Naive Bayes Classification Given Probability Estimation Trees

Zengchang Qin
Berkeley Initiative in Soft Computing (BISC)
EECS Department, University of California
Berkeley, CA 94720, USA
zqin@cs.berkeley.edu

Abstract

Tree induction is one of the most effective and widely used models in classification. Unfortunately, decision trees such as C4.5 [9] have been found to provide poor probability estimates. By the empirical studies, Provost and Domingos [6] found that Probability Estimation Trees (PETs) give a fairly good probability estimation. However, different from normal decision trees, pruning reduces the performances of PETs. In order to get a good probability estimation, we usually need large trees which are not good in terms of the model transparency. In this paper, two hybrid models by combining the Naive Bayes classifier and PETs are proposed in order to build a model with good performance without losing too much transparency. The first model use Naive Bayes estimation given a PET and the second model use a group of small-sized PETs as Naive Bayes estimators. Empirical studies show that the first model outperforms the PET model at shallow depth and the second model is equivalent to Naive Bayes and PET.

Keywords:
Probability Estimation Tree, Decision Tree, Naive Bayes, Classification, Hybrid Classification Model.

1 Introduction

Tree induction algorithms were received a great deal of attention because of their simplicity and effectiveness. There are many decision tree models or tree based models have been proposed. From early discrete decision trees such as ID3 [8] and C4.5 [9] to to a variety types of regression trees. Trees that estimate the probability of class membership are also referred to as Probability Estimation Trees (PETs) [6], where probability of a particular class given a branch is calculated by the proportion of data belonging to this class to all the data covered by the branch.

Like a decision tree, a PET can be represented as a set of rules and hence provides a high level transparency. However, for some complex problems, good probability estimations can only be obtained by large PETs, which are not good in terms of transparency. In such cases, how can we build a model which has a good probability estimation with compact PETs (i.e., PETs with shallow depths and less number of branches)? This question motivates the research presented in this paper.

Naive Bayes is a well known and much studied algorithm in machine learning. It is a simple, effective and efficient learning method. Although Naive Bayes classification makes an unrealistic assumption that the values of the attributes of an instance are independent given the class of the instance, this model is remarkably successful in practice. In this paper, the Naive Bayes classifer is used to build new hybrid models by combining it with PETs.

This paper is organized as follows. Section 2 and 3 introduce the probability estmation tree model and Naive Bayes, respectively. In section 4, we described the first new hybrid by combining the Naive Bayes model and probability estimation trees. In section 5, another new hybrid model of using a group PETs as Naive Bayes estimators is proposed. In section 6, we test the new models by using a series of UCI datasets and the conclusions are given in section 7.

2 Probability Estimation Trees

Decision Tree, more properly a classification tree[1], is used to learn a classification function which predicts the value of a target attribute (class attribute) given the values of the independent (input) attributes by a tree-structured model. A node with no split is called a leaf, which is associated with a particular class label. A new unlabeled data is classified by determining which leaf it leads to. Decision tree induction attracts a great attention for its simplicity and

[1] A decision tree with a range of discrete (symbolic) class labels is called a classification tree, whereas a decision tree's output with a range of continuous (numeric) values is called a regression tree or a prediction tree.

Proceedings of the 5th International Conference on Machine Learning and Applications (ICMLA'06)
0-7695-2735-3/06 $20.00 © 2006 IEEE

effectiveness. Algorithms such as ID3 [8], C4.5 [9] have been well-known not only in machine learning and but also other scientific communities as well. Traditionally, this setting was sufficient for most of the classification problems and applications. However, more and more applications require some kind of reliability, liklihood or numeric assessment of the quality of each classfication. In other words, we do not only want that the model predicts a class value for each example but also that it can be given an estimate of the reliability of each prediction. Such classifiers are usually called soft classifiers [4]. The most general presentation of a soft classifier is a probability estimator, i.e. a model that estimates the probability of a particular class given an unlabeled example. A decision tree classifier is defined as a decision tree with an associated labelling of the leaves with classes. A probability estimation tree (PET) is a decision tree where each leaf is assigned a probability distribution over classes. These probability estimates can for instance be relative frequencies [4].

