Rambabu P, Srinivasa Rao D, Bhupati Sampath Babu,B Harikrishna,Neelima S/ International Journal of Engineering Research and Applications (IJERA) ISSN: 2248-9622 www.ijera.com
Vol. 1, Issue 4, pp.1343-1351

Implementation of FS-DT Algorithm to Enhance the Performance of Decision Tree

Rambabu P[*], Srinivasa Rao D[**], Bhupati Sampath Babu[***], B Harikrishna[****],Neelima S[*****]

[*]Assistant Professor in Nimra Institute of Engineering & Technology
[**]Associate Professor in SSN College of Engineering
[***]Assistant Professor in SSN College of Engineering
[****]Assistant Professor in A1 Global Institute of Engineering & Technology
[*****]Assistant Professor in Pydha college of engineering

Abstract

In recent days Data mining gained a large amount importance because it enables modeling and knowledge extraction from the abundance of data which is available. In which Decision trees are powerful and popular tools for categorization and predicting knowledge discovery. Due to drawback of sharp decision boundaries decision tree algorithms are not that much efficiently implemented to define real time classification problem. An another important parameter to be considered is like when the results are larger and deeper for a decision tree it leads to inexplicable induction rules. In this paper we are proposing a fuzzy supervised learning in Quest decision tree (FS-DT) algorithm where we focused to design a fuzzy decision boundary instead of a crisp decision boundary. To construct a fuzzy binary decision tree we used SLIQ decision tree algorithm which is modified here by FS-DT to reduce the size of the decision tree, using several real-life datasets taken from the UCI Machine Learning Repository performance of the FS-DT algorithm is compared with SLIQ and many comparisons have been proposed.

I .INTRODUCTION

A decision tree is a flow chart like structure where each internal node denotes a test on an attribute, each branch represents an outcome of the test, and leaf nodes represent classes or class distributions. A top most node in a tree is the root node. The basic algorithm for decision tree induction is a greedy algorithm that constructs decision tree in a top-down recursive divide-and-conquer manner. Decision trees, originally implemented in decision theory and statistics, are highly effective tools in other areas such as data mining, text mining, information extraction, machine learning, and pattern recognition.

Decision trees are categorized into six types they are:-

***Classification tree*:** analysis is when the predicted outcome is the class to which the data belongs.

Regression tree:analysis is when the predicted outcome can be considered a real number.

***Classification and Regression Tree (CART*):** analysis is when the predicted outcome is the class to which the data belongs and analysis is when the predicted outcome can be considered a real number

CHi-squared Automatic Interaction Detector (CHAID): Performs multi-level splits when computing classification trees.

***Random Forest*:** classifier uses a number of decision trees, in order to improve the classification rate.

***Boosted Trees*:** can be used for regression-type and classification-type problems.

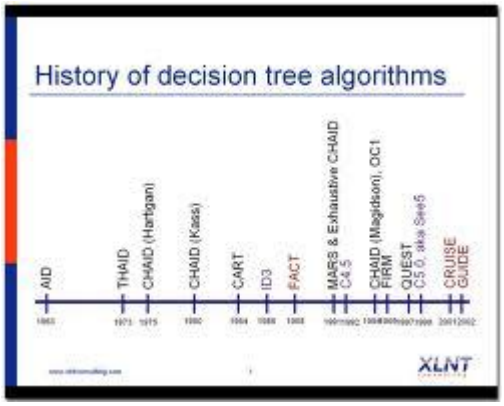


Fig 1: History of Decision tree algorithms

Powerful method of pattern classification is implemented by Top-Down induction of decision trees. Decision trees have an ability to handle complex problems by providing an understandable representation that is easier to interpret by producing logical rules of classification. Extensively used classification models to handle the complex problems are entropy-based decision tree algorithms like ID3 and C4.5, which are used to produce amultiway decision tree. To generate a binary decision tree, supervised learning in Quest (SLIQ) algorithm uses the Gini index as a split measure. Some more algorithms that use the Gini index as a split measure are the SPRINT,CLOUDS, CMP-S, and elegant decision tree algorithms. Using the Gini index in decision trees has been explored for improving the performance of SLIQ by computing the Gini index only at those points where the class information changes.

