Volume 3, Issue 10, October 2013 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Classification of Uncertain Data using Decision Trees

Kiran Siripuri,
M. Tech, Department of Computer Science & Engineering
Kakatiya Institute of Technology and Science,
Warangal, Andhra Pradesh, India

M. Venugopal Reddy
Assistant Professor, Department of Computer Science,
Kakatiya University, Warangal,
Warangal, Andhra Pradesh, India

P. Niranjan Reddy
Professor, Department of Computer Science &Engineering
Kakatiya Institute of Technology and Science,
Warangal, Andhra Pradesh, India

Abstract: Classic decision sapling classifiers assist data whose values are usually known along with precise. We extend such classifiers to manage data using uncertain information .Value scepticism arises in numerous applications throughout the data collection process. Example reasons for uncertainty consist of measurement/quantization problems, data staleness, along with multiple repeated measurements. Along with uncertainty, the worth of any data item is normally represented certainly not by a unitary value, yet by numerous values being created a likelihood distribution. As opposed to abstracting unsure data by means of statistical derivatives (such while mean along with median), we see that the accuracy of a decision sapling classifier is usually much improved if the "complete information" of a data item (taking into account the likelihood density operate (pdf)) is utilized. We extend classical conclusion tree constructing algorithms to manage data tuples using uncertain prices. Extensive experiments have been conducted which usually show that this resulting classifiers will be more accurate than those utilizing value averages.

Keywords— Uncertain Data, Classification, Decision Tree

I. INTRODUCTION

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. Many algorithms, such as ID3 and C4.5, have been devised for decision tree construction. These algorithms are widely adopted and used in a wide range of applications such as image recognition, medical diagnosis, credit rating of loan applicants, scientific tests, fraud detection, and target marketing.

In traditional decision tree classification, a feature (an attribute) of a tuple is either categorical or numerical. For the latter, a precise and definite point value is usually assumed. In many applications, however, data uncertainty is common. The value of a feature/attribute is thus best captured not by a single point value, but by a range of values giving rise to a probability distribution. A simple way to handle data uncertainty is to abstract probability distributions by summary statistics such as means and variances. We call this approach averaging. Another approach is to consider the complete information carried by the probability distributions to build a decision tree. We call this approach Distribution-based. In this paper, we study the problem of constructing decision tree classifiers on data with uncertain numerical attributes. Our goals are 1) to devise an algorithm for building decision trees from uncertain data using the Distribution-based approach, 2) to investigate whether the Distribution-based approach could lead to a higher classification accuracy compared with the Averaging approach, and 3) to establish a theoretical foundation on which pruning techniques are derived that can significantly improve the computational efficiency of the Distribution-based algorithms.

II. RELATED WORK

Direct marketing refers to a process of identifying and mailing to potential customers: retail industries need to identify buyers of certain products, banks and insurance companies need to promote loan insurance products to customers, fundraising organizations need to identify potential donors, etc. Available is a historical database about the previous mailing campaign, including whether a customer responded and the dollar amount collected. The task is to build a model to predict the current customers who are likely to respond. Typically, most records, say 95%, are "not respond" records. Thus, maximizing the accuracy of prediction does not work because simply predicting all customers as "not respond" will give 95% accuracy. In recent years, it is realized that cost-sensitive treatment is required in applications like direct marketing. proposed the MetaCost framework for adopting accuracy based classification to cost-sensitive learning by incorporating a cost matrix C for misclassifying class j into class I, examined the more general case where the benefit B(i; j; x) depends not only on the classes involved but also on the individual customers x. A drawback of these approaches is that they need to estimate the conditional class probability P(jjx), which ignores the customer value of x such as the donation amount. The customer value is only considered “after the fact” via the factor B(i; j; x). In this project, we estimate the profit on a customer directly. This has two advantages. First, it takes into account the customer value from the very beginning. Second, it opens up new avenues for profit estimation. In particular, we propose a profit estimation method by combining association rules and pessimistic estimation of errors. Association rule approach offers an edge of finding correlated features that may never be found in a local search such as in decision tree induction and naive Bayes classifiers.

