

Classification and Prediction (I)

Liaquat Majeed Sheikh

liaquat.majeed@nu.edu.pk

**National University of Computer
and Emerging Sciences**

INTRODUCTION

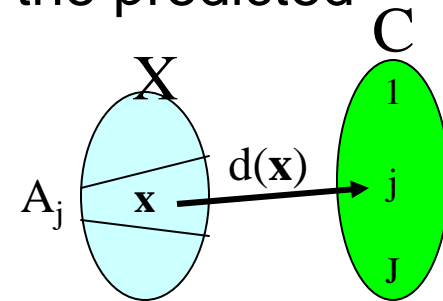
- Given a set of pre-classified examples, build a model or *classifier* to classify new cases.
- *Supervised* learning in that classes are known for the examples used to build the classifier.
- A classifier can be a set of rules, a decision tree, a neural network, etc.
- Typical Applications: credit approval, target marketing, fraud detection, medical diagnosis, treatment effectiveness analysis,

Constructing a Classifier

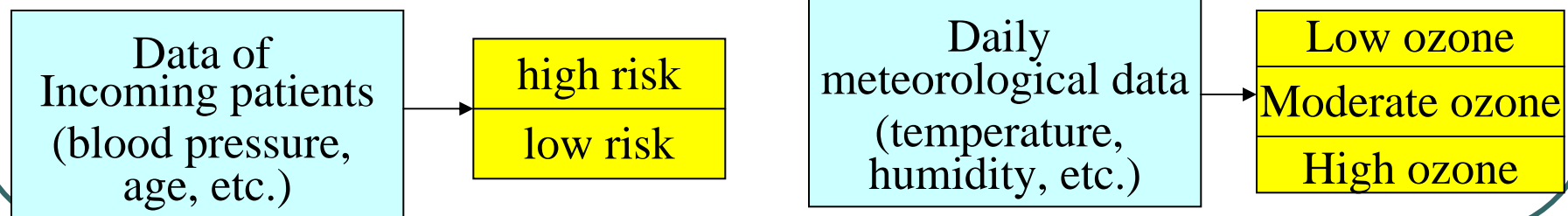
- The goal is to maximize the accuracy on new cases that have similar class distribution.
- Since new cases are not available at the time of construction, the given examples are divided into the *testing set* and the *training set*. The classifier is built using the training set and is evaluated using the testing set.
- The goal is to be accurate on the testing set. It is essential to capture the “structure” shared by both sets.
- Must prune overfitting rules that work well on the training set, but poorly on the testing set.

Definitions of Classification

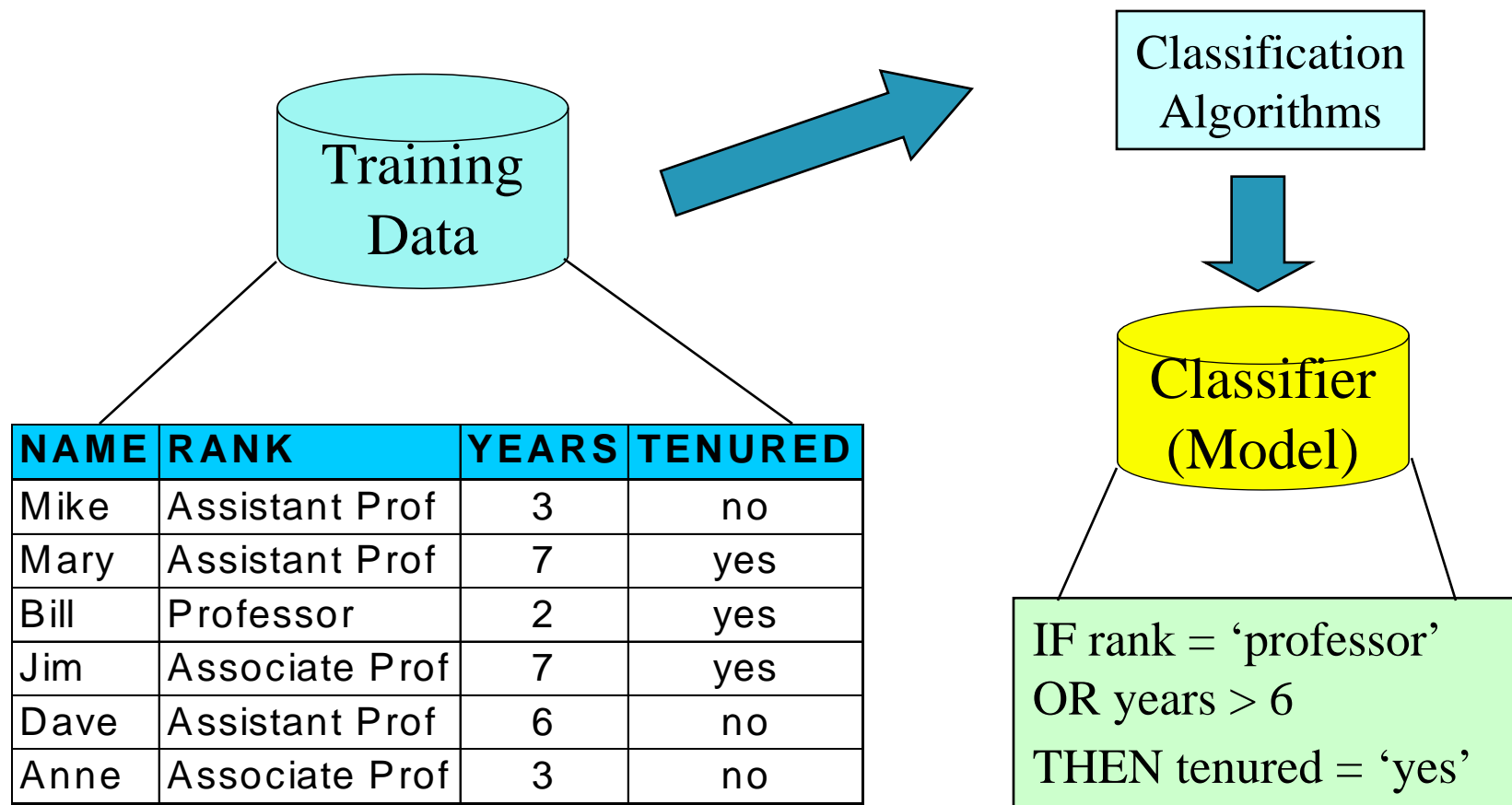
- A classifier or classification rule is a function $d(\mathbf{x})$ on measurement space X so that for every measurement vector \mathbf{x} , $d(\mathbf{x})$ is equal to one of the numbers $1, 2, \dots, J$.
- A classifier is a partition of X into J disjoint subsets A_1, \dots, A_J , $X = \cup_j A_j$ such that for every $\mathbf{x} \in A_j$ the predicted class is j .



