

# A Comparison of Approaches for Learning Probability Trees

*Daan Fierens*

*Jan Ramon*

*Hendrik Blockeel*

*Maurice Bruynooghe*

*Report CW418, July 2005*



Katholieke Universiteit Leuven  
Department of Computer Science

Celestijnenlaan 200A – B-3001 Heverlee (Belgium)

# A Comparison of Approaches for Learning Probability Trees

*Daan Fierens*

*Jan Ramon*

*Hendrik Blockeel*

*Maurice Bruynooghe*

*Report CW 418, July 2005*

Department of Computer Science, K.U.Leuven

## Abstract

Probability trees (or Probability Estimation Trees, PET's) are decision trees with probability distributions in the leaves. Several approaches for learning probability trees have been proposed in the literature. Currently no thorough comparison of these alternative approaches exists.

In this paper we experimentally compare the main approaches using the relational decision tree learner Tilde (both on non-relational and on relational datasets). Next to the main existing approaches, we also consider a novel variant of an existing approach based on the Bayesian Information Criterion (BIC). Our main conclusion is that trees built using the C4.5-approach or the C4.4-approach (C4.5 without post-pruning) typically have the best predictive performance. If the number of classes is low, however, BIC is equally good. An additional advantage of BIC is that its trees are considerably smaller than trees for the C4.5- or C4.4-approach.

**Keywords :** Decision trees, Probability estimation, Inductive Logic Programming

**CR Subject Classification :** I.2.6

# 1 Introduction

Probability trees (or Probability Estimation Trees, PET's) are decision trees with in the leaves probability distributions over a set of classes [13]. Probability trees are useful in a number of ways: they can be used for ranking instances according to the probability of belonging to a certain class [13] or as a compact way of specifying conditional probability distributions, for instance in Bayesian networks or dependency networks (as done in [5, 6] and [7]).

Several approaches for learning probability trees have been proposed in the literature. Provost and Domingos [13] use C4.5 and C4.4 (i.e. C4.5 without post-pruning), Neville et al. [12] use the chi-square statistic and Friedman and Goldszmidt [5, 6] use a Minimum Description Length (MDL) approach. In this paper we also consider a novel variant of the latter based on the Bayesian Information Criterion (BIC). Currently no thorough comparison of the above alternative approaches exists. Hence, it is unclear which approaches are preferable under which circumstances.

The goal of this paper is to compare the above approaches for learning probability trees. We incorporated them into the relational decision tree learner Tilde [2] and evaluate them by performing experiments on some well-known benchmark datasets (both non-relational and relational). We also perform a set of controlled experiments on manipulated datasets to analyze the influence of specific parameters on the predictive performance and the size of the trees.

The rest of this paper is organized as follows. In Section 2 we give a high-level algorithm for learning probability trees, of which the main existing approaches are instantiations. In Section 3 we experimentally compare these approaches. In Section 4 we conclude. Parts of this report have been published before [3, 4].

## 2 Learning Probability Trees

### 2.1 Learning Probability Trees: the High-level Algorithm

Probability trees can be learned from the same kind of data as ordinary classification trees: a dataset  $D$  of instances labelled with their true class. In this paper we use Tilde [2], a *relational* decision tree learner. Tilde uses as a representation for each instance a first-order logic interpretation (basically a set of Prolog facts). Tests in internal nodes are Prolog queries. Since such tests either succeed or fail, trees in Tilde are always binary. The main reason why we use Tilde is that (because of its first-order logic nature) it can handle relational datasets in addition to non-relational ones. When applied on non-relational datasets, Tilde behaves very similar to standard non-relational tree learners such as C4.5 [14].

Probability trees are typically learned in two steps. The first step follows the standard approach of top-down induction of decision trees. A high-level algorithm for top-down induction of probability trees with Tilde is given in Figure 1. Note that each leaf contains a probability distribution on the set of classes. We use Laplace correction when estimating probabilities: the estimated probability of class  $c_i$  in a leaf is  $\frac{N_{leaf,i}+1}{N_{leaf}+NbClasses}$  with  $N_{leaf}$  (resp.  $N_{leaf,i}$ ) the number of examples (resp. the number of examples of class  $c_i$ ) in the leaf and  $NbClasses$  the number of classes. Provost and Domingos showed that the Laplace correction significantly improves the performance of probability trees [13].

To avoid overfitting when learning decision trees, one often applies *post-pruning* [14]: we first build a tree  $\mathcal{T}$  using the above algorithm and then we prune  $\mathcal{T}$  in a bottom-up manner. A high-level algorithm for post-pruning trees is shown in Figure 2: to prune  $\mathcal{T}$ , first prune its subtrees giving  $\mathcal{T}_{pruned}$ , then check whether  $\mathcal{T}_{pruned}$  is 'better' than a single leaf  $\mathcal{L}$  according to some pruning-criterion.

```

To learn a tree from a set of examples  $D$ :
if  $D$  is pure
  then return a leaf
else
  for each candidate-test  $T$ 
    compute heuristic  $h(T)$  of  $T$ 
     $T$  is acceptable iff  $h(T) \geq Thr$ 
  if no candidate-test is acceptable
    then return a leaf
  else
     $T_{root} = \text{argmax}_T(h(T))$ 
    use  $T_{root}$  to split  $D$  in  $D_{left}$  and  $D_{right}$ 
    learn a tree  $\mathcal{T}_{left}$  from  $D_{left}$ 
    learn a tree  $\mathcal{T}_{right}$  from  $D_{right}$ 
    return composite tree  $(T_{root}, \mathcal{T}_{left}, \mathcal{T}_{right})$ 

```

Figure 1: High-level algorithm for top-down induction of probability trees.

```

To post-prune a tree  $\mathcal{T}$ :
if  $\mathcal{T}$  is a leaf
  then return  $\mathcal{T}$ 
else
  prune left subtree of  $\mathcal{T}$  giving  $\mathcal{T}_{left}$ 
  prune right subtree of  $\mathcal{T}$  giving  $\mathcal{T}_{right}$ 
   $T_{root}$  is the root of  $\mathcal{T}$ 
   $\mathcal{T}_{pruned}$  is  $(T_{root}, \mathcal{T}_{left}, \mathcal{T}_{right})$ 
  if pruning-criterion for  $\mathcal{T}_{pruned}$  is satisfied
    then collapse  $\mathcal{T}_{pruned}$  into a leaf  $\mathcal{L}$ 
    return  $\mathcal{L}$ 
  else
    return  $\mathcal{T}_{pruned}$ 

```

Figure 2: High-level algorithm for post-pruning probability trees.

