New Approaches for Multiclass Classification

Shankhpal S.V., Dhawas N.A.

Abstract*: Classification is very important in data mining. It is nothing but categorization of data for its most effective and efficient use. In basic approach to storing data, data can be classified according its importance or how often it needs to be accessed decision tree is one of the classification technique. Decision tree is used to clarify and find solution to complex problem. Structure of decision tree contains multiple possible solutions and displays it in a simple, easy to understand format. There is different algorithm used for classification. In this paper tree is constructed using the geometric structure of data. It builds small decision trees and gives better performance. Now we will use adaptive boosting method for boosting decision tree so it improving the accuracy of decision tree.

***Index Terms*: GDT, Multiclass classification, Oblique decision tree, SVM**

I. INTRODUCTION

A decision tree is a kind of flowchart -- a graphical representation of the process for making a decision or a series of decisions. A decision tree is a diagrammatic representation of a problem and on it we show all possible courses of action that we can take in a particular situation and all possible outcomes for each possible course of action. It is particularly useful where there are a series of decisions to be made and/or several outcomes arising at each stage of the decision-making process. For example, we may be deciding whether to expand our business or not. The decision may be dependent on more than one uncertain variable. Decision trees are easy to use once you understand that. For example, this simple decision tree:

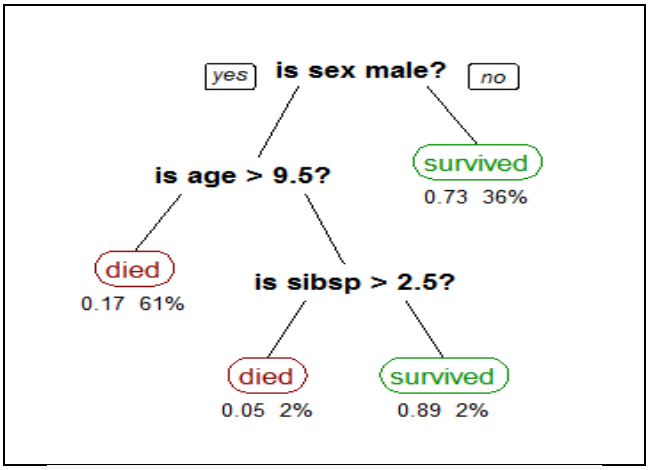

Fig. 1 A Tree Showing Survival of Passengers on the Titanic ("Sibsp" is the Number of Spouses or Siblings Aboard). The figures Under the Leaves Show the Probability of Survival and the Percentage of Observations in the Leaf

Manuscript Received on July 2014.

Shankhpal S.V., Received B.E. Degree in Computer Engineering and pursuing M.E. Computer Engineering at SIT, Lonavala, Maharashtra, India.

Prof. Dhawas N.A., HOD of IT Department at SIT, Lonavala, Maharashtra, India.

Decision trees are so likeable because they are easily understood, as they can be graphically presented as trees as well as in the form of rules, they are powerful and popular tools for classification and prediction. Decision trees are constructed in top down manner. The nodes of a tree represent problem; solution to one question determines which question will be asked next. The process starts in the root node, where a record is tested and the result of the test determines lower node where the process will proceed. It is an iterative process that is repeated until the record reaches a leaf, which represents one class of the data. Every node in the tree represents a test of some case attribute, and a path that leads from root to the leaf represents a rule that was used for classification, that is, every branch that is derived from that node represents a possible value of that attribute. A binary tree is the one that, for instance, answers the question with „yes“ or „no“, so that every leaf has two child“ nodes, and the answer determines which way the data will go to the next level. If the data has *m* attributes, the maximum height of a tree will be *m*. Figure 2 represents a binary tree. Algorithms that are used for building decision trees start with a search for the test which does the best job in splitting the data between the categories. On every other level of a tree, subsets that are created in the previous step are being split by a test that does the best job for them. Some rules are better than the others so we are able to estimate the percentage of the cases classified incorrectly. Also, some leaves have very limited records while others have a lot. We can measure effectiveness of a tree as a whole with its application to new data and by viewing the percentage of the data that is correctly classified. In every node, we can measure: a number of records that enter the node, the way that these records will be classified if that was the leaf node, the percentage of records that are correctly classified in that node

