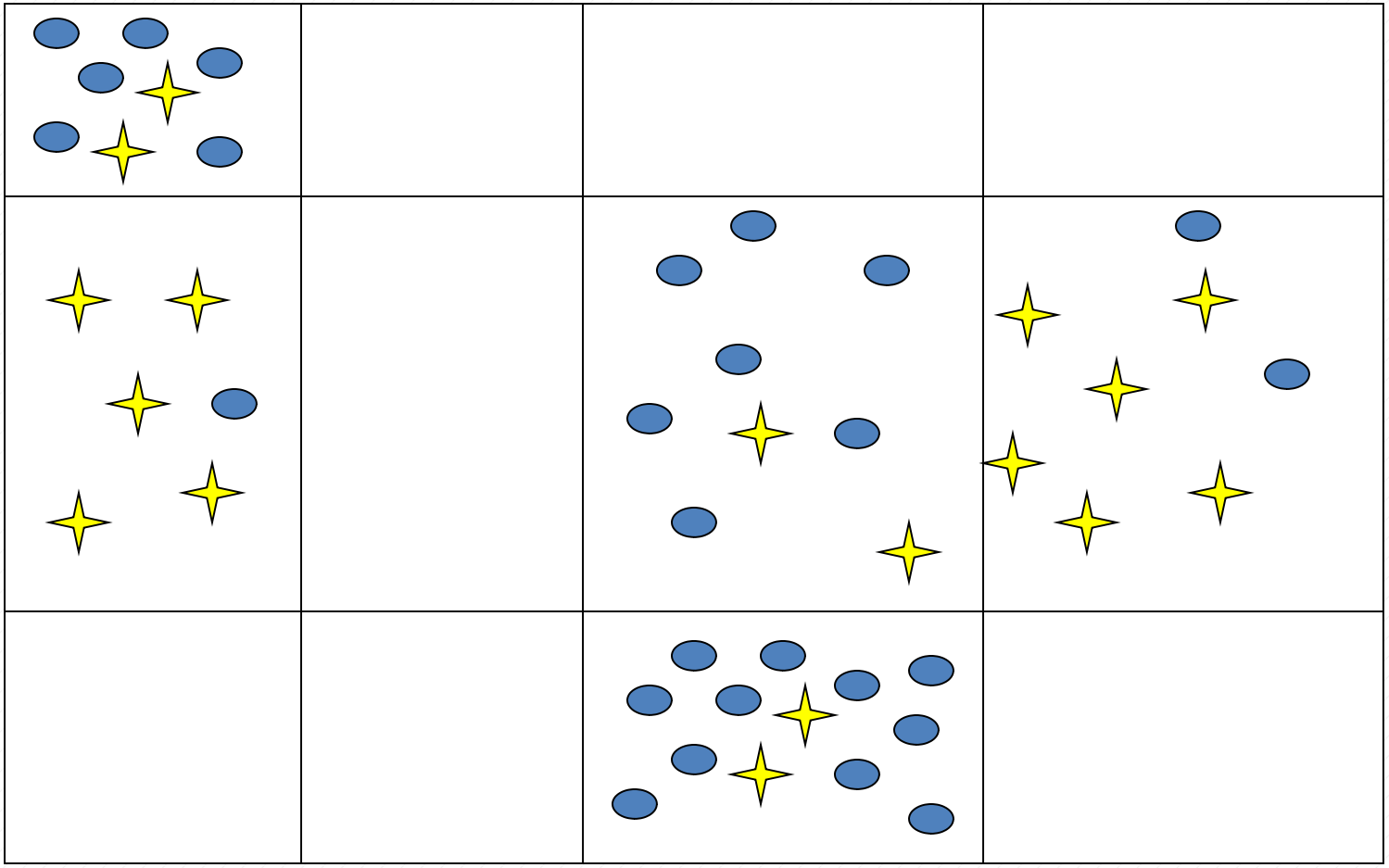
Decision Tree

A programming task



Classification Learning: Definition

Given a collection of records (*training set*)

- Each record contains a set of *attributes*, one of the attributes is the *class*

Find a *model* for the class attribute as a function of the values of the other attributes

Goal: previously unseen records should be assigned a class as accurately as possible

- Use *test set* to estimate the accuracy of the model
- Often, the given data set is divided into training and test sets, with training set used to build the model and test set used to validate it

Classification

- Classification is a most familiar and most popular data mining technique.
- Classification applications includes image and pattern recognition, loan approval, detecting faults in industrial applications.
- All approaches to performing classification assumes some knowledge of the data.
- Training set is used to develop specific parameters required by the technique.

Classification Learning Techniques

- **Decision tree-based methods**
- Rule-based methods
- Instance-based methods
- Probability-based methods
- Neural networks
- Support vector machines
- Logic-based methods

Decision tree

- **A decision tree takes as input an object or situation described by a set of attributes and returns a "decision" the predicted output value for the input.**

  A decision tree is a tree where each node represents a feature(attribute), each link(branch) represents a decision(rule) and each leaf represents an outcome(categorical or continues value).

A decision tree is a natural and simple way of inducing following kind of rules.

If (Age is x) and (income is y) and (family size is z) and (credit card spending is p) then he will accept the loan

It is powerful and perhaps most widely used modeling technique of

Why decision tree?

- Decision trees are powerful and popular tools for classification and prediction.
- Decision trees represent *rules*, which can be understood by humans and used in knowledge system such as database.

key requirements

- **Attribute-value description:** object or case must be expressible in terms of a fixed collection of properties or attributes (e.g., hot, mild, cold).

- **Predefined classes (target values):** the target function has **discrete output values** (bolean or multiclass)

- **Sufficient data:** enough training cases should be provided to learn the model.

Decision tree

A decision tree reaches its decision by performing a sequence of tests.

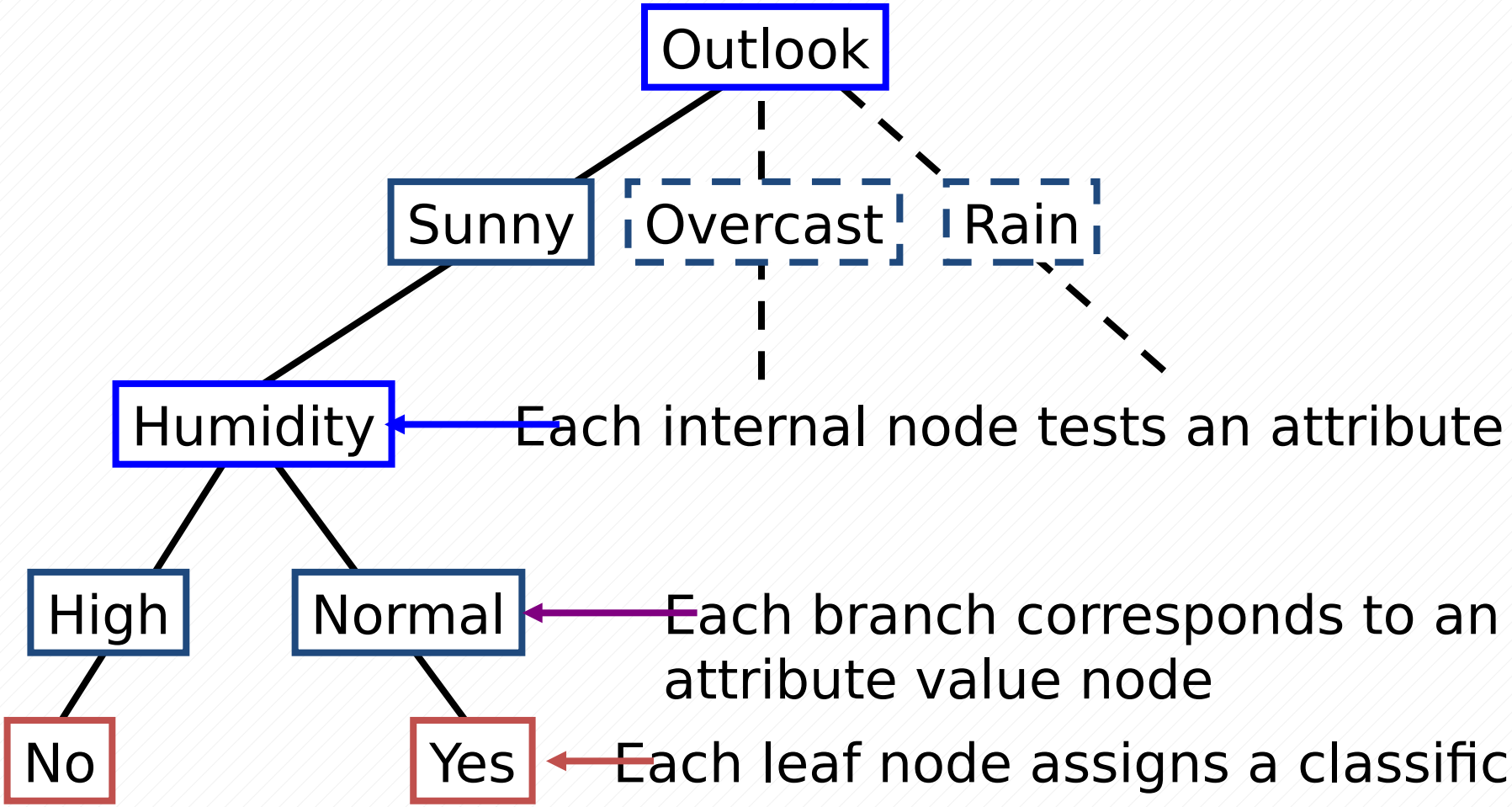
- Tree where the root and each internal node is labeled with a question.
- The arcs represent each possible answer to the associated question.
- Each leaf node represents a prediction of a solution to the problem.

- Populer technique for classification; Leaf node indicates class to which the corresponding tuple belongs.

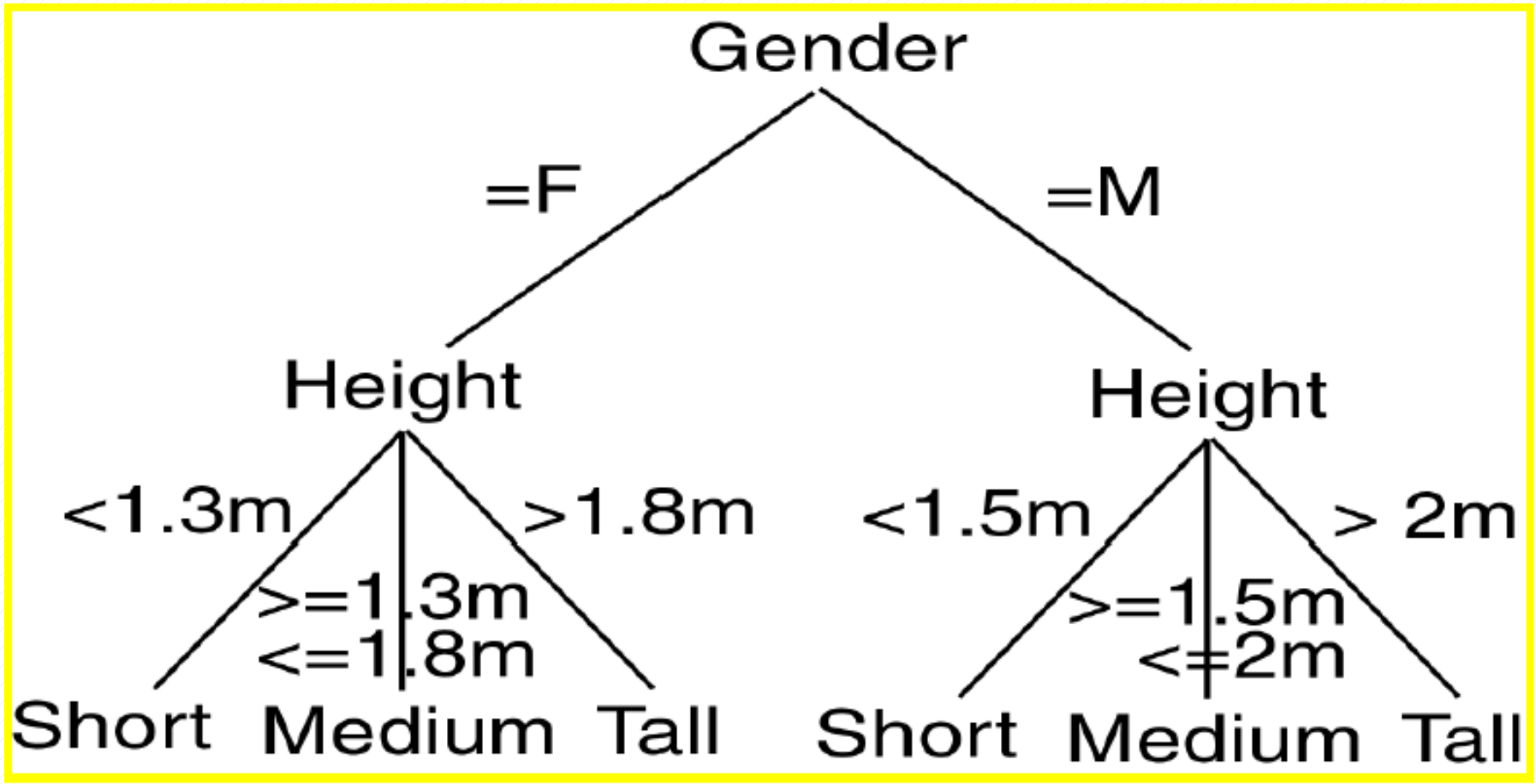
Decision Tree Representation

- Each internal node tests an attribute
- Each branch corresponds to attribute value
- Each leaf node assigns a classification

Decision Tree for PlayTennis



Decision tree



Decision Tree Model

A **Decision Tree Model** is a computational model consisting of three parts:

- **Decision Tree**
- **Algorithm to create the tree**
- **Algorithm that applies the tree to data**

• Creation of the tree is the most difficult part.

• Processing is basically a search similar to that in a binary search tree (although DT may not be binary).