Due to sharp boundary limitations conventional decision tree algorithms do not give rules that are easy to understand and are inefficient. On the other hand it is difficult in classification is its ability to handle quantitative data appropriately. Generally, the quantitative data are partitioned into a set of hard sets recursively until the entire set belongs to a single class. In this paper we are proposing theFuzzy decision trees to defeat this problem.Here Membership functions are given manually so finding a suitable membership function is a major problem even for a expert for datasets having high dimension and size. Most of the fuzzy decision tree algorithms are based on ID3 or C4.5 wherein the quantitative data are converted into fuzzy linguistic terms. In this paper,we propose a newway of fuzzifying the SLIQ decision tree algorithm. Without converting the quantitative values into fuzzy linguistic termsthe fuzzy SLIQ decision tree (FS-DT) algorithm constructs a binary decision tree. the degree of fuzziness of the attribute values given by membership function which indicating whether it is equal to the split value or not.

II.RELATEDWORK

SLIQ is very popular traditional binary decision tree algorithm, developed by the Quest team at IBM.SLIQ used for datasets with numeric attributes, which generates crisp decision boundaries as shown in Fig 2. In a crisp decision tree, the split point is chosen as the midpoint of successive attribute values where the class information changes. Whenever a splitting attribute is chosen with a particular value, there is no guarantee that this is the exact value at which the split has to take place. Traditional decision tree algorithms face the problem of having sharp decision boundaries which are hardly found in any real-life classification problems. Traditional decision tree algorithms do not give rules that are easy to understand and are inefficient due to sharp boundary problems. One of the difficult problems in classification is its ability to handle quantitative data appropriately. Size of the decision is high (Large and deeper Decision tree) so performance of the decision tree is decreased. The decision tree is to be induced from *N* training patterns represented as

$$\left(x^{(1)}, y^{(1)}\right), \left(x^{(2)}, y^{(2)}\right), \ldots, \left(x^{(N)}, y^{(N)}\right)$$

Where$x^{(i)}$ is a vector of *n* attributes and $y^{(i)}\epsilon(\omega 1, \omega 2, \ldots, \omega C)$ is the class label corresponding to theinput $x^{(i)}$. At a particular node *v*, let there be *N*(*v*) trainingpatterns (for the root node, or node 0, *N*(0) = *N*). The number of training patterns at node *v* belonging to class ωk is denotedby $N_{\omega_k}^{(v)}$. $\sum_k N_{\omega_k}^{(v)} = N^{(v)}$, and $N^{(u)}$ is the number of trainingpatterns in the dataset (under consideration) to be partitioned.

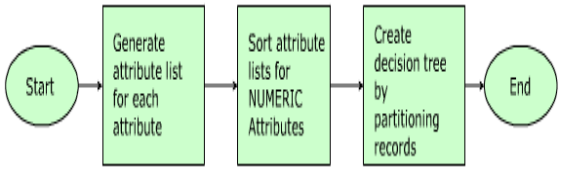


Fig 2: SLIQ methodology

The Gini index used as the node splitting measure is asfollows::

$$D(x_j) = \sum_{v=1}^{V} \frac{N(v)}{N(u)} \left[1 - \sum_{k=1}^{c} \left(\frac{N_{\omega_k}^{(v)}}{N(v)} \right)^2 \right]$$

The decision tree is constructed by splitting one of the attributes at each level as shown in Fig 3. The basic aim is to generate a tree,which is most accurate for prediction. With the help of the Gini index, it is decided which attribute is to be split and the splitting point of the attribute. The Gini index is minimized at each split, so that the tree becomes less diverse as we progress.