The most important parameters of the above high-level approach are a) the heuristic function  $h(\cdot)$  used to assess the quality of a test on the data  $D$ , b) the threshold  $Thr$  to check whether a test’s heuristic value is high enough to be considered and c) whether post-pruning is done or not (and, if so, the pruning-criterion).

## 2.2 Existing Instantiations of the High-level Algorithm

We now briefly review the main existing approaches for learning probability trees showing that they are instantiations of the above high-level algorithm.

**C4.5 (error-based post-pruning)** Provost and Domingos [13] discuss learning probability trees using C4.5 [14]. This means that  $h(T)$  is information-gain of  $T$  ( $gain(T)$ ),  $Thr$  is 0 (any information-gain is acceptable) and error-based post-pruning is applied<sup>1</sup>. Below we refer to Tilde applied with these parameters as **C4.5** (when applied on non-relational data, the only difference between Tilde with the above parameters and the actual C4.5 is that Tilde always builds binary trees).

**C4.4 (no pruning)** Provost and Domingos [13] argue that pruning is harmful for probability trees. The motivation is that probability estimation is conceptually different from majority-classification (the focus of C4.5): e.g. splitting a node with 75% positive examples into a node with 60% positive examples and a node with 90% positive examples is needless for majority-classification but very useful for probability estimation. They propose to use C4.4 which is the same as C4.5 as discussed above, except that no post-pruning is used at all. We refer to Tilde applied with these parameters as **C4.4**. Obviously, **C4.4** builds extremely large trees.

**Minimum Description Length (MDL)** Friedman and Goldszmidt [5] define an MDL-score for probability trees. This MDL-score can be used to derive a *stopping-criterion* for the tree-building. Concretely,  $h(T)$  is  $N_{node} \cdot gain(T)$  and  $Thr$  is  $0.5 (NbClasses - 1) \log_2 N + \log_2 NbTests + 2$ , where  $N$  is the number of examples in the dataset,  $N_{node}$  is the number of examples in the current node and  $NbTests$  is the number of candidate-tests considered. In terms of MDL,  $h(T)$  is the decrease in description length of the data due to adding  $T$  to the tree and  $Thr$  is the increase in description length of the tree due to adding  $T$  (the first term in  $Thr$  is the number of bits to encode a leaf, the second is the number of bits to encode  $T$  and two extra bits are needed for theoretical correctness [5, 6]). We refer to this approach as **MDLs**.

Using MDL as a stopping-criterion (using the above  $Thr$ ) we easily get stuck in local optima of the MDL-score. As an alternative we can use  $Thr = 0$  and apply *post-pruning* based on MDL-reasoning [5, 6]. We refer to this approach as **MDLp**. Note that **MDLp** builds trees at least as large as those for **MDLs**.

**Bayesian Information Criterion (BIC)** Inspired on the above MDL-score, we can define a BIC-score for probability trees. BIC [15] is a general approach related to MDL where only the number of independent parameters of a model is taken into account to define the description length of a model<sup>2</sup>. For probability trees, this means that we take into account the description length of the leaves (since they contain the parameters, i.e. the probabilities) but not the internal nodes (the tests). Concretely,  $h(T)$  is  $N_{node} \cdot gain(T)$  and  $Thr$  is  $0.5 (NbClasses - 1) \log_2 N$ . We refer to this approach as

<sup>1</sup>Like Provost and Domingos, we do not apply ‘collapsing’ (a post-pruning technique taking together two children of the same node if they have the same majority class) since it harms probability estimates too much [13].

<sup>2</sup>The BIC-score as we use it is actually the negative logarithm of the BIC-score as usually stated, but both are of course equivalent.

Table 1: Main existing instantiations of the high level algorithms of Figures 1 and 2.

	heuristic	$Thr$	post-pruning
<b>C4.5</b>	$gain$	0	error-based
<b>C4.4</b>	$gain$	0	none
<b>MDLs</b>	$N_{node.gain}$	$0.5 (NbClasses - 1) \log_2 N + \log_2 NbTests + 2$	none
<b>MDLp</b>	$N_{node.gain}$	0	MDL-based
<b>BICs</b>	$N_{node.gain}$	$0.5 (NbClasses - 1) \log_2 N$	none
<b>BICp</b>	$N_{node.gain}$	0	BIC-based
<b>Chi</b>	$\chi^2$	$\chi^2$ -cutoff for $p=0.1/NbTests$ and $df=NbClasses-1$	none

**BICs.** Note that **BICs** builds trees at least as large as those for **MDLs** (since they use the same  $h(\cdot)$  but  $Thr$  is strictly lower for **BICs** than for **MDLs**).

As an alternative we can again use  $Thr = 0$  and apply post-pruning based on BIC-reasoning. We refer to this approach as **BICp**. Note that **BICp** builds trees at least as large as those for **BICs**.

**Chi-square score** Neville et al. [12] use the chi-square ( $\chi^2$ ) statistic to check how significantly different the class-distributions in two leaves are from the class-distribution in their parent node. Hence,  $h(T)$  is the  $\chi^2$ -score of  $T$ .  $Thr$  is determined by the sampling distribution of  $\chi^2$  for significance level  $p = \frac{0.1}{NbTests}$  and degrees of freedom  $df = NbClasses - 1$ . No post-pruning is used. We refer to this approach as **Chi**.

In theory we could as an alternative again use  $Thr = 0$  and apply  $\chi^2$ -based post-pruning. Because of statistical peculiarities, however, this is complicated and we do not consider it (neither do Neville et al. [12]).

An overview of the above approaches is given in Table 1. As far as we know, we are the first to apply BIC to probability trees (relational or non-relational) and to apply MDL to relational probability trees.

Other approaches for learning probability trees have been proposed as well. Neville et al. [12] use randomization tests to derive  $Thr$ . We do not consider this approach in our comparison because it is computationally much more intensive than any of the other approaches we consider (it requires generating many replicates of the actual dataset [12]). Heckerman et al. [7] and Friedman and Goldszmidt [5, 6] use as the heuristic function  $h(\cdot)$  a Bayesian score incorporating a prior distribution over probability trees. We do not consider this approach since we believe it is often difficult in practice to specify such a prior distribution. Finally, it is possible to simulate probability trees for two-class problems with regression trees (predicting the probability of one class. We do not consider this approach because it only works for two-class problems and not for multi-class problems.

### 3 Experiments and Discussion

To the best of our knowledge, C4.5 and C4.4 are the only two approaches from Table 1 that have already been compared to each other. From an experimental study Provost and Domingos [13] concluded that neither of the two is significantly better than the other (in contrast to the expectation that C4.4 would be better than C4.5 for probability estimation). In this section we make a thorough comparison of all approaches from Table 1.

