Sensitivity Analysis of the Result in Binary Decision Trees

Isabelle Alvarez^{1,2}

¹ LIP6, University of Paris VI, 5 rue Descartes, F-75005 Paris, France

isabelle.alvarez@lip6.fr

² Cemagref-LISC, BP 50085, F-63172 Aubière Cedex, France

Abstract. This paper proposes a new method to qualify the result given by a decision tree when it is used as a decision aid system. When the data are numerical, we compute the distance of a case from the decision surface. This distance measures the sensitivity of the result to a change in the input data. With a different distance it is also possible to measure the sensitivity of the result to small changes in the tree. The distance from the decision surface can also be combined to the error rate in order to provide a context-dependent information to the end-user.

1 Introduction

Decision trees (DT) are very popular as decision aid systems (see [1] for a short review of real world application), since they are supposed to be easy to build and easy to understand. DT are used for instance in medicine to infer the diagnosis or to establish the prognosis of several diseases (see [2] for references). They are used in credit scoring (see [3] for references) and in other domains to solve classification problems. DT algorithms are also integrated in many software for data mining or decision support purpose. The end-user of a DT submits a new case to the DT which predicts a class. Additional information is generally available to help the end-user to appreciate the result: At least the confusion matrix and some estimate of the error rate (accuracy). Specific rates (like specificity, sensitivity and likelihood ratios which are used in diagnosis) and costs matrix are eventually used to take into account the difference between false positive and false negative [4]. This additional information is essential but generally it focuses exclusively on the result and not on the case itself: This is obvious for global error rates (which are identical for all cases), but it is also true for error rates which are estimated at a leaf. Even if local error rates can estimate the posterior probabilities, they carry much information about the result (the probability of the case to belong to the predicted class), but little about the link between the case and the predicted class. In fact, membership of a particular leaf depends on the path followed in the tree, which is an arbitrary description of the partition of the input space induced by the DT. Therefore its relevance is limited as context-dependent information.

We propose here to provide for the end-user context-dependent information about the result given by the DT for a particular case. This is achieved by a

study of the sensitivity of the result to the change in the input data. Sensitivity analysis consists in the study of the relative position of the input case and the decision surface (the boundary between regions with different class label). It measures the robustness of the result to uncertainty or to small changes in the input data, since it gives the distance of the case from the decision surface. It also exhibit the smallest move to apply to the case to make the decision change. It can also give information about the robustness of the result to small changes in the tree. A simple example¹ shows the interest of sensitivity analysis; Let us consider two different cases from the Pima Indian Diabetes (Pima) database [5], which attributes are shown in Table 1. They are classified by the same leaf. Therefore their error rates are the same. One of the cases is nevertheless very close to the decision surface. A very small change of the value of the attributes can change the decision (moreover it is misclassified). The other case is relatively far from the decision surface. The case can cross the decision boundary only if its attributes values are much modified. Conversely, the decision boundary has to move significantly to make the decision change for the latter case, and it is easy to compute the minimum change of the thresholds of the tests of the DT that is necessary to reach the case. So, in this example, the distance from the decision surface clearly carries interesting information that is not contained in the error rate.

This kind of information is available in geometric classifiers, like support vector machines (SVM). A SVM defines a unique hyperplane in the feature space to classify the data. But in the original input space the corresponding decision surface can be very complex, depending on the kernel that defines the dot product in the feature space [6]. The distance from the decision surface is generally visualized by contour lines, and it can be used to estimate the posterior probabilities [7]. In the case of DT operating on numerical data, the decision surface consists in several pieces of hyperplanes instead of a unique hyperplane. For DT with hyperplanes that are normal to axes (NDT), also called axis-parallel DT, a very simple algorithm can be used to compute the distance of a case from the decision surface, if it is possible to define a metric on the input space. This information can then be used to assess the robustness of the result to changes in the input data and the robustness of the result to changes in the thresholds values of the tests of the tree. It can also be combined with error information to provide case-dependent error rates.