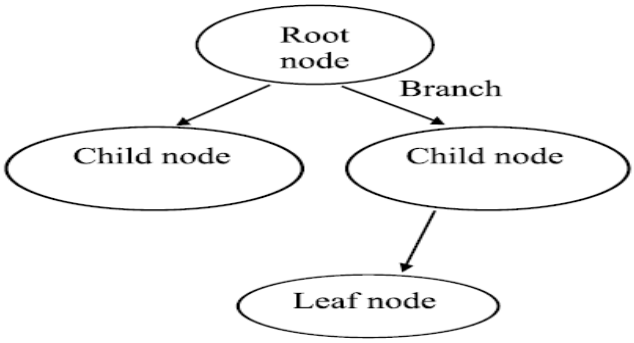

Fig. 2 Structure of Decision Tree

Decision tree cab be broadly categorized into two types i.e. axis parallel and oblique decision tree. Axis parallel decision trees that consider only a single attribute at a time make splits parallel to the axis in the feature space of the dataset and

oblique decision trees split the feature space by considering combinations of the attribute values, be them linear or otherwise .Oblique decision trees have the prospective to outperform regular decision trees because with a smaller Number of splits an oblique hyper plane can achieve better separation of the instances of data that belong to different classes. We proposed use an Adaptive Boosting method for boosting decision tree which will create multiple small geometric decision trees in the view of improving the performance of prediction in terms of accuracy, confusion matrix, precision and recall. The proposed modification to geometric decision tree is applied on standard data set from UCI repository. The rest of the paper is organized as follows. In Section II, we present the proposed algorithm. Mathematical Model Presented in Section III Experimental studies presented in Section Iv, Section V presents conclusions and Future Work.

II. PRPOSED SYSTEM

Proposed method can be applied for two class and multiclass problem. We start with the given training data and decide on the "best" hyper plane, which is attributed to the origin of the tree. Then, we partition the training examples into two sets that go to the left child and the right child of the root node using this hyper plane. Then, at each of the two child nodes, we repeat the same process (using the appropriate subset of the training data). The recursion terminates when the set of training examples that come to a node is pure, that is, all these training patterns are of the same class or the depth of the tree is at a maximum set value. Then, we make it a leaf node and assign that class to the leaf node based on distribution of records in terms of class values.

1. Start with data with last column as classes
2. Divide data in two parts as training (~70%) and testing (~30%) randomly
    a. Build decision tree with training data and test it using test data
3. Identify records for one class (or major class) as form matrix A
4. Create matrix B with another class records (or other than major class)
5. Calculate hyper planes w1 and w2 using matrix A and B respectively
6. Calculate angle bisectors as w3 and w4 using w1 and w2
7. Calculate Gini index for the data separated by angle bisectors w3 and w4
8. Identify best angle bisector whose Gini index is less
9. Split the data using best angle bisector and form left and right nodes for the data one either sides of a best angle bisector, Repeat process from step 3 to 9 on each newly created node and grow nodes until the node is 100% pure or the depth of the node is equal to maximum depth of decision tree provided by user. Apply boosting algorithm which will create multiple GDTs in the view of improving overall accuracy of predictions.

Calculate accuracy in terms of following measures

1. Overall Accuracy
2. Confusion matrix
3. Precision
4. Recall
5. F measure
6. Area under curve

AdaBoost Algorithm:

The algorithm proceeds in the following steps for a training data with given value of boosting

1. Assign weights for each records as 1/n where n is total number of training records
2. Generate next decision tree.
3. Calculate error of decision tree and update the weights for each record.
4. Repeat steps 2 and 3 till total numbers of tree are less than given number of there are sufficient records.

III. MATHEMATICAL MODEL

Set Theory Analysis

1. Identify the input data set as

S= {s1, s2, s3, s4…..}

Where

S - Input data set

2. Identify the data classes

CL = {cl1, cl2, cl3, cl4….}

Where CL - The data class

Process:

Given a set of data points at a node, we find hyper-planes that have most of the data points of same class. Find the best hyper-plane to split the data.