Table 2: Characteristics of the non-relational (left) and relational (right) datasets: number of examples, number of classes and number of candidate-tests for the root.

	$N$	NbClasses	NbTests		$N$	NbClasses	NbTests
<i>asm</i>	999	2	170	<i>biodegradability</i>	328	2	47
<i>audiology</i>	226	24	125	<i>carcinogenesis</i>	330	2	305
<i>pen digits</i>	7494	10	160	<i>diterpenes</i>	1504	23	210
<i>primary tumor</i>	339	22	29	<i>hiv</i>	41768	2	49
<i>voting</i>	435	2	16	<i>mutagenesis</i>	230	2	139
<i>yeast</i>	1484	10	45	<i>trains</i>	25000	2	73

### 3.1 Experiments on Benchmark Datasets: Setup and Results

#### 3.1.1 Experimental Setup

Table 2 gives an overview of the datasets used. All non-relational datasets are from the UCI-repository [10], except *asm* [8]. All relational datasets are standard ILP-benchmarks [1, 9, 16] (*trains* was artificially generated [11]; for *hiv* the classes ‘inactive’ and ‘moderately active’ were taken together).

To evaluate the predictive performance of probability trees we use the *Area Under the ROC-curve (AUC)* [13]. For two-class problems the AUC is the probability that a randomly selected true positive example is predicted more likely to be positive than a randomly selected true negative example. For multi-class problems we use the *Expected AUC*, a weighted average of the AUC’s obtained by using in turn each of the classes as positive and all others as negative [13]. As noted in [13], AUC can be used as a quality measure for probability estimates since a high AUC indicates that, with proper re-calibration of probabilities, probability estimates will be good.

To evaluate the size of the trees learned, we use the *number of leaf nodes*. Note that the number of internal nodes is the number of leaf nodes minus one (since all trees are binary).

We perform 10-fold cross-validation (except for datasets smaller than 500 examples where we perform five times 3-fold cross-validation to keep test-sets large enough) and report averages and standard deviations of the results over the test-sets.

#### 3.1.2 Experimental Results

Table 3 shows our experimental results for AUC. To make results easier to interpret, we statistically compared results for all approaches to results for **C4.4** (using two-tailed paired t-tests ( $p=0.05$ ) on the results of all folds in the cross-validation). A result is shown in boldface if it is significantly better (higher) than the result for **C4.4**. A result is shown underlined if it is significantly worse (lower) than the result for **C4.4**. As a summary, Table 3 also shows the number of wins/ties/losses of each of the approaches versus **C4.4** on all datasets, on two-class datasets and on multi-class datasets.

Similarly, Table 4 shows our experimental results for tree size, again compared to the results for **C4.4** (again boldface means better; in the case of tree size, this means lower).

Table 5 (AUC) and Table 6 (tree size) show the same results as Table 3 and Table 4 but now compared to **C4.5**. We chose **C4.4** and **C4.5** as references for comparisons because they are the two best approaches in our experiments.

Table 7 shows the *running times* for the different approaches. The main conclusion is that the approaches using an explicit stopping-criterion (i.e. the approaches for which  $Thr > 0$ : **MDLs**, **BICs** and **Chi**) are much faster than the others. This is because the others always first build a very large tree, which consumes a lot of time. Note that

Table 3: Experimental results: AUC in % (using **C4.4** as a reference).

	<b>C4.4</b>	<b>C4.5</b>	<b>MDLs</b>	<b>MDLp</b>	<b>BICs</b>	<b>BICp</b>	<b>Chi</b>
<i>asm</i>	58.7±4.7	<b>62.8±3.7</b>	<b>69.6±4.4</b>	<b>69.6±4.4</b>	<b>67.4±3.8</b>	<b>66.0±3.5</b>	<b>69.5±4.3</b>
<i>audiology</i>	98.8±0.8	98.7±0.7	<u>75.3±2.7</u>	<u>75.6±2.4</u>	<u>80.9±5.5</u>	<u>81.2±5.0</u>	<u>97.4±1.1</u>
<i>pen digits</i>	99.5±0.1	<u>99.4±0.1</u>	<u>98.7±0.2</u>	<u>98.7±0.2</u>	<u>98.9±0.3</u>	<u>98.9±0.3</u>	99.4±0.1
<i>primary tumor</i>	71.5±3.1	73.3±4.3	<u>65.8±5.3</u>	<u>67.7±1.8</u>	<u>67.9±1.9</u>	<u>67.5±1.9</u>	72.7±4.0
<i>voting</i>	98.6±0.7	<u>96.5±2.1</u>	<u>97.4±1.1</u>	97.7±1.4	98.3±1.5	98.4±0.9	98.5±1.2
<i>yeast</i>	75.8±3.2	<b>79.6±3.5</b>	<b>78.3±2.5</b>	<b>78.3±2.5</b>	<b>78.3±2.5</b>	<b>78.3±2.5</b>	<b>79.1±4.0</b>
<i>biodegradability</i>	74.9±6.7	75.4±4.8	<u>63.9±4.0</u>	<u>64.1±2.9</u>	73.0±4.7	71.6±5.3	<u>64.5±4.0</u>
<i>carcinogenesis</i>	59.1±5.8	59.3±6.7	<u>50.0±0.0</u>	<u>50.0±0.0</u>	<u>54.0±2.7</u>	57.2±3.2	<u>50.0±0.0</u>
<i>diterpenes</i>	85.4±2.5	85.8±1.8	<u>70.4±3.7</u>	<u>71.2±2.6</u>	<u>71.5±3.8</u>	<u>72.0±2.9</u>	<u>82.0±2.5</u>
<i>hiv</i>	74.8±3.3	53.9±1.1	<u>64.1±2.1</u>	<u>70.5±3.1</u>	<u>66.8±3.3</u>	<u>72.4±3.3</u>	<u>67.0±3.2</u>
<i>mutagenesis</i>	77.0±5.0	<u>71.7±4.9</u>	<u>71.6±7.0</u>	74.5±4.4	72.8±8.2	75.8±6.6	<u>71.9±6.0</u>
<i>trains</i>	86.3±0.5	<b>89.2±0.6</b>	<b>89.0±0.6</b>	<b>89.2±0.5</b>	<b>89.2±0.6</b>	<b>89.4±0.5</b>	<b>89.3±0.5</b>
		wins/ties/losses (all)	w/t/l (two-class)	w/t/l (multi-class)			
<b>C4.5 vs C4.4</b>		3/5/4	2/2/3	1/3/1			
<b>MDLs vs C4.4</b>		3/0/9	2/0/5	1/0/4			
<b>MDLp vs C4.4</b>		3/2/7	2/2/3	1/0/4			
<b>BICs vs C4.4</b>		3/3/6	2/3/2	1/0/4			
<b>BICp vs C4.4</b>		3/4/5	2/4/1	1/0/4			
<b>Chi vs C4.4</b>		3/3/6	2/1/4	1/2/2			

Table 4: Experimental results: tree size (using **C4.4** as a reference).

