Classification Methods

1. Classification- Basic Concepts

2. Classification- Two step process

3. Example of Classification

4. Decision tree Induction

5. Attribute selection Measures

Classification –Basic Concepts

Classification:

- Classification is a form of data analysis task where a model or classifier is constructed to predict class labels.
- loan application data : “safe” or “risky” and Marketing data yes” or “no”
- Labels are categorical and represented as discrete values and unordered.
- Analysis provide better understanding of the data at large.
- Classification is a two step process
- **Learning step** - Building of classification model based on training data
- **Classification step**- If the model’s accuracy is acceptable, use the model to classify the test data.

Classification –Basic Concepts

- Typical applications
  - Credit/loan approval
  - Medical diagnosis: if a tumor is cancerous or benign
  - Fraud detection: if a transaction is fraudulent
  - Web page categorization: which category it is
- In each of these examples the data analysis task **is classification** where a **model or classifier** is constructed to **predict class**

Classification—A Two-Step Process

- Data classification is a two step process consisting of **Learning and Classification** step
- **Learning step:** The classification algorithm built a classifier by analyzing or "learning from" a training set.
- Training set made of database tuples and their associated class label.
- Tuple X is represented by n dimensional vector n (x1,x2,x3,x4…xn) depicting n measurements.
- Each X belong to a predefined class and the class label attribute is discrete-valued and unordered.
- The individual tuples making up the training set are referred to as **training tuples**

Classification—A Two-Step Process

- Training tuples are randomly sampled fron the database under analysis.
- Classification process learns mapping or function y=f(X) that predict the class label y of a given tuple X.
- The mapping or function separates the data classes.
- Mapping represented as classification rules, decision trees, or mathematical formula

Supervised vs. Unsupervised Learning

- Supervised learning (classification):
  - The learning of the classifier is "supervised" when the class labels of each training tuple is known.
  - New data is classified based on the training set.
- Unsupervised learning (clustering)
  - The class labels of training data is unknown.
  - Given a set of measurements, observations, etc. with the aim of establishing the existence of classes or clusters in the data

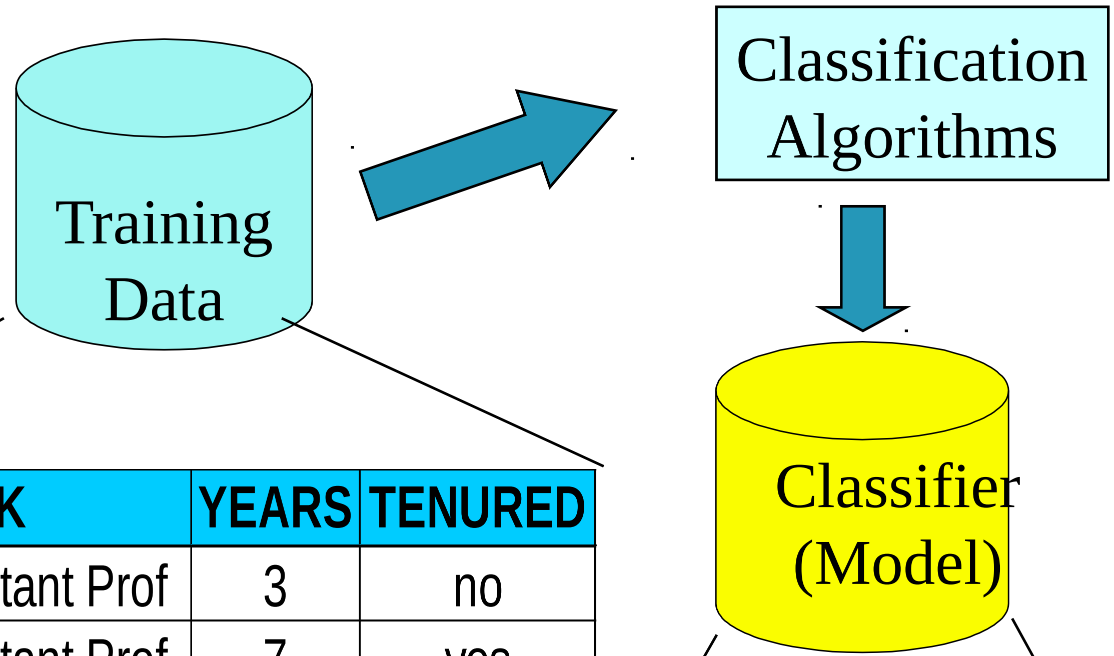
Classification—A Two-Step Process

- **Classification step:** For classifying future or unknown objects use the estimated model (rules).
- The test data are used to esimate the accuracy of classification rules.
- Test set made of test tuples is independent of training set (otherwise overfitting)
  - If the accuracy is acceptable, use the model to classify new data
- The known label of test sample is compared with the classified result from the model.
- **Accuracy**: % of test set samples that are correctly classified by the model
- Note: If the test set is used to select/refine models, it is called **validation (test) set or development test set**

