Decision Trees

SLIQ – fast scalable classifier

Group 12
-Vaibhav Chopda
-Tarun Bahadur

- Paper By -
Manish Mehta, Rakesh Agarwal and Jorma Rissanen
- Source –
http://citeseer.ifi.unizh.ch/mehta96sliq.html

- Material Includes: lecture notes for CSE634 – Prof. Anita Wasilewska
http://www.cs.sunysb.edu/~cse634

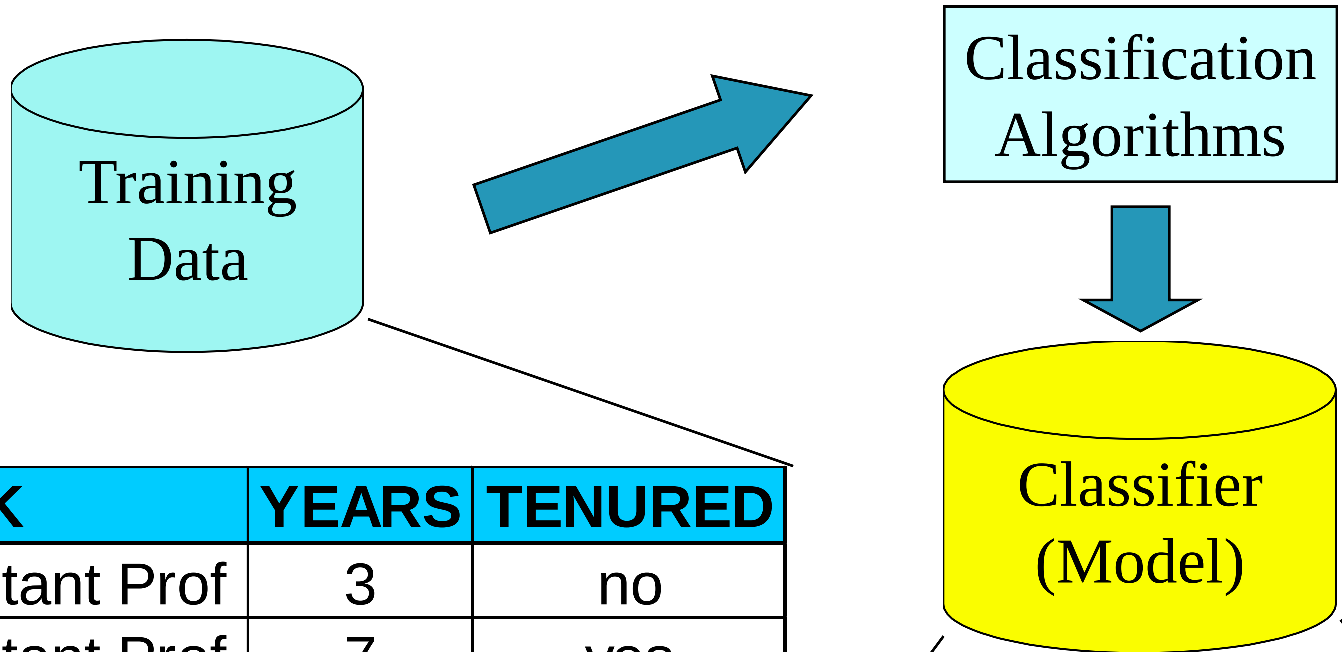
References

1. R. Agrawal, T. Imielinski, and A. Swami. Database mining: A performance perspective. *IEEE Trans. on Knowledge and Data Engineering*, 5(6), Dec. 1993.
2. J. Catlett. *Megainduction: Machine Learning on Very Large Databases.* PhD thesis, University of Sydney, 1991.
3. P. K. Chan and S. J. Stolfo. Meta-learning for multistrategy and parallel learning. In *Proc. Second Intl. Workshop on Multistrategy Learning*, pages 150–165, 1993.
4. L. Breiman et. al. *Classification and Regression Trees.* Wadsworth, Belmont, 1984.
5. R. Agrawal et. al. An interval classifier for database mining applications. In *Proc. of the VLDB Conf.*, Vancouver, British Columbia, Canada, August 1992.
6. M. Mehta, J. Rissanen, and R. Agrawal. MDL-based decision tree pruning. In *Int'l Conf. on Knowledge Discovery in Databases and Data Mining (KDD-95)*, Montreal, Canada, Aug. 1995.
7. D. Michie, D. J. Spiegelhalter, and C. C. Taylor. *Machine Learning, Neural and Statistical Classification.* Ellis Horwood, 1994.
8. NASA Ames Res. Ctr. *Intro. to IND Version 2.1*, GA23-2475-02 edition, 1992.
9. J. R. Quinlan and R. L. Rivest. Inferring decision trees using minimum description length principle. *Information and Computation*, 1989.
10. J. Ross Quinlan. *C4.5: Programs for Machine Learning.* Morgan Kaufman, 1993.
11. J. Rissanen. *Stochastic Complexity in Statistical Inquiry.* World Scientific Publ. Co., 1989.
12. C. Wallace and J. Patrick. Coding decision trees. *Machine Learning*, 11:7–22, 1993.
13. S. M. Weiss and C. A. Kulikowski. *Computer Systems that Learn: Classification and Prediction Methods from Statistics, Neural Nets, Machine Learning, and Expert Systems.* Morgan Kaufman, 1991.

