Advance Access publication on 4 March 2011
doi:10.1093/comjnl/bxr020

Univariate Decision Tree Induction using Maximum Margin Classification

OLCAY TANER YILDIZ*

Department of Computer Engineering, Işık University, TR-34980, Şile, Istanbul, Turkey
**Corresponding author: olcaytaner@isikun.edu.tr*

In many pattern recognition applications, first decision trees are used due to their simplicity and easily interpretable nature. In this paper, we propose a new decision tree learning algorithm called univariate margin tree where, for each continuous attribute, the best split is found using convex optimization. Our simulation results on 47 data sets show that the novel margin tree classifier performs at least as good as C4.5 and linear discriminant tree (LDT) with a similar time complexity. For two-class data sets, it generates significantly smaller trees than C4.5 and LDT without sacrificing from accuracy, and generates significantly more accurate trees than C4.5 and LDT for multiclass data sets with one-vs-rest methodology.

Keywords: statistical learning theory; decision trees

Received 14 June 2010; revised 22 December 2010
Handling editor: Iain Stewart

1. INTRODUCTION

Machine learning aims to determine a description of a given concept from a set of examples provided by teacher and from the background knowledge. Learning examples can be defined as positive or negative for a two-class problem. Background knowledge contains the information about the language used to describe the examples and concepts. For instance, it can include possible values of variables and their hierarchies or predicates. The learning algorithm then builds on the type of examples, on the size and relevance of the background knowledge and on the representational issues [1, 2].

Decision trees are one of the well-known learning algorithms in machine learning. They are tree-based structures which consist of internal nodes having one or more attributes to test and leaves to show the decision made. The type of the split determines the type of the decision tree. In univariate decision trees, the split is based on one attribute. If that attribute is continuous, there will be two children of each internal node (C4.5) [3], if that attribute is discrete, there will be L children of each internal node corresponding to the L different outcomes of the test (ID3) [4]. In multivariate linear decision trees, the split is based on a linear combination of features (CART) [5]. In this paper, we will deal with continuous univariate decision trees.

Linear discriminant tree (LDT) [6] is another univariate decision tree technique which uses a statistical approach to quickly determine the best split. Finding the best split with Fisher's linear discriminant analysis [7] is done as a nested optimization problem. In the inner optimization problem, Fisher's linear discriminant is used for finding a good split for the given two distinct groups of classes. In the outer optimization problem, at each node m, one searches for the best separation of K classes into two groups, C_m^L and C_m^R [8].

Maximum margin classifiers [9], especially support vector machines (SVM), became popular in the recent years for solving problems in classification, regression and one-class classification. Maximum margin classifiers (i) approach the classification problem through the concept of *margin*, which is defined to be the smallest distance between the decision boundary and the closest data points (called support vectors), (ii) determine the model parameters by setting up a convex optimization problem, (iii) use hinge loss instead of misclassification error, and (iv) usually work for two-class problems.

Although this is the first approach using SVM in the univariate decision tree induction, the idea of combining multivariate SVM and decision-tree-like methods is not new. These approaches have been applied for the decomposition and solving multiclass pattern recognition problems with tree-structured SVM. Tibshirani and Hastie [10] propose a tree-based maximum margin classifier, where they search the line that partitions the classes into two groups, that has the maximum margin. Bennett and Blue [11] investigate decision trees with support vector classifiers at each node, but they do not discuss adaptive construction of the tree topology, i.e. the trade-off between the overall tree complexity and the complexity of support vector classifiers at each node remains to be investigated. Vural and Dy [12] propose divide-by-2 approach for class decomposition where they use k-means clustering of the class means to divide the points into two groups at each node, before applying a support vector classifier.

In this paper, we propose a novel decision tree classifier which finds the best split for each attribute at each decision node using convex optimization. Our simulation results on 47 data sets from University of California, Irvine (UCI) repository [13] show that our univariate margin tree (UMT) classifier performs better than C4.5 and LDT in terms of accuracy and tree size.

This paper is organized as follows: in Section 2, we present and discuss our proposed UMT algorithm. We present the experimental setup and results in Section 3 where we compare, in detail, our proposed algorithm with C4.5 and LDT. Section 4 gives the conclusions and discusses possible future directions.

