

EXCELR

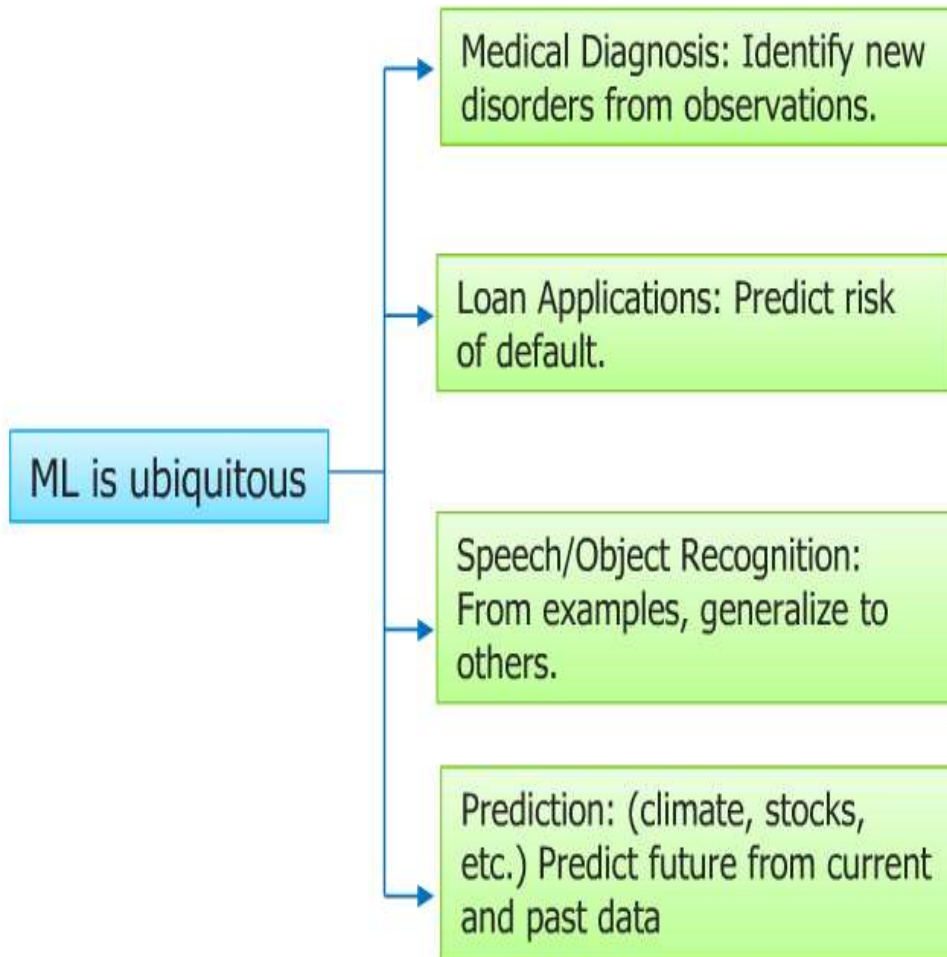
Raising Excellence

1 – Nearest Neighbour Classifier

Data Mining - Supervised

Decision Tree & Random Forest

Machine Learning is ubiquitous



Q. What is learning?

- More than just memorizing facts.
- Learning the underlying structure of the problem or data.

A fundamental aspect of learning is generalization:

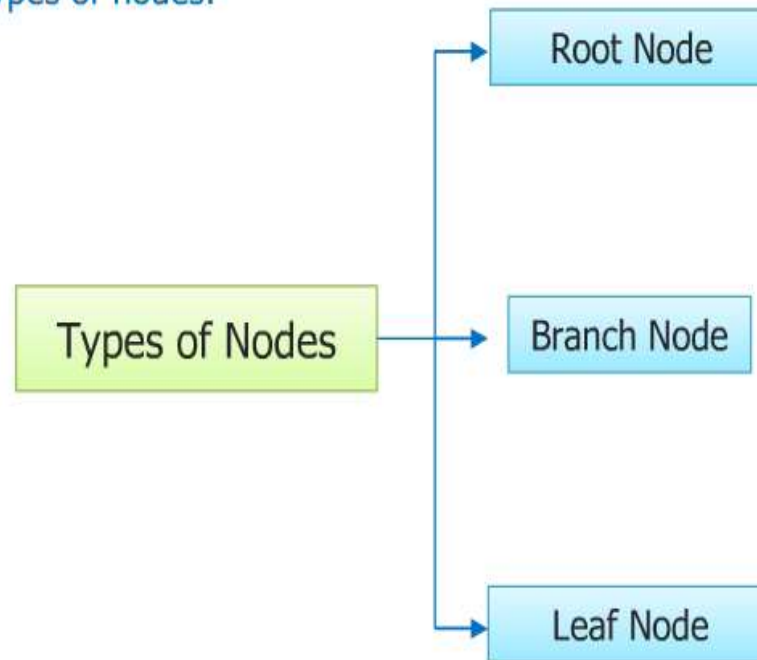
- Given a few examples, can you generalize to others?

What are Decision trees?

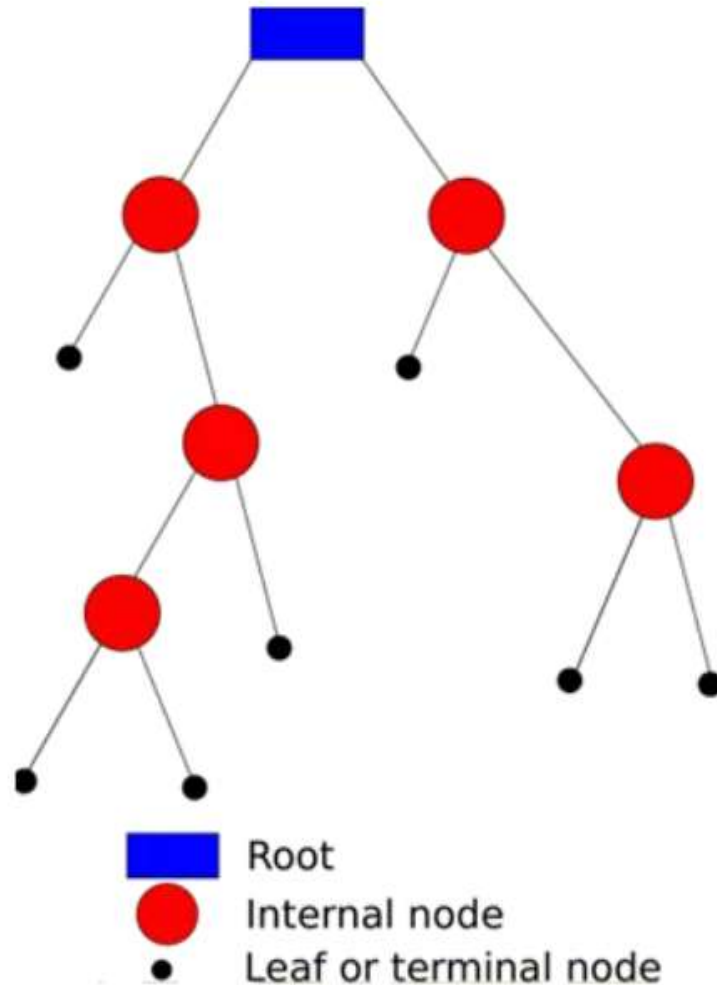
A decision tree is a tree-like structure in which internal node represents test on an attribute, each branch represents outcome of test and each leaf node represents class label (decision taken after computing all attributes).

A path from root to leaf represents classification rules.

A decision tree consists of 3 types of nodes:



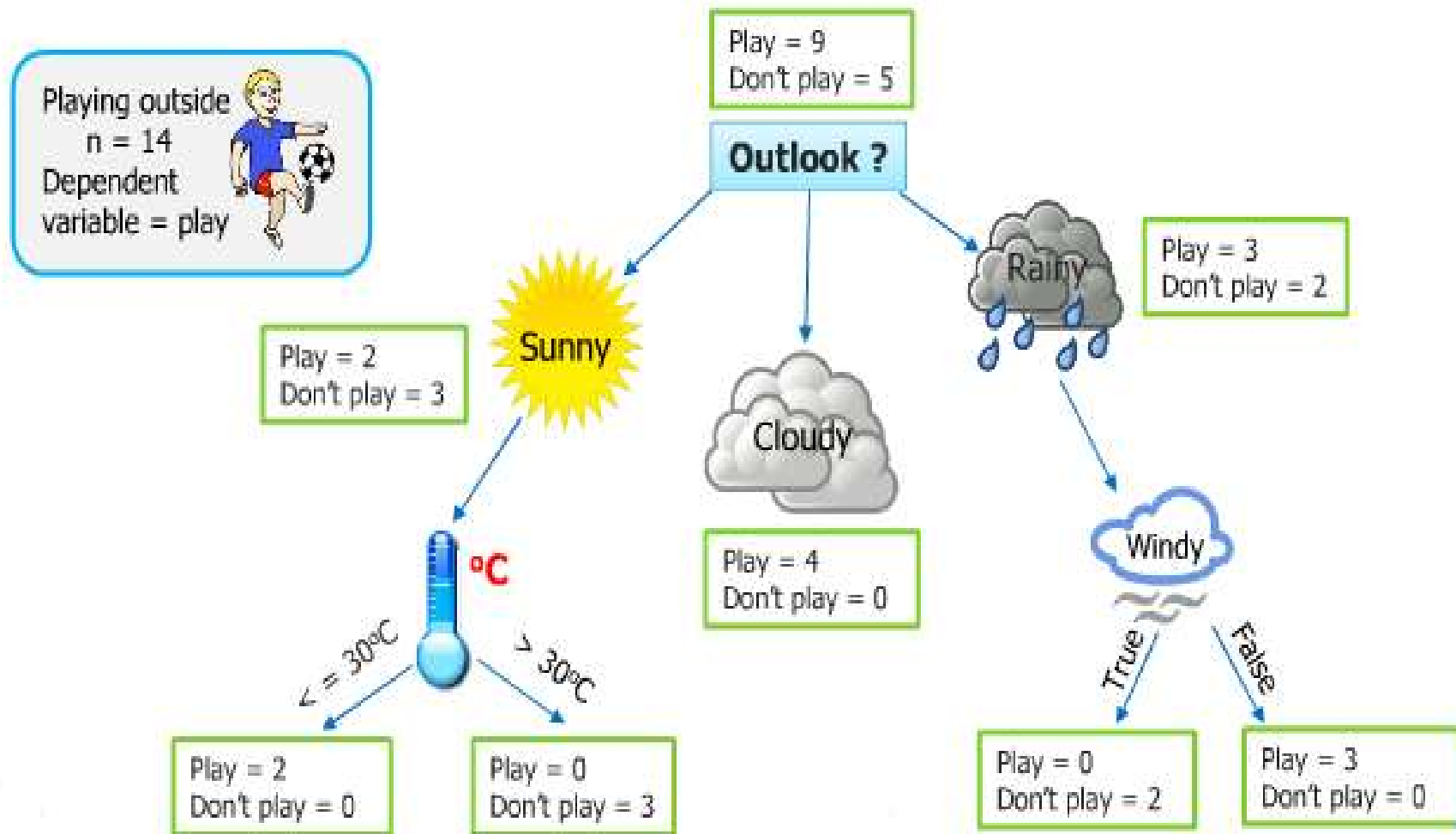
How to build decision trees?