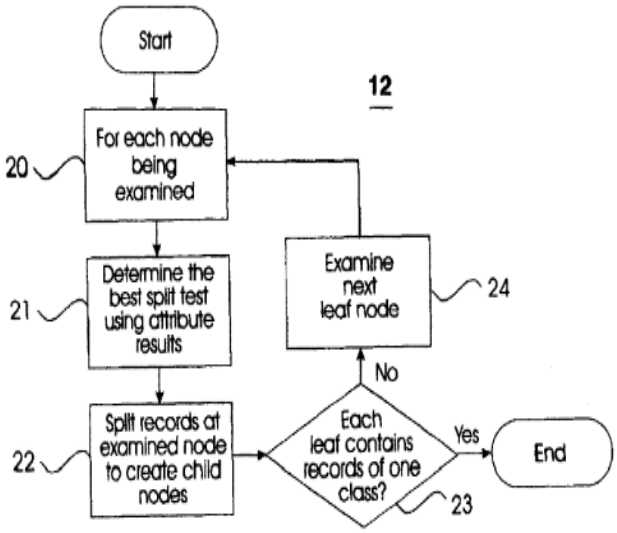


Fig 3: constructing the decision tree

III .PROPOSED ALGORITHM

To generate crisp decision boundaries thevery popular binary decision tree algorithm usedfor datasets with numeric attributes is SLIQ. To generate fuzzy decision trees for datasets with quantitative attributes in this paper we suggest a fuzzification procedurefor the SLIQ algorithm. In a crisp decision tree,the split point is chosen as the midpoint of successive attributevalues where the class information changes. There is no assurance that this is the correct value at which the split has to take placewhenever a splittingattribute is selected with a particular value. For which we design afuzzy membership function that has membership values thatare equal to one for records having values that are very closeto the chosen split point, and for points other than these, thefuzzy membership value depends on the standard deviation ofthe attribute. The decision will be crisper when larger the standard deviation has occurred as shown in Fig 4& Fig 5. For each of the attributes the fuzzy membership values are calculatedby using the following membershipfunction:

$$\text{fuzzy value} = \begin{cases} \frac{lw}{lp+lw-val}, & val < lp \\ 1.0, & lp \le val \le rp \\ \frac{rw}{val-rp+rw}, & val > rp \end{cases}$$

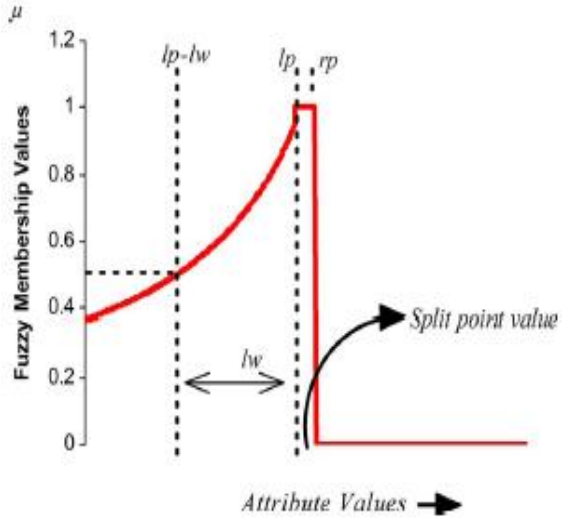


Fig 4: Membership function of attribute values that are less than or equal to the value of the splitting attribute

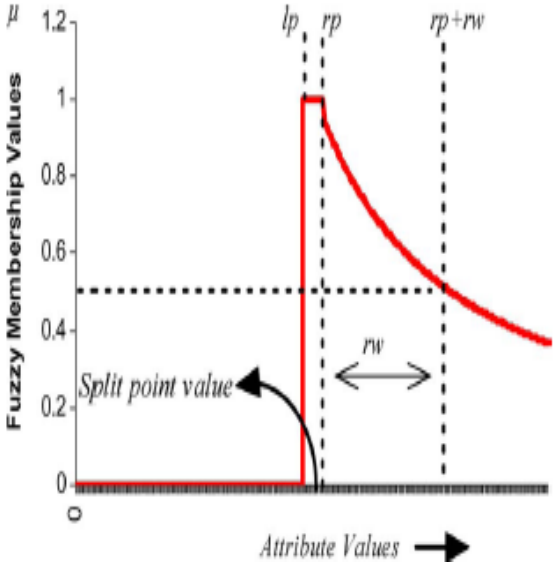


Fig 5: Membership function of attribute values that are greater than or equalto the value of the splitting attribute

Where,

$$\left.\begin{array}{l} lw = \alpha * \sigma \\ rw = 0.0 \\ lp = split\_point - \beta \\ rp = split\_point + \beta \end{array}\right\} \text{attribute value} \leq \text{split point}$$

$$\left.\begin{array}{l} lw = 0.0 \\ rw = \alpha * \sigma \\ lp = split\_point - \beta \\ rp = split\_point + \beta \end{array}\right\} \text{attribute value} \geq \text{split point}$$

σ denotes standard deviation.

The split point are dependent on the standard deviation which are represented using the attribute values like the parameters l_w and r_w for the left and right sides and curve's slope is controlled by the parameters l_w and r_w. The parameter β decides the range of attribute values that will have a membership value that is equal to one. Normally, β takes values between zero and one. When the attribute values have a higher spread, β can take even a value that is greater than one. α *σ decides the spread of the membership function. For a variety of real-life datasets taken from the UCI Machine Learning Repository the fuzzy membership function works well, on thestandard deviation of the attribute values of the dataset the shape of the membership function is dependent.