Agenda

- What is classification ...
- Why decision trees ?
- The ID3 algorithm
- Limitations of ID3 algorithm
- SLIQ – fast scalable classifier for DataMining
- SPRINT – the successor of SLIQ

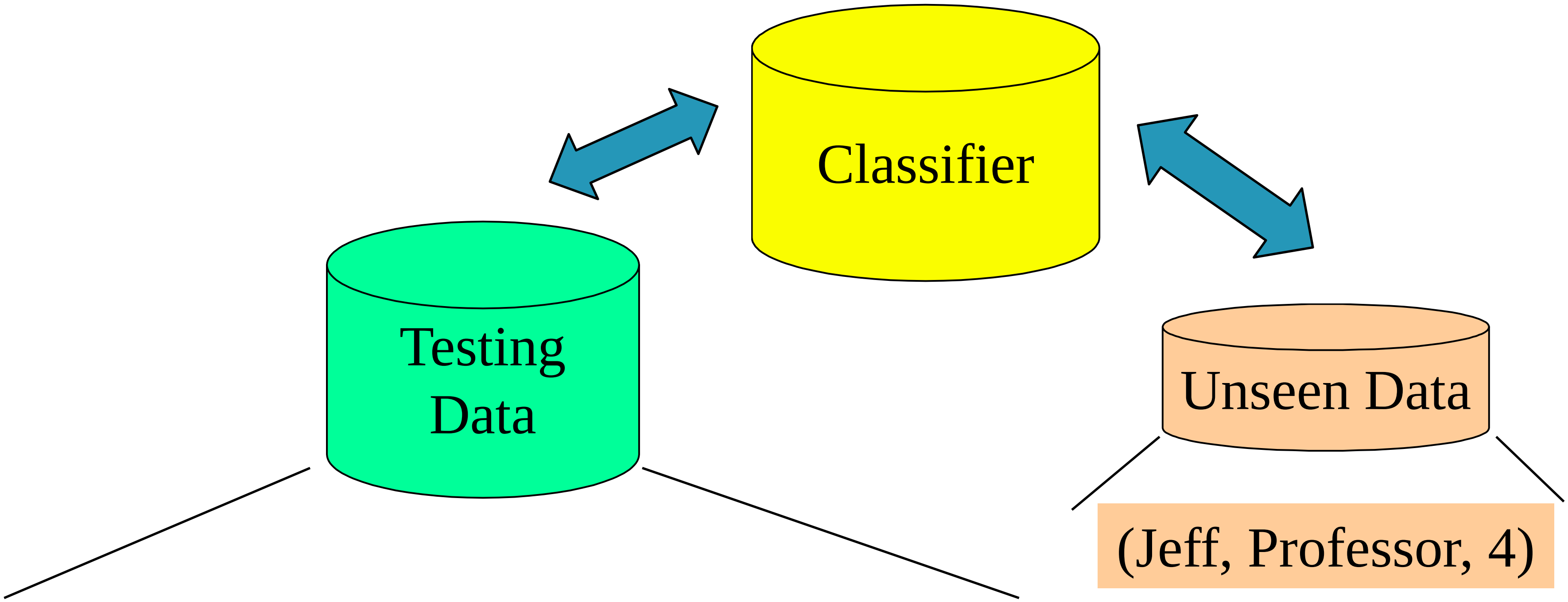
Classification Process : Model Constructio