Each record in the KDD-CUP-98 dataset is described by 479 non-target variables and two target variables indicating the “respond”/“not respond” classes and the actual donation in dollars. About 5% of records are “respond” records and the rest are “not respond” records. The dataset has been pre-split into 50% for learning and 50% for validation. The competition task is to build a prediction model of the donation amount using the learning set. The participants are contested on over all the validation records with predicted donation greater than the mailing cost $0.68. This real life dataset presents two challenges. First, “there is often an inverse correlation between the likelihood to respond and the dollar amount of the gift”. This inverse correlation invalids any probability based ranking because a valuable customer will be ranked low. Second, the high dimensionality of the dataset presents a big challenge for extracting correlated features. Among the 481 variables in the dataset, 208 variables have 10 or more distinct values after discretising continuous variables, making a potential search space of size 10208. We address these issues in three steps. In rule generating, we extract characteristics typical of “respond” records. In model building, we construct a prediction model using the found characteristics to maximize generated profit on the learning set. In model pruning, we prune over fitting characteristics to generalize the model to the whole population. [1]

Given a specification of costs for correct and incorrect predictions, an example should be predicted to have the class that leads to the lowest expected cost, where the expectation is computed using the conditional probability of each class given the example. We leave the case where both reasonableness conditions are violated for the reader to analyse. Margineantu has pointed out that for some cost matrices, some class labels are never predicted by the optimal policy as given by Equation. We can state a simple, intuitive criterion for when this happens. Given a cost matrix, the decisions that are optimal are unchanged if each entry in the matrix is multiplied by a positive constant. This scaling corresponds to changing the unit of account for costs. Similarly, the decisions that are optimal are unchanged B if a constant is added to each entry in the matrix. This shifting corresponds to changing the baseline away from which costs are measured. [2]

In this project, we propose a relevance-based boosting style algorithm1 to build a customized LazyDT ensemble for each test instance. We show that our method significantly improves the performance over the base learner LazyDT and produces significantly (on average 50%) less complex ensembles than AdaBoost while maintaining comparable accuracy. To ameliorate the over-fitting problem for LazyDT, we also propose a new distance-based pruning technique to generate simpler and more accurate lazy decision trees. Currently this technique is implemented and tested only on data sets with numerical features. In our experiments, the proposed pruning method improves the accuracy and comprehensibility for both single lazy decision trees and LazyDT ensembles. The core part of the algorithm is how to select a test { LazyDT chooses a test that \optimizes" the resulting branch taken by the given test instance. Once a test is selected, only instances that take the same branch as the test instance are kept to build the remaining part of the lazy decision tree. We omit the details of the algorithm and refer readers to the original project (Friedman et al., 1996) for an exact description of the LazyDT algorithm. Base learner to different distributions of the training data. Typically it is assumed that the produced base classifiers can be applied to classify the entire instance space. In each iteration, AdaBoost adjusts the weights of training instances according to the classification decisions made by the previously learned classifiers. Misclassified instances will be assigned larger weights in the successive iteration to make the base learner focus on the \hard" instances.

