ACKNOWLEDGEMENT

I express my sincere gratitude to my respected and learned guide, Prof.Sundar S for his valuable help and guidance, I am very thankful to him for the encouragement he has given me in completing this project.

Lastly, I would like to express my deep apperception towards my classmates specially Debanjan Sarkar for advicing me to take project on computer based topic.

Sarwar Alam
(MA16C037)

Contents

1 Classification

1.1 Introduction

Classification is an important problem in the emerging field of data mining.Although classification has been studied extensively in the past most of the classification algorithms are designed only for memory resident data but thats a problem when we deal with a large data sets.This project discusses issues in building a scalable classifier and presents a the design of SLIQ(Supervised Learning in Quest) a classifier.SLIQ is a decision tree classifier that can handle both numerical and categorical attribute.

1.2 What is classification?

Classification is a data mining function that assigns items in a collection to target categories or classes. The goal of classification is to accurately predict the target class for each case in the data.

1.3 Process

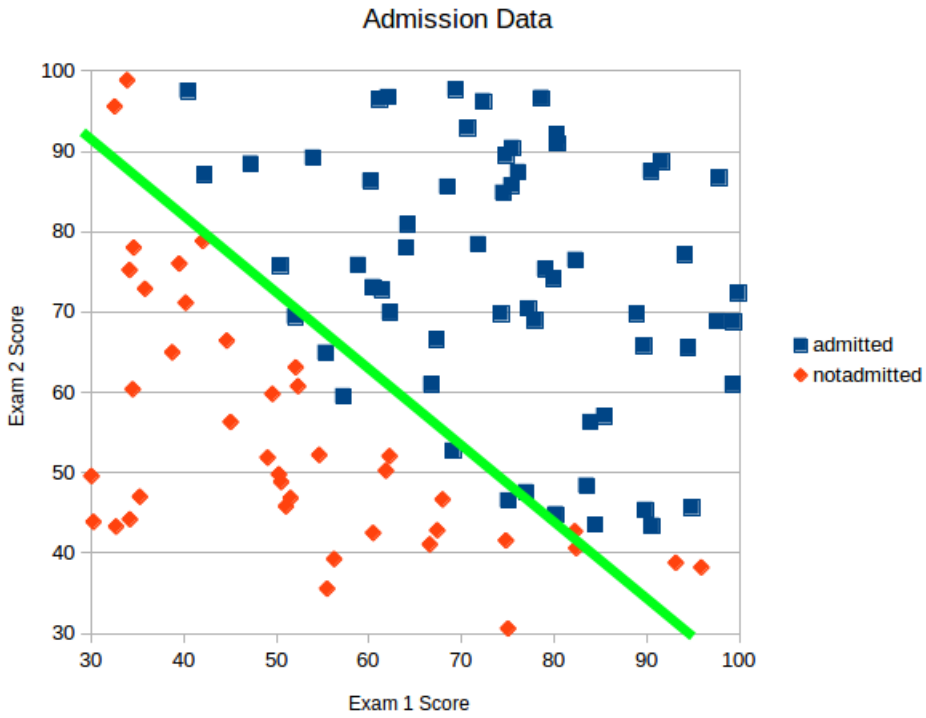

Figure 1: Exam Score.

In Figure1 we have admission data where we classified students on the basis of exam1 score and exam2 score as admitted and not admitted by a classifier of straight line.

Model Construction:

Here we are going to provide some models on how classification works.