Prediction

- Numeric Prediction
  - Eg: The marketing manager wants to predict how much a given customer will spend.
  - The model constructed (predictor) predicts a continuous-valued function or ordered value.
  - The data analysis task is called as numeric prediction.
  - Regression analysis is a statistical methodology often used for numeric prediction

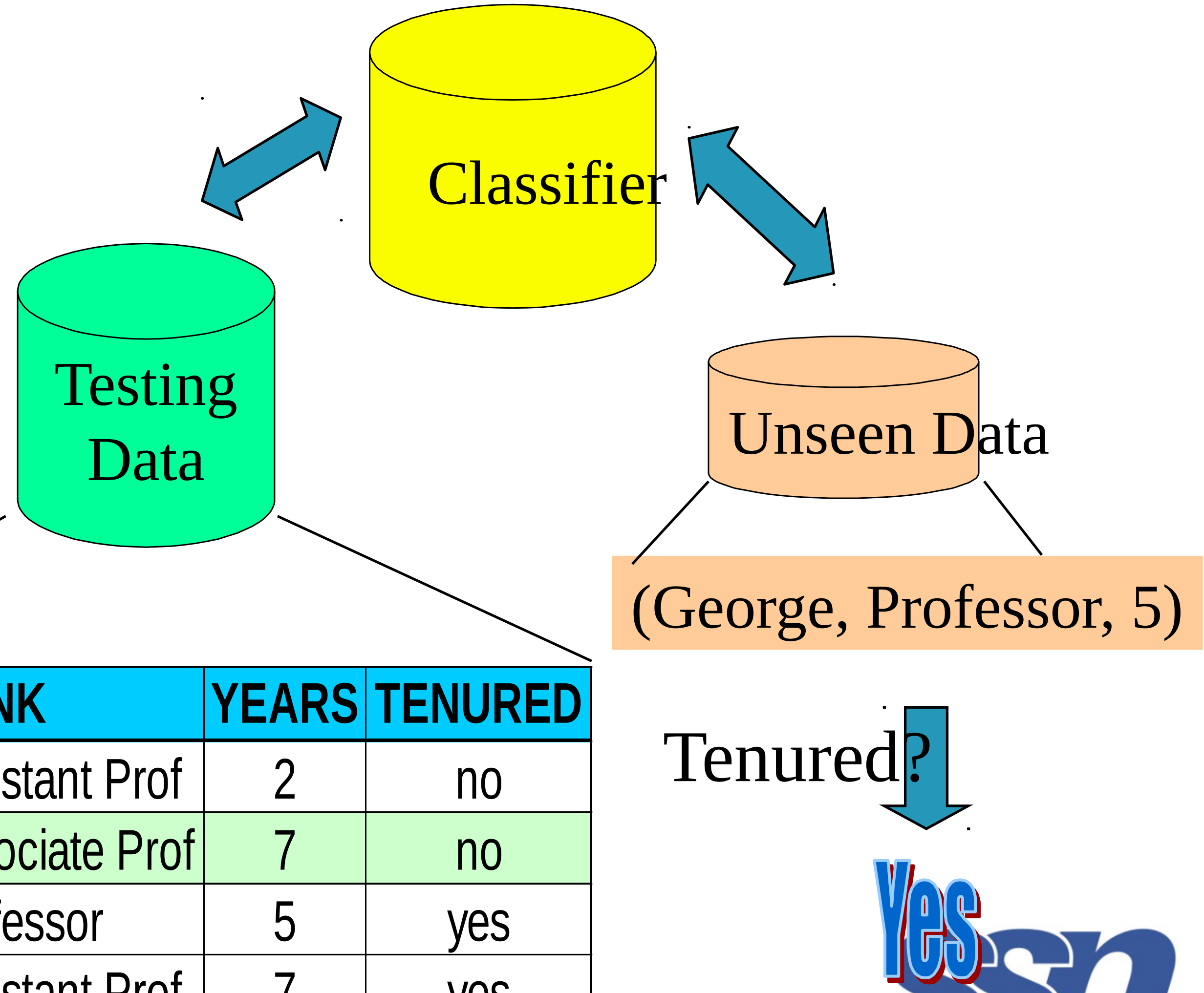
Process (1): Learning Step



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’