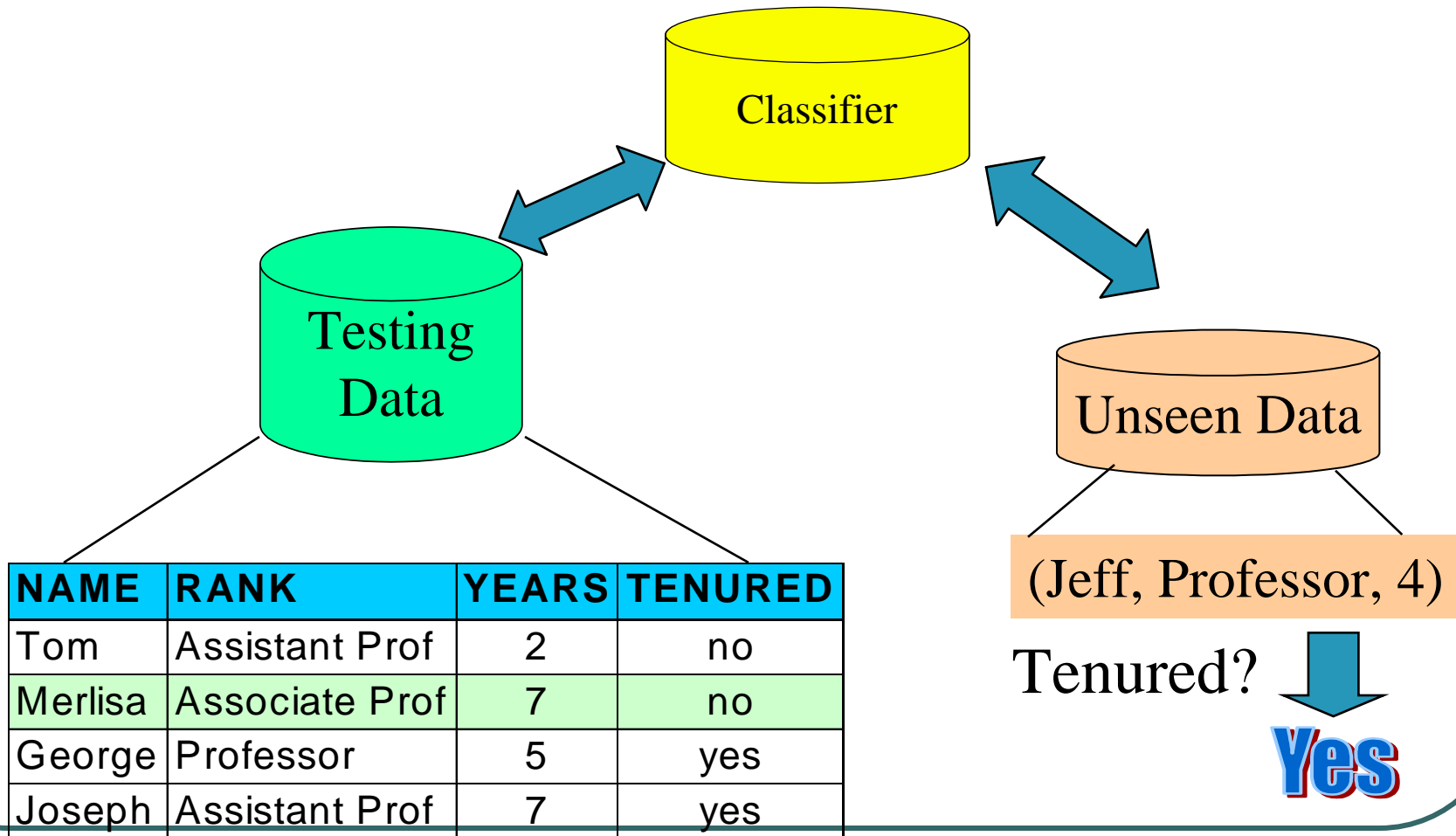
Examples



Classification Process (1): Model Construction



Classification Process (2): Use the Model in Prediction



Supervised vs. Unsupervised Learning

- **Supervised learning (classification)**
 - Supervision: The training data (observations, measurements, etc.) are accompanied by labels indicating the class of the observations
 - 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

