

Extended algorithms for mining frequent patterns using Decision tree in R

P. Regina¹, S.Chinaramu², M.V.Ramana Murthy³, C.R.K.Reddy⁴

¹Research Scholar, Dept. of CSE, Dravidian University,
Kuppam, Aurora's PG college, Uppal, Hyderabad-98

²Dept. of CSE, CBIT, Hyderabad-75

^{3,4}Dept. of Mathematics, CSE MGIT Hyderabad-75,

Former: Dept.of Mathematics & Computer Science, Osmania University

Abstract- For mining frequent patterns, the datasets are classified into different classes. Classification is an important problem in data mining. For a database with large number of records, classification decides to which class a particular record belongs. Decision tree is a classification scheme, which is a best and mostly used supervised learning method that generates a tree and a set of rules representing the model of different classes from a given data set. Tree based algorithms endow predictive models with high accuracy, performance and ease of interpretation. The set of records are divided into two disjoint subsets namely training and testing data sets, former is used to identify the classifier and later for accuracy of the classifier. Greedy strategy is followed to grow decision tree by making a series of locally optimum decisions¹. Decision tree is a graphical representation of choices and their results in form of a tree. The nodes in the graph represent an event or choice and the edges of the graph represent the decision rules or conditions. It is mostly used in Machine Learning and Data Mining applications, which is implemented in a data analytical tool R. By developing visualization decision rules for prediction recursive partitioning is achieved which is fundamental tool in data mining to identify frequent patterns.

Keywords: Decision tree, classification, frequent pattern.

I. INTRODUCTION

Decision tree is a type of supervised learning algorithm that can be used in both regression and classification problems. Decision trees work for numerical attributes and categorical attributes. Attributes whose domain is numeric is called as numerical attributes and whose domain is non-numerical is called as categorical attributes. They provide a clear indication of which attributes are important for prediction or classification. There is one distinguishing attribute called the class label. They generate rules that can be understood easily. The decision tree has nodes where each leaf node is assigned a class label and non-terminal nodes including root node represent attributes and test conditions or rules. Decision tree is a model of the dataset and used to predict the class label for new records. It starts at the root node and at each edge a decision is taken whether to follow that edge or not, depending on its state.²

II. CONSTRUCTION OF THE DECISION TREE

Decision tree construction initially starts with a root node that represents all the records in the training data set. Next they recursively partition the records into each node of the tree and for each partition they create a child to represent it.

Algorithm to partition the data³:

Partition (Data D_i)

1. If all records belong to only one class then the tree said to be homogeneous and it returns one class C_i .
2. Else the tree is non homogeneous and is split for each Attribute A to evaluate it.
3. Use best split found to partition data 'd' into D_1, D_2, \dots, D_n
4. 4. for each D_i : Partition D_i .
5. 5. If the tree contains no cases then it is called trivial and has a leaf but the class to which the leaf belongs must be determined from Information other than tree.

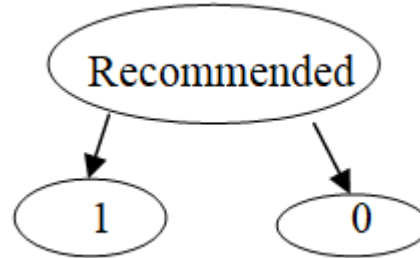
Generally, all decision trees examine only one attribute at a time so that the splitting is done based on a single attribute at any given time.

III. DESIGN ISSUES OF DECISION TREE

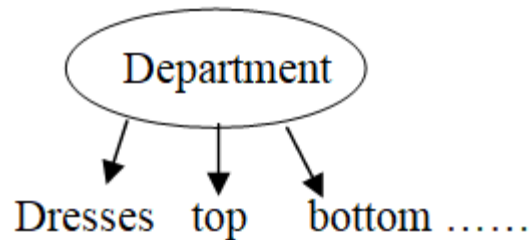
3.1 Splitting of training records

In the recursive procedure, calls to the tree growing algorithms are recursively made to split the records into smaller subsets based on the best attribute test condition.

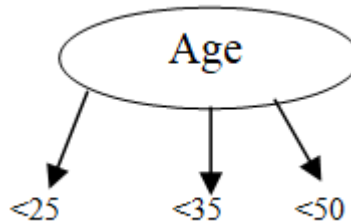
Test condition for binary attributes generates two potential outcomes.



For nominal attributes the test condition generates multiway split as there are many values.