Uses training data to build model.
Tree generator determines:

- Which variable to split at a node and what will be the value of the split.
- Decision to stop (make a terminal node) or split again has to be made.
- Assign terminal nodes to a label.

Decision trees - Example



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)
- Input data is partitioned recursively based on selected attributes
- Test attributes at each node 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

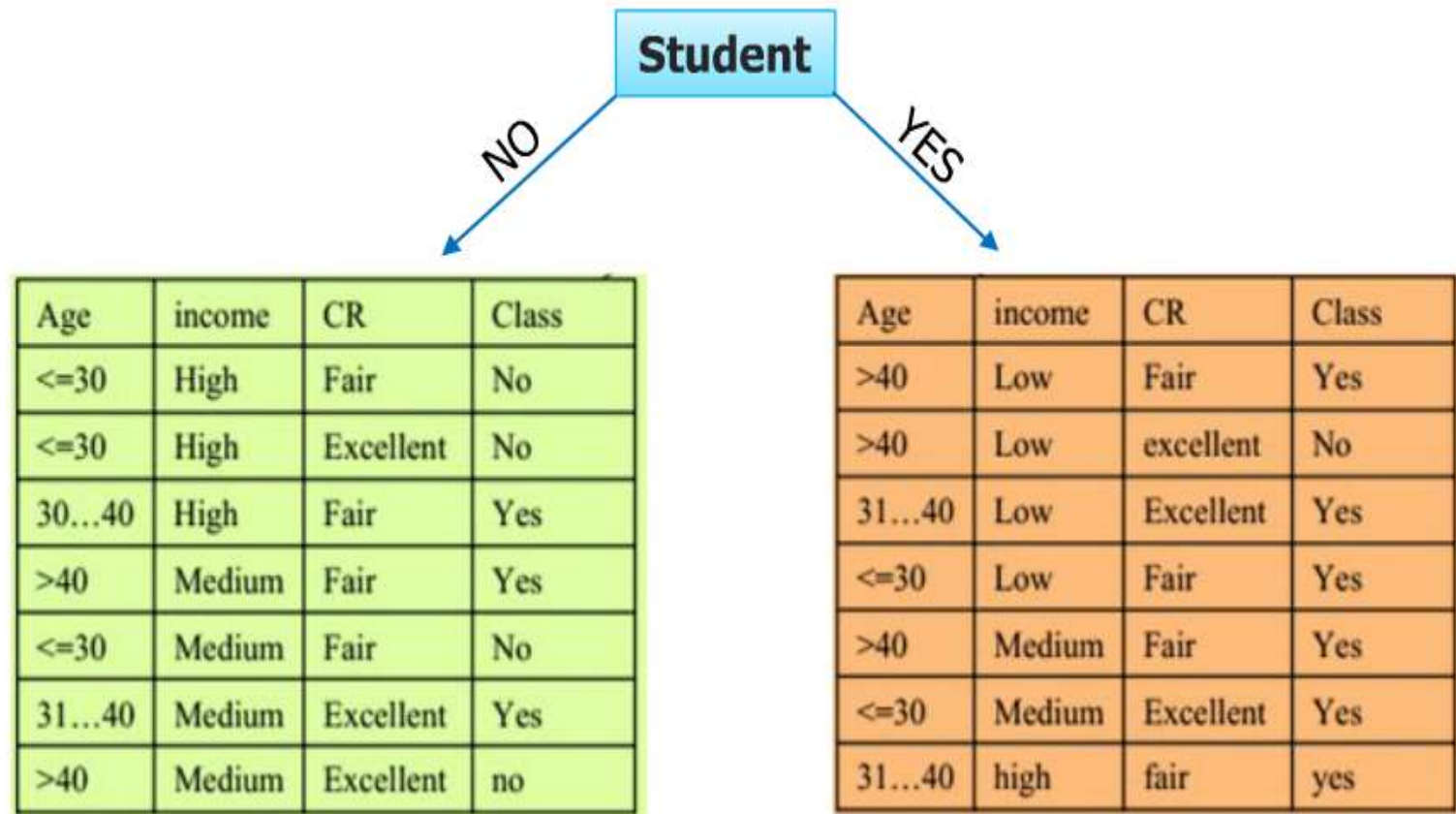
Decision Tree Examples

Training
Data

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

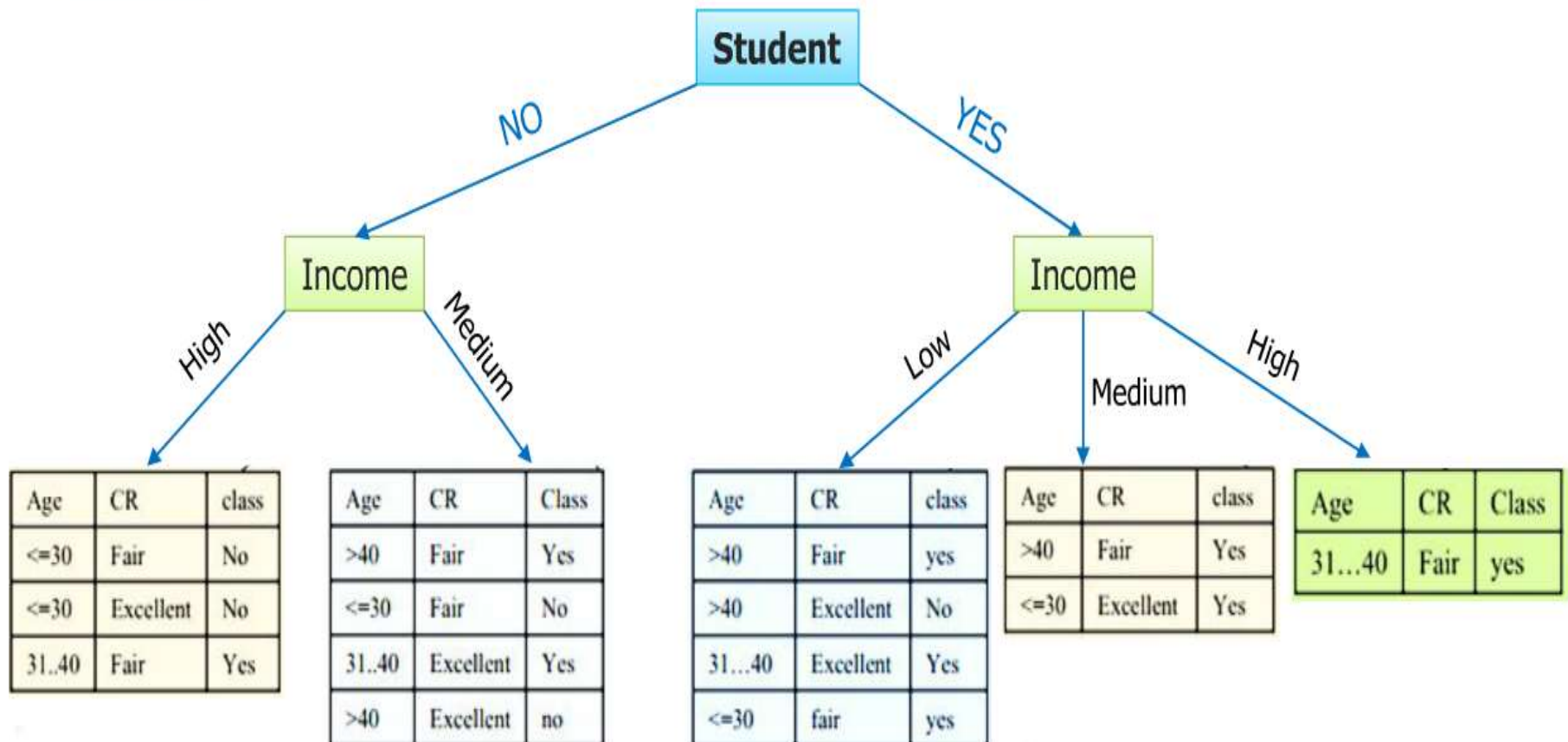
Decision Tree 1, Root: Students

Step-1



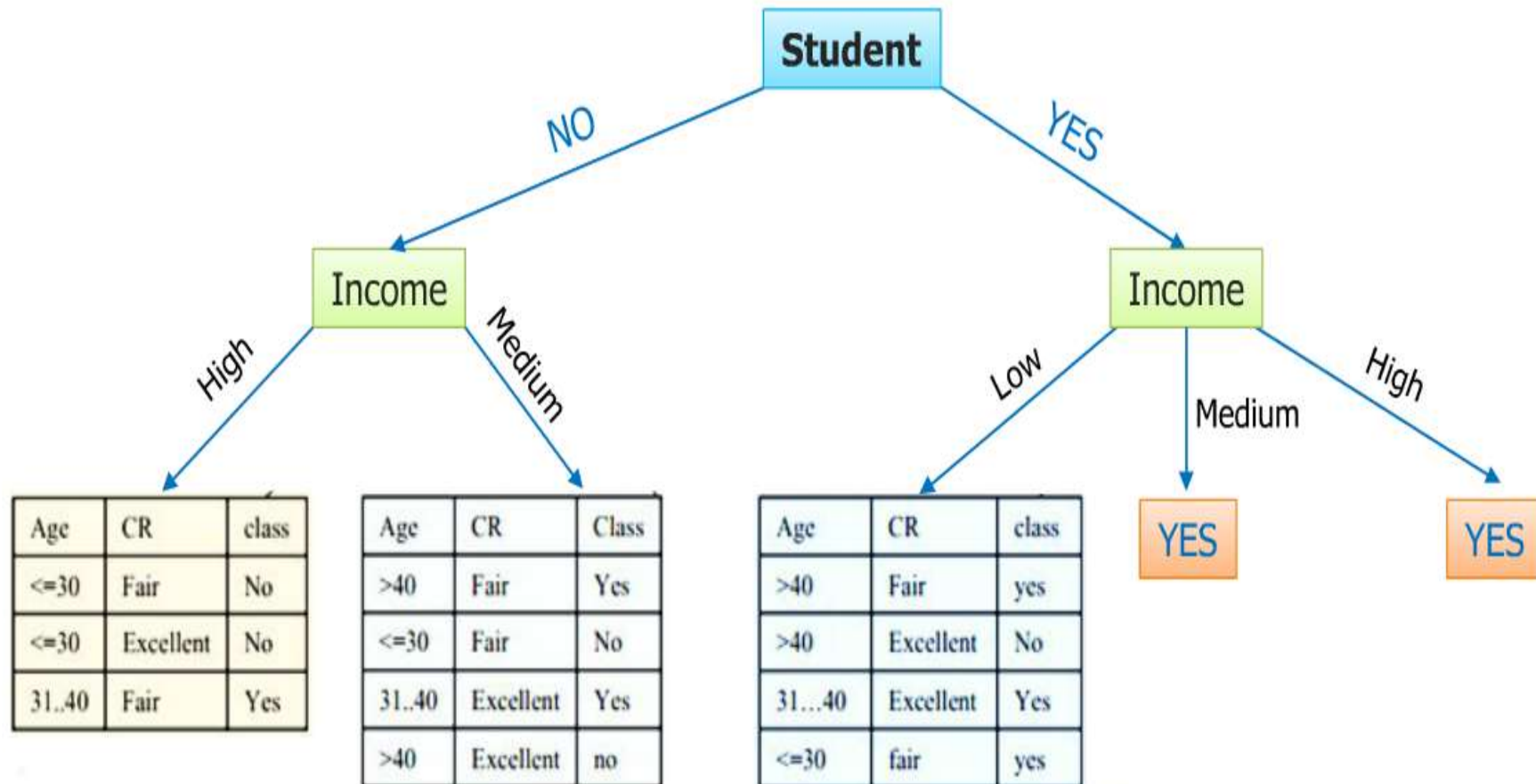
Decision Tree 1, Root: Students

Step-2



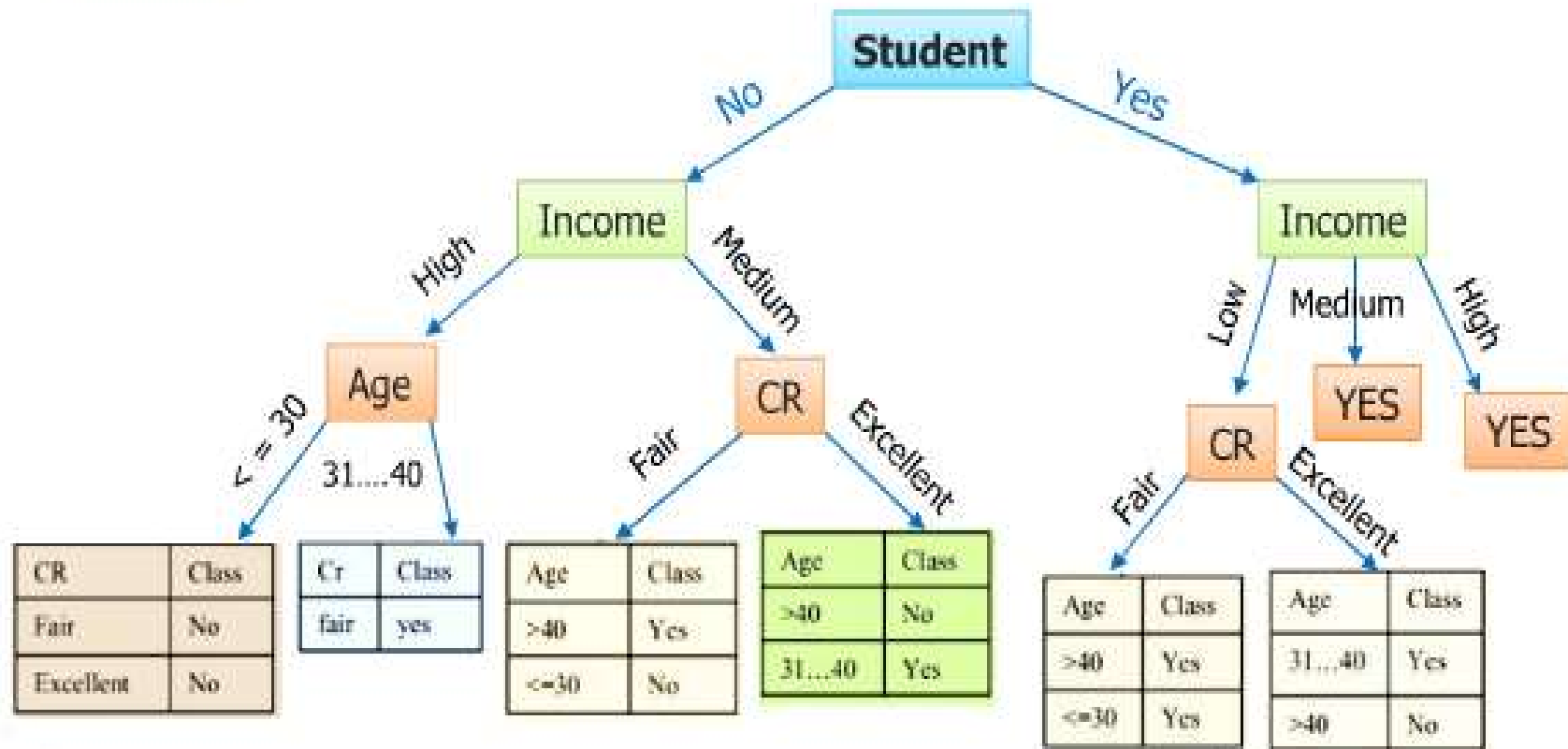
Decision Tree 1, Root: Students

Step-3



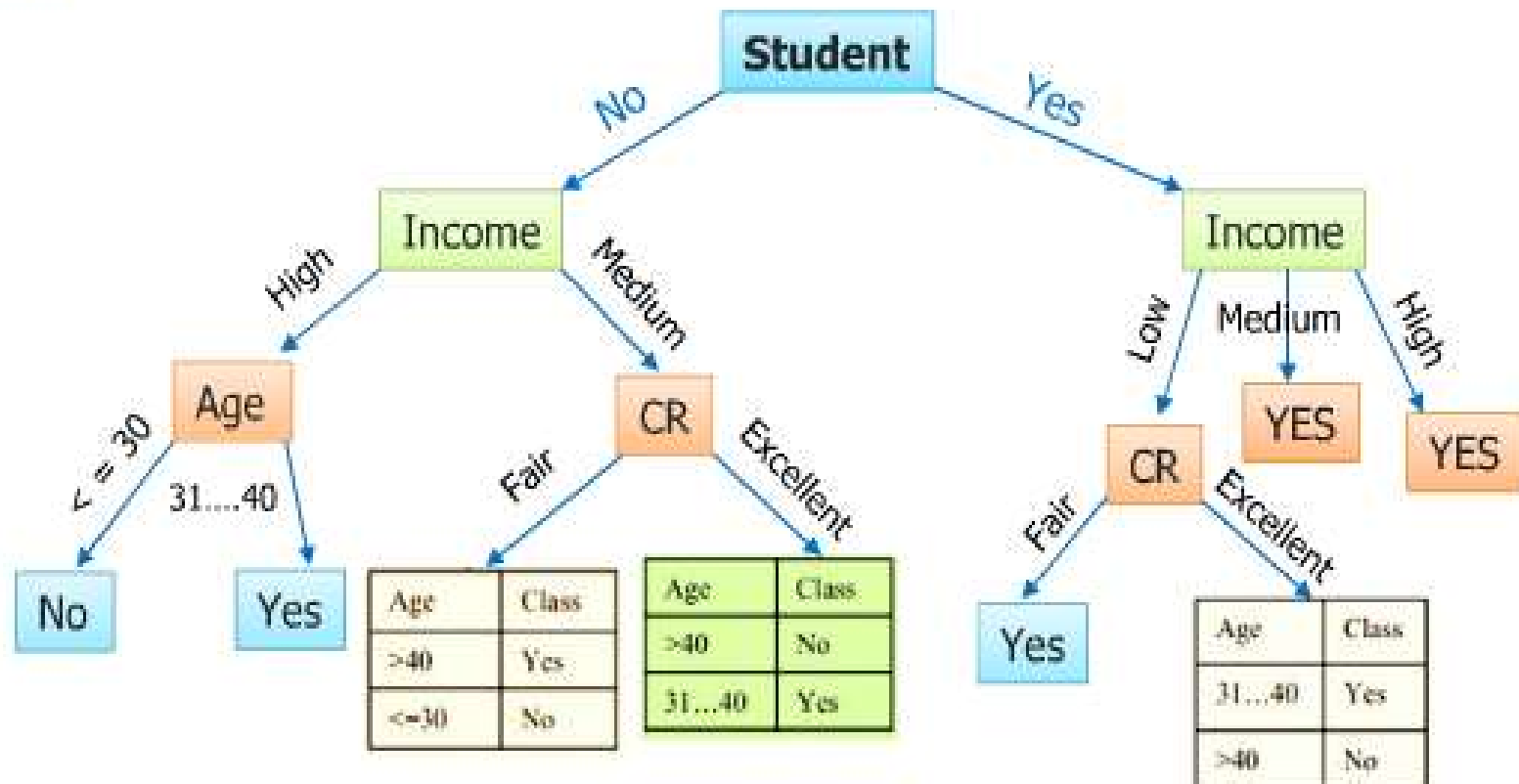
Decision Tree 1, Root: Students

Step-4



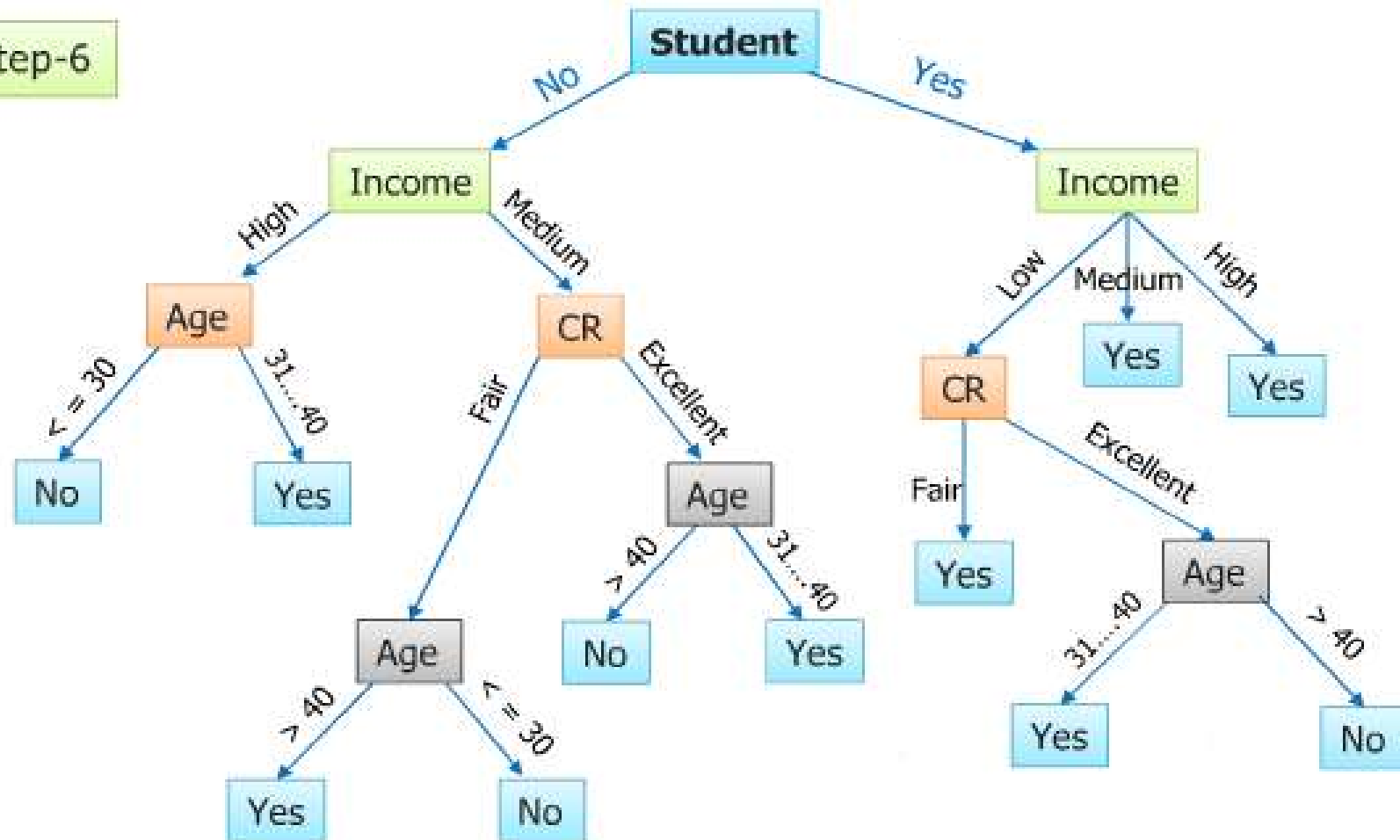