2. UNIVARIATE MARGIN TREE

2.1. Theoretical background

We consider the well-known supervised learning setting where the learning algorithm uses a sample of N labeled points $S = ((\boldsymbol{x}^1, y^1), \ldots, (\boldsymbol{x}^N, y^N)) \in (X \times Y)^N$, where X is the input space and Y the label set, which is $\{-1, +1\}$. The input space X is a continuous vectorial space of dimension d, the number of features. The data pairs $(\boldsymbol{x}^i, y^i)$ are independently and identically distributed according to an unknown but fixed distribution.

In training a multivariate linear test, wide margin classifiers can be used to find the weight vector $\boldsymbol{w} = [w_1, w_2, \ldots, w_d]^{\mathrm{T}}$ and the threshold w_0 that best separates the two classes. In SVM, the classification task is defined as a minimization problem. The distance from the separating hyperplane to the instances closest to it on either side is called margin and the optimal separating hyperplane is the one that maximizes the margin. For the nonseparable case, we require

$$y^t(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}^t + w_0) \geq 1 - \epsilon^t \tag{1}$$

for each data point $\boldsymbol{x}^t$ with output y^t. If $\epsilon^t = 0$, the instance is on the separating hyperplane. If $0 < \epsilon^t < 1$, the instance is correctly classified but it is in the margin. If $\epsilon^t > 1$, it is misclassified. Maximizing margin is equivalent to minimizing $||\boldsymbol{w}||$, and after adding the penalty term $(\sum \epsilon^t)$ to the objective function one gets the following formulation [9]

$$\begin{aligned} \text{Min} \quad & \tfrac{1}{2}||\boldsymbol{w}||^2 + C\sum_t \epsilon^t \\ \text{s.t.} \quad & y^t(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}^t + w_0) \geq 1 - \epsilon^t \\ & \epsilon^t \geq 0 \end{aligned} \tag{2}$$

where C is the penalty factor.

In training a univariate test, one tries to find the best feature x_j and the threshold w_0, namely the split $x_j + w_0 \geq 0$ that best separates two (or K) classes. To separate K classes as good as possible, C4.5 tries to minimize the impurity or maximize the information gain, LDT assumes the two-class groups are normally distributed and tries to maximize the ratio of between class distance to within class distance. In this paper, we take the wide margin approach and express finding best split as a minimization problem. To find the best threshold for feature j, we require

$$y^t(x_j^t + w_0) \geq C - \epsilon^t \tag{3}$$

for each data point x_j^t with output y^t, where $|C|$ is the length of the margin. If $\epsilon^t = 0$, the instance is on the separating line. If $0 < \epsilon^t < |C|$, the instance is correctly classified but it is in the margin. If $\epsilon^t > |C|$, it is misclassified. Since there is only single variable, and its weight is 1, we only need the penalty term $(\sum \epsilon^t)$ in the objective function. Now the formulation is:

$$\begin{aligned} \text{Min} \quad & \sum_t \epsilon^t \\ \text{s.t.} \quad & y^t(x_j^t + w_0) \geq C - \epsilon^t \\ & \epsilon^t \geq 0 \end{aligned} \tag{4}$$

This is a linear programming problem in which ϵ^t and w_0 are variables. By adding Lagrange multipliers ($\alpha^t \geq 0$ and $\mu^t \geq 0$), the problem can be converted to

$$\begin{aligned} L(w_0, \epsilon^t, \alpha^t, \mu^t) = \sum_t \epsilon^t - \sum_t \alpha^t[y^t(x_j^t + w_0) - C + \epsilon^t] \\ - \sum_t \mu^t \epsilon^t \end{aligned} \tag{5}$$

We can remove the primal variables ϵ^t and w_0 by maximization, i.e. set the following derivatives to zero:

$$\begin{aligned} \frac{\partial L}{\partial w_0} = 0 \Rightarrow \sum_t \alpha^t y^t = 0 \\ \frac{\partial L}{\partial \epsilon^t} = 0 \Rightarrow \alpha^t + \mu^t = 1 \end{aligned} \tag{6}$$

