Artificial Intelligence

Learning from observations
Chapter 18, AIMA

Machine Learning

Two types of learning in AI

Deductive: Deduce new/interesting rules/facts from already known rules/facts.

$$\left(A \Rightarrow B \Rightarrow C\right) \Rightarrow \left(A \Rightarrow C\right)$$

We have been talking about this

Inductive: Learn new rules/facts from experience.

Experience can have various forms, one of the common approaches is to use a set of examples from the past D:

$$\mathcal{D} = \left\{\mathbf{x}(n), y(n)\right\}_{n=1\ldots N} \Rightarrow \left(A \Rightarrow C\right)$$

- Data mining

using historical data to improve decisions

- medical records → medical knowledge

- Software engineering

creating applications we are unable to program

autonomous driving

speech recognition

- Self-customising programs

adapting to a particular user/domain

news reader that learns user interests

Slide adapted from T. Mitchell

Learning Problem

Learning = improving with experience at some task

- Improve over task T
- With respect to performance measure P
- Based on experience E

Example:

- T: Decide upon next move in checkers
- P: % of games won in a tournament
- E: opportunity to play against self

Slide adapted from T. Mitchell

Three types of inductive learning

Supervised:

- The machine has access to a teacher who is able to provide the correct decisions for training examples.

active / passive learning

Reinforced:

- The machine is given feedback concerning the decision it makes, but no information about possible alternatives

Unsupervised:

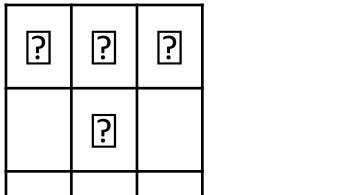
- No feedback is available, the machine must search for "order" and "structure" in the environment

Supervised Learning

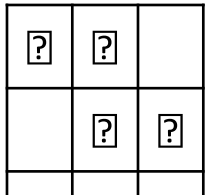
- Classification
  - learning categories (discriminative model)
  - choose between small number of alternatives
- mark news items as interesting/uninteresting
- diagnose diseases

- Regression
  - learning function values (generative model)
  - numerical output

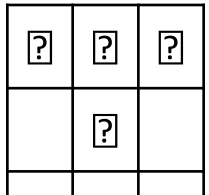
Inductive Learning Example



$$\mathbf{x} = \begin{pmatrix} -1 \\ -1 \\ +1 \\ 0 \\ +1 \\ 0 \\ +1 \\ 0 \\ 0 \end{pmatrix}, f(\mathbf{x}) = +1$$



$$\mathbf{x} = \begin{pmatrix} -1 \\ -1 \\ 0 \\ 0 \\ +1 \\ +1 \\ +1 \\ 0 \\ 0 \end{pmatrix}, f(\mathbf{x}) = -1$$



$$\mathbf{x} = \begin{pmatrix} -1 \\ -1 \\ 0 \\ 0 \\ +1 \\ +1 \\ 0 \\ +1 \\ 0 \end{pmatrix}, f(\mathbf{x}) = 0$$

Etc…

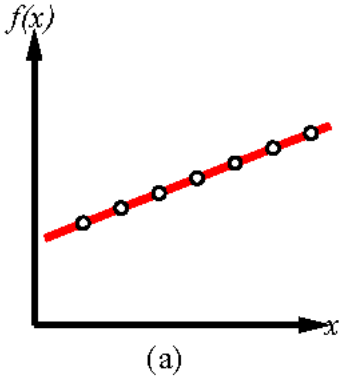
- $f(\mathbf{x})$ is the **target function**
- An **example** is a pair $[\mathbf{x}, f(\mathbf{x})]$
- Learning task: find a **hypothesis** h such that $h(\mathbf{x}) \approx f(\mathbf{x})$

based on a training set of examples $\mathcal{D} = \{[\mathbf{x}_i, f(\mathbf{x}_i)]\}, i = 1,2,\ldots,N$