Issues (1): Data Preparation

- Data cleaning
 - Preprocess data in order to reduce noise and handle missing values
- Relevance analysis (feature selection)
 - Remove the irrelevant or redundant attributes
- Data transformation
 - Generalize and/or normalize data

Issues (2): Evaluating Classification Methods

- Predictive accuracy
- Speed and scalability
 - time to construct the model
 - time to use the model
- Robustness
 - handling noise and missing values
- Scalability
 - efficiency in disk-resident databases
- Interpretability:
 - understanding and insight provided by the model
- Goodness of rules
 - decision tree size
 - compactness of classification rules

Classification by Decision Tree Induction

- Decision tree
 - A flow-chart-like tree structure
 - Internal node denotes a test on an attribute
 - Branch represents an outcome of the test
 - Leaf nodes represent class labels or class distribution
- Decision tree generation consists of two phases
 - Tree construction
 - At start, all the training examples are at the root
 - Partition examples recursively based on selected attributes
 - Tree pruning
 - Identify and remove branches that reflect noise or outliers
- Use of decision tree: Classifying an unknown sample
 - Test the attribute values of the sample against the decision tree

Example 1

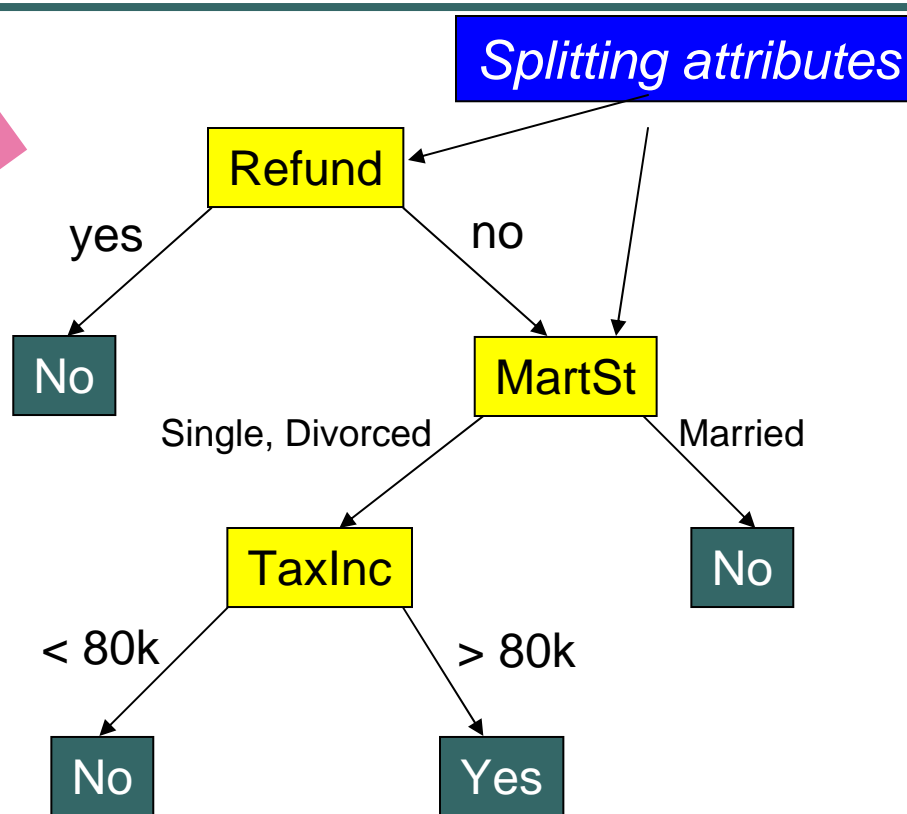
Categorical

Categorical

Continuous

Class

Tid	Refund	Marital Status	Taxable Income	Cheat
1	yes	single	125k	no
2	no	married	100k	no
3	no	single	70k	no
4	yes	Married	120k	no
5	no	Divorced	95k	yes
6	no	Married	60k	no
7	yes	Divorced	220k	no
8	no	single	85k	yes
9	no	married	75k	no
10	no	single	90k	yes



