



International Journal of Emerging Technologies in Computational and Applied Sciences (IJETCAS)

www.iasir.net

Performance Analysis of Classification of Data using Structured Induction Decision Tree Algorithms

Dr.M.Kumarasamy

Professor in Computing Department

ADAMA SCIENCE AND TECHNOLOGY UNIVERSITY (ASTU),

Ethiopia/Africa

Abstract: Decision Trees are one of the most popular Data Mining models. Decision Trees are able to provide a set of rules which improves the user's understanding. Decision Trees represent variables and variable values as trees, branches and leaves from which Decision Rules must be transformed. A Decision Tree classifier builds a model by recursively dividing the training set into partitions so that all or most of the records in a partition have the same class label. Most Decision Tree classifiers perform classification in two phases: tree-induction and tree-pruning. In the tree-induction phase the algorithm starts with the whole data set at the root node. The data set is partitioned according to a splitting criterion into subsets. This procedure is repeated recursively for each subset until each subset contains only members belonging to the same class or is sufficiently small. In the tree-pruning phase the full grown tree is cut back to prevent over-fitting and to improve the accuracy of the tree. The most time-consuming part of Decision Tree induction is obviously the choice of the best attribute selection measure. In this paper we propose decision tree induction algorithm for categorizing the data. We observe from the experimental results that our techniques work well on high dimensional categorical data in which attributes contain distinct values less than ten. For large quantities of categorical values, discretization technique is necessary. In this paper we describe the frequently used decision tree algorithms ID3, C4.5, CART, machine learning concepts and importance of ontologies in classification concepts.

Keywords: Decision Tree, Classification, machine learning, ontology

I. Introduction

Inductive learning algorithms (e.g., decision tree learning) offer a powerful approach to data-driven discovery of complex, a-priori unknown relationships (e.g., classifiers) from data. Most learning algorithms for data-driven induction of pattern classifiers (e.g., the decision tree algorithm), typically represent input patterns at a single level of abstraction – usually in the form of an ordered tuple of attribute values. They typically assume that each pattern belongs to one of a set of disjoint classes. Classification is one of the key data mining technique and has been applied to numerous applications. Until now, a number of different classification techniques have been proposed and the induction of decision trees is a well-known approach. A decision tree is a representation of a decision procedure for determining the class of a given instance and it can be constructed by the non-incremental tree-induction algorithm or the incremental tree-induction algorithm.

II. Decision Trees

Decision trees are powerful and popular tools for classification and prediction. The attractiveness of decision trees is due to the fact that, in contrast to neural networks, **decision trees represent rules**. Rules can readily be expressed so that humans can understand them or even directly used in a database access language like SQL so that records falling into a particular category may be retrieved. In some applications, the accuracy of a classification or prediction is the only thing that matters. In such situations we do not necessarily care how or why the model works. In other situations, the ability to explain the reason for a decision, is crucial. In marketing one has describe the customer segments to marketing professionals, so that they can utilize this knowledge in launching a successful marketing campaign. This domain experts must recognize and approve this discovered knowledge, and for this we need good descriptions. There are a variety of algorithms for building decision trees that share the desirable quality of interpretability. A well known and frequently used over the years is C4.5

A. Types of Decision Trees

Decision tree has three other names:

- Classification tree analysis is a term used when the predicted outcome is the class to which the data belongs.
- Regression tree analysis is a term used when the predicted outcome can be considered a real number (e.g. the price of a house, or a patient's length of stay in a hospital).

- CART analysis is a term used to refer to both of the above procedures. The name CART is an acronym from the words Classification And Regression Trees, and was first introduced by Breiman et al.