	<b>C4.4</b>	<b>C4.5</b>	<b>MDLs</b>	<b>MDLp</b>	<b>BICs</b>	<b>BICp</b>	<b>Chi</b>
<i>asm</i>	352±14	<b>64±11</b>	<b>3±0</b>	<b>3±0</b>	<b>6±2</b>	<b>7±3</b>	<b>3±0</b>
<i>audiology</i>	24±1	24±2	<b>2±0</b>	<b>2±0</b>	<b>3±1</b>	<b>3±1</b>	<b>18±3</b>
<i>pen digits</i>	254±8	<b>214±7</b>	<b>36±2</b>	<b>36±2</b>	<b>42±3</b>	<b>42±3</b>	<b>126±4</b>
<i>primary tumor</i>	129±5	<b>79±7</b>	<b>2±0</b>	<b>2±0</b>	<b>2±0</b>	<b>2±0</b>	<b>5±1</b>
<i>voting</i>	21±3	<b>6±3</b>	<b>3±1</b>	<b>3±1</b>	<b>6±1</b>	<b>6±1</b>	<b>6±1</b>
<i>yeast</i>	447±34	<b>123±13</b>	<b>6±0</b>	<b>6±0</b>	<b>6±0</b>	<b>6±0</b>	<b>23±3</b>
<i>biodegradability</i>	72±5	<b>30±5</b>	<b>2±1</b>	<b>2±1</b>	<b>10±2</b>	<b>12±4</b>	<b>4±2</b>
<i>carcinogenesis</i>	95±8	<b>35±7</b>	<b>1±0</b>	<b>1±0</b>	<b>5±2</b>	<b>8±3</b>	<b>1±0</b>
<i>diterpenes</i>	127±11	<b>64±5</b>	<b>3±1</b>	<b>4±1</b>	<b>4±1</b>	<b>4±1</b>	<b>14±2</b>
<i>hiv</i>	1391±76	<b>15±2</b>	<b>8±3</b>	<b>32±4</b>	<b>26±3</b>	<b>55±4</b>	<b>33±3</b>
<i>mutagenesis</i>	42±4	<b>9±5</b>	<b>2±0</b>	<b>2±0</b>	<b>5±2</b>	<b>5±2</b>	<b>2±0</b>
<i>trains</i>	4664±55	<b>484±29</b>	<b>38±2</b>	<b>49±4</b>	<b>68±6</b>	<b>92±6</b>	<b>64±3</b>
		wins/ties/losses (all)	w/t/l (two-class)	w/t/l (multi-class)			
<b>C4.5 vs C4.4</b>		11/1/0	7/0/0	4/1/0			
<b>MDLs vs C4.4</b>		12/0/0	7/0/0	5/0/0			
<b>MDLp vs C4.4</b>		12/0/0	7/0/0	5/0/0			
<b>BICs vs C4.4</b>		12/0/0	7/0/0	5/0/0			
<b>BICp vs C4.4</b>		12/0/0	7/0/0	5/0/0			
<b>Chi vs C4.4</b>		12/0/0	7/0/0	5/0/0			

Table 5: Experimental Results: AUC in % (using **C4.5** as a reference).

	<b>C4.5</b>	<b>C4.4</b>	<b>MDLs</b>	<b>MDLp</b>	<b>BICs</b>	<b>BICp</b>	<b>Chi</b>
<i>asm</i>	62.8±3.7	<u>58.7±4.7</u>	<b>69.6±4.4</b>	<b>69.6±4.4</b>	<b>67.4±3.8</b>	66.0±3.5	<b>69.5±4.3</b>
<i>audiology</i>	98.7±0.7	98.8±0.8	<u>75.3±2.7</u>	<u>75.6±2.4</u>	<u>80.9±5.5</u>	<u>81.2±5.0</u>	<u>97.4±1.1</u>
<i>pen digits</i>	99.4±0.1	<b>99.5±0.1</b>	<u>98.7±0.2</u>	<u>98.7±0.2</u>	<u>98.9±0.3</u>	<u>98.9±0.3</u>	<u>99.4±0.1</u>
<i>primary tumor</i>	73.3±4.3	71.5±3.1	<u>65.8±5.3</u>	<u>67.7±1.8</u>	<u>67.9±1.9</u>	<u>67.5±1.9</u>	72.7±4.0
<i>voting</i>	96.5±2.1	<b>98.6±0.7</b>	97.4±1.1	97.7±1.4	<b>98.3±1.5</b>	<b>98.4±0.9</b>	<b>98.5±1.2</b>
<i>yeast</i>	79.6±3.5	<u>75.8±3.2</u>	<u>78.3±2.5</u>	<u>78.3±2.5</u>	<u>78.3±2.5</u>	<u>78.3±2.5</u>	79.1±4.0
<i>biodegradability</i>	75.4±4.8	74.9±6.7	<u>63.9±4.0</u>	<u>64.1±2.9</u>	73.0±4.7	71.6±5.3	<u>64.5±4.0</u>
<i>carcinogenesis</i>	59.3±6.7	59.1±5.8	<u>50.0±0.0</u>	<u>50.0±0.0</u>	<u>54.0±2.7</u>	57.2±3.2	<u>50.0±0.0</u>
<i>diterpenes</i>	85.8±1.8	85.4±2.5	<u>70.4±3.7</u>	<u>71.2±2.6</u>	<u>71.5±3.8</u>	<u>72.0±2.9</u>	<u>82.0±2.5</u>
<i>hiv</i>	53.9±1.1	<b>74.8±3.3</b>	<b>64.1±2.1</b>	<b>70.5±3.1</b>	<b>66.8±3.3</b>	<b>72.4±3.3</b>	<b>67.0±3.2</b>
<i>mutagenesis</i>	71.7±4.9	<b>77.0±5.0</b>	71.6±7.0	74.5±4.4	72.8±8.2	<b>75.8±6.6</b>	71.9±6.0
<i>trains</i>	89.2±0.6	<u>86.3±0.5</u>	89.0±0.6	89.2±0.5	89.2±0.6	89.4±0.5	89.3±0.5
		wins/ties/losses (all)	w/t/l (two-class)	w/t/l (multi-class)			
<b>C4.4 vs C4.5</b>		4/5/3	3/2/2	1/3/1			
<b>MDLs vs C4.5</b>		2/3/7	2/3/2	0/0/5			
<b>MDLp vs C4.5</b>		2/3/7	2/3/2	0/0/5			
<b>BICs vs C4.5</b>		3/3/6	3/3/1	0/0/5			
<b>BICp vs C4.5</b>		3/4/5	3/4/0	0/0/5			
<b>Chi vs C4.5</b>		3/4/5	3/2/2	0/2/3			

Table 6: Experimental results: tree size (using **C4.5** as a reference).