When LazyDT is used as the base learner, it differs from regular decision algorithms in two ways. First, LazyDT generates a single decision path for a given test instance. This path can only be used to give predictions to instances that satisfy all of its tests. For those instances that fail to satisfy the tests, no classification information is available. Without complete classification information, the weight changing process in the above framework cannot be directly applied. Second, the LazyDT algorithm has a rather special goal {building a decision path that correctly classifies a given test instance. If a training instance contributes little information to correctly classifying the given test instance, even it is a \hard" instance for the current classifier, not much leverage can be gained by increasing its weight in subsequent iterations. A tentative solution to the first of these two problems is to use the confidence-rated boosting algorithm (Schapire & Singer, 1999), which is capable of handling incomplete classification information. However, it does not address the second problem. Moreover, despite the fact that it has been used in SLIPPER (Cohen & Singer, 1999) to boost decision rules, the confidence-rated boosting algorithm is not an ideal choice for our task. In SLIPPER, a rule makes predictions for the training instances that satisfy all of its tests with a positive confidence and abstains on the others by making a prediction with a zero confidence. The boosting process changes the weight of a training instance only if it is given a classification with nonzero confidence. Our investigation of this method did not give promising results, which we conjecture was due to the lack of diversity in the resulting ensembles. LazyDT grows a decision rule (path) until all the instances that are covered by the rule are from the same class. Typically, only a small part of the training set will be predicted with non-zero confidence and be assigned different weights. Given only small changes of the distribution, LazyDT tends to produce very similar if not identical decision paths for each iteration, resulting in an ensemble that lacks diversity[3].

Ensemble methods such as constructing committees of decision trees as described in this project have been shown to improve the accuracy of single base classifiers with significant effectiveness. The instability of base learning algorithms provides room for ensemble methods to improve the performance of the base classifiers. For example, Bagging and Boosting two of the most widely used ensemble approaches that improve performance through manipulating original training data explore the instability that small changes to the training data can lead to large changes in the learned classifier. They construct multiple base trees, each time using a bootstrapped replicate of the original training data. Our new ideas to construct committees of decision trees were inspired by two interesting observations from our previous studies on gene expression and proteomic profiling data: We found that many ensembles constructed by the Boosting method were singletons. This is due to stopping criteria that is used to terminate the sequential construction of base classifiers in Boosting. If the training error of a current classifier is zero, then Boosting will not proceed to generate any more new classifiers. We also found that many top-ranked features possess similar discriminating merits with little difference for classification.

The first observation indicates that it is necessary to break the constraint of the stopping criteria of Boosting. Otherwise, the ensemble method will have no difference from a base classifier in some cases. The second observation suggests that it is worthwhile to employ different top ranked features as the root nodes for building multiple decision trees. These motivations have led to our cascading approach described below. In our method, we first rank all features of a training set according to their gain ratios (or by other measures such as entropy). Denote as top ranked features. Then in a cascading manner, we force to be the root node of the tree. All the other nodes are then constructed as usual. We call such decision trees cascading trees, emphasizing that the selection of the root nodes is from the highest position to a lower position feature. In this way, we distinguish our cascading trees from those generated by the Bagging, Boosting, or Randomization method.

Note that our method makes use of the instability of base classifiers unlike Bagging and Boosting: Instead of manipulating the training data, we keep the original training data unchanged but change the learning phase of base classifiers. So, our cascading method solves another problem that often occurs with bio-data analysis using Bagging or Boosting. The problem is that: Rules induced from bootstrapped training data may not be correct when applied to the original data as some samples are duplicated and some are removed during training. This is of a critical concern in bio-medical applications such as the understanding and diagnosis of a disease. Although the approach may help to improve the resulting classifiers' accuracy, the Bagging or Boosting rules have to be understood with caution. However, rules from cascading trees are required to be all true with the training data, leading to a more exact understanding of the data.

Cascading trees usually have different structures and contain different features, yet these diversified trees can perform almost equally well on both training and test data. So, the cascading idea can be used to build tree committees such that they contain many "experts". Thus, the committees would bring large potential to improve the performance of the base classifiers. To combine these cascading trees for classification, we share the rules in the trees of a committee in a weighted manner. Basically, we compare important global rules with those rules that are locally satisfied by specific test samples, then we integrate these comparisons into a classification score. Together, the cascading idea guides the construction of tree committees while the sharing idea aggregates the decisions made by those individual trees. For convenience, we therefore name our classifier CS4: cascading-and-sharing for constructing decision tree ensembles.