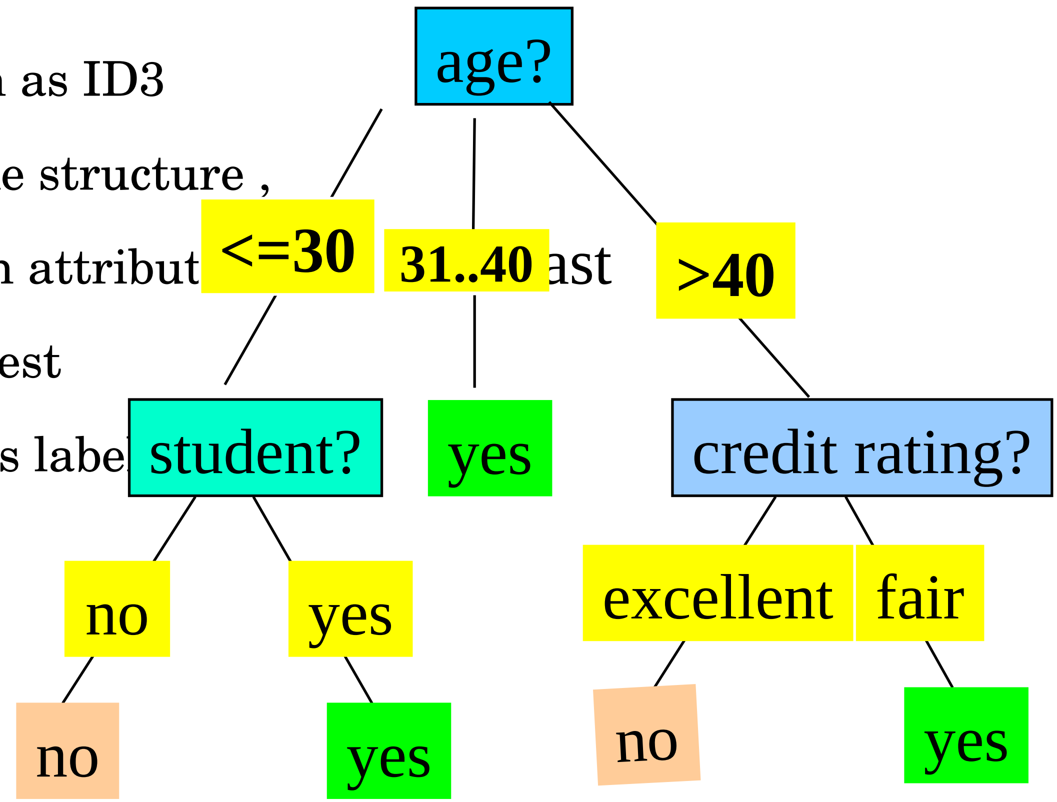
Process (2): Classification Step



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

Decision Tree Induction

- Decision tree induction is the learning of decision trees from class labelled training tuples.
- Decision tree algorithm known as ID3
- Decision tree is a flowchart-like structure ,
  - Internal node=> test on an attribut ast
  - Branch=> outcome of the test
  - Leaf node=> holds the class labe
- Eg: a decision tree for concept buys_computer, internal node reprsents a test on attribute and each leaf node reprsents a class



Algorithm for Decision Tree Induction

- **How are decision trees used for classification?**
  - Given a tuple X for which the associated class label is unknown
  - The attribute values of the tuple are tested against the decision tree
  - A path is traced from the root to the leaf node which holds the class prediction of X
  - Decision trees are easily converted to classification rules.
- **Why decision trees?**
  - Does not require domain knowledge
  - Handles multidimensional data
  - Learning and classification steps are simple and fast
  - Easy to understand and attains good accuracy

Decision Tree Induction

- **Basic algorithm (a greedy algorithm)**
  - Tree is constructed in a top-down recursive divide-and-conquer manner
  - Starts with a training set of tuples and their associated class labels.
  - Training set is recursively partitioned into smaller subsets as the tree is being built.
  - Attribute selection measures are used to select the attribute that best partitions the tuples into distinct classes.
  - The tree nodes are created and the partition is labeled with the splitting criterion, branches are grown out for each outcome of the criteria

Attribute Selection Measures

- Attribute selection measure is a heuristic for selecting the splitting criterion that best separates a given data partition data D of training tuples into individual classes.
- Attribute selection measure are called as **“splitting rules”** because they determine how the tuples at a given node are to be split.
- Three popular attribute selection measures are:
  - Information gain
  - Gain Ratio
  - Gini Index

Attribute Selection Measure: Information Gain (ID3/C4.5)

- Select the attribute with the highest information gain
- Let p_i be the probability that an arbitrary tuple in D belongs to class C_i, estimated by $|C_{i,D}|/|D|$
- Expected information (entropy) needed to classify a tuple in D:

$$Info(D) = -\sum_{i=1}^{m} p_i \log_2(p_i)$$

- Partition the tuples in D on some attribute A having v distinct values {a1,a2,a3..av} observed from the training set.
- Attribute A can be used to split D into v partitions and the partitions corresponds to the branches of the node N
- Information needed to do exact classification :

$$Info_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times Info(D_j)$$

$Gain(A)=Info(D)-Info_A(D)$

- Information gain is defined as the difference between orginal and new requirement

$$Gain(A) = Info(D) - Info_A(D)$$

$Gain(A)=Info(D)-Info_A(D)$

Attribute Selection: Information Gain

- Class P: buys_computer = "yes"
- Class N: buys_computer = "no"

$$Info(D) = I(9,5) = -\frac{9}{14}\log_2\left(\frac{9}{14}\right) - \frac{5}{14}\log_2\left(\frac{5}{14}\right) = 0.940$$