	<b>C4.5</b>	<b>C4.4</b>	<b>MDLs</b>	<b>MDLp</b>	<b>BICs</b>	<b>BICp</b>	<b>Chi</b>
<i>asm</i>	64±11	<u>352±14</u>	<b>3±0</b>	<b>3±0</b>	<b>6±2</b>	<b>7±3</b>	<b>3±0</b>
<i>audiology</i>	24±2	24±1	<b>2±0</b>	<b>2±0</b>	<b>3±1</b>	<b>3±1</b>	<b>18±3</b>
<i>pen digits</i>	214±7	<u>254±8</u>	<b>36±2</b>	<b>36±2</b>	<b>42±3</b>	<b>42±3</b>	<b>126±4</b>
<i>primary tumor</i>	79±7	<u>129±5</u>	<b>2±0</b>	<b>2±0</b>	<b>2±0</b>	<b>2±0</b>	<b>5±1</b>
<i>voting</i>	6±3	<u>21±3</u>	<b>3±1</b>	<b>3±1</b>	6±1	6±1	6±1
<i>yeast</i>	123±13	<u>447±34</u>	<b>6±0</b>	<b>6±0</b>	<b>6±0</b>	<b>6±0</b>	<b>23±3</b>
<i>biodegradability</i>	30±5	<u>72±5</u>	<b>2±1</b>	<b>2±1</b>	<b>10±2</b>	<b>12±4</b>	<b>4±2</b>
<i>carcinogenesis</i>	35±7	<u>95±8</u>	<b>1±0</b>	<b>1±0</b>	<b>5±2</b>	<b>8±3</b>	<b>1±0</b>
<i>diterpenes</i>	64±5	<u>127±11</u>	<b>3±1</b>	<b>4±1</b>	<b>4±1</b>	<b>4±1</b>	<b>14±2</b>
<i>hiv</i>	15±2	<u>1391±76</u>	<b>8±3</b>	<u>32±4</u>	<u>26±3</u>	<u>55±4</u>	<u>33±3</u>
<i>mutagenesis</i>	9±5	<u>42±4</u>	<b>2±0</b>	<b>2±0</b>	<b>5±2</b>	<b>5±2</b>	<b>2±0</b>
<i>trains</i>	484±29	<u>4664±55</u>	<b>38±2</b>	<b>49±4</b>	<b>68±6</b>	<b>92±6</b>	<b>64±3</b>
		wins/ties/losses (all)	w/t/l (two-class)	w/t/l (multi-class)			
<b>C4.4 vs C4.5</b>		0/1/11	0/0/7	0/1/4			
<b>MDLs vs C4.5</b>		12/0/0	7/0/0	5/0/0			
<b>MDLp vs C4.5</b>		11/0/1	6/0/1	5/0/0			
<b>BICs vs C4.5</b>		10/1/1	5/1/1	5/0/0			
<b>BICp vs C4.5</b>		10/1/1	5/1/1	5/0/0			
<b>Chi vs C4.5</b>		10/1/1	5/1/1	5/0/0			

Table 7: Time in seconds for building a single tree.

	<b>C4.5</b>	<b>C4.4</b>	<b>MDLs</b>	<b>MDLp</b>	<b>BICs</b>	<b>BICp</b>	<b>Chi</b>
<i>asm</i>	88.3	88.0	2.2	35.0	2.2	59.0	2.2
<i>audiology</i>	0.5	0.6	0.1	0.2	0.1	0.2	0.9
<i>pen digits</i>	23.4	23.3	7.8	10.3	8.4	10.6	18.9
<i>primary tumor</i>	6.4	6.3	0.1	0.3	0.1	0.2	0.1
<i>voting</i>	0.2	0.2	0.1	0.1	0.1	0.1	0.1
<i>yeast</i>	33.6	33.4	0.4	2.3	0.4	2.6	0.9
<i>biodegradability</i>	3.6	3.5	0.1	1.4	0.5	2.8	0.3
<i>carcinogenesis</i>	32.6	32.5	0.6	19.3	2.8	30.2	0.6
<i>diterpenes</i>	64.3	64.2	7.4	38.7	7.3	39.2	13.2
<i>hiv</i>	461.5	458.8	7.3	325.6	30.6	392.1	33.6
<i>mutagenesis</i>	9.8	9.8	1.9	7.7	3.1	9.9	1.9
<i>trains</i>	2747.4	2741.0	38.4	475.5	44.1	993.8	42.6
Sum	3471.6	3461.6	66.4	916.4	99.7	1540.7	115.3

running times for **MDLp** and **BICp** are normally strictly greater than running times for **C4.4** since the first-step (top-down induction of a tree) is identical, but **MDLp** and **BICp** need a second step (post-pruning) while **C4.4** doesn't. However, for **MDLp** and **BICp**, the algorithm for the first step can easily be optimized<sup>3</sup>. This explains why running times for **MDLp** and **BICp** are smaller than for **C4.4**.

## 3.2 Discussion and Further Experiments

### 3.2.1 Overall Observations

Overall **C4.4** performs best although there are some datasets where it is outperformed (most notably *asm*). **C4.5** performs almost as well (the number of wins/ties/losses of **C4.5** versus **C4.4** is 3/5/4). This confirms the conclusions of Provost and Domingos [13]. Trees for **C4.5** are almost always smaller than trees for **C4.4**. Note that the dramatic drop in performance of **C4.5** on *hiv* is probably due to the strongly skewed class-distribution (only 3.6% of positive examples). Further in this section we discuss a controlled experiment showing that **C4.5** indeed performs badly on strongly skewed datasets.

MDL overall performs clearly worse than **C4.4** or **C4.5** (e.g. wins/ties/losses for **MDLp** versus **C4.4** are 3/2/7, versus **C4.5** 2/3/7). **Chi** also performs worse than **C4.4** or **C4.5** but to a smaller extent than MDL (wins/ties/losses for **Chi** versus **C4.4** are 3/3/6, versus **C4.5** 3/4/5). Trees for MDL and **Chi** are always smaller than trees for **C4.4** and almost always smaller than for **C4.5**.