The rest of the paper is organized as follow: Section 2 discusses related work on sensitivity (to change in the input data or in the model) and context-dependent information. Section 3 presents the geometric sensitivity analysis for DT, the algorithm and some properties of the distance from the decision surface: Robustness, influence of the metric and theoretical justification. Section 4 presents our experimental results and possible applications of the sensitivity analysis method. Section 5 presents in a concluding section our remarks and suggestions for further work.

¹ The complete example is presented in Sect. 4.1.

2 Related Work

As we have noted in the introduction section, sensitivity analysis gives information on the robustness of the result to changes in the input data and also to changes in the decision surface, that is the model itself. A lot of work has been done to assess the robustness of the classifier, but it is generally considered as a global criteria rather than related to a particular case. The objective is to understand better the learning algorithm (see [8] for references), or to produce a better classifier. For example, methods for model selection (see [9], [10]) try to identify the best model according to a given criteria, generally accuracy or specific error rates. The best model then applies to every new case. Methods based on mixture of expert [11] also aim at producing better model, reducing the variance induced by the partition of the input space. But they cannot easily take into account small changes in the input case, since the predicted class (or predicted probability) is a combination of partial results. Fuzzy DT ([12]) try also to take into account the possible fluctuations of the breakpoint value of the tests, that are calculated on a learning sample. This can be done by defining a fuzzy area around the hyperplane supporting a test with membership functions (see [13] for an example). The uncertainty in the input data is represented by fuzzy data. However, information provided by this method is not easy to use. The computation of the final result is opaque to the end-user, in particular because a point can be close to a hyperplane that has no interest for its classification. For cases outside the fuzzy area, it gives no sensitivity information (for example, how to proceed to significantly change the result).

The main information that is relatively context-dependent is the error rate at the leaf (EL). In most cases it is based on the resubstitution error, with some attempt to correct the overoptimistic bias (see [14], [13], [15], [16]). Cross-validation and resampling techniques are widely used for that purpose. In a similar concern, another approach consists in building DT that estimate directly probability estimators (for example PETs [17] and curtailment [18]). When the EL is correctly estimated, it gives a statistical information on the result obtained at the leaf. In practice, these estimators are not always available, since they are developed and used for the construction of the tree and not for the end-user's need. They are also not necessarily accurate ([9], [19]). Moreover, the leaf is an arbitrary division of the connected component of a case. So the link between the case and the decision surface is not easy to understand.

3 Sensitivity Analysis in DT

3.1 Definitions and Notations

Sensitivity analysis is the study of the relative position of the input case and the decision surface. The result of sensitivity analysis consists in the nearest point of the decision surface to the input case (or, equivalently, in the smallest vector of changes that makes the input data cross the decision boundary). Therefore it

assumes that it is possible to define a metric on the input space, eventually with a cost or utility function.

We consider here linear DT (LDT): each test consists in computing the algebraic distance h of a new case x from a hyperplane H . The point x passes the test depending on the sign of $h(x, H)$. When the test involves only one attribute, the hyperplane H is normal to the attribute axis. In NDT, all the hyperplanes defining the tests are normal to an attribute axis.

The DT induces a partition of the input space E . We note Γ the associated decision surface. Γ consists of pieces of hyperplanes, so it is piecewise affine and continuously differentiable almost everywhere. At each point y of Γ , the decision surface is defined by a list $L(y)$ of hyperplanes (often reduced to a unique hyperplane).

Let x be a point of the input space E . The sensitivity at point x is $d(x, \Gamma)$, the distance of x from the decision surface. Actually, any point in the open ball with center at x and radius $d = d(x, \Gamma)$ has the same predicted class as x . There is at least one point in any open Ball with center at x and radius $d + \epsilon$ which predicted class is different from x class. Moreover, if E is a complete metric space, there is at least one point $p(x)$ for which we have $d(x, p(x)) = d(x, \Gamma)$. When E is an Euclidean space, $p(x)$ is the projection of x onto Γ and $\overrightarrow{xp(x)}$ is orthogonal to the vector space associated to $(\bigcap H)_{H \in L(p(x))}$.