Decision Tree Algorithm

```
Input:
    T       //Decision Tree
    D       //Input Database
Output:
    M       //Model Prediction
DTProc Algorithm:
                //Illustrates Prediction Technique using DT
    for each t ∈ D do
        n = root node of T;
        while n not leaf node do
            Obtain answer to question on n applied t;
            Identify arc from t which contains correct answer;
            n = node at end of this arc;
        Make prediction for t based on labeling of n;
```

Decision Tree Algorithm

- The decision tree approach is most useful in classification problems. With this technique, a tree is constructed to model the classification process.
- Once the tree is build, it is applied to each tuple in the database and results in a classification for that tuple.
- There are two basics step in this techinque: Building the tree and Applying the tree to the database.

Decision Tree Algorithm

- The decision tree approach to classification is to divide the search space into rectangular region.
- A tuple is classified based on the region into which it falls.
- Definition: Given a database D={t1……..tn} where ti=<ti1……..tih> and the database schema consist of following attributes {A1,A2,……,Ah}valso a set of classes C={C1, ……,Cm}. A decision
- tree DT or classification tree is a tree associated with D that has the following properties:
  - Each internal node is labeled with an attribute Ai
  - Each arc is labeled with a predicate that can be applied to a attribute associated with a parent.
  - Each leaf node is labeled with a class Cj.

Built Decision Tree Algorithm

- **Algorithm BuiltDT**
- Input: D : Training data set
- Output: T : Decision tree

Steps

1. If all tuples in D belongs to the same class C_j

   Add a leaf node labeled as C_j

   Return *// Termination condition*

2. **Select** an attribute A_i (so that it is not selected twice in the same branch)

3. **Partition** $D = \{ D_1, D_2, \ldots, D_p \}$ based on p different values of A_i in D

4. For each $D_k \in D$

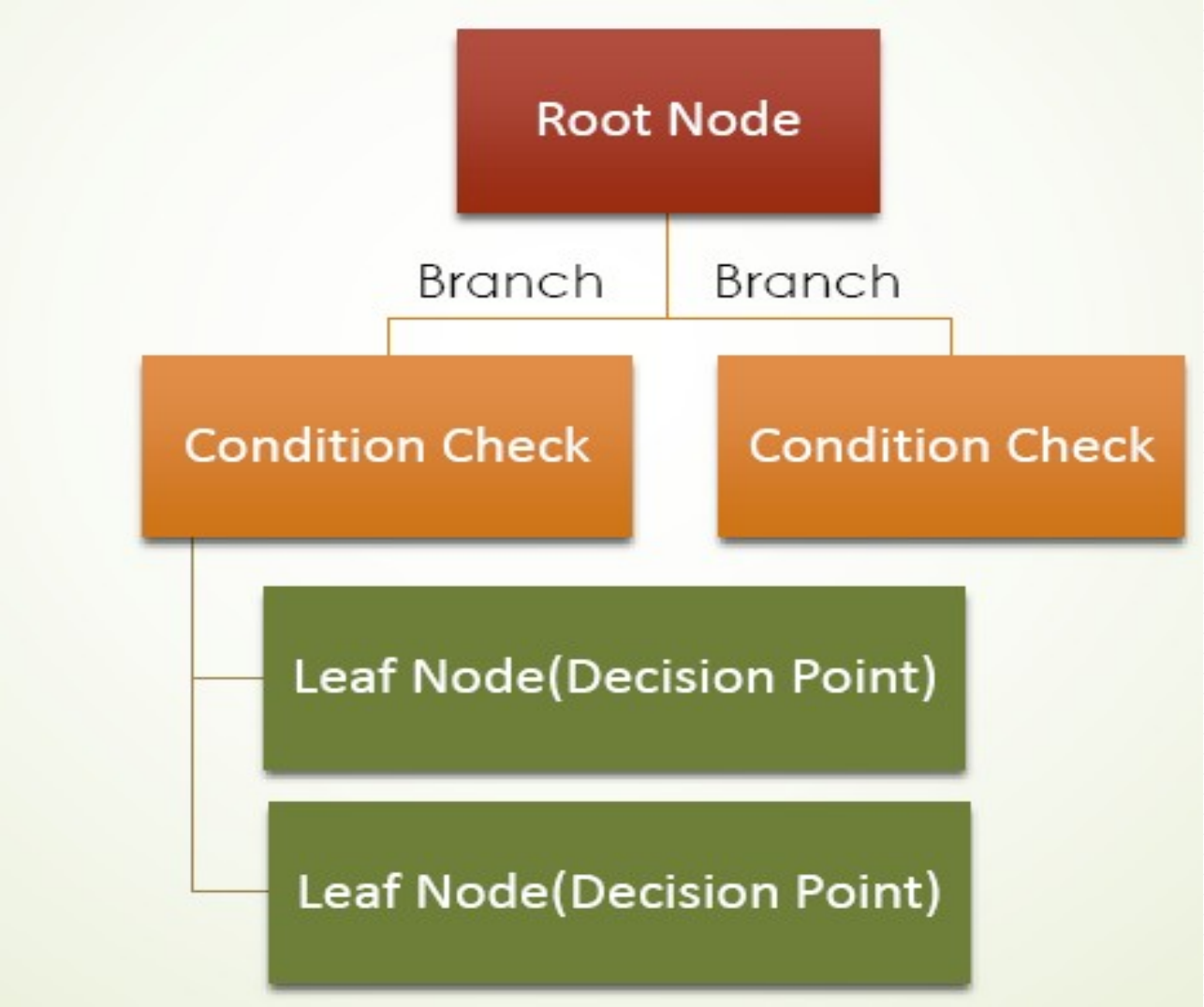
   Create a node and add an edge between D and D_k with label as the A_i's attribute value in D_k

5. For each $D_k \in D$

   BuildTD(D_k) *// Recursive call*

6. Stop

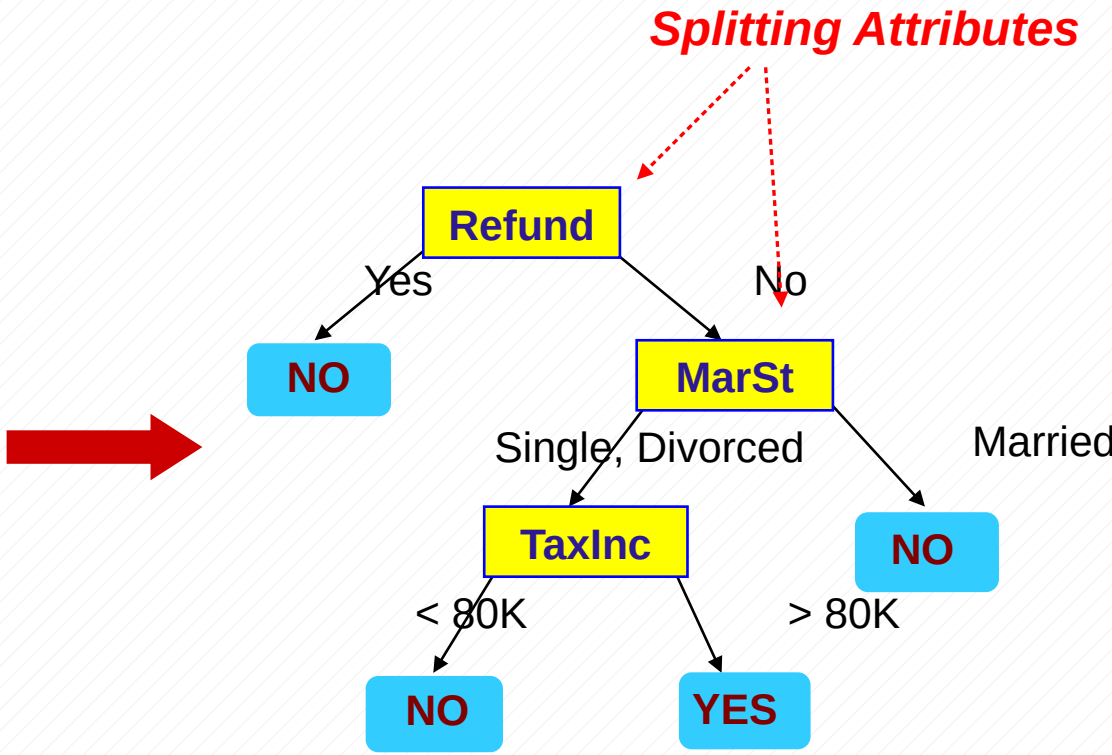
DECISION TREE TERMS



Example of a Decision Tree

	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

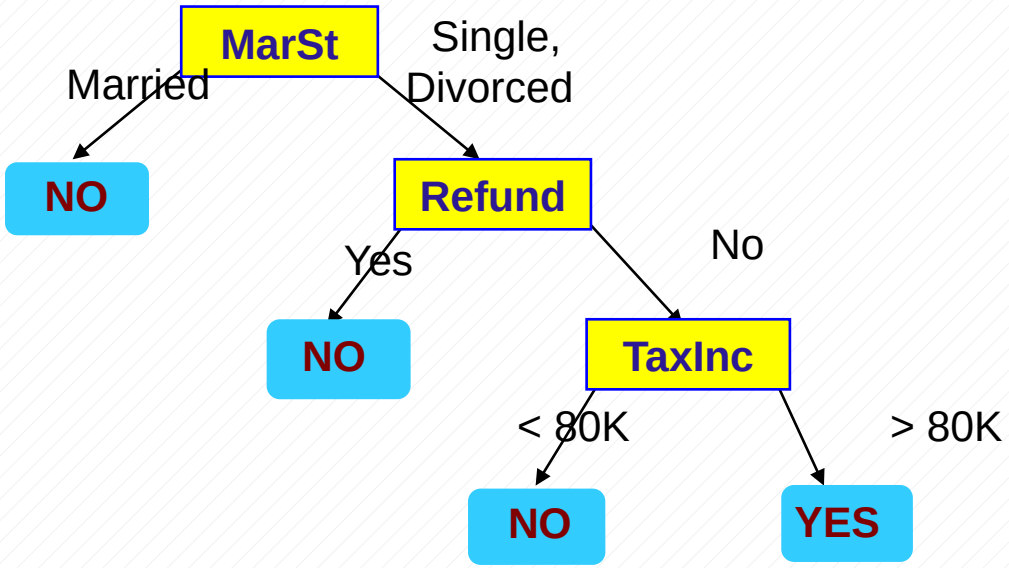
Training Data



Model: Decision Tree

Another Example of Decision Tree

Tid	Refund (categorical)	Marital Status (categorical)	Taxable Income (continuous)	Cheat (class)
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



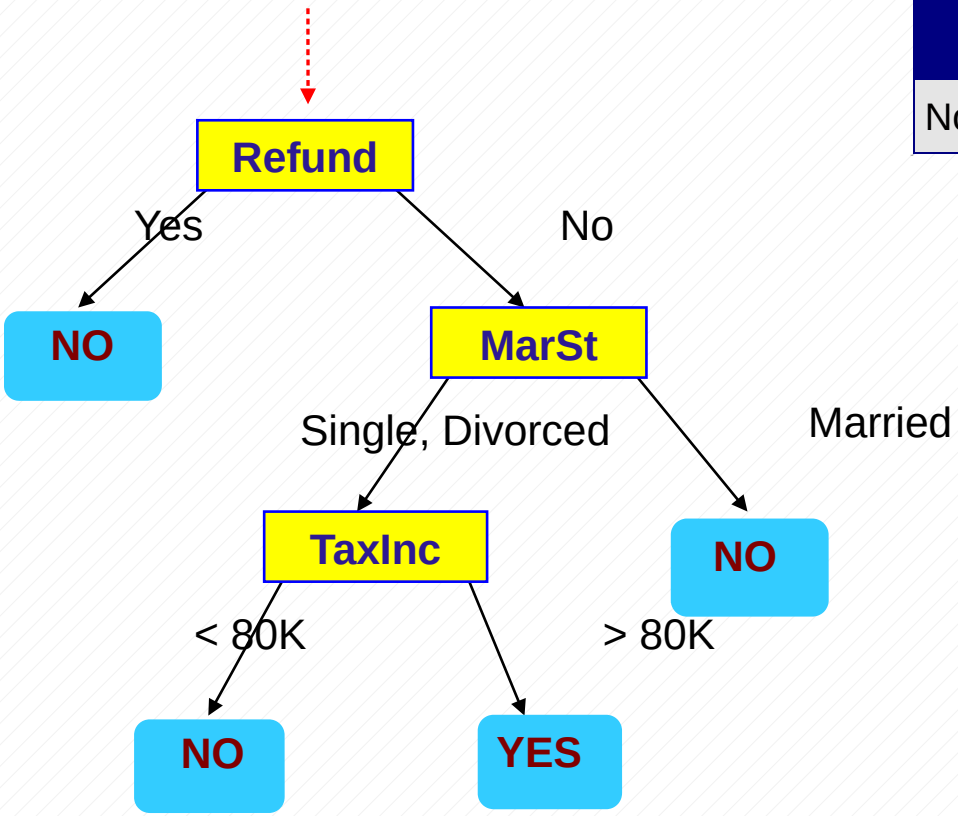
There could be more than one tree that fits the same data!

Decision Tree Classification Task

Decision Tree

Apply Model to Test Data

Start from the root of tree.