If the Equations (6) are plugged in the Equation (5), one gets:

$$D(\alpha^t, \mu^t) = \sum_t \alpha^t(C - y^t x_j^t) \tag{7}$$

Since we have $\alpha^t \geq 0$, $\mu^t \geq 0$ and $\alpha^t + \mu^t = 1$, it follows $0 \leq \alpha^t \leq 1$. Now the dual optimization problem becomes

$$\begin{aligned} \text{Max} \quad & \sum_t \alpha^t(C - y^t x_j^t) \\ \text{s.t.} \quad & \sum_t \alpha^t y^t = 0 \\ & 0 \leq \alpha^t \leq 1 \end{aligned} \tag{8}$$

Neither the primal (Equation (4)) nor the dual (Equation (8)) is easily solvable. To solve the problem, we search all possible solutions in terms of the split point w_0 exhaustively. The inequality $y^t(x_j^t + w_0) \geq C - \epsilon^t$ can be written as

$$\epsilon^t \geq C - x_j^t y^t - w_0 y^t \qquad (9)$$

According to the convex optimization theory, the solution to a convex optimization problem (if there is a solution) lies in one of the vertices of the convex polytope and each vertex is specified by a set of $n+1$ inequalities tight, where $n+1$ represents the number of distinct variables. Setting the left side of the Equation (9) to zero ($\epsilon^i = 0$) gives us all possible values for $w_0 = (C - x_j^t y^t)/y^t$. Then the only remaining thing is to calculate other ϵ^j's and check for the maximum value of the objective function $\sum_t \epsilon^t$.

2.2. The algorithm

The pseudocode for finding the best split at each decision node of the UMT is given in Fig. 1. For each feature i we search for the optimal w_0 exhaustively (Line 3). Given the feature i, there exists at most N different w_0's corresponding to N different

```
Split UnivariateMarginTreeBestSplit($N$, $d$, $S$, $V$)
1  bestError = $+\infty$
2  for $C$ = -2.0 to 2.0 step 0.1
3      for $i = 1$ to $d$
4          minepsilon = $+\infty$
5          for $j = 1$ to $N$
6              $w_0 = \frac{C - x_i^j y^j}{y^j}$
7              sumepsilon = 0.0
8              for $k = 1$ to $N$
9                  $\epsilon^k = C - x_i^k y^k - w_0 y^k$
10                 if $\epsilon^k > 0$
11                     sumepsilon += $\epsilon^k$
12             if sumepsilon < minepsilon
13                 $bestw_0 = w_0$
14         error = ErrorOfSplit($x_i + bestw_0 \geq 0$, $V$)
15         if error < bestError
16             bestError = error
17             bestSplit = $x_i + bestw_0 \geq 0$
18 return bestSplit
```

FIGURE 1. The pseudocode of the search algorithm for finding the best split at each decision node of the UMT: N: number of examples at the decision node, d: number of inputs in the data set, $S = ((\boldsymbol{x}^1, y^1), \ldots, (\boldsymbol{x}^N, y^N))$: sample of N labeled data points at the decision node, V: validation data used to optimize C value.

TABLE 1. Details of the data sets.

Data set	d	K	N	Source	Dataset	d	K	N	Source
ads	1558	2	3279	UCI	braintumor1	5920	5	90	Bio
breast	9	2	699	UCI	braintumor2	10 367	4	50	Bio
bupa	6	2	345	UCI	dermatology	34	6	366	UCI
dlbcl	5469	2	77	Bio	ecoli	7	8	336	UCI
german	24	2	1000	UCI	glass	9	6	214	UCI
haberman	3	2	306	UCI	iris	4	3	150	UCI
heart	13	2	270	UCI	letter	16	26	20 000	UCI
hepatitis	19	2	155	UCI	leukemia1	5327	3	72	Bio
ironosphere	34	2	351	UCI	leukemia2	11 225	3	72	Bio
magic	10	2	19 020	UCI	lungtumor	12 600	5	203	Bio
musk2	166	2	6598	UCI	ocr	256	10	600	UCI
parkinsons	22	2	195	UCI	optdigits	64	10	3823	UCI
pima	8	2	768	UCI	pageblock	10	5	5473	UCI
polyadenylation	169	2	6371	UCI	pendigits	16	10	7494	UCI
prostatetumor	10 509	2	102	Bio	segment	19	7	2310	UCI
ringnorm	20	2	7400	UCI	shuttle	9	7	58 000	UCI
satellite47	36	2	2134	UCI	srbct	2308	4	83	Bio
spambase	57	2	4601	UCI	vehicle	18	4	846	UCI
transfusion	4	2	748	UCI	wave	21	3	5000	UCI
twonorm	20	2	7400	UCI	winequality	11	7	6497	UCI
9tumors	5726	9	60	Bio	wine	13	3	178	UCI
11tumors	12 533	11	174	Bio	yeast	8	10	1484	UCI
14tumors	15 009	26	308	Bio	zipcodes	256	10	7291	UCI
					zoo	16	7	101	UCI