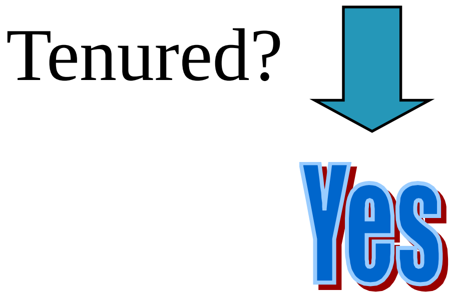
NAME	RANK	YEARS	TENURED
Mike	Assistant Prof	3	no
Mary	Assistant Prof	7	yes
Bill	Professor	2	yes
Jim	Associate Prof	7	yes
Dave	Assistant Prof	6	no
Anne	Associate Prof	3	no

IF rank = ‘professor’
OR years > 6
THEN tenured = ‘yes’

Testing and Prediction (by a classifier)



NAME	RANK	YEARS	TENURED
Tom	Assistant Prof	2	no
Merlisa	Associate Prof	7	no
George	Professor	5	yes
Joseph	Assistant Prof	7	yes



Classification by Decision Tree Induction

- Decision tree (Tuples flow along the tree structure)
  - Internal node denotes an attribute
  - Branch represents the values of the node attribute
  - Leaf nodes represent class labels or class distribution

Classification by Decision Tree Induction

- **Decision tree generation** consists of two phases
  - **Tree construction**
    - At start we choose one attribute as the root and put all its values as branches
    - We choose recursively internal nodes (attributes) with their proper values as branches.
    - We Stop when
      - all the samples (records) are of the same class, then the node becomes the leaf labeled with that class
      - or there is no more samples left
      - or there is no more new attributes to be put as the nodes. In this case we apply MAJORITY VOTING to classify the node.
  - **Tree pruning**
    - Identify and remove branches that reflect noise or outliers

Classification by Decision Tree Induction

- **Wheres the challenge ?**
- **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.

Basic Idea of ID3/C4.5 Algorithm

- greedy algorithm
- constructs decision trees in a top-down recursive divide-and-conquer manner.

- Tree STARTS as a single node (root) representing all training dataset (samples)
- IF the samples are ALL in the same class, THEN the node becomes a LEAF and is labeled with that class
- OTHERWISE, the algorithm uses an entropy-based measure known as *information gain* as a heuristic for selecting the ATTRIBUTE that will **BEST** separate the samples into individual classes. This attribute becomes the node-name (test, or tree split decision attribute)

Basic Idea of ID3/C4.5 Algorithm (2)

- A branch is created for each value of the node-attribute (and is labeled by this value -this is syntax) and the samples are partitioned accordingly (this is semantics; see example which follows)
- The algorithm uses the same process recursively to form a decision tree at each partition. Once an attribute has occurred at a node, it need not be considered in any other of the node’s descendents
- The recursive partitioning **STOPS** only when any one of the following conditions is true.
- All records (samples) for the given node belong to the same class or
- There are no remaining attributes on which the
- Records (samples) may be further partitioning. In this case we convert the given node into a LEAF and label it with the class in majority among samples (*majority voting*)
- There is no records (samples) left – a leaf is created with majority vote for training sample

Example from professor Anita's slide

This follows an example from Quinlan's ID3

age	income	student	credit_rating
<=30	high	no	fair
<=30	high	no	excellent
30…40	high	no	fair
>40	medium	no	fair
>40	low	yes	fair
>40	low	yes	excellent
31…40	low	yes	excellent
<=30	medium	no	fair
<=30	low	yes	fair
>40	medium	yes	fair
<=30	medium	yes	excellent
31…40	medium	no	excellent
31…40	high	yes	fair
>40	medium	no	excellent

Shortcommings of ID3

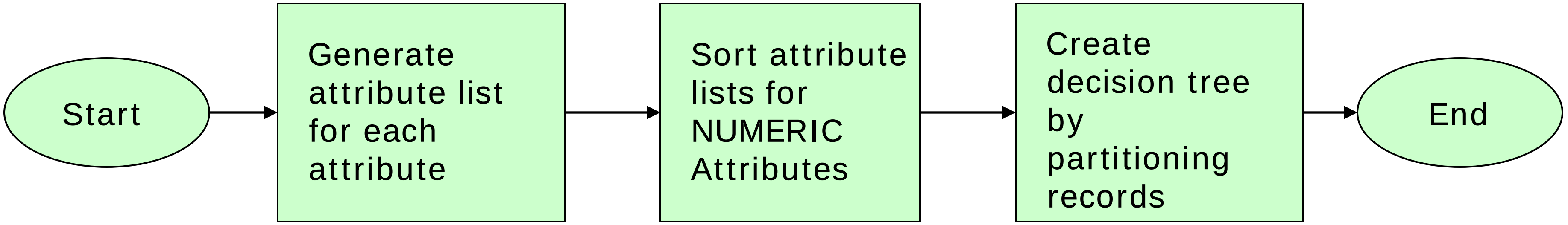
- Scalability ?
  requires lot of computation at every stage of construction of decision tree
- Scalability ?
  needs all the training data to be in the memory
- It does not suggest any standard splitting index for range attributes

