up traversal of the internal nodes. In each node it compares the l-probability error rate estimation with and without pruning. The l-probability error rate estimation is a correction to the simple probability estimation using frequencies.

D. Pessimistic Pruning

Pessimistic pruning avoids the need of pruning set or cross validation and uses the pessimistic statistical correlation test instead. The basic idea is that the error ratio estimated using the training set is not reliable enough. Instead a more realistic measure known as continuity correction for binomial distribution should be used. The pessimistic pruning procedure performs top-down traversing over the internal nodes. If an internal node is pruned then all its descendants are removed from the pruning process, resulting in a relatively fast pruning.

E. Error-based Pruning

Error-based pruning is a simple method of pruning decision trees. Error-Based Pruning is an evolution of the pessimistic pruning. It is implemented in the well-known C4.5 algorithm. It uses the training set error at a node and does not require a validation set. Error-based pruning (EBP) uses the error that is made at a node of the tree on the training data in an estimate of the test set error at that node. The degree of pruning is controlled by the certainty factor CF parameter. One objection to error-based pruning is that it has the general effect of under-pruning.

- If CF is lower then pruning will be done
- If CF is higher then no pruning will be done

superscripts. Do not put footnotes in the reference list. Use letters for table footnotes. put footnotes in the reference list. Use letters for table footnotes.

TABLE II. COMPARATIVE ANALYSIS OF PRUNING METHODS

Methods	**Steps**	**Pruning Set**	**Traversal strategy**	**Operator**	**Features**
Cost Complexity Pruning	2	Yes	All node considered together	Pruning any branch	Can use cross validation sets to estimate accuracy
Reduced Error pruning	1	Yes	Bottom up	Pruning any branch	Find optimal prune tree w.r.t pruning sets
Pessimistic Pruning	1	No	Top down	Pruning any branch	Uses the continuity correction
Minimum Error Pruning	2	Yes	Bottom up	Pruning any branch	Based on m-probability estimate of the error rate
Error-Based Pruning	1	No	Bottom up	Pruning & grafting any branch	Estimate confidence intervals on training data sets

V. CONCLUSIONS

There are various decision tree induction methods available for classification. Among all others, C4.5 algorithm has highest frequency usage, specificity & high accuracy compared to other algorithms because of its simplicity, robustness and effectiveness.

C4.5 and SPRINT algorithms have a good classification accuracy compared to other algorithms used in the study. C4.5 decision tree has the ability to handle data with missing attribute values better than ID3 decision tree algorithm. It also avoids over fitting the data and reduces error pruning. The C4.5 classifier performs better in performance of rules generated and accuracy than ID3 and CART.

REFERENCES

[1] Lior Rokach and Oded Maimon, " Top-Down Induction of Decision Trees Classifiers –A Survey," IEEE Transactions On Systems, Man And Cybernetics-part C: Application And Reviews Vol. 35, no. 4, November 2005.

[2] Anuja Priyama*, Abhijeeta, Rahul Guptaa, Anju Ratheeb, and Saurabh Srivastavab ," Comparative Analysis of Decision Tree Classification Algorithms," International Journal of Current Engineering and Technology, ISSN 2277 - 4106 , Vol.3, No.2 (June 2013), 1 June 2013..

[3] P Venkatesan, N R Yamuna, "Treatment Response Classification in Randomized Clinical Trials: A Decision Tree Approach," Indian Journal

of Science and Technology, ISSN:0974-6846, Vol: 6, Issue: 1, January 2013.

[4] J. Han, and M. Kamber, "Data Mining Concepts and Techniques". Morgan Kaufmann Publishers Inc., United States of America, 2001.

[5] M. N. Anyanwu, and S. G. Shiva, "Comparative Analysis of Serial Decision Tree Classification Algorithms", Vol (3) Issue (3),Pages 230-239, 2009.

[6] S. R. Safavin and D. Landgrebe. A survey of decision tree classifier methodology. IEEE Trans. on Systems, Man and Cybernetics, 21(3):660- 674, 1991.

[7] J. R. Quinlan, C4.5: Programs For Machine Learning. Morgan Kaufmann, Los Altos, 1993.