d: number of attributes, K: number of classes, N: sample size.

TABLE 2. The average and standard deviations of error rates of decision trees generated using C4.5, LDT and UMT for $K = 2$ class data sets.

Data set	C4.5	LDT	UMT
ads	**3.36 ± 0.39**	3.65 ± 0.42	3.90 ± 0.30
breast	6.92 ± 1.24	6.45 ± 0.51	**6.32 ± 1.45**
bupa	**38.53 ± 4.38**	40.78 ± 4.42	41.98 ± 5.19
dlbcl	23.33 ± 8.91	25.56 ± 5.37	**21.48 ± 7.16**
german	29.43 ± 1.18	29.28 ± 1.40	**28.20 ± 1.56**
haberman	26.37 ± 0.31	27.25 ± 2.84	**26.18 ± 0.93**
heart	30.89 ± 3.98	**29.00 ± 4.76**	32.44 ± 2.81
hepatitis	22.31 ± 3.97	21.73 ± 1.30	**19.62 ± 2.53**
ironosphere	14.19 ± 4.44	14.87 ± 4.51	**11.28 ± 5.85**
magic	**17.13 ± 0.40**	17.70 ± 0.28	19.21 ± 0.41
musk2	**4.70 ± 0.60**	5.29 ± 0.60	12.70 ± 1.24
parkinsons	15.38 ± 4.47	15.54 ± 5.40	**14.92 ± 3.63**
pima	28.87 ± 2.72	28.44 ± 4.25	**26.26 ± 1.29**
polyadenylation	30.58 ± 2.11	**29.59 ± 1.73**	29.65 ± 1.27
prostatetumor	**15.71 ± 2.43**	20.29 ± 10.64	22.00 ± 11.43
ringnorm	**12.04 ± 0.71**	23.13 ± 1.18	15.12 ± 2.02
satellite47	**14.59 ± 1.33**	15.21 ± 1.17	14.61 ± 0.76
spambase	9.32 ± 1.22	**9.19 ± 0.80**	10.14 ± 0.56
transfusion	24.00 ± 0.00	**23.76 ± 0.76**	24.00 ± 0.00
twonorm	**17.50 ± 0.64**	17.73 ± 0.69	19.14 ± 0.68

TABLE 3. The average and standard deviations of number of nodes of decision trees generated using C4.5, LDT and UMT for $K = 2$ class data sets.

Data set	C4.5	LDT	UMT
ads	38.80 ± 19.81	50.20 ± 21.60	**31.00 ± 11.83**
breast	11.50 ± 7.65	10.00 ± 5.66	**7.60 ± 3.41**
bupa	18.70 ± 15.13	13.30 ± 16.34	**8.20 ± 6.96**
dlbcl	3.40 ± 1.90	2.50 ± 2.12	**1.90 ± 1.45**
german	**4.30 ± 6.99**	10.90 ± 20.88	9.10 ± 8.72
haberman	4.60 ± 8.69	4.60 ± 8.69	**2.20 ± 3.79**
heart	9.40 ± 5.44	8.20 ± 5.14	**4.90 ± 2.85**
hepatitis	10.90 ± 9.70	**2.50 ± 3.24**	4.90 ± 4.48
ironosphere	**11.80 ± 6.66**	12.40 ± 9.03	12.10 ± 5.84
magic	86.80 ± 28.47	121.30 ± 44.42	**63.40 ± 13.33**
musk2	114.10 ± 25.18	131.50 ± 28.43	**24.40 ± 11.12**
parkinsons	9.10 ± 5.84	8.80 ± 4.05	**5.80 ± 3.52**
pima	15.10 ± 13.27	23.80 ± 24.83	**7.60 ± 5.06**
polyadenylation	**38.50 ± 20.11**	54.70 ± 37.35	81.70 ± 28.44
prostatetumor	5.20 ± 2.10	4.90 ± 2.02	**3.70 ± 0.95**
ringnorm	157.00 ± 44.70	261.10 ± 67.42	**70.90 ± 10.96**
satellite47	40.60 ± 16.42	27.40 ± 20.29	**26.50 ± 9.72**
spambase	79.90 ± 31.78	106.60 ± 38.93	**54.70 ± 17.69**
transfusion	**1.00 ± 0.00**	1.90 ± 2.85	**1.00 ± 0.00**
twonorm	225.10 ± 44.95	248.80 ± 51.27	**171.70 ± 23.60**