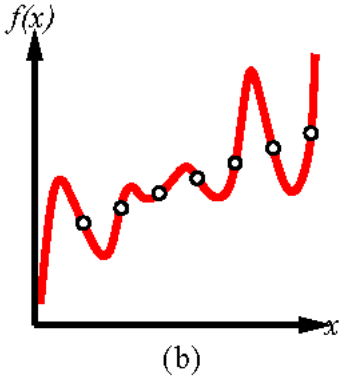
Inspired by a slide from V. Pavlovic

Inductive Learning Example

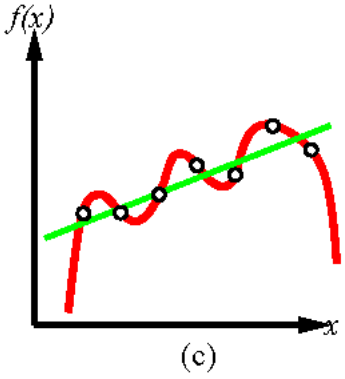
Consistent linear fit



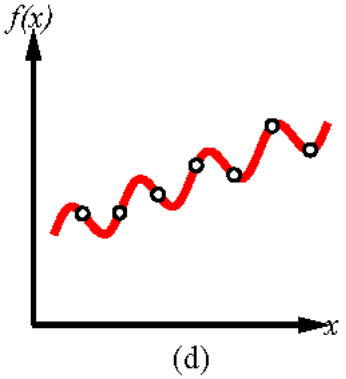
Consistent 7th order polynomial fit



Inconsistent linear fit.
Consistent 6th order polynomial fit.



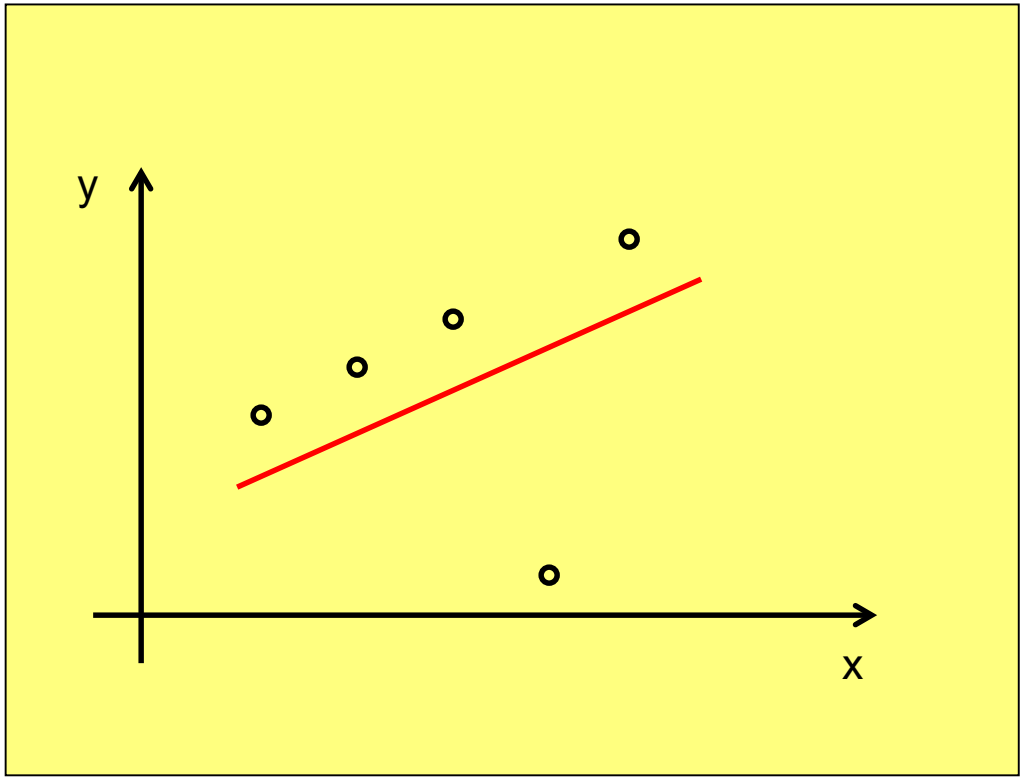
Consistent sinusoidal fit



- Construct h so that it agrees with f.
- The hypothesis h is consistent if it agrees with f on all observations.