Our cascading idea is related to but different from the randomization idea used. Like cascading trees, randomized trees are also derived from the same original training data rather than from bootstrapped training data. Unlike cascading trees, all randomized trees in a committee may share one and only one root node though their internal nodes could be different in a manner of random choice from a set of candidate features. Therefore the diversity of cascading trees has larger potential than that of randomized trees.[4]

The goal of classification learning algorithms is to build a classifier from a set of training examples with class labels such that the classifier can well predict the unseen testing examples. The predictive ability of the classification algorithm is typically measured by its predictive accuracy (or error rate, which is 1 minus the accuracy) on the testing examples. However, most classifiers (including decision trees and Naive Bayes) can also produce probability estimations or "confidence" of the class prediction. Unfortunately, this information is completely ignored in accuracy. That is, the accuracy measure does not consider the probability (be it 0.51 or 0.99) of the prediction; as long as the class with the largest probability estimation is the same as the target, it is regarded as correct. This is often taken for granted since the true probability is unknown for the testing examples anyway. In many data mining applications, however, accuracy is not enough. For example, in direct marketing, we often need to promote the top X percent (X can be 5 or 10) customers during gradual roll-out, or we often deploy different promotion strategies to customers with different likelihood of purchasing. To accomplish these tasks, we need more than a mere classification of buyers and no buyers. We need (at least) a ranking of customers in terms of their likelihoods of buying. Thus, a ranking is much more desirable than just a classification and it can be easily obtained since most classifiers do produce probability estimations that can be used for ranking (testing) examples. If we want to achieve a more accurate ranking from a classifier, one might naturally expect that we must need the true ranking in the training examples. In most scenarios, however, that is not possible. Instead, what we are given is a data set of examples with class labels only. Thus, given only classification labels in training and testing sets, are there better methods than accuracy to evaluate classifiers that also produce rankings? The answer lies in the ROC curve. The ROC (Receiver Operating Characteristics) curve was first used in signal detection theory to represent the trade-off between the hit rates and false alarm rates. It has been extensively studied and applied in medical diagnosis since the 1970s. Spackman was one of the first researchers who used the ROC graph to compare and evaluate machine learning algorithms. In recent years, extensive research on ROC has been done in machine learning. The area under the ROC curve, or simply AUC, provides a good "summary" for the performance of the ROC curves. Below we will provide a brief overview of ROC and its AUC. [5]

Knowledge discovery in databases is the non-trivial process of identifying valid, novel, potentially useful, and ultimately understandable patterns in data. The literature has mostly focused on the "valid" and "understandable" measures that can be defined in objective terms, such as the confidence/support measure and the size/number of patterns. Due to the lack of modelling user knowledge, discovered patterns often are known or not useful to the user. To model the "novel" and "useful" aspects, subjective measures are needed. Classified subjective measures into unexpectedness and action ability. A pattern is unexpected if it is surprising to the user. A pattern is actionable if the user can act upon it to her advantage.

Despite the efforts of initial works on action ability mining (more details in Related Work), it turned out to be very hard to capture formally the "elusive" nature of action ability in all of its manifestations and across different types of applications and settings. The following factors contribute to the difficulties of this problem. First, there is the dilemma that user knowledge should be acquired to define action ability, but the ability to provide such knowledge tends to invalidate the need for mining actionable patterns. Second, it is very hard to model the "usefulness" of a pattern. Strictly speaking, this information is not available until after patterns are deployed in the real world; on the other hand, the data mining algorithm needs to tell if a pattern is actionable before it is deployed. Finally, the search space is likely to be large and a scalable method must exploit the notion of action ability to prune the search as early as possible.