SLIQ - a decision tree classifier

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

SLIQ Methodology:



Example:

Drivers Age	**CarType**	**Class**
23	Family	HIGH
17	Sports	HIGH
43	Sports	HIGH
68	Family	LOW
32	Truck	LOW
20	Family	HIGH

Attribute listing phase :

Rec Id	Age	CarType	Class
0	23	Family	HIGH
1	17	Sports	HIGH
2	43	Sports	HIGH
3	68	Family	LOW
4	32	Truck	LOW
5	20	Family	HIGH

Age	Class	RecId
23	HIGH	0
17	HIGH	1
43	HIGH	2
68	LOW	3
32	LOW	4
20	HIGH	5

Age – NUMERIC attribute

Rec Id	CarType	Rec Id
Family	HIGH	0
Sports	HIGH	1
Sports	HIGH	2
Family	LOW	3
Truck	LOW	4
Family	HIGH	5

CarType – CATEGORICAL attribute

Presorting Phase:

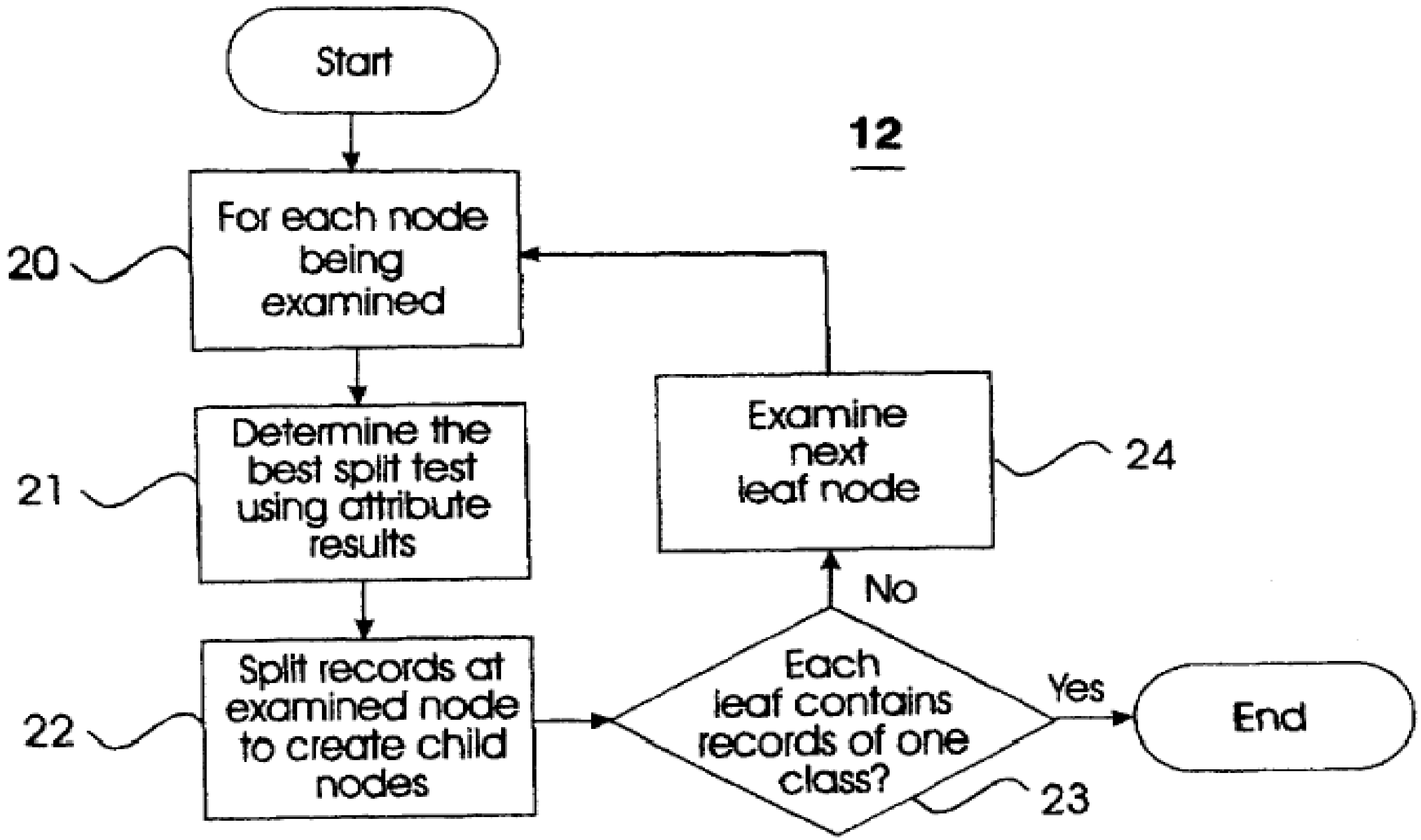
Age	Class	RecId
17	HIGH	0
20	HIGH	5
23	HIGH	0
32	LOW	4
43	LOW	2
68	HIGH	3