Let

C1 be the set of points contains points of the majority class

C2 be the set of points contains points of the remaining classes

Let

A be the matrix containing points of C1

B be the matrix containing points of C2

Let

W1 be the first hyper-plane

W2 be the second hyper-plane

$$\widehat{W_1} = \operatorname{argmin}_{\hat{w}\neq 0} \frac{D + (W,B)}{D - (W,B)} = \operatorname{argmin}_{\hat{w}\neq 0} \frac{\hat{w}^T G\,\hat{w}}{\hat{w}^T H\,\hat{w}} = \operatorname{argmax}_{\hat{w}\neq 0} \frac{\hat{w}^T H\,\hat{w}}{\hat{w}^T G\,\hat{w}}$$

$$\widehat{W_2} = \operatorname{argmin}_{\hat{w}\neq 0} \frac{D + (W,B)}{D - (W,B)} = \operatorname{argmin}_{\hat{w}\neq 0} \frac{\hat{w}^T H\,\hat{w}}{\hat{w}^T G\,\hat{w}}$$

Where G = (1 /nt+) [A] T [A].

H= (1 /nt-) [B] T [B].

Let

W3 be the first bisecting angle

W4 be the second bisecting angle

Such that

W3 = W1 + W2

W4 = W1 – W2

Choose the angle bisectors having lesser Gini index G

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication Pvt. Ltd.

$$Gini(\tilde{w}_t) = \frac{n^{tl}}{n^t}\left[1 - \left(\frac{n^{tl}+}{n^{tl}}\right)^2 - \left(\frac{n^{tl}-}{n^{tl}}\right)^2\right] + \frac{n^{tr}}{n^t}\left[1 - \left(\frac{n^{tr}+}{n^{tr}}\right)^2 - \left(\frac{n^{tr}-}{n^{tr}}\right)^2\right]$$

Where,

n^{tl} +- Number of points in matrix A that goes to left child

n^{tr} + - Number of points in matrix A that goes to right child

n^{tl} − - Number of points in matrix B that goes to right child

n^{tr} − - Number of points in matrix B that goes to right child

Ada Boosting:

Training set: $(x_1, y_1), \ldots, (x_m, y_m)$

Where $x_i \in X, y_i \in Y = \{-1, +1\}$

number of iterations or boosting trails = T

For i=1,….,m

Initialize $D_1(i) = \frac{1}{m}$

For t = 1,….T

From the family of weak classifiers $\mathcal{H}$, find the classifier h_t that maximizes the absolute value of the difference of the corresponding weighted error rate ϵ_t and 0.5 with respect to the distribution D_t :

$$h_t = \underset{h_t \in H}{\operatorname{argmax}} |0.5 - \epsilon_t|$$

Where, $\epsilon_t = \sum_{i=1}^{m} D_t(i) I(y_i \neq h_t(x_i))$

(I is the indicator function with true/false results)

If $|0.5 - \epsilon_t| \leq \beta$, where β is a previously chosen threshold, then stop.

Choose, $\alpha\_t \in R$ typically $\alpha_t = \frac{1}{2} \ln \frac{1-\epsilon_t}{\epsilon_t}$

For i=1,….,m:

$$D_{t+1}(i) = \frac{D_t(i) \exp(\alpha_t (2I(y_i \neq h_t(x_i)) - 1))}{Denom}$$

Where the denominator, Denom, is the normalization factor ensuring that D_{t+1} will be a probability distribution.

Output the final classifier:

$$H(x) = sign\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$$

IV. EXPERIMENTAL RESULTS

In this section, we present experiential results to show the effectiveness of our decision tree algorithm. We test the performance of our algorithm on several data sets. We compare our results with GDT & GDT with Adaptive boosting .The proposed enhancement in geometric decision tree is evaluated on a sample data on wine data set collected from UCI repository. The wine data set has 13 attributes and a target field having three classes.

We have applied simple GDT on wine data and observed following results.

Training Data:

Overall Training Data Accuracy = 90.14%

Confusion Matrix: Rows represents actual numbers

Class	1	2	3	%Accuracy
1	26	0	13	66.66
2	0	55	1	98.21
3	0	0	47	100

Precision and Recall Table:

	Class-1	Class-2	Class-3
Precision	0.67	0.98	1.00
Recall	1.00	1.00	0.77

Test Data:

Overall Test Data Accuracy = 75.0%

Confusion Matrix: Rows represents actual numbers

Class	1	2	3	%Accuracy
1	6	2	1	66.66
2	1	12	2	80
3	2	1	9	75

Precision and Recall Table:

	Class-1	Class-2	Class-3
Precision	0.67	0.80	0.75
Recall	0.67	0.80	0.75

Also same wine data is applied on enhanced GDT using Adaboosting and observed following results.