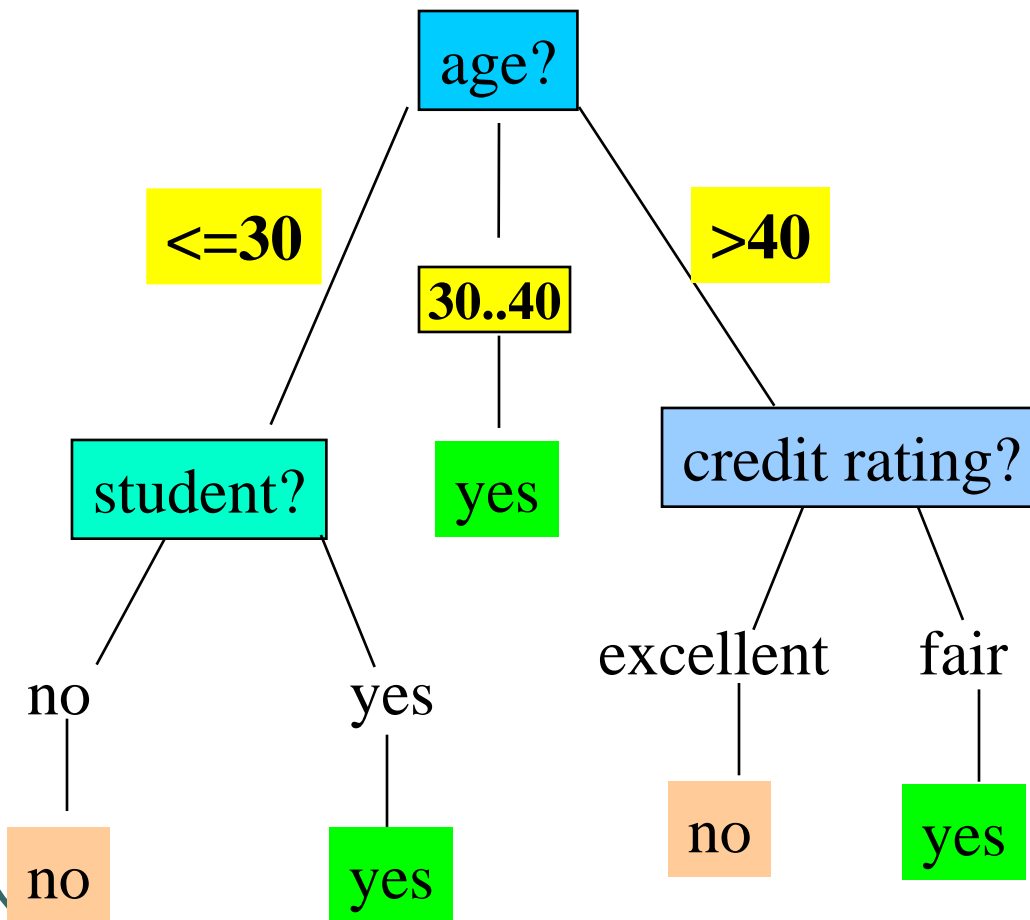
The splitting attribute at a node is determined based on Gini Index.

Example 2: Training Dataset

This follows
an example
from
Quinlan's ID3

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

Output: A Decision Tree for “*buys_computer*”

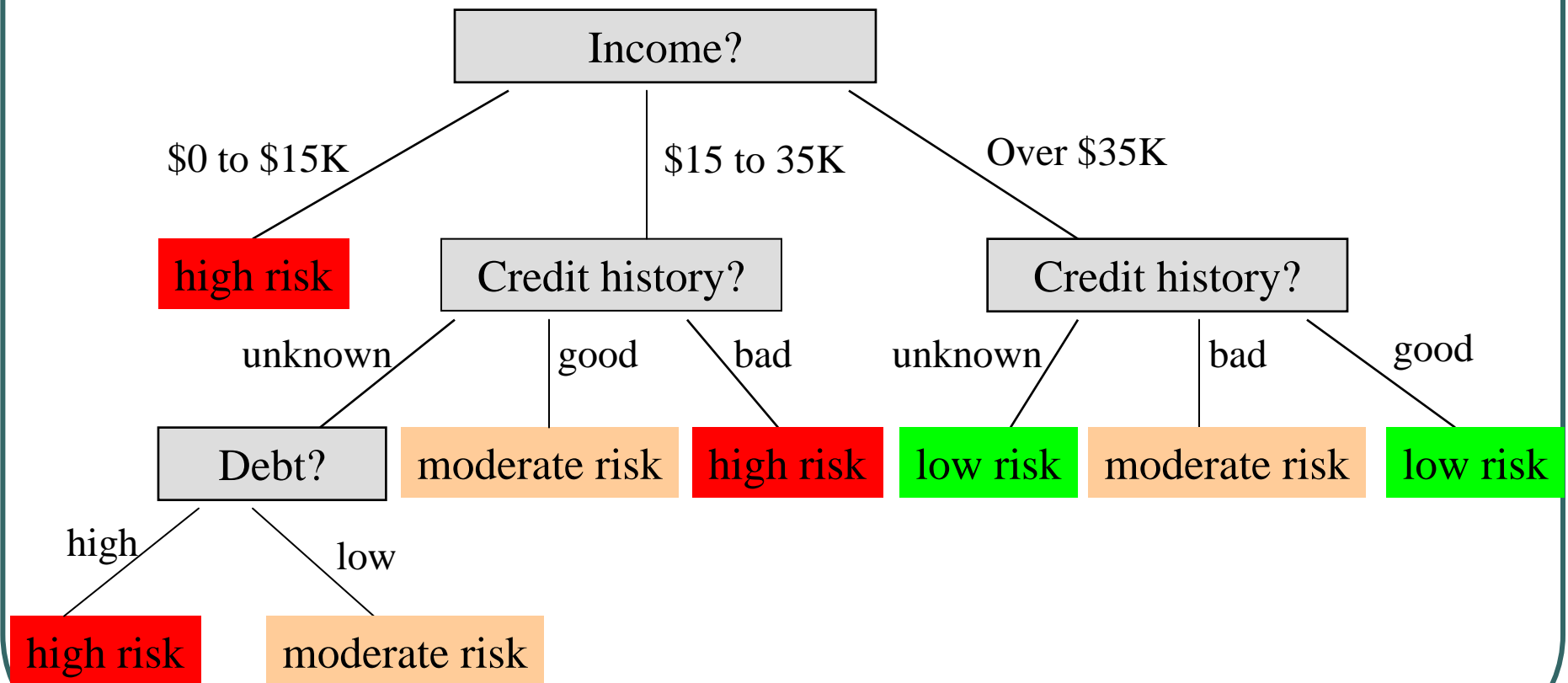


age	income	student	credit_rating	buys_computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
31...40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31...40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes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