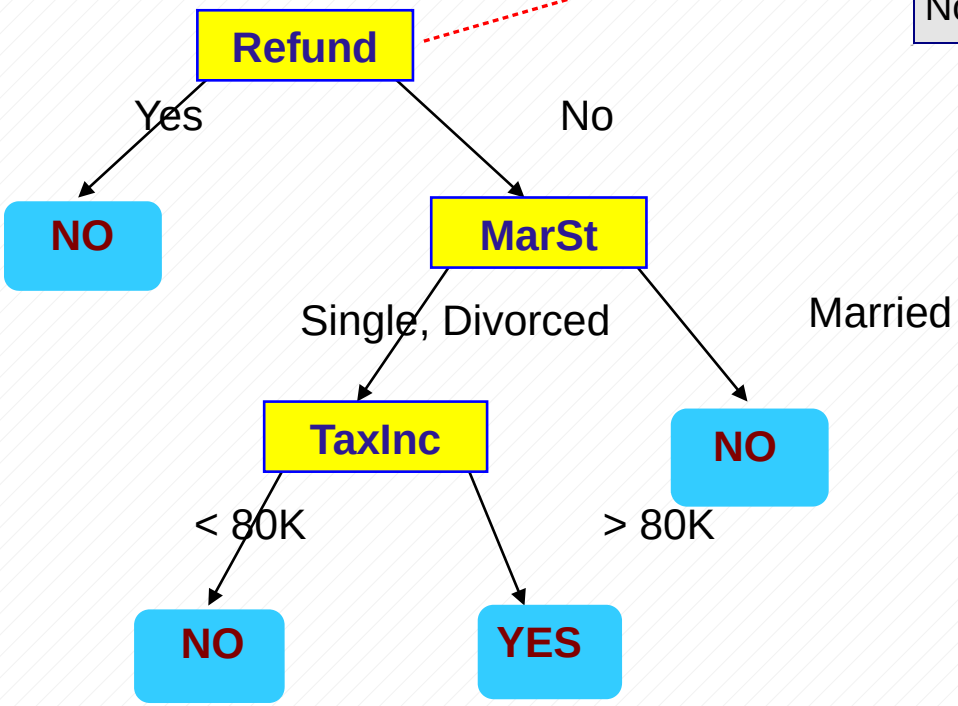
Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

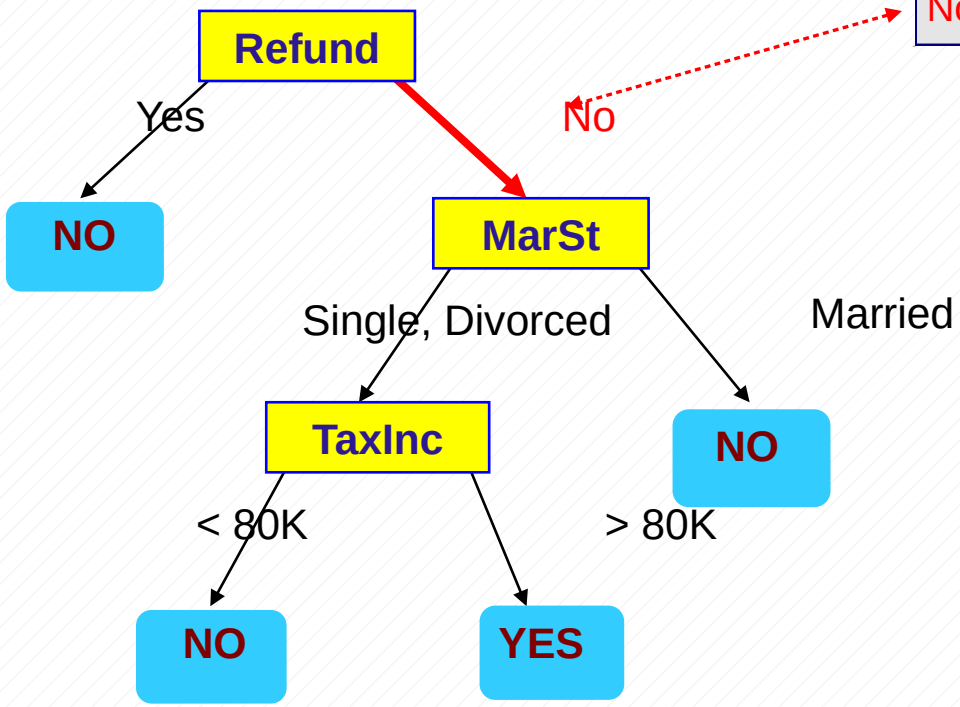
Apply Model to Test Data

Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



Apply Model to Test Data



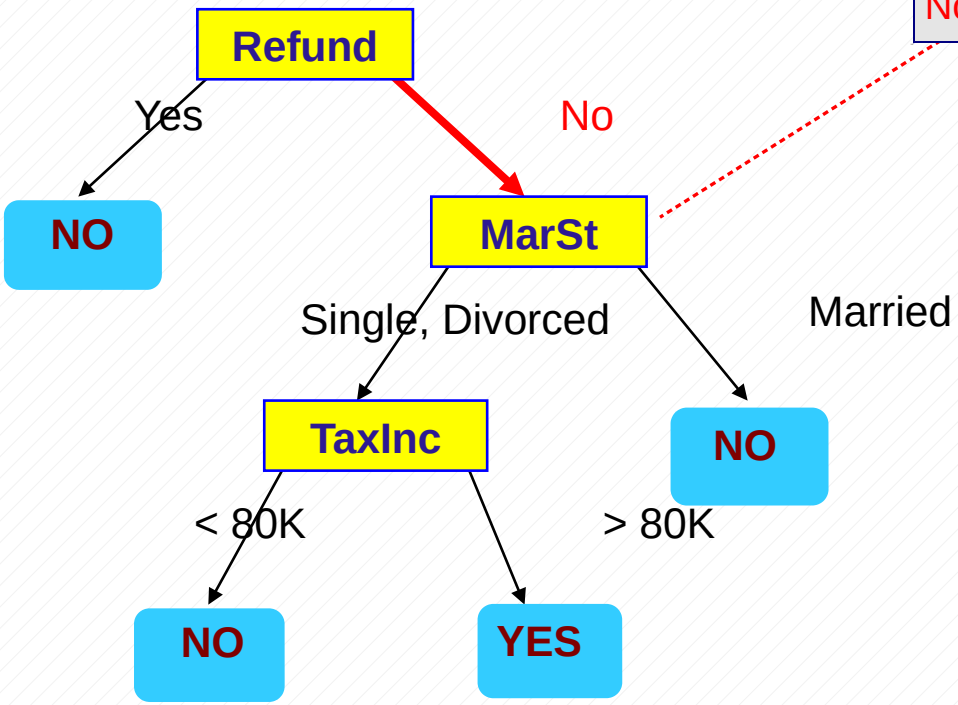
Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Apply Model to Test Data

Test Data

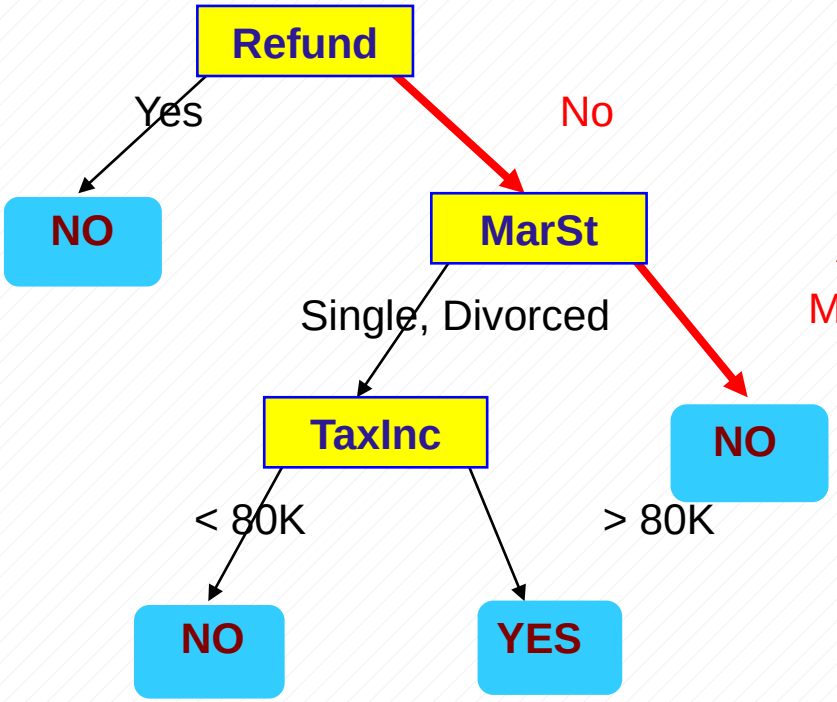
Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



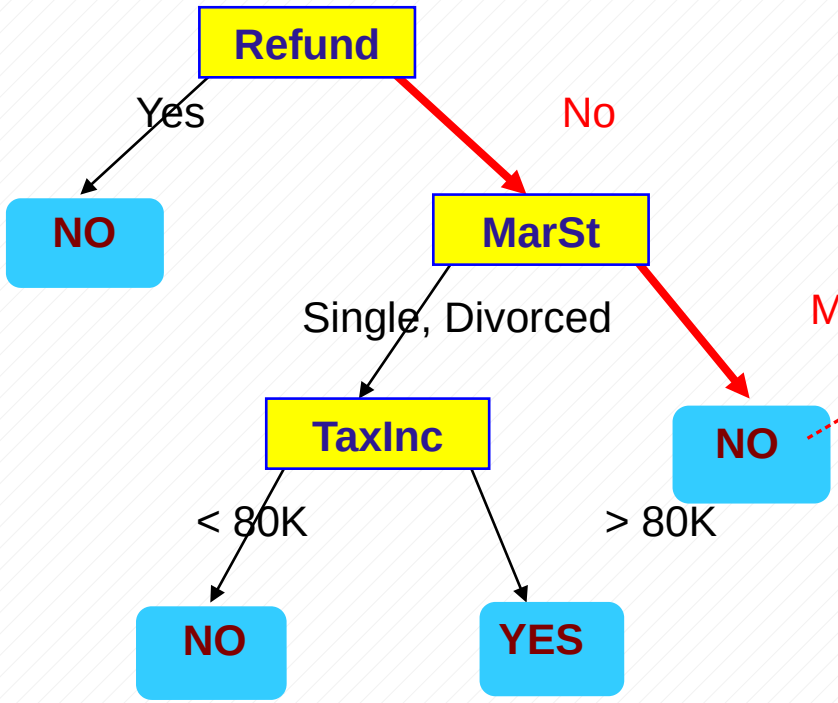
Apply Model to Test Data

Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



Apply Model to Test Data



Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Assign Cheat to “No”

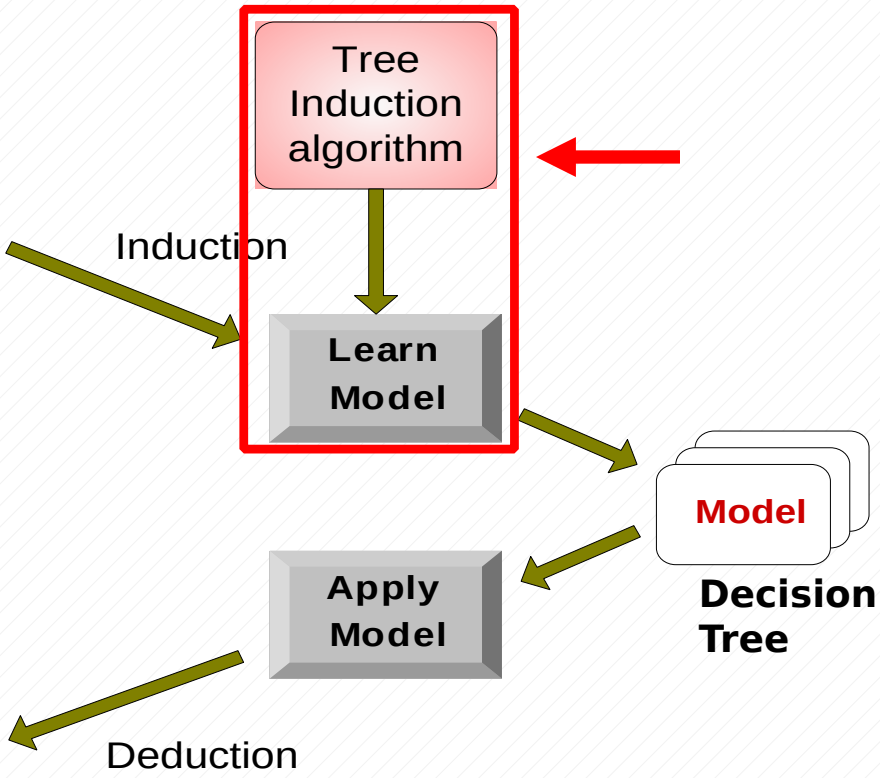
Decision Tree Classification Task

Tid	Attrib1	Attrib2	Attrib3	Class
1	Yes	Large	125K	No
2	No	Medium	100K	No
3	No	Small	70K	No
4	Yes	Medium	120K	No
5	No	Large	95K	Yes
6	No	Medium	60K	No
7	Yes	Large	220K	No
8	No	Small	85K	Yes
9	No	Medium	75K	No
10	No	Small	90K	Yes

Training Set

Tid	Attrib1	Attrib2	Attrib3	Class
11	No	Small	55K	?
12	Yes	Medium	80K	?
13	Yes	Large	110K	?
14	No	Small	95K	?
15	No	Large	67K	?