B. Tree Construction Formulas

1. Gini impurity

Used by the **CART** algorithm (Classification and Regression Trees). It is based on squared probabilities of membership for each target category in the node. It reaches its minimum (zero) when all cases in the node fall into a single target category. Suppose y takes on values in $\{1, 2, \dots, m\}$, and let $f(i, j)$ = frequency of value j in node i . That is, $f(i, j)$ is the proportion of records assigned to node i for which $y = j$.

$$I_G(i) = 1 - \sum_{j=1}^m f(i, j)^2 = \sum_{j \neq k} f(i, j)f(i, k)$$

2. Information gain

Used by the ID3, C4.5 and C5.0 tree generation algorithms. Information gain is based on the concept of entropy used in information theory.

$$I_E(i) = - \sum_{j=1}^m f(i, j) \log_2 f(i, j)$$

III. Decision Tree Algorithms

A. ID3 (Iterative Dichotomiser 3)

This is a decision tree algorithm introduced in 1986 by Quinlan Ross. It is based on Hunt's algorithm. The tree is constructed in two phases. The two phases are tree building and pruning. ID3 uses information gain measure to choose the splitting attribute. It only accepts categorical attributes in building a tree model. It does not give accurate result when there is noise. To remove the noise pre-processing technique has to be used. To build decision tree, information gain is calculated for each and every attribute and select the attribute with the highest information gain to designate as a root node. Label the attribute as a root node and the possible values of the attribute are represented as arcs. Then all possible outcome instances are tested to check whether they are falling under the same class or not. If all the instances are falling under the same class, the node is represented with single class name, otherwise choose the splitting attribute to classify the instances. Continuous attributes can be handled using the ID3 algorithm by discretizing or directly, by considering the values to find the best split point by taking a threshold on the attribute values. ID3 does not support pruning.

B. C4.5

This algorithm is an extension to ID3 developed by Quinlan Ross. It is also based on Hunt's algorithm. C4.5 handles both categorical and continuous attributes to build a decision tree. In order to handle continuous attributes, C4.5 splits the attribute values into two partitions based on the selected threshold such that all the values above the threshold as one child and the remaining as another child. It also handles missing attribute values. C4.5 uses Gain Ratio as an attribute selection measure to build a decision tree. It removes the biasness of information gain when there are many outcome values of an attribute. At first, calculate the gain ratio of each attribute. The root node will be the attribute whose gain ratio is maximum. C4.5 uses pessimistic pruning to remove unnecessary branches in the decision tree to improve the accuracy of classification.

C. CART

CART stands for Classification And Regression Trees introduced by Breiman. It is also based on Hunt's algorithm. CART handles both categorical and continuous attributes to build a decision tree. It handles missing values. CART uses Gini Index as an attribute selection measure to build a decision tree. Unlike ID3 and C4.5 algorithms, CART produces binary splits. Hence, it produces binary trees. Gini Index measure does not use probabilistic assumptions like ID3, C4.5. CART uses cost complexity pruning to remove the unreliable branches from the decision tree to improve the accuracy.

IV. Machine Learning

The idea of learning can be interpreted in many different ways: is it the act of acquiring knowledge; or becoming aware of our environment based on that knowledge; or the ability to reason based on that knowledge? Finding a universally agreed upon definition is difficult because the philosophical, theoretical, and technical definitions concentrate on various aspects of the meaning. Therefore taking philosophical and theoretical matters aside, this paper deals with the definition of learning adopted by the machine learning community, mainly any computational approach to learning. Because of the data classification nature of this paper's topic, it specifically concentrates on the definition adopted by the data-mining field, that is, "a technique for finding and

describing structural patterns in data for helping to explain that data and make predictions from it". This paper employs decision trees as the data-mining algorithm to capture knowledge. As a data structure, decision trees are used to represent the logical structures of classification rules for domain specific empirical data. The basic algorithm selects the property with the highest information gain for a particular class, and creates disjoint subsets based on that property's values. Ordinal attributes are split into two branches on the $<$ and $\geq$ number restriction. For example a *size* property could be split into large and small classes based on the number of records and their size values. Nominal attributes are treated as categorically disjoint sets, with as many branches as there are values. For example, the taxonomical *R* relation "instance of" (India is an instance of Country) would be able to express the ontological relation $x R y : [x \in \{\text{Country}\} \wedge y \in \{\text{India, USA, UK}\}]$. A decision tree classifying Country would be represented with a parent node Country, and three subnodes, India, USA, and UK. These could be further split on an ordinal property such as population size ranges, or another nominal property such as language. These subsets are smaller in cardinality than the previous subset, but more exact in precision in classifying a concept. The key factor in the classifying process is the property and value combinations that identify concepts best and make up the classification rules. If we are to use decision trees for matching purposes, we must ensure the trees are some what consistent, despite being created by different people who may be using different data sets. We can assume the algorithm producing the decision tree is the same as well as the process of converting the result into the final ontology language, and accommodating axioms. What we need is to ensure that the randomness of the algorithm in the learning process has minimal impact on the consistency of the resulting trees for the two ontologies being matched. To minimize these inconsistencies, instead of producing a single decision tree to model each ontology's data set, several smaller decision trees are created, and combined using various techniques. The resulting decision tree is an intelligently generalized version, which does not lose any information but rather is easier to compare to a tree created using a different data set with similar information. Each technique generalizes the set of trees by iteratively combining them in various ways.