Example 3: Training Data Set

No.	Risk	Credit History	Debt	Collateral	Income
1	high	bad	high	none	\$0 to \$15k
2	high	unknown	high	none	\$15 to \$35k
3	moderate	unknown	low	none	\$15 to \$35k
4	high	unknown	low	none	\$0 to \$15k
5	low	unknown	low	none	over \$35k
6	low	unknown	low	adequate	over \$35k
7	high	bad	low	none	\$0 to \$15k
8	moderate	bad	low	adequate	over \$35k
9	low	good	low	none	over \$35k
10	low	good	high	adequate	over \$35k
11	high	good	high	none	\$0 to \$15k
12	moderate	good	high	none	\$15 to \$35k
13	low	good	high	none	over \$35k
14	high	bad	high	none	\$15 to \$35k

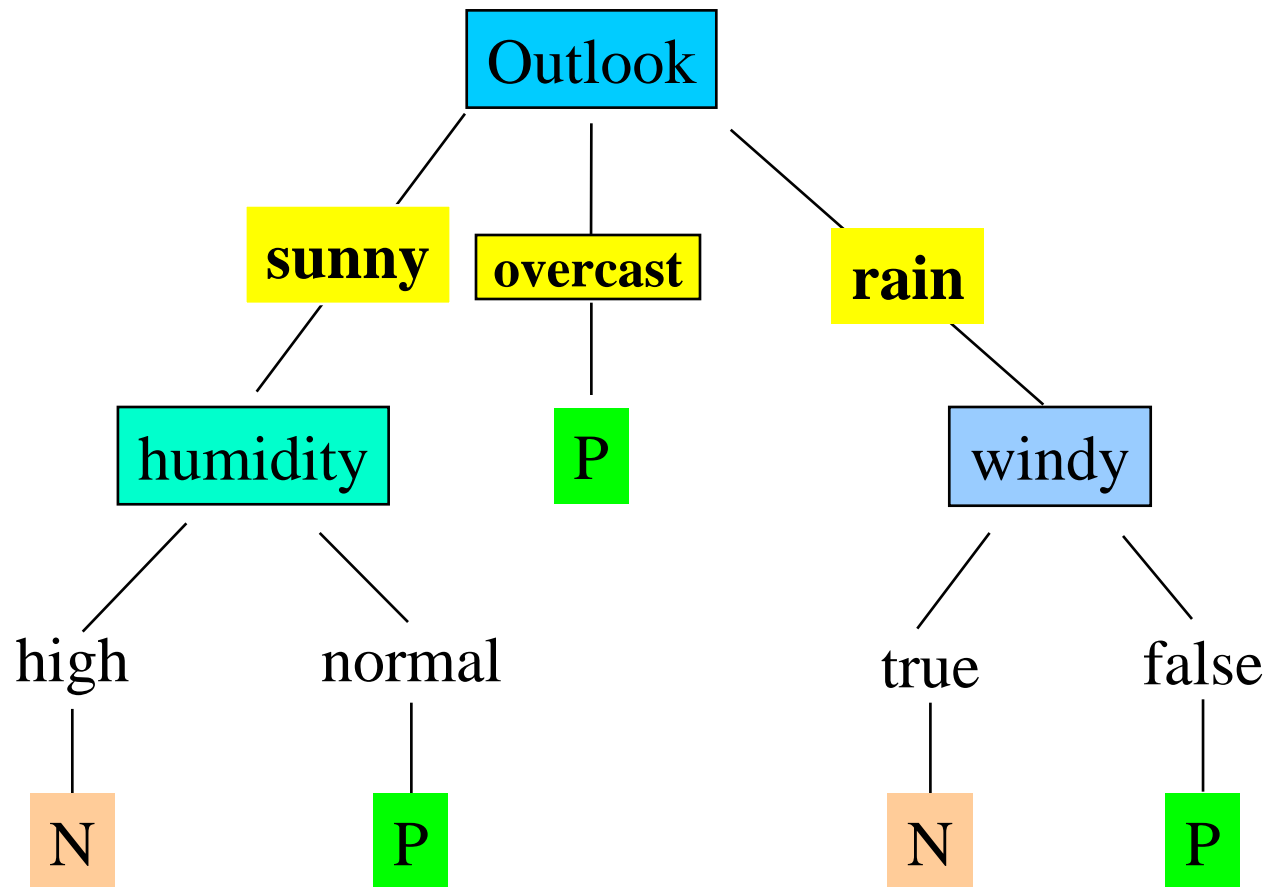
Output: A Decision Tree for “*credit_risk*”



Example 4: Training Set

Outlook	Temperature	Humidity	Windy	Class
sunny	hot	high	false	N
sunny	hot	high	true	N
overcast	hot	high	false	P
rain	mild	high	false	P
rain	cool	normal	false	P
rain	cool	normal	true	N
overcast	cool	normal	true	P
sunny	mild	high	false	N
sunny	cool	normal	false	P
rain	mild	normal	false	P
sunny	mild	normal	true	P
overcast	mild	high	true	P
overcast	hot	normal	false	P
rain	mild	high	true	N

