

Classification Lecture Notes (3)

(cse352)

DECISION TREE CLASSIFICATION

(Supervised Learning)

Professor Anita Wasilewska

Classification Learning

ALGORITHMS

Different Classifiers

- **DESCRIPTIVE:**
 - Decision Trees (ID3, C4.5)
 - Rough Sets
 - Genetic Algorithms
- **STATISTICAL:**
 - Neural Networks
 - Bayesian Networks

Classification Data

- **Data format:** a data table with key attribute removed. Special attribute- class attribute must be distinguished

age	income	student	credit_rating	buys_computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
30...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

Classification (Training) Data with objects

rec	Age	Income	Student	Credit_rating	Buys_computer (CLASS)
r1	<=30	High	No	Fair	No
r2	<=30	High	No	Excellent	No
r3	31...40	High	No	Fair	Yes
r4	>40	Medium	No	Fair	Yes
r5	>40	Low	Yes	Fair	Yes
r6	>40	Low	Yes	Excellent	No
r7	31...40	Low	Yes	Excellent	Yes
r8	<=30	Medium	No	Fair	No
r9	<=30	Low	Yes	Fair	Yes
r10	>40	Medium	Yes	Fair	Yes
r11	<=30	Medium	Yes	Excellent	Yes
r12	31...40	Medium	No	Excellent	Yes
r13	31...40	High	Yes	Fair	Yes
r14	>40	Medium	No	Excellent	No

Classification by Decision Tree Induction (1)

- Decision tree is

A flow-chart-like tree structure;

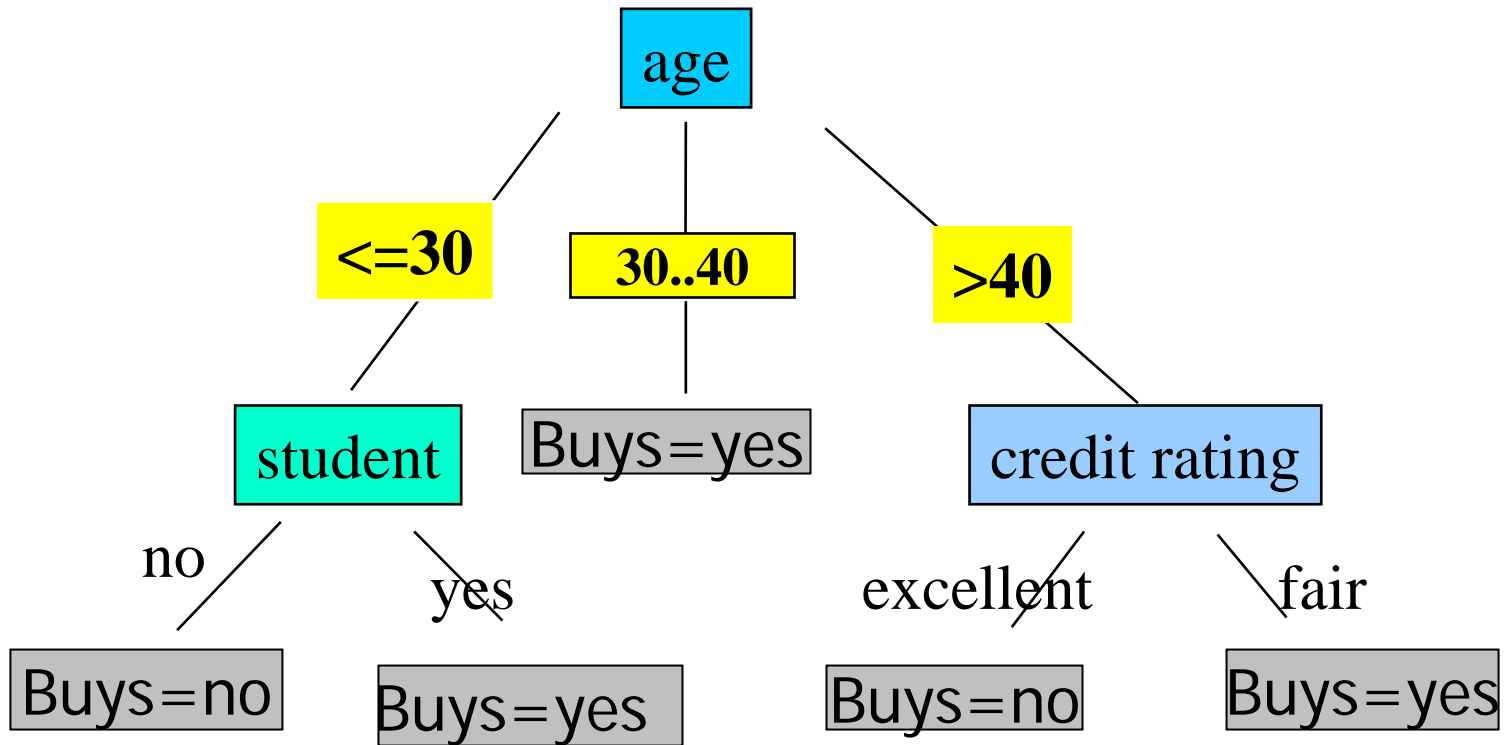
Internal node denotes an attribute;

Branch represents the values of the node attribute;

Leaf nodes represent class labels or class distribution

DECISION TREE

An Example



Classification

by Decision Tree Induction (2)

- **Decision tree generation** consists of two phases

Tree construction

- We choose recursively internal nodes (attributes) with their proper values as branches.
- **At start** we choose one attribute as the root and put all its 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 (records) left. In this case we apply **MAJORITY VOTING** to classify the node.
- **Majority Voting** involves converting the node into a leaf and labeling it with the most common class.
- Some algorithms allow majority voting at any level of the tree.

Tree pruning

- Identify and remove branches that reflect noise or outliers

Classification by Decision Tree Induction (3)

Crucial point

Good choice of the root attribute and internal nodes attributes is a crucial point.

Bad choice may result, in the worst case in a just another knowledge representation: relational table re-written as a tree with class attributes (decision attributes) as the leaves.

- **Decision Tree Induction Algorithms differ on methods of evaluating and choosing the root and internal nodes attributes.**

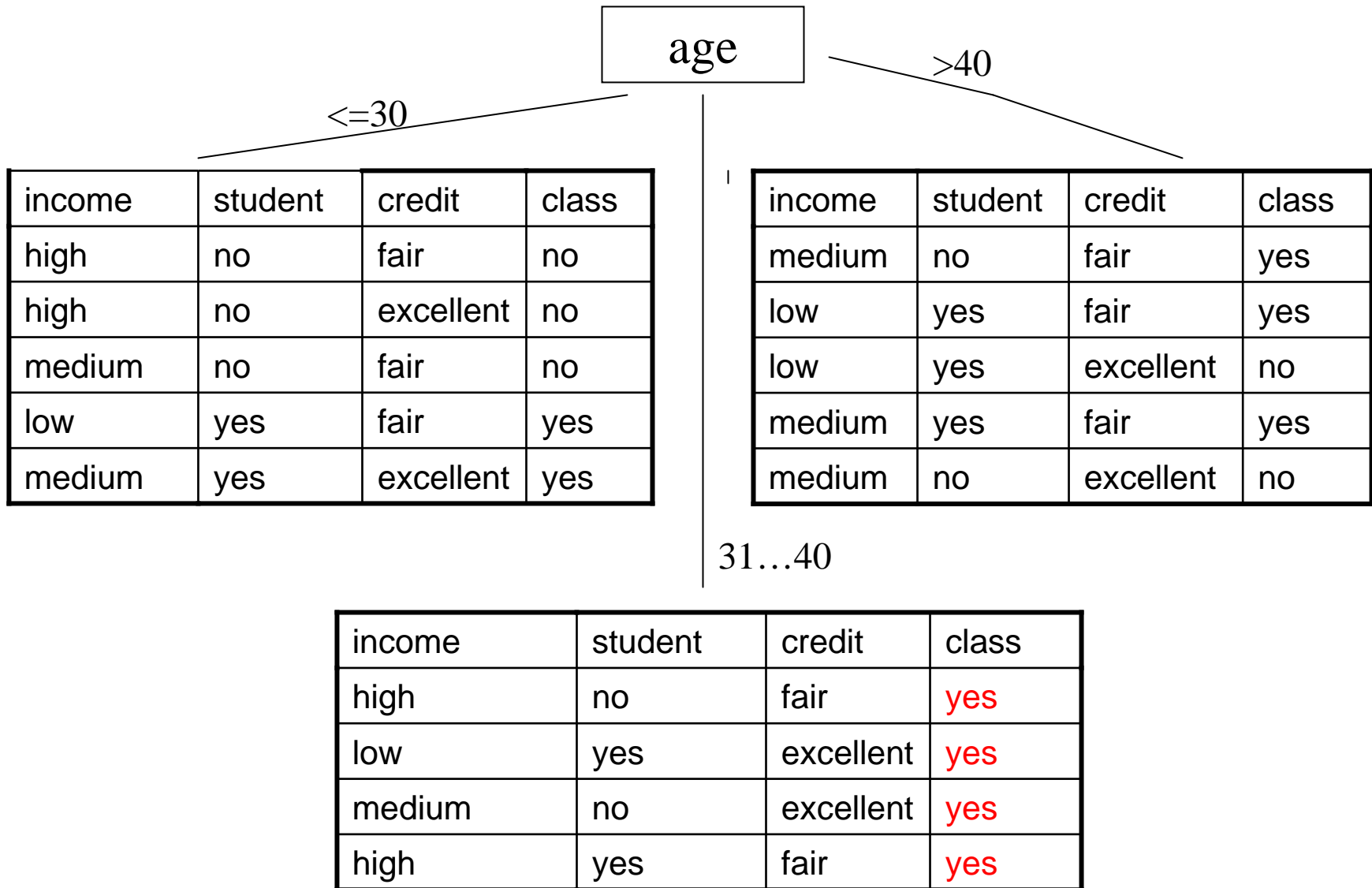
Decision Tree CONSTRUCTION: Example 1.

Consider our TRAINING Dataset (next slide)
We build the DECISION TREE choosing
the attribute AGE as the root of the tree.