Ordinal attributes can produce binary or multiway splits. For continues attributes the test condition can be expressed as a comparison test with binary outcomes like $A >= v$ or $A <= v$ or a range of values like $v_i <= A < v_{i+1}$.



3.2 Splitting procedure termination

A tree splitting procedure has a stopping condition to terminate the tree growing process by expanding a node until all the records belong to the same condition or have identical attribute values⁴.

IV. MEASURES FOR SELECTING THE BEST SPLIT

There are many kinds of measures to determine the best way to split the records in data dataset. These measures specify the distribution of records before and after the split.

Let $p(i)$ denote the fraction of records that belong to class C_i at a given node t . The measures selected for best split are often based on the degree of impurity of the child nodes. The smaller the impurity the more skewed the class distribution be. For example the tree with class distribution (0,1) has zero impurity and a tree with uniform distribution (0.5,0.5) has highest impurity. The measures are listed below:

Entropy:

It measures the goodness of a split.

Gini index: it measures the index of diversity for a set of records. High index indicates even distribution of classes and low index indicates single class predominance.

$$Gini = 1 - \sum_{i=1}^c (p_i)^2$$

Classification error: The lower the classification error rate the better the learning.

$$CE=1-\max[p(i)/t] \quad 5.$$

Partitioning the data: Entire dataset is partitioned into two data set: the training data set and test data set. Training data set is used for supervised learning and test data set is used for unsupervised learning.

The decision tree construction has three main phases6:

Construction phase: the initial decision tree is constructed based on the entire training data set. It recursively partitions the training data set using splitting criteria until the stopping condition is met.

Pruning phase: if the constructed tree does not produce the best possible results due to over fitting some of the lower braches are pruned.

Processioning the pruned tree.

V. ANALYSIS OF THE DECISION TREE WITH SAMPLE DATA

Example dataset is womens_clothing.csv file. It has 10 attributes and 199 records. There are 14 class labels in the dataset. Let the class labels to which each record belongs be c1, c2,c3,.....,cn. In the table below the class label Blouses is taken as c1 and dresses as c2 and so on.

Blouses	c1
Dresses	c2
Fine gauge	c3
Intimates	c4
Jackets	c5
Knits	c6
Lounge	c7
Outerwear	c8
Pants	c9
Skirts	c10
Sleep	c11
Sweaters	c12
Swim	c13
Trend	c14

Table 1

There are 199 samples out of which 44 samples are classified under c1 class, 38 samples are classified under tc2 class and so on.

S	C1	C2	C3	C4	C5	C6	C7	C8	C9.....
199	44	38	15	6	4	43	5	8	7....

Table 2

Decision tree is generated for the entire training dataset and the tree generated is given below.

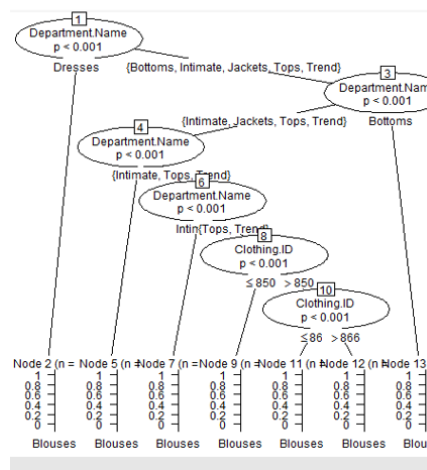


Figure1

Decision can evaluate only one attribute at a time. It decides the best splitting attribute at every node based on the measurements mentioned above.

At the root node based on the department name a decision is made. If the probability of selecting a dress is less than 0.001 then it is classified as Blouse and put in node 2. If the probability of selecting dresses is greater than 0.001 then again splitting, decision is made at the node 3. If the probability of selecting Bottoms is greater than 0.001 then it is a blouse and we reach terminal node 13 else we move to a another decision node 4. Similar process will go on until all records are classified. The above decision tree is generated without pruning.

For generating a decision tree the formula that is used is:

```
Dtree<-rpart(class ~.,method="class",data=s_train)
```

VI. PRUNING THE DECISION TREE

```
Dtree<-rpart(class ~.,method="class",data=s_train control =
rpart.control(cp=.002,minsplit=5,minbucket=5,maxdepth=10)
```

Controls are used to prune the tree.

cp: it is complexity parameter. It is mainly used to set constraints for overfitting. It decides the quality of the split. Its value has to be carefully selected. If its value is too high over fitting occurs, if its value is too low under fitting occurs.

Overfitting: When the tree becomes too large, the test errors still grow as the training errors decrease.