Test Set

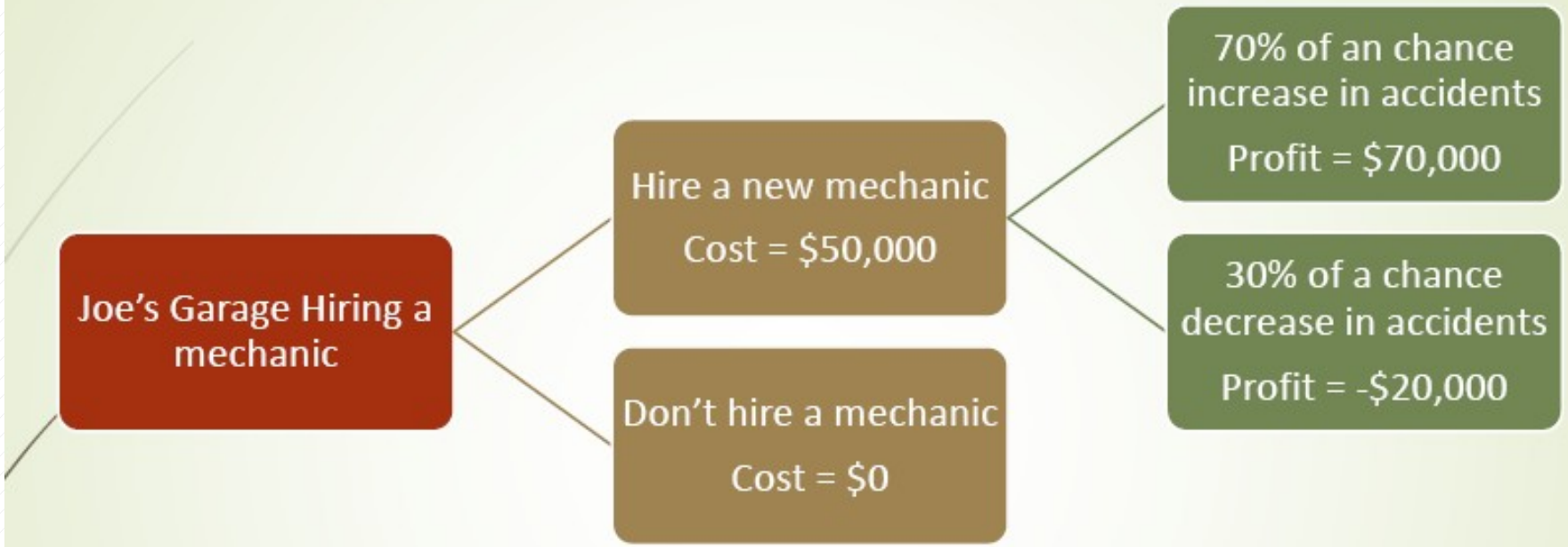


EASY EXAMPLE

- Joe’s garage is considering hiring another mechanic.
- The mechanic would cost them an additional $50,000 / year in salary and benefits.
- If there are a lot of accidents in Iowa City this year, they anticipate making an additional $75,000 in net revenue.
- If there are not a lot of accidents, they could lose $20,000 off of last year’s total net revenues.
- Because of all the ice on the roads, Joe thinks that there will be a 70% chance of “a lot of accidents” and a 30% chance of “fewer accidents”
- Assume if he doesn’t expand he will have the same revenue as last year.

EASY EXAMPLE

continued



- Estimated value of "Hire Mechanic" =
NPV =.7(70,000) + .3(- $20,000) - $50,000 = - $7,000
- Therefore you should not hire the mechanic

Example

- You want to guess the outcome of next week's game between the *MallRats* and the *Chinooks*.

- Available knowledge / Attribute
  - was the game **at Home or Away**
  - was the starting **time 5pm, 7pm or 9pm**.
  - Did **Joe play center**, or forward.
  - whether that **opponent's center was tall or not**.
  - **.....**

Basket ball data

Where	When	Fred Starts	Joe offense	Joe defense	Opp C	*OutCome*
Home	7pm	Yes	Center	Forward	Tall	*Won*
Home	7pm	Yes	Forward	Center	Short	*Won*
Away	7pm	Yes	Forward	Forward	Tall	*Won*
Home	5pm	No	Forward	Center	Tall	*Lost*
Away	9pm	Yes	Forward	Forward	Short	*Lost*
Away	7pm	No	Center	Forward	Tall	*Won*
Home	7pm	No	Forward	Center	Tall	*Lost*
Home	7pm	Yes	Center	Center	Talls	*Won*
Away	7pm	Yes	Center	Center	Short	*Won*
Home	9pm	No	Forward	Center	Short	*Lost*

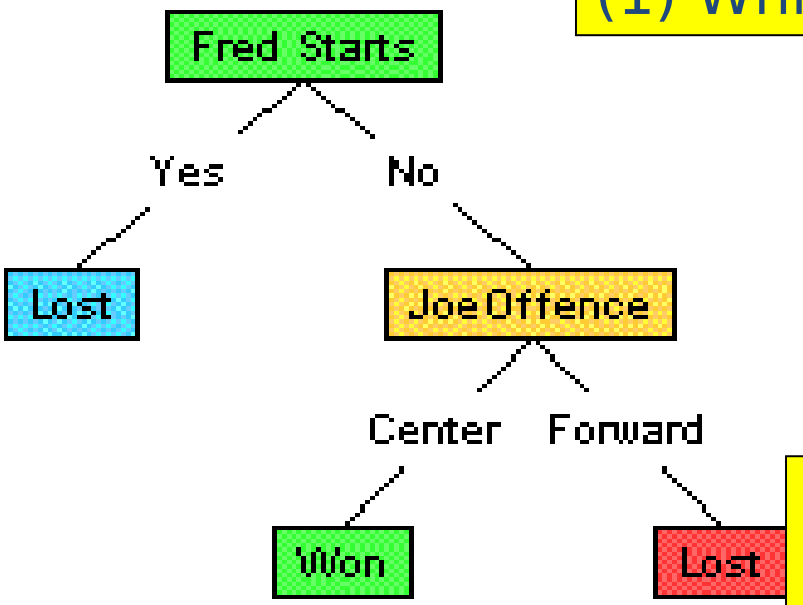
What we know

- The game will be away, at 9pm, and that Joe will play center on offense…

Where	When	Fred Starts	Joe offense	Joe defense	Opp C	*Outcome*
Away	9pm	No	Center	Forward	Tall	??

- A classification problem
- Generalizing the learned rule to new examples

Illustration



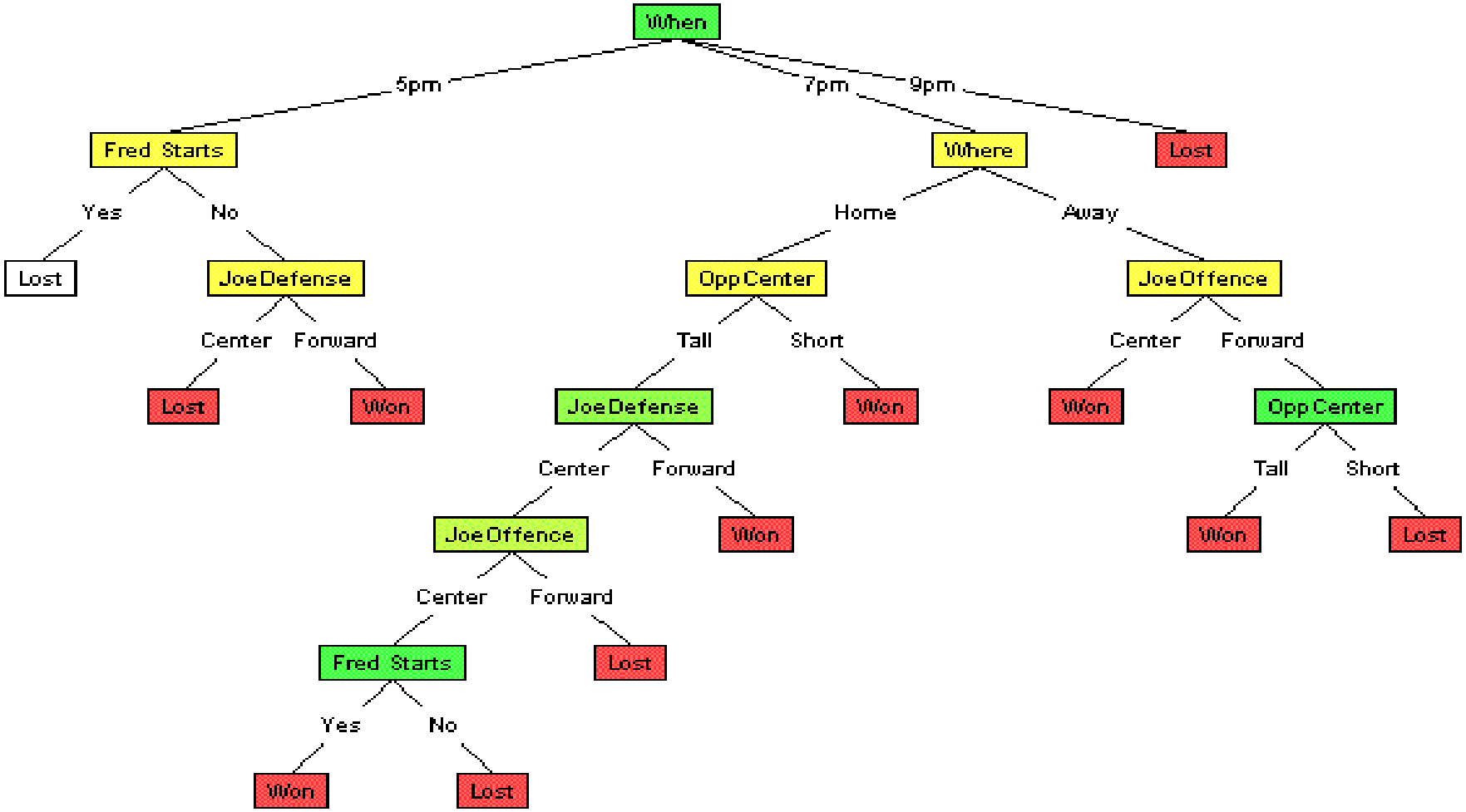
(1) Which to start? (root)

(2) Which node to proceed?

(3) When to stop/ come to conclusion?

Random split

- The tree can grow **h**uge
- These trees are hard to understand.
- Larger trees are typically less accurate than smaller trees.



Principled Criterion

- Selection of an attribute to test at each node - choosing the most useful attribute for classifying examples.
- information gain
  - measures how well a given attribute separates the training examples according to their target classification
  - This measure is used to select among the candidate attributes at each step while growing the tree

ow to build This Decision Tree?

There are couple of algorithms there to build a decision tree , we only talk about a few which are

- ID3 (Iterative Dichotomiser 3) → uses **Entropy function** and **Information gain** as metrics.

- CART (Classification and Regression Trees) → uses **Gini Index(Classification)** as metric.

assification with using the **ID3** algor

Let’s just take a famous dataset in the machine learning world which is weather dataset(playing game Y or N based on weather condition).

We have four X values (outlook,temp,humidity and windy) being categorical and one y value (play Y or N).

so we need to learn the mapping between X and y.

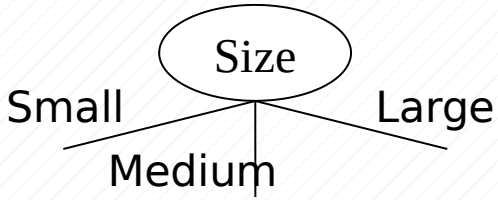
This is a binary classification problem, lets build the tree using the ***ID3*** algorithm

To create a tree, we need to have a root node first and we know that nodes are features/ attributes (outlook temp humidity and windy)

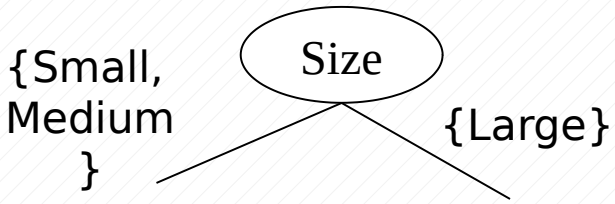
outlook	temp.	humidity	windy	play
sunny	hot	high	false	no
sunny	hot	high	true	no
overcast	hot	high	false	yes
rainy	mild	high	false	yes
rainy	cool	normal	false	yes
rainy	cool	normal	true	no
overcast	cool	normal	true	yes
sunny	mild	high	false	no
sunny	cool	normal	false	yes
rainy	mild	normal	false	yes
sunny	mild	normal	true	yes
overcast	mild	high	true	yes
overcast	hot	normal	false	yes
rainy	mild	high	true	no

Splitting Based on Ordinal Attributes

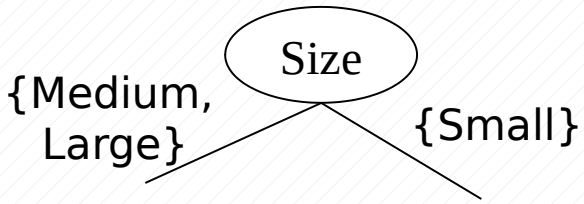
- Multi-way split: Use as many partitions as distinct values.



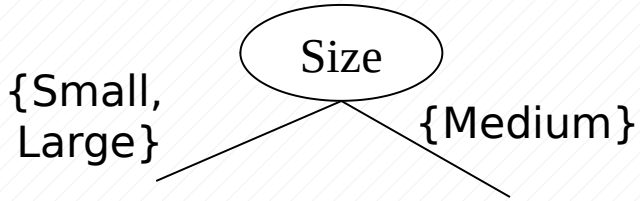
- Binary split: Divides values into two subsets. Need to find optimal partitioning.



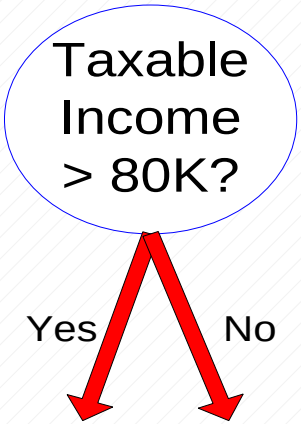
OR



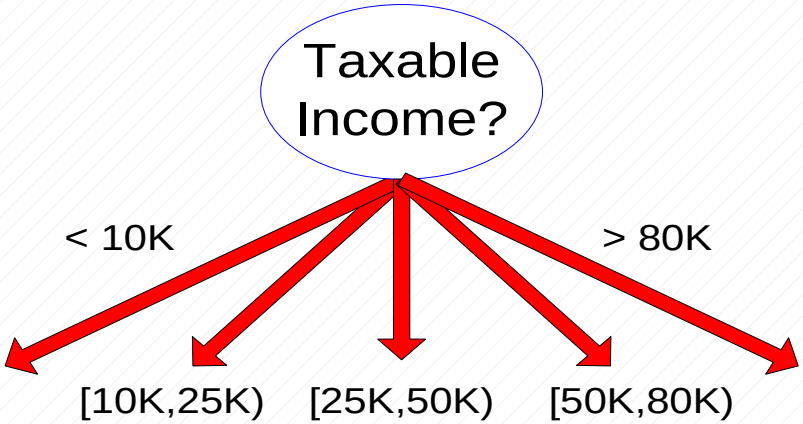
- What about this split?