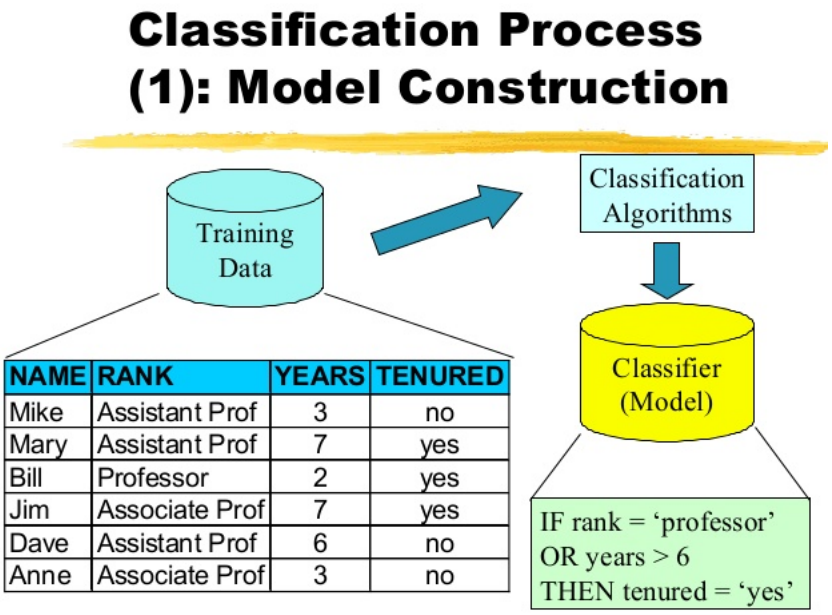

Figure 2: Model Construction

In Figure2 we defined how classification works using classification algorithm. Here the classifier classsified the data as if rank="professor" OR years>6,Then tenured ="yes" otherwise tenured="no".

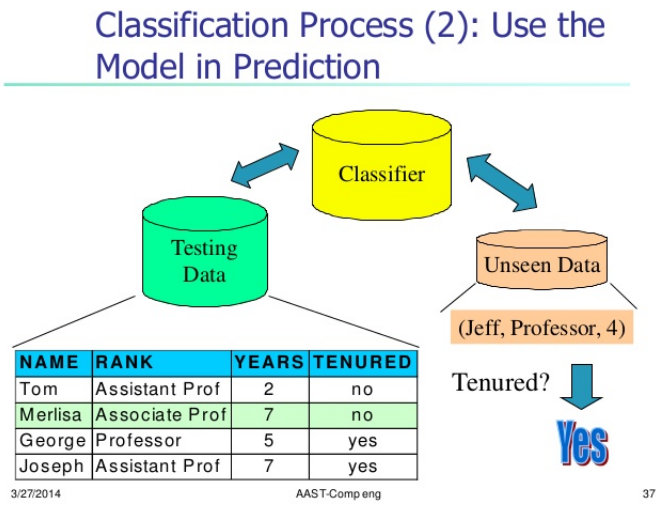

Figure 3: Testing Classifier

In Figure3 we are using testing data to test the classifier whether the classifier is valid or not.In Figure2 the classifier classified the data on the basis of professor or years.Clearly after using the testing data the classifier seems to be valid.

2 Classification by Decision Tree Induction

2.1 What is Decision Tree?

Decision tree is a flow-chart like tree structure.Internal node denotes an attribute.Branch represents the value of the node attribute.Leaf nodes represent class labels or class distribution.An example is given below.

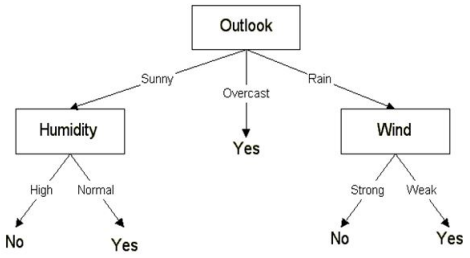

Figure 4: A Decision Tree.

2.2 Decision Tree Generation

A decision tree is grown by repeatedly partitioning the training data.

Tree Construction:

The tree construction algorithm deals with three parameters: D,attribute-list and attribute-selection-method.Here D is the training data set. attribute-list is a list of attributes and attribute-selection-method is a method for selecting the attribute according to class and it uses Information Gain or the Gini Index for selection.Now we are going to present the tree construction algorithm.

Algorithm:

1.create a node N;
2.If tuples in D are all of the same class C then
3.return N as a leaf labeled as class C;
4.If attribute-list is empty,then
5.return N as a leaf node labeled with the majority class in D;//majority voting
6.apply attribute-selection-method(D,attribute-list) to find the "best" splitting-criterion
7.labeled node N with splitting-criterion;
8.If splitting-attribute is discrete-valued and multiway splits allowed then
9.attribute list

→

attribute-list-splitting-attribute
(removing splitting-attribute)
10.for each outcome j of splitting-criterion
(partition the tuples and grow sub-trees for each partition)
11.Let D_j be the set of a data tuples in D satisfying outcome j;
(a partition)
12.If D_j is empty then
13.attach a leaf labeled with the majority class in D to node N;
14.else attach the node returned by
Generate-Decision-Tree(D_j,attribute-list)
to node N;
15.return N;

Tree Pruning:

Identify and remove branches that reflect noise or outliers.