For each data set the best result is shown in boldface.

TABLE 4. The average and standard deviations of error rates of decision trees generated using C4.5, LDT and UMT (one-vs.-one), and UMT (one-vs.-rest) for $K > 2$ class data sets.

Data set	C4.5	LDT	UMT(OVO)	UMT(OVR)
9tumors	85.45 ± 1.92	84.55 ± 4.89	78.64 ± 10.51	**74.09 ± 8.58**
11tumors	46.56 ± 6.05	41.80 ± 5.42	40.98 ± 2.04	**34.10 ± 4.69**
14tumors	77.39 ± 5.75	75.32 ± 3.85	70.54 ± 2.44	**63.42 ± 4.40**
braintumor1	37.50 ± 1.47	**33.44 ± 5.32**	38.75 ± 5.15	39.69 ± 9.55
braintumor2	51.11 ± 8.20	53.89 ± 12.02	**44.44 ± 10.14**	53.33 ± 12.88
dermatology	12.48 ± 1.52	12.48 ± 1.52	11.12 ± 1.38	**9.76 ± 2.26**
ecoli	21.30 ± 2.70	18.70 ± 3.36	23.04 ± 3.51	**16.52 ± 2.56**
glass	37.43 ± 3.43	**36.35 ± 4.79**	45.27 ± 3.50	40.95 ± 2.78
iris	7.65 ± 1.11	7.45 ± 0.83	**7.06 ± 1.89**	**7.06 ± 2.95**
letter	18.44 ± 0.50	19.26 ± 0.57	23.14 ± 0.81	**13.76 ± 0.80**
leukemia1	15.60 ± 2.27	**14.80 ± 3.79**	20.40 ± 12.57	15.20 ± 4.92
leukemia2	**14.40 ± 5.06**	17.60 ± 3.37	30.00 ± 14.88	16.00 ± 7.54
lungtumor	15.65 ± 6.03	**13.33 ± 8.53**	20.72 ± 5.55	14.49 ± 4.73
ocr	23.65 ± 4.18	23.65 ± 4.18	22.50 ± 2.76	**19.80 ± 3.85**
optdigits	14.99 ± 0.69	14.57 ± 1.48	11.71 ± 0.49	**8.50 ± 0.77**
pageblock	3.72 ± 0.48	**3.63 ± 0.45**	4.79 ± 0.51	4.50 ± 0.36
pendigits	5.90 ± 0.47	6.32 ± 0.58	9.78 ± 0.73	**4.66 ± 0.45**
segment	**4.97 ± 1.14**	5.44 ± 0.67	8.43 ± 1.27	6.32 ± 0.98
shuttle	**0.07 ± 0.01**	0.11 ± 0.02	1.07 ± 2.24	0.39 ± 0.07
srbct	29.31 ± 9.92	23.10 ± 11.84	26.21 ± 8.93	**14.83 ± 7.46**
vehicle	29.82 ± 2.43	30.35 ± 2.34	32.15 ± 3.43	**28.80 ± 1.71**
wave	24.67 ± 1.17	24.14 ± 0.58	24.52 ± 0.80	**23.19 ± 1.49**
winequality	45.58 ± 1.26	46.40 ± 0.76	46.12 ± 0.86	**45.10 ± 0.76**
wine	10.33 ± 3.58	13.00 ± 6.89	9.17 ± 2.97	**8.50 ± 3.09**
yeast	47.85 ± 3.16	49.50 ± 2.20	45.06 ± 2.23	**42.71 ± 0.94**
zipcodes	15.60 ± 0.82	17.30 ± 0.84	12.27 ± 0.61	**10.02 ± 0.85**
zoo	**15.95 ± 4.50**	**15.95 ± 4.50**	18.65 ± 5.62	18.11 ± 3.83

