ISSN NO. 2320-5407

Journal homepage: http://www.journalijar.com

INTERNATIONAL JOURNAL OF ADVANCED RESEARCH

RESEARCH ARTICLE

Classifying Climate Data (uncertain) using Decision Tree

Khumesh Patil, Namrata Pagare, Pallavi Narkhede, Prashant Brahmankar
Department of Computer Engineering K.K.Wagh Institute of Engg&Research,Nashik
University of Pune

Manuscript Info

Manuscript History:

Received: 12 February 2014
Final Accepted: 12 March 2014
Published Online: April 2014

Key words:
acute coronary syndromes**,** leptin, interleukine-18.

Corresponding Author

Khumesh Patil

Abstract

Traditional decision tree classifiers work with data whose values are known and precise .Here classification of data is done but for un-certain data approximate value is assumed that does not give accurate result. One of the most popular classification models is the decision tree model. Decisions trees are popular because they are practical and easy to understand. In traditional decision-tree classification, a feature of a tuple is either categorical or numerical. Multiple values are formed by Probability Distribution Function (pdf) that represents the uncertainty value. The accuracy of a decision tree classifier can be improved if the pdf is used. Existing decision tree building algorithms are improved to handle data tuples with uncertain values. Pruning techniques are used which improves the efficiency of the construction of decision trees. Proposed system classifies climate data which is uncertain due to measurement errors, to various classes using decision tree.
Index Term -Decision tree, classification, pruning, PDF, Data Mining

I. INTRODUCTION

The classification technique is a systematic approach to build classification models from an input data set. For example, decision tree classifiers, rule-based classifiers, neural networks, support vector machines, and naive Bayes classifiers are different technique to solve a classification problem. Each technique adopts a learning algorithm to identify a model that best fits the relationship between the attribute set and class label of the input data. Therefore, a key objective of the learning

Uncertainty may be caused by measurement errors,data staleness, repeated measurements, limitations of the datacollection process, etc.
In this paper classifiers are used to handle data with uncertain information i.e. with numerical uncertain data. Pdf is probability distribution function. The accuracy of a decision tree classifier can be improved if the pdf is used. Pdf is used to sort out the values of decision tree. Here existing decision tree building algorithms are improved to handle data tuples with uncertain values. In Proposed system, pruning techniques are used which improves the efficiency of the construction of decision trees.

Easy way to handle data uncertainty is to calculate probability distributions by summary statistics such as means and variances. This approach Averaging. Another approach for considering the complete information carried bythe probability distributions to build a decision tree which is called as Distribution-based approach.

II. RELATED WORKS

Uncertain Data mining is very interesting and growing field. The very famous K-means clustering algorithm is modified to UK-means algorithm which handles data uncertainty. This UK-mean algorithm is applicable to any uncertainty region and pdf. Here in this paper Different pruning techniques are used.

From many decades whatever uncertain data is there is considered as missing data and then classification is done on that basis. There are two algorithms ID3 and C4.5 where C4.5 is enhancement of ID3 algorithm which is used for data classification. C4.5 algorithm handles the uncertain data using fractional tuples. Here in this paper tuples are divided into subsets. To builda decision tree on tuples with numerical, pointvalued data is very demanding.

fuzzy decision tree classification, attributes and class labels can be fuzzy and are represented in fuzzy terms in fuzzy decision tree classification. This paper gives

of information available is exploded by

a factor of s. Hopefully, the richer information allows us tobuild a better classification model.

The most challenging task is to construct a decision treebased on tuples with uncertain values, finding suitable A_{jn} and z_n

for each internal node n, as well as an appropriateprobability distribution P_m for each leave node m

IV. ALGORITHMS

Basically we use two algorithms for handling uncertain data

1)Averaging

It is simple way to deal with the uncertain data is to replace each pdf with its expected value,it converting data tuples into point value tuples.

Example

Tuple	class	mean	Probability distribution				
			-10	-1.0	0.0	+1.0	+10
1	A	+2.0		8/11			3/11
2	A	-2.0	1/9	8/9			
3	A	+2.0		5/8		1/8	2/8
4	B	-2.0	5/19	1/19		13/19	
5	B	+2.0			1/35	30/35	4/35
6	B	-2.0	3/11			8/11	

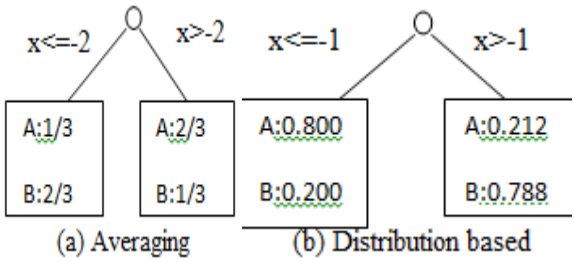


Fig.Decision tree built from example tuples in table 1

The resulting decision tree is shown in fig..now,if we use the 6 tuples in table1 as test tuples,this decision tree will classify tuples 2,4,6 as class "B" (the most likely class label in

L)and rest as "A".Hence it misclassifies tuples 2 and 5.The accuracy is 2/3.