V. Importance of Ontologies

Ontological analysis clarifies the structure of knowledge. Given a domain, its ontology forms the heart of any system of knowledge representation for that domain. Without ontologies, or the conceptualizations that underlie knowledge, there cannot be a vocabulary for representing knowledge. Thus, the first step in devising an effective knowledge representation system, and vocabulary, is to perform an effective ontological analysis of the field, or domain. Weak analyses lead to incoherent knowledge bases. An example of why performing good analysis is necessary comes from the field of databases. Consider a domain having several classes of people (for example, students, professors, employees, females, and males). This study first examined the way this database would be commonly organized: students, employees, professors, males, and female would be represented as *types-of* the class humans. However, some of the problems that exist with this ontology are that students can also be employees at times and can also stop being students. Further analysis showed that the terms students and employee do not describe categories of humans, but are roles that humans can play, while terms such as females and males more appropriately represent subcategories of humans. Therefore, clarifying the terminology enables the ontology to work for coherent and cohesive reasoning purposes. Second, ontologies enable knowledge sharing. Suppose we perform an analysis and arrive at a satisfactory set of conceptualizations, and their representative terms, for some area of knowledge—for example, the electronic-devices domain. The resulting ontology would likely include domain-specific terms such as transistors and diodes; general terms such as functions, causal processes, and modes; and terms that describe behavior such as *voltage*. The ontology captures the intrinsic conceptual structure of the domain. In order to build a knowledge representation language based on the analysis, we need to associate terms with the concepts and relations in the ontology and devise a syntax for encoding knowledge in terms of the concepts and relations. We can share this knowledge representation language with others who have similar needs for knowledge representation in that domain, thereby eliminating the need for replicating the knowledge-analysis process. Shared ontologies can thus form the basis for domain-specific knowledge-representation languages. In contrast to the previous generation of knowledge-representation languages (such as KL-One), these languages are *content-rich*; they have a large number of terms that embody a complex content theory of the domain. Shared ontologies let us build specific knowledge bases that describe specific situations. For example, different electronic devices manufacturers can use a common vocabulary and syntax to build catalogs that describe their products. Then the manufacturers could share the catalogs and use them in automated design systems. This kind of sharing vastly increases the potential for knowledge reuse.

VI. Classification of Data using Decision Tree Methodology

Our proposed system has been named discrete-tree induction to enunciate our intention to design a decision tree induction method to handle categorical data containing numerous discrete values. The framework as shown in Figure1 is composed of the discrete-tree component, which is the main decision tree induction part, and the testing component responsible for evaluating the accuracy of the decision tree model as well as reporting some

statistics such as tree size and running time. Categorical value handling of our discrete-tree induction method can be achieved through the selection of the representative data, instead of learning from each and every training data. These selected data are used further in the tree building phase. Training data are first clustered by clustering module to find the mean point of each data group. The data selection module then uses these mean points as a criterion to select the training data representatives. It is a set of data representatives that to be used as input of the tree induction phase. Heuristics have to be applied as a threshold in the selection step and as a stopping criterion in the tree building phase. The algorithms of a main module as well as the clustering, data selection, and tree induction modules are presented in Figures 2-5, respectively.