For each data set the best result is shown in boldface.

inequalities shown in Equation (9), where each time a different ϵ^t is zero (Line 6). To get the minimum value of the penalty term $\sum_t \epsilon^t$ for a specific w_0, we set ϵ^k to zero if $C - x_i^k y^k - w_0 y^k < 0$ and to $C - x_i^k y^k - w_0 y^k$ if $C - x_i^k y^k - w_0 y^k > 0$ (Line 11). The threshold w_0 corresponding to the minimum overall penalty will be selected as a best split candidate (Lines 12 and 13). Similar to the SVM, to optimize the length of the margin C, we need a separate validation set. The instances are normalized in order to use the same C value for each feature. We calculate the error rate of the current best split candidate on the validation set and compare it with the best error so far (Lines 14 and 15). We return the split with minimum error as the best split (Line 17).

For a given data size N and dimension d, the computational complexity of the algorithm is $\mathcal{O}(dN^2)$. But like C4.5, one can sort w_0's ($\mathcal{O}(N \log N)$) and calculate the minimum overall penalty in $\mathcal{O}(N)$ time resulting in a lower computational complexity of $\mathcal{O}(dN \log N)$, which is the same as C4.5's complexity.

3. EXPERIMENTS

3.1. Experimental setup

In this section, we compare the performance of our proposed UMT algorithm with C4.5 and LDT in terms of generalization error and model complexity as measured by the number of nodes in the decision tree. We use a total of 47 data sets where 36 of them are from UCI repository [13] and 11 are bioinformatics cancer data sets [14] (Table 1).

Our methodology in generating train, validation and test sets is as follows: a data set is first divided into two parts, with one-third as the test set, *test*, and two-third as the training set. The training set is then resampled using 2 × 5 cross-validation to generate 10 training and validation folds, tra_i, val_i, $i = 1, \ldots, 10$. tra_i are used to train the decision trees and val_i are used to prune the decision trees using cross-validation-based postpruning. *test* is used to estimate the generalization error of the decision trees. We use k-fold paired t test ($\alpha = 0.05$) to compare univariate decision tree construction methods (in terms of generalization error and tree complexity) for statistically significant difference.

3.2. Results

Table 2 shows average and standard deviations of error rates of decision trees generated using C4.5, LDT and UMT for $K = 2$ class data sets. We see from the results that UMT is as good as C4.5 and LDT in terms of error rate. In 3 data sets out of 20 UMT has significantly smaller error rate than C4.5 and LDT. On the other hand, C4.5 and LDT have significantly smaller error rate than UMT in 5 data sets out of 20.

Table 3 shows the average and standard deviations of number of nodes (including the leaves) of decision trees generated using C4.5, LDT and UMT for $K = 2$ class data sets. Better than the results above, UMT is significantly better than both C4.5 and LDT in terms of tree complexity. In 9 data sets out of 20, UMT generates significantly smaller trees than C4.5 and LDT; whereas, C4.5 (LDT) generates significantly smaller trees than UMT in only 1 (0) dataset out of 20. We can conclude that for two-class data sets, UMT generates smaller trees than C4.5 and LDT without sacrificing from accuracy.