Underfitting: When the tree size is small test errors and training error rates are high.

minsplit: number samples considered at every node. If sample size is 'x' it will not split the node, if greater then 'x' then the node will be split.

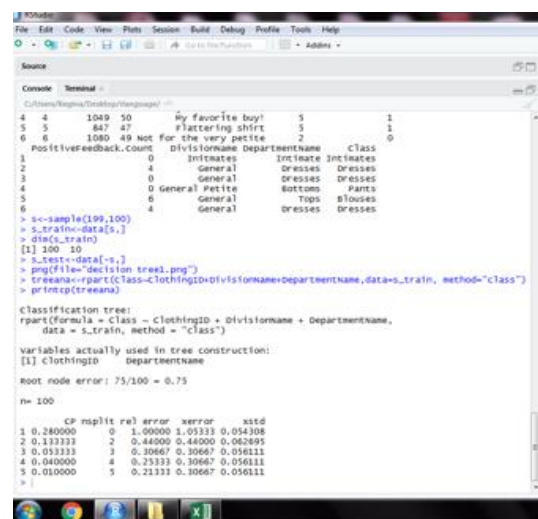
minbucket: minimum size of the bucket at the terminal node.

mindepth: minimum depth of the tree.

Objectives and starting the second loop (iteration) is the subject of an extended study of this research. However, it is anticipated that two loops (in addition to the first one) are needed to tune the anticipated outcomes in a reflective manner. In addition, assessment of the need for a further loop needs to be decided at the end of a current loop should further iterations be required.

VII. EVALUATING THE RESULTS OF THE DECISION TREE

Printcp(treana): printcp is a function used to print the cp values for number of splits in the decision tree. The output of the sample is displayed below.



```

C:\Users\Bipin\Desktop>Rstudio

Source
Console Terminal
C:\Users\Bipin\Desktop>Rstudio

4 4 1049 50 My favorite buy! 5 1
5 5 847 47 Flattering shirt 5 1
6 6 1080 49 Not For the very petite 2 0

positivefeedback.count Divisionname Departmentname class
1 0 Intimates Intimates Intimates
2 4 General Dresses Dresses
3 0 General Dresses Dresses
4 0 General Petite Bottoms Pants
5 6 General Bottoms Tops Blouses
6 4 General dresses Dresses

> s<-sample(199,100)
> s_train<-data[s,]
> dia<-train
[1] 100 50
> s_test<-data[-s,]
> png(file="decision tree1.png")
> treana<-rpart(class~clothID+Divisionname+Departmentname,data=s_train, method="class")
> printcp(treana)

Classification tree:
rpart(formula = class ~ clothID + Divisionname + Departmentname,
      data = s_train, method = "class")

Variables actually used in tree construction:
[1] clothID Departmentname

root node error: 75/100 = 0.75
n= 100

CP nsplit rel error xerror xstd
1 0.280000 0 1.00000 1.05133 0.034308
2 0.131333 2 0.44000 0.44000 0.002095
3 0.053333 3 0.30667 0.30667 0.036111
4 0.040000 4 0.25333 0.30667 0.056111
5 0.010000 5 0.21333 0.30667 0.056111

```

Figure 2

Root node error is the root splitting error which is given to be 0.75 when the cp value is taken as 0.002. The cp value should be taken in a such way that it reduces the cross validation error rate. The cross validation values are given for the entire depth of the tree from which minimum cp value is taken and the tree is constructed once again. The tree generated is given below. It has 8 terminal nodes and xerror which is cross validation error is computer to be zero.