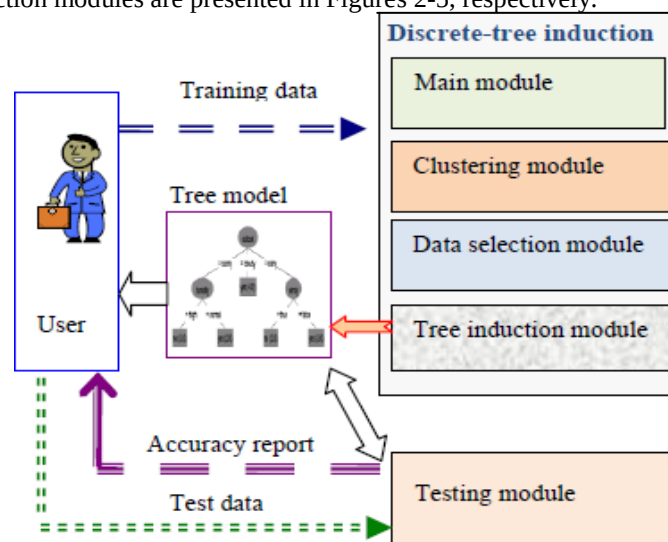


Figure 1. A decision tree induction framework.

Input: Data D with class label and **Output:** A tree model M

Steps:

1. Read D and extract class label to check distinctive values K
2. Cluster D to group data into K groups
3. In each group
 - 3.1 Get mean attribute values
 - 3.2. Compute similarity of each member compared to its mean
 - 3.3 Compute average similarity and variance
 - 3.4 Set threshold $T = 2 * \text{Variance}$
 - 3.5 Select only data with similarity $> T$
4. Set stopping criteria S for tree building as $S = K - \log [(\text{number of removed data} + K) / |D|]$
5. Send selected data and criteria S into tree-induction module
6. Return a tree model

Figure 2. Discrete-tree induction main algorithm.

Steps:

1. Initialize K means
2. Call `find_clusters(K, Instances, Means)`
3. Call `find_means(K, Instances, NewMeans)`
4. If $\text{Means} \neq \text{NewMeans}$ Then repeat step 2
5. Output mean values and instances in each clusters

Figure 3. Categorical data clustering algorithm.

Steps:

1. For each data cluster
2. Compute similarity of each member compared to cluster mean
3. Computer average similarity score of a cluster
4. Computer variance on similarity of a cluster
5. Threshold = $2 * \text{variance}$
6. Remove member with similarity score $< \text{Threshold}$
7. Return K clusters with selected data

Figure 4. Data selection algorithm.

Steps:

1. If data set is empty
2. Then `Assert(node(leaf,[Class/0], ParentNode)`

3. Exit
4. If number of data instances < MinInstances
5. Then Compute distribution of each class
6. Assert(node(leaf, ClassDistribution, ParentNode))
7. If all data instances have the same class label
8. Then Assert(node(leaf, ClassDistribution, ParentNode))
9. If data > MinInstances and data have mixing class labels
10. Then Build Subtree
11. If data attributes conflict with the existing attribute values of a tree
12. Then stop growing and create a leaf node with mixing class labels
13. Return a decision tree

Figure 5. Tree building algorithm.

VII. Experimentation and Results

The Experimental data is collected from UCI Machine Learning Repository, which is publicly available. The results were analysed using Weka tool on the data using 10-fold cross validation to test the accuracy and time complexity of classifiers. To test the accuracy of the proposed discrete-tree induction system, we use the standard UCI data repository including the Wisconsin breast cancer, SPECT heart, DNA splice-junction, and audiology data sets. Each data set is composed of two separate subsets of training and test data. We then run the discrete-tree program and observe the results comparing to other learning algorithms, namely C4.5, Naive Bayes, k-Nearest Neighbor, and support vector machine. The comparison results are graphically shown in Figure 6. It can be noticed from the results that the discrete-tree induction method shows considerably accurate prediction on SPECT heart data set. On Wisconsin breast cancer and DNA splice-junction data sets, our algorithm is as good as the other learning algorithms. But the discrete-tree induction performs poorly on audiology dataset. The poor performance may be due to the fact that such data set contains a single value on many attributes causing our data selection scheme making a poor set of samples. For a SPECT heart data set, we provide tree model obtained from our algorithm to compare against the model obtained from the C4.5 algorithm.