Results for BIC are discussed in the next section.

### 3.2.2 Influence of the Number of Classes

An interesting observation from Table 3 is that BIC performs well for two-class problems but not for the multi-class problems we considered (that all have  $NbClasses \geq 10$ ).

On the *two-class problems*, wins/ties/losses for **BICp** versus **C4.4** are 2/4/1, versus **C4.5** 3/4/0. Results for **BICs** are similar (resp. 2/3/2 and 3/3/1). This means that on two-class problems, BIC performs at least as well as **C4.4** and even performs better than **C4.5**. In addition, BIC has the advantage of building much smaller trees.

<sup>3</sup>Nodes are sometimes split into two child-nodes during the first step while we can derive that these child-nodes will be pruned away by the MDL- or BIC-criterion during the post-pruning anyway. By slightly modifying the algorithm of Figure 1, we can avoid this overhead in some cases.

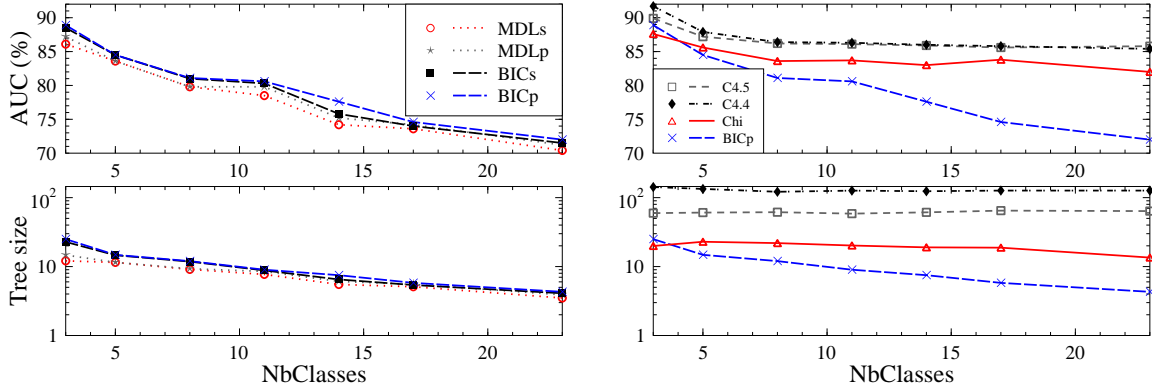


Figure 3: Influence of the number of classes for *diterpenes*.

On the *multi-class problems*, the picture looks rather different, however. Wins/ties/losses for **BICp** versus **C4.4** are 1/0/4, versus **C4.5** 0/0/5. Results for **BICs** are identical. This means that on multi-class problems, **C4.4** and **C4.5** clearly outperform BIC. One explanation for this is the fact that  $Thr$  for **BICs** is  $0.5 (NbClasses - 1) \log_2 N$ . So if  $NbClasses$  is high, then  $Thr$  is high as well and only tests  $T$  with a very high heuristic value  $h(T)$  (larger than  $Thr$ ) are accepted and hence very small trees are built. This suggests that **BICs** could be improved by making the dependency of  $Thr$  on  $NbClasses$  less strong (i.e. less than linear) since then trees built for a high  $NbClasses$  will be larger (although such a modification would deviate from the original theoretical foundations of BIC [15]). Similar remarks apply to **BICp**.

Note that the above observations for BIC (and their explanation) also hold for MDL but to a smaller extent, e.g. for **MDLp** wins/ties/losses on two-class problems are 2/2/3, on multi-class problems 1/0/4.

We performed an additional controlled experiment to investigate the influence of the number of classes. We started from *diterpenes*, a dataset having 23 classes on which **BICs** and **BICp** performed badly. In each step we merged the two least frequent classes, until only three classes were left. Figure 3 shows the results obtained from 10-fold cross-validation. In the top panels we show AUC, in the bottom panels tree size (on a logarithmic axis). We show **MDLs**, **MDLp**, **BICs** and **BICp** in the left panels and **BICp** (the best of the previous four), **C4.5**, **C4.4** and **Chi** in the right panels. We see that results for **MDLs**, **MDLp**, **BICs** and **BICp** are always very close to each other. For these approaches both tree size and AUC quickly decrease as the number of classes increases. Interestingly, there is almost no such decrease for **C4.5**, **C4.4** and **Chi**. Figure 3 shows that as a consequence **MDLs**, **MDLp**, **BICs** and **BICp** can compete with the other approaches when  $NbClasses$  is 3 or 5, but are outperformed when  $NbClasses$  goes higher. This confirms the above observation that MDL and BIC do not work well for multi-class problems, and its explanation.

### 3.2.3 Influence of the Number of Examples

We also investigated the influence of the number of examples in the dataset. We learned trees from subsets of *hiv* containing a variable number of examples (we evaluate them on a separate test-set of 6768 examples). Figure 4 shows the results. We see that results (both AUC and tree size) for **MDLs**, **MDLp**, **BICs** and **Chi** are very close to each other for all sizes of the dataset. For small datasets ( $N < 15000$ ) also **BICp** is very close to the previous four. Interestingly however, for larger datasets **BICp** learns larger trees than the others, resulting in higher AUC's. This suggests that for larger datasets BIC for post-pruning (**BICp**) is more useful as compared to BIC as a stopping-criterion (**BICs**).

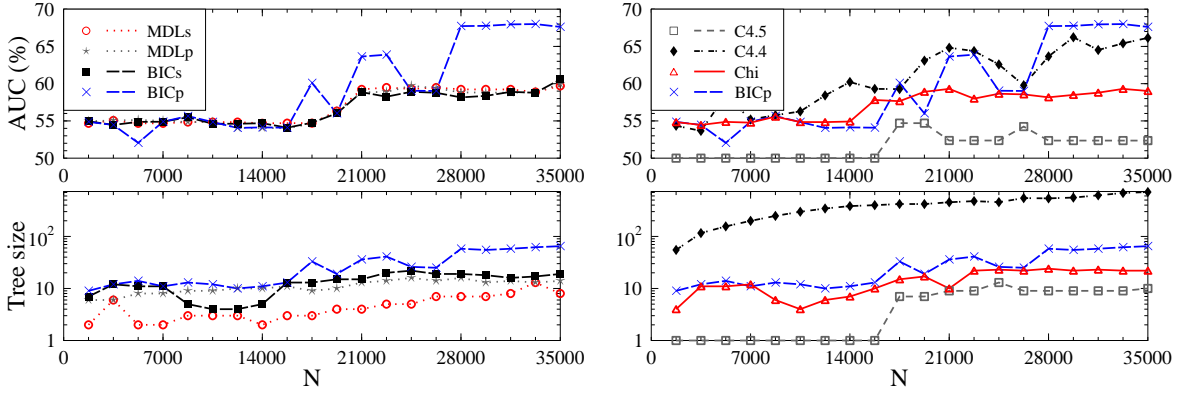


Figure 4: Influence of the number of examples  $N$  for *hiv*.