Splitting Based on Continuous Attributes



(i) Binary split



(ii) Multi-way split

How to determine the Best Split

- Greedy approach:
  - Nodes with homogeneous class distribution are preferred
- Need a measure of node impurity:

C0: 5 C1: 5	C0: 9 C1: 1
Non-homogeneous, **High degree of impurity**	**Homogeneous,** **Low degree of impurity**

o which one do we need to pick first

- Determine the attribute that best classifies the training data;
- Use this attribute at the root of the tree. Repeat this process at for each branch
- This means we are performing top-down, greedy search through the space of possible decision trees.

kay so how do we choose the best ttribute?

- Use the attribute with the highest ***information gain*** in ***ID3***
- In order to define information gain precisely,
- We begin by defining a measure commonly used in information theory, called **entropy**
- Characterizes the (im)purity of an arbitrary collection of examples.”

Entropy

- Entropy H(S) is a measure of the amount in the data set (s) (ie. Entropy characterizes the (dat)set S) $H(S) = \sum_{c \in C} -p(c) log_2 p(c)$
- Where,
- S- The curent (data)set for which entropy is being calculated(changes every iteration of elements of the ID3 Algorithm).
- C- Set of classes in S c=(yes, no)
- P(c)- The proportion of the number of elements in class c to the number of elements in set S.

When H(s)=0 then the set S is perfectly classified (All element are the same class.)
In ID3 entropy is calculated for each remaining attribute. The attribute with the smallest entropy is used to split the set S on this iteration. the higher entropy, the higher potential to improve the classification.

Entropy (as information)

- Entropy at a given node t:

$$Entropy(t) = -\sum_j p(j \mid t) \log p(j \mid t)$$

(where $p(j \mid t)$ is the relative frequency of class j at node t)

Based on Shannon's information theory

For simplicity, assume only 2 classes *Yes* and *No*

Assume *t* is a set of messages sent to a receiver that must guess their class

If *p(Yes | t)*=1 (resp., *p(No | t)*=1), then the receiver guesses a new example as *Yes* (resp., *No*). No message need be sent.

If *p(Yes | t)*=*p(No | t)*=0.5, then the receiver cannot guess and must be told the class of a new example. A 1-bit message must be sent.

If 0<*p(Yes | t)*<1, then the receiver needs less than 1 bit

Entropy (as homogeneity)

- Think chemistry/physics
  - Entropy is measure of disorder or homogeneity
  - Minimum (0.0) when homogeneous / perfect order
  - Maximum (1.0, in general log C) when most heterogeneous / complete chaos
- In ID3
  - Minimum (0.0) when all records belong to one class, implying most information
  - Maximum (log C) when records are equally distributed among all classes, implying least information
  - Intuitively, the smaller the entropy the purer the partition

Entropy

For a binary classification problem

- If all examples are positive or all are negative then entropy will be ***zero*** i.e, low.

- If half of the examples are of positive class and half are of negative class then entropy is **one** i.e, high.

Information Gain

Information gain IG(A) is the measure of the difference in entropy from before the after the set S is split on the an attribute A. or uncertainity in S is reduced after the splitting set S on attribute A

$$IG(A,S) = H(S) - \sum_{t \epsilon T} p(t) H(t)$$

Where

H(s)- Entropy of set S

T- the subset created from splitting set S by attribute A such that $S = \bigcup_{t \epsilon T} t$

P(t)- the proportin of the number of elements in **t** to the number of elments in set **S**.

H(t)-Entropy of subset t.

In ID3, Information gain is calculated for each remaining attribute. The attribute with the largest information gain is used to split the set S on this iteration.

Information Gain

- Information Gain:

$$GAIN_{split} = Entropy(p) - \left(\sum_{i=1}^{k} \frac{n_i}{n} Entropy(i) \right)$$

(where parent Node, p is split into k partitions, and n_i is number of records in partition i)

  - Measures reduction in entropy achieved because of the split maximize
  - ID3 chooses to split on the attribute that results in the largest reduction, i.e, (maximizes GAIN)
  - Disadvantage: Tends to prefer splits that result in large number of partitions, each being small but pure.

Information Gain

Okay lets apply these metrics to our dataset to split the data(getting the root node) Steps:

1.Compute the entropy for data-set

2.for every attribute/feature:

1.calculate entropy for all categorical values

2.take average information entropy for the current attribute

3.calculate gain for the current attribute

3. Pick the highest gain attribute.

4. Repeat until we get the tree we desired

Compute the entropy for the weather data set:

$$H(S) = \sum_{c \in C} -p(c) \log_2 p(c)$$

C ={yes,no}

Out of 14 instances, 9 are classified as yes, and 5 as no

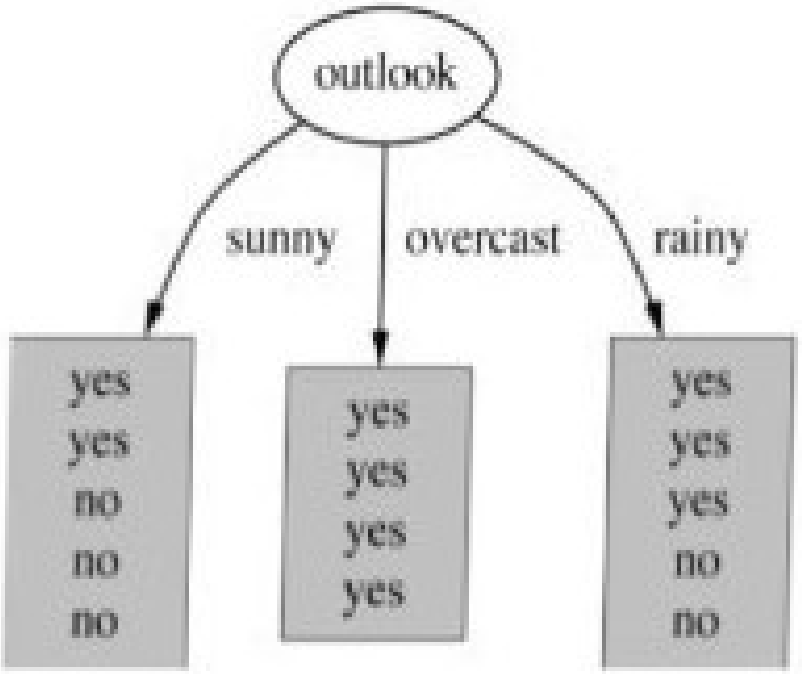
pyes = −(9/14)*log2(9/14) = 0.41

pno = −(5/14)*log2(5/14) = 0.53

H (S) = pyes + pno = 0.94

outlook	temp.	humidity	windy	play
sunny	hot	high	false	no
sunny	hot	high	true	no
overcast	hot	high	false	yes
rainy	mild	high	false	yes
rainy	cool	normal	false	yes
rainy	cool	normal	true	no
overcast	cool	normal	true	yes
sunny	mild	high	false	no
sunny	cool	normal	false	yes
rainy	mild	normal	false	yes
sunny	mild	normal	true	yes
overcast	mild	high	true	yes
overcast	hot	normal	false	yes
rainy	mild	high	true	no

Compute the entropy for the weather data set:



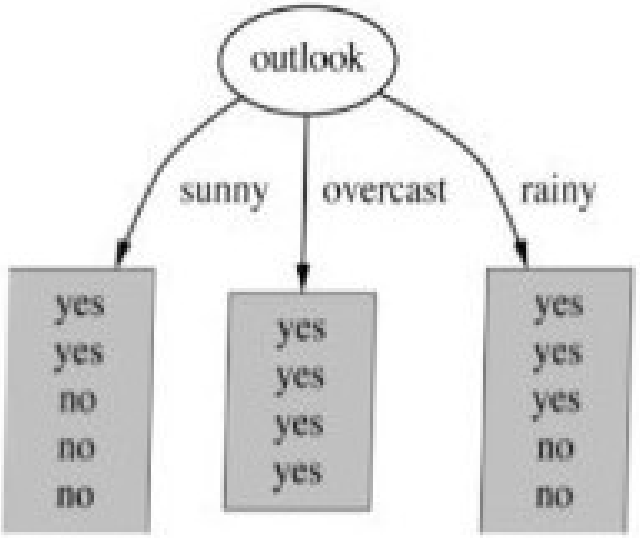
E (Outlook=sunny) = $-\frac{2}{5}\log\left(\frac{2}{5}\right)-\frac{3}{5}\log\left(\frac{3}{5}\right)=0.971$

E (Outlook=overcast) = $-1\log(1)-0\log(0)=0$

E (Outlook=rainy) = $-\frac{3}{5}\log\left(\frac{3}{5}\right)-\frac{2}{5}\log\left(\frac{2}{5}\right)=0.971$

} H(S,Outlook)

Compute the entropy for the weather data set:



Average Entropy information for Outlook

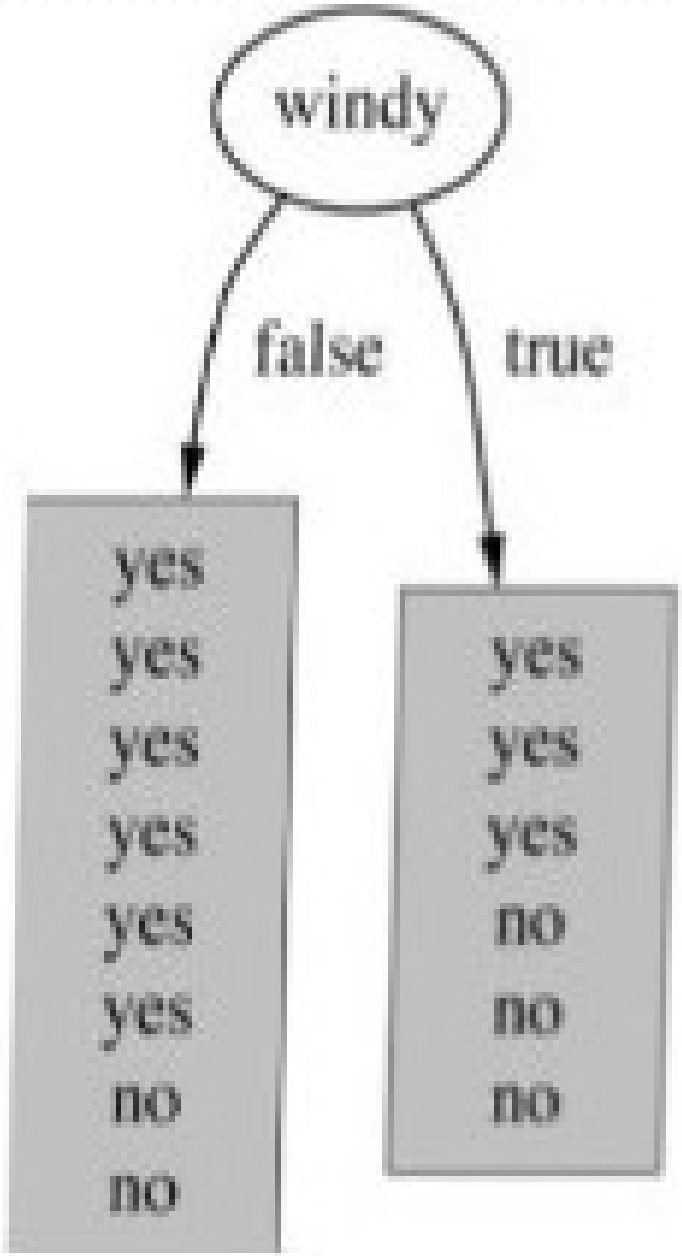
$$I(\text{Outlook}) = \frac{5}{14} * 0.971 + \frac{4}{14} * 0 + \frac{5}{14} * 0.971 = 0.693$$

$$\sum_{t \in T} p(t) H(t)$$

$$\text{Gain (Outlook)} = E(S) - I(\text{outlook}) = 0.94\text{-}.693 = 0.247$$

$$IG(A, S) = H(S) - \sum_{t \in T} p(t) H(t)$$

Compute the entropy for the weather data set:



$$E(\text{Windy=false}) = -\frac{6}{8}\log\left(\frac{6}{8}\right) - \frac{2}{8}\log\left(\frac{2}{8}\right) = 0.811$$

$$E(\text{Windy=true}) = -\frac{3}{6}\log\left(\frac{3}{6}\right) - \frac{3}{6}\log\left(\frac{3}{6}\right) = 1$$

Average entropy information for Windy

$$I(\text{Windy}) = \frac{8}{14} * 0.811 + \frac{6}{14} * 1 = 0.892$$

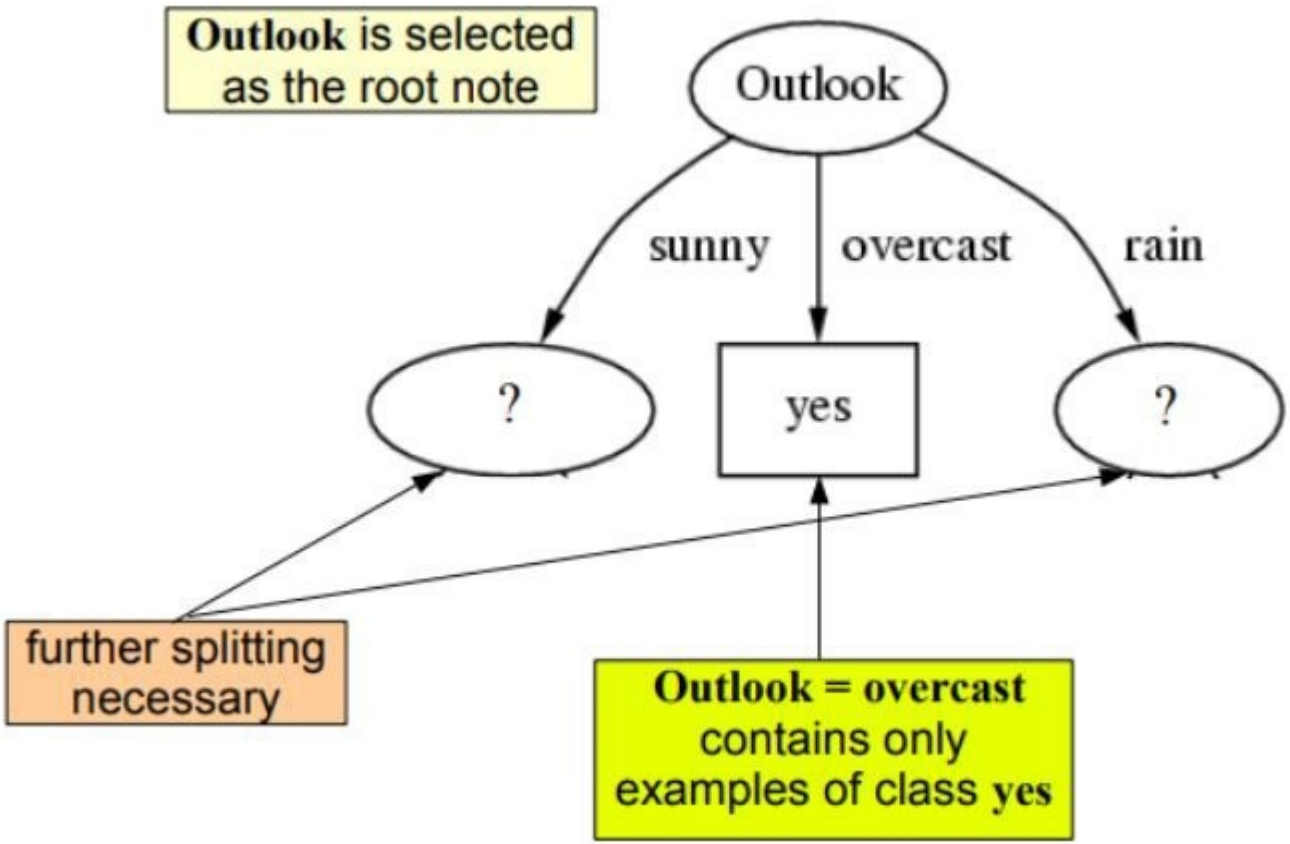
Gain (Windy) = E(S) – I (Windy) = 0.94-0.892=0.048

Similarity we can calculate for other two attributes(Humidity and Temp).