Definition 1. *The sensitivity at point x is $d(x, \Gamma)$. The sensitive move at point x is the vector $\overrightarrow{xp(x)}$ with $d(x, p(x)) = d(x, \Gamma)$.*

When E is an Euclidean space, there is generally only one sensitive move at point x , since points outside the skeleton² of the connected component of their class have a unique projection onto Γ .

3.2 Robustness of the Sensitivity

Sensitivity is not very sensitive to the uncertainty in the new case, and it is relatively robust to the noise in the training data, assuming that only the thresholds of the tests are modified (not the attributes).

Theorem 1. *Close points have similar sensitivity. Let x and y be two input points. We have:*

$$d(x, y) < \epsilon \Rightarrow |d(y, \Gamma) - d(x, \Gamma)| < \epsilon . \quad (1)$$

Theorem 2. *Small local deformations of the decision surface lead to small variation of the sensitivity. Let Φ be a small local deformation of the decision surface Γ (such that $\Phi(\Gamma)$ is still a DT surface decision). We have:*

$$\forall z \in \Gamma, d(z, \Phi(z)) < \epsilon \Rightarrow |d(x, \Phi(\Gamma)) - d(x, \Gamma)| < \epsilon . \quad (2)$$

² The skeleton is the locus of the centers of maximal balls. In the case of linear DT, it is complementary to a dense open set, so it is a rather small set.

Proof of Theorems 1 and 2 is straightforward because of triangle inequality.

Theorem 3. *In the NDT case, if E is an Euclidean space, small changes of the thresholds of the tests of the tree lead to relatively small variation of the sensitivity. Let Φ be a deformation of the decision surface such that the thresholds of the tests of the NDT are changed by less than ϵ . We have:*

$$|d(x, \Phi(\Gamma)) - d(x, \Gamma)| < \epsilon \sqrt{\dim(E)} . \quad (3)$$

Proof. A change of the threshold values implies a translation of the hyperplanes along their normal vector. The points at which Γ is differentiable move from ϵ . The points at which Γ is not differentiable can move from ϵ to $\epsilon \sqrt{\dim(E)}$, since $\sqrt{\dim(E)}$ is the diagonal of the unit hypercube in the input space E .

Theorem 3 is verified only in the NTD case, since in the general case of oblique linear trees, the decision surface can vary a lot at points where it is not differentiable.

Similar theorems exist for the sensitive move, but the variation ϵ (of the distance of the input point, of the local deformation, of the thresholds) has to be smaller than the distance of x from the skeleton.

3.3 Algorithm in the NDT Case

In the following we shall assume that the input space E is an Euclidean space. Computing the distance from a piecewise linear surface is a very classical problem and many fast converging algorithms are available (see [20] for a review). In the NDT case, however, the set C_f of points classified by a leaf f is an hyperrectangle, and the straightforward projection is easy to compute. Following an idea from [21], we associate to f the list of the tests $(h(x, H_i))_{i \in I}$ that lead to it. f classifies the points that belong to the intersection C_f of the half spaces $E(H_i)$ defined by the tests:

$$C_f = \bigcap_{i \in I} E(H_i) . \quad (4)$$

The sensitivity at point x is the smallest distance of x from the leaves which class label $c(f)$ is different from x predicted class $c(x)$. The algorithm `sensitivityAt(x,DT)` that computes the sensitivity and the sensitive move at point x consists in projecting x onto every leaf f such that $c(f) \neq c(x)$. Then it computes and ranks the distance between x and its projections.

Algorithm 1 *sensitivityAt(x,DT)*

1. Gather the set F of leaves f of the DT which class $c(f) \neq c(x)$;
2. For each $f \in F$ do: {
3. compute $p_f(x) = \text{projectionOntoLeaf}(x, f)$;
4. compute and rank $d(x, p_f(x))$ }
5. Return $(d(x, p_n(x)), \overline{xp_n(x)})$ with $n = \text{argmin}_{f \in F} (d(x, p_f(x)))$

The projection onto a leaf f is straightforward since the hyperplanes defining f in Equation 4 are either parallel or orthogonal.