Decision Tree 1, Root: Students

Step-5



Decision Tree 1, Root: Students

Step-6



Decision Tree 1, Root: Student

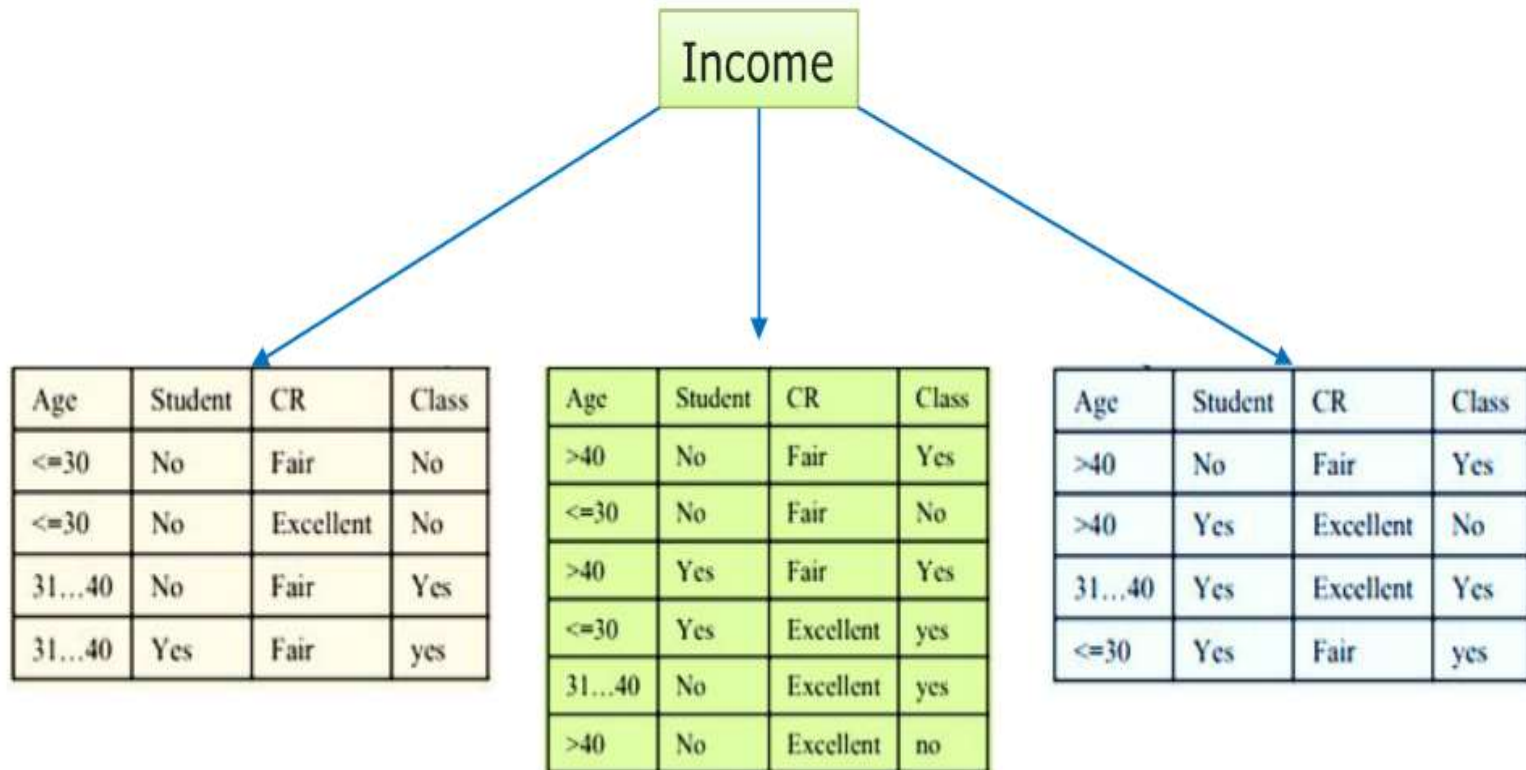
Classification Rules:

- 1. $\text{student}(\text{no}) \wedge \text{income}(\text{high}) \wedge \text{age}(\leq 30) \Rightarrow \text{buys_computer}(\text{no})$
- 2. $\text{student}(\text{no}) \wedge \text{income}(\text{high}) \wedge \text{age}(31 \dots 40) \Rightarrow \text{buys_computer}(\text{yes})$
- 3. $\text{student}(\text{no}) \wedge \text{income}(\text{medium}) \wedge \text{CR}(\text{fair}) \wedge \text{age}(> 40) \Rightarrow \text{buys_computer}(\text{yes})$
- 4. $\text{student}(\text{no}) \wedge \text{income}(\text{medium}) \wedge \text{CR}(\text{fair}) \wedge \text{age}(\leq 30) \Rightarrow \text{buys_computer}(\text{no})$
- 5. $\text{student}(\text{no}) \wedge \text{income}(\text{medium}) \wedge \text{CR}(\text{excellent}) \wedge \text{age}(> 40) \Rightarrow \text{buys_computer}(\text{no})$