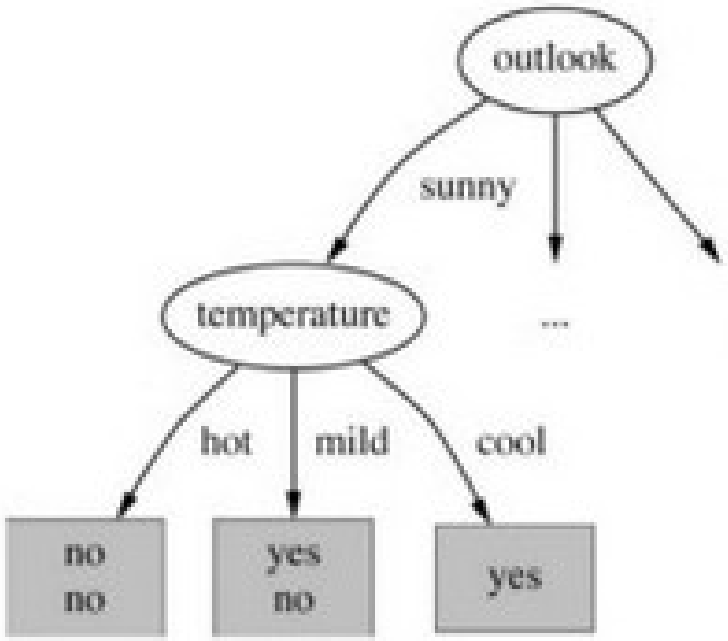
Pick the highest gain attribute.

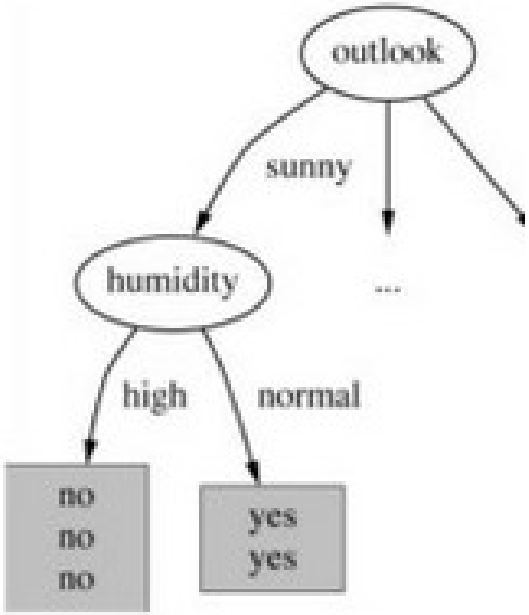
Outlook		Temperature	
Info:	0.693	Info:	0.911
Gain: 0.940-0.693	0.247	Gain: 0.940-0.911	0.029
Humidity		**Windy**	
Info:	0.788	Info:	0.892
Gain: 0.940-0.788	0.152	Gain: 0.940-0.892	0.048

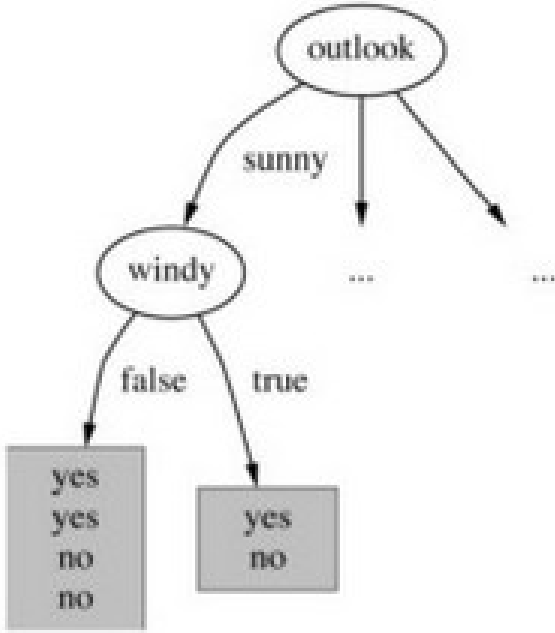
So our root node is **Outlook.**



eat the same thing for sub-trees till we get the tree.



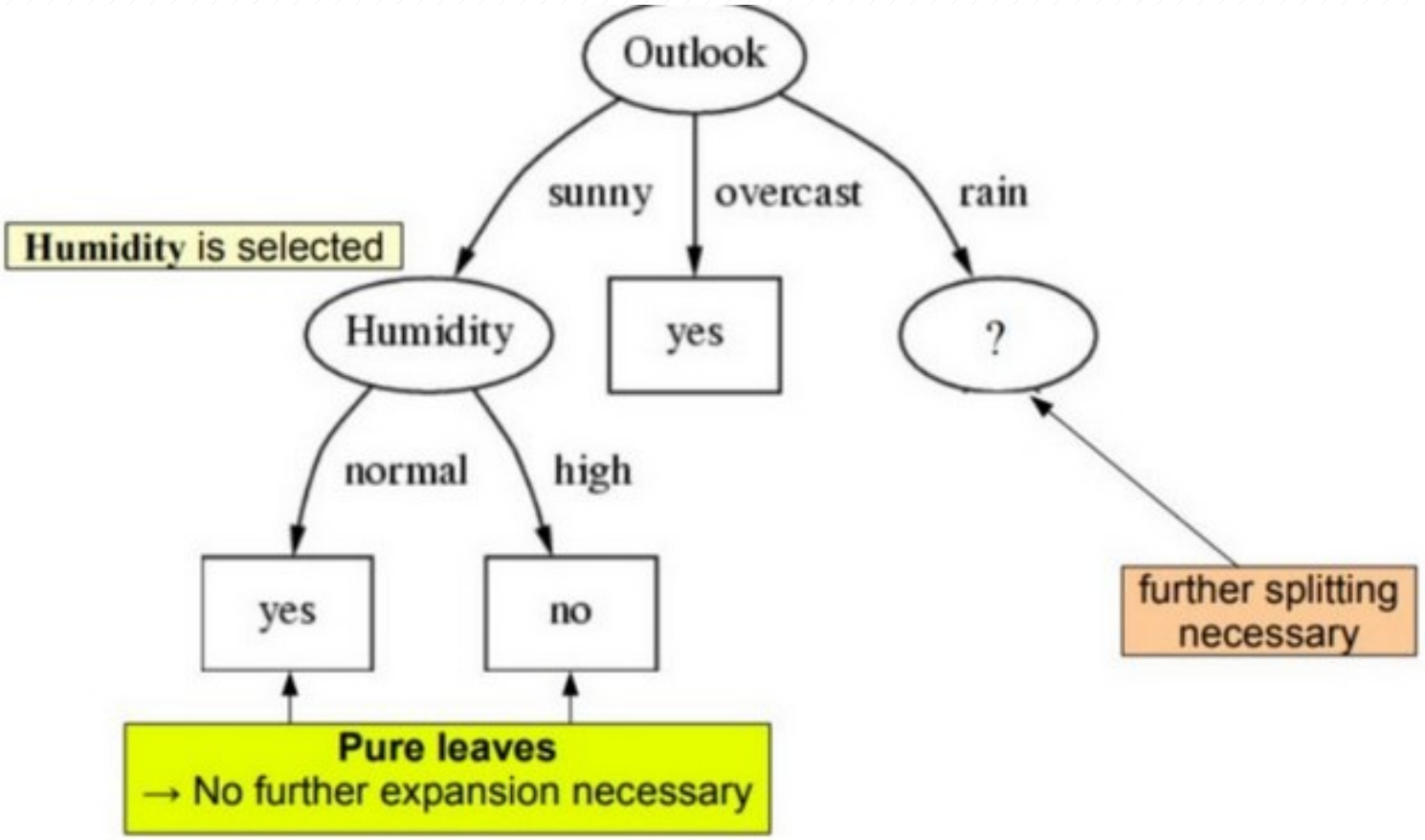




Gain(*Temperature*) = 0.571 bits
Gain(*Humidity*) = 0.971 bits
Gain(*Windy*) = 0.020 bits

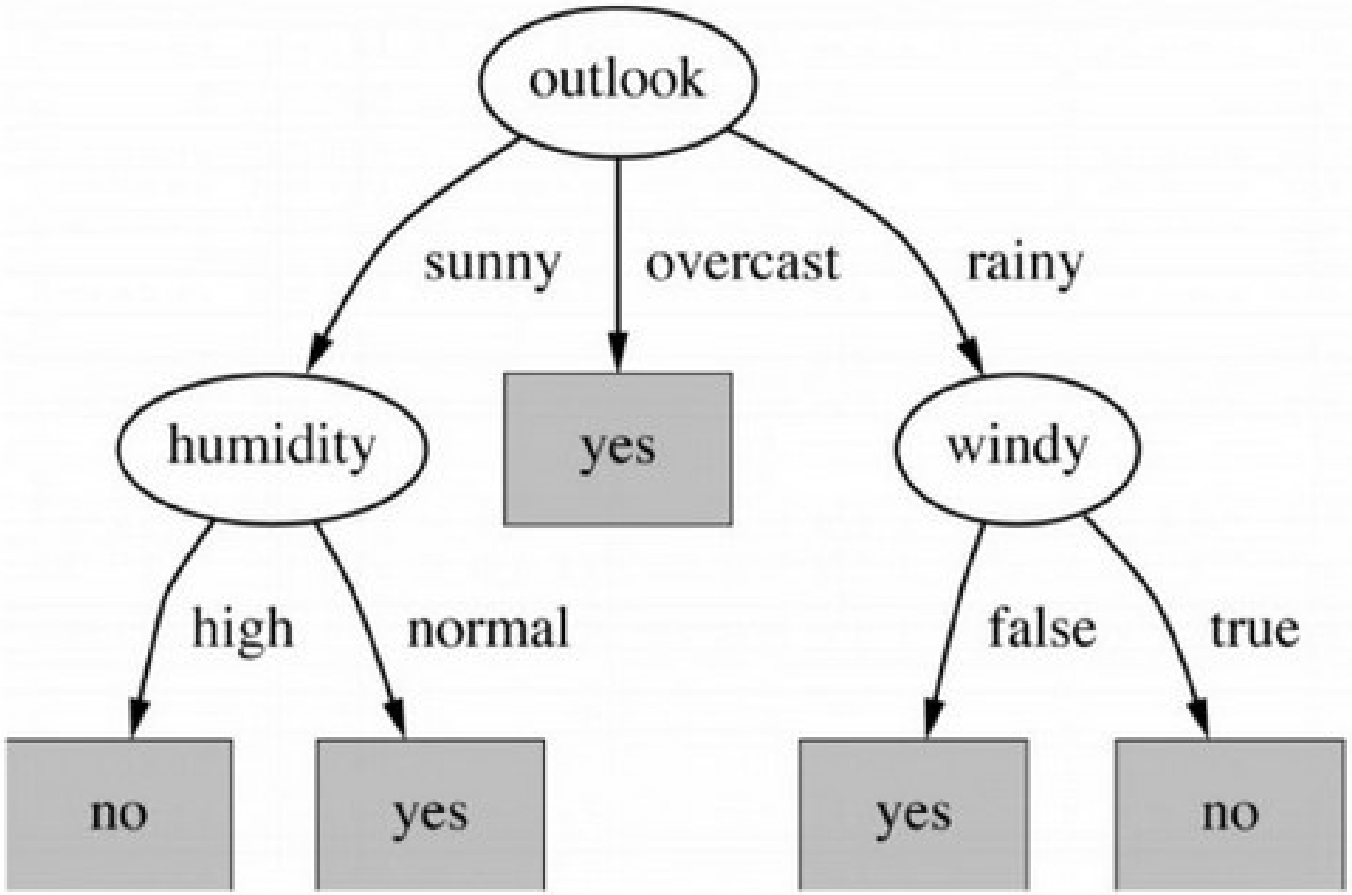
Humidity is selected

eat the same thing for sub-trees till we get the tree.



Finally we get the tree something like his.

Finale Decision tree



Entropy of a Training Set

Example 9.10: OPTH dataset

Consider the OTPH data shown in the following table with total 24 instances in it.