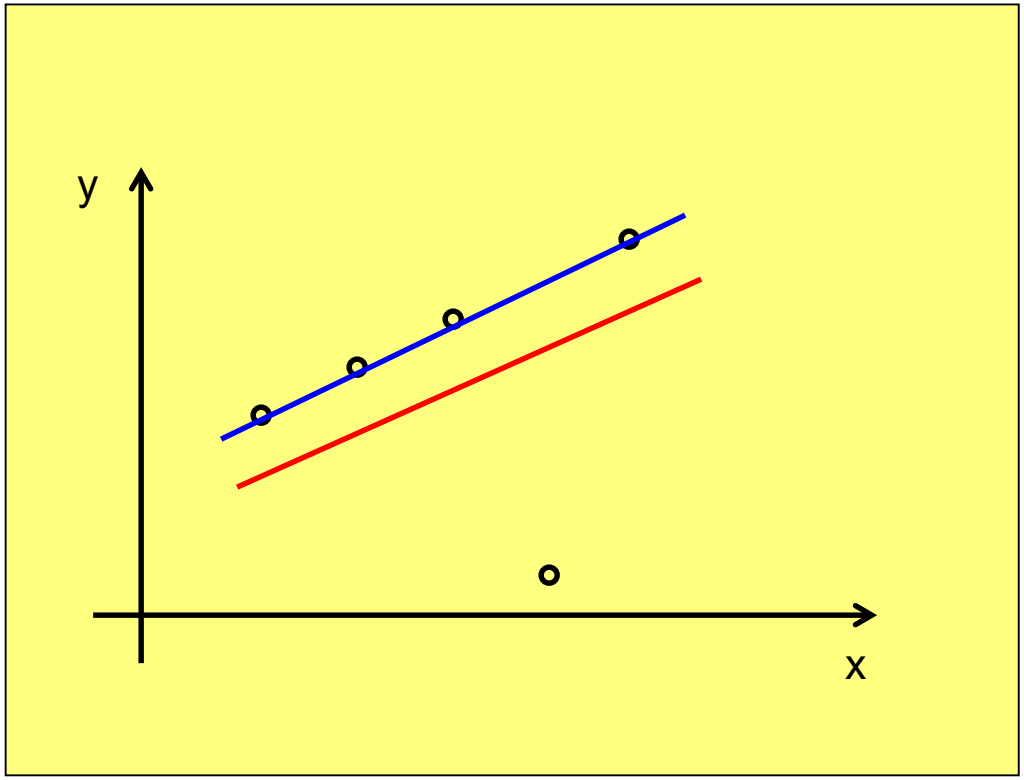
- How to achieve good generalization?
- Ockham's razor: Select the simplest consistent hypothesis.