Pdf is used to sort out the data and generate the tuples.Tuples means the collection of elements. Averaging algorithm directly calculate the

affects the structure of the resulting decision tree.

V. PRUNING ALGORITHMS

A. PRUNING EMPTY AND HOMOGENEUS INTERVALS

The attribute with the best split point giving the lowest entropy is taken as the result of BEST SPLIT. We define the set of end-points of tuples in S on attribute A_j as $Q_j = f_q$ j (q = $a_{h;j}$) _ (q = $b_{h;j}$) for some t_h 2 S_g.We assume that there are v such end-points, q_1; q_2; : : : ; q_v,sorted in ascending order. Within [q_1; q_v], we want to find an optimal split point for attribute A_j .

Definition 1: For a given set of tuples S, an optimal split point for an attribute A_j is one that minimises H(z;A_j). (Note that the minimisation is taken over all z_2 [q_1; q_v].) The end-points define v -1 disjoint intervals: (q_i; q_{i+1}] for i = 1; : : : ; v - 1. We will examine each interval separately.For convenience, an interval is denoted by (a; b].

Definition 2 (Empty interval): An interval (a; b] is empty if integration of a to b $f_{h;j}$(x) dx = 0 for all $t_h$belongs to S.

Definition 3 (Homogeneous interval): An interval (a; b] is homogeneous if there exists a class label c belong to C such that integration of a to b $f_{h;j}$(x) dx 6= 0) c_h = c for all t_h belongs to S.

Intuitively, an interval is empty if no pdf's intersect it; an interval is homogeneous if all the pdf's that intersect it come from tuples of the same class.

Definition 4 (Heterogeneous interval): An interval (a; b] is heterogeneous if it is neither empty nor homogeneous.

Definition 5 (Tuple Density): Given a class C,
an attribute A_i , and a set of tuples S, we define the tuple density function $g_{c,j}$as:

$$g_{c,j} = \sum_{t_h \in S : c_h = c} w_h f_{h,j}$$

Where $w_h$is the weight of fractional tuples t_h∈ S.

B. PRUNING BY BOUNDING

First we compute the entropy H(q;A_j) for all end-points q belongs to Q_j . Let H* j = $\min_{q2Qj}$fH(q;A_j)g be the smallest of such end-point entropy values. Next, for each heterogeneous interval (a; b], we compute a lower bound, Lj , of H(z;Aj) over all candidate split points z belongs to (a; b]. If L_j H_{*j} , we know that none of the candidate split points within the interval (a; b] can give an entropy that is smaller than $H_{*\ j}$ and thus the whole interval can be pruned.

We note that the number of end-points is much smaller than the total number of candidate split points. So, if a lot of heterogeneous intervals are pruned in this manner, we can eliminate many entropy calculations. So, the key to this pruning technique is to find a lower bound of H(z;A_j) that
is not costly to compute, and yet is reasonably tight for the pruning to be effective.

C.END POINT SAMPLING

In some settings, UDT-GP reduces the number of "entropy calculations" (including the calculation of entropy values of split points and the calculation of entropylike lower bounds for intervals) to only 2.7% of that of UDT. On a closer inspection, we find that many of these remaining entropy calculations come from the determination of end-point entropy values. In order to further improve the algorithm's performance, we propose a method to prune these end-points.

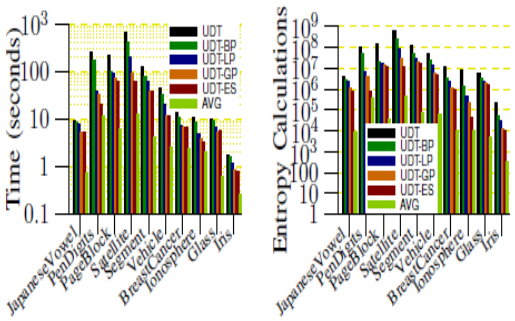
We note that the entropy H(q;A_j) of an end-point q is computed for two reasons. Firstly, for empty and homogeneous intervals, their end-points are the only candidates for theoptimal split point. Secondly, the minimum of all end-point entropy values is used as a pruning threshold. For the latter purpose, we remark that it is unnecessary that we consider all end-point entropy values. We can take a sample of the end-points (say 10%) and use their entropy values to derive a pruning threshold. This
threshold might be slightly less effective as the one derived from all end-points, however, finding it requires much fewer entropy calculations. Also, we can concatenate a few consecutive intervals, say
I1; I2; I3, into a bigger interval I,compute a lower bound.and attempt to prune I. If successful, we have effectively