We propose an action ability mining approach that addresses three goals. Find the right balance between providing enough domain knowledge and making the problem scalable, tractable and non-trivial. Model action ability in a domain-independent manner and provide a way to estimate the success of actionable patterns. Utilize user knowledge early in the search of actionable patterns. The main idea is as follows. Suppose that a database contains historical observations about several features and a success measure called the utility. Also, suppose that we know some actions can influence features in certain (simple) ways. If some features are correlated with the utility in some population of the data, a change in those features would imply a change in the utility. Now if some actions can influence those features, this influence will cascade to the utility through the correlation. We are interested in the patterns that summarize those actions and populations where such cascaded influences increase the utility. An example explains these points. [6]

III. PROPOSED WORK

i. *Unlimited Resource Case*

Our first step is to consider how to extract actions when there is no restriction on the number of actions to produce. Our first industrial case study of an application corresponds to this case. We call this the unlimited-resource case. Our data set consists of descriptive attributes and a class attribute. For simplicity, we consider a discrete-value problem, in which the class values are discrete values. Some of the values under the class attribute are more desirable than others. For example, in the banking application, the loyal status of a customer "stay" is more desirable than "not stay." The overall process of the algorithm can be briefly described in the following four steps: Import customer data with data collection, data cleaning, data pre-processing, and so on. Build customer profiles using an improved decision tree learning algorithm from the training data. In this case, a decision tree is built from the training data to predict if a customer is in the desired status or not. This is a key component of the data mining system Proactive Solution. Produce reports for domain experts to review the actions and selectively deploy the actions.

ii. *Leaf-Node Search*

In first consider the simpler case of unlimited resources where the case serves to introduce our computational problem in an intuitive manner. The leaf-node search algorithm searches for optimal actions to transfer each leaf node to another leaf node with a higher probability of being in a more desirable class. After a customer profile is built, the resulting decision tree can be used to classify, and more importantly, provide the probability of customers in the desired status such as being loyal or high-spending. When a customer, who can be either a training example used to build the decision tree or an unseen testing example, falls into a particular leaf node with a certain probability of being in the desired status, the algorithm tries to "move" the customer into other leaves with higher probabilities of being in the desired status. The probability gain can then be converted into an expected gross profit. However, moving a customer from one leaf to another means some attribute values of the customer must be changed. This change, in which an attribute A's value is transformed from v1 to v2, corresponds to an action. These actions incur costs. The cost of all changeable attributes are defined in a cost matrix by a domain expert. The leaf-node search algorithm searches all leaves in the tree so that for every leaf node, a best destination leaf node is found to move the customer to. The collection of moves is required to maximize the net profit, which equals the gross profit minus the cost of the corresponding actions.

iii. *Limited Resource Case*

Our previous case considered each leaf node of the decision tree to be a separate customer group. For each such customer group, we were free to design actions to act on it in order to increase the net profit. However, in practice, a company may be limited in its resources. For example, a mutual fund company may have a limited number k (say three) of account managers, each manager can take care of only one customer group. Thus, when such limitations exist, it is a difficult problem to optimally merge all leave nodes into k segments, such that each segment can be assigned to an account manager. To each segment, the responsible manager can several apply actions to increase the overall profit. This limited-resource problem can be formulated as a precise computational problem. Consider a decision tree DT with a number of source leaf nodes that correspond to customer segments to be converted and a number of candidate destination leaf nodes, which correspond to the segments we wish customers to fall in. Formally, the bounded segmentation problem (BSP) is defined as follows: Given: a decision tree DT built from the training examples, with a collection S of m source leaf nodes and a collection D of n destination leaf nodes. The meaning of a goal is to transform customers that belong to the source nodes S to the destination node D via a number of attribute-value changing actions. Our aim is to find a solution with the maximal net profit. Our first solution is an exhaustive search algorithm for finding the k optimal action sets with maximal net profit.