age	p_i	n_i	$I(p_i, n_i)$
<=30	2	3	0.971
31…40	4	0	0
>40	3	2	0.971

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

$$Info_{age}(D) = \frac{5}{14} I(2,3) + \frac{4}{14} I(4,0)$$

$$+ \frac{5}{14} I(3,2) = 0.694$$

$\frac{5}{14} I(2,3)$ means "age <=30" has 5 out of 14 samples, with 2 yes'es and 3 no's. Hence

$$Gain(age) = Info(D) - Info_{age}(D) = 0.246$$

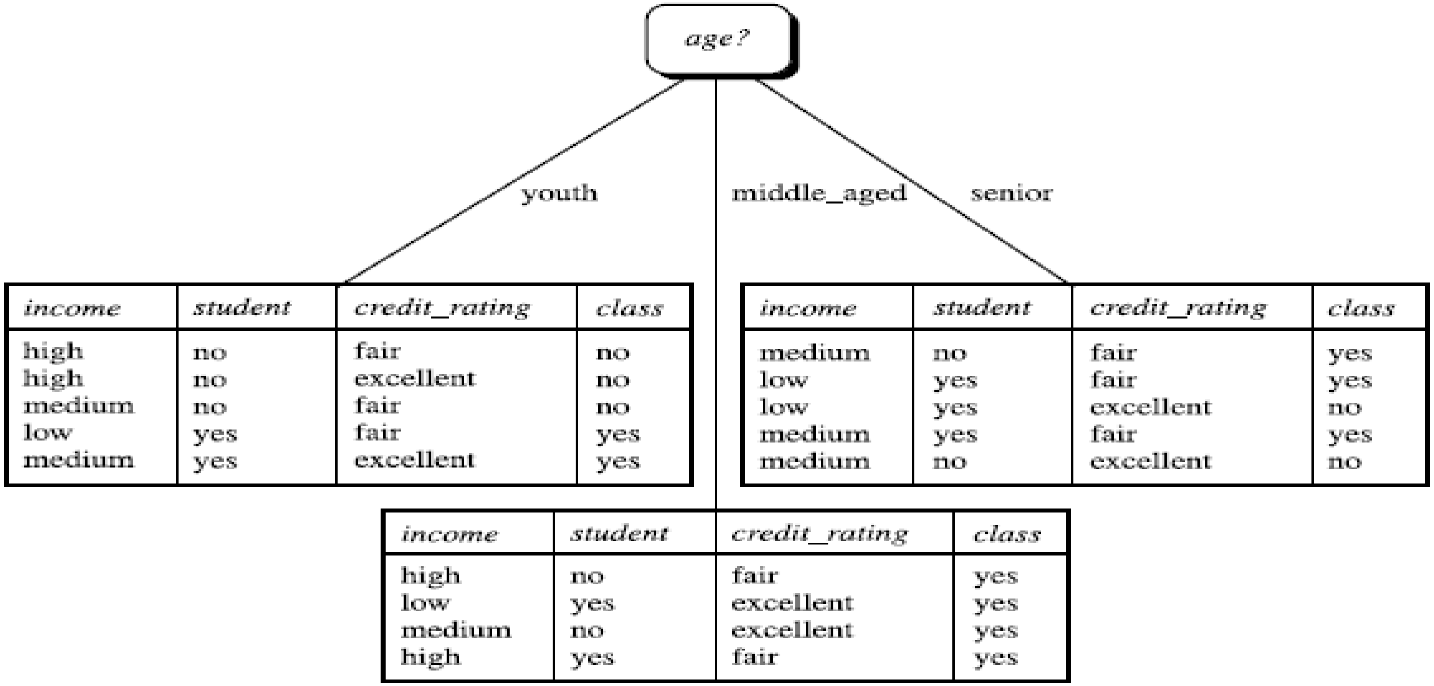
Similarly:

$$Gain(income) = 0.029$$

$$Gain(student) = 0.151$$

$$Gain(credit_{rating}) = 0.048$$

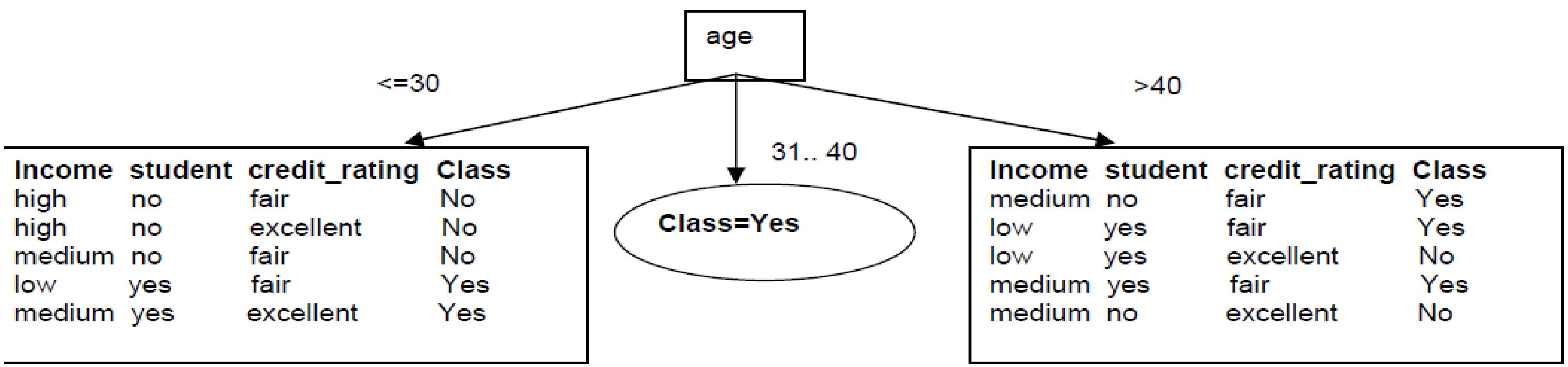
Attribute selection-Information gain



income	*student*	*credit_rating*	*class*
high	no	fair	no
high	no	excellent	no
medium	no	fair	no
low	yes	fair	yes
medium	yes	excellent	yes

income	*student*	*credit_rating*	*class*
high	no	fair	yes
low	yes	excellent	yes
medium	no	excellent	yes
high	yes	fair	yes

income	*student*	*credit_rating*	*class*
medium	no	fair	yes
low	yes	fair	yes
low	yes	excellent	no
medium	yes	fair	yes
medium	no	excellent	no



Income	**student**	**credit_rating**	**Class**
high	no	fair	No
high	no	excellent	No
medium	no	fair	No
low	yes	fair	Yes
medium	yes	excellent	Yes

Income	**student**	**credit_rating**	**Class**
medium	no	fair	Yes
low	yes	fair	Yes
low	yes	excellent	No
medium	yes	fair	Yes
medium	no	excellent	No

Attribute Selection: Information Gain

The mutual information is $I(S_{Yes}, S_{No}) = I(2,3) = -2/5 \log_2(2/5) - 3/5 \log_2(3/5) = 0.97$

- For Income we have three values $income_{high}$ (0 yes and 2 no), $income_{medium}$ (1 yes and 1 no) and $income_{low}$ (1 yes and 0 no)

$$\begin{aligned} Entropy(income) &= 2/5(0) + 2/5\,(-1/2\log(1/2) - 1/2\log(1/2)) + 1/5\,(0) \\ &= 2/5\,(1) = 0.4 \end{aligned}$$

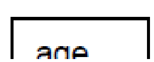
$Gain(income) = 0.97 - 0.4 = 0.57$

- For Student we have two values $student_{yes}$ (2 yes and 0 no) and $student_{no}$ (0 yes 3 no)

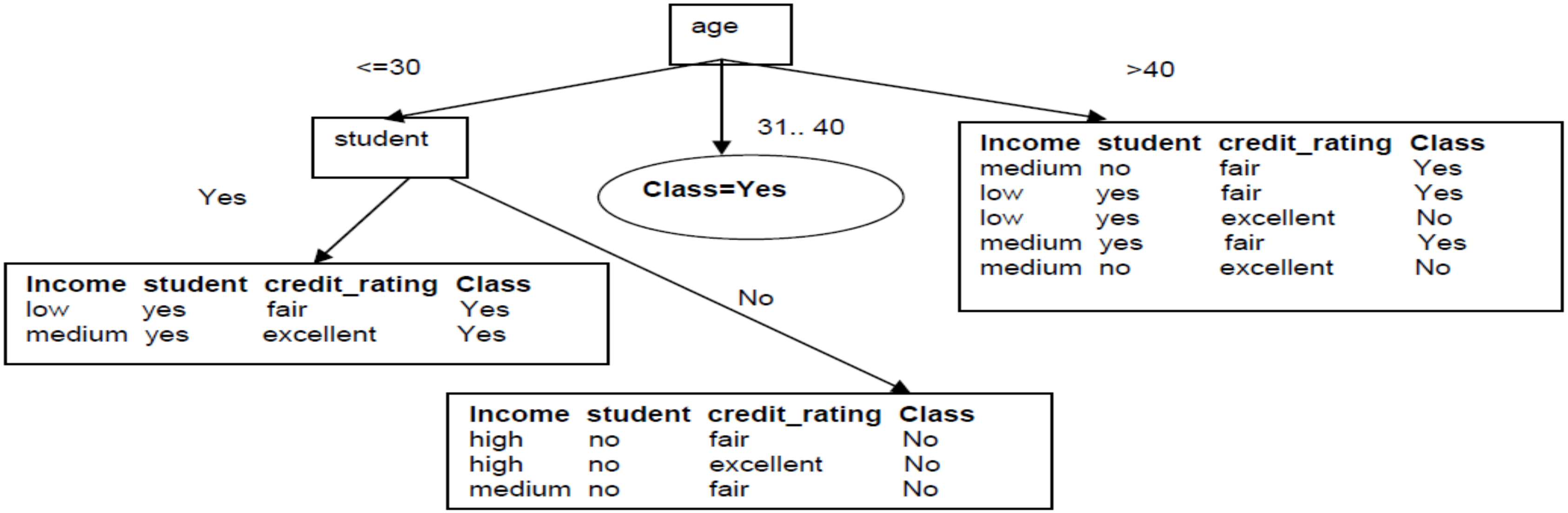
$Entropy(student) = 2/5(0) + 3/5(0) = 0$