Training Data:

Overall Training Data Accuracy = 98.59%

Confusion Matrix: Rows represents actual numbers

Class	1	2	3	%Accuracy
1	39	0	0	100
2	2	54	0	96.42
3	0	0	47	100

Precision and Recall Table:

	Class-1	Class-2	Class-3
Precision	0.95	1.00	1.00
Recall	1.00	0.96	1.00

Test Data:

Overall Test Data Accuracy = 83.33%

Confusion Matrix: Rows represents actual numbers

Class	1	2	3	%Accuracy
1	6	3	0	66.66
2	1	13	1	86.66
3	0	1	11	91.66

Precision and Recall Table:

	Class-1	Class-2	Class-3
Precision	0.67	0.87	0.92
Recall	0.67	0.87	0.92

Based on results we can observe confusion matrix shows that modified decision tree model has improved miss classification of class 3 from 75% to 91.66% on test data without losing on misclassification accuracy for other two classes which is significant.

Fig. 3 Comparison of Accuracy Obtained for all Three Classes for Simple GDT and GDT with Ada Boosting for Training Data

V. CONCLUSION & FUTURE WORK

In this Paper a new algorithm is design which creates multiple decision trees at each node of decision tree found two clustering hyperplanes and angle bisector as split rule. Based on this we create multiple decision trees by which we got average split value and it increases accuracy in terms of confusion matrix, precision & recall.

VI. FUTURE WORK

1. Apply gradient boosting instead of adaptive boosting.
2. Introduce pruning process to remove the excess nodes created in the decision tree so that the generalization performance of the decision tree can be improved.
3. There are multiple pruning processes available in literature. So need to find out which one performs best with GDT.

ACKNOWLEDGMENT

Our sincere thanks go to Sinhgad Institute of Technology for providing a strong platform to develop our skill and capabilities. We would like to thanks to our guide & respected teachers for their constant support and motivation for us. Last but not least, we would like to thanks all those who directly or indirectly help me in presenting the given paper.

REFERENCES

1. A. Kołakowska and W. Malina, "Fisher sequential classifiers," IEEE Trans. Syst., Man, Cybern. B, Cybern, Vol. 35, No. 5, Pp. 988–998, Oct. 2005.
2. D. Dancey, Z. A. Bandar, and D. McLean, "Logistic model tree extraction from artificial neural networks," IEEE Trans. Syst., Man, Cybern. B, Cybern, vol. 37, no. 4, pp. 794–802, Aug. 2007.
3. Erick Cantú-Paz, Chandrika Kamath, ―Inducing Oblique Decision Trees with Evolutionary Algorithms. IEEE Transaction on Evolutionary Computation, Vol. 7, No. 1, February 2003.
4. Haitang Zhang, Hongze Qiu "Sensitivity degree based fuzzy SLIQ decision tree." 978-1-4244-7941-2/10/ ©2010 IEEE.
5. J. Quinlan, ―Induction Of Decision Trees, Mach. Learn., Vol. 1, No. 1, Pp. 81–106, 1986.
6. K. P. Bennett and J. A. Blue, ―A Support Vector Machine Approach To Decision Trees, In Proc. IEEE World Congr. Computer. Intell, Anchorage, AK, May 1998, Vol. 3, Pp. 2396–2401.
7. L. Breiman, J. Friedman, R. Olshen, and C. Stone, ―Classification and Regression Trees. Belmont, Ca: Wadsworth and Brooks, 1984, Ser. Statistics/Probability Series.
8. Lior Rokach and Oded Maimon, "Top-Down Induction of Decision Trees Classifiers – A Survey." IEEE Transactions on Systems, Man and Cybernetics: Part C, Vol. 1, No. 11, November 2002.
9. M. F. Amasyali and O. Ersoy, "Cline: A new decision-tree family," IEEE Trans. Neural Netw., vol. 19, no. 2, pp. 356–363, Feb. 2008.
10. M. Lu, C. L. P. Chen, J. Huo, and X. Wang, "Multi-stage decision tree based on inter-class and inner-class margin of SVM," in Proc. IEEE Int. Conf. Syst., Man, Cybern., San Antonio, TX, 2009, pp. 1875–1880.
11. Naresh Manwani and P. S. Sastry, ―"Geometric Decision Tree", IEEE Transactions on Systems, Man, and Cybernetics—Part B: Cybernetics, Vol. 42, No. 1, February 2012.