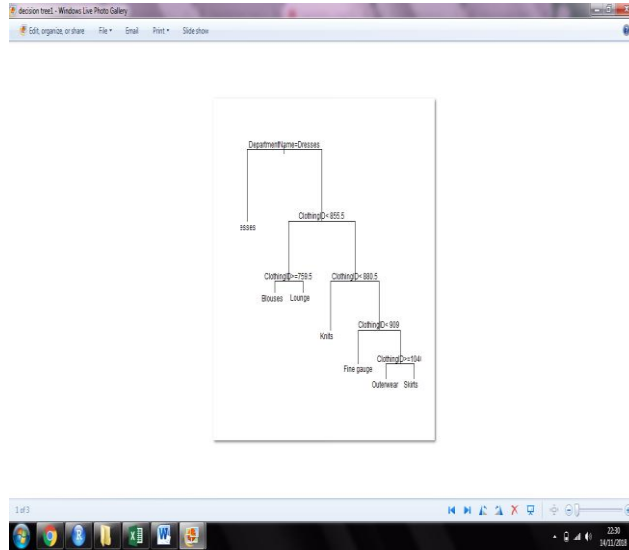


Figure 3

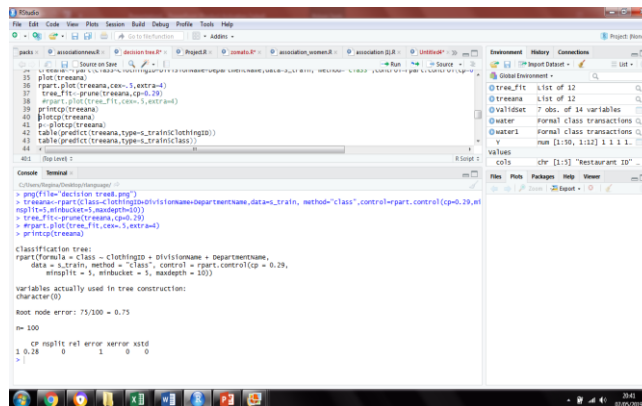


Figure 4

Analysis:

If the department name is dresses it is classified as dresses else it goes to another decision making node, and based on the clothingid if id <855.5 then goes to the right and again decision tree is splitting takes place and they are classified as blouses or lounge. Likewise the entire tree is constructed.

To get the accurate results the tree is still pruned further by recompute the cp value:

Minimum xerror value is taken and added to the standard deviation to select appropriate cp value. The newly computed value is 0.0356. Then the tree is generated based the new value. It is a pruned tree where the level of the tree given in figure 13 is reduced by one level. This reduces the computing time and increases performance.

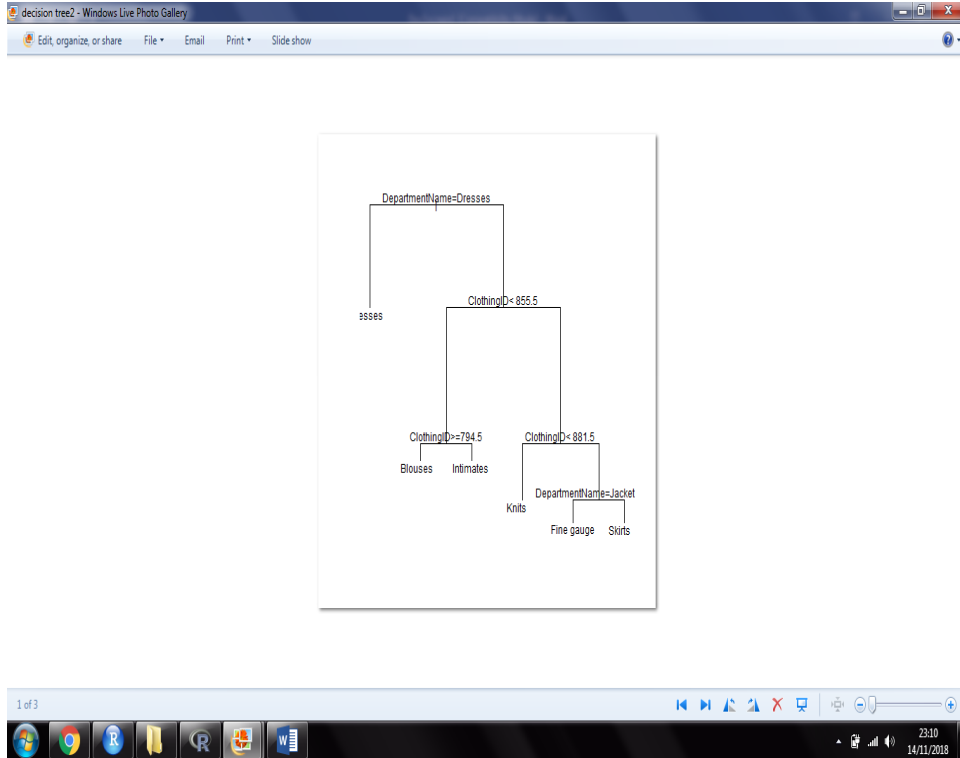


Figure 5

The tree depth is still reduced by one level.

VIII. EVALUATING THE PERFORMANCE OF THE CLASSIFIER

The estimated error helps in learning algorithm model selection. i.e to find the model of the right complexity that is not susceptible to overfitting. Once of the methods used is cross validation.