- 6. $\text{student}(\text{no}) \wedge \text{income}(\text{medium}) \wedge \text{CR}(\text{excellent}) \wedge \text{age}(31 \dots 40) \Rightarrow \text{buys_computer}(\text{yes})$
- 7. $\text{student}(\text{yes}) \wedge \text{income}(\text{low}) \wedge \text{CR}(\text{fair}) \Rightarrow \text{buys_computer}(\text{yes})$
- 8. $\text{student}(\text{yes}) \wedge \text{income}(\text{low}) \wedge \text{CR}(\text{excellent}) \wedge \text{age}(31 \dots 40) \Rightarrow \text{buys_computer}(\text{yes})$
- 9. $\text{student}(\text{yes}) \wedge \text{income}(\text{low}) \wedge \text{CR}(\text{excellent}) \wedge \text{age}(> 40) \Rightarrow \text{buys_computer}(\text{no})$
- 10. $\text{student}(\text{yes}) \wedge \text{income}(\text{medium}) \Rightarrow \text{buys_computer}(\text{yes})$
- 11. $\text{student}(\text{yes}) \wedge \text{income}(\text{high}) \Rightarrow \text{buys_computer}(\text{yes})$

Decision Tree 2, Root: Income

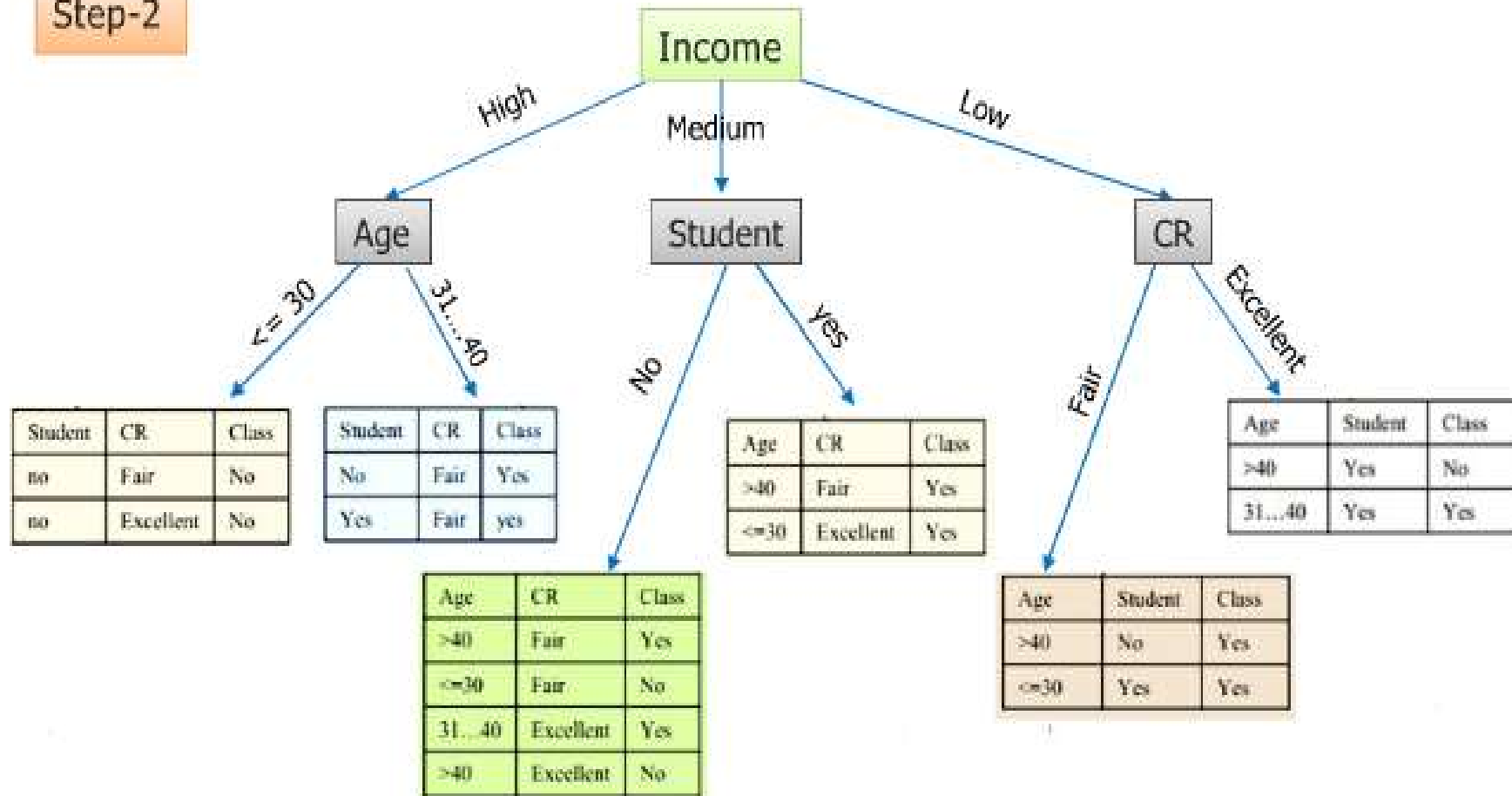
Step-1

Now consider another approach to build the decision tree.
Let's take the attribute: Income