iv. *Greedy BSP*

We choose any combination of k action sets, Group the leaf nodes into k groups and evaluate the net benefit of the action sets on the groups. The k action sets with associated leaf node groups that have the maximal net benefit. Since the optimal-BSP needs to examine every combination of k action sets, which is exponential in the value of k. To avoid the exponential worst-case complexity, we have also developed a greedy algorithm which can reduce the computational cost and guarantee the quality of the solution at the same time. The goal is to choose k action sets so as to maximize the net profit of covered leaf nodes. We can solve this problem using a greedy algorithm below, where C is the resulting k action sets. We consider the intuition of the Greedy-BSP algorithm using an example profit matrix M. Each number is a profit Pij value computed from the input parameters. The greedy algorithm processes this matrix in a sequential manner for k iterations. In each iteration, it considers adding one additional column of the M matrix, until it has considered all k columns. Initially, Greedy-BSP starts with an empty result set C. The algorithm then compares all the column sums that correspond to converting all leaf nodes S1 to S4 to each destination leaf node Di in turn. Thus, the resultant action set C is assigned. Next, Greedy-BSP considers how to expand the customer groups by one. To do this, it considers which additional column will increase the total net profit to a highest value, if we can include one more column. If we in addition consider the first column, then the node S1 can choose to be converted to D1, instead of D2. In that case, the profit of C can be increased by two units, which is the maximum increase among all other columns. Thus, we choose this subset to be the current action set C.

v. *Algorithm:*

```
1.  Assign initial values for cluster means c1 to ck
2.  repeat
3.    for  i=1 to n do
4.        Assign each data xi to cj where E(||cj-xi||)
          is the minimum
5.  End for
6.  For j=1 to k do
7.  Recalculate cluster mean cj of Cj
8   End For
9.  until Convergence
10.   Return C
```

IV. EXPERIMENTATION

We use an IRIS dataset as input to find the optimised decision tree in it. The dataset consists of information regarding sepals and petals of eye. As it is a huge dataset there may be a chance of uncertain data present in it. We perform various pruning techniques such as LP, GP, and Averaging and get the effective data by combining the three techniques together. We calculate the effectiveness of bounding values or intervals such that multiple bounding ranges are taken and the optimal range is calculated as end point for pruning purpose. This results in effective decision tree.

V. RESULTS

The concept of this paper is implemented and different results are shown below, The proposed paper is implemented in Java technology on a Pentium-IV PC with minimum 20 GB hard-disk and 1GB RAM. The propose paper's concepts shows efficient results and has been efficiently tested on different Datasets.

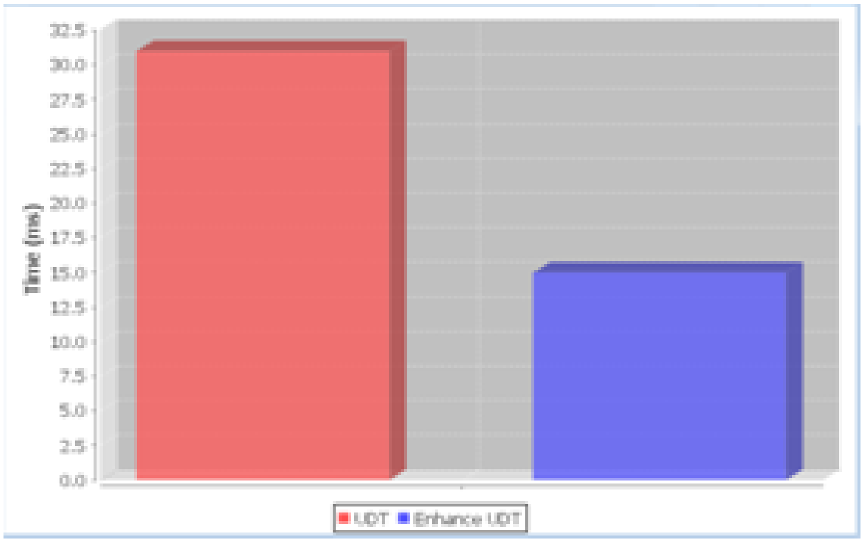

Fig. 1 Comparative execution time for Dataset IRIS-1.