pruned the end-points of I_1, I_2 and I_3.
We incorporate these End-point Sampling

intervals obtained after pruning is Y_{00} (row 9), which is a much smaller candidate than the set of candidate intervals when no end-point sampling is used. For the candidate intervals in Y_{00}, we compute the values $H(z;A_j)$ for all pdf sample points to find the minimum entropy value.

VI. EXPERIMENTS ON EFFICIENCY

we are going to implement the above algorithm using JDK 1.6 and a bunch of experiments were done on a PC with Intel Core 2 Duo 2.66 GHz processor.



(a) Execution time (b) Pruning effectiveness

Fig. 4. Performance of the pruning algorithms

A. Execution Time:

We will first test execution time of algorithms which are given in Fig (a). AVG constructs different decision from UDT-based algorithms, and that AVG generally constructs algorithms which have low accuracy. The AVG algorithm does not show any information about uncertainty, takes very less time to finish. But it is not as accurate as the distribution-based algorithms. Hence, we conclude that in this experiment, each pdf is shown by 100 sample points (i.e. s=100). All UDT-based algorithms have to handle 99 times more data than AVG, which only process one average for each pdf.

B. Pruning Effectiveness:

Fig (b) shows the number of entropy
calculations done by each algorithm. Since the computation time of the lower bound of an interval is comparable to that of computing an entropy, the number of entropy calculations for UDT-LP, UDT-GP and UDT-ES include the number of lower bounds computed. Figure shows that our pruning techniques are highly effective. Hence, UDT-ES

[8] Y. Yuan and M. J. Shaw, "Induction of fuzzy decision trees," Fuzzy Sets
Syst., vol. 69, no. 2, pp. 125–139, 1995.
[9] T. Elomaa and J. Rousu, "General and efficient multisplitting of numerical attributes," Machine Learning, vol. 36, no. 3, pp. 201–244, 1999.
[10] U. M. Fayyad and K. B. Irani, "On the handling of continuous-valued
attributes in decision tree generation," Machine Learning, 1992. algorithm is to build predictive model that accurately predict the class labels of previously unknown records.
Decision Tree Classifier is a simple and widely used classification technique. It applies a straitforward idea to solve the classification problem. Decision Tree Classifier poses a series of carefully crafted questions about the attributes of the test record. Each time it receives an answer, a follow-up question is asked until a conclusion about the class label of the record is reached. The decision tree classifiers organized a series of test questions and conditions in a tree structure. In the decision tree, the root and internal nodes contain attribute test conditions to separate records that have different characteristics. Build an optimal decision tree is key problem in decision tree classifier. In general, may decision trees can be constructed from a given set of attributes. While some of the trees are more

accurate than others, finding the optimal tree is computationally infeasible because of the exponential size of the search space.

However, various efficient algorithms have been developed to construct a reasonably accurate, decision tree in a reasonable amount of time. These algorithms usually employ a greedy strategy that grows a decision tree by making a series of locally optimum decisions about which attribute to use for partitioning the data. For example, Hunt's algorithm, ID3, C4.5, CART, SPRINT are greedy decision tree induction algorithms.

In computer science, **uncertain data** is the data that contains specific uncertainty. Uncertain data is typically found in the area of sensor networks. When representing such data in a database, some indication of the probability of the various values.

classification results as a distribution:

A distribution telling how likely it belongs to each class,for each test tuple is given by us.