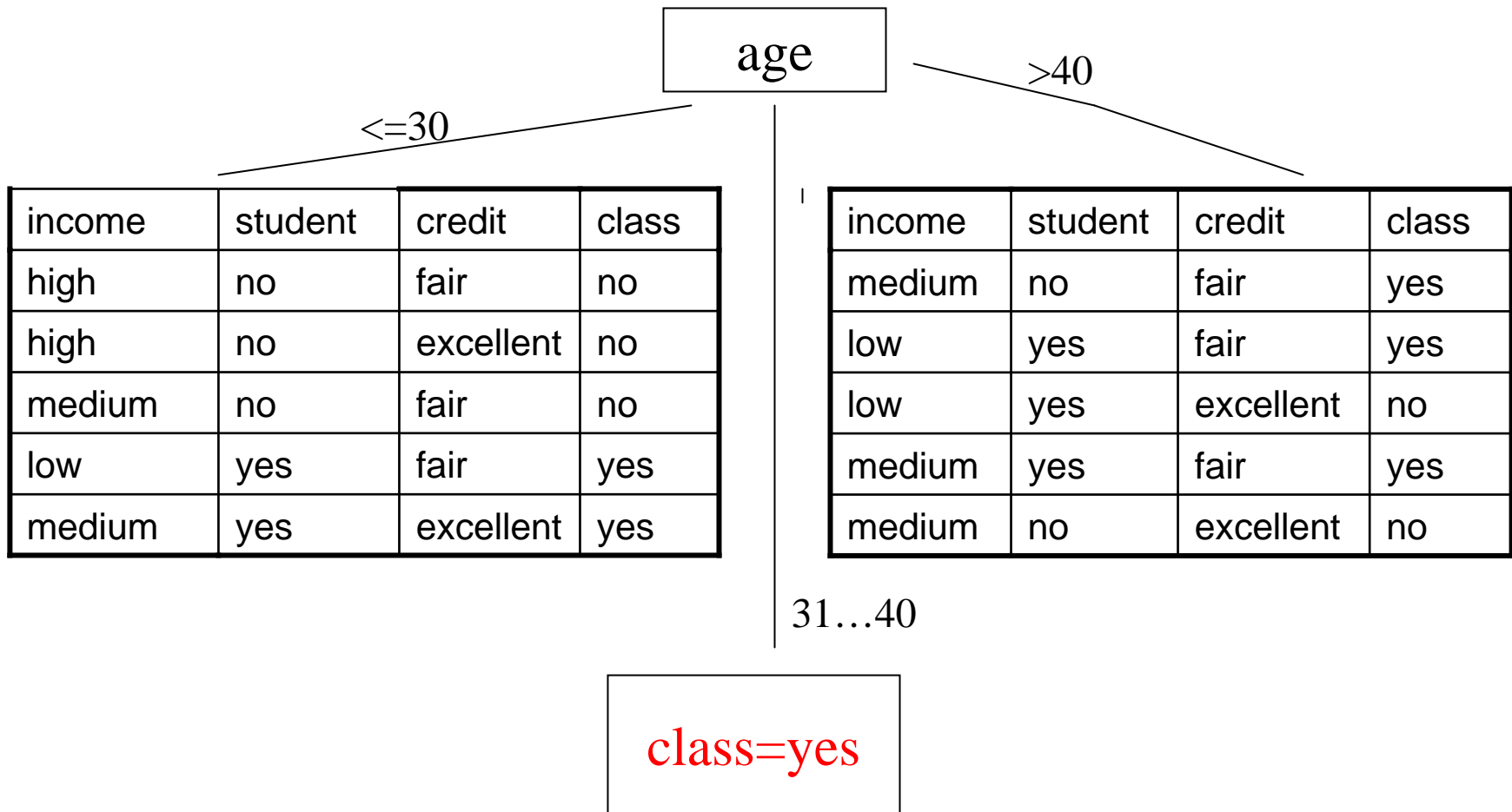
Training Data with objects

rec	Age	Income	Student	Credit_rating	Buys_computer(CLASS)
r1	<=30	High	No	Fair	No
r2	<=30	High	No	Excellent	No
r3	31...40	High	No	Fair	Yes
r4	>40	Medium	No	Fair	Yes
r5	>40	Low	Yes	Fair	Yes
r6	>40	Low	Yes	Excellent	No
r7	31...40	Low	Yes	Excellent	Yes
r8	<=30	Medium	No	Fair	No
r9	<=30	Low	Yes	Fair	Yes
r10	>40	Medium	Yes	Fair	Yes
r11	<=30	Medium	Yes	Excellent	Yes
r12	31...40	Medium	No	Excellent	Yes
r13	31...40	High	Yes	Fair	Yes
r14	>40	Medium	No	Excellent	No

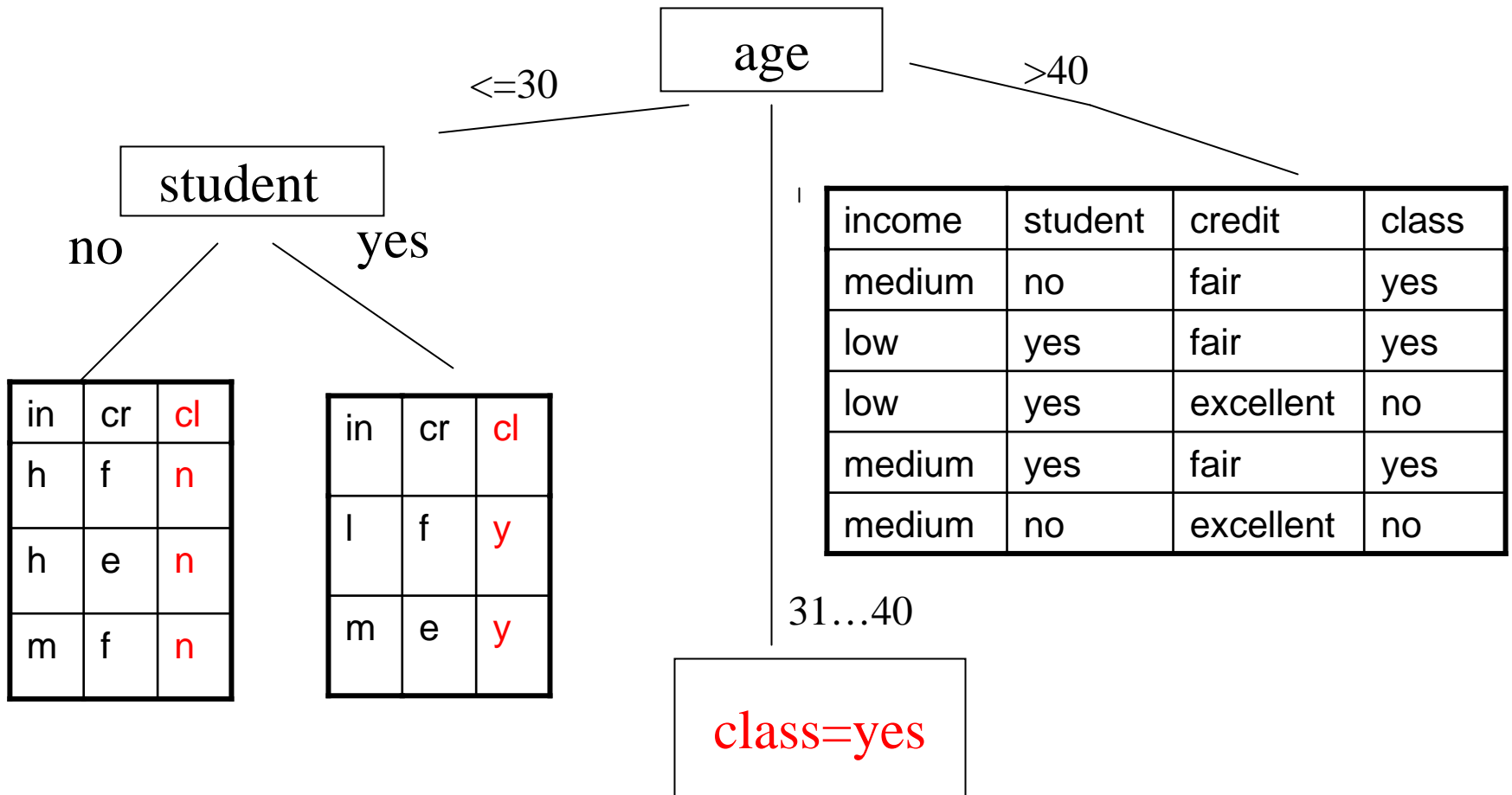
Building The Tree: we choose “age” as a root



