Discovery of Relevant New Features by Generating Non-Linear Decision Trees

Andreas Ittner
Dept. of Computer Science
Chemnitz University of Technology
D-09107 Chemnitz, GERMANY
andreas.ittner@informatik.tu-chemnitz.de

Michael Schlosser
Dept. of Electrical Engineering
Fachhochschule Koblenz
D-56075 Koblenz, GERMANY
schlosser@koblenz.fh-rpl.de

Abstract

Most decision tree algorithms using selective induction focus on univariate, i.e. axis-parallel tests at each internal node of a tree. Oblique decision trees use multivariate linear tests at each non-leaf node. One well-known limitation of selective induction algorithms, however, is its inadequate description of hypotheses by task-supplied original features. To overcome this limitation this paper reports a novel approach to constructive induction, called non-linear decision trees. The crux of this method consists of the generation of new features and the augmentation of the original primitive features with these new ones. This method can be considered as a powerful tool in KDD, because the constructed new features remain understandable and permit an interpretation by experts. The resulted non-linear decision trees are more accurate than their axis-parallel or oblique counterparts. Experiments on several artificial and real-world data sets demonstrate this property.

Introduction

One well-known limitation of selective induction algorithms is its inadequate description of hypotheses by task-supplied primitive features. To overcome this limitation, constructive induction algorithms transform the original feature space into a more adequate space by creating new features and augmenting the primitive features with the new ones. This method can be considered as a powerful tool in Knowledge Discovery and Data Mining (KDD), if the new features "... may be interpreted as useful or interesting knowledge" (Fayyad et al. 1996).

This paper introduces an approach for discovery relevant new features by generating non-linear decision trees (NDT) (Ittner 1995). This kind of decision trees is based on the augmentation of the feature space. Section 2 (The Problem) is dedicated to the problem of feature construction and state of the art solutions. Section 3 (Manufacturing New Features) elaborates the field of manufacturing new features. This is the key idea underpinning the discovery of relevant new features by a non-linear decision tree method. The fourth section (NDT) deals with a comparison of different kinds of decision trees. Moreover, we introduce our novel method for non-linear decision tree generation with respect to feature construction. Results of using this method to classify several real-world and one artificial data sets are presented in section 5 (Experiments). Section 6 (Conclusions) summarizes the lessons learned from these experiments.

The Problem

Good representations are crucial for solving difficult problems in the fields of Artificial Intelligence (AI) as well as in KDD. Feature construction and the extraction of constructed new features are essential steps to achieve this goal. But what does feature construction mean? The following definition of it was stated in (Matheus & Rendell 1989):

> **Feature Construction:** the application of a set of constructive operators $\{o_1, o_2, ..., o_n\}$ to a set of existing features $\{f_1, f_2, ..., f_m\}$ resulting in the construction of one or more new features $\{f'_1, f'_2, ..., f'_N\}$ intended for use in describing the target concept.

The construction of a new feature may be regarded as a combination of existing features, depending on kind of existing features.

The investigation of all combinations of features is a means used to construct a subset of the H most important features from the h possible ones. The number of these combinations is $\binom{h}{H}$. It is obvious that this method is not applicable to practical problems if the feature space is of high dimension. For that reason combinations of features are limited to pairwise or only few combinations and to simple arithmetical operations, like addition, subtraction, multiplication and division up to now.

There are representative examples of systems that perform and employ a variety of feature construction techniques. For instance the system BACON (Langley et al.1984) focuses on the discovery of empirical laws that summarize numerical data. In order to achieve this goal, BACON requires some information about the form that plausible laws may take. The technique used in ABACUS (Falkenhainer & Michalski 1990) depicts quantitative discovery as a search through the space of equations that could possibly describe the behavior of the observed data. This search process mathematically combines variables representing terms to form new terms. For example x and y might be combined to form $x + y$. But also in the field of modeling, producing new features plays an important role. If we consider the problem of classifying the chessboard positions, for example, formed by randomly placing the three pieces White king, White rook and Black king as 'illegal' or 'legal' (Michie et al. 1994). One important step here is to augment the six features (rank and raw of each piece) with fifteen new ones, generated by forming all possible pairwise differences among the original six. In this way it is possible to express in a decision-tree language certain key sub-description, such as crucial same-file and same-rank relations between White rook and Black king.