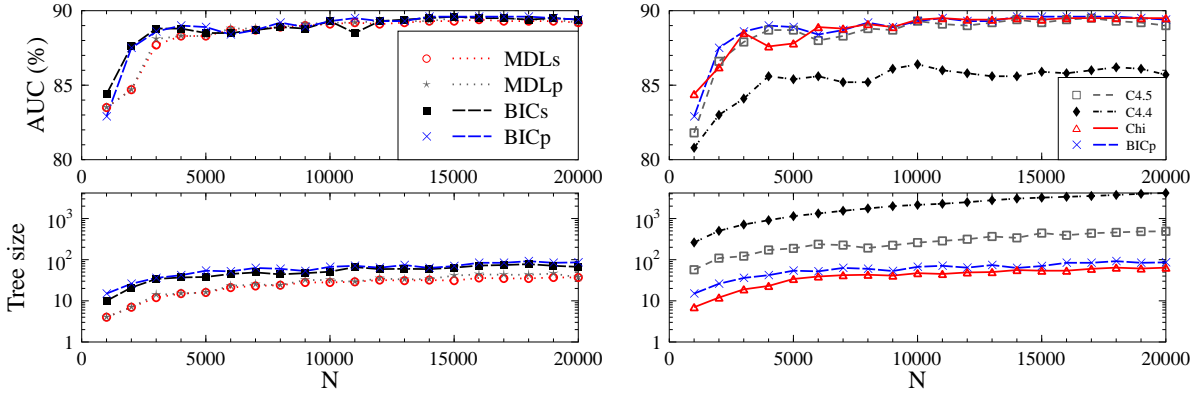


Figure 5: Influence of the number of examples  $N$  for *trains*.

We currently do not have an explanation for this.

We performed the same experiment for *trains* (using a test-set of 5000 examples), see Figure 5. Again results for **MDLs**, **MDLp**, **BICs** and **Chi** are close to each other for all sizes of the dataset (except the very low ones,  $N \leq 3000$ ). Unlike for *hiv*, however, **BICp** is very close to the previous four for all sizes of the dataset and does not become better than these four for larger datasets. Also we see that **C4.4** seems to overfit for all sizes of the dataset (it builds the largest trees but has the lowest AUC). The degree of overfitting is not heavily influenced by the size of the datasets. This is probably due to two competing effects [13]. On the one hand: if the dataset grows, trees grow as well (Figure 5), increasing the probability of overfitting. On the other hand, if the dataset grows, the number of examples in the leaves would increase, making probability estimates more reliable, thus decreasing the probability of overfitting. Why **C4.4** overfits on some datasets but not on others is currently an open question.

### 3.2.4 Influence of the Skewness

We also investigated the influence of the skewness of the class-distribution for two-class problems. This was motivated by the above observation that **C4.5** performed surprisingly bad on (subsets of variable size of) *hiv*, a strongly skewed dataset (see Table 3 and Figure 4). We investigated the issue on the *trains* problem by generating datasets of 3000 examples each, containing a variable percentage of positive examples. Figure 6 shows the results obtained from 10-fold cross-validation. We see that for all approaches AUC increases as the dataset goes from non-skewed (50%) to slightly skewed (70%). For all approaches except **C4.5** AUC stays nearly constant if the dataset becomes more

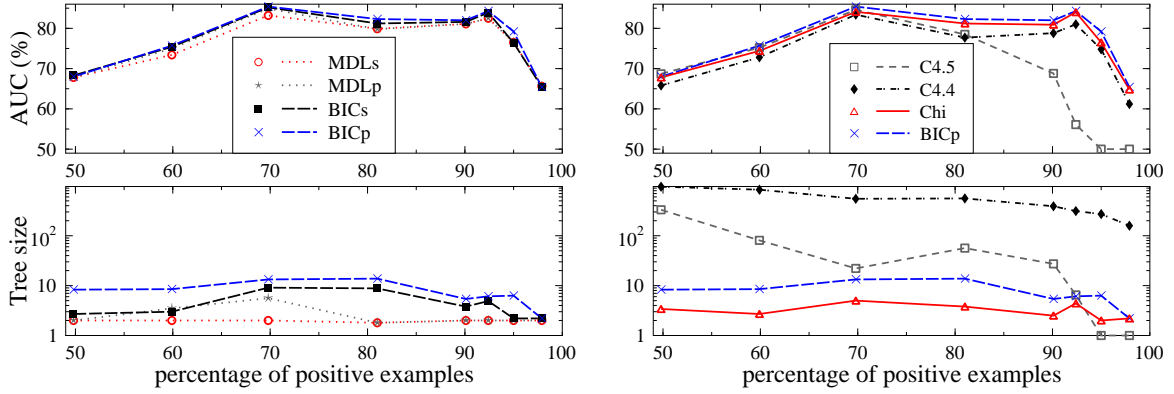


Figure 6: Influence of skewness of the class-distribution for *trains*.

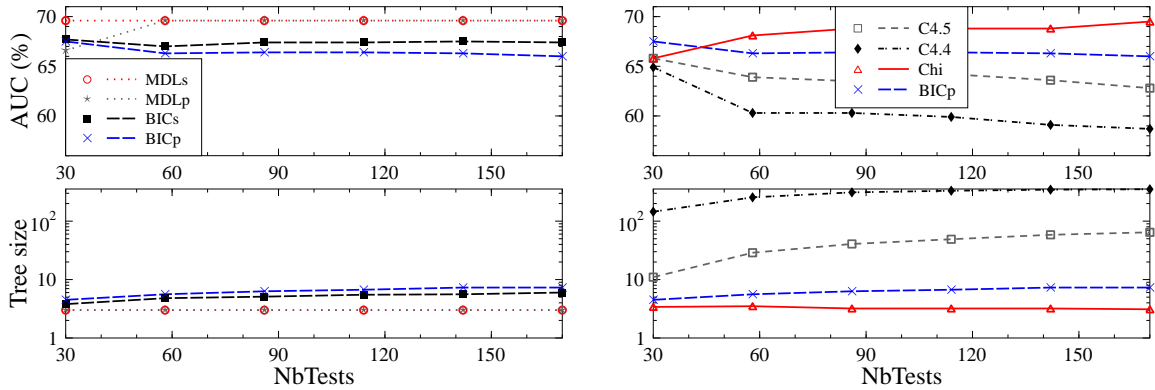


Figure 7: Influence of the number of candidate-tests for *asm*.

skewed, until it is extremely skewed and AUC's decrease heavily. For **C4.5**, however, the decrease in AUC begins earlier. Going from a skewness of 80% to 90%, AUC for **C4.5** already decreases by 10%. For very skewed datasets, **C4.5** builds trees consisting of a single leaf (performing very badly, of course). We conclude that from all approaches **C4.5** is the least robust to skewness.