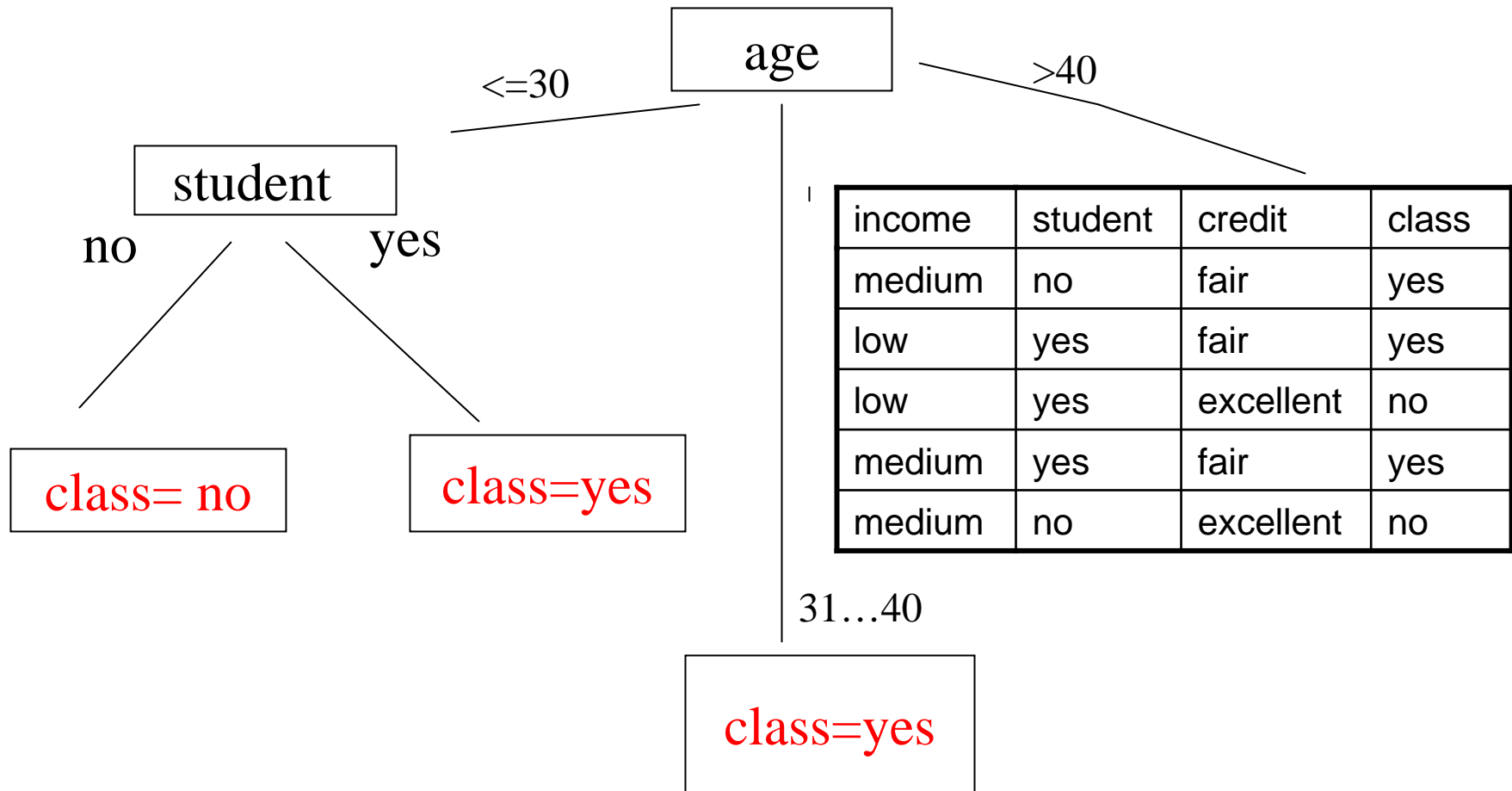
Building The Tree: “age” as the root



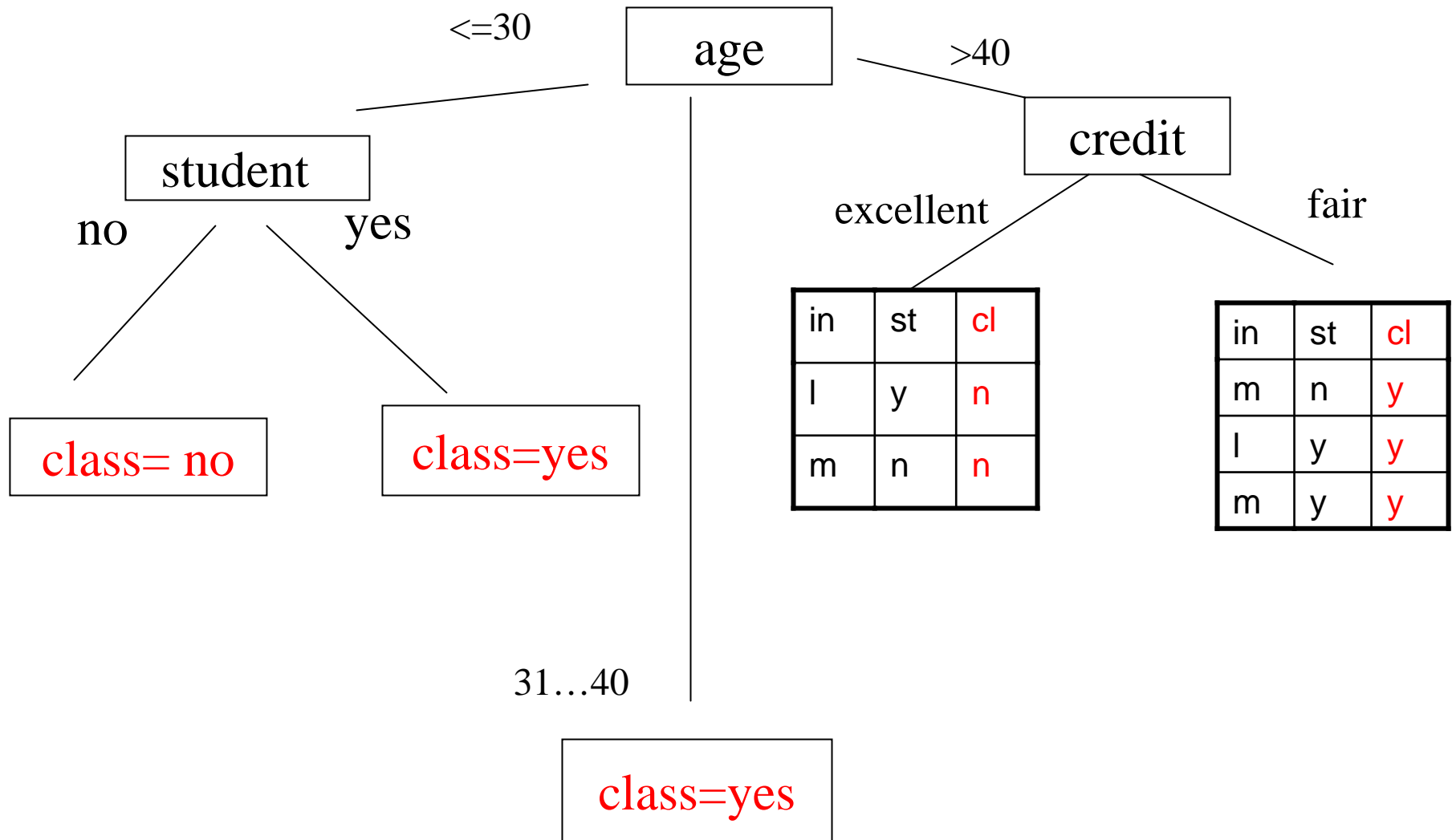
Building The Tree: we chose "student" on ≤ 30 branch



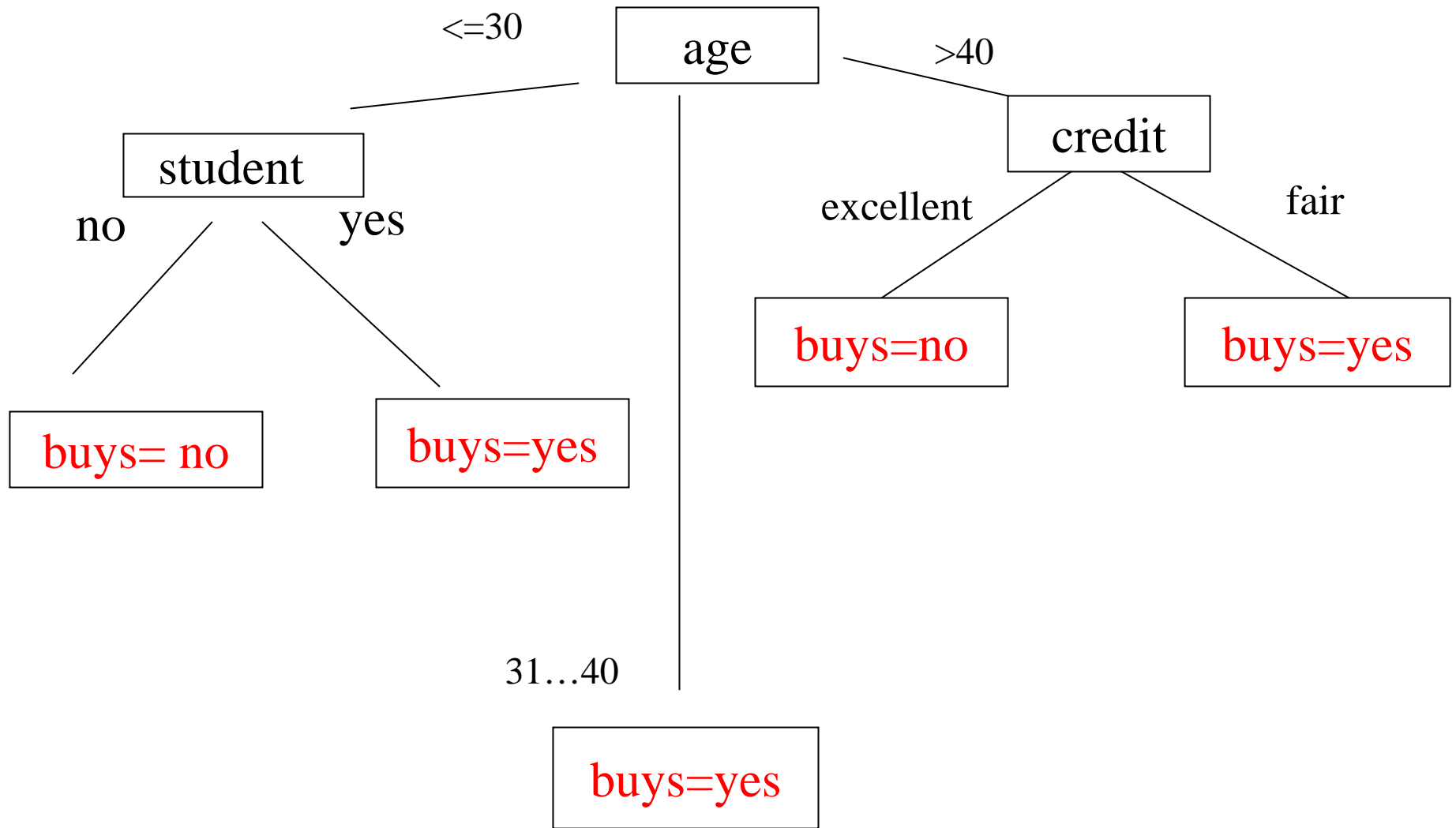
Building The Tree: we chose "student" on ≤ 30 branch



Building The Tree: we chose "credit" on >40 branch



Finished Tree for class="buys"



Extracting Classification Rules from Trees

- Goal: Represent the knowledge in the form of
- **IF-THEN** rules (determinant);
- One rule is created for each path from the root to a leaf;
- Each attribute-value pair along a path forms a conjunction;
- The leaf node holds the class prediction;
- **Rules are easier for humans to understand**

RULES (Discriminant) extracted from our TREE

- The rules are:

IF *age* = “<=30” AND *student* = “no” THEN *buys_computer* = “no”

IF *age* = “<=30” AND *student* = “yes” THEN *buys_computer* = “yes”

IF *age* = “31...40” THEN
buys_computer = “yes”

IF *age* = “>40” AND *credit_rating* = “excellent” THEN
buys_computer = “no”

IF *age* = “>40” AND *credit_rating* = “fair” THEN
buys_computer = “yes”

Rules format for testing and applications

- In order to use rules for testing, and later when testing is done and predictive accuracy is acceptable we write rules in a predicate form:

IF *age(x, <=30)* AND *student(x, no)* THEN

buys_computer(x, no)

IF *age(x, <=30)* AND *student(x, yes)* THEN

buys_computer(x, yes)

- Attributes and their values of the new record *x* are matched with the IF part of the rule and the record is classified accordingly to the THEN part of the rule.