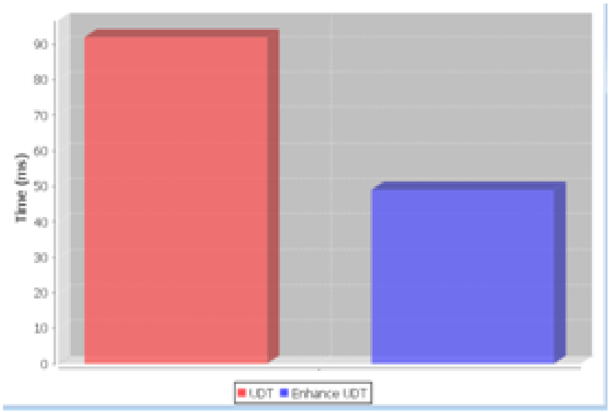

Fig. 1 Comparative execution time for Dataset IRIS-2

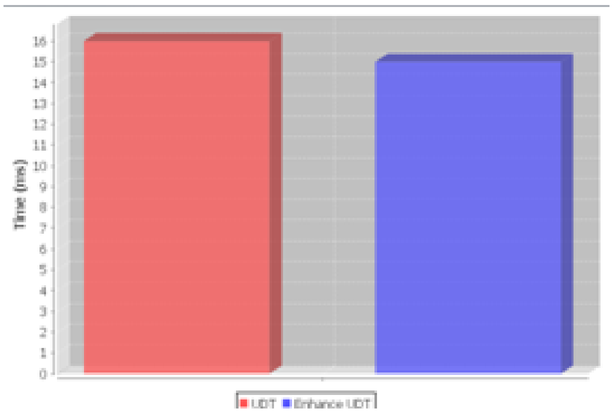

Fig. 3 Comparative execution time for Dataset Page-1

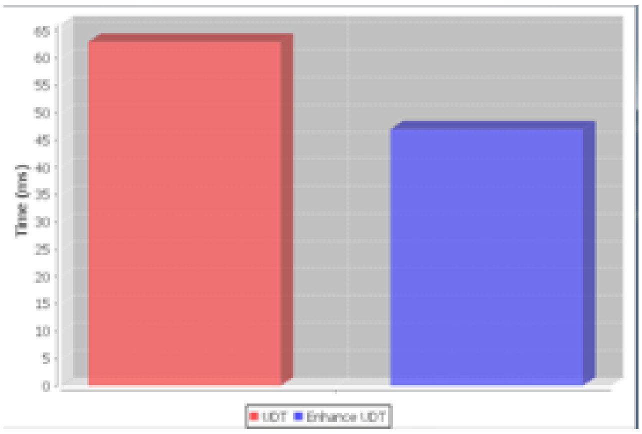

Fig. 4 Comparative execution time for Dataset Page-2

VI. Conclusions

Through this paper we propose a series of pruning techniques to improve tree construction efficiency. Our algorithms have been experimentally verified to be highly effective. Their execution times are of an order of magnitude comparable to classical algorithms. Some of these pruning techniques are generalizations of analogous techniques for handling point-valued data. Other techniques, namely pruning by bounding and end point sampling, are novel. Although our novel techniques are primarily designed to handle uncertain data, they are also useful for building decision trees using classical algorithms when there are tremendous amounts of data tuples.

Future Directions

The paper extended the model of decision tree classification to accommodate data tuples having numerical attributes with uncertainty described by arbitrary pdfs. We have modified classical decision tree building algorithms based on the framework of C4.5 to build decision trees for classifying such data. Future plans will find empirically that when suitable pdfs are used, exploiting data uncertainty leads to decision trees with remarkably higher accuracies. We therefore advocate that data be collected and stored with the pdf information intact. Finally, performance is an issue, though, because of the increased amount of information to be processed, as well as the more complicated entropy computations involved.