TheGini index is evaluated at every successive midpoint of the attributevalues in the SLIQ algorithm. The efficiency ofthe SLIQ decision tree algorithm can be improved by evaluatingthe Gini index only at midpoints of attributes where the class information changes, and by doing so, the performance and the decision tree constructed do not change. In this paper, the candidate split value of the attribute is chosen to be the midpoint of the attribute at class boundaries. The Gini index is evaluated by using the fuzzy values associated with the attribute at particular split value and the fuzzy membership values of each record in the dataset, i.e., $N_{\omega_k}^{(v)} = \text{sum}$ of the product of fuzzy membership of the values of the attribute in each record and the membership of the respective records for a particular classω_k. Initially, every record is assumed to have a membership value that is equal to one divided by the number of classes for the root node. Once the root node is selected, the membership values of the records are modified by using the membership values of the attribute chosen as the split point. This procedure recursively repeated until any one of the following three conditions are satisfied.

1) For any class in the dataset, if the ratio of the sum of the membership values of records to that of the total membership value of the entire dataset is greater than or equal to a particular threshold value decided by the user.

2) The number of records in the dataset is less than α *size of the complete dataset, where α is a value between zero
and one.

3) Depth of the decision tree is more than a user-specified threshold (max_depth). The leaf nodes will have the presence of all the classes with the corresponding normalized fuzzy membership values.

The decision made at each node of the decision tree is whether the test attribute value is equal or not equal to the split value of the attribute at each node, unlike SLIQ where the decision made is less than or greater than the attribute at each node. The algorithm is given as follows.

Function Decision-Tree-Construction (Level L) Algorithm:

1. If *L* is root then
2. Initialize the fuzzy membership values for each record in the dataset to 1/number of classes.
3. Else
4. Fuzzy membership value of each record is updated by multiplying it with the fuzzy membership values of the chosen attribute.
5. End
6. For each node at level *L* do
7. If splitting criterion is not met then
8. For each attribute *a* in the dataset do
9. Sort the attribute values along with class information
and fuzzy membership values of records
10. For each midpoint value *v* of the distinct values in
the presorted attribute list where class information
changes do
11. Evaluate-Gini-index (*a*, *v*)
12. End
13. Find the attribute value *v* for attribute *a*, which has

the least Gini index value
14. End
15. Select the attribute *a_* that has the least Gini index to
partition the dataset at value *v_*.
16. While the splitting attribute *a_* at value *v_* is already the
parent of the current node at Level *L* do
17. Select the next best attribute as the splitting attribute.
18. End
19. If none of the attributes are selected then
20. Make this node a leaf node
21. Assign membership values of each class to this node.
22. End
23. If the node at Level *L* has been partitioned then
24. Decision-tree-construction (Level *L* + 1)
25. End
26. End

***Function* Evaluate-Gini-index (attribute *a*, split point value *v*)**
For each value of attribute *a* do
Find the Fuzzy membership (*a*, *v*, left)
Find the Fuzzy membership (*a*, *v*, right)
End
Evaluate the Gini_index using the fuzzy membership values of the attribute and that of the records;
***function* Fuzzy membership (attribute *a*, Split_point*v*, position pos)**
Compute standard deviation σ of the attribute *a*
Set the values of α and β
If pos==left then do
For each value *v* of attribute *a* do

$$\begin{aligned} \text{lw} &= 0.0 \\ \text{rw} &= \alpha * \sigma \\ \text{lp} &= \text{split\_point} - \beta \\ \text{rp} &= \text{split\_point} + \beta \end{aligned}$$

$$\text{fuzzy value} = \begin{cases} \frac{\text{lw}}{\text{lp}+\text{lw}-v}, & v < \text{lp} \\ 1.0, & \text{lp} \le v \le \text{rp} \\ \frac{\text{rw}}{v-\text{rp}+\text{rw}}, & v > \text{rp} \end{cases}$$

End
End
If pos===right then do
For each value v of attribute a do

$$\begin{aligned} \text{lw} &= \alpha * \sigma \\ \text{rw} &= 0.0 \\ \text{lp} &= \text{split\_point} - \beta \\ \text{rp} &= \text{split\_point} + \beta \end{aligned}$$

$$\text{fuzzy value} = \begin{cases} \frac{\text{lw}}{\text{lp}+\text{lw}-v}, & v < \text{lp} \\ 1.0, & \text{lp} \le v \le \text{rp} \\ \frac{\text{rw}}{v-\text{rp}+\text{rw}}, & v > \text{rp} \end{cases}$$

End
End

IV.DESIGN IMPLEMENTATION

Sequence diagrams are used to formulate the behavior of a system and visualize communication among objects. They are useful for identifying additional objects that Participate in the use cases. We call objects involved in the use cases as participating Objects. These represent the interaction that take place among objects.