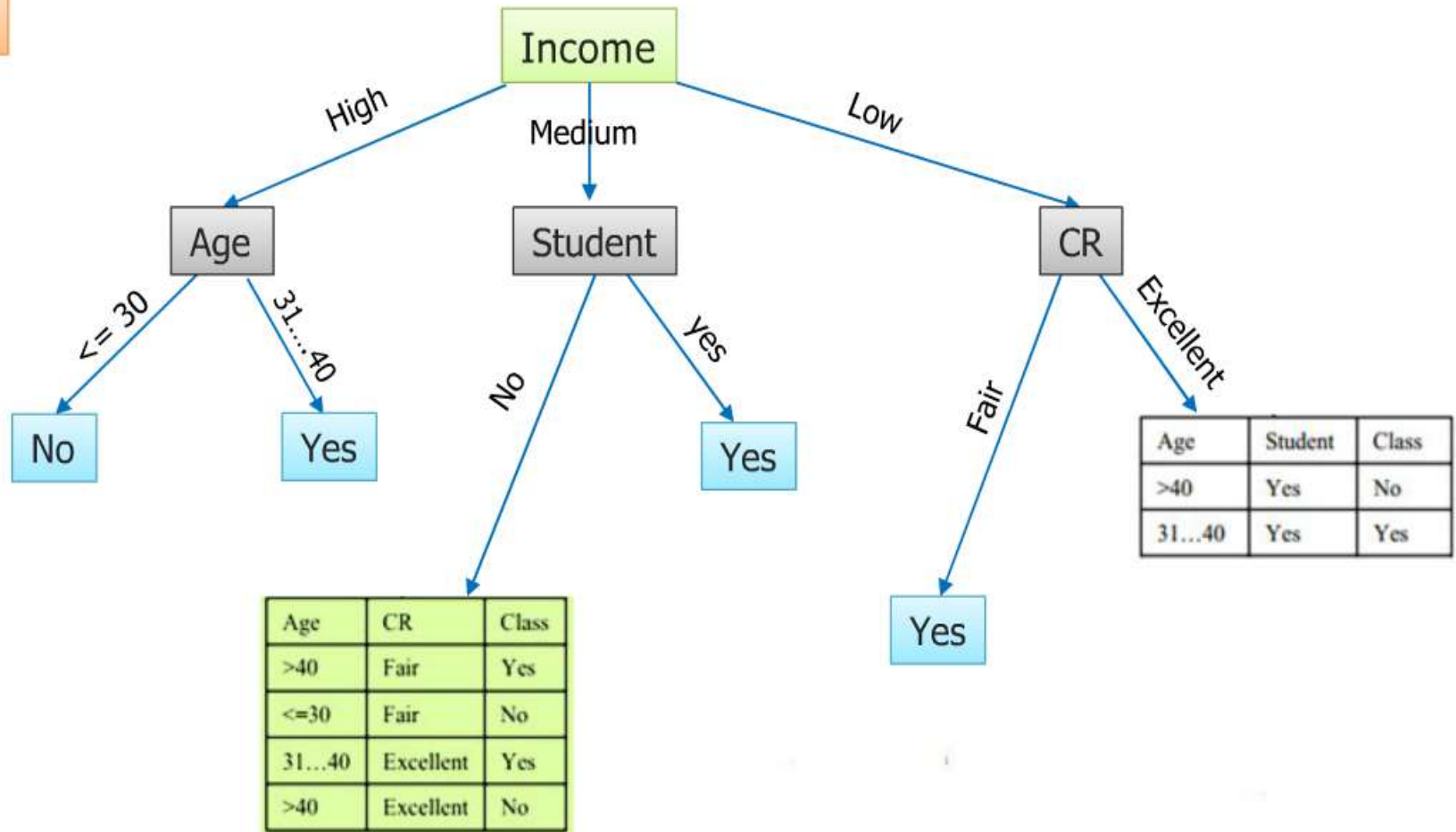
Decision Tree 2, Root: Income

Step-2



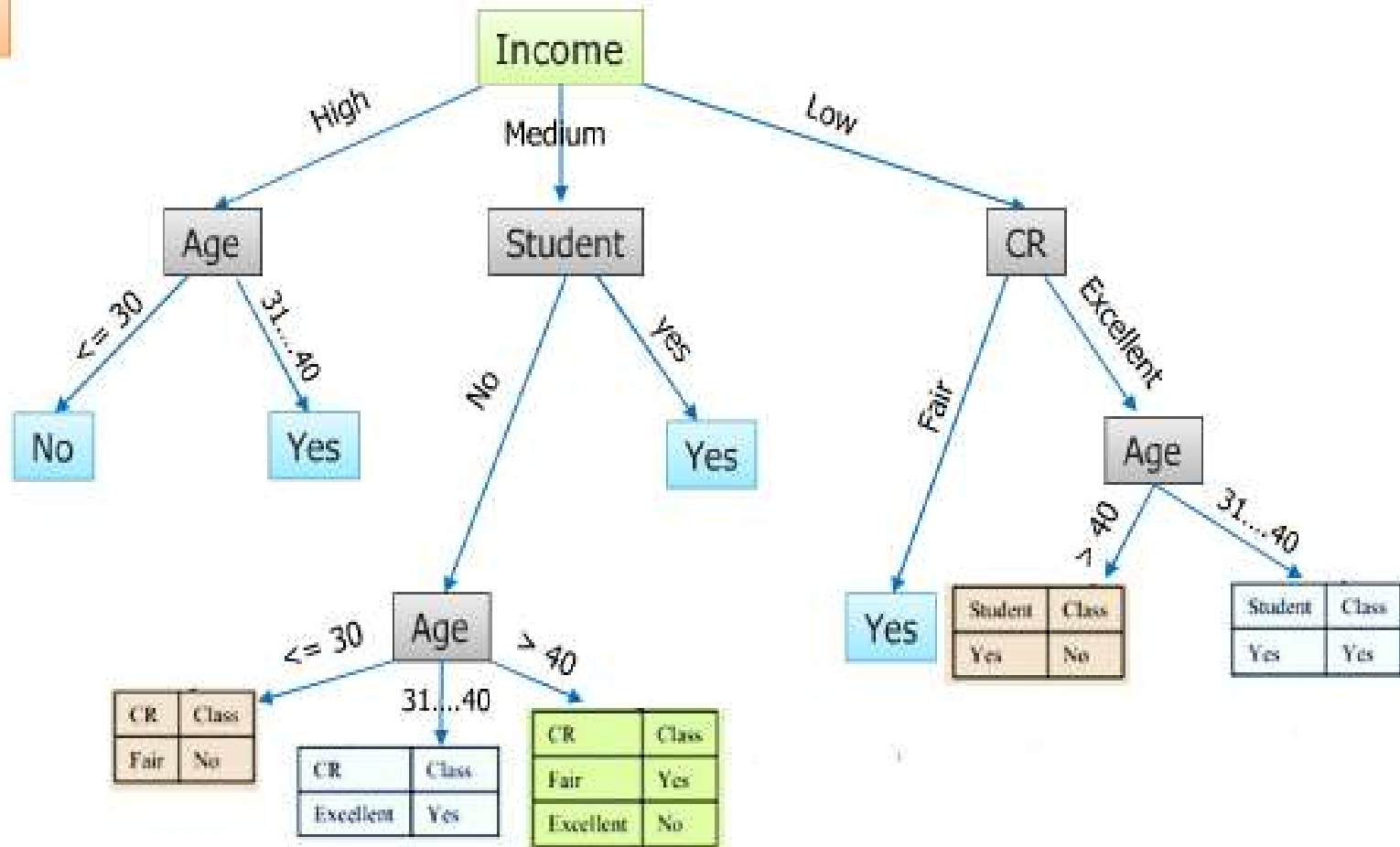
Decision Tree 2, Root: Income

Step-3



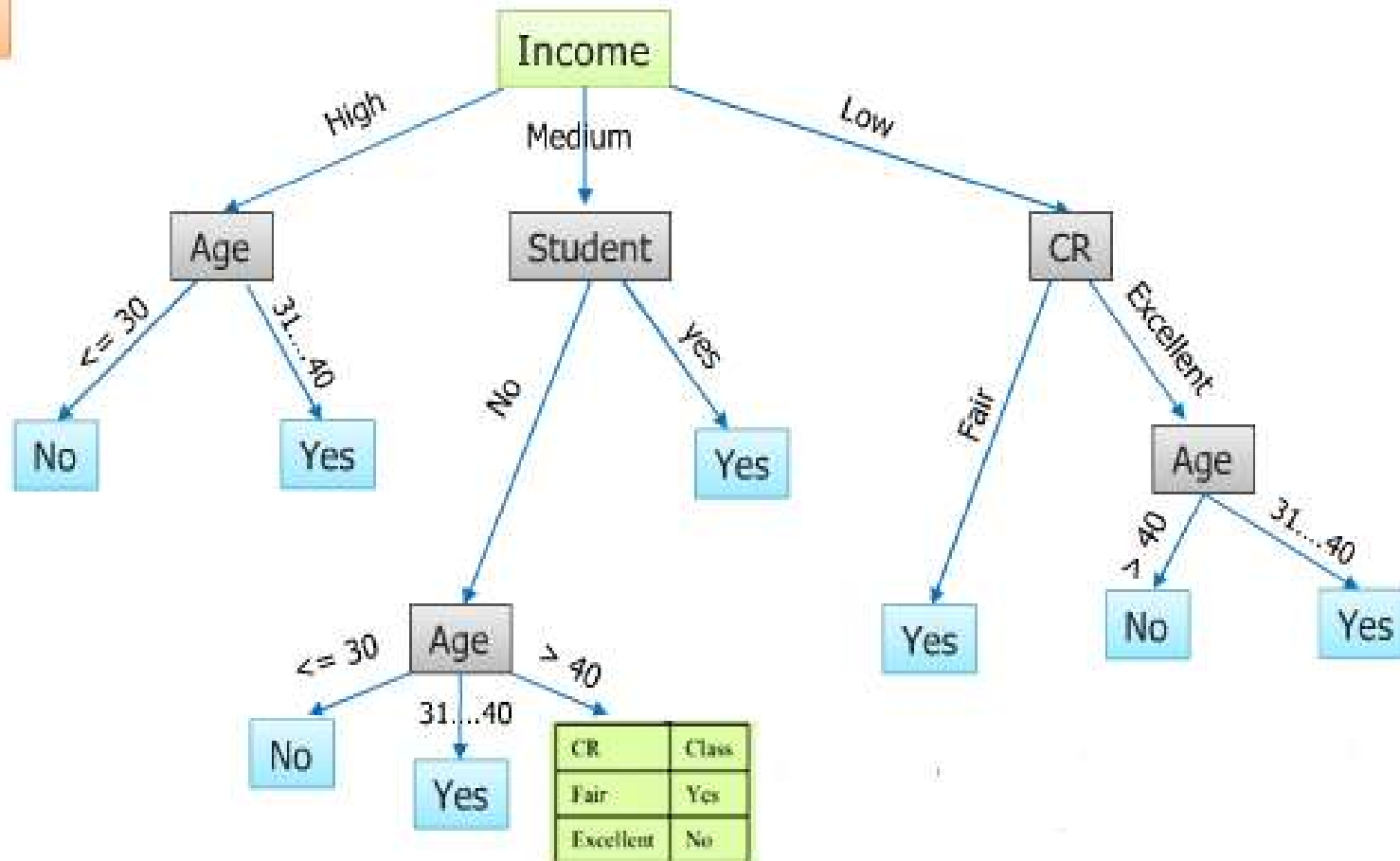
Decision Tree 2, Root: Income

Step-4



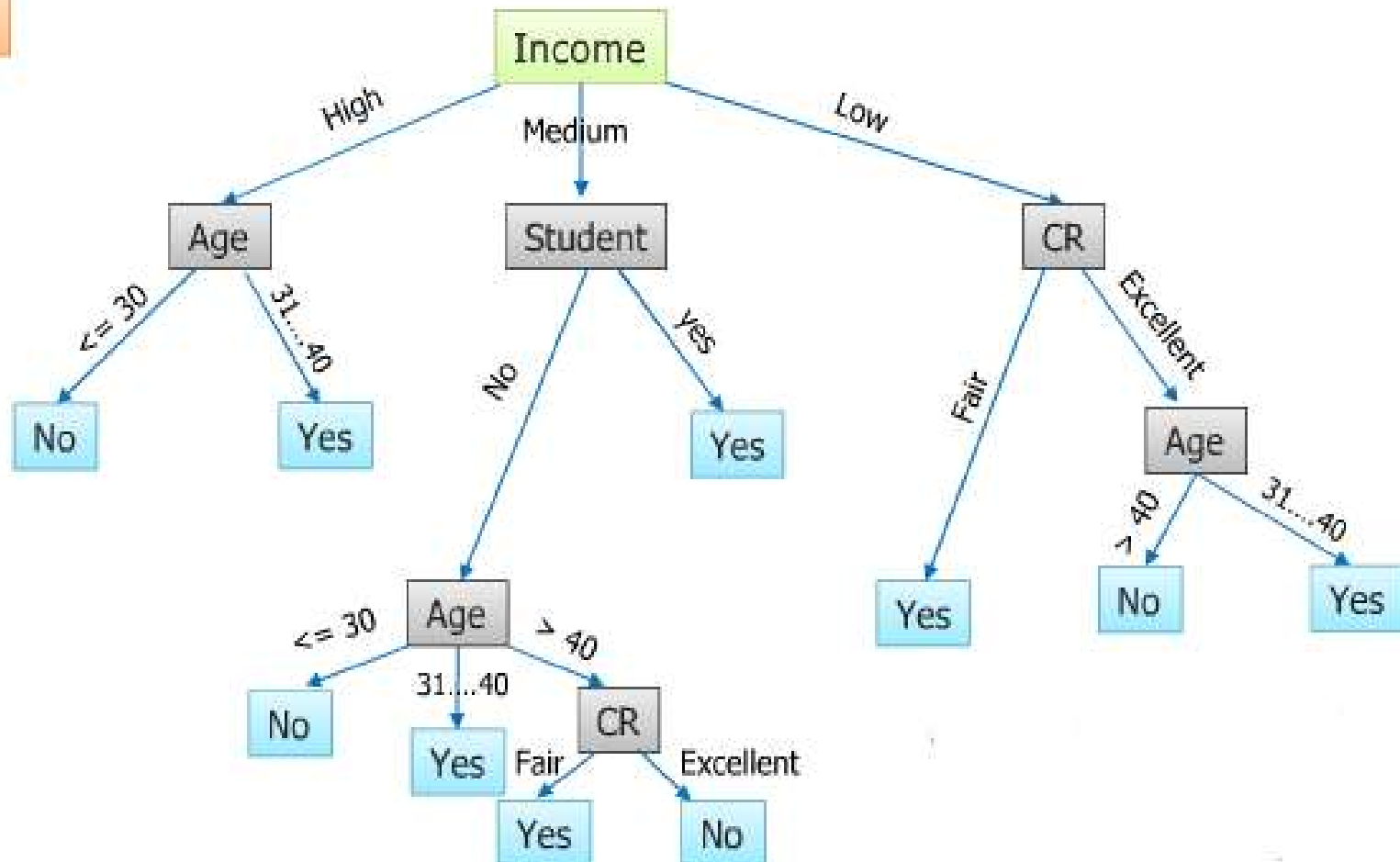
Decision Tree 2, Root: Income

Step-5



Decision Tree 2, Root: Income

Step-6



Decision Tree 2, Root: Income

Classification Rules:

- 1. $\text{income}(\text{high}) \wedge \text{age}(\leq 30) \Rightarrow \text{buys_computer}(\text{no})$
- 2. $\text{income}(\text{high}) \wedge \text{age}(31 \dots 40) \Rightarrow \text{buys_computer}(\text{yes})$
- 3. $\text{income}(\text{medium}) \wedge \text{student}(\text{no}) \wedge \text{age}(\leq 30) \Rightarrow \text{buys_computer}(\text{no})$
- 4. $\text{income}(\text{medium}) \wedge \text{student}(\text{no}) \wedge \text{age}(31 \dots 40) \Rightarrow \text{buys_computer}(\text{yes})$
- 5. $\text{income}(\text{medium}) \wedge \text{student}(\text{no}) \wedge \text{age}(> 40) \wedge \text{CR}(\text{fair}) \Rightarrow \text{buys_computer}(\text{yes})$
- 6. $\text{income}(\text{medium}) \wedge \text{student}(\text{no}) \wedge \text{age}(> 40) \wedge \text{CR}(\text{excellent}) \Rightarrow \text{buys_computer}(\text{no})$
- 7. $\text{income}(\text{medium}) \wedge \text{student}(\text{yes}) \Rightarrow \text{buys_computer}(\text{yes})$
- 8. $\text{income}(\text{medium}) \wedge \text{CR}(\text{fair}) \Rightarrow \text{buys_computer}(\text{yes})$
- 9. $\text{income}(\text{medium}) \wedge \text{CR}(\text{excellent}) \wedge \text{age}(> 40) \Rightarrow \text{buys_computer}(\text{no})$
- 10. $\text{income}(\text{medium}) \wedge \text{CR}(\text{excellent}) \wedge \text{age}(31 \dots 40) \Rightarrow \text{buys_computer}(\text{yes})$

Greedy Approach and Entropy

We want to use the attribute that does the "best job" splitting up the training data, but can this be measured?

We use entropy and information gain

Entropy:

- Measure of disorder or impurity
- We will find entropy of the output values of a set of training instances
- If output values split 50-50% set is impure – 1