Output: A Decision Tree for *“play”*



Algorithm for Decision Tree Induction

- Basic algorithm (a greedy algorithm)
 - Tree is constructed in a **top-down recursive divide-and-conquer manner**
 - At start, all the training examples are at the root
 - Attributes are categorical (if continuous-valued, they are discretized in advance)
 - Examples are partitioned recursively based on selected attributes
 - Test attributes are selected on the basis of a heuristic or statistical measure (e.g., **information gain**)
- 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

Attribute Selection Measure

- **Information gain (ID3/C4.5)**
 - All attributes are assumed to be categorical
 - Can be modified for continuous-valued attributes
- **Gini index (IBM IntelligentMiner)**
 - All attributes are assumed continuous-valued
 - Assume there exist several possible split values for each attribute
 - May need other tools, such as clustering, to get the possible split values
 - Can be modified for categorical attributes

Information Gain (ID3/C4.5)

- Select the attribute with the highest information gain
- Assume there are two classes, P and N
 - Let the set of examples S contain p elements of class P and n elements of class N
 - The amount of information, needed to decide if an arbitrary example in S belongs to P or N is defined as

$$I(p, n) = -\frac{p}{p+n} \log_2 \frac{p}{p+n} - \frac{n}{p+n} \log_2 \frac{n}{p+n}$$

Information Gain in Decision Tree Induction

- Assume that using attribute A a set S will be partitioned into sets $\{S_1, S_2, \dots, S_v\}$
 - If S_i contains p_i examples of P and n_i examples of N , the **entropy**, or the expected information needed to classify objects in all subtrees S_i is

$$E(A) = \sum_{i=1}^v \frac{p_i + n_i}{p + n} I(p_i, n_i)$$

- The encoding information that would be gained by branching on A

$$Gain(A) = I(p, n) - E(A)$$

Attribute Selection by Information Gain

- Class P: buys_computer = “yes”
- Class N: buys_computer = “no”
- $I(p, n) = I(9, 5) = 0.940$
- Compute the entropy for *age*:

$$I(2,3) = -\frac{2}{5} \log_2\left(\frac{2}{5}\right) - \frac{3}{5} \log_2\left(\frac{3}{5}\right) = .970$$

age	p_i	n_i	$I(p_i, n_i)$
≤ 30	2	3	0.971
30...40	4	0	0
> 40	3	2	0.971

$$\log_b X = \frac{\log_c(X)}{\log_c(b)}$$

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

$$E(\text{age}) = \frac{5}{14} I(2,3) + \frac{4}{14} I(4,0) + \frac{5}{14} I(3,2) = 0.69$$

Hence

$$\text{Gain}(\text{age}) = I(p, n) - E(\text{age})$$

$$\text{Gain}(\text{age}) = 0.940 - 0.69$$

Attribute Selection by Information Gain (Cont.)

- Class P: buys_computer = “yes”
- Class N: buys_computer = “no”
- $I(p, n) = I(9, 5) = 0.940$
- Compute the entropy for *income*:

income	p_i	n_i	$I(p_i, n_i)$
high	2	2	0.998
medium	4	2	0.918
low	3	1	0.8111

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

$$E(\text{income}) = \frac{4}{14} I(2,2) + \frac{5}{14} I(4,2) + \frac{5}{14} I(3,1) = 0.911$$

Hence

$$\text{Gain}(\text{income}) = I(p, n) - E(\text{income})$$

$$\text{Gain}(\text{income}) = 0.940 - 0.911$$

$$\text{Gain}(\text{income}) = 0.029$$

Attribute Selection by Information Gain (cont.)

- Class P: buys_computer = “yes”
- Class N: buys_computer = “no”
- $I(p, n) = I(9, 5) = 0.940$
- Compute the entropy for *student*.

student	p_i	n_i	$I(p_i, n_i)$
yes	6	1	0.5916
no	3	4	0.9815

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

$$E(\text{student}) = \frac{7}{14} I(6,1) + \frac{7}{14} I(3,4)$$

$$E(\text{student}) = .2958 + .4925 = .7883$$

Hence

$$\text{Gain}(\text{student}) = I(p, n) - E(\text{student})$$

$$\text{Gain}(\text{student}) = 0.940 - 0.7883$$

$$\text{Gain}(\text{student}) = 0.151$$

Attribute Selection by Information Gain (cont.)

