FINDING THE TREE IN THE FOREST

Rikard König*, Ulf Johansson* and Lars Niklasson**
*School of Business and Informatics, University of Borås, Sweden
**Informatics Research Centre, University of Skövde, Sweden

ABSTRACT

Decision trees are often used for decision support since they are fast to train, easy to understand and deterministic; i.e., always create identical trees from the same training data. This property is, however, only inherent in the actual decision tree algorithm, nondeterministic techniques such as genetic programming could very well produce different trees with similar accuracy and complexity for each execution. Clearly, if more than one solution exists, it would be misleading to present a single tree to a decision maker. On the other hand, too many alternatives could not be handled manually, and would only lead to confusion. Hence, we argue for a method aimed at generating a suitable number of alternative decision trees with comparable accuracy and complexity. When too many alternative trees exist, they are grouped and representative accurate solutions are selected from each group. Using domain knowledge, a decision maker could then select a single best tree and, if required, be presented with a small set of similar solutions, in order to further improve his decisions. In this paper, a method for generating alternative decision trees is suggested and evaluated. All in all,four different techniques for selecting accurate representative trees from groups of similar solutions are presented. Experiments on 19 UCI data sets show that it often exist dozens of alternative trees, and that one of the evaluated techniques clearly outperforms all others for selecting accurate and representative models.

KEYWORDS

Decision support, decision trees, genetic programming, alternative solutions, inconsistency.

1. INTRODUCTION

Decision support systems based on predictive modeling are today a crucial part of many organizations since data often is collected in amounts and with a complexity that exceed the capabilities of human decision makers. Even if accuracy normally is the main goal for predictive modeling Goodwin (2002) states that most decision makers would require at least a basic understanding of a predictive model to use it for decision support. Furthermore, Domingos (1977) points out that comprehensibility is important since it facilitates the process of interactive refinement that is at the heart of most successful applications.

One of the most popular techniques creating comprehensible models is decision trees algorithms such as C4.5 (Quinlan 1986). Decision trees are popular since they are fast to train and easy to understand. However, the trees still need to have a reasonable size to be considered comprehensible. Curruble et al. (1995) suggest that it becomes nearly impossible to get a global idea of a model if it consists of more than one or two dozens of rules.

A well known deficiency present in most decision trees algorithms is that they are unstable; i.e., slight variations in the training data can result in quite different attribute selections in the splits, see e.g., (Roiger & Geatz 2003). The problem then, of course, becomes which decision tree that should be trusted if small variations in the data will produce very different trees. Turney (1995) gives an example where engineers are disturbed and lose confidence in the decision trees when different batches of data from the same process result in radically different trees.

According to Dietterich (1996), the most fundamental source of instability is that the hypothesis space is too large. If an algorithm searches a very large hypothesis space and outputs a single hypothesis, then in the absence of huge amounts of training data, the algorithm will need to make many more or less arbitrary decisions, decisions which might be different if the training set were only slightly modified. This is called *informational instability*, instability caused by the lack of information.

This informational instability is for example often experienced in the medical domain, where datasets often contain small number of instances (sometimes 100 or less) but still relatively large number of features. In such cases (large spaces with sparse samples) it is quite likely that different logical expressions may accidently classify some data well, thus many data mining systems may find solutions which are precise but not meaningful according to experts; see e.g., (Grąbczewski & Duch 2002).

One approach to handle this problem is to instead benefit from the instability by creating a diverse ensemble of models using, for instance *bagging* (Breiman 1996) or *boosting* (Schapire 1990). However, even if these types of ensemble techniques most often are more accurate and stable in their predictions, they are not comprehensible since a large number of models would need to be interpreted to understand a prediction.

Another approach is taken by Grąbczewski & Duch (2002), who point out that many sets of rules with similar complexity and accuracy may exist, using for example different feature subsets, bringing more information of interest to the domain expert. Providing experts with several alternative descriptions make it easier for the experts to find interesting explanations compatible with their experience, and may lead to a better understanding of the problem. Consequently, data mining methods aimed at finding several different descriptions of the same relationship, are potentially valuable, and deserve investigation.

The approach of providing experts with alternative solutions is also supported by, for instance, Plish (1998), who notes that modern support systems for group decisions in situational centers largely depends on the availability of a procedure that generates "reasonable" (nearly optimal) alternative decisions in real time. If more than one solution exists, it would actually be misleading to present a single solution to a decision maker.