Only NUMERIC attributes sorted

CarType	Class	Rec Id
Family	HIGH	0
Sports	HIGH	1
Sports	HIGH	2
Family	LOW	3
Truck	LOW	4
Family	HIGH	5

CATEGORICAL attribute need not be sorted

Constructing the decision tree



Constructing the decision tree

- (block 20) for each leaf node being examined, the method determines a split test to best separate the records at the examined node using the attribute lists in block 21.
- (block 22) the records at the examined leaf node are partitioned according to the best split test at that node to form new leaf nodes, which are also child nodes of the examined node.
- The records at each new leaf node are checked at block 23 to see if they are of the same class. If this condition has not been achieved, the splitting process is repeated starting with block 24 for each newly formed leaf node until each leaf node contains records from one class.

  In finding the best split test (or split point) at a leaf node, a splitting index corresponding to a criterion used for splitting the records may be used to help evaluate possible splits. This splitting index indicates how well the criterion separates the record classes. The splitting index is preferably a gini 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

$$gini(T) = 1 - \sum p_j^2$$

Where pj is the relative ferquency of class j in the data set T.

- After splitting *T* into two subset *T1* and *T2* the *gini* index of the split data is defined as

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

Gini Index : The preferred splitting index

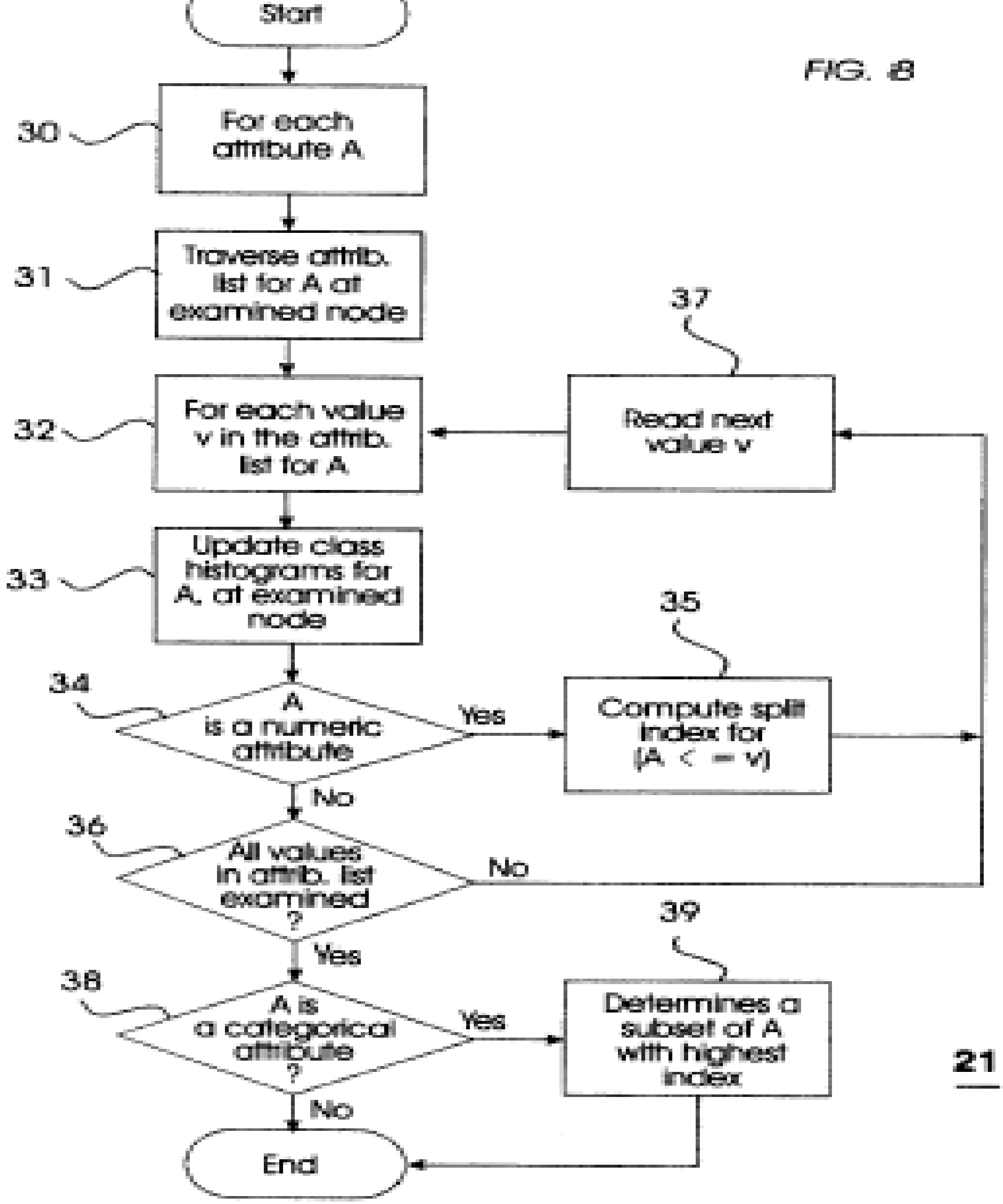
- **Advantage of the gini index:**

  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.

- The following is the splitting test (next slide) –

  The flow chart will fit in the block 21 of decision tree construction


Start
FIG. 8
30
For each attribute A
31
Traverse attrib. list for A at examined node
37
32
For each value v in the attrib. list for A
Read next value v
33
Update class histograms for A, at examined node
35
34
A is a numeric attribute
Yes
Compute split index for (A < = v)
No
36
All values in attrib. list examined ?
No
39
Yes
38
A is a categorical attribute ?
Yes
Determines a subset of A with highest index
No
End
21

Numeric Attributes splitting index

Attribute List

Age	Class	Rec ID
17	High	1
20	High	5
23	High	0
32	Low	4
43	High	2
68	Low	3

Position of cursor in scan

- Position 0
- Position 3
- Position 6

FIG. 9a

State of Class Histograms

Cursor Position 0:

	High	Low
C_{below}	0	0
C_{above}	4	2

Cursor Position 3:

	High	Low
C_{below}	3	0
C_{above}	1	2

Cursor Position 6:

	High	Low
C_{below}	4	2
C_{above}	0	0

FIG. 9b

Splitting for catergorical attributes

Attribute List

Car Type	Class	Rec.ID
family	High	0
sports	High	1
sports	High	2
family	Low	3
truck	Low	4
family	High	5