2.3 Challenges

- Good choice of root attribute
- Good choice of the internal nodes attributes is a crucial point
- Decision Tree Induction Algorithms differ on methods of evaluating and choosing the root and internal nodes attributes.

Shortcomings of ID3 Algorithm

It is a tree building algorithm nicknamed as Greedy Algorithm constructs decision tree in a top-down recursive divide-and-conquer manner.

Scalability ??

- requires lot of computation at every stage of construction of decision tree
- needs all the training data to be in the memory
- It does not suggest any standard splitting index for range attributes.

3 Scalability Issues

3.1 Tree Building

Mainly there are two operations running during tree building.
i)evaluation of splits for each attribute and the selection of the best split.
ii)creation of partitions using the best split.
Partition can be created by simple application of the splitting criterion to the data.The most important problem is in determining the best split for each attribute.Attribute can be numeric or categorical or both.Now we are going to describe how to choose the best split.

3.2 Splitting Index(Gini Index)

- The gini index is used to evaluate the goodness of the alternative splits for an attribute
- If a data set T contains examples from n classes, gini(T) is defined as

$$gini(T) = 1 - \sum_{j=1}^{n} p_j^2$$

Where p_j is the relative frequency of class j in the data set T. After splitting T into two subset T_1 and T_2 the gini index of the split data is defined as

$$gini(T)_{split} = \frac{|T_1|}{|T|} \cdot gini(T_1) + \frac{|T_2|}{|T|} \cdot gini(T_2)$$

Advantage of the gini index:

Its calculation requires only the distribution of the class values in each record partition.

To find the best split point for a node, the node's attribute lists are scanned to evaluate the splits for the attributes. The attribute containing the split point with the lowest value for the gini index is used for splitting the node's records.

Find reduction in impurity using formula:

$$\Delta gini(split) = gini(T) - gini_{split}(T)$$

- The best split of an attribute will be selected on the basis of minimum gini index.
- After choosing best split point we will move on to the attribute selection.
- Now we will select best attribute which gives the minimum gini index overall i.e with greater reduction in impurity will give the best attribute.

3.3 Splits for Numeric Attribute

A numeric attribute of the form $A \leq v$ where v is a real number.Now our job is to evaluate splits for A that will be done by sorting the training data .Let $v_1, v_2, v_3, ..., v_n$ be the sorted values of A.Between $v_i$and v_{i+1} any value will divide the set into the same two subsets.Therefore we need to examine only $n-1$ possible splits.The cost of evaluating splits for A is dominated by the cost of sorting the values.Therefore a scalability issue on the reduction of sorting costs will arise.

3.4 Splits for Categorical Attribute

Let A be a categorical attribute and S be the set of all possible values of A.Then split will contain some subset $S\prime$ of S.But if $|S| = n$,then we will have 2^n subsets and it will be very expensive for choosing the best split.Therefore we need a fast algorithm for subset selection.

3.5 Tree Pruning

The tree pruning phase examines the initial tree grown using the training data and choose the subtree with the least estimated error.There are two main approaches to calculate the error rate.One is using the originial dataset and the other is using an independent dataset for error estimation.The firts category is called the cross-validation which is done by taking the multiple samples from the training data and a tree is grown for each sample.These multiple trees are used to estimate the error rate of the subtree of the original tree.But for large data sets this technique is very much expensive.The second category divides the training data into two

parts where one part is used for building the tree and the other for pruning the tree.But if we have a large data sets then again it will be very much expensive to go through these methods.Therefore the challenge for a scalable data classifier is to use an algorithm that is fast,and leads to accurate and compact decision tree.

4 SLIQ Classifier

4.1 Features of SLIQ

- Applies to both numerical and categorical attributes.
- Builds compact and accurate trees
- Uses a pre-sorting technique in the tree growing phase and an inexpensive pruning algorithm.
- Suitable for classification of large disk-resident datasets, independently of the number of classes, attributes and records.
- Use new a data structure call class-list.

