A Decision Tree Algorithm for Uncertain Data

K. Anuradha, N. Tulasi Radha, T. Pavan Kumar

Abstract— Classification is a classical problem in machine learning and data mining. Given a set of training data tuples, each having a class label and being represented by a feature vector, the task is to algorithmically build a model that predicts the class label of an unseen test tuple based on the tuple's feature vector. One of the most popular classification models is the decision tree model. Decision trees are popular because they are practical and easy to understand. Rules can also be extracted from decision trees easily. Tree learning algorithms can generate decision tree models from a training data set. When working on uncertain data or probabilistic data, the learning and prediction algorithms need handle the uncertainty cautiously, or else the decision tree could be unreliable and prediction results may be wrong. This paper presents a new decision tree algorithm for handling uncertain data.

Index Terms—Classification, Decision tree, Prediction, Uncertain data.

I. INTRODUCTION

Decision trees are powerful and popular tools for classification and prediction[1]. 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[2] and ID3[3].

Decision tree is a classifier in the form of a tree structure (Figure 1), where each node is either:

- a *leaf node* - indicates the value of the target attribute (class) of examples, or
- a *decision node* - specifies some test to be carried out on a single attribute-value,

With one branch and sub-tree for each possible outcome of the test.

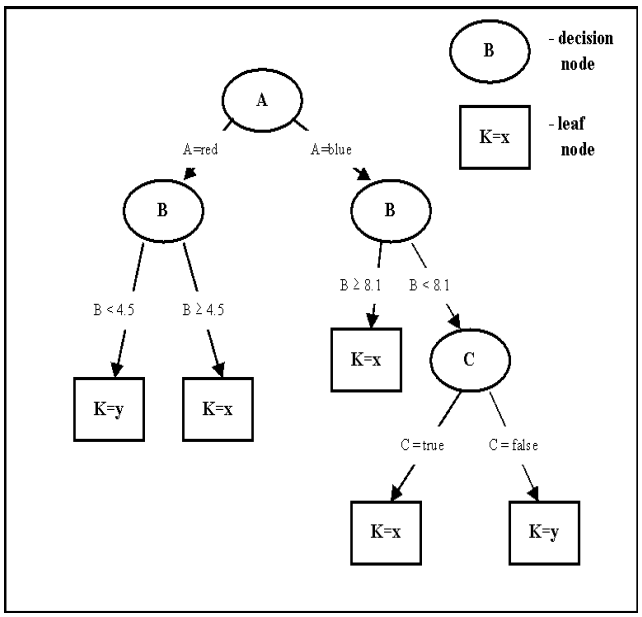

Figure 1: An example of a simple decision tree

A decision tree can be used to classify an example by starting at the root of the tree and moving through it until a leaf node, which provides the classification of the instance.

Decision tree induction[4] is a typical inductive approach to learn knowledge on classification. The key requirements to do mining with decision trees are:

- *Attribute-value description*: object or case must be expressible in terms of a fixed collection of properties or attributes. This means that we need to discretize continuous attributes, or this must have been provided in the algorithm.
- *Predefined classes (target attribute values)*: The categories to which examples are to be assigned must have been established beforehand (supervised data).
- *Discrete classes*: A case does or does not belong to a particular class, and there must be more cases than classes.
- *Sufficient data*: Usually hundreds or even thousands of training cases.

Decision Tree is a widely used data classification technique for both certain and uncertain data. Data uncertainty arises in many applications during the data collection process. Example sources of uncertainty include measurement/quantization errors, data staleness, and multiple repeated measurements. With uncertainty, the value of a data item is often represented not by one single value, but by multiple values forming a probability distribution. This paper presents a decision tree algorithm for handling uncertain data.

Manuscript received March 31, 2012

K.Anuradha, MCA., (M.Tech) KAUSHIK Engineering College, A.P., India. (Email : anuradha.kadimisetty@gmail.com).

N.Tulasi Radha, M.Tech, KAUSHIK Engineering College, A.P., India. (Email : radhi789@gmail.com).