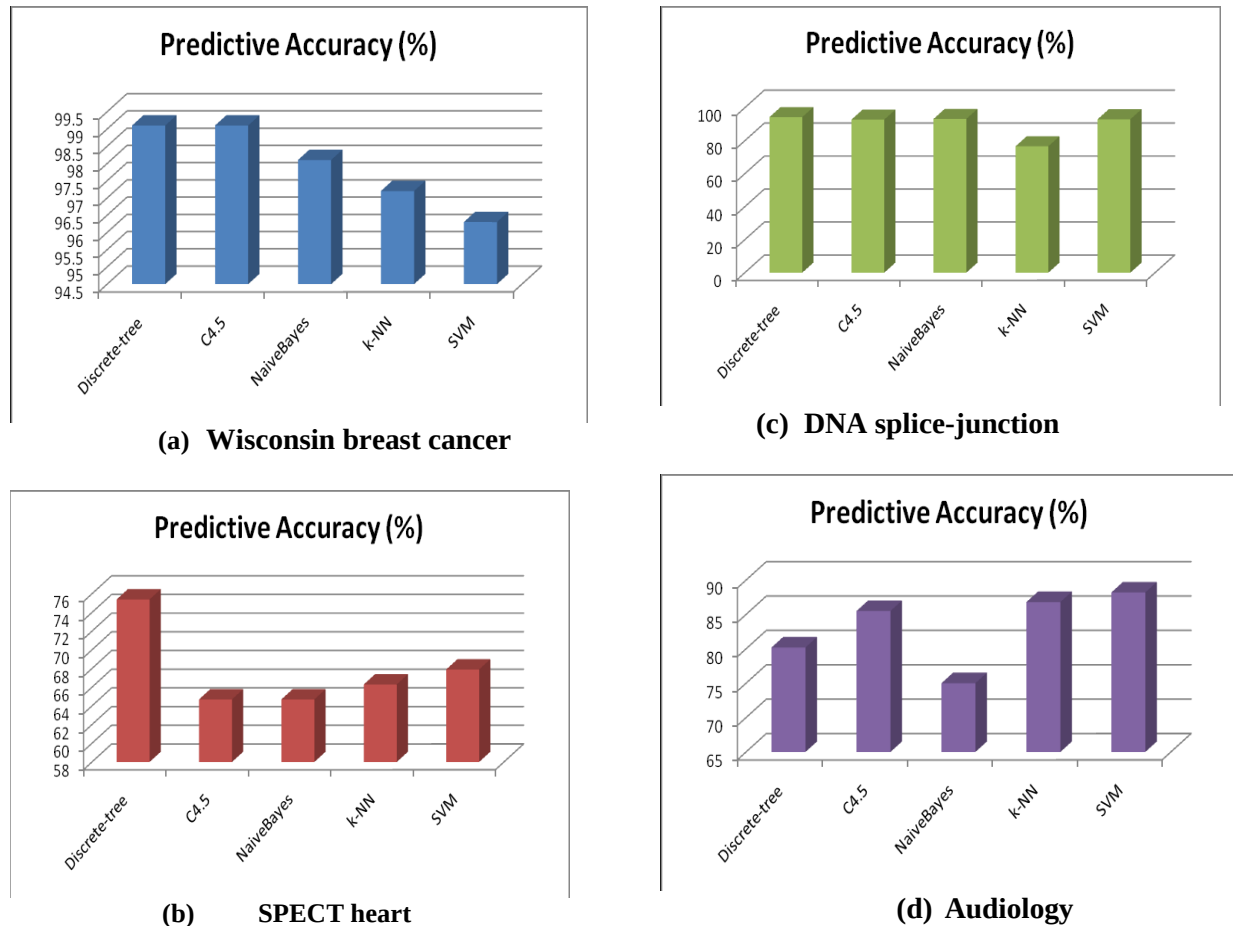


Figure 6 Comparison results of prediction accuracy.