- Class P: buys_computer = “yes”
- Class N: buys_computer = “no”
- $I(p, n) = I(9, 5) = 0.940$
- Compute the entropy for *credit_rating*:

credit	p_i	n_i	$I(p_i, n_i)$
excellent	3	3	0.9998
fair	6	2	0.5611

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

$$E(\text{credit_rating}) = \frac{6}{14} I(3,3) + \frac{8}{14} I(6,2)$$

$$E(\text{credit_rating}) = .4285 + .3206$$

Hence

$$\text{Gain}(\text{credit_rating}) = I(p, n) - E(\text{credit_rating})$$

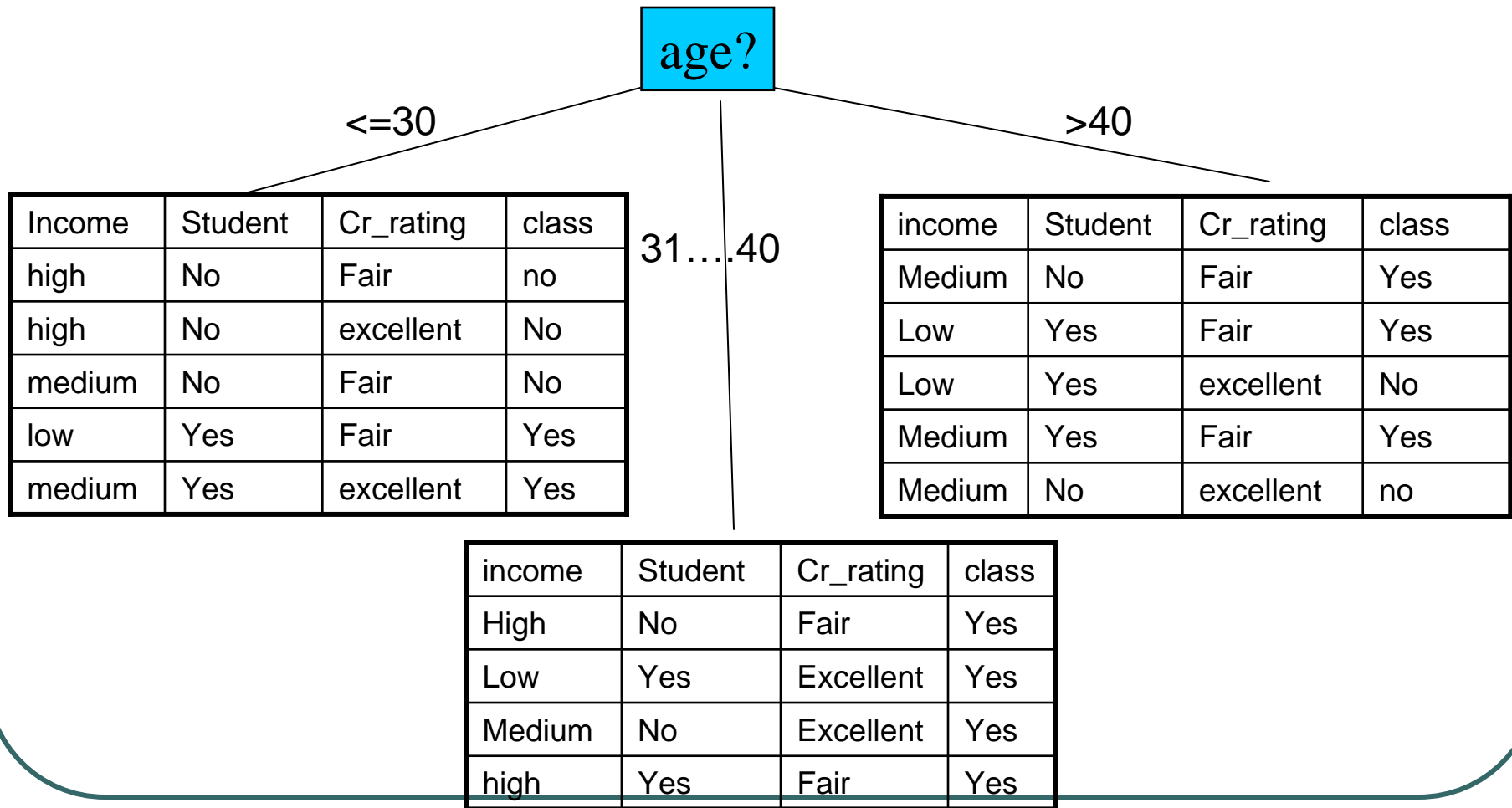
$$\text{Gain}(\text{credit_rating}) = 0.940 - .7491 = .19$$