T. Pavan Kumar, M.CA, M.Tech, KAUSHIK Engineering College, A.P., India. (Email : chanti.legend@gmail.com).

II. DECISION TREE INDUCTION ALGORITHM

The algorithm is shown below:

Algorithm DTU Induction:

input: the training dataset D; the set of candidate attributes att-list

output: An uncertain decision tree begin

1: create a node N;
2: if (D are all of the same class, C) then
3: return N as a leaf node labeled with the class C;
4: else if (attribute-list is empty) then
5: return N as a leaf node labeled with the highest weight class in D;
6: end if ;
7: select a test-attribute with the highest probabilistic information gain ratio to label node N;
8: if (test-attribute is numeric or uncertain numeric) then
9: binary split the data from the selected position y;
10: for (each instance Rj) do
11: if (test-attribute <= y) then
12: put it into Dl with weight Rj .w;
13: else if (test-attribute > y) then
14: put it into Dr with weight Rj .w;
15: else
16: put it into Dl with weight Rj .w* $\int_{x1}^{y} f(x)dx$;
17: put it into Dr with weight Rj .w * $\int_{y}^{x2} f(x)dx$;
18: end if ;
19: end for;
20: else
21: for (each value ai(i = 1, . . . , n) of the attribute) do
22: grow a branch Di for it;
23: end for;
24: for (each instance Rj) do
25: if (test-attribute is uncertain) then
26: put it into Di with Rj .ai.w * Rj .w weight;
27: else
28: put it into a certain Di with weight Rj .w;
29: end if
30: end for;
31: end if ;
32: for each Di do
33: attach the node returned by DTU(Di, att-list);
34: end for;

end

The basic strategy is as follows:

1. The tree starts as a single node representing the training samples (step 1).

2. If the samples are all of the same class; then the node becomes a leaf and is labeled with that class (steps 2 and 3).

3. Otherwise, the algorithm uses a probabilistic entropy-based measure, known as the probabilistic information gain ratio, as the criteria for selecting the attribute that will best separate the samples into an individual class (step 7). This attribute becomes the ”test” attribute at the node.

4. If the test attribute is numerical or uncertain numerical, we split for the data at the selected position y (steps 8 and 9).

5. A branch is created for test-attribute ≤ y or test-attribute > y respectively. If an instance's test attribute value [x1, x2] is less than or equal to y (x2 ≤ y), it is put into the left branch with the instance's weight Rj .w. If an instance's test attribute value [x1, x2] is larger than y (x1 > y), it is put into the right branch with the instance's weight Rj .w. If an attribute's value [x1, x2] covers the split point y (x1 ≤ y < x2), it is put into the left branch with weight Rj .w∗ $\int_{x1}^{y} f(x)dx$ and the right branch with weight Rj .w∗ $\int_{y}^{x2} f(x)dx$. Then the dataset is divided into Dl and Dr (steps 10-19).

6. If the test attribute is categorical or uncertain categorical, we split the data multiway (steps 21-30). A branch is created for each value of the test attribute, and the samples are partitioned accordingly. For each value ai of the attribute,an instance is put into Di with Rj .w weight when the attribute is certain. If the attribute is uncertain, assume the probability of the attribute value ai be Rj .ai.p, then the instance is put into the branch ai with the weight Rj .ai.p ∗ Rj .w.

7. The algorithm recursively applies the same process to generate a decision tree for the samples.

8. The recursive partitioning process stops only when either of the following conditions becomes true:

1) All samples for a given node belong to the same class (steps 2 and 3), or

2) There are no remaining attributes on which the samples may be further partitioned (step 4). In this case, the highest weight class is employed (step 5). This involves converting the given node into a leaf and labeling it with the class having the highest weight among samples. Alternatively, the class distribution of the node samples may be stored.

III. DECISION TREE FOR UNCERTAIN DATA

The core issue in a decision tree induction algorithm is to decide the method of records being split. Each step of the tree-grow process needs to select an attribute test condition to divide the records into smaller subsets. A decision tree algorithm must provide a method for specifying the test condition for different attribute types as well as an objective measure for evaluating the goodness of each test condition. There are many measures that can be used to determine the best way to split the records. These measures are usually defined in terms of the class distribution of the records before and after splitting. Widely used splitting measures such as information entropy and Gini index are all based on the purity of the nodes and choose the split those results in the highest node purity. These measures are not applicable to uncertain data. For example, for data in Table I, if the splitting condition is Ai < 115, it cannot be determined whether instances 1, 2, 4, 5, 11 belong to the left or right node. Our approach is that when the cutting point of an attribute lies within the uncertain interval of an instance, the instance is split into both branches and the weights of being in both branches are calculated according to the probability distribution function f(x).