Algorithm 2 *projectionOntoLeaf*($x, (E(H_i))_{i \in I}$)

1. $y = x$;
2. For $i = 1$ to $\text{size}(I)$ do: {
3. if $y \notin E(H_i)$ then $y_u = b$;
 with u the coordinate corresponding to the attribute defining H_i
 b the threshold value for H_i . }
4. Return y

Remark 1 (invariance of the projection onto a leaf). The projection of a point onto a leaf is invariant under a change of metric (for instance, a dilatation) that conserves the hyperplanes that define the leaf.

The projectionOntoLeaf algorithm doesn't depend on the dimension of the input space. The complexity of the sensitivity algorithm is in $O(Nd)$ where N is the number of tests of the tree and d the dimension of the input space.

4 Experimental Results and Application

Sensitivity information can be used in two ways. First, it gives sensitivity results for individual cases. Second, it measures the sensitivity of the result to uncertainty (in the attributes values), and also the sensitivity to thresholds changes.

4.1 Sensitivity to Input Changes

The first use of the sensitivity information is to measure the possibility of small changes of the input data without changing the class.

The following example come from the Pima Indian Diabetes database (Pima) from the UCI repository [5]. The attributes are female patient information and medical test measurements linked to diabetes. The attributes are meaningful and numerical, and (theoretically) there is no missing value. The database was divided randomly into a training base and a test base (1/3 of the cases), with respect of the prevalence of diabetes in the base (35%). Weka j48 [22], a C4.5-based algorithm [13] was used to grow decision trees on the training database. Then the sensitivity algorithm was applied to the test cases.

We illustrate the first use with two test cases from the Pima database (see Table 1). Both cases are classified by the same leaf.

Two different metrics were used to apply the sensitivity algorithm, the Min-Max (MM) metric and the standard (s) metric. Both metrics are defined with the basic information available on the training data set: An estimate of the range of each attribute i or an estimate of its mean E_i and of its standard deviation s_i . The new coordinate system is defined by Equation 5.

$$y_i^{MM} = \frac{x_i - \text{Min}_i}{\text{Max}_i - \text{Min}_i} \quad \text{or} \quad y_i^s = \frac{x_i - E_i}{s_i} . \quad (5)$$

Table 2 shows the sensitivity for both cases. We can see that the sensitivity of Case 188 is very small, since for the Min-Max metric, the maximal size of a

Table 1. Cases of the Pima test database

Attribute	Case 188	Case 63
Glucose concentration in plasma	90	93
Diastolic blood pressure (mmHg)	68	50
Body mass index (kg/m ²)	38.2	28.7
Diabetes pedigree function	0.503	0.356
Age (years)	27	23
Predicted class	0 (not diabetic)	0
Real class	1 (diabetic)	0

vector is $\sqrt{5} \approx 2.24$. It is also very small compared to the norm of the standard error vector. It can also be compared to the maximum possible distance (in the Pima application, one leaf is defined by a single hyperplane, so it gives a base distance). On the contrary, Case 63 is not so close to the decision surface.

The sensitive move of the case 188 is shown in Table 3 (in Table 4 for the case 63). The sensitive move is the smallest change that has to be applied to the case to make the decision change (its norm is the sensitivity). It applies to several attributes that have to be modified together. Its coordinates can be compared to their present value and to several indicators that are generally used to measure uncertainty (percentage of the range, percentage of the standard deviation). We can see that Case 188 is near the decision surface: all the coordinates of its sensitive move are small. Conversely, Case 63 is relatively far from the decision surface: The value of one of its attribute has to increase by more than 50% of its value, which represents also more than 50% of the standard deviation.

The individual sensitivity results can be gathered from a test base, in order to estimate the impact of a variation of the input data on the predicted class. Because of the use of the Euclidean distance, the different attributes can substitute for one another. Table 5 shows the proportion of cases of the test database which class is likely to change for several databases from the UCI repository.

4.2 Sensitivity to Uncertainty and Thresholds Changes

The sensitivity can also be computed, in order to estimate the impact of the uncertainty on the predicted class.