VIII. Conclusion

Decision trees are the single most popular data mining tool which is easy to understand, implement, use and cheap. They do classification and predict a categorical output from categorical and/or real inputs. Categorical data can cause serious problem to many tree learning algorithms in terms of distorted results and the decrease in predicting performance of the learning results. In this paper, we proposed a methodology to deal with categorical values in a decision tree induction algorithm. Our intuitive idea is to select only potential representatives, rather than applying the whole training data that some values are highly dispersed, to the tree induction algorithm. Data selection process starts with clustering in order to obtain the mean point of each data group. For each data group, the heuristic $T = 2 * \text{Varianceof-cluster-similarity}$ will be used as a threshold to select only data around mean point within this T distance. Data that lie far away from the mean point are considered prone to noise and outliers; we thus remove them. The removed data still play their role as one factor of a tree building stopping criterion, which can be formulated as $S = K - \log[(\text{number of removed data instances} + K) / |D|]$, where K is the number of clusters, which has been set to be equal to the number of class labels, and D is the number of training data. From experimental results, it turns out that our heuristic-based decision tree induction method produces a good predictive model on categorical data set. It also produces a compact tree model. With such promising results, we thus plan to improve our methodology to be incremental such that it can learn model from streaming data.

References

- [1] L. A. Badulescu, Data Mining Algorithms Based On Decision Trees, Annals of the Oradea University. Fascicle of Management and Technological Engineering, Publishing House of Oradea University, Vol. V (XV), ISSN 1583 - 0691, pages 1621-1628, 2006.
- [2] K. Blackmore, T. Bossomaier, S. Foy and D. Thomson, Data Mining of Missing Persons Data, S.K. Halgamuge and L. Wang (eds.), Classification and Clustering for Knowledge Discovery, Studies in Computational Intelligence, vol. 4, Springer-Verlag Berlin Heidelberg, ISBN 3-540-26073-0, page 309, 2005.
- [3] J. R. Quinlan, C4.5: Programs for Machine Learning. Morgan Kauffman, 1993.
- [4] K. Liu, K. Bhaduri, K. Das, P. Nguyen, and H. Kargupta, "Client-side Web Mining for Community Formation in Peer-to-Peer Environments," *SIGKDD Explorations*, vol. 8, no. 2, pp. 11–20, 2006.
- [5] E. Alfaro, N. Garca, M. Gamez, and D. Elizondo. Bankruptcy forecasting: An empirical comparison of adaboost and neural networks. *Decision Support Systems*, 45(1):110-122, 2008.
- [6] Algowiki.net. An implementation of the Floyd-Warshall algorithm in Java, 2011. Visited on November 1st, 2011.
- [7] V. Papataxiarhis, V. Tsetsos, I. Karali, and P. Stamatiopoulos, "Developing rule-based applications for the Web: Methodologies and Tools", in *Handbook of Research on Emerging Rule-Based Languages and Technologies: Open Solutions and Approaches*, A. Giurca, D. Gasevic, and K. Taveter, eds., Information Science Reference, 2009.
- [8] F. Giunchiglia, P. Shvaiko, and M. Yatskevich, "S-Match: an Algorithm and an Implementation of Semantic Matching," in *Proc. European Semantic Web Symposium (ESWS'04)*, 2004, pp. 61-75.
- [9] Kuowj, Chang RF, Chen DR and Lee CC, Data Mining with decision trees for diagnosis of breast tumor in medical ultrasonic images, March 2001.
- [10] H. Ren, —Clinical diagnosis of chest pain, *Chinese Journal for Clinicians*, vol. 36, 2008.
- [11] UC Irvine Machine Learning Repository, www.ics.uci.edu/~mllearn/MLRepository.html
- [12] J.R. Quinlan, c4.5: Programs for Machine Learning, Morgan Kaufmann Publishers, Inc, 1992.
- [13] J. Han and H. Kamber, *Data Mining: Concepts and Techniques*, 2nd ed., Morgan Kaufmann, 2006.