In UMT, for $K > 2$ class problems, we take the most commonly used approach and reduce the single multiclass problem into multiple binary classification problems. UMT uses two well-known methods to build binary classifiers where each classifier distinguishes between (i) one of the labels to the rest (one-vs.-rest) or (ii) between every pair of classes (one-vs.-one). Table 4 shows average and standard deviations of error rates of decision trees generated using C4.5, LDT and UMT (one-vs.-one), and UMT (one-vs.-rest) for $K > 2$ class data sets. We see from the results that, UMT (one-vs.-one) is as good as C4.5 and LDT in terms of error rate. In 5 data sets out of 27, UMT (one-vs.-one) has significantly smaller error rate than C4.5 and LDT, whereas C4.5 and LDT have significantly smaller error rate than UMT (one-vs.-one) in 8 data sets out of 27. On the other hand, UMT (one-vs.-rest) is significantly better than C4.5 and LDT in terms of error rate. In 13 data sets out of 27, UMT (one-vs.-rest) has significantly smaller error rate than C4.5 and LDT, whereas C4.5 and LDT have significantly smaller error rate than UMT (one-vs.-rest) only in 3 data sets out of 27.

4. CONCLUSION

In this paper we propose UMT, where the best split is found using convex optimization at each decision node. The main idea comes from simplifying the formulation of multivariate linear SVM. Simplification is done by (i) replacing the multivariate discriminant $\boldsymbol{w}$ with the univariate axis-orthogonal split $x_j + w_0 \geq 0$, (ii) removing the $||\boldsymbol{w}||^2$ factor from the objective function since there is only single variable x_j, and (iii) redefining the margin in the one-dimensional space. None of the resulting primal and dual formulations is easily solvable, therefore we resort to searching for the optimal w_0 exhaustively. Although the first-come to mind exhaustive search is expensive, with the same trick applied in C4.5, one can get a computational complexity of $\mathcal{O}(dN \log N)$, which is much better than the usual time complexity of the traditional SVM.

Experimental results on 36 data sets from UCI repository and 11 bioinformatics data sets show that for two-class problems, our proposed UMT not only performs as good as C4.5 and LDT (both univariate decision tree classifiers) in terms of accuracy, but it produces significantly smaller trees to do that. For multiclass problems, we tried two well-known reduction techniques, namely, one-vs.-one and one-vs.-rest. In terms of generalization error, one-vs.-rest is significantly better than one-vs.-one technique, which is also as good as C4.5 and LDT.

FUNDING

This work has been supported by the Turkish Scientific Technical Research Council TÜBİTAK EEEAG 107E127.

REFERENCES

[1] Mitchell, T. (1997) *Machine Learning*. McGraw-Hill, MA.

[2] Alpaydın, E. (2010) *Introduction to Machine Learning*. The MIT Press, MA.

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

[4] Quinlan, J.R. (1986) Induction of decision trees. *Mach. Learn.*, **1**, 81–106.

[5] Breiman, L., Friedman, J.H., Olshen, R.A. and Stone, C.J. (1984) *Classification and Regression Trees*. John Wiley and Sons, New York.

[6] Yıldız, O.T. and Alpaydın, E. (2005) Linear discriminant trees. *Int. J. Pattern Recognit. Artif. Intell.*, **19**, 323–353.

[7] Duda, R.O., Hart, P.E., and Stork, D.G. (2001) *Pattern Classification*. John Wiley and Sons, New York.

[8] Guo, H. and Gelfand, S.B. (1992) Classification trees with neural network feature extraction. *IEEE Trans. Neural Netw.*, **3**, 923–933.

[9] Vapnik, V. (1995) *The Nature of Statistical Learning Theory*. Springer, New York.

[10] Tibshirani, R. and Hastie, T. (2007) Margin trees for high-dimensional classification. *J. Mach. Learn. Res.*, **8**, 637–652.

[11] Bennett, K. and Blue, J. (1998) A support vector machine approach to decision trees. *Proceedings of the International Joint Conference on Neural Networks*, Anchorage, Alaska, 4–9 May, pp. 2396–2401. IEEE.

[12] Vural, V. and Dy, J.G. (2004) A hierarchical method for multi-class support vector machines. *Proc. 21st Int. Conf. Machine Learning*, Banff, Alberta, Canada, 4–8 July, pp. 831–838. ACM, New York.

[13] Asuncion, A. and Newman, D.J. (2007) *UCI Machine Learning Repository*. http://archive.ics.uci.edu/ml.

[14] Statnikov, A., Aliferis, C., Tsamardinos, I., Hardin, D. and Levy, S. (2005) A comprehensive evaluation of multicategory classification methods for microarray gene expression cancer diagnosis. *Bioinformatics*, **21**, 631–643.