Exercise: Predictive Accuracy

Calculate the predictive accuracy of our set of rules with respect of the TEST data given by the next slide.

R1: IF *age* = “<=30” AND *student* = “no” THEN
buys_computer = “no”

R2: IF *age* = “<=30” AND *student* = “yes” THEN
buys_computer = “yes”

R3: IF *age* = “31...40” THEN
buys_computer = “yes”

R4: IF *age* = “>40” AND *credit_rating* = “excellent”
THEN *buys_computer* = “no”

R5: IF *age* = “>40” AND *credit_rating* = “fair” THEN
buys_computer = “yes”

TEST Data for predictive accuracy evaluation

rec	Age	Income	Student	Credit_rating	Buys_computer(CLASS)
r1	<=30	Low	No	Fair	yes
r2	<=30	High	yes	Excellent	No
r3	<=30	High	No	Fair	Yes
r4	31...40	Medium	yes	Fair	Yes
r5	>40	Low	Yes	Fair	Yes
r6	>40	Low	Yes	Excellent	yes
r7	31...40	High	Yes	Excellent	Yes
r8	<=30	Medium	No	Fair	No
r9	31...40	Low	no	Excellent	Yes
r10	>40	Medium	Yes	Fair	Yes

Basic Idea of ID3/C4.5 Algorithm (1)

- The basic algorithm for decision tree induction is a greedy algorithm that constructs decision trees in a top-down recursive divide-and – conquer manner.
- The basic strategy is as follows.
- Tree **STARTS** as a single node representing all training dataset (data table with records called samples)
- **IF** the samples (records in the data table) are ALL in the same class, **THEN** the node becomes a **LEAF** and is **labeled with that class**
(some algorithms apply majority voting or other method to decide the class on the leaf at any level of the tree)
- **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 (split the data table) 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 (data table at the node) are partitioned accordingly
- 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 descendants
- The recursive partitioning **STOPS** only when any one of the following conditions is true.

Basic Idea of ID3/C4.5 Algorithm (3)

- TERMINATION CONDITIONS:
- All records (samples) for the given node belong to the same class or
- There are no remaining attributes on which the samples (records in the data table) may be further partitioned.
- 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

Heuristics: Attribute Selection Measures

- Construction of the tree depends on the order in which root attributes are selected.
- Different choices produce different trees; some better, some worse
- Shallower trees are better; they are the ones in which classification is reached in fewer levels.
- These trees are said to be more efficient as the classification, and hence termination is reached quickly

Attribute Selection Measures

- Given a training data set (set of training samples) there are many ways to choose the root and nodes attributes while constructing the decision tree
- **Some possible choices:**
- Random
- Attribute with smallest/largest number of values
- Following certain order of attributes
- We present here a special order: **information gain** as a measure of **the goodness of the split**
- The attribute with the highest information gain is always chosen as the split decision attribute for the current node while building the tree.

Information Gain Computation (ID3/C4.5): Case of Two Classes

- Assume there are two classes, P (positive) and N (negative)

Let S be a training data set consisting of s examples (records):

$$|S|=s$$

And S contains p elements of class P and n elements of class N

The amount of information, needed to decide if an arbitrary example (record) 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 Measure

- Assume that using attribute **A** a set S will be partitioned into sets $\{S_1, S_2, \dots, S_v\}$ (v is number of values of the attribute A)

If S_i contains p_i examples of P and n_i examples of N , the entropy $E(A)$, or the expected information needed to classify objects in all sub-trees 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

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

Example: Attribute Selection by Information Gain Computation

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

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

$$E(\text{age}) = \frac{5}{14} I(2,3) + \frac{4}{14} I(4,0) + \frac{5}{14} I(3,2) = 0.694$$

Hence

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

$$Gain(\text{age}) = 0.246$$

Similarly

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

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

$$Gain(\text{credit_rating}) = 0.048$$

The attribute “age” becomes the root.

Decision Tree Induction, Predictive Accuracy and Information Gain Calculations

EXAMPLES

Example 1

TASK: 1. Use Decision Tree Induction algorithm

without majority voting. Use different choices of the root attribute to FIND discriminant rules that determine whether a person buys a computer or not.

2. Compute Information gain for all nodes of the tree.

1. We choose attribute *buys_computer* as the CLASS attribute.
2. We perform DT algorithm “by hand” using different choices of the root attribute, and different “by hand” choices of the following nodes.
3. We build two trees with attributes: *Income* and *Credit Rating* respectively, as the root attribute to derive rules

Training Data

age	income	student	credit_rating	buys_computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
30...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

Training Data with objects

rec	Age	Income	Student	Credit_rating	Buys_computer
r1	<=30	High	No	Fair	No
r2	<=30	High	No	Excellent	No
r3	31...40	High	No	Fair	Yes
r4	>40	Medium	No	Fair	Yes
r5	>40	Low	Yes	Fair	Yes
r6	>40	Low	Yes	Excellent	No
r7	31...40	Low	Yes	Excellent	Yes
r8	<=30	Medium	No	Fair	No
r9	<=30	Low	Yes	Fair	Yes
r10	>40	Medium	Yes	Fair	Yes
r11	<=30	Medium	Yes	Excellent	Yes
r12	31...40	Medium	No	Excellent	Yes
r13	31...40	High	Yes	Fair	Yes
r14	>40	Medium	No	Excellent	No