III. PROBEM DEFINATION

Here, a dataset consists of d training tuples, $\{t_1, t_2, \ldots, t_d\}$, and k numerical (real-valued) feature attributes, $A_1, \ldots A_k$. Each tuple $t_i$is associated with a feature vector $V_i = (f_{i,1}, f_{i,2}, \ldots, f_{i,k})$ and a class label $c_i \in C$.Here, each $f_{i,j}$is a pdf with domain $[a_{i,j}, b_{i,j}]$ modelling theuncertain value of attribute $A_j$in tuple t_i. The classificationproblem is to construct a model M that maps each featurevector $(f_{x,1}, \ldots, f_{x,k})$ to a probability distribution $P_x$on Csuch that given a testtuple $t_0 = (f_{0,1}, \ldots, f_{0,k}, c_0)$, $P_0 = M(f_{0,1}, \ldots, f_{0,k})$ predicts the class label c_0 with high accuracy. We say that P_0 predicts c_0 if $c_0 = \text{argmax}_{c \in C}\{P_0(c)\}$. Here binary decision trees are considered with tests on numericalattributes. Each internal node n of a decision tree is associatedwith an attribute $A_{jn}$and a split point z_n,giving a binarytest $x \leq z_n$. An internal node has exactly 2 children. Eachleaf node m in the decision tree is associated with a discreteprobability distribution P_m over C.To determine the class label of a given test tuple $t_0=(f_{0,1}, \ldots, f_{0,k}, ?)$, we traverse the tree top down, starting fromthe root node. When we visit an internal node n, we split thetuple into two parts at $z_n$and distribute each part recursively down the child nodes accordingly. Eventually, we reach leafnodes. Theprobability distribution P_m at each leaf node m

Contributes1to the final distribution P_0 for predicting the classlabel of t_0. A pdf $f_{i,j}$could be programmed analytically if it can

be specified in closed forms. More typically, it would beimplemented numerically by storing a set of s sample points $x \in [a_{i,j}, b_{i,j}]$ with the associated value $f_{i,j}(x)$, effectivelyapproximating $f_{i,j}$by a discrete distribution. We adopt this numerical approach for the rest this paper. With this representation, the amount

mean value from the tuples and generate the tree that is result of averaging algorithm. it is

easy to understand but not sufficient to generate decision tree for uncertain data like climate,banking dataset.Averaging take the result of pdf as it is and classifythe data in tree form.

2)Distribution based

The another approach is distribution based algorithm. Here after calculating mean,variens of the tuples,it will check duplication and missing value in our dataset,if it is present then we can easily remove this unwanted data through pruning tech. from dataset.

For uncertain data,we use the same decision tree building framework,including the tech. of preprunning and postprunning.the key to building a good decision tree is good choice of an attribute and split points for each node n,however, the no. of choices of a split point given an attribute is not limited to m-1 point value,but the union of domains of all pdfs equal to 1…..m.Representing each with s sample points,there are in total ms sample points.So,there are ms-1 possible split points.UDT is s time expensive than AVG.

It reduce the complexity of decision tree.Traditional decision tree algorithm such as ID3 and C4.5 are also used for averaging.

When processing a node,we examine a set of tuples S. The algorithm start with node and with S being the set of all training tuples.At each node n, we first check if all the tuples in S have the same class label c.If so, make n leaf node and set $p_n(c)=1$ otherwise we select an attribute and split points and divide the tuples into two subset that is left and right.

To build a good decision tree,we also use the blackbox algorithm Bestsplit,which take a set of tuples as parameter and return the best choice of attribute and split point for those tuples.Bestsplit is designed to select theattribute and split point that minimises the degree of dispersion.The degree of dispersion can be measured in many ways such as entropy or Gini index.The choiceof dispersion function

Definition 6 (Tuple Count): For an attribute A_i , the tuple count for class c 2 C in an interval (a; b] is:

$$\gamma_{c,j}(a,b) = \int_a^b g_{c,j}(x)\,dx$$

Theorem 1: If an optimal split point falls in an empty interval, then an end-point of the interval is also an optimal split point.

Proof: By the definition of information gain, if the optimal split point can be found in the interior of an empty interval (a; b], then that split point can be replaced by the endpoint a without changing the resulting entropy.

As a result of this theorem, if (a; b] is empty, we only need to examine the end-point a when looking for an optimal split point. There is a well-known analogue for the point-data case, which states that if an optimal split point is to be placed between two consecutive attribute values, it can be placed anywhere in the interior of the interval Therefore, when searching for the optimal split point, there is no need to examine the interior of empty intervals.

Theorem 2: If an optimal split point falls in a homogeneous interval, then an end-point of the interval is also an optimal split point P Proof Sketch: Using the substitution x is equal to,

$$\sum_{c\in C} \gamma_{c,j}(a,z)$$

and

$$y = \sum_{c\in C} \gamma_{c,j}(z,b)$$

The implication of this theorem is that interior points in homogeneous intervals need not be considered when we are looking for an optimal split point.

Theorem 3: Suppose the tuple count for each class increases linearly in a heterogeneous interval

$$(a,b] \text{ (i.e., } \forall c \in C, \forall t \in [0,1],$$

$$\gamma_{c,j}(a,(1-t)a+tb) = \beta_c t$$

If an optimal split point falls in (a; b], then an end-point of the interval is also an optimal split point.shown that d2H dt2 _ 0 and hence H(t) is a concave function. Consequently, H attains its minimum value at one of the extreme points t = 0 and t = 1, which correspond to z = a and z = b, respectively.

strategies into UDT-GP. The resulting algorithm is called UDT-ES.