Given a data set contains N instances or examples: $\mathcal{D} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$, where each instance is determined by n attributes, i.e., $\mathbf{x} = \langle x_1, \ldots, x_n \rangle$. Given a set of classes $\mathcal{C} = \{C_1, \cdots, C_m\}$, each instance is labelled with a particular class C_k, formally: $\mathbf{x} \rightarrow C_k$ where $C_k \in \mathcal{C}$. A branch B of a PET is defined by $B = \{x_1 = X_1, \ldots, x_m = X_m\}$ where X_i is a value of the attribute x_i. Given the training dataset $\mathcal{D}$, the class probability of C_k given a branch B is calculated by the proportion of data belonging to C_k to all the data covered by B:

$$P(C_k|B) = \frac{\sum_{(i:\mathbf{x}_i \in B \wedge \mathbf{x}_i \rightarrow C_k)} \mathbf{x}_i}{\sum_{(i:\mathbf{x}_i \in B)} \mathbf{x}_i} \tag{1}$$

where $\mathbf{x} \in B$ represents that the data element $\mathbf{x}$ is covered by B. In decision tree learning, a decision tree may not contain all the attributes of the given training data $\mathcal{D}$. Given a new example $\mathbf{x}$, it is classified by a trained PET T as follows:

$$P(C_k|\mathbf{x}) = \sum_{j \in |T|} P(C_k|B_j) P(B_j|\mathbf{x}) \tag{2}$$

where $|T|$ is the number of branches of the tree T. $P(B_j|\mathbf{x}) = 1$ if $\mathbf{x}$ is covered by branch B_j. Otherwise, $P(B_j|\mathbf{x}) = 0$. For continuous attributes, a percentile-based discretization method is used. By this method, we simply partition the continuous universe into a fixed number of intervals based on the data distribution in order that each interval covers approximately the same number of data elements.

3 Naive Bayes Classification

So-called "naive" Bayes classification is a method of supervised learning if the attributes are conditionally independent given the classes. Although this assumption is almost always violated in practice, Naive Bayesian learning is remarkably effective in practice [2]. Given a test instance is presented, the learner is asked to predict its class according to the evidence provided by the training data. We define c as a random varibale denoting the class of an instance: $\mathbf{x} = \langle x_1, \ldots, x_n \rangle$ as a vector of variables denoting the observed attribute values. The expected classification error can be minimized by choosing $argmax_c(P(c = C|\mathbf{x}))$, according to Bayes's theorem:

$$P(\mathcal{H}|\mathcal{D}) = \frac{P(\mathcal{D}|\mathcal{H})}{\mathcal{H}} \tag{3}$$

Given a particular class $C_k \in \mathcal{C}$, where $\mathcal{C}$ represents the set of classes, we can obtain:

$$P(C_k|\mathbf{x}) = \frac{P(\mathbf{x}|C_k)P(C_k)}{P(\mathbf{x})} \tag{4}$$

Since the denominator in eq. 4 is invariant across classes, we can consider it as a normalization parameter. So, we obtain:

$$P(C_k|\mathbf{x}) \propto P(\mathbf{x}|C_k)P(C_k) \tag{5}$$

Now suppose we assume for each variable x_j that its outcome is independent of the outcome of all other variables given class C_k. In this case we can obtain the so-called naive Bayes classifier as follows:

$$P(C_k|\mathbf{x}) \propto \prod_{j=1}^{n} P(x_j|C_k)P(C_k) \tag{6}$$

where $P(x_j|C_k)$ is often called the likelihood of the data x_j given C_k. For a qualitative attribute, it can be estimated from corresponding frequencies. For a quantitative or continuous attribute, either probability density estimation or discretization can be employed to estimate its probabilities. Under probability density estimation, if the assumed density is not a proper estimate of the true density, the Naive-Bayes classification accuracy tends to degrade. Yang and Webb [10] argue that as long as the attribute independence assumption holds and discretization satisfies $P(c = C_k|\mathbf{x}^* = \mathbf{X}^*) = P(c = C_k|\mathbf{x} = \mathbf{X})$ (where instance $\mathbf{X}_i^*$ is the discretized version of instance $\mathbf{x}$), discretization will result in Naive-Bayes classifiers delivering probability estimates directly proportional to those that would be obtained if the correct probability density function were employed. This is also the reason why we use the percentile-based discretization for continuous attributes.

4 Naive Bayes Estimation Given a PET

Given a probability estimation tree T which is learnt from $\mathcal{D}$. According to the Bayesian theorem: A data element $\mathbf{x} = \langle x_1, \ldots, x_n \rangle$ can be classified by:

$$P(C_k|\mathbf{x},T) \propto P(\mathbf{x}|C_k,T)P(C_k|T) \quad (7)$$

We can then divide the attributes into 2 disjoint groups denoted by $\mathbf{x}_T = \{x_1, \cdots, x_m\}$ and $\mathbf{x}_B = \{x_{m+1}, \cdots, x_n\}$, respectively. $\mathbf{x}_T$ is the vector of the variables that are contained in the given tree T and the rest variables are contained in $\mathbf{x}_B$. Under the independence assumption, we obtain:

$$P(\mathbf{x}|C_k,T) = P(\mathbf{x}_T|C_k,T)P(\mathbf{x}_B|C_k,T) \quad (8)$$

Because $\mathbf{x}_B$ is independent of the given decision tree T. If we assume the variables in $\mathbf{x}_B$ are independent each other given a particular class, we can obtain:

$$P(\mathbf{x}_B|C_k,T) = P(\mathbf{x}_B|C_k) = \prod_{j\in\mathbf{x}_B} P(x_j|C_k) \quad (9)$$

Now consider $\mathbf{x}_T$. According to the Bayes theorem,

$$P(\mathbf{x}_T|C_k,T) = \frac{P(C_k|\mathbf{x}_T,T)P(\mathbf{x}_T|T)}{P(C_k|T)} \quad (10)$$

Combine eq. 8, 9 and 10:

$$P(\mathbf{x}|C_k,T) = \frac{P(C_k|\mathbf{x}_T,T)P(\mathbf{x}_T|T)}{P(C_k|T)} \prod_{j\in\mathbf{x}_B} P(x_l|C_k) \quad (11)$$

Combine eq. 7 and 11:

$$P(C_k|\mathbf{x},T) \propto P(C_k|\mathbf{x}_T,T)P(\mathbf{x}_T|T) \prod_{j\in\mathbf{x}_B} P(x_j|C_k) \quad (12)$$

Since $P(\mathbf{x}_T|T)$ is independent from C_k, so that:

$$P(C_k|\mathbf{x},T) \propto P(C_k|\mathbf{x}_T,T) \prod_{j\in\mathbf{x}_B} P(x_j|C_k) \quad (13)$$

where $P(x_j|C_k)$ is evaluated by the frequency of $x_j = X_j$ in all the data covered by C_k and $P(C_k|\mathbf{x}_T,T)$ is the just the class probabilities associated to each branch of the given tree T which is evaluated by eq. 2.

The basic idea of using Naive Bayes estimation given a PET is to use the PET as one estimator and the rest of the attributes as other independent estimators. Consider the two extreme cases for eq. 13. If all the attributes are used in building the tree (i.e. $\mathbf{x}_T = \mathbf{x}$), the probability estimations are from the tree only, that is:

$$P(C_k|\mathbf{x},T) \propto P(C_k|\mathbf{x}_T,T)$$

If none of the attributes are used in developing the tree (i.e. $\mathbf{x} = \mathbf{x}_B$), the probability estimation will become:

$$P(C_k|\mathbf{x},T) \propto \prod_{j\in\mathbf{x}_B} P(x_j|C_k)$$

which is simply a Naive Bayes classifier. If we extend this idea, we use a set of small-sized LDTs as estimators, we then have the second hybird model which is described in the next section.

5 Naive Bayes Estimation From a Set of Trees

Given a training dataset, a small-sized tree (usually the depth is less than 3) can be learnt based on the method we discussed in section 4. Then we learn another tree with the same size based on the rest of the attributes, i.e., the attributes which have not been used in previous trees. Successively, a set of trees can be built from training set. If we denote the trees by $\mathcal{T} = \langle T_1, \ldots, T_W \rangle$, for each tree T_w, the set of attributes $\mathbf{x}_{T_w}$ are exclusive each other for $w = 1, \ldots, W$. For a given unclassified data element $\mathbf{x}$, we can partition it into W group of disjoint set of attributes $\langle \mathbf{x}_{T_1}, \ldots, \mathbf{x}_{T_W} \rangle$. If we assume:

$$P(C_k|\mathbf{x}) = P(C_k|\mathbf{x}_{T_1}, \ldots, \mathbf{x}_{T_W}) \approx P(C_k|T_1, \ldots, T_W) \quad (14)$$

Which means: the class probability given a vector of variables can be estimated by the product of class probabilities of a set of disjoint trees (i.e., the attributes for building these trees are disjoint groups). Then, according to the Bayesian theorem:

$$P(C_k|T_1, \ldots, T_W) = \frac{P(T_1, \ldots, T_W|C_k)P(C_k)}{P(T_1, \ldots, T_W)} \quad (15)$$

Since we assume the trees are built independently and we assume the attributes are independent to each other. So that:

$$P(T_1, \ldots, T_W|C_k) = \prod_{w=1}^{W} P(T_w|C_k) \quad (16)$$

For a particular tree T_w for $w = 1, \ldots, W$, we have

$$P(T_w|C_k) = \frac{P(C_k|T_w)P(T_w)}{P(C_k)} \quad (17)$$

So that,

$$\prod_{w=1}^{W} P(T_w|C_k) = \frac{\prod_{w=1}^{W} P(C_k|T_w) \prod_{i=1}^{W} P(T_w)}{P(C_k)^W} \quad (18)$$

Combining eq. 15, 16 and 18 gives

$$P(C_k|\mathcal{T}) \propto \frac{\prod_{w=1}^{W} P(C_k|T_w) \prod_{w=1}^{W} P(T_w)}{P(C_k)^{W-1}} \quad (19)$$

Since $\prod_{w=1}^{W} P(T_w)$ is independent from C_k, we finally obtain:

$$P(C_k|\mathcal{T}) \propto \frac{\prod_{w=1}^{W} P(C_k|T_w)}{P(C_k)^{W-1}} \quad (20)$$

where $P(C_k|T_w)$ is evaluated according to eq. 2. The depth of the disjointed trees are predefined with a small number. E.g., if the depth is 2: given the training data, we selected two most discriminate attributes to build a tree. Then, we select the other two from the remaining attributes to build the 2nd tree, successively, till all the attributes have been used.

Table 1. Descriptions of the datasets used in experiments. They are taken from the UCI [1] Machine Learning Repository. Num: Numeric attributes, Nom: Nominal attributes.

Dataset	Class	Size	Missing Values	Num. of Attri. *Num.*	*Nom.*
Ecoli	8	336	no	7	1
Glass	6	214	no	9	0
Heptitis	2	155	yes	6	13
Ionosphere	2	351	no	34	0
Iris	3	150	no	4	0
Liver	2	345	no	6	0
Pima	2	768	no	8	0
Wine	3	178	no	14	0
Wis. Cancer	2	699	no	9	0

Table 3. Result comparisons at the depth 2 based on t-test with 90% confidence, where '$\surd$' represents significant better, '$-$' equivalence and '$\times$' significant worse.

Database	BPET vs NB	BPET vs PET	FPET vs NB	FPET vs PET
Ecoli	$-$	$\surd$	$-$	$\surd$
Glass	$\surd$	$\surd$	$\times$	$-$
Heptitis	$-$	$-$	$-$	$-$
Iono.	$-$	$-$	$\surd$	$-$
Iris	$-$	$-$	$-$	$-$
Liver	$-$	$-$	$-$	$\times$
Pima	$-$	$\times$	$-$	$\times$
Wine	$-$	$\surd$	$-$	$\surd$
Wis. Cancer	$-$	$\surd$	$\surd$	$\surd$

6 Experimental Studies

We evaluated the PET model, Naive Bayes and two new proposed hybrid models: single PET with Bayesian estimation (denoted by BPET) and Bayesian estimation with a set of trees (denoted by FPET - namely, a forest of PETs) on 9 datasets taken from the UCI Machine Learning repository [1]. Table 1 shows the datasets, the number of classes, the number of instances and the number of attributes. Unless otherwise stated, attributes are discretized by 3 intervals based on the percentile-based discretization (or equal-point discretization, each interval approximately covers the same number of points. See section 2), and classes are evenly split into two sub-datasets, one half for training and the other half for testing, this is referred to as 50-50 split experiment. We first compare the performances of the BPET and PET model. For each dataset, we ran 50-50 random split experiment for 10 times and the average test accuracies with standard deviations are shown against depths of the trees are shown in figures 1 and 2. The results of Naive Bayes and the best resutls of PET, BPET and FPET are shown in table 2, where 'Depth' for PET, BPET and FPET represents the depth at which the best results were obtained.

From all the figures, we can see that BPET model generally peroforms better at the shallow depths than the PET model. However, with the increasing of the depth, the peformance for BPET model keeps same or decreasing, while the accuracy curves for PETs go upward. For datasets Ecoli, Glass, Ionosphere, Live, Wine and Wisconsin-Cancer, the BPET model performs better at most of depths. For Heptitis, the differences are insignificant at all depths. For Iris and Pima, the PET model even performs better than the BPET model in most the depths but the differences are not significant.

We performed t-tests with a confidence level of 90% [2] to compare the models at depth 2 (except for the Naive Bayes model) and the results are shown in table 3. We can see that the BPET model performs better than the PET model and it has equivalent performance comparing to Naive Bayes. The FPET model is equivalent to the PET model and Naive Bayes. However, the computational complexity is much larger than Naive Bayes.

From figures 1 and 2, we found that most best results for BPET are obtained at shallow depths, but for PETs the best results are always obtained with deep depths. So, we can conclude that BPET model is more efficient than PET. Compare to the BPET model, the FPET model performs relative worse and less efficient, the reasons are probably because that small-trees are not good estimators. But this still needs more further investigation. In summary, the BPET model outperforms PET at shallow depths and has equivalent performance to PET in general. In the comparisons between FPET and PET, due to the limited number of test sets, at least we can say they are equivalent instead of which one is better than another.

7 Conclusions

In this paper, we propsed two hybrid models by combining Naive Bayes classifier and probability estimation trees. Through experimental studies, we found that the BPET (the Naive Bayes estimation model given a PET) model outper-

[2] We generally believe that the confidence level of 90 % is enough to be significant for comparisons among different learning models given these relatively simple data sets.

Table 2. Experimental results on 9 UCI datasets: average accuracy with standard deviation from 10 runs of random 50-50 split experiments.

	Naive Bayes	PET		BPET		FPET	
Data	Accuracy	Accuracy	Depth	Accuracy	Depth	Accuracy	Depth
Ecoli	78.29±2.29	73.76±4.52	6	79.24±1.88	1	80.18±3.45	1
Glass	57.06±5.22	52.20±3.19	9	60.92±5.92	4	50.46±3.21	2
Hep.	80.64±2.38	80.38±3.42	4	82.05±2.34	2	80.64±3.11	1
Ionosphere	83.30±3.50	82.39±2.32	3	85.28±2.07	3	87.44±3.83	2
Iris	92.53±2.96	94.93±2.07	2	92.53±3.57	1	93.07±2.65	1
Liver	63.24±2.42	61.39±3.71	4	63.24±3.36	1	65.20±2.69	2
Pima	72.06±2.86	73.52±2.62	2	71.48±2.52	1	70.36±1.13	1
Wine	94.33±3.61	87.22±2.41	2	94.22±3.66	1	95.89±3.06	1
Wis. Cancer	96.63±0.40	94.51±1.16	3	96.54±0.39	1	98.00±1.09	2

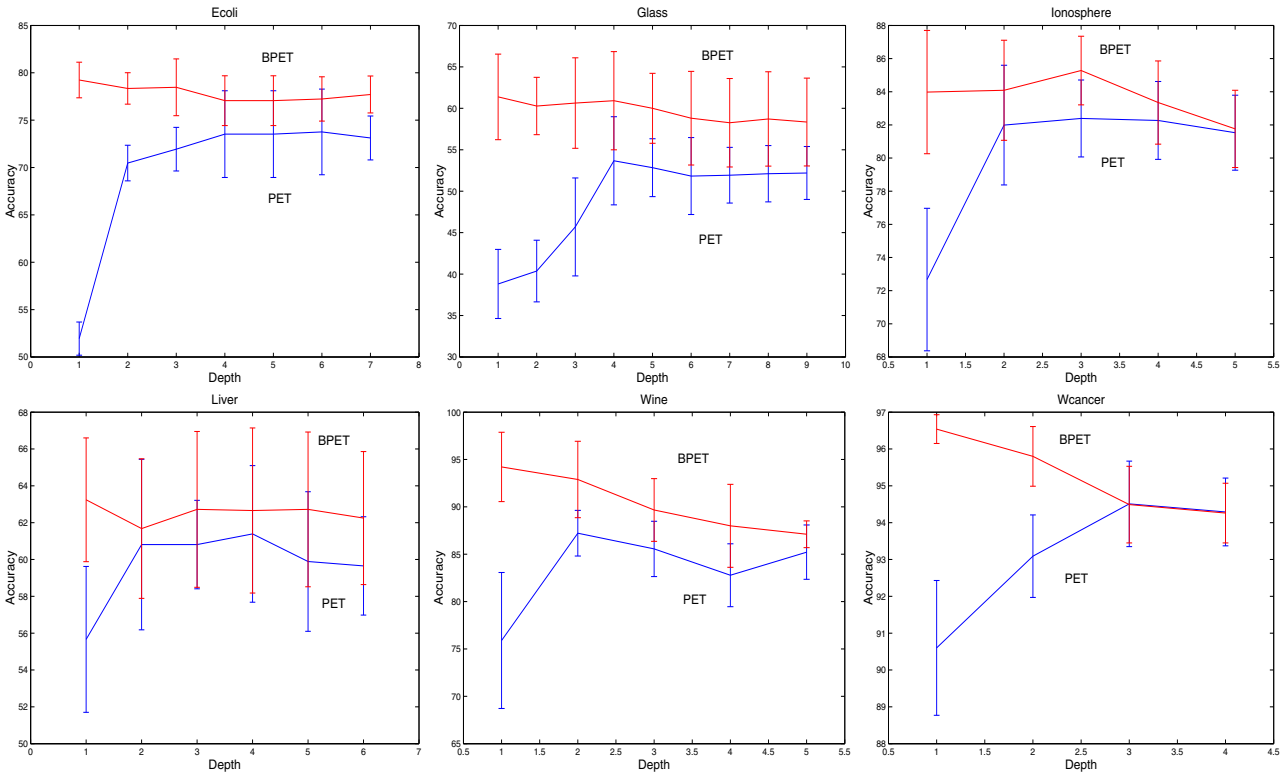

Figure 1. Results for single PET with Naive Bayes estimation: average accuracy with standard deviation on each dataset against the depth of the tree. Datasets: Ecoli, Glass, Ionosphere, Liver, Wine and Wisconsin-Cancer.

Proceedings of the 5th International Conference on Machine Learning and Applications (ICMLA'06)
0-7695-2735-3/06 $20.00 © 2006 IEEE

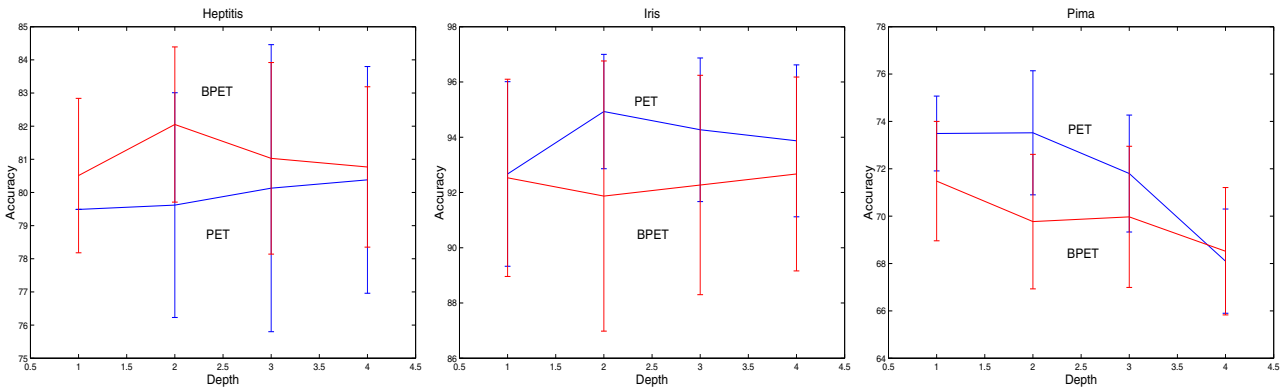

Figure 2. Results for single PET with Naive Bayes estimation: average accuracy with standard deviation on each dataset against the depth of the tree. Datasets: Heptitis, Iris and Pima.

forms the probability estimation tree model at shallow tree depths. Comparing to Naive Bayes, BPET has the equivalent performance. However, the transparency is greatly improved since Naive Bayes is a pure black-box model. The FPET (using a set of small-size PETs as Bayesian estimators) model has the equivalent performance comparing to Naive Bayes classifier PETs.

In this paper, the continuous attributes are arbitrarily discretized by predefined number of discrete intervals. However, we used the same discretization for all the models for comparisons. We think that it is fair for studying the relative performance among these models. If we use fuzzy labels for discretization, PETs then become fuzzy probability estimation trees. More research about fuzzy probability estimation trees can be found in [7]. Further research focus on investigating the reasons that FPETs are not good Bayesian estimators and testing on more datasets.

Acknowledgment

This work was partly conducted when the author was at the Engineering Mathematics Department of the University of Bristol. This work was funded by the ORS (UK) and BT/BISC Fellowship.

References

[1] C. Blake and C.J. Merz. UCI machine learning repository. *http://www.ics.uci. edu/∼mlearn/ MLRepository.html*.

[2] C. Elkan. Naive Bayesian Learning. *Technical Report No. CS97-557, Dept. of Computer Science and Engineering, UCSD*, 1997.

[3] C. Ferri, P.A. Flach and J. Hernández-Orallo. Learning decison trees using the area under the ROC curve. *Proceedings of the ICML-02*. Sydney, Australia, pp. 139-146, Morgan Kaufmann, 2002.

[4] C. Ferri, P.A. Flach and J. Hernández-Orallo. Improving the AUC of Probabilistic Estimation Trees. *The Proceedings of ECML2003*, LNAI 2837, pp. 121-132, 2003.

[5] F. Provost, T. Fawcett, and R. Kohavi. The case against accuracy estimation for comparing induction algorithms. in J. Shavlik, editor, *Prced. of ICML98*, pp. 445-453, 1998.

[6] F. Provost and P. Domingos. Tree induction for probability-based ranking. *Machine Learning*. 52, 199-215, 2003.

[7] Z. Qin and J. Lawry. Decision tree learning with fuzzy labels. *Information Sciences*. 172/1-2, pp. 91-129, 2005.

[8] J.R. Quinlan. Induction of decision trees. *Machine Learning* 1: 81-106. 1986

[9] J.R. Quinlan. *C4.5: Programs for Machine Learning*. San Mateo: Morgan Kaufmann, 1993.

[10] Y. Yang and G. I. Webb, On why discretization works for Naive-bayes classifiers, *Australian Conference on Artificial Intelligence*, pp. 440-452, 2003.

[11] I. H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*, Morgan Kaufmann, 1999. http://www.cs.waikato.ac.nz/∼ml/weka/