Age	Eye sight	Astigmatic	Use Type	Class
1	1	1	1	3
1	1	1	2	2
1	1	2	1	3
1	1	2	2	1
1	2	1	1	3
1	2	1	2	2
1	2	2	1	3
1	2	2	2	2
2	1	1	1	3
2	1	1	2	2
2	1	2	1	3
2	1	2	2	1
2	2	1	1	3
2	2	1	2	2
2	2	2	1	3
2	2	2	2	3
3	1	1	1	3
3	1	1	2	3
3	1	2	1	3
3	1	2	2	3
3	2	1	1	3
3	2	1	2	2
3	2	2	1	3
3	2	2	2	3

A coded forms for all values of attributes are used to avoid the cluttering in the table.

Information Gain Calculation

Example 9.11 :: Information gain on splitting OPTH

- Let us refer to the OPTH database discussed in previous slide
- Splitting on **Age** at the root level, it would give three subsets D_1, D_2 and D_3 as shown in the tables in the following three slides.
- The entropy $E(D_1), E(D_2)$ and $E(D_3)$ of training sets D_1, D_2 and D_3 and corresponding weighted entropy $E_{Age}(D_1)$, $E_{Age}(D_2)$ and $E_{Age}(D_3)$ are also shown alongside.
- The Information gain $\alpha\ (Age, OPTH)$ is then can be calculated as **0.0394**.
- Recall that entropy of OPTH data set, we have calculated as $E(OPTH) =$ **1.3261**

(see Slide #49)

Information Gain Calculation

Example 9.11 :: Information gain on splitting OPTH

Training set: D_1 (Age = 1)

Age	Eye-sight	Astigmatism	Use type	Class
1	1	1	1	3
1	1	1	2	2
1	1	2	1	3
1	1	2	2	1
1	2	1	1	3
1	2	1	2	2
1	2	2	1	3
1	2	2	2	1

$$E(D_1) = -\frac{2}{8}log_2(\frac{2}{8}) - \frac{2}{8}log_2(\frac{2}{8}) - \frac{4}{8}log_2(\frac{4}{8}) = \mathbf{1.5}$$

$$E_{Age}(D_1) = \frac{8}{24} \times 1.5 = \mathbf{0.5000}$$

Calculating Information Gain

Training set: ($Age = 2$)

Age	Eye-sight	Astigmatism	Use type	Class
2	1	1	1	3
2	1	1	2	2
2	1	2	1	3
2	1	2	2	1
2	2	1	1	3
2	2	1	2	2
2	2	2	1	3
2	2	2	2	3

$$E(D_2) = -\frac{1}{8}log_2(\frac{1}{8}) - \frac{2}{8}log_2(\frac{2}{8}) - \frac{5}{8}log_2(\frac{5}{8})$$
$$= 1.2988$$

$$E_{Age}(D_2) = \frac{8}{24} \times 1.2988 = 0.4329$$

Calculating Information Gain

Training set: ($Age = 3$)

Age	Eye-sight	Astigmatism	Use type	Class
3	1	1	1	3
3	1	1	2	3
3	1	2	1	3
3	1	2	2	1
3	2	1	1	3
3	2	1	2	2
3	2	2	1	3
3	2	2	2	3

$$E(D_3) = -\frac{1}{8}log_2(\frac{1}{8}) - \frac{1}{8}log_2(\frac{1}{8}) - \frac{6}{8}log_2(\frac{6}{8}) = 1.0613$$

$$E_{Age}(D_3) = \frac{8}{24} \times 1.0613 = 0.3504$$

$$Gain(D_3) = 1.3261 - (0.5000 + 0.4329 + 0.3504) = 0.0394$$

Information Gains for Different Attributes

- In the same way, we can calculate the information gains, when splitting the OPTH database on Eye-sight, Astigmatic and Use Type. The results are summarized below.
- Splitting attribute: Age

$$\alpha(Age, OPTH) = 0.0394$$

- Splitting attribute: Eye-sight

$$\alpha(Eye - sight, OPTH) = 0.0395$$

- Splitting attribute: Astigmatic

$$\alpha(\text{Astigmatic}, OPTH) = 0.3770$$

- Splitting attribute: Use Type

$$\alpha(\text{Use Type}, OPTH) = 0.5488$$

Illustrative Training Set

Risk Assessment for Loan Applications

Client #	Credit History	Debt Level	Collateral	Income Level	RISK LEVEL
1	Bad	High	None	Low	**HIGH**
2	Unknown	High	None	Medium	**HIGH**
3	Unknown	Low	None	Medium	**MODERATE**
4	Unknown	Low	None	Low	**HIGH**
5	Unknown	Low	None	High	**LOW**
6	Unknown	Low	Adequate	High	**LOW**
7	Bad	Low	None	Low	**HIGH**
8	Bad	Low	Adequate	High	**MODERATE**
9	Good	Low	None	High	**LOW**
10	Good	High	Adequate	High	**LOW**
11	Good	High	None	Low	**HIGH**
12	Good	High	None	Medium	**MODERATE**
13	Good	High	None	High	**LOW**
14	Bad	High	None	Medium	**HIGH**

Algorithm CART

CART Algorithm

- It is observed that information gain measure used in ID3 is biased towards test with many outcomes, that is, it prefers to select attributes having a large number of values.
- L. Breiman, J. Friedman, R. Olshen and C. Stone in 1984 proposed an algorithm to build a binary decision tree also called CART decision tree.
  - CART stands for **Classification and Regression Tree**
  - In fact, invented independently at the same time as ID3 (1984).
  - ID3 and CART are two cornerstone algorithms spawned a flurry of work on decision tree induction.
- CART is a technique that generates a **binary decision tree;** That is, unlike ID3, in CART, for each node only two children is created.
- ID3 uses Information gain as a measure to select the best attribute to be splitted, whereas CART do the same but using another measurement called **Gini index** . It is also known as **Gini Index of Diversity** and is denote as .

Gini Index of Diversity

Definition 9.6: **Gini Index**

Suppose, D is a training set with size $|D|$ and $C = \{c_1, c_2, \ldots, c_k\}$ be the set of k classifications and $A = \{a_1, a_2, \ldots, a_m\}$ be any attribute with m different values of it. Like entropy measure in ID3, CART proposes Gini Index (denoted by G) as the measure of impurity of D. It can be defined as follows.

$$G(D) = 1 - \sum_{i=1}^{k} p_i^2$$

where p_i is the probability that a tuple in D belongs to class c_i and p_i can be estimated as

$$p_i = \frac{|C_{i,D}|}{D}$$

where $|C_{i,D}|$ denotes the number of tuples in D with class c_i.

Gini Index of Diversity

Note

- $G(D)$ measures the "impurity" of data set D.
- The smallest value of $G(D)$ is zero
  - which it takes when all the classifications are same.
- It takes its largest value $= 1 - \frac{1}{k}$
  - when the classes are evenly distributed between the tuples, that is the frequency of each class is $\frac{1}{k}$.

Gini Index of Diversity

Definition 9.7: **Gini Index of Diversity**

Suppose, a binary partition on A splits D into D_1 and D_2, then the weighted average Gini Index of splitting denoted by $G_A(D)$ is given by

$$G_A(D) = \frac{|D_1|}{D}.G(D_1) + \frac{|D_2|}{D}.G(D_2)$$

This binary partition of D reduces the impurity and the reduction in impurity is measured by

$$\gamma(A, D) = G(D) - G_A(D)$$

Gini Index of Diversity and CART

- This $\gamma(A, D)$ is called the Gini Index of diversity.
- It is also called as "impurity reduction".
- The attribute that maximizes the reduction in impurity (or equivalently, has the minimum value of $\hat{G}_A(D)$) is selected for the attribute to be splitted.

CART Algorithm : Illustration

Example 9.15 : CART Algorithm

Suppose we want to build decision tree for the data set EMP as given in the table below.

Age
Y : young
M : middle-aged
O : old

Salary
L : low
M : medium
H : high

Job
G : government
P : private

Performance
A : Average
E : Excellent

Class : Select
Y : yes
N : no

Tuple#	Age	Salary	Job	Performance	Select
1	Y	H	P	A	N
2	Y	H	P	E	N
3	M	H	P	A	Y
4	O	M	P	A	Y
5	O	L	G	A	Y
6	O	L	G	E	N
7	M	L	G	E	Y
8	Y	M	P	A	N
9	Y	L	G	A	Y
10	O	M	G	A	Y
11	Y	M	G	E	Y
12	M	M	P	E	Y
13	M	H	G	A	Y
14	O	M	P	E	N

CART Algorithm : Illustration

For the EMP data set,

$$G(EMP) = 1 - \sum_{i=1}^{2} p_i^2$$

$$= 1 - \left[\left(\frac{9}{14}\right)^2 + \left(\frac{5}{14}\right)^2\right]$$

$$= 0.4592$$

Now let us consider the calculation of for **Age**, **Salary**, **Job** and **Performance**.

Now let us consider the calculation of $G_A(EMP)$ for **Age**, **Salary**, **Job** and **Performance**.

CART Algorithm : Illustration

Attribute of splitting: Age

The attribute age has three values, namely Y, M and O. So there are 6 subsets, that should be considered for splitting as:

$$\underset{age_1}{\{Y\}},\ \underset{age_2}{\{M\}},\ \underset{age_3}{\{O\}},\ \underset{age_4}{\{Y,M\}},\ \underset{age_5}{\{Y,O\}},\ \underset{age_6}{\{M,O\}}$$

$$G_{age_1}(D) = \frac{5}{14} * \left(1 - \left(\frac{3}{5}\right)^2 - \left(\frac{2}{5}\right)^2\right) + \frac{9}{14}\left(1 - \left(\frac{6}{14}\right)^2 - \left(\frac{8}{14}\right)^2\right) = \mathbf{0.4862}$$

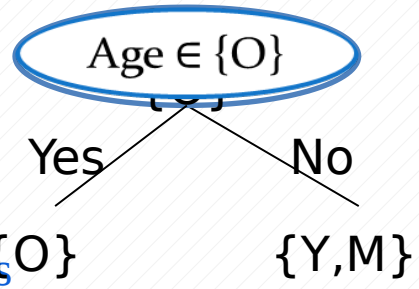
$G_{age_2}(D) = ?$

$G_{age_3}(D) = ?$

$G_{age_4}(D) = G_{age_3}(D)$

$G_{age_5}(D) = G_{age_2}(D)$

$G_{age_6}(D) = G_{age_1}(D)$



The best value of Gini Index while splitting attribute Age is

The best value of Gini Index while splitting attribute Age is $\gamma(Age_3, D) = \mathbf{0.3750}$

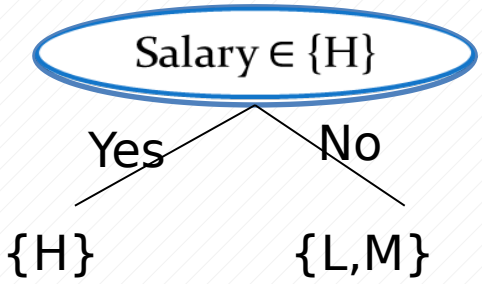
CART Algorithm : Illustration

Attribute of Splitting: Salary

The attribute salary has three values namely L, M and H. So, there are 6 subsets, that should be considered for splitting as:

$\{L\}$ sal_1' $\{M,H\}$ sal_2' $\{M\}$ sal_3' $\{L,H\}$ sal_4' $\{H\}$ sal_5' $\{L,M\}$ sal_6

$$G_{sal_1}(D) = G_{sal_2}(D) = 0.3000$$
$$G_{sal_3}(D) = G_{sal_4}(D) = 0.3150$$
$$G_{sal_5}(D) = G_{sal_6}(D) = 0.4508$$



$$\gamma(salary_{(5,6)}, D) = 0.4592 - 0.4508 = 0.0084$$

CART Algorithm : Illustration

Attribute of Splitting: job

Job being the binary attribute, we have

$$G_{job}(D) = \frac{7}{14} G(D_1) + \frac{7}{14} G(D_2)$$

?

$$= \frac{7}{14}\left[1 - \left(\frac{3}{7}\right)^2 - \left(\frac{4}{7}\right)^2\right] + \frac{7}{14}\left[1 - \left(\frac{6}{7}\right)^2 - \left(\frac{1}{7}\right)^2\right] = ?$$

?

$$\gamma(job, D) = ?$$

CART Algorithm : Illustration

Attribute of Splitting: Performance

Job being the binary attribute, we have

$G_{Performance}(D) = ?$

$\gamma(performance, D) = ?$

Out of these, gives the maximum value and hence, the attribute **Salary** would be chosen for splitting subset or.

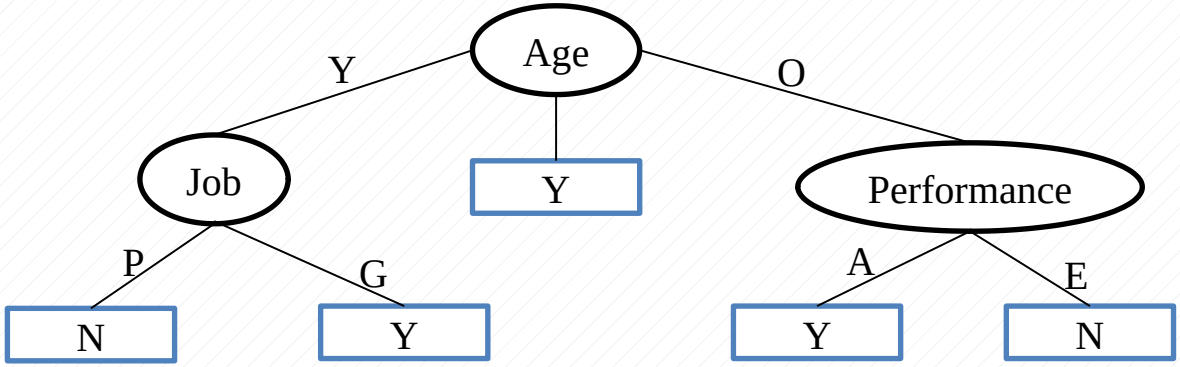
Note:

It can be noted that the procedure following "information gain" calculation (i.e.) and that of "impurity reduction" calculation (i.e.) are near about.

Decision Trees with ID3 and CART Algorithms

Example 9.17 : Comparing Decision Trees of EMP Data set

Compare two decision trees obtained using ID3 and CART for the EMP dataset. The decision tree according to ID3 is given for your ready reference (subject to the verification)



Decision Tree using ID3

?