Index: 1

Gain=0.027

Income

Low

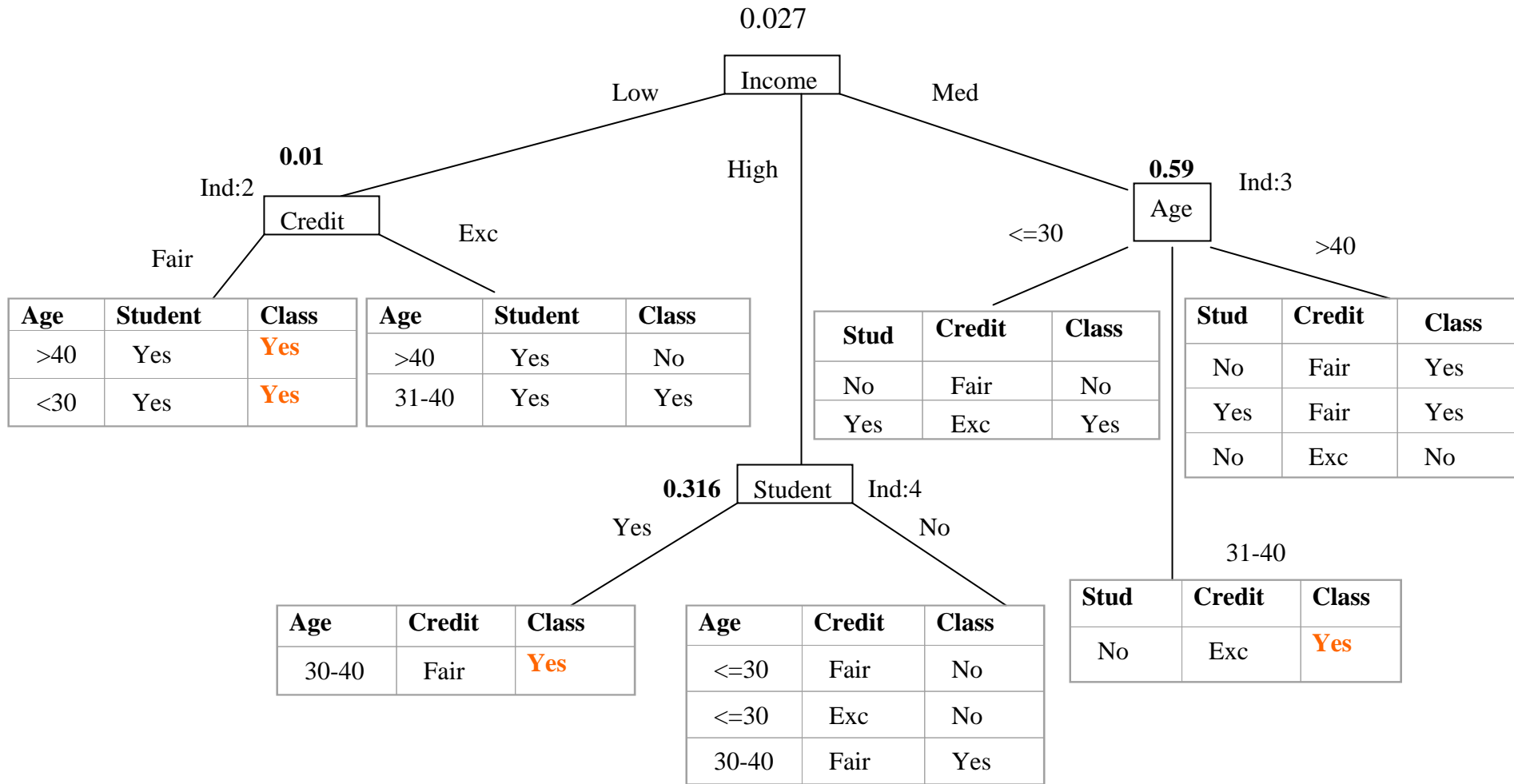
Med

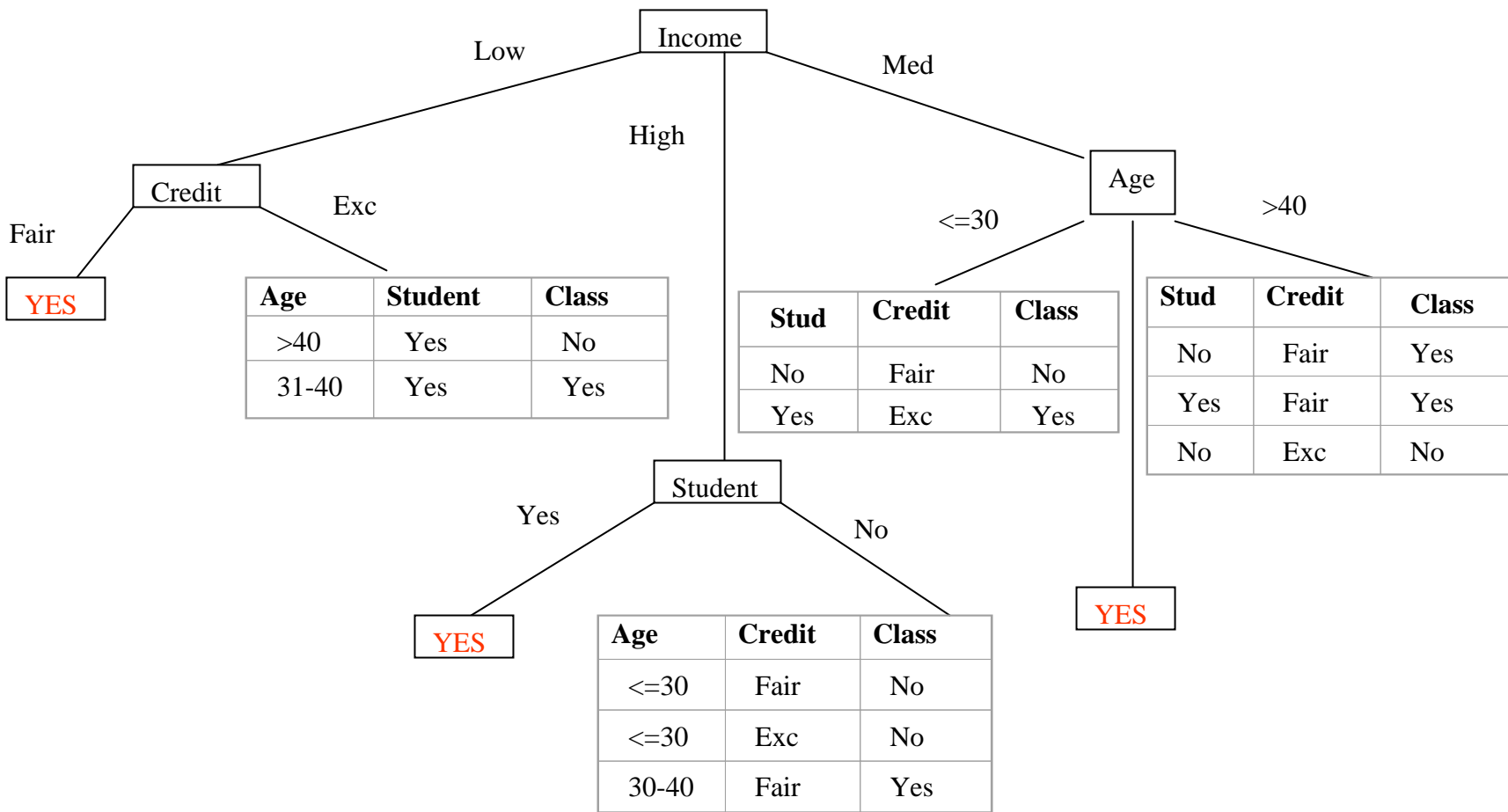
High

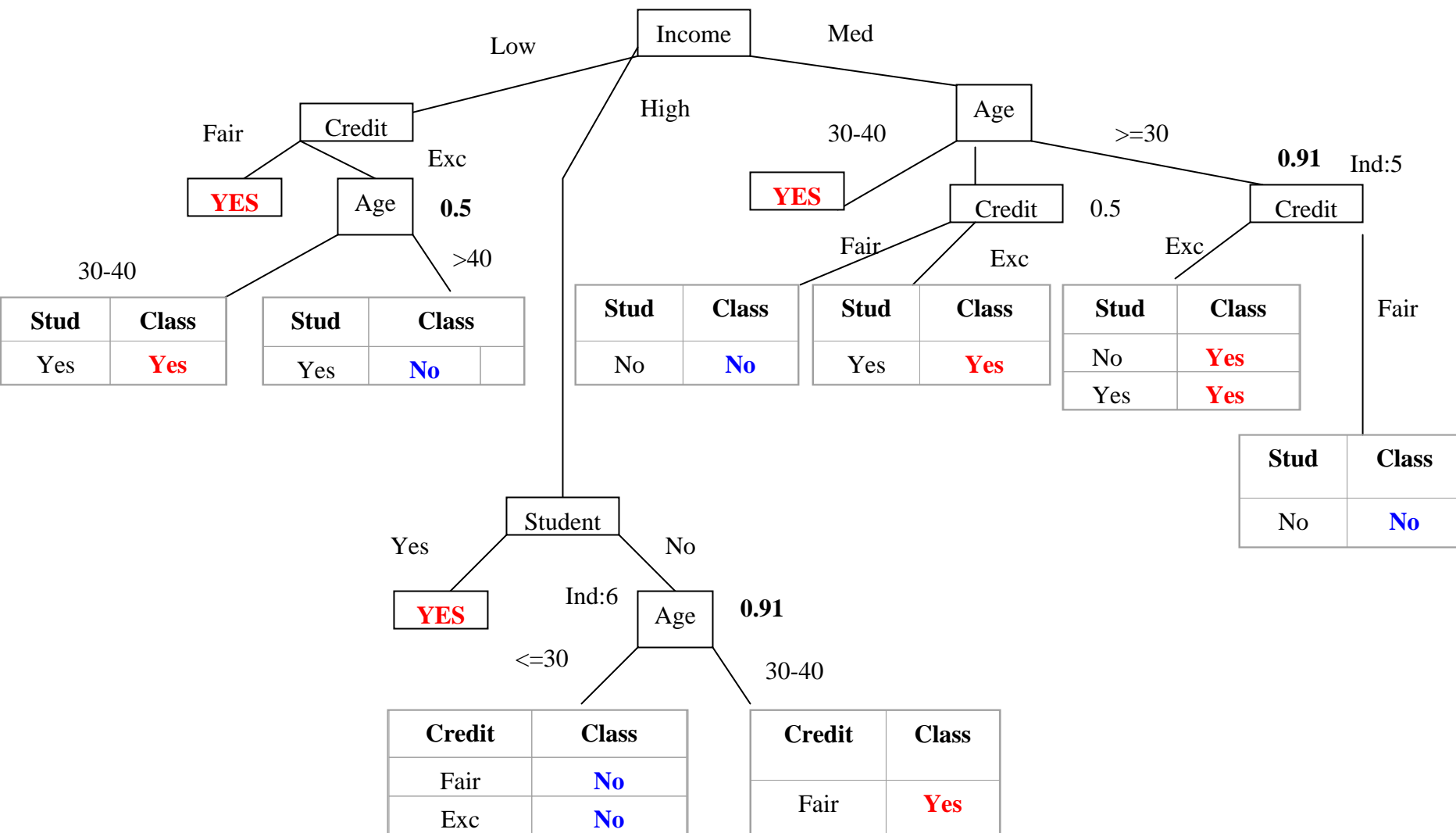
Age	Student	Credit	Class
>40	No	Fair	Yes
<=30	No	Fair	No
>40	Yes	Fair	Yes
<=30	Yes	Exc	Yes
31-40	No	Exc	Yes
>40	No	Exc	No

Age	Student	Credit	Class
>40	No	Fair	Yes
<=30	No	Fair	No
>40	Yes	Fair	Yes
<=30	Yes	Exc	Yes
31-40	No	Exc	Yes
>40	No	Exc	No

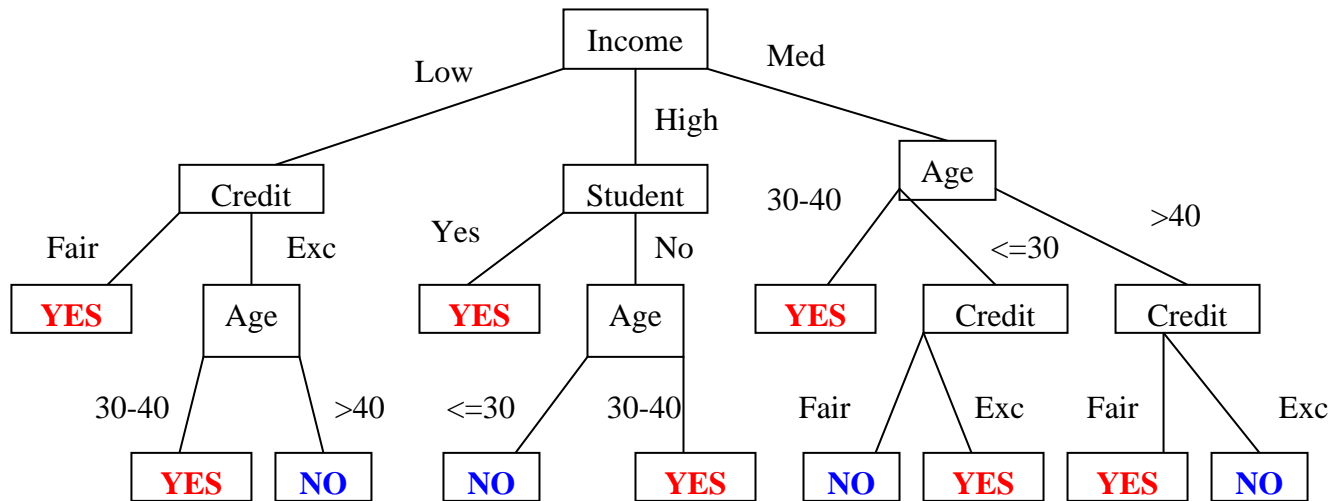
Age	Student	Credit	Class
<=30	No	Fair	No
<=30	No	Exc	No
30-40	No	Fair	Yes
30-40	Yes	Fair	Yes







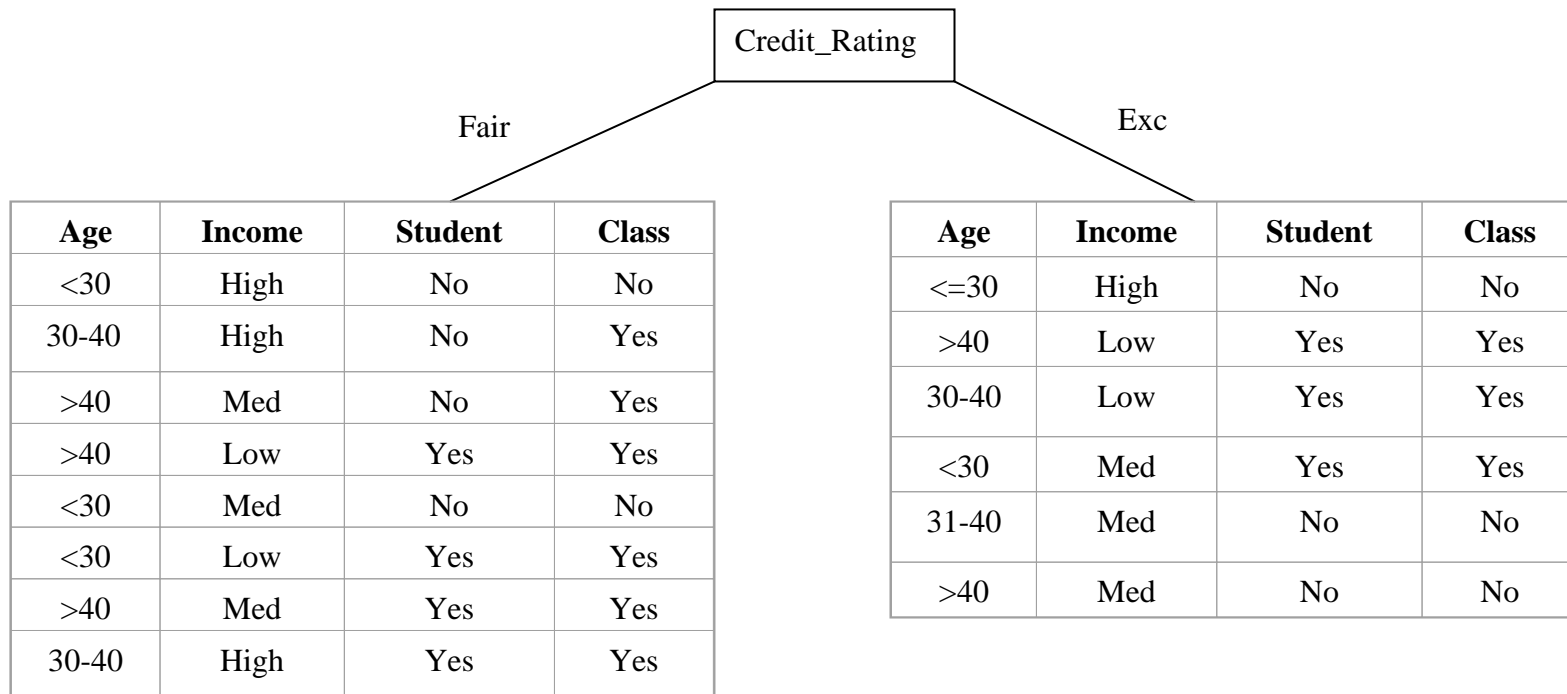
Tree 1 with root attribute Income

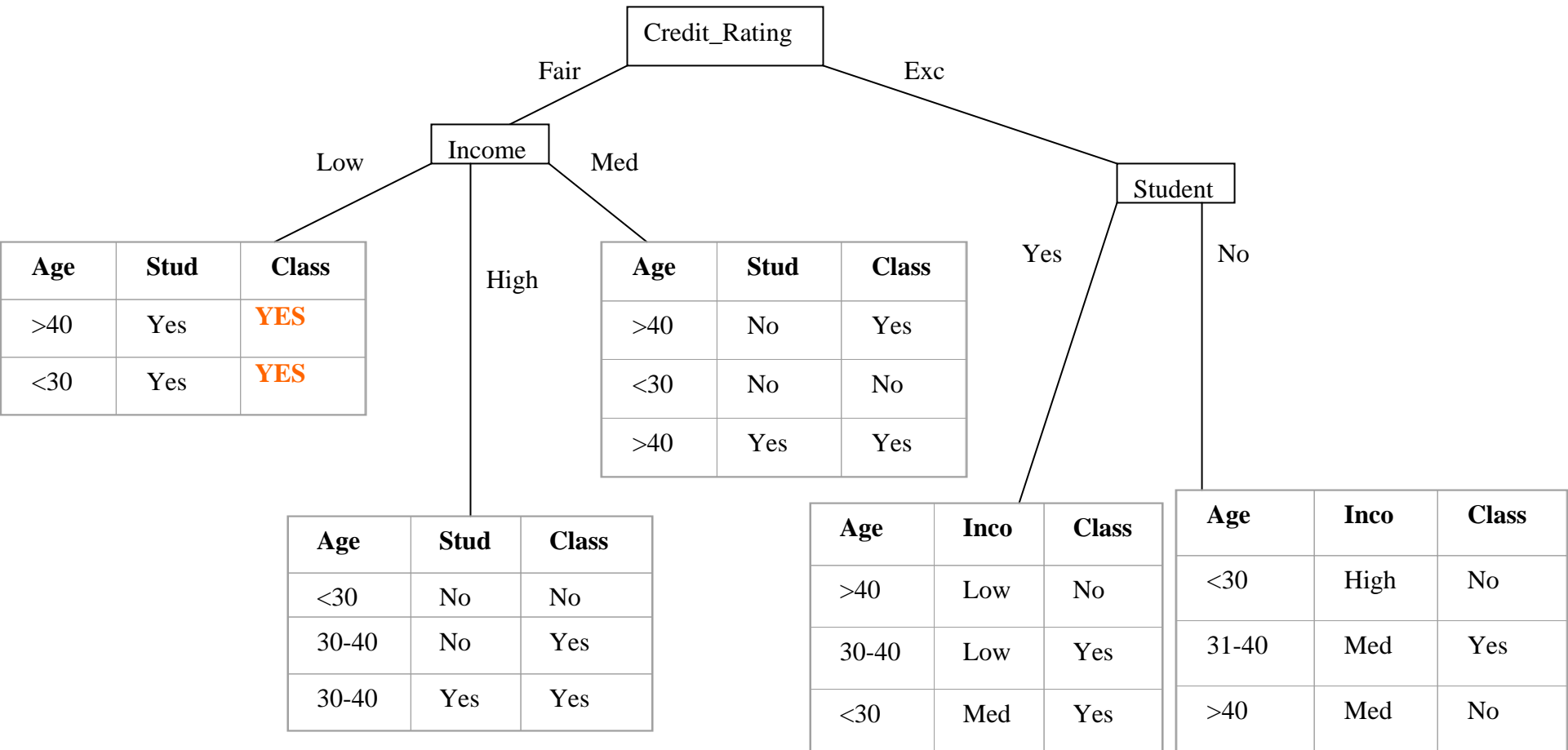


Rules derived from tree 1 (predicate form for testing)

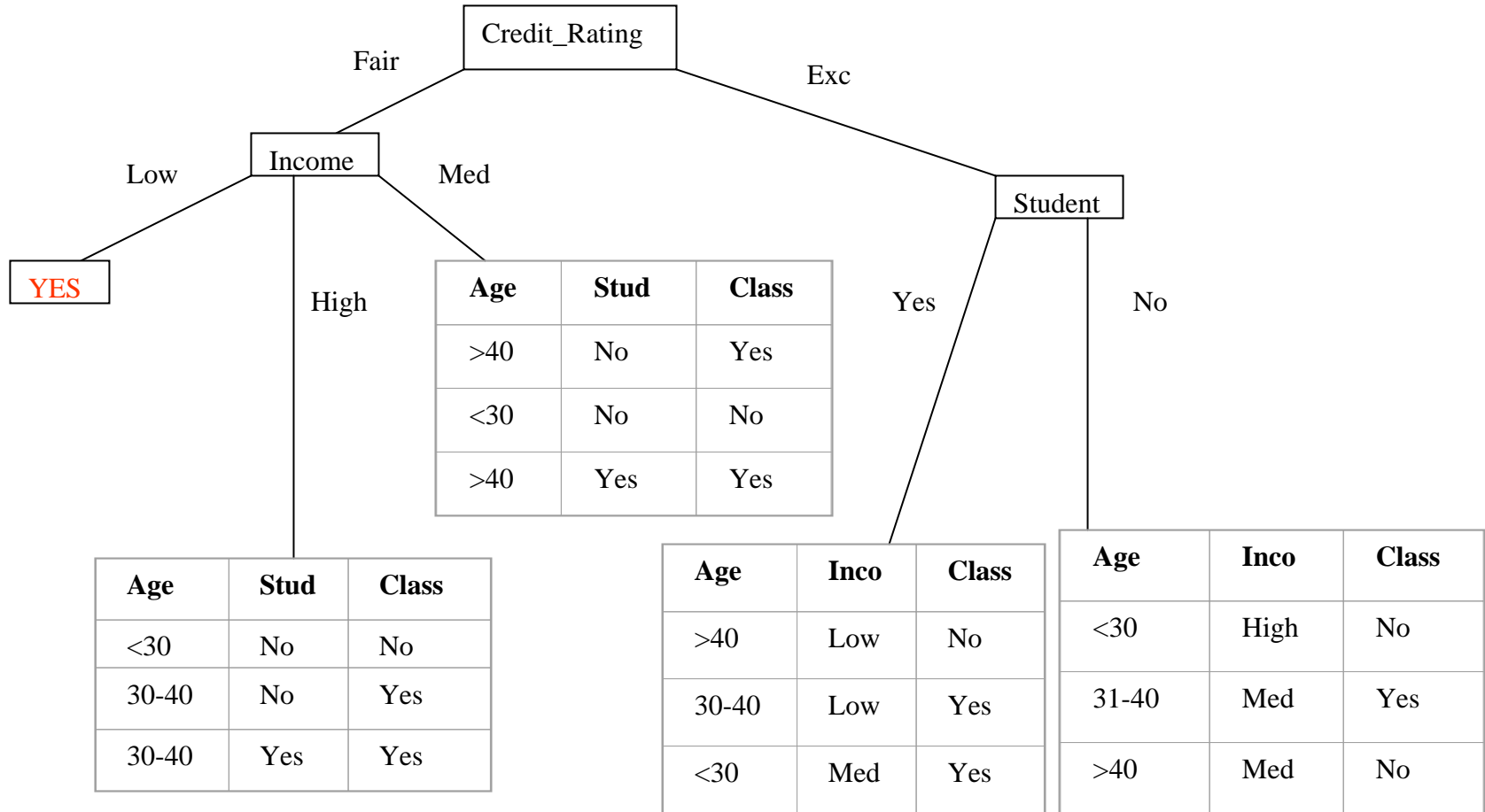
1. $\text{Income}(x, \text{Low}) \wedge \text{Credit}(x, \text{Fair}) \rightarrow \text{buysComputer}(x, \text{Yes})$.
2. $\text{Income}(x, \text{Low}) \wedge \text{Credit}(x, \text{Exc}) \wedge \text{Age}(x, 30-40) \rightarrow \text{buysComputer}(x, \text{Yes})$.
3. $\text{Income}(x, \text{Low}) \wedge \text{Credit}(x, \text{Exc}) \wedge \text{Age}(x, >40) \rightarrow \text{buysComputer}(x, \text{No})$.
4. $\text{Income}(x, \text{High}) \wedge \text{Student}(x, \text{Yes}) \rightarrow \text{buysComputer}(x, \text{Yes})$.
5. $\text{Income}(x, \text{High}) \wedge \text{Student}(x, \text{No}) \wedge \text{Age}(x, <30) \rightarrow \text{buysComputer}(x, \text{No})$.
6. $\text{Income}(x, \text{High}) \wedge \text{Student}(x, \text{No}) \wedge \text{Age}(x, 30-40) \rightarrow \text{buysComputer}(x, \text{Yes})$.
7. $\text{Income}(x, \text{Medium}) \wedge \text{Age}(x, 30-40) \rightarrow \text{buysComputer}(x, \text{Yes})$.
8. $\text{Income}(x, \text{Medium}) \wedge \text{Age}(x, <30) \wedge \text{Credit}(x, \text{Fair}) \rightarrow \text{buysComputer}(x, \text{No})$.
9. $\text{Income}(x, \text{Medium}) \wedge \text{Age}(x, <30) \wedge \text{Credit}(x, \text{Exc}) \rightarrow \text{buysComputer}(x, \text{Yes})$.
10. $\text{Income}(x, \text{Medium}) \wedge \text{Age}(x, >40) \wedge \text{Credit}(x, \text{Fair}) \rightarrow \text{buysComputer}(x, \text{Yes})$.
11. $\text{Income}(x, \text{Medium}) \wedge \text{Age}(x, >40) \wedge \text{Credit}(x, \text{Exc}) \rightarrow \text{buysComputer}(x, \text{No})$.