$Gain\ (student) = 0.97 - 0 = 0.97$

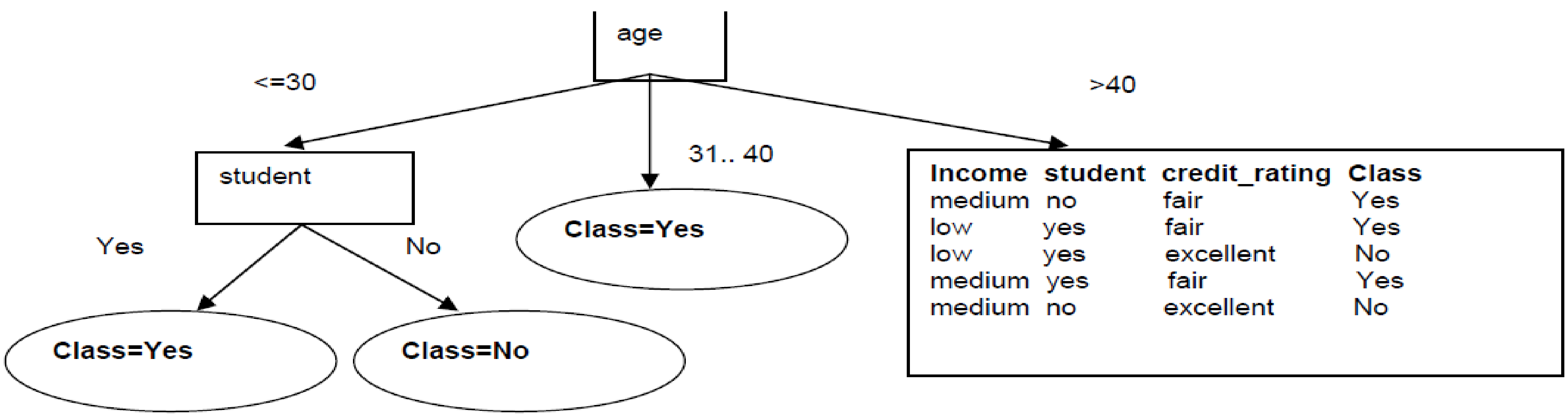
We can then safely split on attribute student without checking the other attributes since the information gain is maximized.



Attribute Selection: Information Gain



Since these two new branches are from distinct classes, we make them into leaf nodes with their respective class as label:



Attribute Selection: Information Gain

Again the same process is needed for the other branch of age.

The mutual information is $I(S_{Yes}, S_{No}) = I(3,2) = -3/5 \log_2(3/5) - 2/5 \log_2(2/5) = 0.97$

- For Income we have two values $income_{medium}$ (2 yes and 1 no) and $income_{low}$ (1 yes and 1 no)

$$\text{Entropy(income)} = 3/5(-2/3\log(2/3)-1/3\log(1/3)) + 2/5\ (-1/2\log(1/2)-1/2\log(1/2))$$
$$= 3/5(0.9182)+2/5\ (1) = 0.55+0.\ 4= 0.95$$

Gain(income) = 0.97 – 0.95 = 0.02

- For Student we have two values $student_{yes}$ (2 yes and 1 no) and $student_{no}$ (1 yes and 1 no)

$$\text{Entropy(student)} = 3/5(-2/3\log(2/3)-1/3\log(1/3)) + 2/5(-1/2\log(1/2)-1/2\log(1/2)) = 0.95$$