Decision Tree using CART

Algorithm C4.5

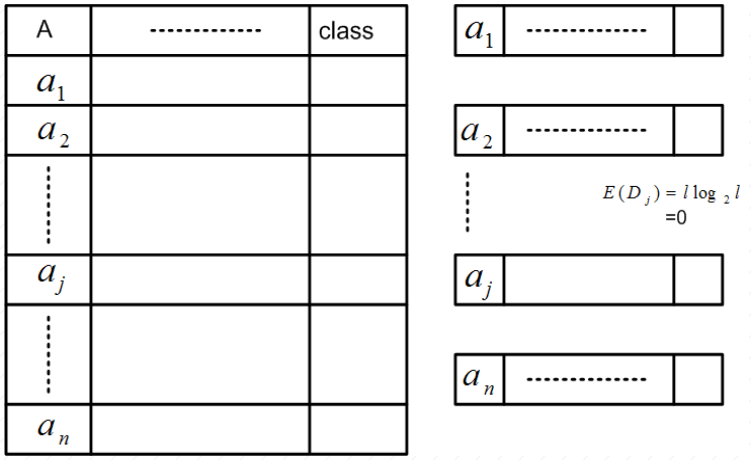
Algorithm C 4.5 : Introduction

- **J. Ross Quinlan**, a researcher in machine learning developed a decision tree induction algorithm in 1984 known as ID3 (Iterative Dichotomiser 3).
- Quinlan later presented C4.5, a successor of ID3, addressing some limitations in ID3.
- ID3 uses information gain measure, which is, in fact biased towards splitting attribute having a large number of outcomes.
- For example, if an attribute has distinct values for all tuples, then it would result in a large number of partitions, each one containing just one tuple.
  - In such a case, note that each partition is pure, and hence the purity measure of the partition, that is $E_A(D) = 0$
    - In such a case, note that each partition is pure, and hence the purity measure of the partition, that is

Algorithm C4.5 : Introduction

Example 9.18: Limitation of ID3

In the following, each tuple belongs to a unique class. The splitting on A is shown.



$$E_A(D) = \sum_{j=1}^{n} \frac{|D_j|}{|D|} . E(D_j) = \sum_{j=1}^{n} \frac{1}{|D|} . 0 = 0$$

Thus, $gain(A, D) = E(D) - E_A(D)$ is maximum in such a situation.

Algorithm: C 4.5 : Introduction

- Although, the previous situation is an extreme case, intuitively, we can infer that ID3 favours splitting attributes having a large number of values
  - compared to other attributes, which have a less variations in their values.

- Such a partition appears to be useless for classification.

- This type of problem is called **overfitting problem**.

Note:

Decision Tree Induction Algorithm ID3 may suffer from overfitting problem.

Algorithm: C 4.5 : Introduction

- The overfitting problem in ID3 is due to the measurement of information gain.
- In order to reduce the effect of the use of the bias due to the use of information gain, C4.5 uses a different measure called **Gain Ratio**, denoted as β.
- Gain Ratio is a kind of normalization to information gain using a **split information**.

Algorithm: C 4.5 : Gain Ratio

Definition 9.8: **Gain Ratio**

The gain ratio can be defined as follows. We first define split information $E_A^*(D)$ as

$$E_A^*(D) = -\sum_{j=1}^{m} \frac{|D_j|}{|D|} . \log \frac{|D_j|}{|D|}$$

Here, m is the number of distinct values in A.

The gain ratio is then defined as $\beta(A, D) = \frac{\alpha(A,D)}{E_A^*(D)}$, where $\alpha(A, D)$ denotes the information gain on splitting the attribute A in the dataset D.

Physical Interpretation of $E_A^*(D)$

Split information $E_A^*(D)$

- The value of split information depends on
  - the number of (distinct) values an attribute has and
  - how uniformly those values are distributed.
- In other words, it represents the potential information generated by splitting a data set D into m partitions, corresponding to the m outcomes of on attribute A.
- Note that for each outcome, it considers the number of tuples having that outcome with respect to the total number of tuples in D.

Summary of Decision Tree Induction Algorithms

- We have learned the building of a decision tree given a training data.
  - **The decision tree is then used to classify a test data.**
- For a given training data *D*, the important task is to build the decision tree so that:
  - **All test data can be classified accurately**
  - **The tree is balanced and with as minimum depth as possible, thus the classification can be done at a faster rate.**
- In order to build a decision tree, several algorithms have been proposed. These algorithms differ from the chosen splitting criteria, so that they satisfy the above mentioned objectives as well as the decision tree can be induced with minimum time complexity. **We have studied three decision tree induction algorithms namely ID3, CART and C4.5. A summary of these three algorithms is presented in the following table**.

Table 11.6

Algorithm	Splitting Criteria	Remark
ID3	**Information Gain** $\alpha(A,D) = H(S) - E_A(D)$ Where $H(S)$ = Entropy of S (a measure of uncertainty) = $-\sum_{i=1}^{k} p_i \log_2 p_i$ where S is with set of k classes , , … , and = ; Here, is the set of tuples with class in S. (S) = Weighted average entropy when D is partitioned on the values of attribute A =) Here, m denotes the distinct values of attribute A.	• The algorithm calculates $\alpha(A_i,D)$ for all A_i in D and choose that attribute which has **maximum** $\alpha(A_i,D)$. • The algorithm can handle both categorical and numerical attributes. • It favors splitting those attributes, which has a large number of distinct values.

ID3 Information Gain $\alpha(A,D) = H(S) - E_A(D)$ Where $H(S)$ = Entropy of S (a measure of uncertainty) = $-\sum_{i=1}^{k} p_i \log_2 p_i$ where S is with set of k classes $E_A(S)$ = Weighted average entropy when D is partitioned on the values of attribute A = $\sum_{j=1}^{m} \frac{|S_j|}{|S|} E(S_j)$ Here, m denotes the distinct values of attribute A.
• The algorithm calculates $\alpha(A_i,D)$ for all A_i in D and choose that attribute which has **maximum** $\alpha(A_i,D)$.
• The algorithm can handle both categorical and numerical attributes.
• It favors splitting those attributes, which has a large number of distinct values.

Algorithm	Splitting Criteria	Remark
CART	**Gini Index** $\gamma(A,D) = G(D) - G_A(D)$ where $G(D)$ = Gini index (a measure of impurity) $= 1 - \sum_{i=1}^{k} p_i^2$ Here, $p_i = \frac{\lvert C_{i,D} \rvert}{\lvert D \rvert}$ and D is with k number of classes $G_A(D) = \frac{\lvert D_1 \rvert}{\lvert D \rvert} G(D_1) + \frac{\lvert D_2 \rvert}{\lvert D \rvert} G(D_2)$, when D is partitioned into two data sets D_1 and D_2 based on some values of attribute A.	• The algorithm calculates all binary partitions for all possible values of attribute A and choose that binary partition which has the **maximum** $\gamma(A,D)$. • The algorithm is **computationally very expensive** when the attribute A has a large number of values.

Algorithm	Splitting Criteria	Remark
C4.5	Gain Ratio $\beta(A,D) = \frac{\alpha(A,D)}{E_A^*(D)}$ where $\alpha(A,D)$ = Information gain of D (same as in ID3, and $E_A^*(D)$ = splitting information $= -\sum_{j=1}^{m} \frac{\lvert D_j \rvert}{\lvert D \rvert} \log_2 \frac{\lvert D_j \rvert}{\lvert D \rvert}$ when D is partitioned into D_1, D_2, ... , D_m partitions corresponding to m distinct attribute values of A.	• The attribute A with maximum value of $\beta(A,D)$ is selected for splitting. • Splitting information is a kind of normalization, so that it can check the biasness of information gain towards the choosing attributes with a large number of distinct values.

Algorithm	Splitting Criteria	Remark
C4.5	Gain Ratio $\beta(A,D) = \frac{\alpha(A,D)}{E_A^*(D)}$ where $\alpha(A,D)$ = Information gain of D (same as in ID3, and $E_A^*(D)$ = splitting information $= -\sum_{j=1}^{m} \frac{\lvert D_j \rvert}{\lvert D \rvert} \log_2 \frac{\lvert D_j \rvert}{\lvert D \rvert}$ when D is partitioned into D_1, D_2, ... , D_m partitions corresponding to m distinct attribute values of A.	• The attribute A with maximum value of $\beta(A,D)$ is selected for splitting. • Splitting information is a kind of normalization, so that it can check the biasness of information gain towards the choosing attributes with a large number of distinct values.

In addition to this, we also highlight few important characteristics of decision tree induction algorithms in the following.

Notes on Decision Tree Induction algorithms

1. **Optimal Decision Tree:** Finding an optimal decision tree is an NP-complete problem. Hence, decision tree induction algorithms employ a heuristic based approach to search for the best in a large search space. Majority of the algorithms follow a greedy, top-down recursive divide-and-conquer strategy to build decision trees.

2. **Missing data and noise:** Decision tree induction algorithms are quite robust to **the data set with missing values and presence of noise**. However, proper data pre-processing can be followed to nullify these discrepancies.

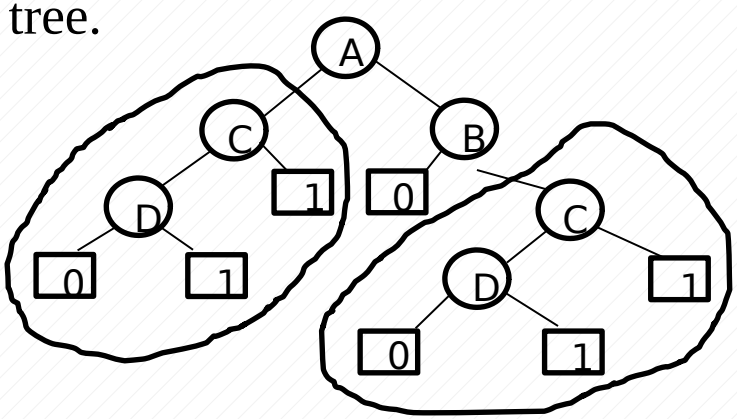
3. **Redundant Attributes:** The presence of **redundant attributes does not adversely affect the accuracy of decision trees**. It is observed that if an attribute is chosen for splitting, then another attribute which is redundant is unlikely to chosen for splitting.

4. **Computational complexity:** Decision tree induction algorithms are computationally inexpensive, **in particular, when the sizes of training sets are large, Moreover, once a decision tree is known, classifying a test record is extremely fast**, with a worst-case time complexity of O(d), where d is the maximum depth of the tree.

Notes on Decision Tree Induction algorithms

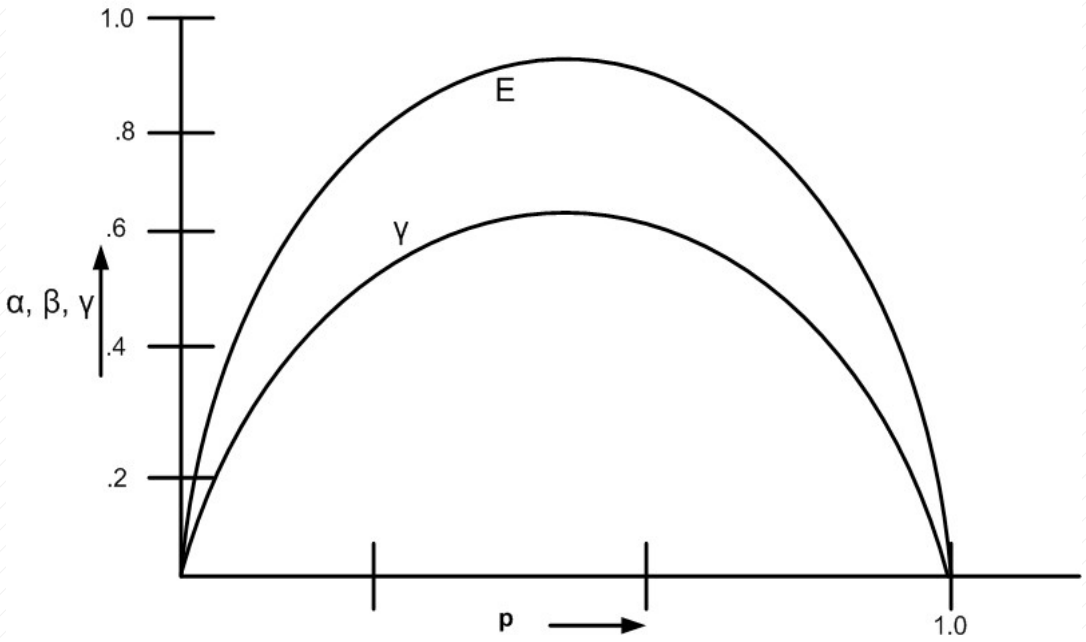
5. **Data Fragmentation Problem:** Since the decision tree induction algorithms employ a **top-down, recursive partitioning approach, the number of tuples becomes smaller as we traverse down the tree.** At a time, the number of tuples may be too small to make a decision about the class representation, such a problem is known as the data fragmentation. To deal with this problem, further splitting can be stopped when the number of records falls below a certain threshold.

6. **Tree Pruning:** A sub-tree can replicate two or more times in a decision tree (see figure below). This makes a decision tree unambiguous to classify a test record. To avoid such a sub-tree replication problem, all sub-trees except one can be pruned from the tree.



Notes on Decision Tree Induction algorithms

7. **Decision tree equivalence:** The different splitting criteria followed in different decision tree induction algorithms have little effect on the performance of the algorithms. This is because the different heuristic measures (such as information gain (α), Gini index (γ) and Gain ratio (β) are quite consistent with each other); also see the figure below.



Reference

- The detail material related to this lecture can be found in

Data Mining: Concepts and Techniques, (3rd Edn.), Jiawei Han, Micheline Kamber, Morgan Kaufmann, 2015.

Introduction to Data Mining, Pang-Ning Tan, Michael Steinbach, and Vipin Kumar, Addison-Wesley, 2014

Any question?

You may post your question(s) at the “Discussion Forum” maintained in the course Web page!