Following the argumentation above, this study will present a method for generating alternative solutions which all have comparable accuracy and complexity. Since too many alternatives risk to confuse a decision maker, the method also guides an expert among a large set of alternative solutions. In more detail, the suggested method first groups similar solutions and then selects accurate representative solutions from each group. Using domain knowledge, a decision maker could then select the best tree from only a few representative solutions. Finally, the decision maker could also request a small set of similar solutions to further improve his understanding of the relationship.

A solutions in the context of decision making could be any form, but this study only consider decision trees since they are one of the most popular machine learning techniques used for decision support. Solution and decision tree are used interchangeably in the rest of this paper.

2. RELATED WORK

The following section will first describe some techniques for creating alternative solutions. In the last section different approaches for selecting a single model from several models will be discussed.

2.1 Generating Alternative Solutions

A straightforward and frequently used way of generating different solutions for a certain data set is to train different models on different parts of the training data. Since most machine learning techniques are instable, this will result in different solutions. Bagging is a well know example of this technique. In bagging, a new *bootstrap* training set is created for each model by randomly selecting instances (with replacement) from the original training set.

As mentioned above Pilsh (1998) acknowledge the importance of supplying alternative solutions to a decision maker. Pilsh also suggest an algorithm for generating alternative solutions for a multi-criterion linear programming model. Alternative solutions are generated by slightly altering either the objective or the constraints, and then solving the resulting problem with the help of supplementary constraints. Structural changes are also considered, but the algorithm is limited to multi-criterion linear problems, and cannot be applied to decision trees.

An algorithm for generating alternative decision trees is presented by Grąbczewski & Duch (2002), who use a variant of standard beam search to create heterogeneous forests of decision trees. The number of possible alternative solutions is restricted to the beam size. To create a forest, all trees that are found during the search are ordered according to their accuracy, estimated on validation set. An infinite beam size corresponds to a breadth first search, which is unpractical for most problems. Grąbczewski & Duch report that their algorithm finds several good alternative solutions for three UCI (Blake & Mertz 1998) data sets. However, it is somewhat unclear exactly how the solutions were evaluated, and the number of alternative solutions that were found is not reported.

Li & Lui (2003) present a simple method for creating ensembles called *cascading trees*. First all features are ranked according to their gain ratio. Next trees are created in a cascading manner where the root node of the i^{th} tree corresponds to the i^{th} ranked features; the rest of the tree is created as normal. The authors stress the fact that unlike bagging or boosting, cascading trees do not in any way modify the original data. This is of a critical concern in, for instance, bio-medical applications such as the understanding and diagnosis of a disease, where it is important that all training instances are classified correctly. The method is evaluated on two medical related dataset using 10-fold cross-validation. Several trees that could classify the training data without error were found for all folds. An interesting observation, made by the authors, is that the tree with the best test accuracy often did not have the feature with the highest gain ratio in its root.

It should be noted that ensemble creation techniques often create base classifiers by sacrificing accuracy for diversity. Ensemble members are usually less accurate than single model while the ensemble is more accurate. Hence, most ensemble methods are not suitable for creating several alternative standalone solutions.

In a previous study (Johansson et. al. 2010) we used GP to generate alternative models based on all data. Since GP is inherently inconsistent, no data needs to be scarified or modified to achieve alternative solutions. Several alternative trees where found and most trees were more accurate than a single decision tree created using CART (Breiman 1984).

2.2 Selecting Solutions

The most straightforward approach to selecting a single model from several alternative models, is of course, to compare all models and pick the *n* having the highest accuracy on either training data or on an additional (validation) data set. In this study however the starting point is models which all have comparable training accuracy which could complicate selection based on training accuracy.

Holding out a validation set is still applicable but previous work such as (Johansson et. al. 2010), has shown that even if a validation set is useful for selecting models, it also lowers the accuracy for the generated model since all data is not available for training. Again, the use of all available data for the actual modeling is especially important for data sets with relatively few instances to start with. Hence, selection based on validation accuracy will not be considered in this study.

A different approach is to select the *n* trees with the highest gain ratio, but as seen in the work of Li & Liu (2003) this does not yield very promising result.