- If output is 0, set is Pure – 0
- If the output values are split 25-75% then entropy – 0.811

Information Gain:

The information gain is based on the decrease in entropy after a dataset is split on an attribute. Constructing a decision tree is all about finding attribute that returns the highest information gain (i.e., the most homogeneous branches).

Entropy(Information Theory)

Measure of uncertainty(unpredictability) of random variable
Measure of information content

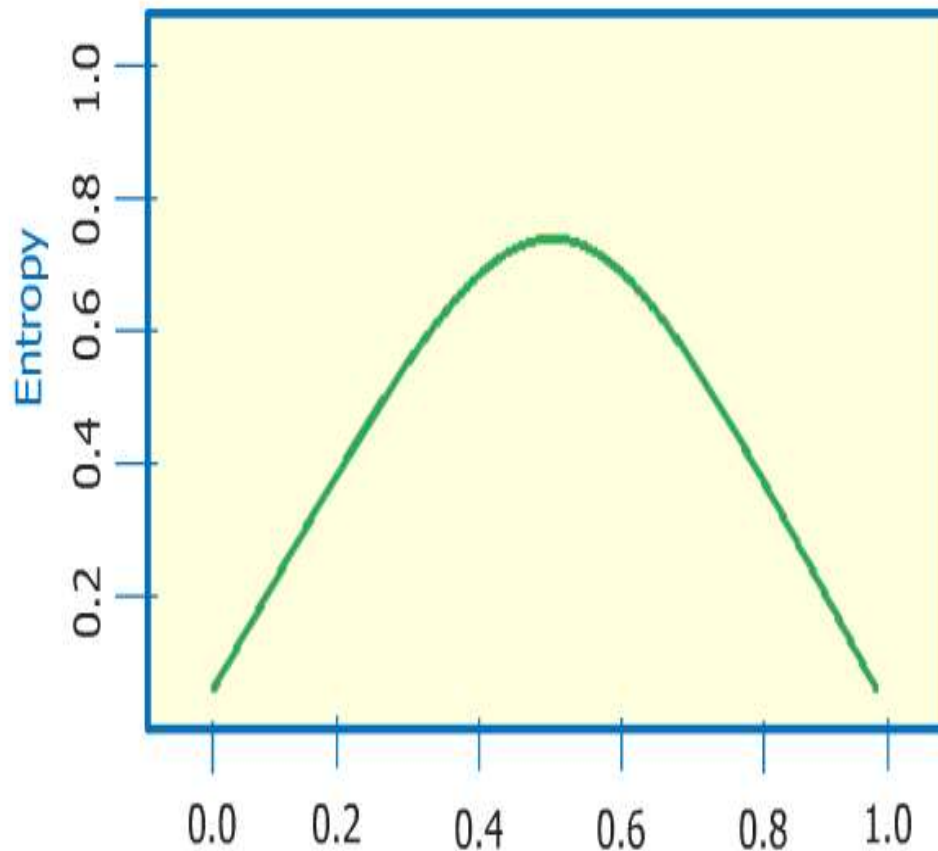
Highly unpredictable = High information content = Large entropy
Less unpredictable = Low information content = Small entropy

Discrete random variable X taking m distinct values $\{x_1, \dots, x_m\}$

Mathematical Formula:

$$H(X) = - \sum_{i=1}^m p_i \cdot \log_2(p_i), p_i = P(X = x_i)$$

Entropy(Information Theory)



Example:

X = Winning party in 2014 elections

x_1 = Congress, x_2 = BJP

$p_1=0.5, p_2=0.5 \rightarrow H(X) = 1$

$p_1=0.2, p_2=0.8 \rightarrow H(X) = 0.515$

$p_1=0.9, p_2=0.1 \rightarrow H(X) = 0.468$

In previous example

$P(\text{Buys}=\text{YES}) = 9/14, P(\text{Buys}=\text{No}) = 5/14$

Information Gain: Attribute Selection Measure

Heuristic: **Select the attribute with the highest information gain** i.e., attribute that results in most homogeneous branches

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 :

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

→ **Information** needed (after using A to split D into v partitions) to classify D :

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

→ **Information gained** by branching on attribute A

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

Creating a Perfect Decision Tree

Decision Tree Examples

Training
Data

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

Attribute Selection Example

→ Two classes

» Positive: buys_computer=yes $P(buys = yes) = \frac{9}{14}$

» Negative: buys_computer=no $P(buys = no) = \frac{5}{14}$

→ Entropy in D

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

→ What is information gain if we split on "Age"?