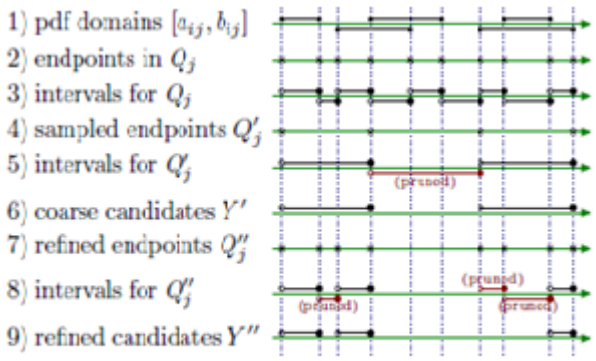


Fig. 5. Illustration of end-point sampling

The figure shows 9 rows, illustrating 9 steps of the pruning process. Each row shows an
arrowed line representing the real number line. On this line are end points (represented by crosses) or intervals (represented by line segments) drawn. Row 1 shows the intervals obtained
from the domains of the pdf's. The collection of end-points of these intervals constitute the set Q_j (row 2). From

these end-points, disjoint intervals are derived (row 3). So far, the process is the same as global-pruning. The next step differs from the
global-pruning algorithm: Instead of using the set of all end-points Q_j (row 2), we take a sample Q_{0j}
(row 4) of these points. The choice of the sample size is a trade-off between fewer entropy continue with the global pruning algorithm as before, using the sampled end-points Q_{0j} instead of Q_j . The algorithm thus operates on the intervals derived from Q_{0j} (row 5) instead ofthose derived from Q_j(row 3). Note that intervals in row 3 are concatenated to form intervals in row 5 and hence fewer intervals and end-points need to beprocessed. After all the prunings on the coarser intervals are done, we are left with a set Y_0 of candidate intervals (row 6). (Note that a couple of end-points are pruned in the second interval of row 5.)For each unpruned candidate interval $(q_y; q_{y+1}]$ in row 6, we bring back the original set of end-points inside the interval(row 7) and their original finer intervals (row 8). We re-invoke global-pruning again using the end-points in Q_{00}j (carefully caching the already calculated values of $H(q;A_j)$ for q_2 Q_{0j}). The candidate set of
reduces the number of entropy computations to 0.56%-28% when compared with UDT. Hence, it achieves a pruning effectiveness in the range of
72% to 99.44%.

VII. CONCLUDING REMARKS

We have done extension to the model of decision-tree classification and tree-construction algorithms to accommodate data tuples having numerical
attributes with uncertainty explained by arbitrary pdf's. Experiments conclude that exploiting data uncertainty leads to decision trees with very high accuracies. Even though our novel techniques are basically designed to handle uncertain data, they are very useful for building decision trees using classical algorithms when there are very high amounts of data tuples.

REFERENCES

[1] R. Agrawal, T. Imielinski, and A. N. Swami, “Database mining: A performance perspective,” IEEE Trans. Knowl. Data Eng., 1993.
[2] J. R. Quinlan, “Induction of decision trees,” Machine Learning, 1986.
[3] ——, C4.5: Programs for Machine Learning. Morgan Kaufmann, 1993.
[4] C. L. Tsien, I. S. Kohane, and N. McIntosh, “Multiple signal integration
by decision tree induction to detect artifacts in the neonatal intensive care
unit,” Artificial Intelligence in Medicine, vol. 19, no. 3, 2000
[5] M. Chau, R. Cheng, B. Kao, and J. Ng, “Uncertain data mining: An
example in clustering location data,” in PAKDD, 2006, pp. 199–204.
[6] W. K. Ngai, B. Kao, C. K. Chui, R. Cheng, M. Chau, and K. Y. Yip,
“Efficient clustering of uncertain data,” in ICDM, 2006, pp. 436–445.
[7] S. D. Lee, B. Kao, and R. Cheng, “Reducing UK-means to K-means,”
in 1st Workshop on Data Mining of Uncertain Data, in ICDM, 2007.

[11] T. Elomaa and J. Rousu, “Efficient multisplitting revisited: Optimapreserving elimination of partition candidates,” Data Mining and Knowledge Discovery, vol. 8, no. 2, pp. 97–126, 2004.
[12] T. M. Mitchell, Machine Learning. McGraw-Hill, 1997.
[13] A. Asuncion and D. Newman, “UCI machine learning repository,” 2007.
[Online]. Available: