2014 International Joint Conference on Neural Networks (IJCNN)
July 6-11, 2014, Beijing, China

Learning Optimization for Decision Tree Classification of Non-categorical Data with Information Gain Impurity Criterion

K.I. Sofeikov, I. Yu. Tyukin, A.N.Gorban, E.M.Mirkes, D.V. Prokhorov, and I.V.Romanenko

***Abstract*—We consider the problem of construction of decision trees in cases when data is non-categorical and is inherently high-dimensional. Using conventional tree growing algorithms that either rely on univariate splits or employ direct search methods for determining multivariate splitting conditions is computationally prohibitive. On the other hand application of standard optimization methods for finding locally optimal splitting conditions is obstructed by abundance of local minima and discontinuities of classical goodness functions such as e.g. information gain or Gini impurity. In order to avoid this limitation a method to generate smoothed replacement for measuring impurity of splits is proposed. This enables to use vast number of efficient optimization techniques for finding locally optimal splits and, at the same time, decreases the number of local minima. The approach is illustrated with examples.**

I. INTRODUCTION

Decision trees have long been known as a viable and efficient tool for solving various classification problems (see e.g. [6], [11], [12], [8] references therein). Popularity of decision trees as classifiers is perhaps best explained in terms of the combination of properties they offer, including self-explanatory final classifiers, ability to handle various types of data, dealing with incomplete and corrupted data, no need for any prior models of date, and easily interpretable decision tests [6].

Despite these advantages there are few technical issues preventing successful induction of decision trees in practice. According to [6] these are: most algorithms require discrete valued target attributes, over-sensitivity to training sets, and issues (both at the level of learning and performance) related to standard univariate split criteria.

Huge body of work exists to date extending classical univariate splits [11] to the multivariate ones. Oblique decision trees [9], perceptron [14] and neural network based trees [4], [5] are few examples of such generalizations. Notwithstanding success of these results in a range of problems, learning in these trees presents a hard computational problem [9]. This is particularly true for cases in which impurity measures, such as e.g. information gain, Gini impurity etc., reflecting homogeneity of data within a given subset is used as metrics to decide which split of the data is best. Contributing to resolving the issue of computational complexity of learning in trees with multivariate splits is the main focus of this article.

We begin with presenting classical decision tree inducing algorithms in Section II. In Section II-E we discuss a common feature of these algorithms which obstructs efficient use of conventional gradient-based optimization techniques to derive univariate and multivariate (locally) optimal splitting criteria. Section III presents main results of the article, Section IV contains illustrative examples, and Section V concludes the paper.

II. BASIC APPROACHES FOR BUILDING DECISION TREES

In what follows we will assume that the data includes vectors of attributes, $x = (x_1, \ldots, x_n)$, with j-th attribute x_j taking values in $\mathbb{R}$, and classes c from a discrete and finite set of classes C; $(x_{1,i}, \ldots, x_{n,i})$, $i = 1, \ldots, N$ will denote elements from the training set, and S is the initial (bounded) set from which the training set is drawn. The overall organization of data assumes the following form:

$$\chi = (\mathbf{x}, c) = (x_1, x_2, \ldots, x_n, c), \qquad (1)$$

K.I. Sofeikov is with the University of Leicester, Department of Mathematics, UK, and with Apical Ltd, Apical Technical Centre, Leicester, UK (e-mail: sofeykov@gmail.com)

I. Yu. Tyukin is with the University of Leicester, Department of Mathematics, UK and with Saint-Petersburg State Electrotechnical University, Russia (e-mail: I.Tyukin@le.ac.uk)

A.N. Gorban is with the University of Leicester, Department of Mathematics, UK (e-mail: ag153@le.ac.uk)

E.M. Mirkes is with the University of Leicester, Department of Mathematics, UK (e-mail: em322@le.ac.uk)

D.V. Prokhorov is with Toyota Technical Centre, Ann Arbor, MI, USA (e-mail: dvprokhorov@gmail.com)

I. Romanenko is with Apical Ltd, Apical Technical Centre, Leicester, UK (e-mail: ilya@apical.co.uk)

978-1-4799-1484-5/14/$31.00 ©2014 IEEE

A. ID3

The process of constructing a decision tree with $ID3$ [11] can be briefly described as follows. For each attribute x_j we introduce a set of thresholds $\{t_{j,1}, \ldots, t_{j,M}\}$ that are equally spaced in the interval $[\min x_j, \max x_j]$. With each threshold $t_{j,k}$ we will associate two subsets $S^+(t_{j,k}) = \{x \in S |\ x_j \geq t_{j,k}\}$ and $S^-(t_{j,k}) = \{x \in S |\ x_j < t_{j,k}\}$. It is clear that $S = S^+(t_{j,k}) \cup S^-(t_{j,k})$, and in this sense thresholds $t_{j,k}$ split the original set S into two disjoint subsets with respect to the values of attribute x_j. All points in the training set are supposed to be already correctly classified into classes c from the set of admissible classes C. Furthermore, the following statistical characterizations of the training set are supposed to be readily available: 1) $|S|$, $|S^+(t_{j,k})|$, $|S^-(t_{j,k})|$ – the total numbers of elements in the sets S, $S^+(t_{j,k})$, and $S^-(t_{j,k})$; 2) $p(c, S)$, $p(c, S^+(t_{j,k}))$, and $p(c, S^-(t_{j,k}))$ – the ratios of the number of elements from S, $S^+(t_{j,k})$, and $S^-(t_{j,k})$ classified as from class c to the total number of elements in S, $S^+(t_{j,k})$, and $S^-(t_{j,k})$ respectively.

For the sets S, $S^+(t_{j,k})$, and $S^-(t_{j,k})$ defined above we introduce quantities specifying variability of classes within these sets. In this case standard Shannon entropy [15] is used:

$$\begin{array}{c} H(S) = -\sum_{c \in C} p(c, S) \log_2 p(c, S) \\ H(S^+) = -\sum_{c \in C} p(c, S^+) \log_2 p(c, S^+) \\ H(S^-) = -\sum_{c \in C} p(c, S^-) \log_2 p(c, S^-) \end{array} \quad (2)$$

Obviously, if e.g. $H(S^+(t_{j,k})) = 0$ (or $H(S^-(t_{j,k})) = 0$) then the set $S^+(t_{j,k})$ (or $S^-(t_{j,k})$) contains objects of only one class. Finally, we specify conditional entropy $H(S|t_{j,k})$

$$\begin{array}{c} H(S|t_{j,k}) = \dfrac{|S^+(t_{j,k})|}{|S|} H(S^+(t_{j,k})) \\ + \dfrac{|S^-(t_{j,k})|}{|S|} H(S^-(t_{j,k})), \end{array} \quad (3)$$

and relative information gain $RIG(S|t_{j,k})$

$$RIG(S|t_{j,k}) = (H(S) - H(S|t_{j,k}))/H(S). \quad (4)$$

Algorithm 1: The algorithm for constructing binary decision trees can now be described as follows:

1) Consider initial set S
2) Create a set of thresholds $\{t_{j,k}\}$
3) For every $t_{j,k}$ calculate $RIG(S|t_{j,k})$
4) Determine

$$t_{l,m} = \arg \max_{j=1,\ldots,n; k=1,\ldots,M} RIG(S|t_{j,k})$$

5) Create a node with attribute x_l being a decision variable, and $x_l < t_{l,m}$, $x_l \geq t_{l,m}$ being its corresponding branching conditions; split the initial set S into two sets $S^+_{l,m}$ and $S^-_{l,m}$
6) Remove $t_{l,m}$ from the list of thresholds and repeat this procedure recursively for each subsequent subsets $S^+_{l,m}$ and $S^-_{l,m}$ until a stopping condition is met

Extensions of $ID3$ such as $C4.5$ and $C5.0$ allow to handle incomplete and "continuous" data, deal with overfitting and improve memory utilization [2].

B. CART and Oblique Decision Trees

Classification And Regression Trees (CART) inducers [1] extend classical $ID3$ algorithm in various directions. In particular they allow for multivariate linear splits (CART-LC) of the original set S in the following form

$$\begin{array}{l} S^+(w) = \{x \in S | \alpha(w, x) > 0\}, \\ S^-(w) = \{x \in S | \alpha(w, x) \leq 0\}, \end{array} \quad (5)$$

where the function α

$$\alpha(w, x) = w_0 + w_1 x_1 + w_2 x_2 + \cdots + w_n x_n, \quad (6)$$

$w = (w_0, \ldots, w_n)$, $w \in \mathbb{R}^{n+1}$, is the corresponding splitting criterion. Finding the values of w in this algorithm requires exploration of the space of parameters in each i-th direction by looking for the values of w_i maximizing goodness of the split $w_0 + w^T x - \delta(w_i + \gamma)$ with respect to δ and three fixed values of $\gamma = \{-1/4, 0, 1/4\}$. The data is assumed to be normalized.

Oblique trees inducers [9] develop the above idea further by introducing random perturbations to avoid local minima after the best values of w_i have been found by direct search.

C. Perceptron Learning Induced Decision trees

Perceptron learning [13] combined with the pocket algorithm [3] has been proposed in [14] as a method for finding linear splits that maximize information gain. As a result of this procedure a decision tree is produced with linear multivariate splits in each node, and the tree is implementable as a multilayered perceptron.

D. Decision trees with neural network feature extraction

Extending the idea of perceptron trees as well as replacing splits (5) with α linear in w, (6), with more general criteria

$$\alpha : \mathbb{R}^p \times \mathbb{R}^n \to \mathbb{R}, \quad (7)$$

leads naturally to configurations know as classification trees with neural network feature extraction [4]. The network output, $NN(w, x)$, is used as the splitting criterion $\alpha(w, x)$. Learning in these trees, however, is not based on information gain or other standard impurity measures. The best values of w are defined as minimizers of $\sum_i (NN(w, x_i) - c_i)^2$.

E. Issues with standard inducers of decision trees

The above brief review of conventional methods for growing of decision trees is by all means non-exhaustive. Yet, this review itself as well as benchmarking of these methods in practice already reveal a few issues of the overall very successful approach that may require improvements.

The first issue is the *computational complexity* of finding (locally) optimal multivariate splits. Even in the case of linear splits, as e.g. in CART-LC inducers and oblique trees, determining the best splitting hyperplane involves coordinate-wise direct search. As the number of data attributes grows or if a finer search grid is required, the sheer amount of computations needed to build a split may become impractically large. Using perceptron feature extraction is an attempt to avoid this difficulty. However, since perceptron rule is not guaranteed to converge for arbitrary data, and that cycling can occur, the method is not ideal. Furthermore, if the cycling occurs, the number of searched splitting hyperplanes is obviously limited. Neural network-base feature extraction uses gradient methods for finding (locally) optimal splits [4]. This drastically reduces computations at each step. Yet, the price for such a reduction is that the split goodness criterion is not impurity-based anymore.

The second issue is *sensitivity* to training data. The latter is best illustrated with an example. Consider, the problem of finding a classifier for data shown in Fig. 1. The data is comprised of points on the real line which are labelled as "class 1" (circles) and "class 2" (crosses). The corresponding information gain is shown as a black solid curve. Notice that the minimal value of conditional entropy is attained at $t_j = 5$. At this point the conditional entropy has a narrow drop which is due to the presence of just a few crosses and circles in its neighborhood. The optimal classifier will therefore be dependent on whether these points are present or absent from the data, and hence the sensitivity.

The third issue is *discontinuity of goodness measures and potential abundance of local minima*. This is related to the previous issue and also can be seen from the figure.

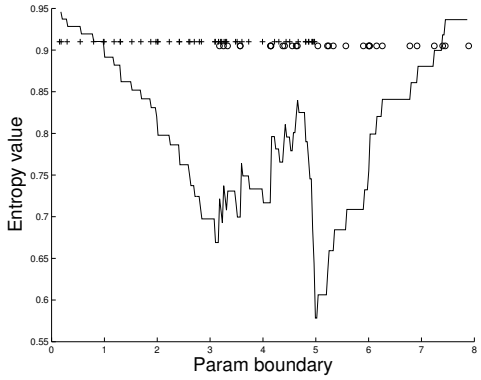

Figure 1: Points that belong to different classes are marked with crosses and circles recpectively. Solid curve is the conditional entropy, $H(S|t_j)$ plotted as a function of the threshold values, t_j.

With respect to computational complexity, a remedy could be to make use of various optimization methods with linear and super-linear convergence (gradient, quasy-Newton, Newton). This, however, is not directly plausible since, as we have already shown, the goodness function is largely discontinuous, with a large number of local minima. While discontinuity per se does not possess severe limitations from optimization point of view, local minima hinder performance of non-gradient alternatives such as the Nelder Mead algorithm [10]. Therefore, in order to enable application of conventional optimization algorithms for efficient finding of splitting criteria in each node, a modification of decision tree inducers is needed.

The issues exemplified above will apparently be alleviated if goodness criteria can somehow be smoothed. This, on one hand, will remove spurious local minima and, on the other hand, will open up a possibility to employ the wealth of conventional optimization methods for finding (locally) optimal splits in each node. The latter will address the complexity issue, and the former will deal with sensitivity and discontinuity.

In what follows we present an approach for deriving smoothed goodness functions for decision tree growing.

III. Decision trees with smoothed impurity-based goodness functions

A. Univariate case

For simplicity, we begin with considering the standard $ID3$ algorithm described in II-A. With each point $\chi_i = (\mathbf{x}_i, c_i) = (x_1^i, \ldots, x_n^i, c_i)$ ($n+1$-tuple) of the original data set we associate an auxiliary integrable and non-negative "smearing" function $f_{\chi_i} : \mathbb{R}^n \to \mathbb{R}_{\geq 0}$.

Even though the choice of specific f_{χ_i} is not discussed in this article few obvious candidates of f_{χ_i} are

$$\begin{aligned}&\text{Gaussian}: \ \frac{1}{\sqrt{(2\pi)^n|\mathbf{\Sigma}|}}e^{\left(-\frac{1}{2}(x-\mathbf{x}_i)^T\mathbf{\Sigma}^{-1}(x-\mathbf{x}_i)\right)},\\ &\text{Inverse multiquadric}: \ \frac{1}{\sqrt{1+(x-\mathbf{x}_i)^T\mathbf{\Sigma}^{-1}(x-\mathbf{x}_i)}}\\ &\text{Delta}-\text{function}: \ \delta(x_1-x_1^i)\cdots\delta(x_n-x_n^i),\end{aligned}$$

where $\mathbf{\Sigma}$ is a positive definite symmetric matrix, and $|\mathbf{\Sigma}|$ is the determinant of $\mathbf{\Sigma}$. Having defined f_{χ_i} we introduce

$$\begin{aligned}D(S) &= \textstyle\int_S \sum_{i=1}^N f_{\chi_i}(x)dx\\ D_c(S) &= \textstyle\int_S \sum_{i=1}^N f_{\chi_i}(x)I_c(\chi_i)dx,\end{aligned}$$

where $I_c(\chi_i)$ is the indicator function:

$$I_c(\chi_i) = \begin{cases} 1, & c = c_i \\ 0, & c \neq c_i \end{cases} .$$

Finally we define $p_f(c,S)$

$$p_f(c,S) = \frac{D_c(S)}{D(S)}$$

$$H_f(S) = -\textstyle\sum_{c\in C} p_f(c,S)\log_2 p_f(c,S)$$

$$\begin{aligned}H_f(S|t_{j,k}) = &\frac{D(S^+(t_{j,k}))}{D(S)}H_f(S^+(t_{j,k}))\\ &+\frac{D(S^-(t_{j,k}))}{D(S)}H_f(S^-(t_{j,k})),\end{aligned}$$

and

$$RIG_f(S|t_{j,k}) = (H_f(S) - H_f(S|t_{j,k}))/H_f(S). \quad (8)$$

Note that $D(S), D_c(S), p_f(c,S) \geq 0$, $\sum_c p_f(c,S) = 1$, $D(S^+(t_{j,k})) + D(S^-(t_{j,k})) = D(S)$. Replacing RIG with RIG_f in Algorithm 1 gives rise to the proposed modification.

The following characterizations of the newly introduced $RIG_f(S|\cdot)$ are immediate

Proposition 1:

P1) Let $f_{\chi_i}(\cdot)$ be piece-wise continuous for all $i \in \{1,\ldots,N\}$, then $RIG_f(S|\cdot)$ is continuous. If f_{χ_i} are continuous then $RIG_f(S|\cdot)$ is differentiable.

P2) Let $f_{\chi_i}(\cdot)$ be the delta-function: $f_{\chi_i}(x) = \delta(x_1 - x_1^i)\delta(x_2 - x_2^i)\cdots\delta(x_n - x_n^i)$, then

$$RIG(S|t_{j,k}) = RIG_f(S|t_{j,k}) \text{ for all } t_{j,k}.$$

P3) Let $f_{\chi_i}(\cdot)$ be the indicator-function 1_S, then

$$RIG_f(S|t_{j,k}) = \text{const for all } t_{j,k}.$$

Properties P2), P3) are straightforward. Property P1 follows from that piece-wise continuity (continuity) implies that $p_f(c,S^+(t_{j,k}))$, $p_f(c,S^-(t_{j,k}))$, $D(S^+(t_{j,k}))$, and $D(S^-(t_{j,k}))$ are continuous (differentiable). Hence so are the functions $H_f(S|t_{j,k})$ (with respect to $t_{j,k}$) and, consequently, $RIG_f(S|t_{j,k})$.

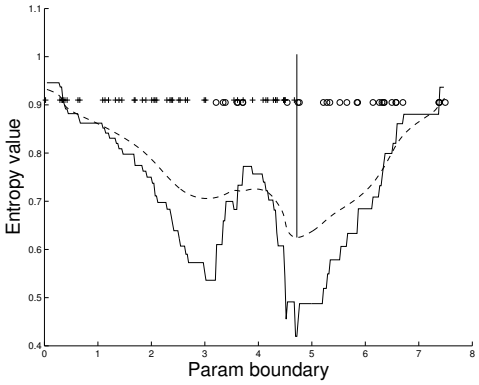

Figure 2: Points that belong to different classes are marked with crosses and circles. Solid curve shows conditional entropy curve. Dotted curve shows smoothed version of the same curve. Solid line that is perpendicular to param boundary axis indicates minimal value for the smoothed continuous version.

According to the Proposition, using "broad" identical f_{χ_i} flattens the shape of $RIG_f(S|\cdot)$, $RIG_f(S|\cdot)$ with f_{χ_i} concentrated at χ_i resembles (in the limit) the shape of $RIG(S|\cdot)$. Figure 2 shows how $H_f(S|\cdot)$, derived for f_{χ_i} Gaussian, compares to $H(S|\cdot)$ for a randomly drawn data sample S (represented by $o, +$ in the figure). The function $RIG_f(S|\cdot)$, obviously, is just a scaled and translated version of $H_f(S|\cdot)$. Note that $H_f(S|\cdot)$ is a quite smooth curve, which agrees with property P1 in Proposition 1. Hence one can use a range of standard optimization methods to infer the optimal values of $t_{j,k}$. It is also clear that $H_f(S|\cdot)$ (and $RIG_f(S|\cdot)$) may still have a number of local minima. These can, however, be addressed by starting optimization procedures from various initial conditions. While this approach may increase the total number of calculations in total, it is generally more advantageous than direct search especially when nominal dimensionality of the data is high. Note that the number of local minima may be controlled by the width of the smearing functions f_{χ_i}. Fig. 3 shows how the shape of $H_f(S|t_{j,k})$, changes with the width of f_{χ_i} for f_{χ_i} Gaussian. The filtering feature of $RIG_f(S|\cdot)$ and $H_f(S|\cdot)$ related to local minima removal is becoming particularly relevant for data sets with large number of attributes. In order to illustrate this point we show in Fig. 4 how $H_f(S|\cdot)$ and $H(S|\cdot)$ look like in the case of a two-attribute data sample.

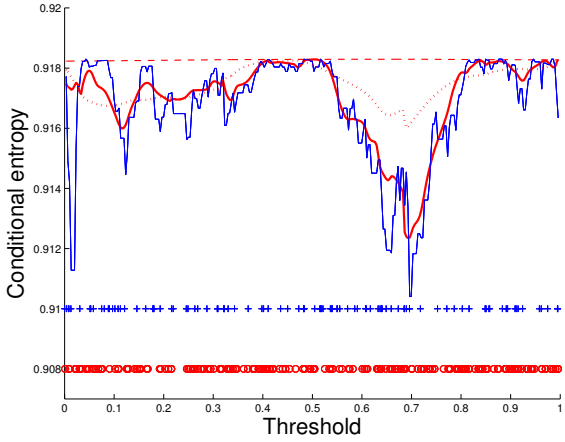

Figure 3: Dependence of $H_f(S|t_{j,k})$ on the width of f_{χ_i} (Gaussian, and σ is the corresponding width parameter). Solid blue curve depicts the original $H(S|t_{j,k})$. Thick red solid curve shows $H(S|t_{j,k})$ for $\sigma = 0.05$. Dotted line corresponds to the case of $\sigma = 0.01$, and dashed line stands for $\sigma = 0.0001$.

B. Multivariate case

The procedure for constructing continuous and differentiable goodness criteria can be straightforwardly extended to general multivariate case. Indeed, consider splitting criterion (5), (7) and let

$$H_f(S|w) = \frac{D(S^+(w))}{D(S)} H_f(S^+(w)) + \frac{D(S^-(w))}{D(S)} H_f(S^-(w)),$$

$$RIG_f(S|w) = (H_f(S) - H_f(S|w))/H_f(S). \quad (9)$$

The following property of $RIG_f(S|w)$ is now immediate.

Proposition 2: Suppose that for every value of w the set

$$\mathcal{A}(w) = \{x \in S | \alpha(x, w) = 0\}$$

is an $n-1$ dimensional manifold, and it is such that that for any $\varepsilon > 0$ there is a $\delta > 0$: $\|w_1 - w_2\| < \delta$ implies that $\max_{x \in \mathcal{A}(w_1)} \mathrm{dist}(\mathcal{A}(w_2), x) < \varepsilon$. Furthermore, let f_{χ_i} be continuous. Then $RIG_f(S|\cdot)$ is differentiable.

IV. Examples

A. A synthetic example

In order check plausibility of the approach we first considered a synthetic data set which is shown by green circles and red crosses in Fig. 5. Circles correspond to "class 1", and crosses correspond to "class 2". The task was to induce a decision tree for classifying the data. We

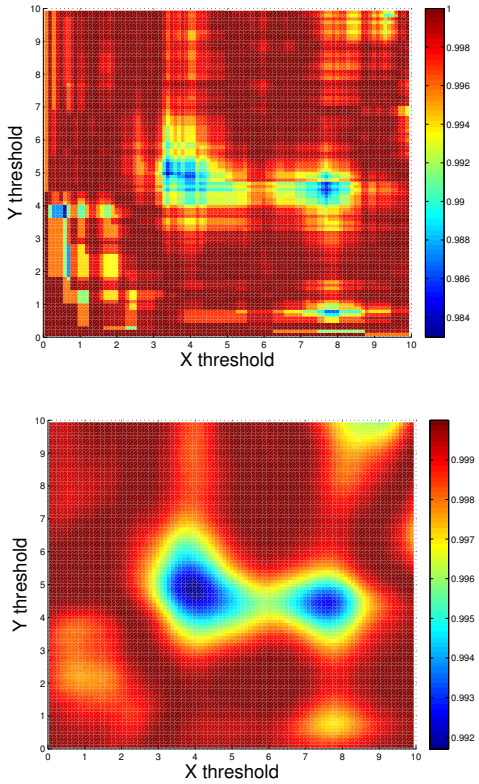

Figure 4: Contour plots of functions $H(S|\cdot)$ (upper panel) and $H_f(S|\cdot)$ (lower panel) for a randomly drawn sample of two-attribute data set. The number of classes is 2. The number of local minima is drastically reduced in the lower panel.

started with the decision tree inducer with multivariate linear splits in each node and $RIG_f(S|w)$ instead of $RIG(S|w)$. The values of w were determined by the Nelder-Mead algorithm. The corresponding linear splits are shown in the figure as blue solid lines. The classifier had the following performance characterisation: the rate of false-false detections is 98%, the rate of positive-positive is 92%, and the overall rate of correct detections is 95%.

Performance of the classifier had been compared with the classifier induced by standard $ID3$ algorithm (the corresponding univariate splitting conditions are shown as red dashed lines in the figure). Stopping conditions were set identical to the multivariate one. The rates of positive-positive and false-false detections for this classifier were 91% and 92% respectively. This shows that the proposed inducer with Nelder-Mead optimization and multivariate splits performs better than $ID3$ algorithm in this task. Note that this advantage did

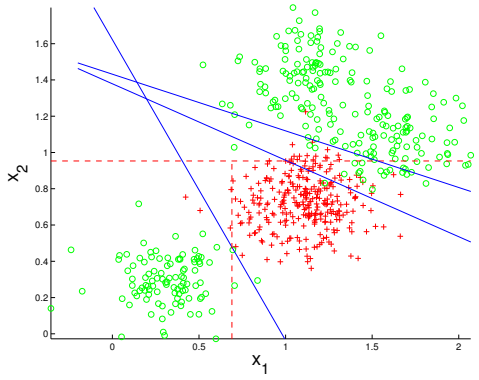

Figure 5: Splitting conditions for the proposed multivariate inducer (solid blue lines) and standard univariate splits of $ID3$ (dashed red lines).

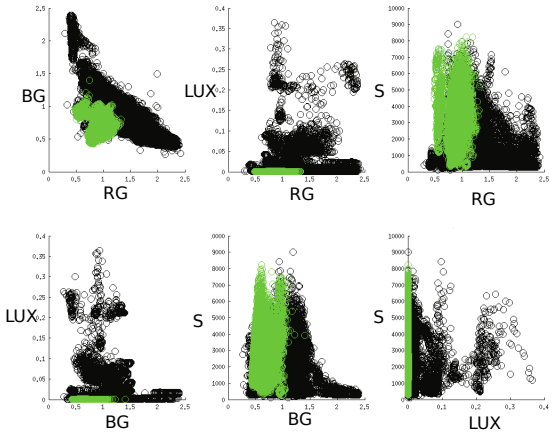

Figure 6: Representation of the dataset on the planes of pairs of attributes. Green circles correspond to patches of grass, black circles mark patches of images without grass.

not have excessive computational cost attached.

B. Multivariate vs Univariate splits

Previous example showed that our method is feasible for construction of multivariate splits. Next we examine what advantages multivariate splits may offer over the univariate ones in a real-life problem. Consider the problem of AWB (Automatic White Balance) setting in challenging light conditions [16]. In this context decision trees are used to detect grass and sky areas in an image. These are then used as clues for color adjustments. In our task the variables available for direct observation have been restricted to: 1) average R/G value, 2) average B/G value, 3) variance of R/G value, 4) variance of B/G value, 5) Illumination of scene (lux value) 6) Intensity variation. Every photo was split into 15×15 pieces and for every such piece we formed a corresponding $6D$ parameter/feature vector. Each training set was built from about 100 photos, and the total number of training $6D$ points was about 22500. Training sets were organized as follows.

The training set was split into two subsets: positive (with grass featuring in the images) and negative (no grass objects in the images). Images in the positive training set contained large amount of different textures of grass and foliage in different light conditions. The size occupied by grass/foliage patches varied from one image to another. The negative training set contained photos with different objects and colors and, at the same time, did not contain any grass/foliage or sharp green real world objects.

Figure 6 shows examples of possible $2D$ projections of clusters of the original $6D$ feature vectors. As we can see from these pictures relationships between individual components of feature vectors corresponding to different clusters is rather complicated, with large overlapping areas in relevant $2D$ projections.

In the original work [16] univariate splitting criteria have been chosen. Results of grass detection in a sample image with this algorithm is shown in 6. As expected, the quality of detection improves when standard univariate splits are replaced with linear multivariate ones (Fig. 6, third column).

C. The benefits of smoothing

In Section III we showed that using smoothed goodness functions $RIG_f(S|\cdot)$, $H_f(S|\cdot)$ may help to reduce the number of local minima whilst looking for (locally) optimal parameterizations of splitting conditions $\alpha(x, w) = 0$. This has been done on randomly drawn data samples. Let us know show how this approach works in the case of real experimental data. We consider the problem of computational diagnosis of canine lymphoma [7]. The problem has been resolved using decision trees inducers involving direct search in the direction derived from linear discriminant analysis (Fisher discriminant). Let us examine if our smoothed $RIG_f(S|\cdot)$, $H_f(S|\cdot)$ may offer an advantage. Fig. 8 shows original (solid thin blue line) $H(S|\cdot)$ and $H_f(S|\cdot)$ for Gaussian f_{χ_i} and for various values of σ. Notice that, remarkably, for a large interval of values of σ the function $H_f(S|\cdot)$ is apparently unimodal. Moreover smoothed goodness functions are not sensitive to presence/removal of few data points from training samples making the process of inducing trees more robust as compared to standard methods. We would like to note

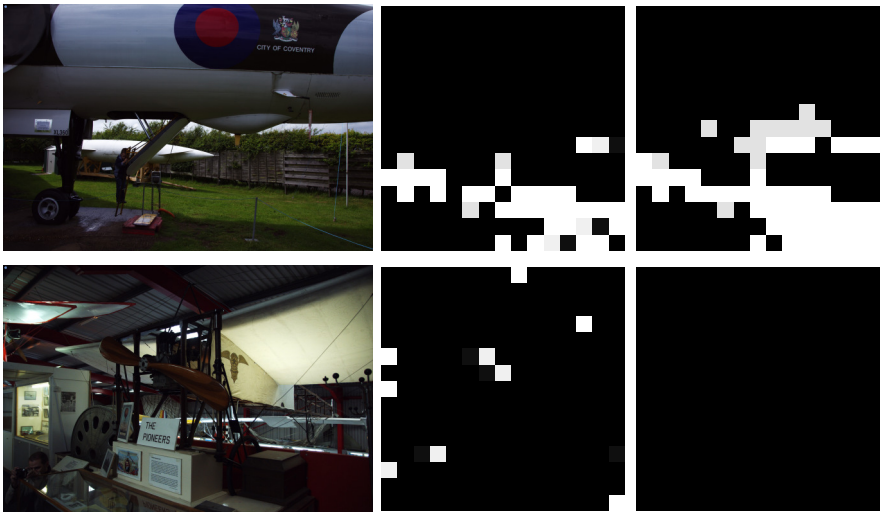

Figure 7: Examples of grass detection. White patches in the second and third columns stand for high grass probability in the current zone. First column from the left : source images. Center column: probability/likelyhood maps of grass obtained with $ID3$ algorithm. Third column: probability/likelyhood maps of grass obtained with using of binary trees with combined attributes in nodes. We may see from the first row of this figure that the number of Positive-Negatives detections was reduced. From the second row we may conclude that the number of Negative-Positive detections has also been reduced. Moreover, we registered these improvement for nearly 75% of all images in our testing set.

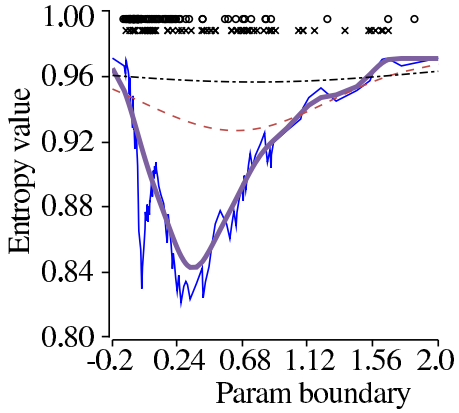

Figure 8: Original and smoothed conditional entropy for canine lymphoma data [7].

that that minima of the smoothed conditional entropy profiles, as well as the corresponding splitting criteria, may differ from those obtained for the non-smoothed conditional entropy (see e.g. Fig. 2). On the other hand, as follows from Proposition 2, the discrepancy can be controlled by proper choice of σ. Investigating how to choose the values of σ in order to keep the balance of precision vs robustness at an optimum is the subject of our future work.

D. Statistical Analysis

In order to assess quality of the proposed classifiers we built several decision trees for the example problem described in Section IV-B and compared their performance to standard $ID3$ algorithm with 10 nodes per each attribute. The data comprised of 60000 of points, and each point had 6 attributes.Training sets in each experiment consisted of 10% of the total number of data points. Remaining 90% of data has been used to validate the model. In order to derive a splitting criterion in each node we run Nelder-Mead algorithm in the original $6D$ space from 5 randomly chosen initial conditions. The best outcome from these 5 runs has been chosen as the splitting criterion for the node. Table I presents typical values of false-positive and false-negative rates observed in these experiments. As we can see from the Table, the overall classification quality tends to be lower for smaller values of σ. This may be explained bt that the smoothed RIG surface approaches the original RIG for σ small. The latter, however, is discontinuous and has many local minima that are not optimal. The Nelder-Mead algorithm stucks in one of these, and hence the procedure results in trees with poorer performance. In addition we compare performance of trees obtained with to that of decision trees resulting from classical $ID3$ for the same problem (last two columns in Table I).

Table I: Examples of False-Positives and False-Negatives scores obtained with suggested approach and $ID3$

New approach						$ID3$	
$\sigma = 0.75$		$\sigma = 0.35$		$\sigma = 0.15$			
FP	FN	FP	FN	FP	FN	FP	FN
0.02	0.1	0.07	0.17	0.2	0.06	0.03	0.07
0.06	0.11	0.02	0.12	0.2	0.05	0.03	0.08
0.1	0.06	0.07	0.17	0.2	0.2	0.13	0.01
0.04	0.07	0.07	0.11	0.14	0.07	0.04	0.05
0.09	0.08	0.03	0.09	0.19	0.17	0.09	0.02
0.09	0.08	0.08	0.18	0.08	0.17	0.11	0.2
0.12	0.13	0.09	0.08	0.05	0.13	0.12	0.01

Notice that since training and validations sets for each experiments have been chosen randomly, the values of False-Positives and False-Negatives vary in each column. In order to provide a clearer performance picture we built random forests for $\sigma = 0.75$ and $ID3$, and used the majority rule in order to work out classification outcomes. Best results for both approaches are shown in Table II. As follows from this Table, performance of forests generated by standard $ID3$ algorithm with 10 nodes per each attribute is higher than that of ours. This, however, comes at a price of exhaustive search. Even though in our example problem computational complexity of $ID3$ did not exceed that of ours the issue of complexity will become more apparent for problems with higher dimensionality of data.

Table II: False-Positives and False-Negatives scores for random forests

New approach		$ID3$	
FP	FN	FP	FN
0.06	0.08	0.03	0.04

V. Conclusion and future work

In this article we discussed and analysed limitations of standard decision tree growing methods for data with large number of attributes. We argue that for a range of impurity measures and discrete training data sets the issue is mostly due to the fact that the impurity measures are discontinuous and are prone to having large number of local minima. In order to relax these limitations a method for smoothing impurity measures is introduced. We show that the resulting impurity measures can always be made continuous and differentiable and, at the same time, they preserve shape of the original impurity. Moreover, in the limit when smoothing kernels (smearing functions) approach delta-functions, our newly generated impurity measure converges to the original one.

The approach is illustrated with an example of construction of a decision tree classifier with linear multivariate splitting condition. We note though, that the proposed modification is not limited to linear cases. It can be applied to neural network based feature selection too. Doing so is the focus of our ongoing work in this direction.

References

[1] L. Breiman, J. Friedman, R. Olshen, and C. Stone. *Classification and Regression Trees*. Wadsworth International Group, 1984.

[2] Margaret H. Dunham. *Data Mining: Introductory and Advanced topics*. Pearson Education, 2003.

[3] S. I. Gallant. Optimal linear discriminants. In *Proc. 8th Int. Conf. Pattern Recognition*, pages 849–852, 1986.

[4] H. Guo and S. Gelfand. Classification trees with neural network feature extraction. *IEEE Trans. on Neural Networks*, 3(6):923–933, 1992.

[5] Jae. H. Yoo Ishwar K. Sethi. Structure-driven induction of decision tree classifiers through neural learning. *Pattern Recognition*, 30:939–947, 1994.

[6] Rokach L. and Maimon O. *Data Mining and Knowledge Discovery Handbook*. Springer, 2010.

[7] E.M. Mirkes, I. Alexandrakis, K. Slater, R. Tuli, and A.N. Gorban. Computational diagnosis of canine lymphoma. *J. Phys.: Conf. Ser. 490 012135*, 2014.

[8] S.K. Murthy. Automatic construction of decision trees from data: a multi-disciplinary survey. *Data mining and knowledge discovery*, 2(4):345–389, 1998.

[9] S.K. Murthy, S. Kasis, and S. Salzberg. A system for induction of oblique decision trees. *Journal of Artificial Intelligence*, 2:1–32, 1994.

[10] R. Mead Nelder, John A. A simplex method for function minimization. *Computer Journal*, 7:308313, 1965.

[11] J. R. Quinlan. Induction of decision trees. *Machine Learning*, 1:81–106, 1986.

[12] L. Rokach and O. Maimon. Top-down induction of decision trees classifiers a survey. *IEEE Trans. on Systems Man and Cybernetics: Part C*, 1(11), 2001.

[13] F. Rosenblatt. The perceptron–a perceiving and recognizing automaton. *Report 85-460-1, Cornell Aeronautical Laboratory*, 1957.

[14] I.K. Sethi and J.H. Yoo. Design of multicategory multifeature split decision trees using perceptron learning. *Pattern recognition*, 27(7):939–947, 1994.

[15] C. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27:379–423, 623–656, 1948.

[16] K. Sofeikov, I. Romanenko, I. Tyukin, and A.N. Gorban. Scene analysis assisting for awb using binary decision trees and average image metrics. In *IEEE Conference on Consumer Electronics*, pages 488–491. 2013.