Cross validation: It is an alternative to random sampling. In this approach each record is used same number of times for training and exactly once for testing. If the data is divided into two equal subsets and if one subset is used once for training and the other for test and next if these two are swapped then it is called two-fold validation.

The optimal tree is obtained when the tree size is 4 nodes to seven nodes which is depicted in the figure below.

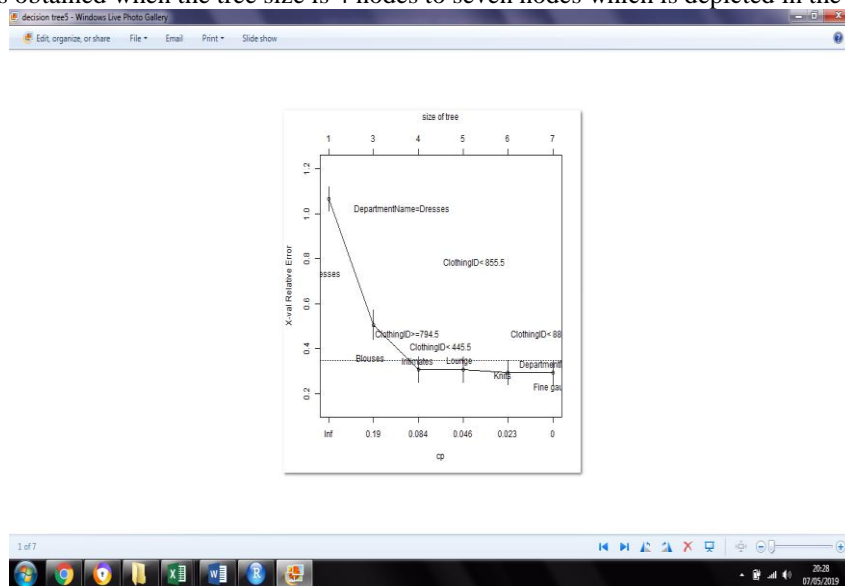


Figure 6

In the experiment if cp is taken minimum then the cross validation is generated will have xerror rate as 0.

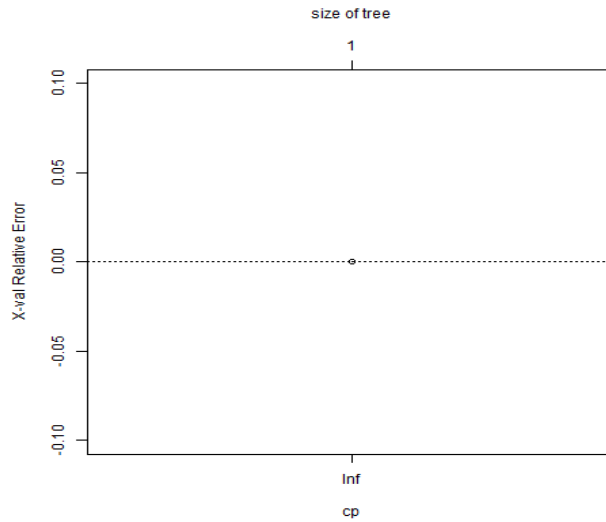


Figure 7

IX. CONCLUSION

Decision tree is a non-parametric approach for building classification models. Smaller decision trees are easy to interpret. The accuracies of the trees comparable with other classification techniques. The classification of the data for women clothing is done based on the class attribute with less error rate. When the confusion matrix was generated the over all error was 28.6% as shown below.

```

Predicted
Actual 1  2  3  4  5 Error
1 0  0.0 0.0 0  0.0 NaN
2 0 10.7 0.0 0  0.0  0
3 0  0.0 3.6 0  3.6  50
4 0  0.0 0.0 0 25.0 100
5 0  0.0 0.0 0 57.1  0

Overall error: 28.6%, Averaged class error: 37.5%

Rattle timestamp: 2018-11-03 16:23:40 Fr.Johannes
    
```

X. REFERENCES

- [1] Data mining techniques by Arun K Pujari 2007 pp 155-156
- [2] Data mining by Vikram Pudi 2012 pp 102
- [3] Data mining by Vikram Pudi 2012 pp 103
- [4] Introduction to Data mining by Pang-Ning Tan Michael Steinbach 2015 pp 155-157
- [5] Introduction to Data mining by Pang-Ning Tan Michael Steinbach 2015 pp 158
- [6] Data mining techniques by Arun K Pujari 2007 pp 160
- [7] Introduction to Data mining by Pang-Ning Tan Michael Steinbach 2015 pp 168-169