FIG. 10a

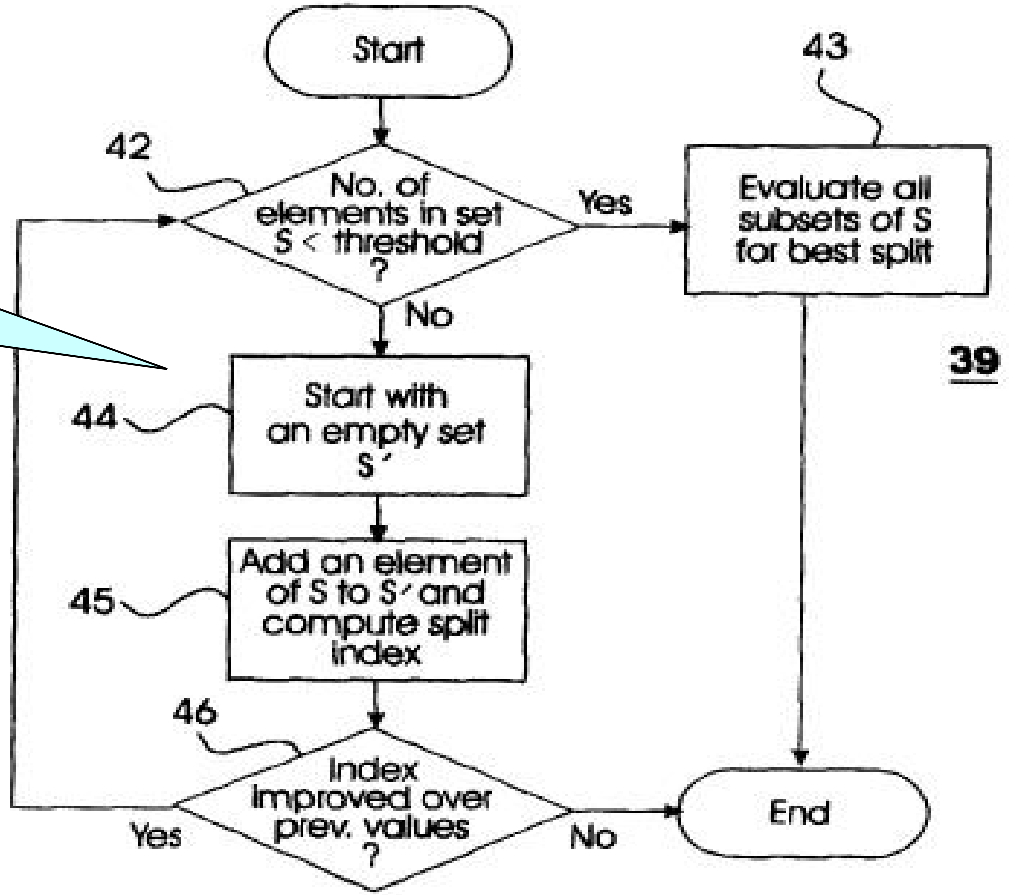
State of Class Histogram

	High	Low
family	2	1
sports	2	0
truck	0	1

FIG. 10b

Determining subset of highest index

Greedy algorithm may be used here



The logic of finding best subset – substitute for block 39

The decision tree getting constructed (level 0)

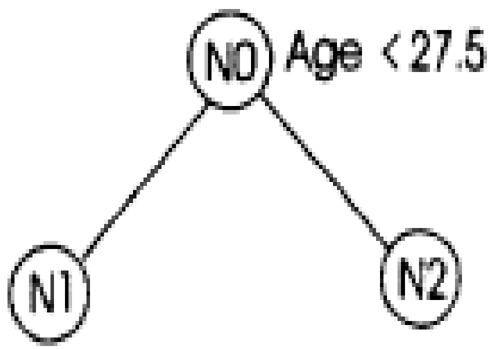


FIG. 13

Attribute Lists for node N0

Age	Class	Rec ID
17	High	1
20	High	5
23	High	0
32	Low	4
43	High	2
68	Low	3

62

Car Type	Class	Rec.ID
family	High	0
sports	High	1
sports	High	2
family	Low	3
truck	Low	4
family	High	5

63

FIG. 14

Decision tree (level 1)

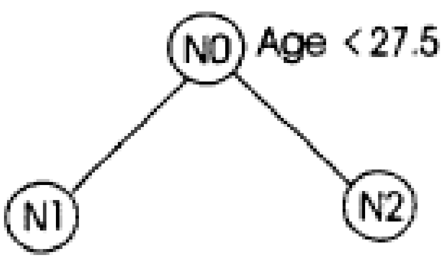


FIG. 13

Attribute Lists for node N1

65

Age	Class	Rec.ID
17	High	1
20	High	5
23	High	0

67

Car Type	Class	Rec.ID
family	High	0
sports	High	1
family	High	5

Attribute Lists for node N2

66

Age	Class	Rec.ID
32	Low	4
43	High	2
68	Low	3

68

Car Type	Class	Rec.ID
sports	High	2
family	Low	3
truck	Low	4

FIG. 15

The classification at level 1

Rec.ID	Node
1	N1
5	N1
0	N1
4	N2
2	N2
3	N2

FIG. 16

Performance:

SLIQ has been tested on these data sets

Dataset	Domain	#Attributes	#Classes	#Examples
Australian	Credit Analysis	14	2	690
Diabetes	Disease diagnosis	8	2	768
DNA	DNA Sequencing	180	3	3186
Letter	Handwriting 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

Performance: Classification Accuracy

Dataset	IND-Cart	IND-C4	SLIQ
Australian	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	84.6
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