Tree 2 with root attribute Credit Rating

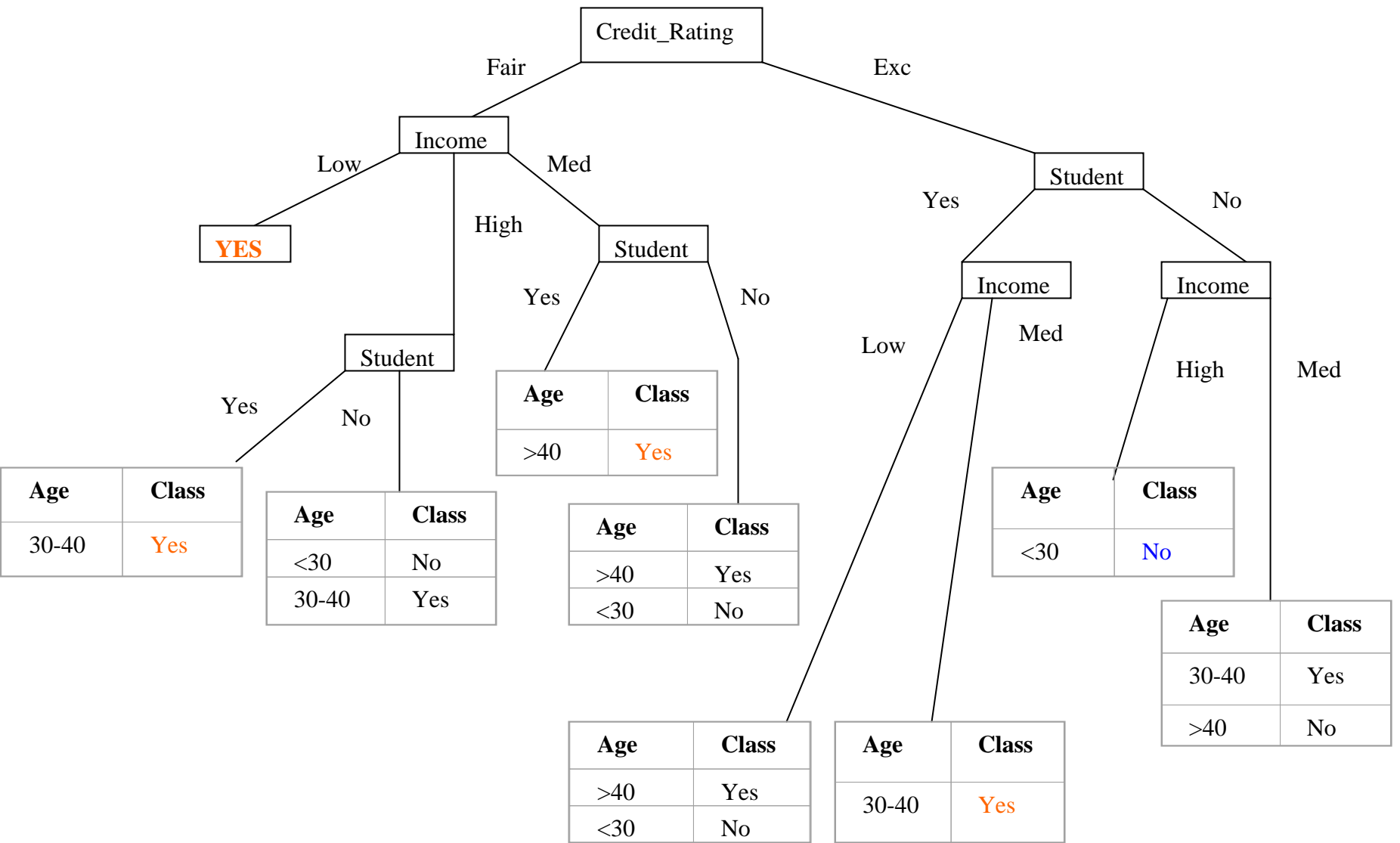


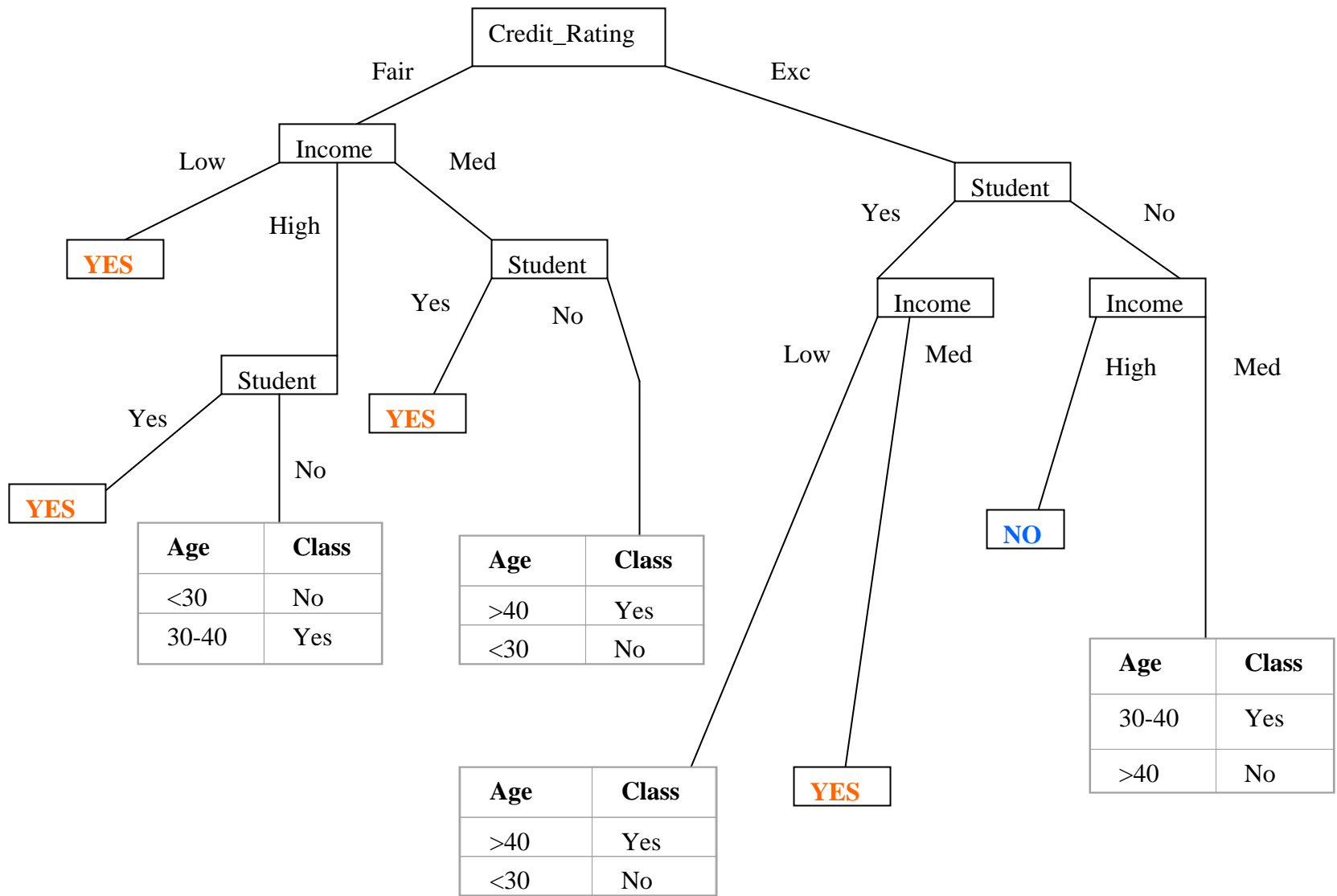


Tree 2 with next level attributes Income ans Student



Tree 2 with root attribute Credit Rating





YES

YES

YES

Age	Class
<30	No
30-40	Yes

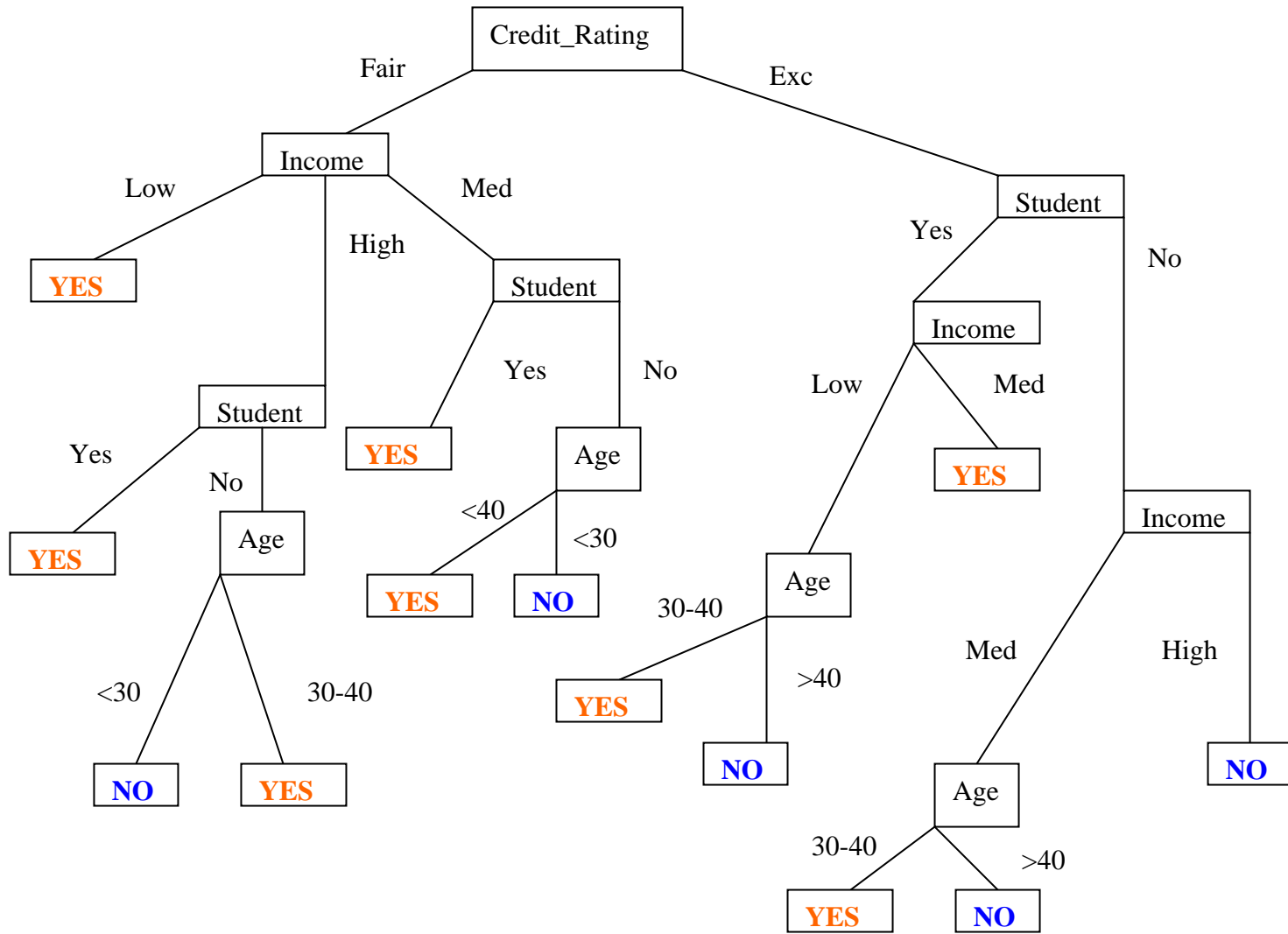
Age	Class
>40	Yes
<30	No

Age	Class
>40	Yes
<30	No

NO

YES

Age	Class
30-40	Yes
>40	No



Final Tree 2 with root attribute Credit Rating

The Decision tree with root attribute *Credit_Rating* has produced twelve rules, one more than with root attribute *Income*

1. $\text{Credit}(x, \text{Fair}) \wedge \text{Income}(x, \text{Low}) \rightarrow \text{buysComp}(x, \text{Yes})$.
2. $\text{Credit}(x, \text{Fair}) \wedge \text{Income}(x, \text{High}) \wedge \text{Student}(x, \text{Yes}) \rightarrow \text{buysComp}(x, \text{Yes})$.
3. $\text{Credit}(x, \text{Fair}) \wedge \text{Income}(x, \text{High}) \wedge \text{Student}(x, \text{No}) \wedge \text{Age}(<30) \rightarrow \text{buysComp}(x, \text{No})$.
4. $\text{Credit}(x, \text{Fair}) \wedge \text{Income}(x, \text{High}) \wedge \text{Student}(x, \text{No}) \wedge \text{Age}(30-40) \rightarrow \text{buysComp}(x, \text{Yes})$.
5. $\text{Credit}(x, \text{Fair}) \wedge \text{Income}(x, \text{Med}) \wedge \text{Student}(x, \text{Yes}) \rightarrow \text{buysComp}(x, \text{Yes})$.
6. $\text{Credit}(x, \text{Fair}) \wedge \text{Income}(x, \text{Med}) \wedge \text{Student}(x, \text{No}) \wedge \text{Age}(<40) \rightarrow \text{buysComp}(x, \text{Yes})$.
7. $\text{Credit}(x, \text{Fair}) \wedge \text{Income}(x, \text{Med}) \wedge \text{Student}(x, \text{No}) \wedge \text{Age}(<30) \rightarrow \text{buysComp}(x, \text{No})$.
8. $\text{Credit}(x, \text{Exc}) \wedge \text{Student}(x, \text{Yes}) \wedge \text{Income}(x, \text{High}) \wedge \text{Age}(30-40) \rightarrow \text{buysComp}(x, \text{Yes})$.
9. $\text{Credit}(x, \text{Exc}) \wedge \text{Student}(x, \text{Yes}) \wedge \text{Income}(x, \text{Med}) \wedge \text{Age}(>40) \rightarrow \text{buysComp}(x, \text{No})$.
10. $\text{Credit}(x, \text{Exc}) \wedge \text{Student}(\text{No}) \wedge \text{Income}(x, \text{High}) \rightarrow \text{buysComp}(x, \text{Yes})$.
11. $\text{Credit}(x, \text{Exc}) \wedge \text{Student}(x, \text{No}) \wedge \text{Income}(x, \text{Med}) \wedge \text{Age}(x, 30-40) \rightarrow \text{buysComp}(x, \text{Yes})$.
12. $\text{Credit}(x, \text{Exc}) \wedge \text{Student}(x, \text{No}) \wedge \text{Income}(x, \text{Med}) \wedge \text{Age}(x, >40) \rightarrow \text{buysComp}(x, \text{No})$.

Example 2

- We use some random records (tuples) to calculate the **Predictive Accuracy** of the set of rules from the Example 1.

Predictive Accuracy is the % of well classified records not from training set for which the class attribute is known.

Random Tuples to Check Predictive Accuracy based on three sets of rules

Obj	Age	Income	Student	Credit_R	Class