age	pos	neg	I (pos, neg)
<=30	2	3	0.971
31...40	4	0	0
>40	3	2	0.971

$$Info(Age \leq 30) = 0.971$$

$$Info(31 \leq Age \leq 40) = 0$$

$$Info(Age > 40) = 0.971$$

$$Info_{age}(D)$$

$$= \frac{5}{14} Info(Age \leq 30) + \frac{4}{14} Info(Age = 31..40) + \frac{5}{14} Info(Age > 40)$$

$$= \frac{5}{14} * 0.971 + \frac{4}{14} * 0 + \frac{5}{14} * 0.971$$

$$= 0.694$$

Age	Income	Student	Credit Rating	Buys Computer
<=30	High	No	Fair	No
<=30	High	No	Excellent	No
31...40	High	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

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

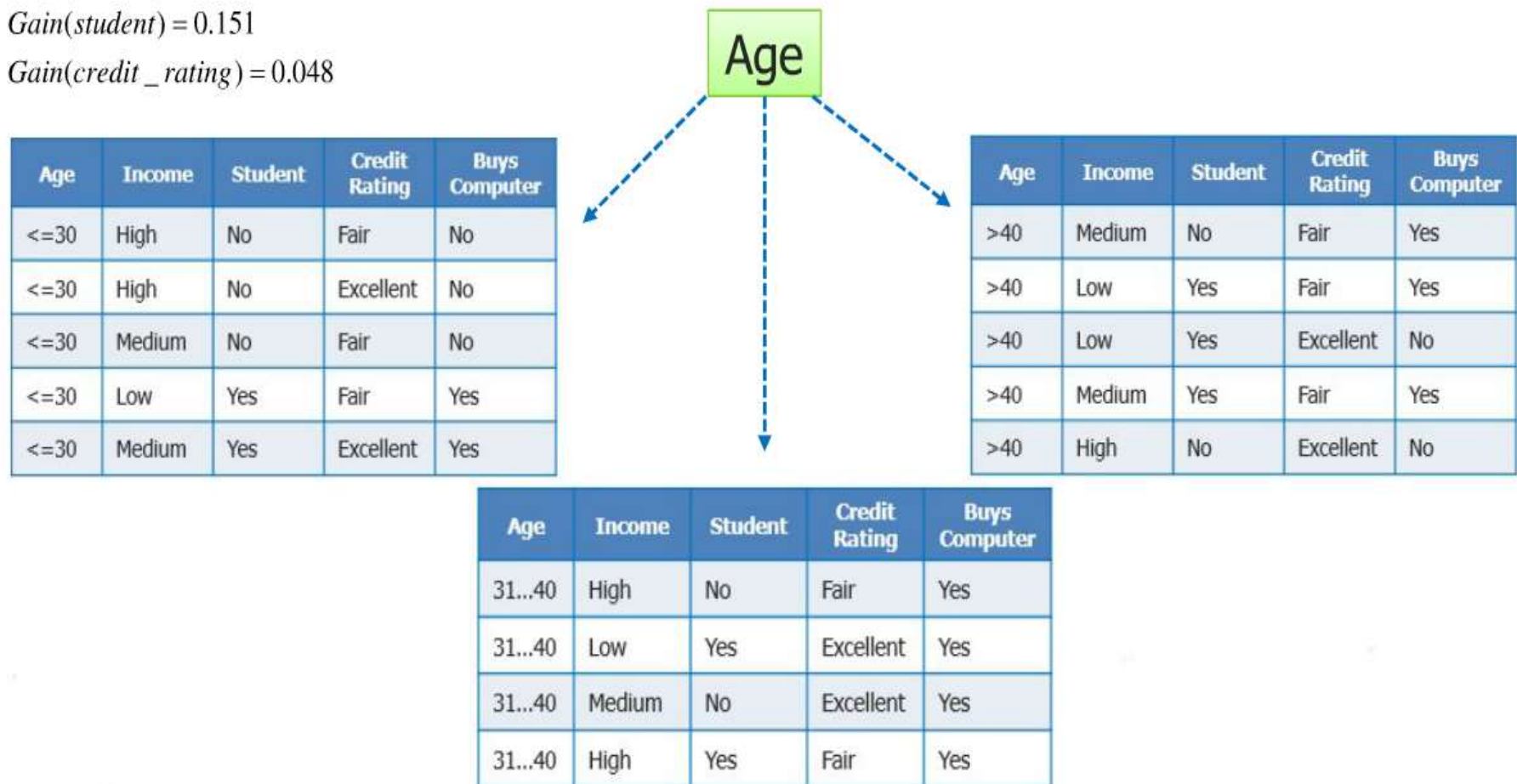
Attribute Selection Example

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

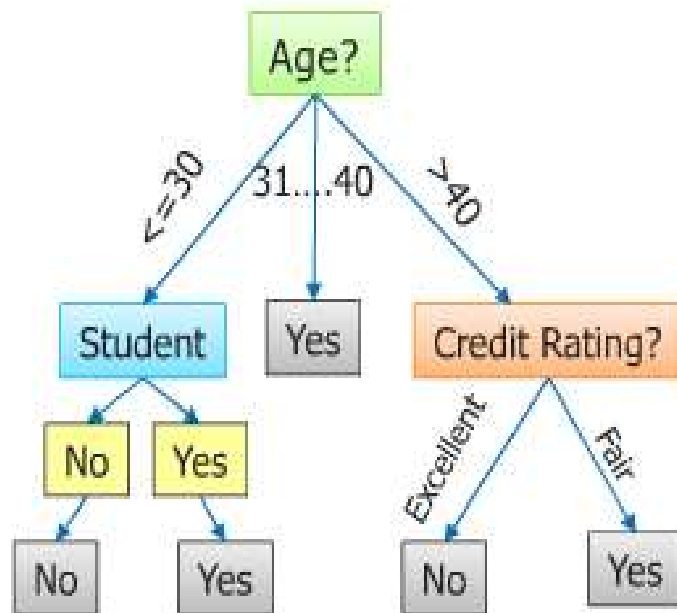
$$\text{Gain}(\text{income}) = 0.029$$

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

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

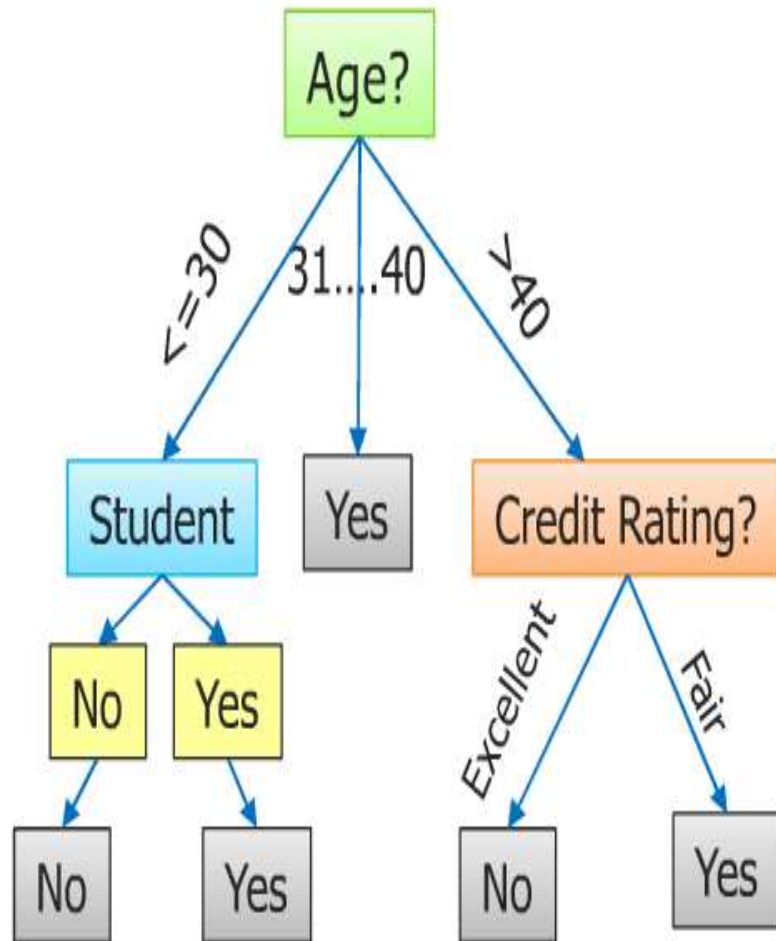


Final Tree



Age	Income	Student	Credit Rating	Buys Computer
<=30	High	No	Fair	No
<=30	High	No	Excellent	No
31...40	High	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 Tree



- $\text{Age}(<30) \wedge \text{student}(\text{no}) = \text{NO}$
- $\text{Age}(<30) \wedge \text{student}(\text{yes}) = \text{YES}$
- $\text{Age}(31\ldots40) = \text{YES}$
- $\text{Age}(>40) \wedge \text{credit_rating}(\text{excellent}) = \text{NO}$
- $\text{Age}(>40) \wedge \text{credit_rating}(\text{fair}) = \text{Yes}$

Random Forest

Random Forest: An example



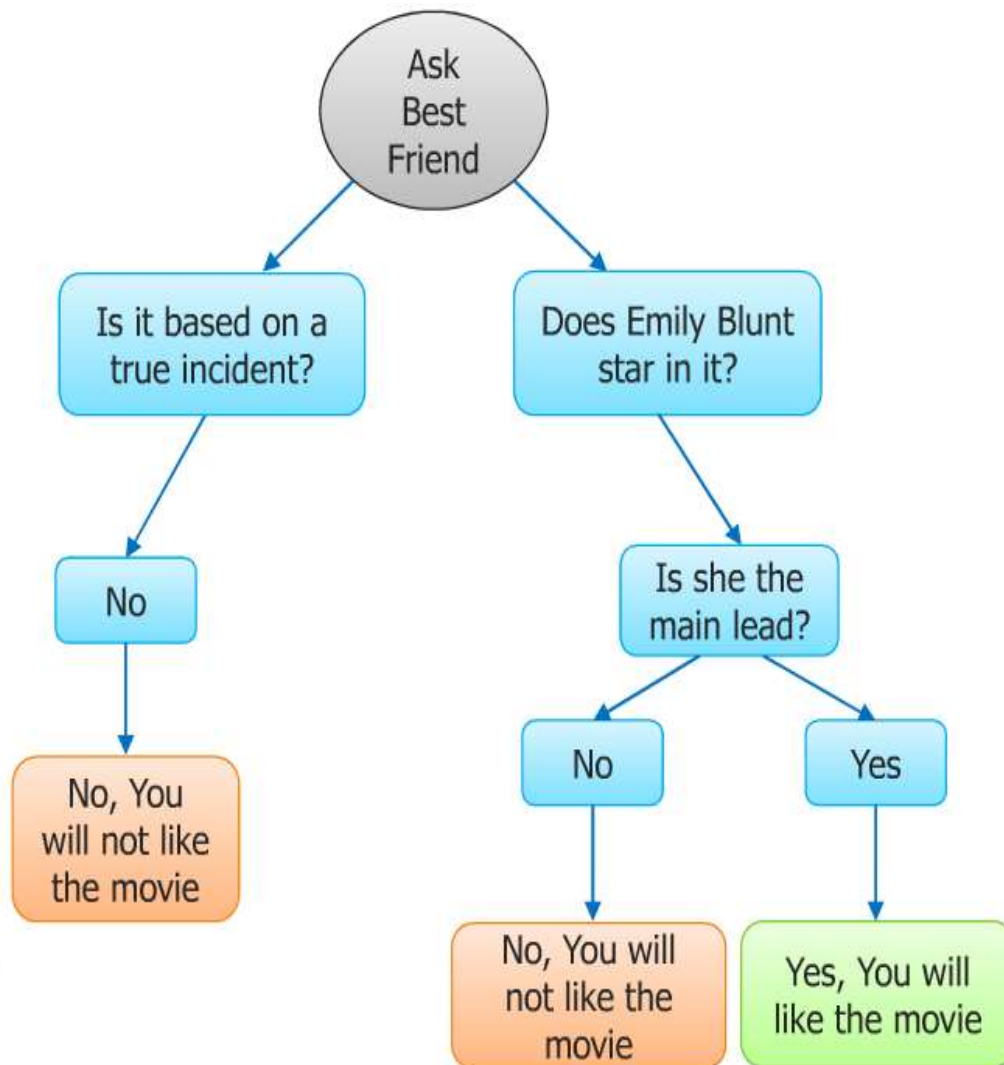
Suppose you're very indecisive about watching a movie.