Manufacturing New Features

In the field of classification there exist many examples of feature construction. Especially in an exploratory study, practitioners often combine features in an attempt to increase the descriptive power of the resulting decision tree/rules (Michie et al. 1994). Data set providers often think that particular combinations, like the sum of two features $x + y$ or ratios like $\frac{x}{x+y}$ are potentially more useful and important than each feature separately.

Background knowledge of a domain is often helpful in determining what combination of the primitive features to use. In the well-known Iris data set (see section Experiments), for example, the product of the features *F3=Petal Length* and *F4=Petal Width* gives a single feature which has the dimension area, and might be labeled as *Petal Area* (Michie et al. 1994). In this case, the single feature *Petal Area* is a basis for a decision rule that produces only four errors in the whole data set. The notion "area", as a the product of length and width, is well-understood in geometry and can be viewed as a new quality in describing the underlying concept of data. Because "... feature construction is a difficult and poorly understood problem" (Matheus & Rendell 1989) one solution is to have a system to construct new features automatically. One approach, for example, consists of the pairwise generation of the new features from the primitive ones. After the feature construction we can use a selective induction method, for example a decision tree algorithm to evaluate these new features. In the case of Iris data the decision tree, based on the originally existing features, is shown in Figure 1.

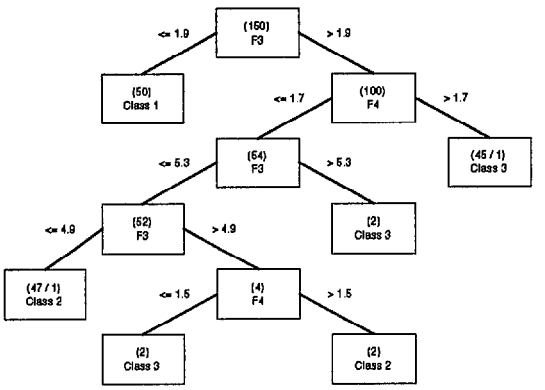

Figure 1: Decision tree of the Iris data set (based on the originally existing primitive features)

After the construction of the pairwise products of the given primitive features and the augmentation to the primitive ones we obtain the following tree [Figure 2].

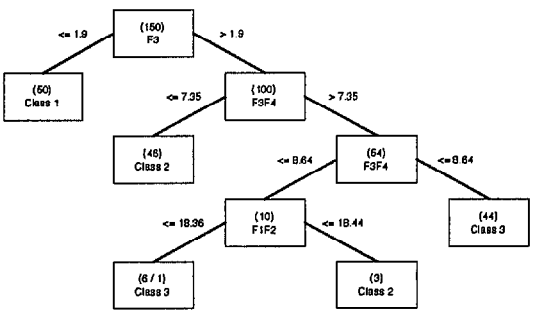

Figure 2: Decision tree of the Iris data set (based on the primitive and new features)

This one is a decision tree with linear or non-linear tests at each internal node. Figure 3 shows the linear and non-linear separations of examples from the three different classes (+,-,x) in the feature space (*F3-F4*-space).

In this case, the simple combination of the original features is the source of power for the resulting decision tree (confer the size of the trees and the expressive power of its internal tests in Figure 1 and 2). Except that, we obtain new ultimately understandable

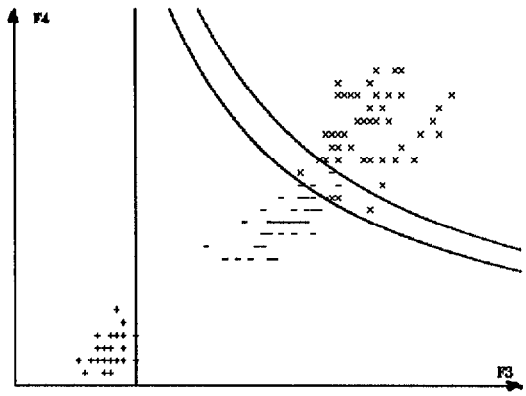

Figure 3: Iris data (linear and non-linear separation)

features (*F3F4* and *F1F2*) that permit an interpretation by the data set provider. In the spirit of KDD this means "... to make patterns understandable to humans in order to facilitate a better understanding of the underlying data" (Fayyad et al. 1996).

The approach of constructive induction algorithms is not limited to continuous-valued features. There are also many methods for the domain of binary and nominal features. The algorithms FRINGE and GREEDY3 (Pagallo & Haussler 1989) create new Boolean features only by using logical operators to construct new features that are more adequate to describe hypotheses. ID2-of-3 (Murphy & Pazzani 1991) creates M-of-N representations as new Boolean features. X-of-N (Zheng 1995) can be considered as an extension of M-of-N, that constructs new nominal features. The common main advantage of all constructive induction algorithms lies in the stronger expressive power of a concept found by these algorithms. In the following section we describe a method to discover relevant new features as combinations of the continuous primitive ones.