When the probability distribution function f(x) is unavailable, we can use domain knowledge to find the appropriate pdf or assume commonly used distribution such as uniform or Gaussian distribution. An example of such a split is shown in figure 2.

Table 1. An uncertain dataset

ID	A_i	...	A_j	class
1	110-120	...	(V_1: 0.3; V_2:0.7)	No
2	100-120	...	(V_1: 0.3; V_2:0.7)	No
3	60-85	...	(V_1: 0.3; V_2:0.7)	No
4	110-145	...	(V_1: 0.3; V_2:0.7)	No
5	110-120	...	(V_1: 0.3; V_2:0.7)	Yes
6	50-80	...	(V_1: 0.3; V_2:0.7)	No
7	170-250	...	(V_1: 0.3; V_2:0.7)	No
8	85-100	...	(V_1: 0.3; V_2:0.7)	Yes
9	80-100	...	(V_1: 0.3; V_2:0.7)	No
10	120-145	...	(V_1: 0.3; V_2:0.7)	Yes
11	105-125	...	(V_1: 0.3; V_2:0.7)	Yes
12	80-95	...	(V_1: 0.3; V_2:0.7)	No

Table I shows an uncertain data set. Among all the attributes, Ai is an Uncertain Numerical Attribute(UNA) whose precise value is unavailable. We only know the range of the Ai of each tuple. Aj is an Uncertain Categorical Attribute (UGA). It can be either V1 or V2,, each with associated probability.

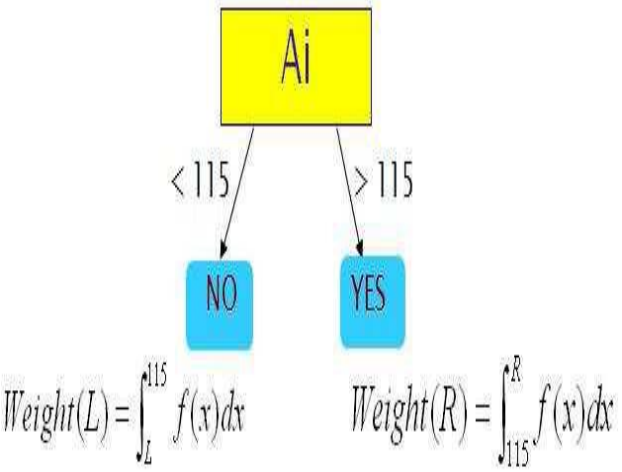

Figure 2. Uncertain numerical data split

Splitting based on an UCA A is an n-ary split, assume attribute A has n possible values ai, (1≤i≤ n), If an uncertain categorical attribute is identified as the best splitting attribute, a branch is created for each known value of the test attribute, and the data are partitioned accordingly. For each value ai of the attribute, the instance is put into all of the branches with the weight equal to the probability of the attribute to be ai, as shown in figure 3.