$$\text{Gain}(\text{credit_rating}) = 0.19$$

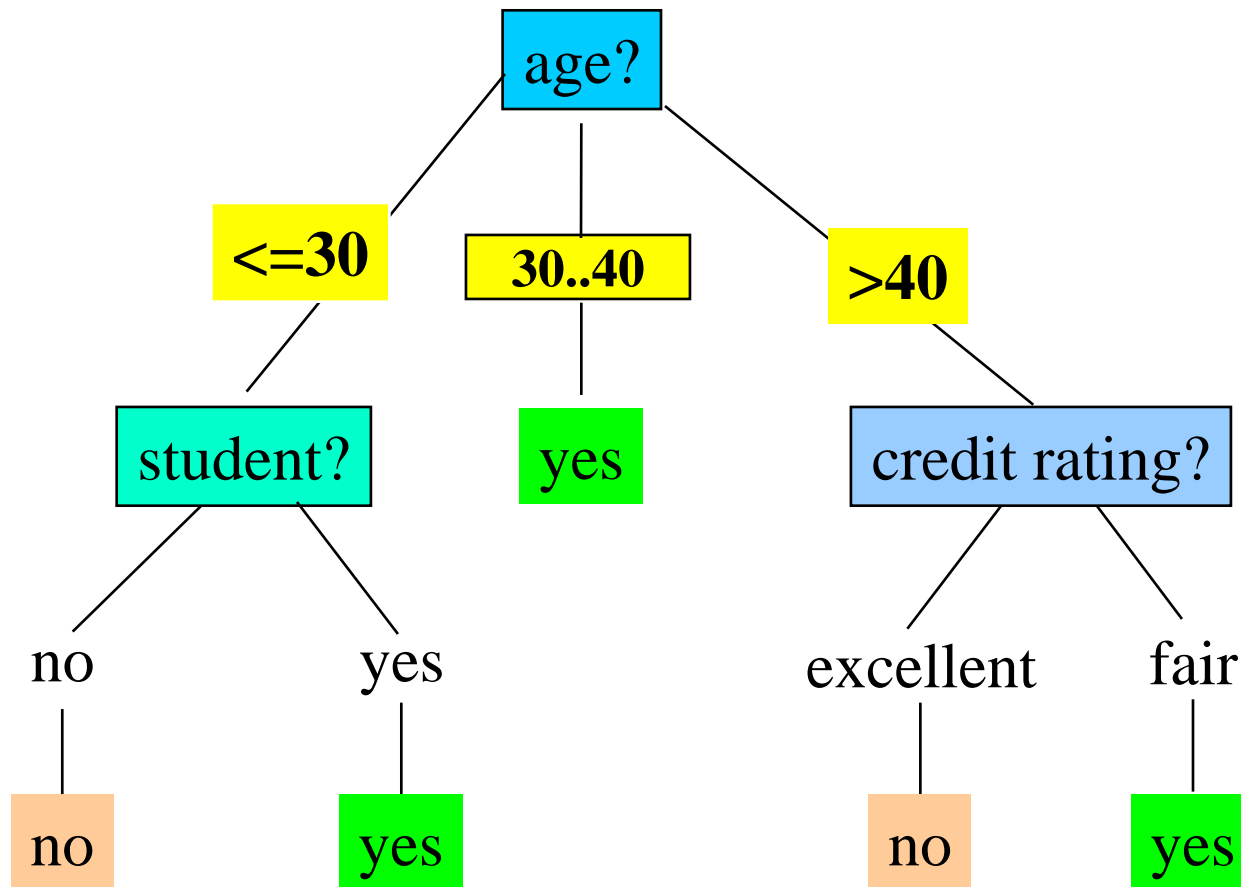
Selection of Split Attribute

- Information gains of attributes:
 - Gain (age) = 0.250
 - Gain (income) = 0.029
 - Gain (student) = 0.151
 - Gain (credit_rating) = 0.19
- Max (Gain (age), Gain (income), Gain (student), Gain (credit_rating))
- Max (0.250, 0.029, 0.151, 0.19) = 0.25
- Selected attribute: age (Maximum Information Gain)

Output: A Decision Tree for “*buys_computer*”



Output: A Decision Tree for “*buys_computer*”



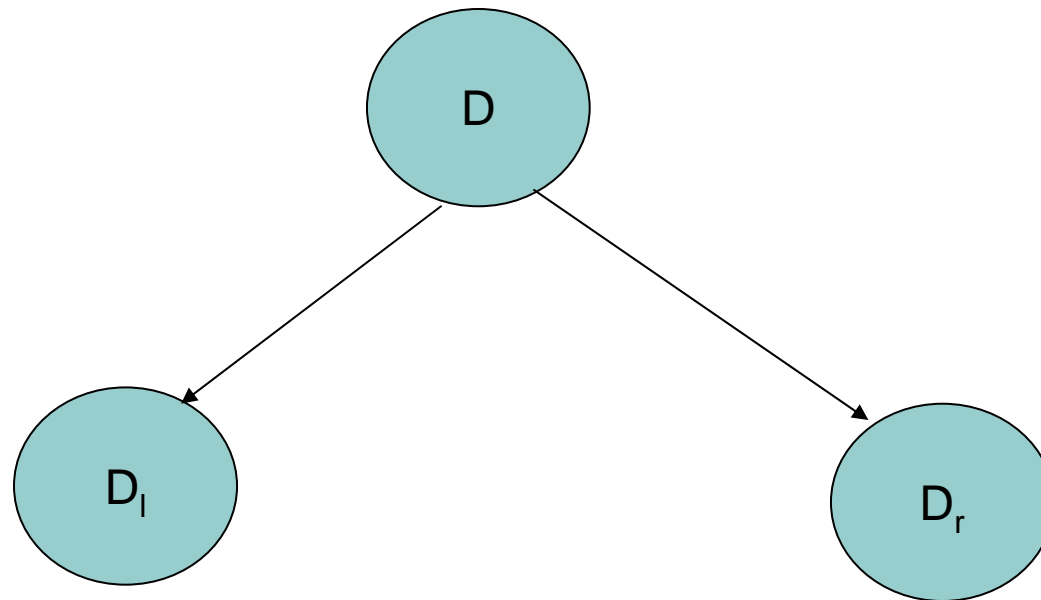
Basic Algorithm for inducing a decision tree Step - 1

- Divide the database into all possible combinations and find the “Best” database “split” point using entropy, or gini calculations.

Build Decision Tree Algorithm

Step - 2

- Split the database D into the two subdatabases, split by highest gain.



Build Decision Tree Algorithm

Step - 3

- If either of the two sub trees D_l and D_r is “pure” return their class labels and END.
- If their sub database are not pure, call BuildTree() on the impure databases.
- Buildtree(D_l) and/or Buildtree (D_r)

Where a “pure” sub database contains transactions of only one class