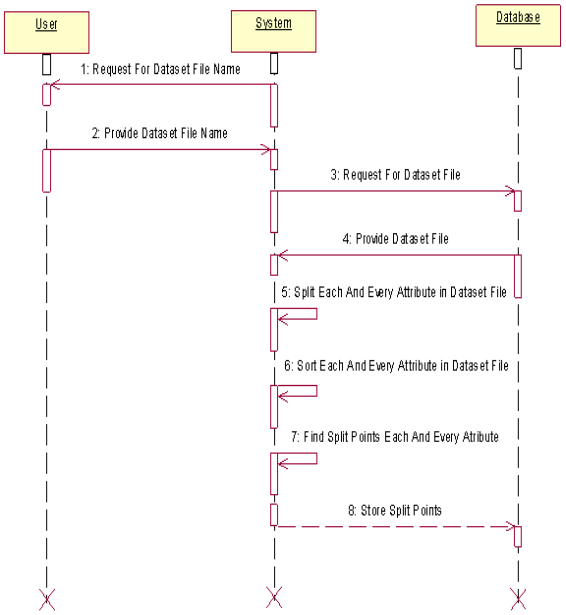
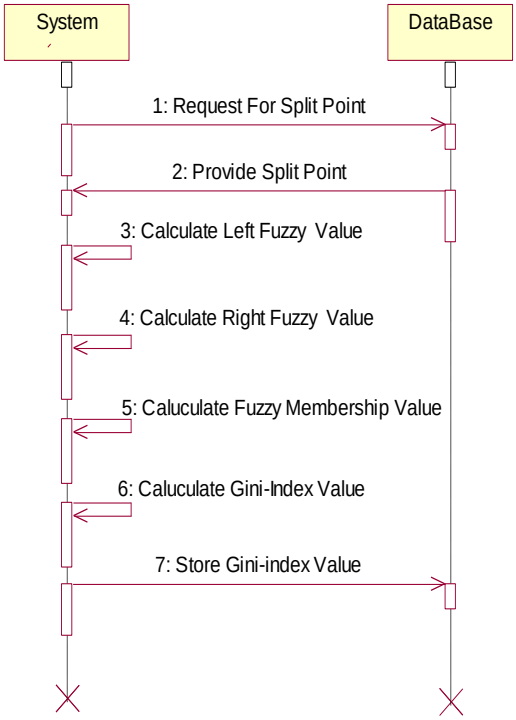


Fig 6: split-point sequence diagram.



. Fig 7: Gini-Index sequence diagram

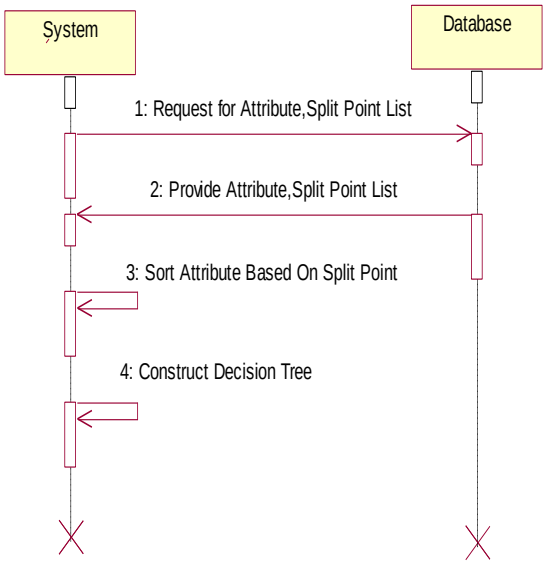


Fig 8: Decision tree sequence diagram

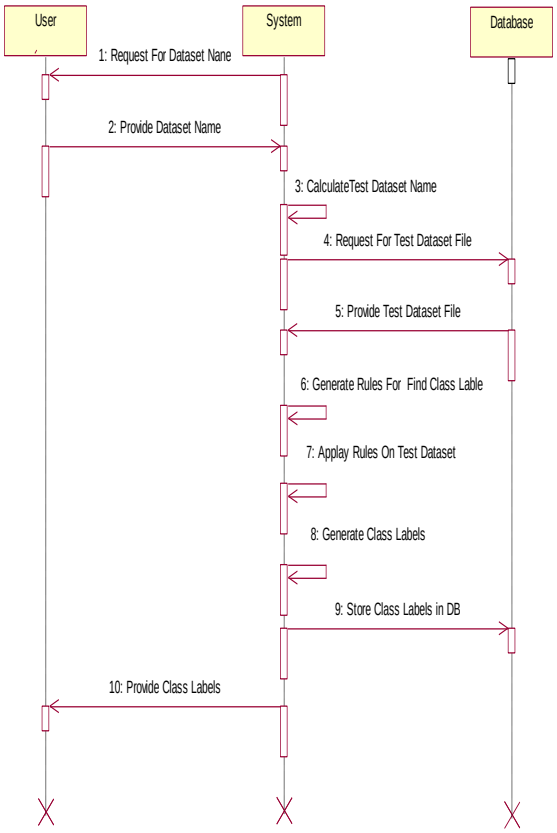


Fig 9: Class label prediction sequence diagram

V.RESULTS INPUT SCREENS

To run the project, JVM can interact with the user to ask to enter dataset name, for which we wants to generate decision tree. After the user provides dataset name, JVM can execute the project and can generate decision tree. The dataset name provided here is wine quality UCI machine learning dataset which contain eleven Attributes and seven class labels.

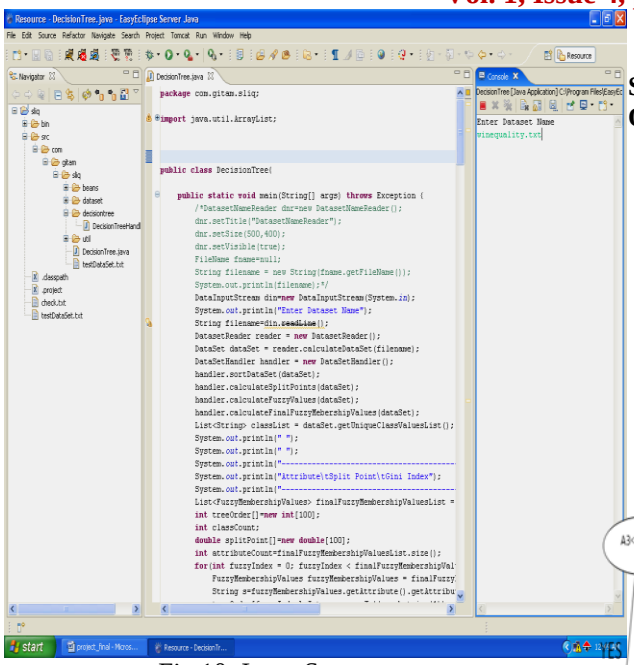


Fig 10: Input Screen

DECISION TREE OUTPUT SCREEN

After the user provides wine quality UCI machine learning dataset as input, JVM can execute the project and can generate wine quality decision tree. The generated wine quality decision tree is shown below. The generated wine quality decision tree contain thirteen nodes, twelve leaf nodes and Attribute eleven is root node.