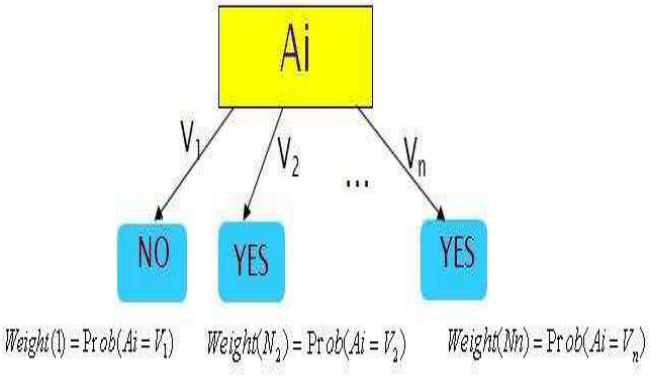

Figure 3. An uncertain categorical Data

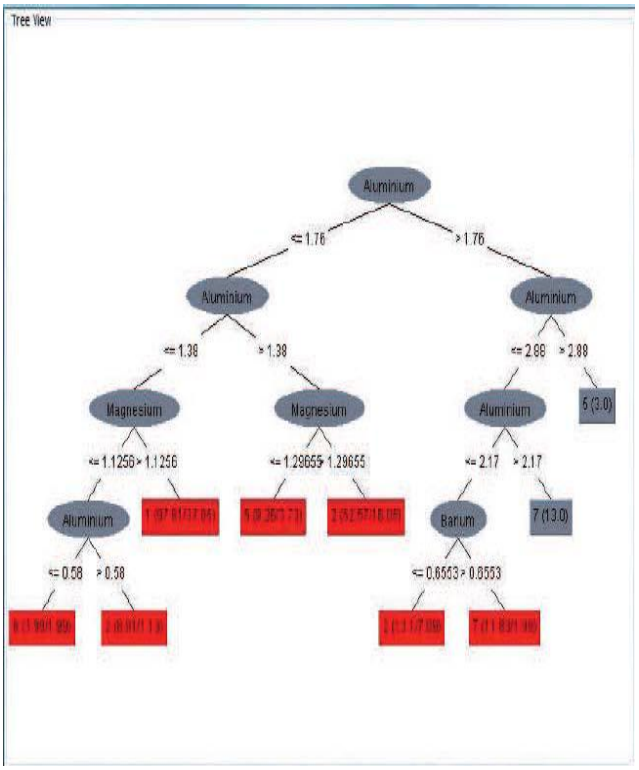

Figure 4. An uncertain decision tree

Figure 4 shows the uncertain decision tree for an uncertain glass dataset and the decision tree determines the type of glass based on the oxide content. In case an uncertain data instance covers a test point, it is split into both branches according to the cumulative probability distributions; our visualization routine highlights those leaf nodes in red. The leaf nodes indicate the class type of the node Ci, followed by two real values x/y. x is the total probabilistic cardinality of the node, that is, the total number of instance fall in that node, and y is the number of false positives, that is, the number of instance fall in that node but not in class Ci. Since both x and y are calculated according to probability distribution function, they are floating-point numbers instead of integers, which is different from traditional decision tree. Detailed algorithm for uncertain decision tree can be found in [1, 2].

IV. CONCLUSION

Decision Tree is a widely used data classification technique for both certain and uncertain data. Data uncertainty arises in many applications during the data collection process. Example sources of uncertainty include measurement/quantization errors, data staleness, and multiple repeated measurements. With uncertainty, the value of a data item is often represented not by one single value, but by multiple values forming a probability distribution. This paper presents a decision tree algorithm for handling uncertain data.

REFERENCES

1. R. Agrawal, T. Imielinski, and A. N. Swami, "Database mining: A performance perspective," IEEE Trans. Knowl. Data Eng., vol. 5, no. 6, pp. 914–925, 1993.
2. C4.5: Programs for Machine Learning. Morgan Kaufmann, 1993, ISBN 1-55860-238-0.
3. J. R. Quinlan, "Induction of decision trees," Machine Learning, vol. 1, no. 1, pp. 81–106, 1986.
4. http://gautam.lis.illinois.edu/monkmiddleware /public/ analytics/ decision tree. html

AUTHORS PROFILE

K.Anuradha, MCA., (M.Tech) . She is pursuing her M.Tech from KAUSHIK College of Engineering, A.P., India. Her research interests includes Computer Networks and Data warehousing & Data Mining.

N. Tulasi Radha,, M.Tech, She is currently working as Assistant Professor in Department of Computer Science & Engineering at KAUSHIK College of Engineering, A.P., India. Her research interests includes Data Mining, Network Security and Image Processing.

T. Pavan Kumar, M.CA, M.Tech, He is currently working as Assistant Professor in Department of Computer Science & Engineering at SANKETIKA Engineering College, A.P., India. His research interests includes Data Mining, Network Security and Image Processing.