Non-Linear Decision Trees

A decision tree algorithm is an approach of selective induction. This section deals with different decision tree paradigms.

Decision trees have been used for classification since the 1980's. Breiman's work on CART (Breiman et al. 1984) and Quinlan's work (Quinlan 1983), (Quinlan 1986) on ID3 and C4.5 provided the foundations for what has become a large field of research on one of the central techniques of machine learning. Originally decision trees were proposed for classification in domains with symbolic-valued features (Quinlan 1986). Later Quinlan extended them to numeric domains (Quinlan 1993), where the tests have the form $x_i > t$, where x_i is one feature and t is a constant, namely the cut-point of this feature. Consequently, this binarization can be viewed as a special case of feature construction and an essential requirement for the following feature selection.

This kind of decision trees may be called *univariate* or *axis-parallel*, because the tests on each non-leaf node of the tree are equivalent to axis-parallel hyperplanes in the feature space [Figure 6]. Another class of decision trees tests a linear combination of the features at each internal node (Breiman et al. 1984), (Utgoff & Brodley 1991), (Murthy et al. 1993). This kind is called *multivariate linear* or *oblique* decision tree, because these tests are equivalent to hyperplanes at an oblique orientation to the axes of the feature space [Figure 7]. Note that axis-parallel decision trees produce partitionings of the feature space in form of hyper-rectangles that are parallel to the feature axes, while oblique decision trees produce polygonal partitionings of the feature space. In contrast to these both techniques, our approach, called *non-linear multivariate* decision tree, produces partitionings in form of a curved hypersurface, namely a hypersurface of the second degree [Figure 8].

The novel method introduced now is based on the combination of primitive features and the augmentation of the feature space before tree generation. For example, as a result of combination of the primitive features $f_1 = x_1$ and $f_2 = x_2$ we see a new feature $f_1' = o_1(f_1, f_2) = x_1 o x_2$ as a new dimension in the feature space [Figure 4].

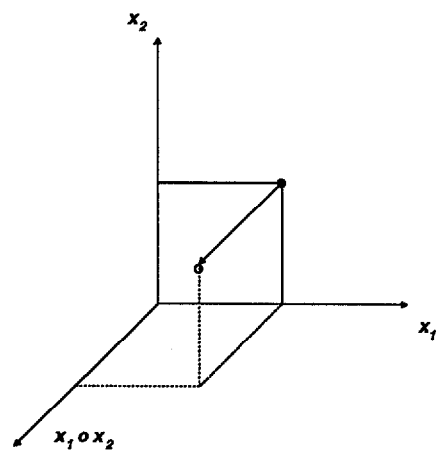

Figure 4: Augmentation of the Feature Space

Because the space of possible new features is exponential, we constrain ourselves to a special kind of feature combination. The key idea is the construction of all possible pairwise products and squares of n numerical primitive features. That means that we use only multiplication as a constructive operator. As a result

we obtain $\frac{n^2+3n}{2}$ features. That is the sum of n primitive ones, n squared ones and $\frac{n(n-1)}{2}$ pairwise products of primitive features.

The second applied constructive operator (addition) in this case is a result of linear combinations of these terms. This operator application to the linear terms, products and squares is not explicitly determined by the user. In contrast, the linear combinations result automatically by a decision tree algorithm (see below).

The linear combination of the constructed terms form an equation of a hypersurface of the second degree. For example, in the two-dimensional case, the form of an equation of a curve of the second degree is:

$$ax_1^2 + 2bx_1x_2 + cx_2^2 + 2dx_1 + 2ex_2 + f = 0.$$

An ellipse, a circle, a hyperbola, a parabola, and a pair of two lines are described by this equation. In the m-dimensional case ($m > 2$) we see elliptoids, hyperboloids, paraboloids and so on.

Now we use a decision tree algorithm, in our case OC1 (Murthy et al. 1993), to construct an oblique decision tree in the new created higher-dimensional feature space. This algorithm generates hyperplanes at an oblique orientation as a test of a linear combination of primitive and new created features at each internal node. In general these hyperplanes correspond to non-linear hypersurfaces in the original feature space of primitive features.

If we consider a two-dimensional feature space with the original features x_1 and x_2, for example, the number of dimensions of the new feature space is five. We obtain two renamed features $y_1 = x_1$, $y_2 = x_2$ and three constructed ones, $y_3 = x_1x_2$, $y_4 = x_1^2$ and $y_5 = x_2^2$. Now it is possible to generate an oblique decision tree in this y-feature space. As a result we obtain an oblique decision tree that is equivalent to a non-linear decision tree in the original x-space after a re-transformation of the features.

In the next section, we present empirical studies, using artificial and real-world data sets, that analyze the ability of our approach to construct non-linear decision trees that are more accurate than their axis-parallel or oblique counterparts.

Experiments

We present results of experiments we performed creating NDT on four real-world and one artificial data sets. The results are summarized in Tables 1-3. We compare the accuracies, the number of leaves and the depths of the trees with axis-parallel (C4.5-like) and oblique (OC1) decision trees. The best results are highlighted in the following tables.

Table 1: Accuracy of the Trees (%)

	iris	diab	heart	vehic	spiral
C4.5-like	93.33	72.92	75.93	71.87	56.25
OC1-two	96.00	71.88	75.19	69.03	53.12
OC1-gain	94.67	72.27	77.78	71.04	46.35
OC1-gini	**96.67**	73.18	76.67	68.32	43.75
NDT-two	96.00	**75.00**	**78.15**	72.70	79.69
NDT-gain	**96.67**	73.96	77.41	72.34	**81.25**
NDT-gini	**96.67**	73.18	77.41	**73.76**	76.56

Table 2: Number of Leaves of the Trees

	iris	diab	heart	vehic	spiral
C4.5-like	3.1	12.8	**4.0**	52.4	28.7
OC1-two	3.2	10.6	4.9	51.1	13.0
OC1-gain	**3.0**	11.2	4.8	54.6	17.2
OC1-gini	3.2	8.8	5.6	35.5	12.1
NDT-two	3.2	9.9	6.5	38.2	**8.0**
NDT-gain	**3.0**	13.9	5.3	40.7	9.7
NDT-gini	**3.0**	**5.9**	9.9	**31.7**	9.5

We used the OC1 algorithm with different impurity and goodness measures to construct an axis-parallel tree (C4.5-like) in the original feature space and oblique decision trees in the original and in the new created higher-dimensional feature space, respectively. The measures are the Twoing Rule (two), the Gini-Index (gini), and the Gain Criterion (gain). They are exhaustively described in (Murthy et al. 1994). All our experiments used 10-fold cross-validation trials. Table 4 summarizes the time of computation for the tree generation. The increasing time of computation can be considered as one weakness of our approach.

Data Sets

We used several well-known data sets for our experiments. The data sets are Iris data (iris), Diabetes

Table 3: Depth of the Trees

	iris	diab	heart	vehic	spiral
C4.5-like	2.1	5.9	2.4	13.8	15.7
OC1-two	2.2	4.9	2.6	10.7	6.8
OC1-gain	**2.0**	5.9	**2.1**	14.5	13.1
OC1-gini	2.2	4.2	2.6	**9.6**	8.1
NDT-two	2.2	4.9	2.8	13.3	**6.0**
NDT-gain	**2.0**	5.7	2.4	16.7	8.1
NDT-gini	**2.0**	**3.6**	3.9	12.5	7.5

Table 4: Time of Computation (sec.)

	iris	diab	heart	vehic	spiral
OC1	110	2526	620	5891	382
NDT	192	6766	939	10742	373

Diagnosis (diab), Heart Disease (heart), and Vehicle Silhouettes (vehic). The artificial data set is the Spiral Data Set (spiral).

Spiral Data Set (spiral). This is an artificial data set, which offers the opportunity to demonstrate the ability of non-linear partitioning of feature space exemplarily. Each of the 192 examples is described by two features. Each one belongs to one of two categories. All examples of a class form a spiral in the 2D feature space [Figure 5]. As Figure 5 shows, the examples of the two categories of the artificial spiral data set are quite hard to separate from each other. An axis-parallel, i.e. univariate partitioning of the feature space, is shown in Figure 6.

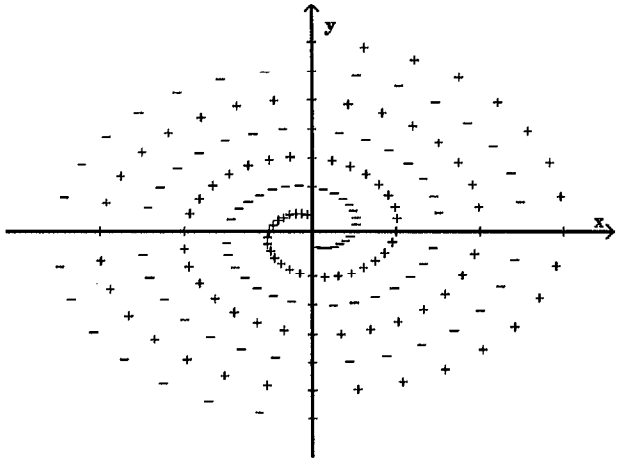

Figure 5: Spiral Data Set

Figure 7 gives an illustration of a multivariate linear partitioning of the feature space with the remarkably number of 38 'hyperplanes'. The oblique decision tree was generated by the OC1 algorithm. Figure 8 shows the non-linear partitioning of the feature space with only 11 curves of the second order. As a special kind of these curves we see two axis-parallel partitioning of the feature space. The corresponding decision tree is a non-linear decision tree that tests a non-linear combination of the original primitive features at each internal node. These tests correspond to tests of a linear combination of the new created features, which can

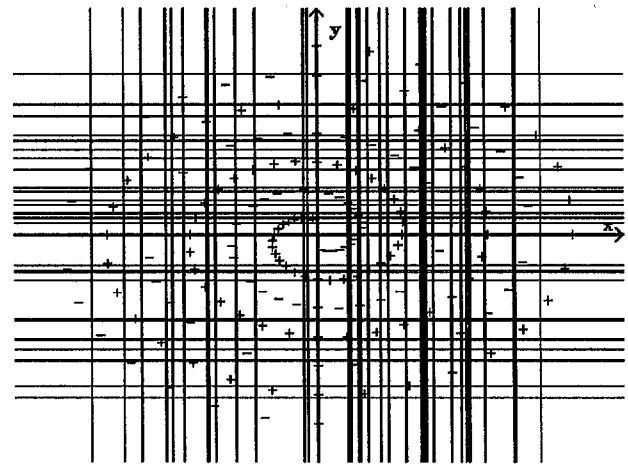

Figure 6: Axis-Parallel Partitioning

be considered as the axes of a higher-dimensional new feature space.

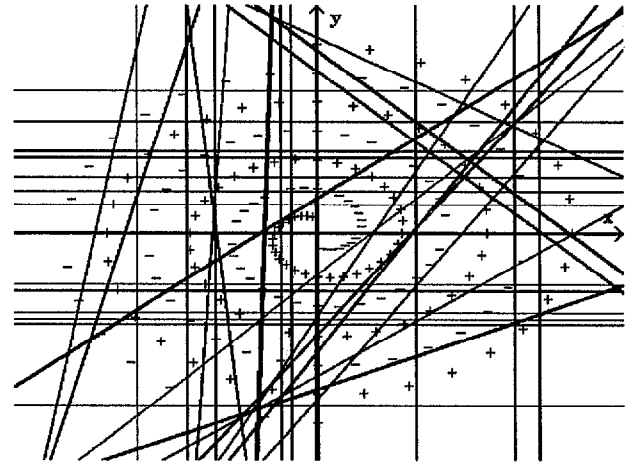

Figure 7: Oblique Partitioning

Conclusions and Further Work

This paper has described a new approach for constructing non-linear decision trees. The ability to produce non-linear splits at each internal node broadens the capabilities of decision tree algorithms and contributes to the society of KDD methods. Our experiments presented here are a convincing demonstration of the usefulness of non-linear separations with respect to the accuracy and the descriptive power of the underlying concept of data. The simple combination of primitive features to new ones gives the opportunity of an oblique partitioning in the higher-dimensional feature space. This oblique partitioning corresponds to a non-

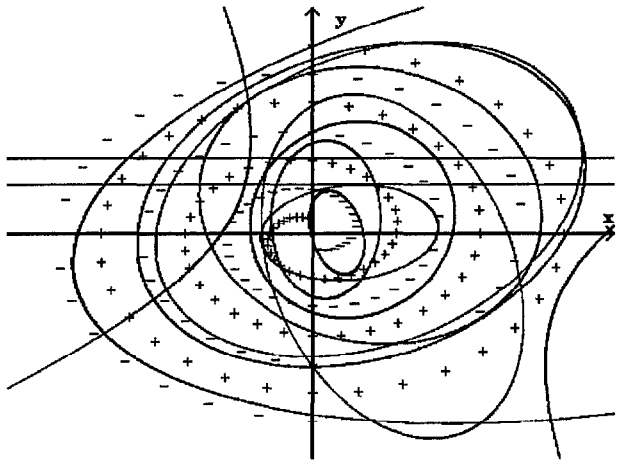

Figure 8: Non-Linear Partitioning of the Space

linear partitioning of the primitive feature space. Our approach fulfills a main goal of KDD to construct new features (patterns), which are easy to understand and to interpret by experts.