1	<=30	High	Yes	Fair	Yes
2	31-40	Low	No	Fair	Yes
3	31-40	High	Yes	Exc	No
4	>40	Low	Yes	Fair	Yes
5	>40	Low	Yes	Exc	No
6	<=30	Low	No	Fair	No

Predictive accuracy:

1. Against Lecture Notes: $4/6 = 66.66\%$
2. Against Tree 1 rules with root att. *Income*: $3/6 = 50\%$
3. Against Tree 2 rules with root att. *Credit*: $5/6 = 83.33\%$

Example 3

- **Predictive accuracy depends heavily on a choice of the test and training data.**
- Find a small set of TEST records such that they would give a predictive accuracy 100% for rules from examples 1 and 2.

1. TEST DATA applied against rules in Lecture Notes
that gives predictive accuracy 100%

No	Age	Income	Student	Credit_R	Class
1	<=30	Med	No	Exc	No
2	<=30	High	Yes	Fair	Yes
3	31-40	Low	No	Exc	Yes
4	>40	High	Yes	Exc	No
5	<=30	Low	No	Fair	Yes
6	31-40	High	Yes	Fair	Yes

2. TEST DATA that applied against the rules with root attribute *Income* give predictive accuracy 100%

No	Age	Income	Student	Credit_R	Class
1	31-40	Low	Yes	Fair	Yes
2	>40	Low	No	Exc	No
3	<=30	High	Yes	Fair	Yes
4	31-40	High	No	Exc	Yes
5	31-40	Med	No	Fair	Yes
6	>40	Med	Yes	Exc	No

3. TEST DATA that applied against the rules with root attribute *Credit Rating* gives predictive accuracy 100%

No	Age	Income	Student	Credit_R	Class
1	31-40	Low	No	Fair	Yes
2	<=30	High	Yes	Fair	Yes
3	<=30	Med	No	Fair	No
4	31-40	High	Yes	Exc	Yes
5	>40	Med	Yes	Exc	No
6	>40	Med	No	Exc	No

Calculation of Information gain at each level of tree with root att. *Income*

1. Original Table:

Class P: *buys_computer* = yes; Class N: *buys_computer* = No

$$I(P,N) = -\frac{P}{P+N} \log_2 \left(\frac{P}{P+N} \right) - \frac{N}{P+N} \log_2 \left(\frac{N}{P+N} \right) \text{-----(equation 1)}$$

$$I(P,N) = I(9,5) = \left(-\frac{9}{9+5} \log_2 \left(\frac{9}{9+5} \right) - \left(\frac{5}{9+5} \log_2 \left(\frac{5}{9+5} \right) \right) \right) \\ = 0.940$$

2. Index:1

Income	Pi	Ni	I(Pi,Ni)
Low	3	1	0.8111
Med	4	2	0.9234
High	2	2	1

$$E(\text{Income}) = \frac{4}{14} I(3,1) + \frac{6}{14} I(4,2) + \frac{4}{14} I(2,2) \text{-----(eq.2)}$$

$$I(3,1) = 0.8111 \text{ (Using equation 1)}$$

$$I(4,2) = 0.9234 \text{ (Using equation 1)}$$

$$I(2,2) = 1$$

Contd.....

Information gain calculation for Index 1 contd:

Substituting the values in eq.2 we get,

$$E(\text{Income}) = 0.2317 + 0.3957 + 0.2857 = 0.9131$$

$$\begin{aligned} \text{Gain}(\text{Income}) &= I(P,N) - E(\text{Income}) \\ &= 0.940 - 0.9131 = 0.027 \end{aligned}$$

2. Index 2

Credit	Pi	Ni	I(Pi,Ni)
Fair	2	1	0.913
Exc	2	1	0.913

$$I(P,N) = I(4,2) = 0.9234 \text{ (Using equation 1)}$$

$$E(\text{Credit}) = 3/6 I(2,1) + 3/6 I(2,1) \text{ -----(3)}$$

$$I(2,1) = 0.913 \text{ (Using equation 1)}$$

$$E(\text{Credit}) = 0.913 \text{ (Substituting value of } I(2,1) \text{ in (3))}$$

$$\begin{aligned} \text{Gain}(\text{Credit}) &= I(P,N) - E(\text{Credit}) = 0.9234 - 0.913 \\ &= 0.01 \end{aligned}$$

Similarly we can calculate Information gain of tables at each stage.