Gain (student) = 0.97 – 0.95 = 0.02

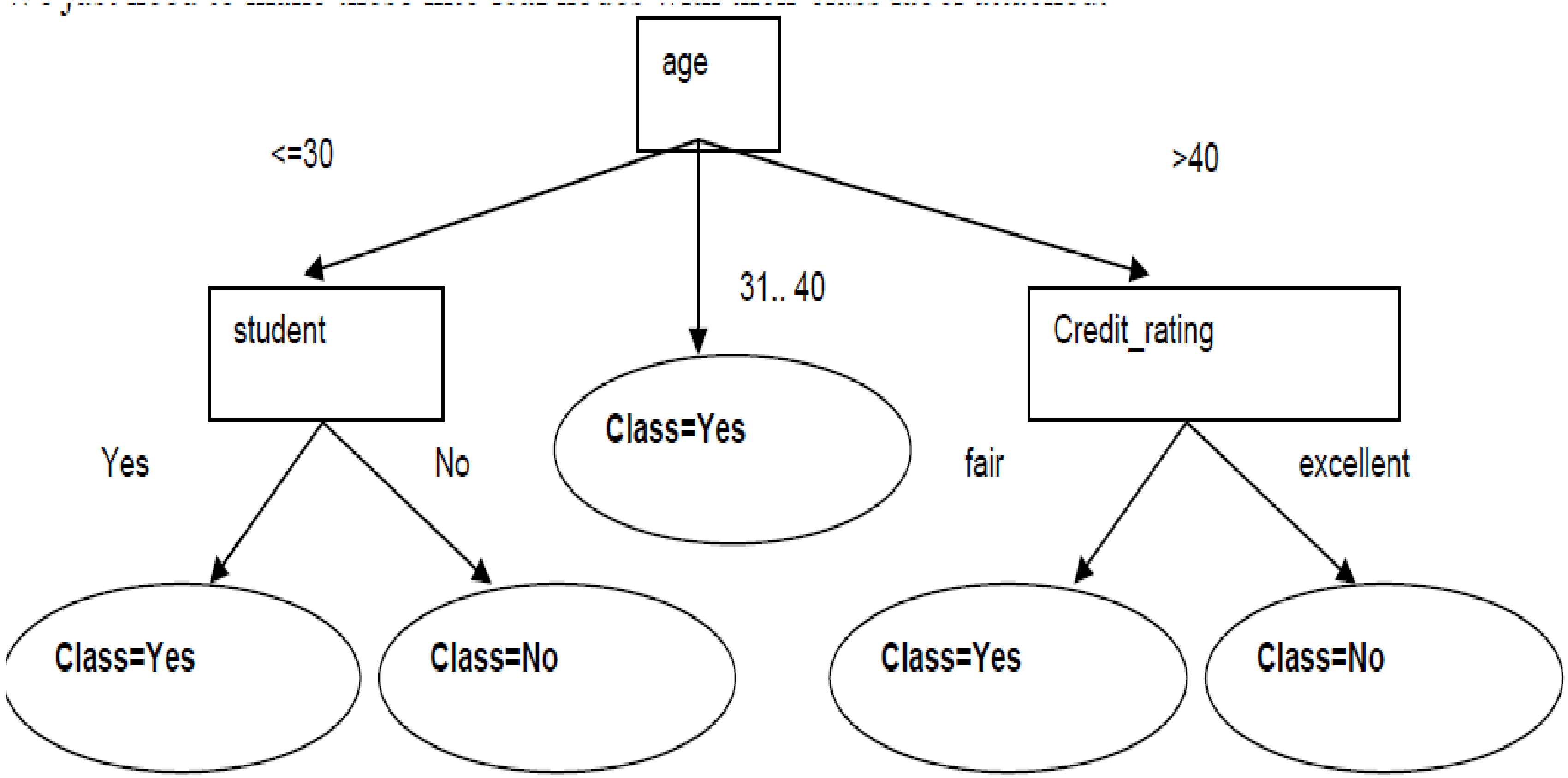
- For Credit_Rating we have two values $credit\_rating_{fair}$ (3 yes and 0 no) and $credit\_rating_{excellent}$ (0 yes and 2 no)

Entropy(credit_rating) = 0

Gain(credit_rating) = 0.97 – 0 = 0.97

We then split based on credit_rating. These splits give partitions each with records from the same class. We just need to make these into leaf nodes with their class label attached:

Output



New example: age<=30, income=medium, student=yes, credit-rating=fair

Follow branch(age<=30) then student=yes we predict Class=yes ➔ Buys_computer = yes

Computing Information-Gain for Continuous-Valued Attributes

- Let attribute A be a continuous-valued attribute
- Must determine the *best split point* for A
  - Sort the value A in increasing order
  - Typically, the midpoint between each pair of adjacent values is considered as a possible *split point*
    - $(a_i+a_{i+1})/2$ is the midpoint between the values of a_i and a_{i+1}
  - The point with the *minimum expected information requirement* for A is selected as the split-point for A
- Split:
  - D1 is the set of tuples in D satisfying A ≤ split-point, and D2 is the set of tuples in D satisfying A > split-point

Algorithm for Decision Tree Induction

- Algorithm is called with three parameters D, attribute_list and attribute_selection_method
- D=>complete set of training tuples and associated class labels
- attribute_list => list of attribute describing the tuples.
- attribute_selection_method:
  - heuristics procedure to determine the splitting criterion.
  - Determines which attribute to test at node N by determining the “best” way to separate the tuples D into classes
  - Determines which branch to grow from node N w.r to outcome of the chosen test.
- Attribute selection measure: Information gain and Gini index

Generation of Decision Tree

Algorithm: Generate_decision_tree. Generate a decision tree from the training tuples of data partition, D.