The empirical studies have demonstrated that non-linear decision tree algorithms produce more accurate trees than their axis-parallel or oblique counterparts. We plan to extend our experiments with non-linear decision trees to an inductive generation of new features by gradually increasing of the complexity of these ones.

Acknowledgments. The authors thank Werner Dilger, Rainer Staudte and Sarah Malloy for providing comments. Thanks also to two anonymous reviewers for helpful suggestions.

References

Breiman, L., Friedman, J. H., Olshen, R. A. & Stone, C. J. (1984). *Classification and Regression Trees*, Wadsworth International Group.

Brodley, C. E., Utgoff, P. E. (1995). Multivariate Decision Trees, In *Machine Learning*, **19**, 45-77.

Falkenhainer, B. C., Michalski, R. S. (1990). Integrating Quantitative and Qualitative Discovery in the ABACUS System. In Y. Kodratoff, R. S. Michalski (eds.), *Machine Learning - An Artificial Intelligence Approach*, Vol. 3, Morgan Kaufmann.

Fayyad, U. M., Piatetsky-Shapiro, G., Smyth, P. (1996). From Data Mining to Knowledge Discovery: An Overview., In Fayyad, U. M., Piatetsky-Shapiro, G., Smyth, P. & Uthurusamy R. (eds.), *Advances in Knowledge Discovery and Data Mining.*, AAAI/MIT Press.

Ittner, A. (1995). *Ermittlung von funktionalen Attributabhängigkeiten und deren Einfluß auf maschinelle Lernverfahren*, Diplomarbeit, TU Chemnitz-Zwickau.

Langley, P., Bradshaw, G. L., Simon, H. A. (1984). Rediscovering Chemistry with the BACON System, In R. S. Michalski, J. G. Carbonell & T. M. Mitchell (eds.), *Machine Learning: An Artificial Intelligence Approach*, Morgan Kaufmann, San Mateo, CA.

Matheus, C. J., Rendell, L. A. (1989). Constructive Induction on Decision Trees. In *Proceedings of the 11th International Joint Conference on Artificial Intelligence (IJCAI-89).*, Morgan Kaufmann.

Michie, D., Spiegelhalter, D. J., Taylor, C. C. (eds.) (1994). *Machine Learning, Neural and Statistical Classification*, Ellis Horwood.

Murphy, P. M., Pazzani, M. J. (1991). ID2-of-3: Constructive induction of M-of-N concepts for discriminators in decision trees, In *Proceedings of the 8th International Machine Learning Workshop*, Morgan Kaufmann.

Murthy, S., Kasif, S., Salzberg, S., Beigel, R. (1993). OC1: Randomized induction of oblique decision trees, In *Proceedings of the 11th National Conference on Artificial Intelligence (AAAI-93)*, MIT-Press.

Murthy, S., Kasif, S., Salzberg, S. (1994). A System for Induction of Oblique Decision Trees, In *Journal of Artificial Intelligence Research*, Vol 2, Morgan Kaufmann.

Pagallo, G., Haussler, D. (1989). Two algorithms that learn DNF by discovering relevant features, In *Proceedings of the 6th International Machine Learning Workshop*, Ithaca N.Y., Morgan Kaufmann.

Quinlan, J. R. (1983). Learning efficient classification procedures and their application to chess end games, In R. S. Michalski, J. G. Carbonell & T. M. Mitchell (eds.), *Machine Learning: An Artificial Intelligence Approach*, Morgan Kaufmann, San Mateo, CA.

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

Quinlan, J. R. (1993). *C4.5: Programs for Machine Learning*, Morgan Kaufmann, San Mateo, CA.

Utgoff, P. E., Brodley, C. E. (1991). *Linear Machine Decision Trees*, COINS Technical Report 91-10, Dept. of Computer Science, University of Massachusetts.

Zheng, Z. (1995). Constructing Nominal X-of-N Attributes, In *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, Morgan Kaufmann.