4.2 SLIQ Methodology

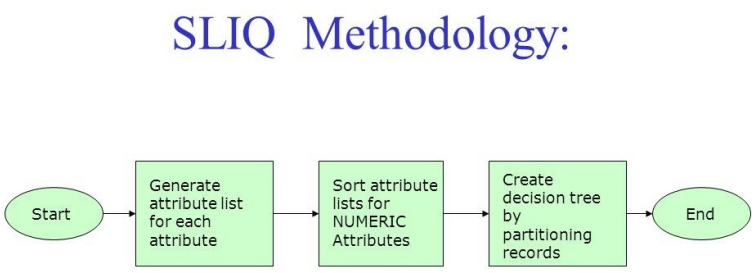

Figure 5: SLIQ Methodology

4.3 Pre-Sorting and Breadth-First Growth

- For numeric attribute the sorting is the dominant factor for finding the best split but the categorical attribute need not be sorted.
- For numeric attribute the training data sorted only once at the beginning of the tree growth phase.
- To achieve this sorting we create a separate list for each attribute of the training data and additionally,separate list,called class list is created for class labels.
- Attribute list has two fields one is attribute value and the other one is an index into the class list.Similarly class list has two fields one is class label and the other one is a reference to a leaf node.
- The ith entry in the class list corresponds to ith example in the training data.Each leaf node represents a partition of data.

An example of pre-sorting is given below:

Age	Salary	Class
30	60	G
23	15	B
40	75	G
55	40	B
55	100	G
45	60	G

Figure 6:Training Data

After Pre-Sorting

Age	Class List Index	Salary	Class List Index
23	2	15	2
30	1	40	4
40	3	60	6
45	6	65	1
55	5	75	3
55	4	100	5

Index	Class	Leaf
1	G	N1
2	B	N1
3	G	N1
4	B	N1
5	G	N1
6	G	N1

Figure 7:Age List,Salary List and Class List

Algorithm for split evaluation:

EvaluateSplits()
for each attribute A
do traverse attribute list of A
for each value v in the attribute list
do find the corresponding entry in the class list, and
hence the corresponding class and the leaf node (say L)
update the class histogram in the leaf L
if A is a numeric attribute **then**
compute splitting index for test(A$\leq v$) for leaf L
if A is a categorical attribute **then**
for each leaf of the tree
do find subset of A with best split

4.4 Processing node splits

- For computing the gini splitting index for an attribute at a node,we need the frequency distribution of class values in the data partition corresponding to the node.
- We need class histogram for computing the same because the distribution is accumulated in a class histogram.
- For a numeric attribute the class histogram is a list of the pairs of the form

$$<class, frequency>$$

- For a categorical attribute the histogram is a list of the triples of the form

$$<attributevalue, class, frequency>$$

- For each value v in the attribute list of the attribute A,we find the corresponding entry in the class list and which result the corresponding class and leaf node.
- If A is a numeric attribute we compute the splitting index for A$\leq v$ for all v for this leaf.
- If A is a categorical attribute we wait till the attribute list has been completely scanned and then find subset of A with best split.

Now we are going to provide an example on how EvaluationSplits work:

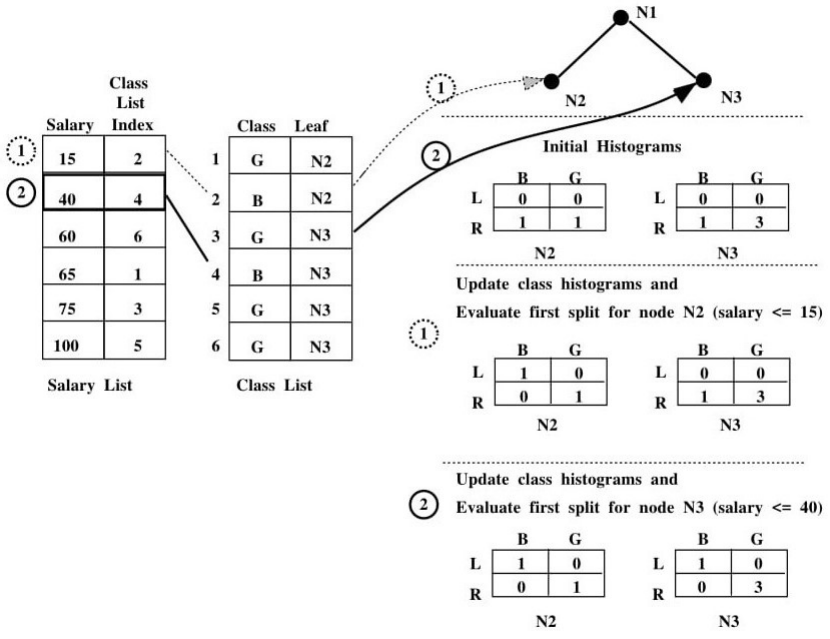

Figure 8:Evaluation Splits:An example

The above example illustrates the following:

- Initially the data has been split on the age attribute(age≤ 35)
- The class histogram consists of the distribution of the points at each leaf node
- L represents the the distribution for example that satisfy the test
- R does not
- The first value in the salary list belongs to node N_2. Therefore the first split evaluated is (salary≤ 15)
- Hence the example(salary 15,class index 2) which satisfies the predicate belongs to the left branch and the rest belong to the right branch and then class histogram of N_2 is updated
- Similarly the split (salary≤ 40) is evaluated for node N_3.After the split the example (Salary 40,class index 4) belongs to the left branch and then the class histogram of N_3 is updated.
- Now we got new example for node N_2 and for node N_3 ,the classifier again do all those things that occurred in processing node splits section again will happen for node N_2 and N_3.

4.5 Updating the class lists

Here the job is to create child nodes and update the class list

Algorithm for update class labels:

UpdateLabels()
for each attribute A used in a split
do traverse attribute list of A
for each value v in the attribute list
do find the corresponding entry in the class list(say e)
find the new class c to whcih v belongs by applying the splitting test at node referenced from e
update the class label for e to c
update node referenced in e to the child corresponding to the class c

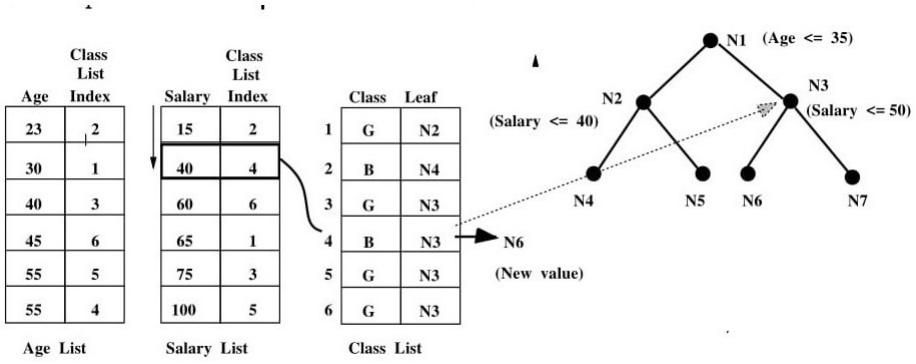

Figure 9:Class List Update:An example

The figure 8 shows that the class list being updated after nodes N_2 and N_3 have been split on the salary attribute.Here node N_2 updated to salary$\leq$ 40 and N_3 updated to salary$\leq$ 50 andN_6 becomes salary$\leq$ 40.

4.6 An optimization

While growing the tree,the two steps of splitting nodes and upadating labels are repeated until each leaf becomes a pure node and no further splits are required.The pre-sorting and breadth first growth these strategies allow SLIQ to scale large data sets with no loss in accuracy.

4.7 Subsetting for Categorical Attribute

Let A be a categorical attribute and S be the set of all possible values of A.Then split will contain some subset Sof S.But if $|S| = n$,then we will have 2^n subsets and it will be very expensive for choosing the best split.If the cardinality of S is large it will be very much expensive.But SLIQ uses a hybrid approach to overcome this issue.If the cardinality of S is less a threshold,then all the subsets of S are evaluated.Otherwise a greedy algorithm is used to obtain the desired the subset.The algorithm starts with empty subset S' and then adds on it one element of S which gives the best split .This process continues until there is no improvement in the splits.

4.8 SLIQ Pruning

SLIQ uses Minimum Description Length(MDL) principle for tree pruning.The MDL principle states that the best model for a given data set is the one that minimizes the sum of the length the encoded data by the model plus the length of the mode.If M be a model that encodes the data D,then the total cost(length) is given by
cost(M,D)= cost($D|M$)+cost(D)
where cost(M)= cost of the model(length)
(How large is the decision tree?)
and cost($D|M$)= cost to describe the data with the model.
(How many classification errors have been occurred ? that is it is sum of all classification errors)