Input:

- Data partition, D, which is a set of training tuples and their associated class labels;
- *attribute_list*, the set of candidate attributes;
- *Attribute_selection_method*, a procedure to determine the splitting criterion that "best" partitions the data tuples into individual classes. This criterion consists of a *splitting_attribute* and, possibly, either a *split-point* or *splitting subset*.

Output: A decision tree.

Method:

```
(1)  create a node N;
(2)  if tuples in D are all of the same class, C, then
(3)      return N as a leaf node labeled with the 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)  label node N with splitting_criterion;
(8)  if splitting_attribute is discrete-valued and
         multiway splits allowed then // not restricted to binary trees
(9)      attribute_list ← attribute_list − splitting_attribute; // remove splitting_attribute
(10) for each outcome j of splitting_criterion
     // partition the tuples and grow subtrees for each partition
(11)     let D_j be the set of 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;
     endfor
(15) return N;
```

Algorithm for Decision Tree Induction

- Conditions for stopping partitioning
  - All samples for a given node belong to the same class
  - There are no remaining attributes for further partitioning —**majority voting** is employed for classifying the leaf
  - There are no samples left

Gain Ratio for Attribute Selection (C4.5)

- Information gain measure is biased towards attributes with a large number of values
- C4.5 (a successor of ID3) uses gain ratio to overcome the problem (normalization to information gain)

$$SplitInfo_A(D) = -\sum_{j=1}^{v} \frac{|D_j|}{|D|} \times \log_2\left(\frac{|D_j|}{|D|}\right)$$

  – GainRatio(A) = Gain(A)/SplitInfo(A)

$$SplitInfo_{income}(D) = -\frac{4}{14} \times \log_2\left(\frac{4}{14}\right) - \frac{6}{14} \times \log_2\left(\frac{6}{14}\right) - \frac{4}{14} \times \log_2\left(\frac{4}{14}\right) = 1.557$$

  – gain_ratio(income) = 0.029/1.557 = 0.019

- The attribute with the maximum gain ratio is selected as the splitting attribute

Gini Index (CART, IBM IntelligentMiner)

- If a data set *D* contains examples from *n* classes, gini index, *gini*(*D*) is defined as

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

where p_j is the probability that a tuple in D belongs to class Ci

- If a data set *D* is split on A into two subsets D_1 and D_2, the *gini* index *gini*(*D*) is defined as

$$gini_A(D) = \frac{|D_1|}{|D|} gini(D_1) + \frac{|D_2|}{|D|} gini(D_2)$$

- The subset with minimum gini index is selected as splitting subset
- Reduction in Impurity:

$$\Delta gini(A) = gini(D) - gini_A(D)$$

Computation of Gini Index

- Ex. D has 9 tuples in buys_computer = "yes" and 5 in "no"

$$gini(D) = 1 - \left(\frac{9}{14}\right)^2 - \left(\frac{5}{14}\right)^2 = 0.459$$

- Suppose the attribute income partitions D into 10 in D_1: {low, medium} and 4 in D_2

$$gini_{income \in \{low, medium\}}(D) = \left(\frac{10}{14}\right) Gini(D_1) + \left(\frac{4}{14}\right) Gini(D_2)$$

$$= \frac{10}{14}\left(1 - \left(\frac{7}{10}\right)^2 - \left(\frac{3}{10}\right)^2\right) + \frac{4}{14}\left(1 - \left(\frac{2}{4}\right)^2 - \left(\frac{2}{4}\right)^2\right)$$
$$= 0.443$$
$$= Gini_{income \in \{high\}}(D).$$

$Gini_{\{low,high\}}$ is 0.458; $Gini_{\{medium,high\}}$ is 0.450. Thus, split on the {low,medium} (and {high}) since it has the lowest Gini index

Comparing Attribute Selection Measures

- The three measures, in general, return good results but
  - **Information gain**:
    - biased towards multivalued attributes
  - **Gain ratio**:
    - tends to prefer unbalanced splits in which one partition is much smaller than the others
  - **Gini index**:
    - biased to multivalued attributes
    - has difficulty when # of classes is large
    - tends to favor tests that result in equal-sized partitions and purity in both partitions

Overfitting and Tree Pruning

- Overfitting: An induced tree may overfit the training data
  - Too many branches, some may reflect anomalies due to noise or outliers
  - Poor accuracy for unseen samples
- Two approaches to avoid overfitting
  - Prepruning: *Halt tree construction early*- do not split a node if this would result in the goodness measure falling below a threshold (Difficult to choose best threshold)
  - Postpruning: *Remove branches* from a "fully grown" tree—get a sequence of progressively pruned trees
    - Use a set of data different from the training data to decide which is the "best pruned tree"

Classification in Large Databases

- Classification—a classical problem extensively studied by statisticians and machine learning researchers
- Scalability: Classifying data sets with millions of examples and hundreds of attributes with reasonable speed
- Why is decision tree induction popular?
  - relatively faster learning speed (than other classification methods)
  - convertible to simple and easy to understand classification rules
  - can use SQL queries for accessing databases
  - comparable classification accuracy with other methods