In our previous study (Johansson et al. 2010), one tree was selected from a group of trees based on ensemble fidelity. The method used the fact that an ensemble of models most often is better than its individual members. First alternative trees were created from all training data using GP and all available training data. Next an imaginary ensemble was created from the evaluated models and used to generate predictions for both training and test instances. Finally, the predictions of each model were compared to the ensemble prediction and the model that was most faithful (in making the same predictions) was selected.

In the field of semi-supervised learning, this is referred to as *coaching*. Ensemble predictions could be produced even for the test instances, as long as the problem is one where predictions are made for sets of instances, rather than one instance at a time. Fortunately, in most real-world data mining projects, bulk predictions are made, and there is no shortage of unlabeled instances. Experiments using tenfold cross validation on 25 UCI data sets clearly showed that it is better to select models based on the fidelity against the imaginary ensemble than to use training or validation accuracy.

3. METHOD

In this study we argue that alternative solutions may enrich a decision making situation. However, in the same way that complex decision trees become hard to comprehend, a large number of alternative solutions will also reduce the benefit of alternative solutions. Simply put, decision makers cannot interpret and evaluate dozens of trees, so there is a need for automatic strategies for selecting a subset of accurate trees.

We define an *alternative solution* as a decision tree that is of the same size or smaller than the original solution, while having equal or better training accuracy. Furthermore, an alternative decision tree should classify the data with a unique partitioning of the training instances, i.e., to be an alternative the solution needs to base its decision on different facts. It should be noted that two decision trees can classify the data in exactly the same way, but still partitioning the instances differently.

In this paper the *original solution* is represented by a decision tree created using the J48 algorithm in the WEKA (Witten & Frank 2005) workbench.

Naturally, the numbers of alternative solutions are dependent on the *size* of the tree, i.e. the number of *splits*, in the original tree and the number of possible attribute values in the data set. A *split* is a combination of an *attribute*, an *operator* and a *value* which creates a partition of the data set. In the experiments, (and in most decision tree algorithms), only *relevant splits* are considered, since the aim is to present truly alternative solutions. *Relevant* splits for an attribute is the splits that are needed to divide the data set into *pure* and *unpure* partitions. A *pure* partition is a set of instances of the same target class.

For an original tree with n splits and a dataset with r relevant splits there are n^r possible solutions. Of these solutions, only the ones with equal or higher accuracy are considered to be alternative solutions. Furthermore, if several alternative trees partition the data set in the same way, only the smallest tree is considered to be an alternative solution. Since the number of possible solutions is related to the size of the original solution, trees of the same size must be evaluated for all data sets.

3.1 Creation of Original Trees

Since the pruning in J48 does not support creation of trees of a certain size, the algorithm cannot be used in its original form. Instead a very large J48 tree is first created by setting the confidence factor to 0.5, (higher values yields warnings in WEKA). Next, the tree is pruned to a certain size in the following manner:

1. CUT_DEPTH is set to MAX_SIZE
2. All branches are cut at CUT_DEPTH and replaced with leaf node predicting the majority class of the training instance reaching the new leaf.
3. Redundant leaves are removed, i.e. if both leaves of any root split are predicting the same class the split is replaced by one of the leaves. This is done recursively since replacing a split can result in new redundancy higher up in the tree.
4. If the tree SIZE > MAX_ SIZE then CUT_DEPTH is set to = CUT_DEPTH-1. Return to 2.

Even if this pruning algorithm does not guarantee an exact size of the final tree, it is much more consistent than J48 original pruning algorithm.

3.2 Creation of Alternative Solutions

To be able to generate alternative solutions, a method needs to guarantee that the solutions have a certain training accuracy, a certain size and partition the data set in a unique way. None of the techniques discussed in the related work fulfill all three requirements.

Our approach is based on GP and continuously evolves a population of decision trees. If any of the trees in the population meets the requirement for being an alternative tree (accuracy, size and uniqueness) they are put in a growing list of alternative solutions. If two trees partition the dataset in the same way, only the smallest tree is kept in the list. To always drive the evolution towards new alternative solutions, a fitness function based on three metrics is used:

- A reward based on accuracy
- A punishment in relation to the tree size which increases if a tree is bigger than the original tree
- A punishment in relation to how similar the tree is to other alternative solutions.

Similarity is calculated by counting the number of identical splits that occurs in the same position in both trees. To ensure that each tree makes a unique partition of the data set, the GP is only allowed to search among relevant splits. If all splits are relevant, and the tree is not a copy of another tree, it will partition the data in a unique way.

The GP used in the experiments are more or less vanilla GP using tournament selection. A difference is that only two trees are selected for each tournament to slow down the convergence of the population. The idea is to look for more alternative trees in the neighborhood of discovered solutions. Another important difference from standard GP is that five batches are used in the experiments. A batch always starts from a new randomly generated population, but the list of alternative trees is kept during all batches. In this way, each population can start to look for solutions in new directions, even if a previous batch has converged to a certain solution.

Finally the trees are grouped based on their root split, putting all trees that start with the same split in the same group.

3.3 Selection of Representative Trees

As described in the related work, there are several ways of selecting a single tree from a group of trees. This study will evaluate four different techniques *random*, *training accuracy*, *ensemble fidelity* and *similarity*. *Random* selects a tree randomly and will be estimated by calculating the average accuracy of all trees. The other three techniques select one tree from each group of solutions. *Training accuracy* selects the tree with the highest training accuracy from each group. *Ensemble fidelity* is based on (Johansson et al. 2010) coaching technique and uses all solutions as an ensemble. The difference is that instead of selecting only one tree, the tree that is most faithful to the ensemble's test prediction is selected from each group. Finally, s*imilarity* selects the trees that have most in common with the other trees in the same group. The idea is that important splits will be used more often and hence the tree that has most in common with the other group members should contain more important splits.

4. EXPERIMENTS

The experiments are divided in three main stages, i.e., generation of alternative solutions, selection of representative trees and tree evaluation. All selection strategies were evaluated on the same groups of alternative solutions. It should be noted that only groups with three or more members were counted and evaluated in the experiments since *simlilarity* has no meaning for two trees. All experiments were performed on 19 data sets from the UCI–machine learning repository using 10-fold cross-validation with stratification.

In the experiments, J48 trees were created with 3, 5 and 7 splits. The values 3 and 7 were selected since they correspond to balanced trees. Trees of these sizes may seem small and simple but as pointed out by (Holte 1993), simple classification rules perform well for most common problems. Hence, small trees are a good starting point for a decision support system.

The GP process was implemented in G-REX, our publicly available GP-framework (König et al. 2008). In the experiments, a batch consisted of a population of 300 individual that were evolved during 100 generations with a crossover probability of 0.8 and a mutation probability of 0.001.

5. RESULTS

The result of the first experiments which concerns creation of alternative solutions is presented in Table 1 below. *#Inst* is the number of instances in the data set and *#Splits* is the number of relevant splits that each data set contains. *Size* is the actual size of the J48 tree after the second phase of pruning has been performed. The average number of alternative *trees* and *groups* (for ten folds) are presented for each of the evaluated tree sizes 3,5,7 splits (*3S, 5S, 7S*).

As seen in the table, a large amount of alternative solutions could be found for all target tree sizes. As could be expected, a larger tree size results in more alternative solutions. Ten or less alternative solutions could only be found for some data sets, i.e. seven for *3S*, four for *5S* and one for *7S*. The average number of solutions (81.5, 103.5 and 173.9) is obviously too large for a decision maker to handle manually.

Another interesting result is that when larger trees are allowed, fewer groups of similar trees are found, in spite of an increasing number of solutions.

Table 1. Number of Solutions, Groups and average ACC

Data set	**#Inst.**	**#Splits**	**Size**			**#Trees**			**#Groups**		
			3S	**5S**	**7S**	**3S**	**5S**	**7S**	**3S**	**5S**	**7S**
Breast-cancer	286	32	2.6	4.2	6.1	64.3	90.3	159.9	6.7	5.4	3.6
Breast-w	699	74	2.0	5.0	5.3	113.0	325.4	302.4	12.2	13.8	12.9
Colic	368	311	2.0	3.5	5.7	4.6	24.7	95.7	0.0	0.8	2.5
C.-Lenses	24	4	2.9	2.9	2.9	8.2	7.4	7.4	1.6	1.4	1.3
Credit-a	690	860	2.5	4.0	6.0	26.0	106.8	237.2	1.9	3.4	2.7
Cylinderbands	540	1211	2.8	4.4	6.9	218.4	249.2	420.5	13.7	9.3	6.3
Diabetes	768	919	2.5	3.0	4.4	5.2	29.0	13.4	0.5	2.1	0.6
Glass	214	739	2.8	4.8	5.6	427.4	15.5	22.9	28.9	1.7	1.2
Haberman	306	80	2.4	3.9	6.4	142.5	324.0	551.9	12.1	17.2	20.1
Heart-c	303	316	2.6	4.5	5.5	76.9	17.3	34.1	3.5	2.0	2.1
Heart-Statlog	270	300	1.8	4.5	5.4	9.7	43.1	30.5	1.0	1.2	1.4
Hepatitis	155	201	1.9	3.6	5.3	19.0	145.9	280.2	1.6	6.1	6.5
Iris	150	59	2.9	3.8	3.8	28.5	21.8	23.8	4.5	2.7	3.2
Liver-disorder	345	274	2.1	4.6	5.6	66.7	106.2	88.1	5.8	7.6	3.8
Lymph	148	40	2.2	4.5	5.4	155.8	292.2	284.9	19.3	8.4	6.4
TAE	151	87	2.7	2.9	6.3	176.4	164.1	648.9	17.1	15.7	12.1
Tic-tac-toe	958	18	1.2	1.2	6.0	1.1	1.1	72.7	0.0	0.0	2.5
Wine	178	772	2.9	4.2	4.5	4.2	1.4	12.7	0.4	0.1	0.7
ZOO	101	96	3.0	5.0	6.8	2.7	1.9	16.4	0.1	0.2	0.7
MEAN	**350**	**336**	**2.4**	**3.9**	**5.5**	**81.6**	**103.5**	**173.9**	**6.9**	**5.2**	**4.8**

It should be noted that an alternative solutions is dependent on both the J48 size and accuracy. Hence, Table 2 below presents average test accuracy (*acc*) for each data set, tree *size*, number of alternative *trees* and *groups* with more than three trees. As could be expected, a larger J48 tree is more accurate than a smaller tree. It is, however, surprising that even if the accuracy increases, the possible number of solutions also increases. Of course a larger tree facilitates more combinations of the splits, but the search for an accurate tree also becomes harder since there are more trees to search among and less trees that actually have the required accuracy. Table 2 also shows that a possible explanation to the decreasing number of groups is increasing tree accuracy.

Table 2. Size and acc vs. #trees and #groups

	Size	J48 acc	Rnd acc	#trees	#Groups
3S	2.4	73.5	75.1	81.6	6.9
5S	3.9	76.5	77.5	103.5	5.2
7S	5.5	76.6	78.2	173.9	4.8

The test accuracies for all selection techniques are presented in Table 3 where *Rnd* is the average accuracy of all alternative solutions, which represents selecting a tree at random. *Trn*, *Ens* and *Str* are the average accuracies for selecting a single solution from each group of solutions. *Trn* selects a solutions from the group based on training accuracy, *Ens* uses ensemble fidelity as described above and *Sim* selects solutions that is most similar (in terms of splits) to the other trees in the group. For each data set and tree size the best result is marked with bold numbers.

Table 3. Accuracy

	3S					**5S**					**7S**				
Data set	**J48**	**Rnd**	**Trn**	**Ens**	**Sim**	**J48**	**Rnd**	**Trn**	**Ens**	**Sim**	**J48**	**Rnd**	**Trn**	**Ens**	**Sim**
Breast-cancer	70.7	72.4	**73.0**	**73.0**	72.7	71.0	72.6	73.0	**73.3**	73.2	70.7	72.1	71.7	**73.5**	71.9
Breast-w	92.9	93.7	94.6	**95.0**	94.3	94.4	95.0	95.3	**95.4**	95.1	94.6	95.1	**95.4**	**95.4**	**95.4**
Colic	83.7	85.2	**85.9**	**85.9**	83.7	84.5	85.1	84.9	**85.3**	85.1	**85.1**	85.0	84.4	85.0	**85.1**
C.-Lenses	75.0	77.6	75.0	**78.3**	75.0	75.0	78.0	75.0	**81.7**	75.0	75.0	77.9	75.0	**78.3**	**78.3**
Credit-a	**85.7**	85.2	84.4	85.0	85.0	84.1	**84.8**	84.4	84.7	84.3	84.3	**84.8**	**84.8**	84.6	84.6
Cylinderbands	66.3	67.4	68.0	**68.5**	68.1	68.0	69.2	69.5	**69.8**	69.0	**70.9**	68.9	68.4	70.0	69.0
Diabetes	74.3	**74.4**	74.0	**74.4**	74.3	74.5	74.9	**75.1**	74.9	74.9	**74.6**	74.4	74.3	74.4	74.1
Glass	47.2	53.4	**56.5**	55.9	53.3	64.0	65.0	**65.7**	**65.7**	**65.7**	64.5	64.1	63.9	**64.8**	63.4
Haberman	68.3	72.4	72.2	**72.5**	72.1	69.0	72.1	72.9	**73.1**	72.2	67.7	72.5	72.2	72.7	**72.3**
Heart-c	75.2	74.4	74.0	**75.3**	74.9	78.2	79.9	79.9	79.7	**80.0**	77.2	80.8	81.0	**81.3**	80.7
Heart-Statlog	**72.2**	70.5	70.8	71.4	71.3	76.3	78.5	**79.6**	78.8	78.8	77.4	80.4	79.7	**81.9**	79.8
Hepatitis	**80.1**	78.8	78.3	78.2	77.7	78.8	81.3	**82.2**	81.9	81.6	76.2	77.9	78.2	**78.5**	**78.5**
Iris	94.7	95.0	94.9	**95.2**	94.8	94.0	95.2	**95.3**	95.0	95.2	94.0	95.4	94.7	94.8	**94.9**
Liver-disorder	64.4	66.5	66.5	**67.2**	66.2	65.0	64.4	**65.2**	64.7	65.0	**65.5**	64.5	63.7	64.5	63.6
Lymph	59.5	69.0	72.3	**72.4**	67.5	**78.3**	75.6	75.5	77.3	75.1	75.0	76.5	74.9	**76.8**	75.5
TAE	45.8	49.1	51.6	**54.0**	48.8	46.5	49.6	51.1	**53.0**	49.6	48.5	54.7	54.5	**55.7**	54.4
Tic-tac-toe	**68.8**	**68.8**	**68.8**	**68.8**	**68.8**	**68.8**	**68.8**	**68.8**	**68.8**	**68.8**	71.6	75.3	77.4	**77.7**	76.0
Wine	88.0	90.8	90.7	**91.4**	90.5	91.6	**92.9**	**92.2**	**92.2**	**92.2**	92.0	93.3	92.2	**94.6**	92.2
ZOO	83.3	81.5	**83.3**	**83.3**	**83.3**	**91.2**	90.2	90.2	90.2	90.2	91.1	93.1	94.1	**94.1**	94.1
Mean	**73.5**	**75.1**	**75.5**	**76.1**	**74.9**	**76.5**	**77.5**	**77.7**	**78.2**	**77.4**	**76.6**	**78.2**	**77.9**	**78.9**	**78.1**

For each experiment *Ens* achieves the highest overall accuracy, while the original J48 trees attain the lowest. It is also relevant to note that the alternative solutions generated with GP (Rnd) have a higher overall accuracy than J48. Since the other techniques select a subset of these solutions, they should also be better than *J48*.

Ens is clearly the best techniques with the highest overall accuracy for all experiments, and with the highest accuracy on 15 data sets for *3S*, 10 for *5S* and 11 for *7S.* A pairwise sign test at 0.05 significance level (presented in Table 4) shows that *Ens* is significantly better than all other techniques for *3S* and *7S.* For *5S* the results are not significant, but *Ens* clearly outperforms the other techniques and only loses three times agains J48, five times against *All*, six times against *Trn* and three times against *Sim*.

Table 4. Pairwise sign test

α=0.05	J48	All	Trn	Sim
3S Ens	**0.0125**	**0.0013**	**0.0042**	**0.0003**
5S Ens	**0.0075**	0.0963	0.6072	**0.0352**
7S Ens	**0.0192**	**0.0044**	**0.0013**	**0.0127**

Table 5. Average likeness of *Ens* trees

	Real # Ifs	Similarity Grp	Similarity All
3S	2.4	1.04 / 43%	0.27 / 11%
5S	4.0	1.94 / 48%	0.75 / 19%
7S	5.5	2.56 / 46%	1.02 / 19%

Finally, the group likeness of the trees selected by *Ens* is presented in the Table 5 above. The idea is that the selected trees should be accurate and representative for the group. To be representative the minimum requirement is that the solutions is more similar to the trees in the group than to trees not in the group. Since the groups are created from trees with the same root split, the similarity is at least 1.0 for any tree selected from a group (*Grp)* of trees.

Clearly the trees selected by *Ens* are more similar to the trees in the corresponding group than to all trees. Furthermore, the group similarity increases with size of the trees. On average the trees selected by *Ens* shares 46% of the splits with any member which should be compared to 16% shared with all alternative trees.

6. CONCLUSION

In this paper we argue for a method aimed at generating a suitable number of alternative decision trees with comparable accuracy and complexity. When too many alternative trees exist, they are grouped and representative accurate solutions are selected from each group. Using domain knowledge, a decision maker could then select a single best tree and, if required, be presented with a small set of similar solutions, in order to further improve his decisions. The experiments support the feasibility of the purposed method since they show that:

- it is often possible to create many alternative trees, which all have comparable training accuracy and complexity. In average 120 alternative trees could be created for each original tree, which of course are more than what could be handled manually. Larger trees increase the number of alternative solutions even if the larger trees attain a higher accuracy.
- ensemble fidelity can be used to select several accurate trees from groups of alternative trees. In the experiment, trees were group based on their root split and an ensemble was created for each group. The trees that were most faithful to each ensemble (in terms of predictions) clearly outperform the average tree. Furthermore, trees selected in this manner are significantly better than the original tree and are also superior to selecting trees based on their training accuracy.
- the selected trees can be considered to be representative for their group, since they are more similar to trees inside than outside their group.

REFERENCES

Blake, C. & Merz, C., 1998. UCI repository of machine learning databases.

Breiman, L., 1996. Bagging predictors. *Machine Learning*, 24(2), 123-140.

Breiman, L., 1984. *Classification and regression trees*, Chapman & Hall/CRC.

Corruble, V., Thiré, F. & Ganascia, J., 1995. Comprehensible exploratory induction with decision graphs. *In Workshop on Machine Learning and Comprehensbility* (IJCAI).

Dietterich, T., 1996. Editorial. *Machine Learning*, 2(24), 1-3.

Domingos, P., 1997. Knowledge Acquisition from Examples Via Multiple Models. *In International Conference on Machine Learning*. Citeseer, p. 98–106.

Goodwin, P., 2002. Integrating management judgment and statistical methods to improve short-term forecasts. *Omega*, 30(2), 127–135.

Grąbczewski, K. & Duch, W., 2002. Heterogeneous Forests of Decision Trees. *Artificial Neural Networks (ICANN).*

Holte, R., 1993. Very simple classification rules perform well on most commonly used datasets. *Machine learning*, 11(1), 63–90.

Johansson, U., König, R., Löfström, T., Niklasson, L., 2010. Using Imaginary Ensembles to Select GP Classifiers. *In European Conference on Genetic Programming*, pp. 278-288

Johansson, U., König, R. & Niklasson, L., 2007. Inconsistency - Friend or Foe. *In 2007 International Joint Conference on Neural Networks*, pp. 1383-1388.

König, R., Johansson, U. & Niklasson, L., 2008. G-REX: A Versatile Framework for Evolutionary Data Mining. *International Conference on Data Mining Workshops*, 2008. ICDMW'08. p. 971–974.

Li, J. & Liu, H., 2003. Ensembles of cascading trees. *In International Conference on Data Mining (ICDM)*. pp. 585-588.

Plish, V., 1998. Algorithms generating alternative solutions for a multicriterion linear programming model. Cybernetics *and Systems Analysis*, 34(2), 301–304.

Quinlan, J.R., 1996. Bagging, boosting, and C4. 5. *In Proceedings of the National Conference on Artificial Intelligence*. p. 725–730.

Quinlan, J.R., 1986. Induction of decision trees. *Machine Learning*, 1(1), 81-106.

Roiger, R. & Geatz, M., 2003. *Data Mining: A tutorial-based primer*.

Schapire, R.E., 1990. The strength of weak learnability. *Machine Learning*, 5(2), 197-227.

Turney, P., 1995. Technical note: Bias and the quantification of stability. *Machine Learning*, 20(1-2), 23-33.

Witten, I. & Frank, E., 2005. *Data Mining: Practical machine learning tools and techniques,* Morgan Kaufman.