Here we are going to provide some examples based on MDL

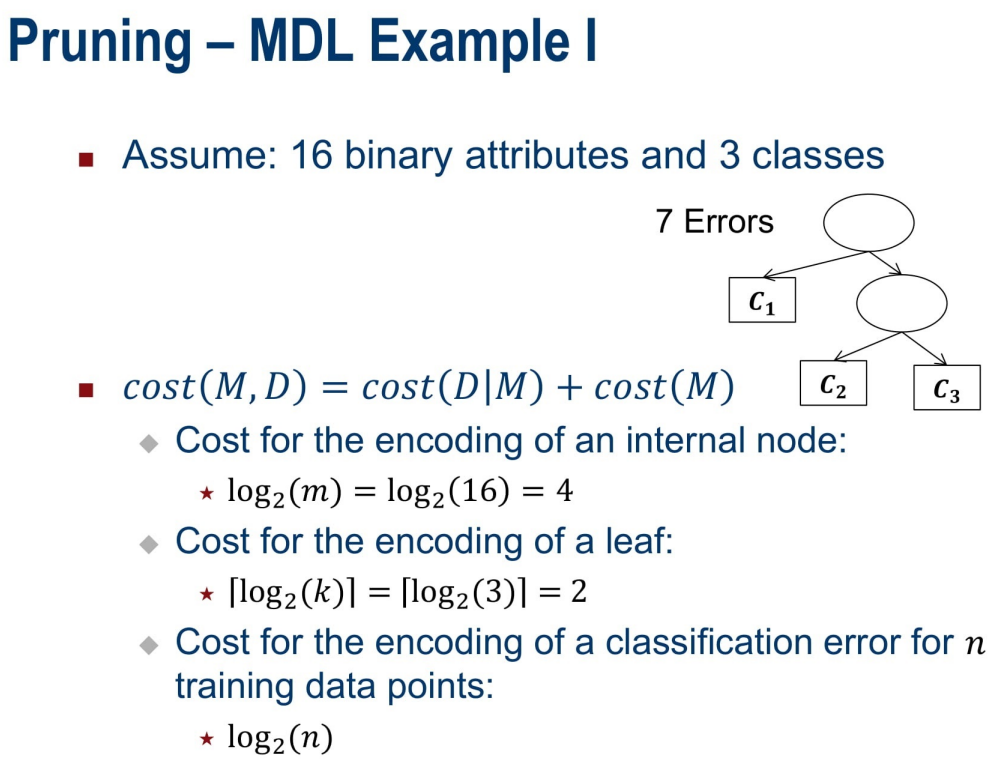

Figure 10:MDL Example 1

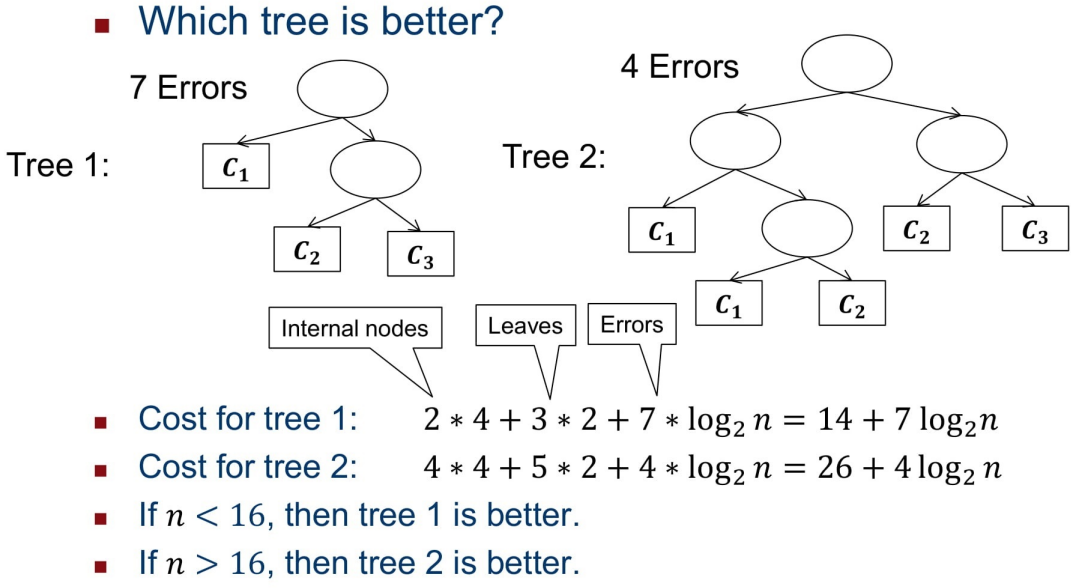

Figure 11:MDL Example 2

5 Performance

Here we are going to present classification of some real life based examples using SLIQ and comparing with other sorts of decision tree generation algorithm.