"Edge of Tomorrow"

You can decide to perform either of the two operations about it:

- Either you ask your best friend, whether you will like the movie.
- Or You can ask your group of friends.

Random Forest: An example



In order to answer, your best friend first needs to figure out what movies you like, so you give her a bunch of movies and tell her whether you liked each one or not (i.e., you give her a labelled training set).

Example:

You like movies starring Emily Blunt.

Based on such labels, she finds out whether or not you will like "Edge of Tomorrow".

Thus, a decision tree is created of your best friend(attribute).

Random Forest: An example



But your best friend might not always generalize your preferences very well (i.e., she overfits).

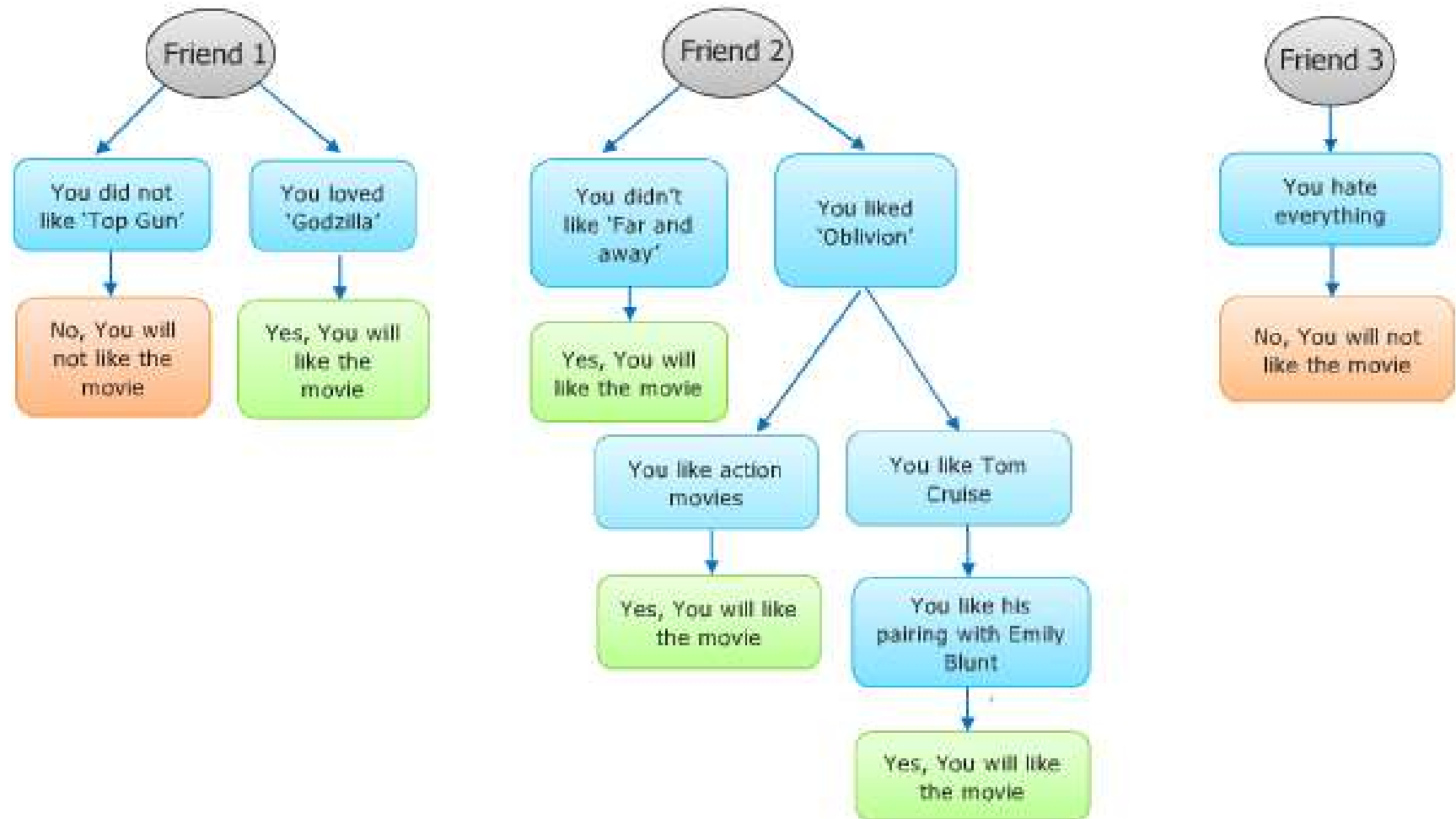
In order to get more accurate recommendations, you'd like to ask a bunch of your friends, whether you will like the movie "Edge of Tomorrow" or not.

You want to ask your other friends as well e.g. Friend#1, Friend #2, and Friend #3 and they vote on whether you will like a movie.

The majority of the votes will decide the final outcome.

Thus, you build an ensemble classifier of the decision tree (Random Forest) of the group of friends.

Random Forest: An example

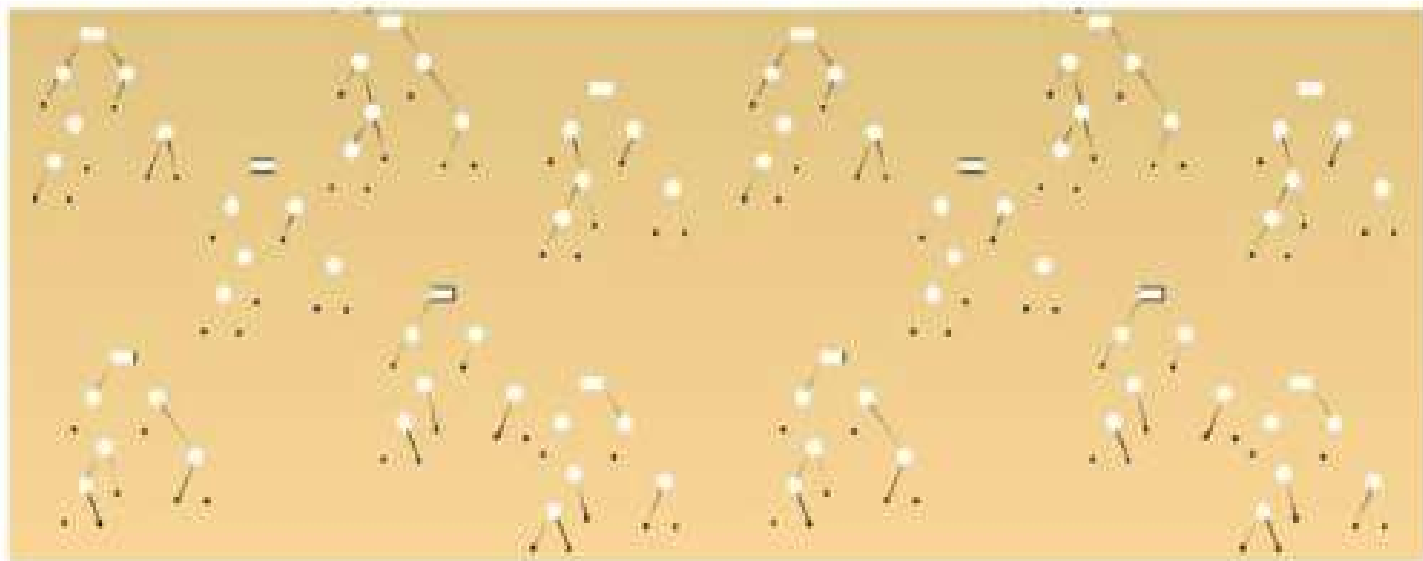


What is Random Forest?

Random Forest is an ensemble classifier made using many decision tree models.

What are ensemble models?

- Ensemble models combine the results from different models.
- The result from an ensemble model is usually better than the result from one of the individual models.



How Random Forest works?

Each tree is grown as follows:

- Let the number of training cases be N , and the number of variables in the classifier be M .
- From M input variables, a number $m \ll M$ is specified such that at each node, m variables are selected at random out of the M and the best split on these m is used to split the node. The value of m is held constant during the growth of the forest.
- Each tree is grown to the largest extent possible.
- Decision is made by the number of votes from all of the trees.

A different subset of the training data are selected ($\sim 2/3$), with replacement, to train each tree.

Remaining training data is used to estimate error and variable importance.

Features of Random Forest

- It is unexcelled in accuracy among current algorithms.
- It runs efficiently on large data bases.
- It can handle thousands of input variables without variable deletion.
- It gives estimates of what variables are important in the classification.
- It generates an internal unbiased estimate of the generalization error as the forest building progresses.
- It has an effective method for estimating missing data and maintains accuracy when a large proportion of the data is missing.
- Generated forests can be saved for future use on other data.
- Prototypes are computed, that gives information about the relation between the variables and the classification.

Features of Random Forest

- It is unexcelled in accuracy among current algorithms.
- It runs efficiently on large data bases.
- It can handle thousands of input variables without variable deletion.
- It gives estimates of what variables are important in the classification.
- It generates an internal unbiased estimate of the generalization error as the forest building progresses.
- It has an effective method for estimating missing data and maintains accuracy when a large proportion of the data is missing.
- Generated forests can be saved for future use on other data.
- Prototypes are computed, that gives information about the relation between the variables and the classification.

Applications

A manager has to decide whether he should hire more human resources or not in order to optimize the work load balance

An individual has to make a decision such as whether or not to undertake a capital project, or must chose between two competing ventures

A bank wants to classify its future customers into two categories "Risky" and "Good" based on customer's available attributes.

Let's say a customer xyz has the following attributes. How will the bank know to which category this customer belong.

Undergrad	Marital Status	Taxable Income	City Population	Work Experience (Yrs)	Urban	Category
No	Married	98,727	1,01,894	14	NO	????

Possible algorithms

Such type of problems comes under “**classification**”

It is the separation or ordering of objects into classes

★ There are few techniques in classification method, like:

- ✓ Decision Tree
- ✓ Naïve Bayes
- ✓ k-Nearest Neighbor
- ✓ Support Vector Machine etc..

Thank you