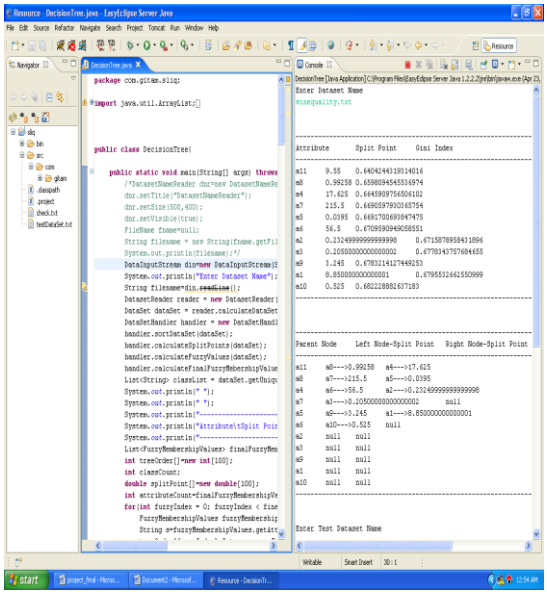


Fig 11: Decision Tree Output Screen

SLIQ DECISION TREE FOR WINE QUALITY DATASET

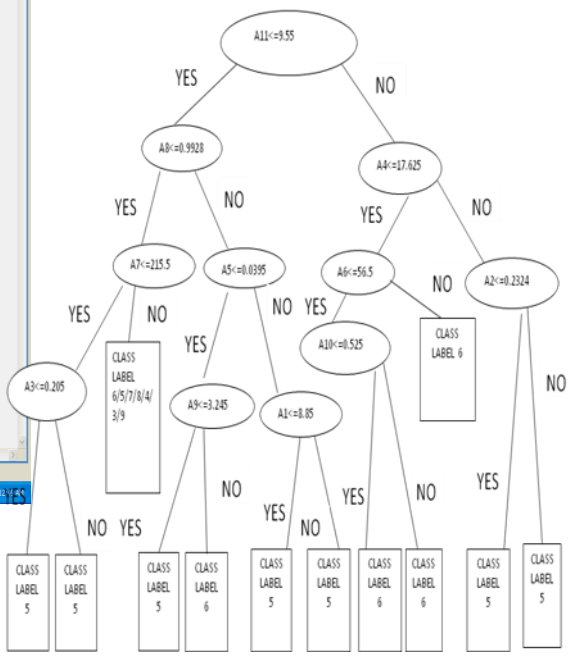


Fig 12: SLIQ wine quality Decision Tree

FUZZY-SLIQ DECISION TREE FOR WINE QUALITY DATASET

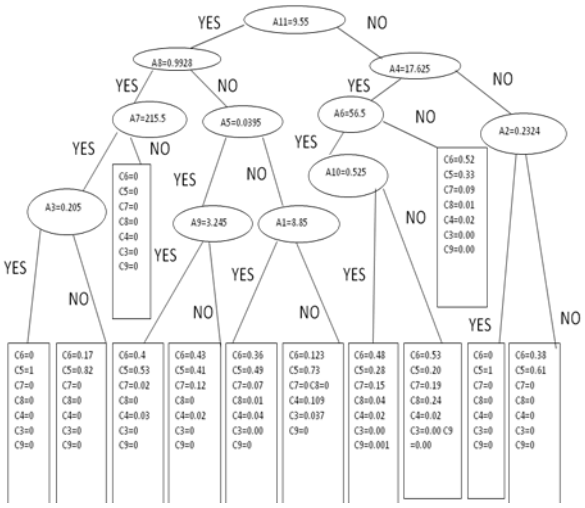


Fig 13: Fuzzy-SLIQ wine quality Decision Tree

PREDICTION OUTPUT SCREEN

After construction of decision tree, the classification rules were generated for test wine quality dataset by using Fuzzy SLIQ decision tree. By applying classification rules on test dataset, the class label can be generated. The below output screen shows predicted class label for test wine quality dataset.

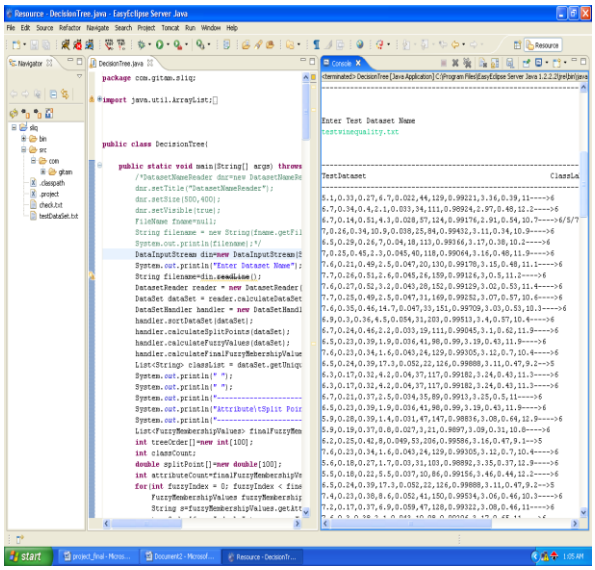


Fig 14: Prediction Output Screen

Conclusion

Over the hard binary decision tree algorithm (SLIQ),the FS-DT algorithm has provided contemptible performance .The classification accuracy has better enhancement when compared to that of SLIQ in case of FS-DT. The size of the decision tree developed using the proposed algorithm is considerably less compared to that of SLIQ. FS-DT algorithm should be worked on any Quantitative Dataset having hundred of data records. Using Online Learning System (OLS) dataset, FDT algorithm can be implemented which supports Online Learning System (OLS) dataset and attributes of dataset having quantitative values. The predictive module should be used to find missing class labels by using generated rules from FS-DT. It is able to find any length Test data set having missing class labels.