Let x be a point of the input space. When the thresholds of the tests of the tree are modified by ϵ , or when the value of the attributes of x can move at the

Table 2. Sensitivity information for the Pima cases

Metric	Sensitivity		Norm of the SE vector	Max. of the distance
	Case 188	Case 63		
Min-Max	0.038	0.123	0.35	0.7
Standard	0.223	1.474	1	4.3

Table 3. Sensitive move for the Pima case 188

Attribute	Sensitive Move	% of the value	% of the range	% of the standard deviation
Glucose concentration	4.5	5.0	2.3	14.2
Diastolic blood pressure	0	0	0	0
Body mass index	0	0	0	0
Diabetes pedigree function	0.043	8.5	1.8	13.0
Age	1.5	5.6	2.5	12.5

Table 4. Sensitive move for the Pima case 63

Attribute	Sensitive Move	% of the value	% of the range	% of the standard deviation
Glucose concentration	1.5	1.6	0.7	4.7
Diastolic blood pressure	0	0	0	0
Body mass index	0	0	0	0
Diabetes pedigree function	0.19	53.4	8.1	57.6
Age	5.5	23.9	9.2	45.8

most from ϵ , the relative position between x and the decision surface has to be compared not with the Euclidean distance but with the sup-norm distance. The sup-norm of $z = (z_i)_{i \in \{1, \dots, n\}}$ is defined by:

$$\|z\|_\infty = \sup_{i \in \{1, \dots, n\}} |z_i| . \quad (6)$$

Let $p_f(x)$ be the projection of x onto the leaf f , and let $L(p_f(x))$ be the list of the hyperplanes that define Γ at $p_f(x)$ (cf. Section 3.1). We have:

Theorem 4. *Let Φ be a deformation of the decision surface such that the thresholds of the tests of the DT are changed by less than ϵ . We note $c_\Phi(x)$ the predicted class of x given by the new DT, $\Phi(DT)$. The class of x remains unchanged if the sup-norm distance $d_\infty(x, \Gamma)$ of x from the decision surface Γ is less than the move of the thresholds.*

$$\min_{f \in F} \left(\max_{H_i \in L(p_f(x))} d(x, H_i) \right) < \epsilon \Rightarrow c_\Phi(x) = c(x) . \quad (7)$$

Table 5. Proportion of likely class-modified cases in function of the input change

Base	Variation of the distance				
	0.01	0.02	0.05	0.1	0.2
Pima	5.9	10.2	28.1	53.9	90.6
Ionosphere	6.6	12.6	39.1	64.2	81.5
Wine	1.6	1.6	1.6	5.0	6.7

Theorem 5. *Let Φ be a move of the point x such that the coordinates of x are changed by less than ϵ . The class of $\Phi(x)$ is the same as the class of x if the sup-norm distance of x from the decision surface Γ is less than ϵ .*

$$\min_{f \in F} \left(\max_{H_i \in L(p_f(x))} d(x, H_i) \right) < \epsilon \Rightarrow c(\Phi(x)) = c(x) . \quad (8)$$

The algorithm `sup-sensitivityAt(x,DT)` is directly derived from the algorithm `sensitivityAt(x,DT)`. In step 4 and 5 the Euclidean distance is replaced by the sup-norm distance, which is easier to compute.

Algorithm 3 *sup-sensitivityAt(x,DT)*

1. Gather the set F of leaves f of the DT which class $c(f) \neq c(x)$;
2. For each $f \in F$ do: {
3. compute $p_f(x) = \text{projectionOntoLeaf}(x, f)$;
4. compute and rank $d_\infty(x, p_f(x))$ }
5. Return $(d_\infty(x, p_n(x)), u)$ with $n = \text{argmin}_{f \in F} (d_\infty(x, p_f(x)))$
and $u = \text{argmax}_{i \in (1, \dots, \dim E)} d_\infty(x, p_n(x)) = |x_i - p_n(x)_i|$

The choice of normalization will be guided by the data source. For instance, the accuracy of most sensors is a function of the range, so it suggests to use an adapted version of the Min-Max coordinate system. If a_i is the accuracy of attribute i , the modified coordinate system y_i^{mMM} is defined by Equation 9:

$$y_i^{mMM} = \frac{1}{a_i} \cdot \frac{x_i - \text{Min}_i}{\text{Max}_i - \text{Min}_i} . \quad (9)$$

Remark 2. Since it is a dilatation, if the sensitivity analysis was already done with another coordinate system, it is only necessary to compute and rank again the distance between the test cases and their projections (which are invariant under this change of coordinate system).

