How to Interpret Decision Trees?

Petra Perner

Institute of Computer Vision and Applied Computer Sciences, IBaI,
Postbox 30 11 14, 04251, Leipzig
pperner@ibai-institut.de
www.ibai-institut.de

Abstract. Data mining methods are widely used across many disciplines to identify patterns, rules or associations among huge volumes of data. While in the past mostly black box methods such as neural nets and support vector machines have been heavily used in technical domains, methods that have explanation capability are preferred in medical domains. Nowadays, data mining methods with explanation capability are also used for technical domains after more work on advantages and disadvantages of the methods has been done. Decision tree induction such as C4.5 is the most preferred method since it works well on average regardless of the data set being used. This method can easily learn a decision tree without heavy user interaction while in neural nets a lot of time is spent on training the net. Cross-validation methods can be applied to decision tree induction methods; these methods ensure that the calculated error rate comes close to the true error rate. The error rate and the particular goodness measures described in this paper are quantitative measures that provide help in understanding the quality of the model. The data collection problem with its noise problem has to be considered. Specialized accuracy measures and proper visualization methods help to understand this problem. Since decision tree induction is a supervised method, the associated data labels constitute another problem. Re-labeling should be considered after the model has been learnt. This paper also discusses how to fit the learnt model to the expert´s knowledge. The problem of comparing two decision trees in accordance with its explanation power is discussed. Finally, we summarize our methodology on interpretation of decision trees.

1 Introduction

Data mining methods are widely used across many disciplines to identify patterns, rules or associations among huge volumes of data. Different methods can be applied to accomplish this. While in the past mostly black box methods such as neural nets and support vector machines (SVM) have been heavily used in technical domains, methods that have explanation capability have been particularly used in medical domains since a physician likes to understand the outcome of a classifier and map it to his domain knowledge; otherwise, the level of acceptance of an automatic system is low. Nowadays, data mining methods with explanation capability are also used for technical domains after more work on advantages and disadvantages of the methods has been done.

P. Perner (Ed.): ICDM 2011, LNAI 6870, pp. 40–55, 2011.

The most preferred method among the methods with explanation capability is decision tree induction method [1]. This method can easily learn a decision tree without heavy user interaction while in neural nets a lot of time is spent on training the net. Cross-validation methods can be applied to decision tree induction methods; these methods ensure that the calculated error rate comes close to the true error rate. A large number of decision tree methods exist but the method that works well on average on all kinds of data sets is still the C4.5 decision tree method and some of its variants. Although the user can easily apply this method to his data set thanks to all the different tools that are available and set up in such a way that none computer-science specialist, can use them without any problem, the user is still faced with the problem of how to interpret the result of a decision tree induction method. This problem especially arises when two different data sets for one problem are available or when the data set is collected in temporal sequence. Then the data set grows over time and the results might change.

The aim of this paper is to give an overview of the problems that arise when interpreting decision trees. This paper is aimed at providing the user with a methodology on how to use the resulting model of decision tree induction methods.

In Section 2, we explain the data collection problem. In Section 3, we review how decision tree induction based on the entropy principle works. In Section 4, we present quantitative and qualitative measures that allow a user to judge the performance of a decision tree. Finally, in section 5 we discuss the results achieved so far with our methodology.

2 The Problem

Many factors influence the result of the decision tree induction process. The data collection problem is a tricky pit fall. The data might become very noisy due to some subjective or system-dependent problems during the data collection process.

Newcomers in data mining go into data mining step by step. First, they will acquire a small data base that allows them to test what can be achieved by data mining methods. Then, they will enlarge the data base hoping that a larger data set will result in better data mining results. But often this is not the case.

Others may have big data collections that have been collected in their daily practice such as in marketing and finance. To a certain point, they want to analyze these data with data mining methods. If they do this based on all data they might be faced with a lot of noise in the data since customer behavior might have changed over time due to some external factors such as economic factors, climate condition changes in a certain area and so on.

Web data can change severely over time. People from different geographic areas and different nations can access a website and leave a distinct track dependent on the geographic area they are from and the nation they belong to.

If the user has to label the data, then it might be apparent that the subjective decision about the class the data set belongs to might result in some noise. Depending on the form of the day of the expert or on his experience level, he will label the data properly or not as well as he should. Oracle-based classification methods [12][13] or similarity-based methods [14][15] might help the user to overcome such subjective factors.

If the data have been collected over an extended period of time, there might be some data drift. In case of a web-based shop the customers frequenting the shop might have changed because the products now attract other groups of people. In a medical application the data might change because the medical treatment protocol has been changed. This has to be taken into consideration when using the data.

It is also possible that the data are collected in time intervals. The data in time period _1 might have other characteristics than the data collected in time period_2. In agriculture this might be true because the weather conditions have changed. If this is the case, the data cannot make up a single data set. The data must be kept separate with a tag indicating that they were collected under different weather conditions.

In this paper we describe the behavior of decision tree induction under changing conditions (see Figure 1) in order to give the user a methodology for using decision tree induction methods. The user should be able to detect such influences based on the results of the decision tree induction process.

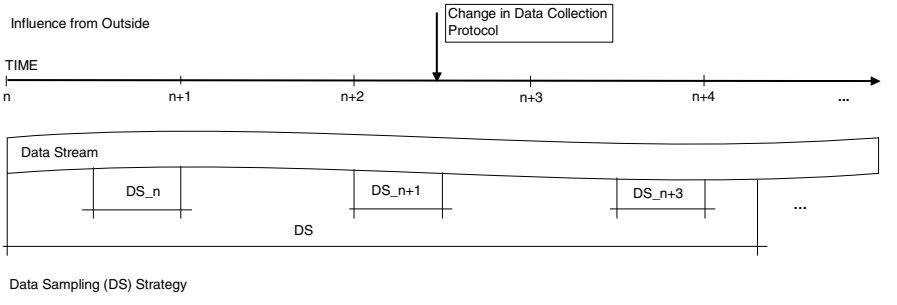

Fig. 1. The Data Collection Problem

3 Decision Tree Induction Based on the Gain Ratio (Entropy-Based Measure)

The application of decision tree induction methods requires some basic knowledge of how decision tree induction methods work. This section reviews the basic properties of decision tree induction.

Decision trees recursively split the decision space into subspaces based on the decision rules in the nodes until the final stop criterion is reached or the remaining sample set does not suggest further splitting (see Figure 2). For this recursive splitting process, the tree building process must always pick from all attributes that one that shows the best result on the attribute selection criteria for the remaining sample set. Whereas for categorical attributes the partition of the attribute values is given a-priori, the partition (also called attribute discretization) of the attribute values for numerical attributes must be determined.

Attribute discretization can be done before or during the tree building process [2]. We will consider the case where the attribute discretization is done during the tree building process. The discretization must be carried out before the attribute selection process since the selected partition regarding the attribute values of a numerical attribute highly influences the prediction power of that attribute.

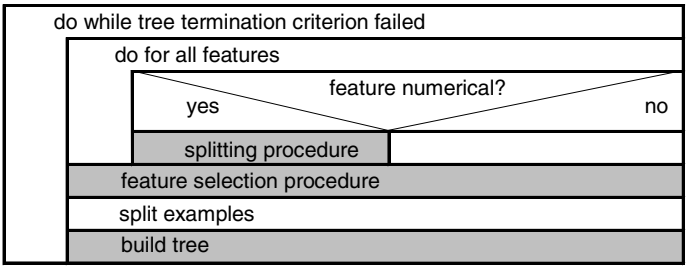

Fig. 2. Overall Tree Induction Procedure

After the attribute selection criterion has been calculated for all attributes based on the remaining sample set, the resulting values are evaluated and the attribute with the best value for the attribute selection criterion is selected for further splitting of the sample set. Then the tree is extended by two further nodes. To each node the subset created by splitting based on the attribute values is assigned and the tree building process is repeated. Decision tree induction is a supervised method. It requires that the data is labeled by its class.

The induced decision tree tends to overfit the data. In Figure 3 we have demonstrated this situation based on a tree induced based on the well-known IRIS data set. Overfit is typically due to noise in the attribute values and class information present in the training set. The tree building process will produce subtrees that fit this noise. This causes an increased error rate when classifying unseen cases. Pruning the tree, which means replacing subtrees with leaves, will help to avoid this problem (see Figure 4). In case of the IRIS data set the pruned tree provides better accuracy than the unpruned tree. However, pruning is often based on a statistical model assumption that might not always fit the particular data. Therefore, their might be a situation where the unpruned tree gives better results than the pruned tree even when checked with new data.

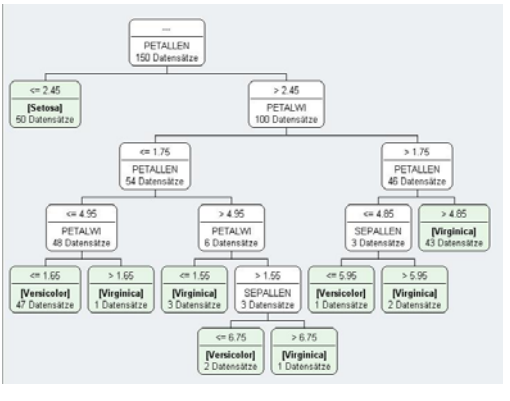

Fig. 3. Decision Tree original

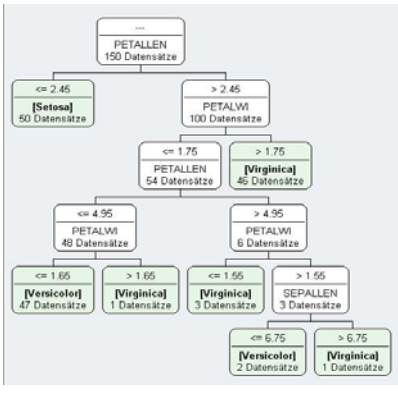

Fig. 4. Decision Tree punend

3.1 Attribute Splitting Criteria

Following the theory of the Shannon channel, we consider the data set as the source and measure the impurity of the received data when transmitted via the channel. The transmission over the channel results in partitioning of the data set into subsets based on splits on the attribute values *J* of the attribute *A*. The aim should be to transmit the signal with the least loss of information. This can be described by the following criterion:

$$IF \quad I(A) = I(C) - I(C/J) = Max \quad THEN \quad Select \quad Attribute - A$$

where *I(A)* is the entropy of the source, *I(C)* is the entropy of the receiver or the expected entropy to generate the message C_1, C_2, ..., C_m, and *I(C/J)* is the lost entropy when branching on the attribute values *J* of attribute *A*.

For the calculation of this criterion we consider first the contingency table in Table 1 with *m* the number of classes, *n* the number of attribute values *J*, *n* the number of examples, L_i the number of examples with the attribute value J_i, R_j the number of examples belonging to class C_j, and x_{ij} the number of examples belonging to class C_j and having attribute value A_i.

Now, we can define the entropy of the class C by:

$$I(C) = -\sum_{j=1}^{m} \frac{R_j}{N} \bullet ld \frac{R_j}{N} \tag{1}$$

The entropy of the class given the feature values is:

$$I(C/J) = \sum_{i=1}^{n} \frac{L_i}{N} \cdot \sum_{j=1}^{m} -\frac{x_{ij}}{L_i} ld \frac{x_{ij}}{L_i} = \frac{1}{N} \left(\sum_{i=1}^{n} L_i ld L_i - \sum_{i=1}^{n} \sum_{j=1}^{m} x_{ij} ld x_{ij} \right) \tag{2}$$

The best feature is the one that achieves the lowest value for (2) or, equivalently, the highest value of the "mutual information" *I(C) - I(C/J).* The main drawback of this measure is its sensitivity to the number of attribute values. In the extreme, a feature that takes N distinct values for *N* examples achieves complete discrimination between different classes, giving *I(C/J)=0*, even though the features may consist of random noise and may be useless for predicting the classes of future examples. Therefore, Quinlan [3] introduced a normalization by the entropy of the attribute itself:

$$\text{G(A)=I(A)/I(J) with } I(J) = -\sum_{i=1}^{n} \frac{L_i}{N} ld \frac{L_i}{N} \tag{3}$$

Other normalizations have been proposed by Coppersmith et. al [4] and Lopez de Montaras [5]. Comparative studies have been done by White and Lui [6].

Table 1. Contingency Table for an Attribute

Class / Attribute Values	C_1		C_j		C_m	SUM
J_1	x_{11}	…	x_{1j}	…	x_{1m}	L_1
	…	…	…	…	…	
J_i	x_{i1}	…	x_{ij}	…	x_{im}	L_i
	…	…	…	…	…	
J_n	x_{n1}	…	x_{nj}	…	x_{nm}	
SUM	R_1		R_j		R_m	L_n

The behavior of the entropy is very interesting [7]. Figure 5 shows the graph for the single term $-p\ ld\ p$. The graph is not symmetrical. It has its maximum when 37% of the data have the same value. In that case this value will trump all other values.

In case of a binary split, we are faced with the situation that there are two sources with the signal probability of p and 1-p. The entropy is shown in Figure 6. It has its maximum when all values are equally distributed. The maximum value for the splitting criterion will be reached if most of the samples fall on one side of the split. The decision tree induction algorithm will always favor splits that meet this situation. Figure 11 demonstrates this situation based of the IRIS data set. The visualization shows to the user the location of the class-specific data dependent on two attributes. This helps the user to understand what changed in the data.

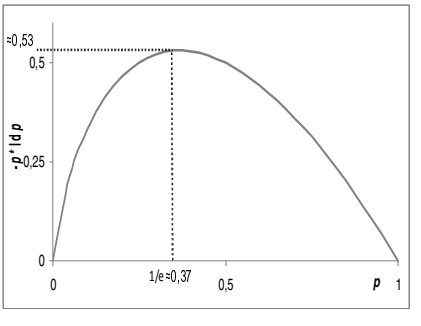

Fig. 5. Diagram of $-p\ ld\ p$; The maximum is at p=1/e; H=0.5 is assumed for p=0.25 and p=0.5

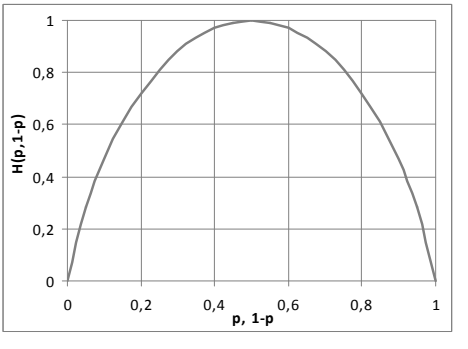

Fig. 6. Behavior of *H(p,1-p)* under the condition that two sources are of the signal probability *p* and *1 - p*

4 How to Interpret the Results of a Decision Tree

4.1 Quantitative Measures of the Quality of the Decision Tree Model

One of the most important measures of the quality of a decision tree is accuracy. This measure is judged based on the available data set. Usually, cross-validation is used for

evaluating the model since it is never clear if the available data set is a good representation of the entire domain. Compared to test-and-train, cross-validation can provide a measure statistically close to the true error rate. Especially if one has small sample sets, the prediction of the error rate based on cross-validation is a must. Although this is a well-known fact by now, there are still frequently results presented that are based on test-and-train and small sample sets. In case of neural nets there is hardly any work available that judges the error rate based on cross-validation. If a larger data set is available, cross-validation is also a better choice for the estimation of the error rate since one can never be sure if the data set covers the property of the whole domain. Faced with the problem of computational complexity, n-fold cross-validation is a good choice. It splits the whole data set into blocks of n and runs cross-validation based theorem .

The output of cross-validation is mean accuracy. As you might know from statistics it is much better to predict a measure based on single measures obtained from a data set split into blocks of data and to average over the measure than predict the measure based on a single shot on the whole data set. Moreover the variance of the accuracy gives you another hint in regard to how good the measure is: If the variance is high, there is much noise in the data; if the variance is low, the result is much more stable.

The quality of a neural net is often not judged based on cross-validation. Cross-validation requires setting up a new model in each loop of the cycle. The mean accuracy over all values of the accuracy of the each single cycle is calculated as well as the standard deviation of accuracy. Neural nets are not automatically set up but decision trees are. A neural network needs a lot of training and people claim that such a neural net – once it is stable in its behavior - is the gold standard. However, the accuracy is judged based on the test-and-train approach and it is not sure if it is the true accuracy.

Bootstrapping for the evaluation of accuracy is another choice but it is much more computationally expensive than cross-validation; therefore, many tools do not provide this procedure.

Accuracy and the standard deviation are an overall measure, respectively. The standard deviation of the accuracy can be taken as a measure to evaluate how stable the model is. A high standard deviation might show that the data are very noisy and that the model might change when new data become available. More detailed measures can be calculated that give a more detailed insight into the behavior of the model [8].

The most widely used evaluation criterion for a classifier is the error rate $f_r = N_f/N$ with N_f the number of false classified samples and N the whole number of samples. In addition, we use a contingency table in order to show the qualities of a classifier, see Table 2. The table contains the actual and the real class distribution as well as the marginal distribution c_{ij}. The main diagonal is the number of correct classified samples. The last row shows the number of samples assigned to the class shown in row 1 and the last line shows the real class distribution. Based on this table, we can calculate parameters that assess the quality of the classifier.

Table 2. Contingency Table

Assigned Class Index \ Real Class Index	**1**	**i**	**...**	**m**	**Sum**
1	c_{11}	...	...	c_{1m}	
...	...	...	...	...	
i	c_{1i}	c_{ii}	...	c_{1m}	
...	...	...	...	...	
j	...	c_{ji}	...	...	
...	...	...	...	...	
m	c_{m1}	...	...	c_{mm}	
Sum					

The ***correctness p*** is the number of correct classified samples over the number of samples:

$$p = \frac{\sum_{i=1}^{m} c_{ii}}{\sum_{i=1}^{m}\sum_{j=1}^{m} c_{ji}} \tag{4}$$

For the investigation of the classification quality we measure the classification quality p_{ki} according to a particular class i and the number of correct classified samples p_{ti} for one class i:

$$p_{ki} \frac{c_{ii}}{\sum_{j=1}^{m} c_{ji}} \qquad p_{ti} = \frac{c_{ii}}{\sum_{i=1}^{m} c_{ji}} \tag{5}$$

Other criteria shown in Table 3 are also important when judging the quality of a model.

Table 3. Criteria for Comparison of Learned Classifiers

Generalization Capability of the Classifier	Error Rate based on the Test Data Set
Representation of the Classifier	Error Rate based on the Design Data Set
Classification Costs	• Number of Features used for Classification • Number of Nodes or Neurons
Explanation Capability	Can a human understand the decision
Learning Performance	Learning Time
	Sensitivity to Class Distribution in the Sample Set

One of these criteria is the cost for classification expressed by the number of features and the number of decisions used during classification. The other criterion is the

time needed for learning. We also consider the explanation capability of the classifier as another quality criterion. It is also important to know if the classification method can learn correctly the classification function (the mapping of the attributes to the classes) based on the training data set. Therefore, we not only consider the error rate based on the test set, we also consider the error rate based on the training data set.

4.2 Explanation Capability of the Decision Tree Model

Suppose we have a data set X with n samples. The outcome of the data mining process is a decision tree that represents the model in a hierarchical rule-based fashion. One path from the top to the leave of a tree can be described by a rule that combines the decisions of each node by a logical AND. The closer the decision is to the leave, the more noise is contained in the decision since the entire data set is subsequently split into two parts from the top to the bottom and in the end only a few samples are contained in the two data sets. Pruning is performed to avoid that the model overfits the data. Pruning provides a more compact tree and often a better model in terms of accuracy.

The pruning algorithm is based on an assumption regarding the distribution of the data. If this assumption does not fit the data, the pruned model does not have better accuracy. Then it is better to stay with the unpruned tree.

When users feel confident about the data mining process, they are often keen on getting more data. Then they apply the updated data set that is combined of the data set X and the new data set X' containing $n+t$ samples ($n < n+t$) to the decision tree induction. If the resulting model only changes in nodes close to the leaf of a decision tree, the user understands why this is so.

There will be confusion when the whole structure of the decision tree has been changed especially, when the attribute in the root node changes. The root node decision should be the most confident decision. The reason for a change can be that there were always two competing attributes having slightly different values for the attribute selection criteria. Now, based on the data, the attribute ranked second in the former procedure is now ranked first. When this happens the whole structure of the tree will change since a different attribute in the first node will result in a different first split of the entire data set.

It is important that this situation is visually presented to the user so that he can judge what happened. Often the user has already some domain knowledge and prefers a certain attribute to be ranked first. A way to enable such a preference is to allow the user to actively pick the attribute for the node.

These visualization techniques should allow to show to user the location of the class-specific data dependent on two attributes, as shown in Figure 11. This helps the user to understand what changed in the data. From a list of attributes the user can pick two attributes and the respective graph will be presented.

Another way to judge this situation is to look for the variance of the accuracy. If the variance is high, this means that the model is not stable yet. The data do not give enough confidence in regard to the decision.

The described situation can indicate that something is wrong with the data. It often helps to talk to the user and figure out how the new data set has been obtained. To give you an example: A data base contains information about the mortality rate of patients that have been treated for breast cancer. Information about the patients, such

as age, size, weight, measurements taken during the treatment, and finally the success or failure, is reported. In the time period T1, treatment with a certain cocktail of medicine, radioactive treatment and physiotherapy has taken place; the kind of treatment is called a protocol. In the time period T2, the physicians changed the protocol since other medicine is available or other treatment procedures have been reported in the medical literature as being more successful. The physicians know about the change in protocol but they did not inform you accordingly. Then the whole tree might change and as a result the decision rules are changing and the physicians cannot confirm the resulting knowledge since it does not fit their knowledge about the disease as established in the meantime. The resulting tree has to be discussed with the physicians; the outcome may be that in the end the new protocol is simply not good.

4.3 Revision of the Data Label

Noisy data might be caused by wrong labels applied by the expert to the data. A review of the data with an expert is necessary to ensure that the data labels are correct. Therefore, the data are classified by the learnt model. All data sets that are misclassified are reviewed by the domain expert. If the expert is of the opinion that the data set needs another label, then the data set is relabeled. The tree is learnt again based on the newly labeled data set.

4.4 Comparison of Two Decision Trees

Two data sets of the same domain that might be taken at different times, might result in two different decision trees. Then the question arises how similar these two decision trees are. If the models are not similar then something significant has changed in the data set.

The path from the top of a decision tree to the leaf is described by a rule like "IF attribute A<= x and attribute B<=y and attribute C<=z and … THEN Class_1". The transformation of a decision tree in a rule-like representation can be easily done. The location of an attribute is fixed by the structure of the decision tree.

Comparison of rule sets is known from rule induction methods in different domains [9]. Here the induced rules are usually compared to the human-built rules [10][11]. Often this is done manually and should give a measure about how good the constructed rule set is.

These kinds of rules can also be automatically compared by substructure mining. The following questions can be asked: a) How many rules are identical? b) How many of them are identical compared to all rules? b) What rules contain part structures of the decision tree?

We propose a first similarity measure for the differences of the two models as follows.

1. Transform two decision trees d_1 and d_2 into a rule set.
2. Order the rules of two decision tress according to the number n of attributes in a rule.

3. Then build substructures of all l rules by decomposing the rules into their Substructures.
4. Compare two rules i and j of two decision trees d_1 and d_2 for each of the n_j and n_i substructures with s attributes.
5. Build similarity measure SIM_{ij} according to formula 6-8.

The similarity measure is:

$$SIM_{ij} = \frac{1}{n}(Sim_1 + Sim_2 + ... + Sim_k + ... + Sim_n) \tag{6}$$

with $n = \max\{n_i, n_j\}$ and

$$Sim_k = \begin{cases} 1 & \text{if substucture identity} \\ 0 & \text{if otherwise} \end{cases} \tag{7}$$

If the rule contains a numerical attribute $A<=k_1$ and $A´<=k_2=k_1+x$ then the similarity measure is

$$Sim_{num} = 1 - \frac{A - A'}{t} = 1 - \frac{k_1 - k_1 - |x|}{t} = 1 - \frac{|x|}{t} \; for \; x < t$$
$$Sim_k = 0 \text{ for } x \geq t \tag{8}$$

with t a user chosen value that allows x to be in a tolerance range of s % (e.g. 10%) of k_1. That means as long as the cut-point k_1 is within the tolerance range we consider the term as similar, outside the tolerance range it is dissimilar. Small changes around the first cut-point are allowed while a cut-point far from the first cut-point means that something seriously has happened with the data.

The similarity measure for the whole substructure is:

$$Sim_k = \frac{1}{s}\sum_{z=1}^{s} \begin{cases} Sim_{num} \\ 1 \quad \text{for } A = A' \\ 0 \quad \text{otherwise} \end{cases} \tag{9}$$

The overall similarity between two decision trees d_1 and d_2 is

$$Sim_{d_1,d_2} = \frac{1}{l}\sum_{i=1}^{l} \max_{\forall j} Sim_{ij} \tag{10}$$

for comparing the rules i of decision d_1 with rules j of decision d_2. Note that the similarity $Sim_{d1,d2}$ must not be the same.

The comparison of decision tree_1 in Figure 7 with decision tree_2 in Figure 8 gives a similarity value of 0.75 based on the above described measure. The upper structure of decision tree_2 is similar to decision tree_1 but decision tree_2 has a few more lower leaves. The decision tree_3 in Figure 9 is similar to decision tree_1 and decision_tree_2 by a similarity value of 0.125. Decision tree_3 in Figure 10 has no similarity at all compared to all other trees. The similarity value is zero.

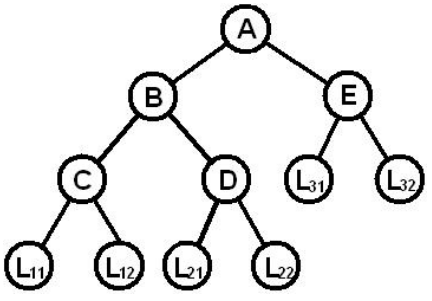

Fig. 7. Decision_Tree_1, $Sim_{d1,d1}=1$

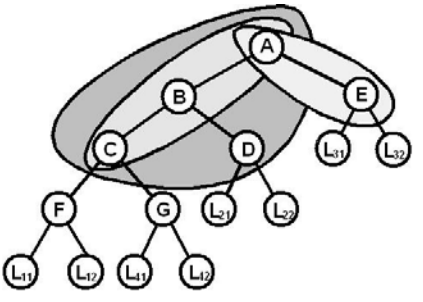

Fig. 8. Substructures of Decision Tree_1 to Decision Tree_2; $Sim_{d1,d2}=0.9166$

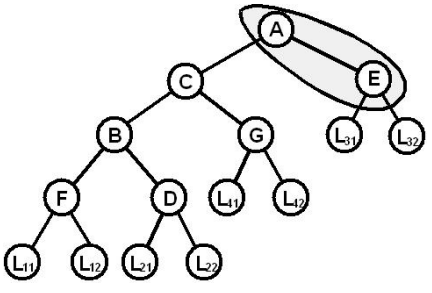

Fig. 9. Substructures of Decision Tree_1 to Decision Tree_3; $Sim_{d1,d3}=0,375$; $Sim_{d2,d3}=0.375$

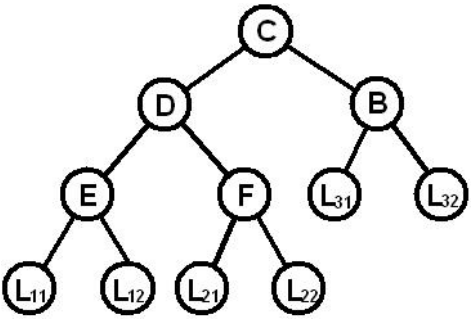

Fig. 10. Decision Tree_4 dissimilar to all other Decision Trees, $Sim_{d1,d4}=0$

Such a similarity measure can help an expert to understand the developed model and also help to compare two models that have been built based on two data set, wherein one contains N examples and the other one contains $N+L$ samples.

There are other options for constructing the similarity measure. It is left for our further work.

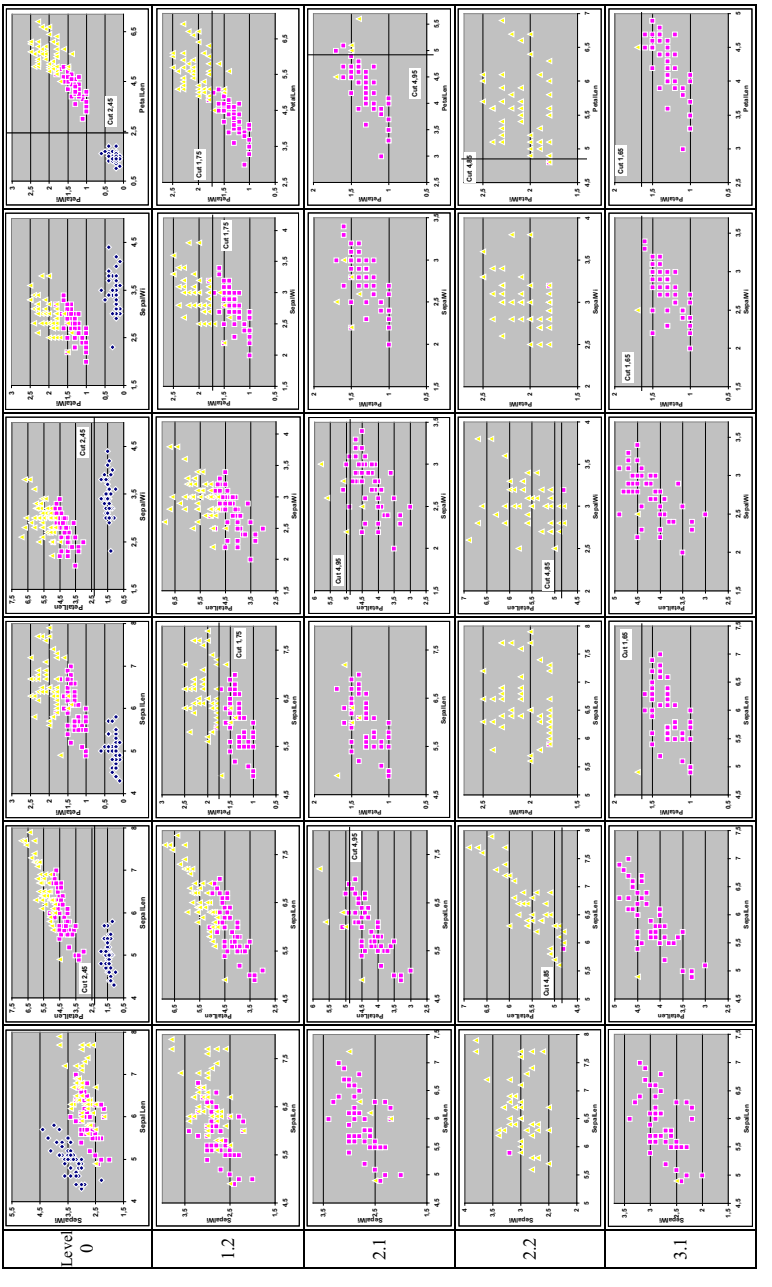

Fig. 11. Decision Surface for two Attributes on each Level of the Decision Tree shown in Figure 3

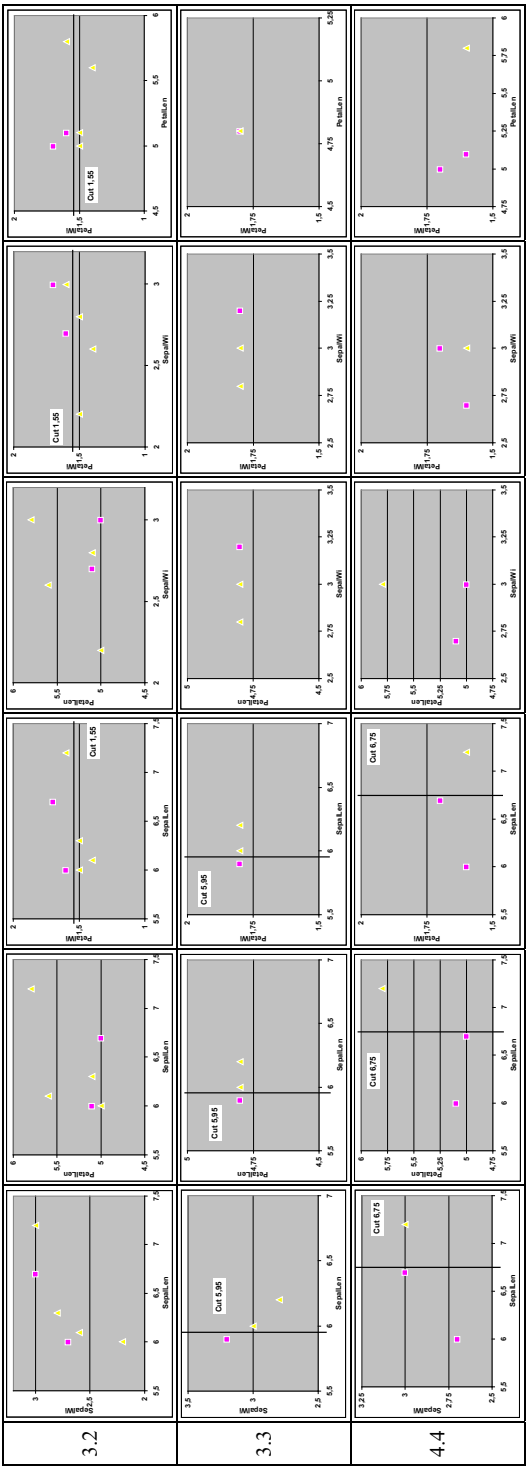

Fig. 11. (*Continued*)