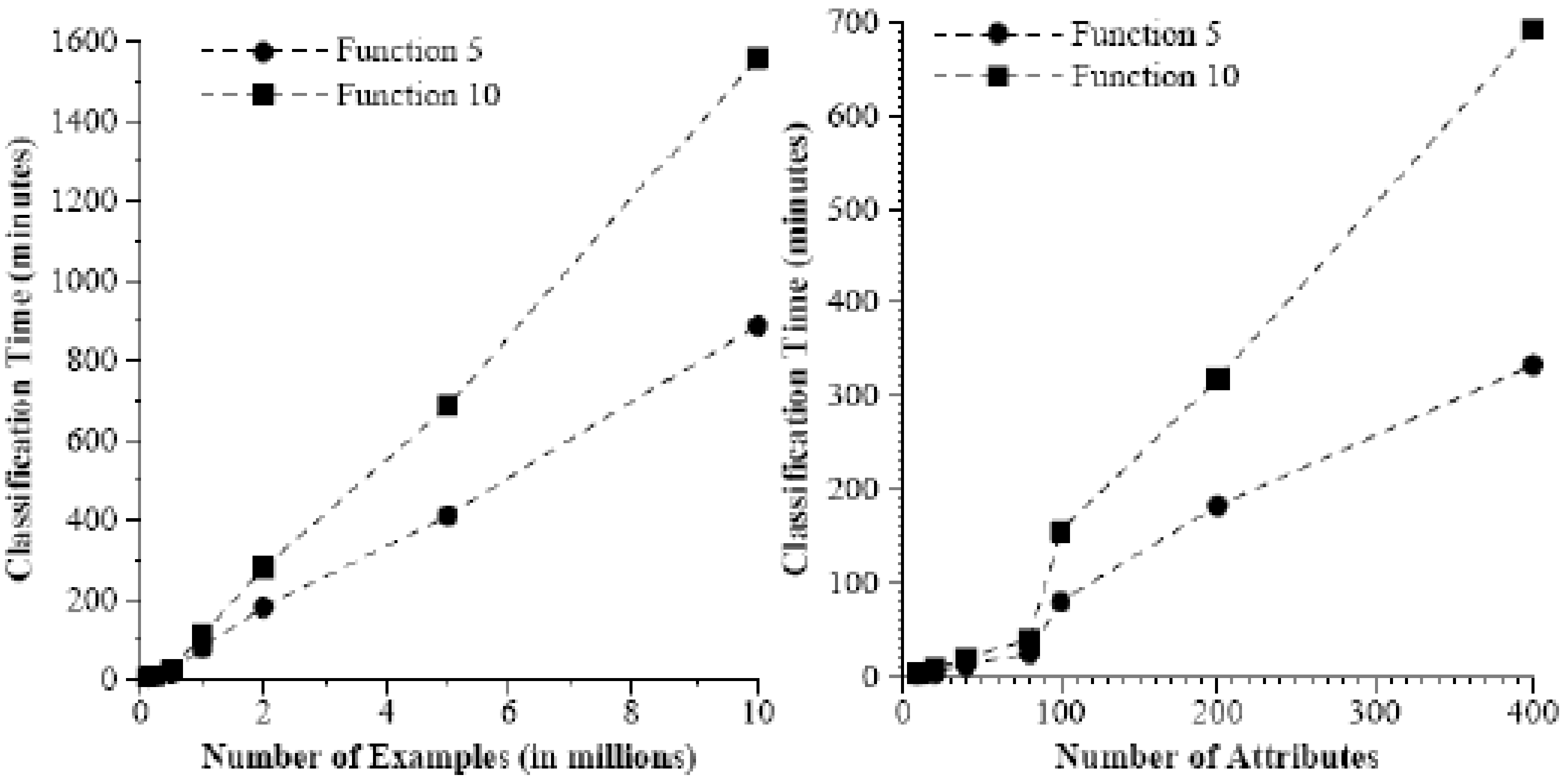
Performance: Decision Tree Size

Dataset	IND-Cart	IND-C4	SLIQ
Australian	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	27
Vehicle	50.1	249.0	49.4

Performance: Execution time

Dataset	IND-Cart	IND-C4	SLIQ
Australian	2.1	1.5	7.1
Diabetes	2.5	1.4	1.8
DNA	33.4	9.21	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

Performance: Scalability



Conclusion:

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

THANK YOU !!!