REFERENCES

[1] The kdd-cup-98 result: http://www.kdnuggets.com/meetings/kdd98/kdd-cup-98-results.html, 2005.

[2] N. Abe, E. Pednault, H. Wang, B. Zadrozny, W. Fan, and C. Apte, "Empirical Comparison of Various Reinforcement Learning Strategies for Sequential Targeted Marketing," Proc. Second IEEE Int'l Conf. Data Mining (ICDM '02), 2002.

[3] M. Belkin, P. Niyogi, and V. Sindhwani, "On Manifold Regularization," Proc. 10th Int'l Workshop Artificial Intelligence and Statistics, pp. 17-24, Jan. 2005.

[4] C. Elkan, "The Foundations of Cost-Sensitive Learning," Proc. 17th Int'l Joint Conf. Artificial Intelligence (IJCAI '01), 2001.

[5] J. Huang and C.X. Ling, "Using Auc and Accuracy in Evaluating Learning Algorithms," IEEE Trans. Knowledge and Data Eng., vol. 17, no. 3, pp. 299-310, 2005.

[6] J. Li and H. Liu, "Ensembles of Cascading Trees," Proc. IEEE Int'l Conf. Data Mining (ICDM '03), pp. 585-588, 2003.

[7] C.X. Ling, T. Chen, Q. Yang, and J. Cheng, "Mining Optimal Actions for Intelligent CRM," Proc. IEEE Int'l Conf. Data Mining (ICDM), 2002.

[8] E. Pednault, N. Abe, and B. Zadrozny, "Sequential Cost-Sensitive Decision Making with Reinforcement Learning," KDD '02: Proc. Eighth ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining, pp. 259-268, 2002.

[9] K. Wang, Y. Jiang, and A. Tuzhilin, "Mining Actionable Patterns by Role Models," Proc. IEEE Int'l Conf. Data Eng., 2006.

[10] K. Wang, S. Zhou, Q. Yang, and J.M.S. Yeung, "Mining Customer Value: From Association Rules to Direct Marketing," Data Mining and Knowledge Discovery, vol. 11, no. 1, pp. 57-79, 2005.

[11] A. Tuzhilin, Y. Jiang, K. Wang, and A. Fu, "Mining Patterns that Respond to Actions," Proc. IEEE Int'l Conf. Data Mining, pp. 669- 672, 2005.

[12] Q. Yang, J. Yin, C.X. Ling, and T. Chen, "Postprocessing Decision Trees to Extract Actionable Knowledge," Proc. IEEE Conf. Data Mining (ICDM '03), pp. 685-688, 2003.

AUTHORS BIOGRAPHY

Kiran siripuri: is pursuing M.Tech in Software Eng. from Kakatiya Institute of Technology. Completed B.Tech from SR Engg.college Warangal in 2011.His interested areas are data mining, database administration**.**

Venugopal Reddy M. received B.E. (Mechanical Engineering) from Osmania University in 1990, M.B.A. from Kakatiya University and M.Tech. (Computer Science and Engineering) from NIT, Warangal in the year 2002. He is doing a part-time research in IIIT, Hyderabad since 2006. He published one paper in International Journal, one paper in International Conference and two papers in National Conferences. Presently he is Associate Professor in the Department of CSE in KITS, Warangal. He is the member of the ISTE

Niranjan Polala received the B.E Computer Science from Nagpur University in 1992 and M.Tech (Computer Science and Engineering) from NIT, Warangal in the year 2001.He worked as a Lecturer and Assistant Professor in the department of CSE of KITS, Warangal, Since 1996. He is doing a part-time research in Kakatiya University, Warangal since 2007. He authored two Text books, Theory of computation and Computer Graphics in the field of Computer Science. He published 10 papers in International Journals and 6 papers in International Conferences. Presently he is Professor and HOD of CSE in KITS, Warangal. He is the member of the ISTE and CSI