References

[1] S. Ruggieri, "Efficient C4.5 [classification algorithm]," IEEE Trans.Knowl. Data Eng., vol. 14, no. 2, pp. 438–444, Mar./Apr. 2002.

[2] J. C. Shafer, R. Agrawal, and M. Mehta, "SPRINT: A scalable parallelclassifier for data mining," in Proc. 24th Int. Conf. Very Large Data Bases,Mumbai, India, Sep. 1996, pp. 544–555.

[3] S. R. Safavian and D. Landgrebe, "A survey of decision tree classifiermethodology," IEEE Trans. Syst., Man, Cybern., vol. 21, no. 3, pp. 660–674, May/Jun. 1991.

[4] A. Suárez and J. F. Lutsko, "Globally optimal fuzzy decision trees for classification and regression," IEEE Trans. Pattern Anal. Mach. Intell.,vol. 21, no. 12, pp. 1297–1311, Dec. 1999.

[5] M. Umanol, H. Okamoto, I. Hatono, H. Tamura, F. Kawachi, S. Umedzu,and J. Kinoshita, "Fuzzy decision trees by fuzzy ID3 algorithm and itsapplication to diagnosis systems," in Proc. 3rd IEEE Conf. Fuzzy Syst.,Jun. 1994, vol. 3, pp. 2113–2118.

[6] H. Wang and C. Zaniolo, "CMP: A fast decision tree classifier using multivariate predictions," in Proc. 16th Int. Conf. Data Eng., 2000,pp. 449–460.

[7] Y. Yuan and M. J. Shaw, "Induction of fuzzy decision trees," Fuzzy Sets Syst., vol. 69, no. 2, pp. 125–139, Jan. 1995.

[8] B. Chandra and P. Pallath, "Robust algorithm for classification using decision trees," in Proc. IEEE Int. Conf. CIS-RAM, 2006, pp. 608–612.

[9] B. Chandra and P. P. Varghese, "On improving efficiency of SLIQ decision tree algorithm," in Proc. IEEE IJCNN, 2007, pp. 66–71.

[10] C. Z. Janikow, "Fuzzy processing in decision trees," in Proc. 6th Int.Symp. Artif. Intell., 1993, pp. 360–367.

[11] C. Z. Janikow, "Fuzzy decision trees: Issues and methods," IEEE Trans.Syst., Man, Cybern. B, Cybern., vol. 28, no. 1, pp. 1–14, Feb. 1998.

Authors :

Mr. Rambabu Pemula received B.Tech(CSE),M.Tech from JNTUH and M.B.A from ANU, Guntur. He is a research scholar in the field of Data Mining and Software Engineering. Presently he is working as Assistant Professor in Nimra Institute of Engineering & Technology, Ongole, Andhra Pradesh, India. He is having 5+ years of teaching experience in the field of Computer Science & Engineering. He can be reached at rpemula@gmail.com

Mr. Srinivasa Rao Divvela received B.Tech(CSIT) from JNTUH and M.Tech(CSE) from JNTUA. He is a research scholar in the field of Data Mining and Software Engineering. Presently he is working as Associate Professor in SSN College of Engineering, Ongole, Andhra Pradesh, India. He is having 6+ years of teaching experience in the field of Computer Science & Engineering. He can be reached at srinumtechcse2007@gmail.com

Mr. Bhupati Sampath Babu received B.Tech(CSE) from JNTUH and M.Tech(CSE) from JNTUA. He is a research scholar in the field of Data Mining and Software Engineering. Presently he is working as Assistant Professor in SSN College of Engineering, Ongole, Andhra Pradesh, India. He is having 5+ years of teaching experience in the field of Computer Science & Engineering. He can be reached at sampathbabu.b@gmail.com

Mr. Bommala Harikrishna received B.Tech(CSE) from JNTUk and M.Tech(CSE) from ANU. He is a research scholar in the field of Data Mining. Presently he is working as Assistant Professor in A1 Global Institute of Engineering & Technology, Markapur, Prakasm, Andhra Pradesh, India. He is having 1+ years of teaching experience in the field of Computer Science & Engineering. He can be reached at haribommala@gmail.com

Neelima Sadineni received Bachelors degree in Computer science and Information Technology from JNTUH, Pursuing M.Tech in Software Engineering from JNTUK. She is a research scholar in field of Data Mining and Software Engineering. She is having experience of 5 Years in the field of Computer Science and Engineering, presently working as Assistant Professor in the department of CSE, Pydah College of engineering Kakinada, A.P,INDIA. She can be reached at