SLIQ has been tested on these data sets

Dataset	Domain	# Attributes	#Classes	#Examples
Australia	Credit Analysis	14	2	690
Diabetes	Diesease Diagnosis	8	2	768
DNA	DNA Sequencing	180	3	3186
Letter	Handwritting Recognition	16	26	20000
Satimage	Landusage Images	36	6	6435
Segment	Image Segmentation	19	7	2310
Shuttle	Space shuttle radiation	9	7	57000
Vehicle	Vehicle Identification	18	4	846

Figure 12:Real Life Based Example

Performance:Classification Accuracy,Decision Tree Size,Execution Time

Dataset	IND-Cart	IND-C4	SLIQ
Australia	85.3	84.4	84.9
Diabetes	74.6	70.1	75.4
DNA	92.2	92.5	92.1
Letter	84.7	86.8	54.6
Satimage	85.3	85.2	86.3
Segment	94.9	95.9	94.6
Shuttle	99.9	99.9	99.9
Vehicle	68.8	71.1	70.3

Figure 13:Classification Accuracy

Dataset	IND-Cart	IND-C4	SLIQ
Australia	5.2	85	10.6
Diabetes	11.5	179.7	21.2
DNA	35.0	171.0	45.0
Letter	1199.5	3241.3	879.0
Satimage	90.0	563.0	133.0
Segment	52.0	102.0	16.2
Shuttle	27	57.0	27
Vehicle	50.1	249.0	49.4

Figure 14:Pruned Tree Size

Dataset	IND-Cart	IND-C4	SLIQ
Australia	2.1	1.5	7.1
Diabetes	2.5	1.4	1.8
DNA	33.4	9.4	19.3
Letter	251.3	53.08	39.0
Satimage	224.7	37.06	16.5
Segment	30.2	9.7	5.2
Shuttle	460	80	33
Vehicle	7.62	2.7	1.8

Figure 15:Execution Time

Performance:Scalability

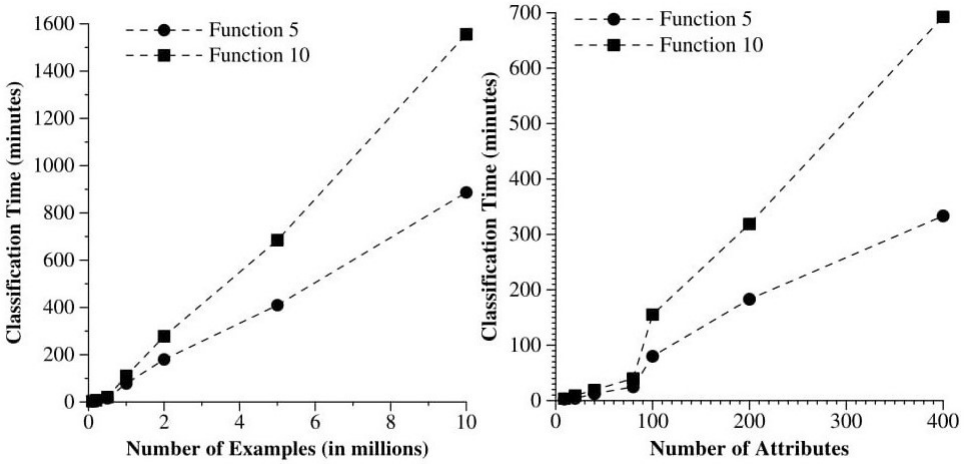

Figure 16:Scalability Demonstration

6 Conclusion

SLIQ demonstrates to be a fast,low-cost and scalable classifier that builds accurate trees.An empirical performance shows that compared to other classifier,SLIQ achieves comparable accuracy but produces small decision trees and has small classification time.

References:

@articleM.mehtaJ.RissanenR.AgarwalJ, author = "M.Mehta and J.RIssanen and R.Agarwal", title = "MDL-based decision tree pruning.International conference on Knowledge Discovery in Database and Data Mining(KDD-95)", address="Montreal,Canada", year = "Aug-1995"

The End