### 3.2.5 Influence of the Number of Candidate-Tests

We also investigated the influence of the number of candidate-tests (remember that this parameter occurs in the definition of *Thr* for MDL and **Chi**). We started from *asm* (having 170 candidate-tests for the root) and in each step randomly left out some more candidate-tests from consideration. Figure 7 shows the results obtained from 10-fold cross-validation<sup>4</sup>. Unfortunately, it is difficult to draw clear conclusions from these results. The main observation is that for all approaches except **C4.5** and **C4.4** results are largely insensitive to the number of candidate-tests (except for very small numbers). For **C4.5** and **C4.4** tree size increases for an increasing number of candidate-tests, and AUC decreases slightly (presumably due to overfitting).

## 4 Conclusions

We reviewed the main existing approaches for learning probability trees and also considered a novel variant of an existing approach based on the Bayesian Information Criterion (BIC). We carried out a comparative study by performing a set of experiments

<sup>4</sup>The curves are nearly constant in the parts occluded by the legend.

on benchmark datasets. In order to clarify the most remarkable observations about these experiments, we also analyzed the influence of various parameters on predictive performance and tree size through a set of controlled experiments.

One conclusion is that overall the C4.4-approach performs best, and the C4.5-approach second best. When the class-distribution is strongly skewed, however, the C4.5-approach fails. An advantage of the C4.5-approach is that its trees are much smaller than for the C4.4-approach.

Another conclusion is that BIC performs at least as well as the C4.5- or C4.4-approaches if the number of classes is low. Also, trees for BIC are considerably smaller than trees for the C4.5- or C4.4-approaches. If the number of classes is too high ( $> 5$  in our experiments), however, BIC fails because trees are too small.

Altogether, these conclusions lead us to the follow recommendations for building probability trees. If the number of classes is high (e.g.  $> 5$ ) and the class-distribution is strongly skewed, use the C4.4-approach. If the number of classes is high but the class-distribution is not strongly skewed, use the C4.5-approach (trees perform as well as for the C4.4-approach but are smaller). If the number of classes is not high, use BIC (trees perform as well as for the C4.4- and C4.5-approaches but are smaller).

One idea for future research is to alter the BIC-approach by decreasing the influence of the number of classes on the stopping- or post-pruning-criterion. This would deviate from the original theoretical foundations of BIC, but might help in practice to make BIC better on datasets with a high number of classes.

Also it seems interesting to study the effect of bagging on the approaches considered in this paper. Provost and Domingos already noticed that bagging improves probability estimates of C4.4 and, to an even larger extent, C4.5 [13].

## Acknowledgements

Daan Fierens is supported by the Institute for the Promotion of Innovation through Science and Technology in Flanders (IWT Vlaanderen). Jan Ramon and Hendrik Blockeel are post-doctoral fellows of the Fund for Scientific Research (FWO) of Flanders.

## References

- [1] ILPnet2 Applications Descriptions. <http://www-ai.ijs.si/~ilpnet2/apps/>.
- [2] H. Blockeel and L. De Raedt. Top-down induction of first order logical decision trees. *Artificial Intelligence*, 101(1-2):285–297, June 1998.
- [3] D. Fierens, J. Ramon, H. Blockeel, and M. Bruynooghe. A comparison of approaches for learning first-order logical probability estimation trees. In *Inductive Logic Programming, 15th International Conference (ILP05), Late-breaking Papers*, 2005.
- [4] D. Fierens, J. Ramon, H. Blockeel, and M. Bruynooghe. A comparison of approaches for learning probability trees. In *Proceedings of the 16th European Conference on Machine Learning (ECML-05)*, 2005. To appear.
- [5] N. Friedman and M. Goldszmidt. Learning Bayesian networks with local structure. In *Proceedings of 12th Conference on Uncertainty in Artificial Intelligence (UAI-1996)*, pages 252–262, 1996.
- [6] N. Friedman and M. Goldszmidt. Learning Bayesian networks with local structure. In *Learning and Inference in Graphical Models*. Cambridge: MIT Press, 1998.

- [7] D. Heckerman, D. Chickering, C. Meek, R. Rounthwaite, and C. Kadie. Dependency Networks for Inference, Collaborative Filtering, and Data Visualization. *Journal of Machine Learning Research*, 1:49–75, 2000.
- [8] A. J. Knobbe. Data mining for adaptive system management. In *Proceedings of the first international conference and exhibition on the Practical Application of Knowledge Discovery and Data Mining (PADD-1997)*, 1997.
- [9] S. Kramer, L. De Raedt, and C. Helma. Molecular feature mining in HIV data. In *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-2001)*, pages 136–143, 2001.
- [10] C. Merz and P. Murphy. UCI repository of machine learning databases <http://www.ics.uci.edu/~mllearn/mlrepository.html>, 1996. Irvine, CA: University of California, Department of Information and Computer Science.
- [11] D. Michie, S. Muggleton, D. Page, and A. Srinivasan. To the international computing community: A new east-west challenge. Technical report, Oxford University Computing Laboratory, Oxford, UK, 1994. Available at <ftp.comlab.ox.ac.uk>.
- [12] J. Neville, D. Jensen, L. Friedland, and M. Hay. Learning relational probability trees. In *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-2003)*, 2003.
- [13] F. Provost and P. Domingos. Tree induction for probability-based ranking. *Machine Learning*, 52:199–216, 2003.
- [14] J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann series in Machine Learning. Morgan Kaufmann, 1993.
- [15] G. Schwarz. Estimating the dimension of a model. *Annals of Statistics*, 6:461–464, 1978.
- [16] A. Srinivasan, R. King, and D. Bristol. An assessment of ILP-assisted models for toxicology and the PTE-3 experiment. In *Proceedings of the seventh international conference on Inductive Logic Programming (ILP-1999)*, 1999.