Remark 3. In the case of oblique trees also, the sensitivity to attributes changes is the sup-sensitivity, so projection algorithms like in [20] should be modified to use the sup-norm instead of the Euclidean norm. But it is no longer equivalent to the sensitivity to thresholds changes, which has to be computed differently. We suggest to replace Step 3 and 4 of the sensitivity algorithm by the following expression (where $L(f)$ lists the hyperplanes defining f):

compute and rank $\epsilon_f = |h(x, H_f)|$ with $H_f = \text{argmin}_{L(f)} h(x, H)$.

It selects the absolute value of the most negative algebraic distance for each leaf. The minimum ϵ of ϵ_f over F give the sensitivity to threshold changes. If the hyperplanes move along their normal vector from ϵ (in the direction opposite to the interior of the leaf when its class is different than x class), then the decision surface reaches x .

Table 6 shows the proportion of cases of the test database which class is likely to change for several databases from the UCI repository. The proportion of cases can grow quicker than in Table 5 since a move of 1% per attribute leads to a move of total length of $\sqrt{\dim E}$.

Table 6. Proportion of likely class-modified cases in function of the input change (in % of the range for each attribute)

Base	Variation of the value of the attributes					
	$\frac{1\%}{\sqrt{\dim E}}$	$\frac{2\%}{\sqrt{\dim E}}$	1%	2%	5%	10%
Pima	3.9	8.2	10.9	25.0	59.8	94.5
Ionosphere	6.6	12.6	24.5	52.3	77.5	90.1
Wine	1.6	1.6	1.6	1.6	1.6	5.0

Fig. 1. Frequency histogram of the distance from the decision surface (Pima test base)

4.3 Sensitivity and Error Information

When a sample of labelled test data is available, it could be very interesting to estimate the probability of each class f conditionally to the distance from the decision surface. This is done for other geometric method like SVM (see for references [7]). The posterior probabilities have to be estimated on the test base, not on the training base, since supervised learning algorithms play a part in the localization of errors. Here we just show for the Pima and the Ionosphere databases the frequency histogram of the distribution of the distance from the decision surface, with the proportion of errors and well-classified cases. Better estimates could be obtained with resampling methods. We already see in Figure 1 that it confirms the ordinary hypothesis that errors are mostly located near the decision surface. The end-user can see that for case 188, the proportion of errors at this distance is greater than 40%. For case 63, it is less than 20%. Retrospectively, it also suggests that the distance could be used to rank new and unlabelled cases.

5 Conclusion

In this paper, we have proposed an approach based on geometry, which objective is to provide the end-user with information about the quality of the output of

Fig. 2. Frequency histogram of the distance from the decision surface (Ionosphere)

a decision tree used as a prediction tool. Sensitivity analysis gives information that error rates cannot produce. Examples show that it is easy to implement in the case of numerical decision trees with linear separators normal to axes. In the case of oblique trees, the main difficulty concerns the algorithm of projection onto a leaf, which is not as simple (but other algorithms could be used, see in [20]). The main limit of the method is that current efficient algorithms need a scalar product. However, the choice of the metric in order to apply sensitivity analysis is not as crucial as it is for other geometric methods, where an optimal metric is sought to improve the classification efficiency. For sensitivity analysis the use of the metric is related to a particular point and it has a limited range. All we need is a metric that can be used in a relatively small neighborhood of a particular input case. If it is not possible to define a metric operating everywhere (particularly in the case of non numerical data), it will be necessary to define local metric and scalar product, depending on the domain, for instance by the way of cost or utility functions (or with the help of a domain expert).

The use of geometry was limited here to the definition of a metric for the sensitivity analysis. But it can also be used to describe the relative situation of a case in its own class. Further work is in progress to compute the projection of an input case onto the skeleton of its connected component and see how to use this information to give a global description of the case to the end-user.

References

1. Murthy, S.K.: Automatic Construction of Decision Trees from Data: A Multi-Disciplinary Survey. *Data Mining and Knowledge Discovery*, **2** 4 (1998) 345-389
2. Nelson L.M., Bloch D.A., Longstreth W. T., Shib H.: Recursive Partitioning for the Identification of Disease Risk Subgroups: A Case-Control Study of Subarachnoid Hemorrhage. *Journal of Clinical Epidemiology* 51, 3 (1998) 199-209

3. West, D.: Neural network credit scoring models. *Computers & Operations Research* 27 (2000) 1131-1152
4. Domingos, P.: MetaCost: A General Method for Making Classifiers Cost-Sensitive. *Proc. of the Fifth Int. Conf. on K.D. and Data Mining* (1999) 155-164
5. Sigillito, V., Blake, C.L., Merz, C.J. : Pima Indian Diabetes. UCI Repository of machine learning databases. University of California, Irvine (1990).
6. Burges, C.: A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery Vol. 2*, 2 (1998) 955-974
7. Platt, J.: Probabilistic outputs for support vector machines. In: Smola, A.J., Bartlett, P., Schoelkopf, B., Schuurmans, D. (eds.): *Advances in Large Margin Classifiers*. MIT Press (2000) 61-74
8. Domingos, P.: A Unified Bias-Variance Decomposition and its Applications. *Proc. of the 17th Int. Conf. on Machine Learning: Morgan Kaufmann* (2000) 231-238
9. Kohavi, R.: A study of cross-validation and bootstrap for accuracy estimation and model selection. *Proc. Int. Joint Conf. on Artificial Intelligence* (1995) 1137-1143
10. Provost, F., Fawcett, T.: Analysis and visualization of classifier performance: Comparison under imprecise class and cost distributions. *Proc. Third Int. Conf. on Knowledge Discovery and Data Mining*(1997) 43-48.
11. Jordan, M. I., Jacobs, R. A.: Hierarchical mixtures of experts and the EM algorithm. *Neural Computation* 6 (1994) 181-214
12. Umano, M., Okomato, K., Hatono, I., Tamura, H., Kawachi, F., Umezu, S., Kinoshita, J.: *Proc. of the 3rd IEEE Int. Conf. on Fuzzy Systems* (1994) 2113-2118
13. Quinlan, J. R.: *C4.5: Programs for Machine Learning*. Morgan Kaufmann (1993).
14. Breiman, L., Friedman, J. H., Olshen, R. A., Stone, C. J.: *Classification and Regression Trees*. Belmont: Wadsworth (1984)
15. Esposito, F., Malerba, D., Semeraro G.: A comparative analysis of methods for pruning decision trees. *IEEE Trans. on Pattern Analysis and Machine Intelligence* 19(5) (1997) 476-491
16. Cestnik, B.: Estimating probabilities: A crucial task in machine learning. In *Proceedings of the European Conference on Artificial Intelligence* (1990) 147-149
17. Domingos, P., Provost, F.: Well-trained PETs: Improving probability estimation trees. *CDER Working Paper #00-04-IS Stern School of Business* (2000)
18. Zadrozny, B., Elkan, C.: Obtaining calibrated probability estimates from decision trees and naive bayesian classifiers. *Proc. of the 18th Int. Conf. on Machine Learning* (2001) 609-616
19. Kearns, M. J., Ron, D.: Algorithmic stability and sanity-check bounds for leave-one-out cross-validation. *Proc. of the Tenth Conf. on Computational Learning Theory* (1997) 152-162
20. Bauschke, H., Borwein, J.M.: On projection algorithms for solving convex feasibility problems. *SIAM Review* 38, 3 (1996) 367-426.
21. Bennett, K., Blue, J.: Optimal decision trees. Technical Report 214. R.P.I. Math. Science Dept., Troy, NY 12180(1996).
22. Witten, I., Frank, E.: *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementation*. Morgan Kaufmann (2000)