5 Conclusions

The aim of this paper is to discuss how to deal with the result of data mining methods such as decision tree induction. This paper has been prompted by the fact that domain experts are able to use the tools for decision tree induction but have a hard time interpreting the results. A lot of factors have to be taken into consideration. The quantitative measures give a good overview in regard to the quality of the learnt model. But computer science experts claim that decision trees have explanation capabilities and that, compared to neural nets and SVM, the user can understand the decision. This is only partially true. Of course, the user can follow the path of a decision from the top to the leaves and this provides him with a rule where the decisions in the node are combined by logical ANDs. But often this is tricky. A user likes rules that fit his domain knowledge and make sense in some way. Often this is not the case since the most favored attributes of the user do not appear at a high position.

That makes the interpretation of a decision trees difficult. The user´s domain knowledge, even if it is only limited, is an indicator whether he accepts the tree or not. Some features a decision tree induction algorithm should have are mentioned in this paper. The decision tree induction algorithm should allow the user to interact with the induction algorithm. If two attributes are ranked more or less the same the user should be able to choose which one of the attributes to pick. The noise in the data should be checked with respect to different aspects. Quality measures of the model like mean accuracy, standard deviation and class-specific accuracy are necessary in order to judge the quality of a learnt decision tree right. The evaluation should be done by cross validation as test-and-train methods are not up-to-date anymore.

Among other things, explanation features are needed that show the split of the attributes and how it is represented in the decision space. Simple visualization techniques, like 2-d diagram plots, are often helpful to discover what happened in the data.

Wrong labels have to be discovered by an oracle-based classification scheme. This should be supported by the tool.

The comparison of trees is another important issue that a user needs in order to understand what has changed. Therefore, proper similarity measures are needed that give a measure of goodness.

This paper is a first step toward a more complex methodology on how to interpret decision trees.

References

1. Perner, P. (ed.): Data Mining on Multimedia Data. LNCS, vol. 2558. Springer, Heidelberg (2002)
2. Dougherty, J., Kohavi, R., Sahamin, M.: Supervised and Unsupervised Discretization of Continuous Features. In: 14th IJCAI on Machine Learning, pp. 194–202 (1995)
3. Quinlan, J.R.: Decision trees and multivalued attributes. In: Hayes, J.E., Michie, D., Richards, J. (eds.) Machine Intelligence, vol. 11. Oxford University Press, Oxford (1988)
4. Copersmith, D., Hong, S.J., Hosking, J.: Partitioning nominal attributes in decision trees. Journal of Data Mining and Knowledge Discovery 3(2), 100–200 (1999)

5. de Mantaras, R.L.: A distance-based attribute selection measure for decision tree induction. Machine Learning 6, 81–92 (1991)
6. White, A.P., Lui, W.Z.: Bias in information-based measures in decision tree induction. Machine Learning 15, 321–329 (1994)
7. Philipow, E.: Handbuch der Elektrotechnik, Bd 2 Grundlagen der Informationstechnik, pp. 158–171. Technik Verlag, Berlin (1987)
8. Perner, P., Zscherpel, U., Jacobsen, C.: A Comparision between Neural Networks and Decision Trees based on Data from Industrial Radiographic Testing. Pattern Recognition Letters 22, 47–54 (2001)
9. Georg, G., Séroussi, B., Bouaud, J.: Does GEM-Encoding Clinical Practice Guidelines Improve the Quality of Knowledge Bases? A Study with the Rule-Based Formalism. In: AMIA Annu Symp Proc. 2003, pp. 254–258 (2003)
10. Lee, S., Lee, S.H., Lee, K.C., Lee, M.H., Harashima, F.: Intelligent performance management of networks for advanced manufacturing systems. IEEE Transactions on Industrial Electronics 48(4), 731–741 (2001)
11. Bazijanec, B., Gausmann, O., Turowski, K.: Parsing Effort in a B2B Integration Scenario - An Industrial Case Study. In: Enterprise Interoperability II, Part IX, pp. 783–794. Springer, Heidelberg (2007)
12. Muggleton, S.: Duce - An Oracle-based Approach to Constructive Induction. In: Proceeding of the Tenth International Join Conference on Artificial Intelligence (IJCAI 1987), pp. 287–292 (1987)
13. Wu, B., Nevatia, R.: Improving Part based Object Detection by Unsupervised, Online Boosting. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2007, pp. 1–8 (2007)
14. Whiteley, J.R., Davis, J.F.: A similarity-based approach to interpretation of sensor data using adaptive resonance theory. Computers & Chemical Engineering 18(7), 637–661 (1994)
15. Perner, P.: Prototype-Based Classification. Applied Intelligence 28(3), 238–246 (2008)