Inductive Learning Example



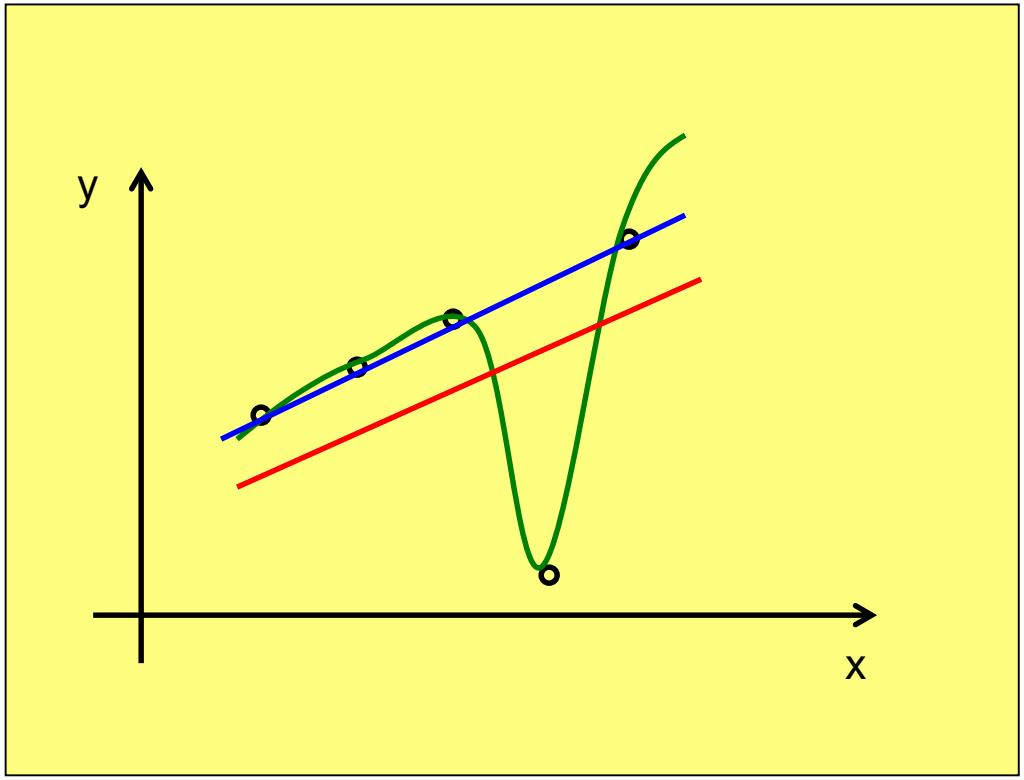
Example from V. Pavlovic @ Rutgers

Inductive learning – example C



Example from V. Pavlovic @ Rutgers

Inductive learning – example C

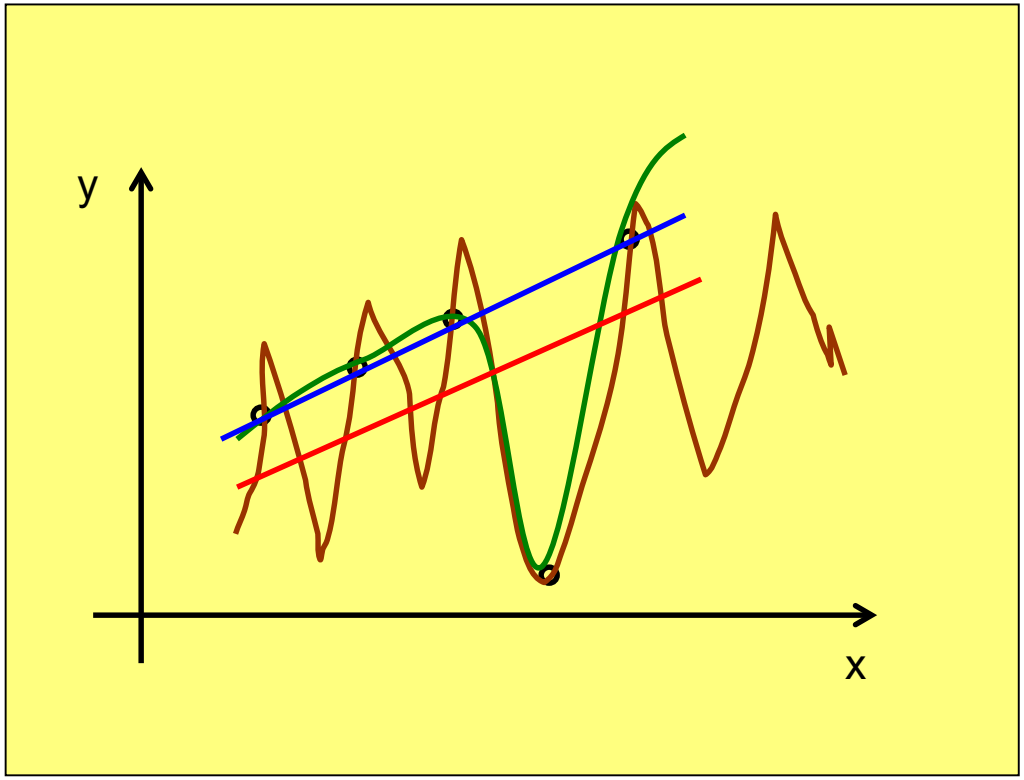


Example from V. Pavlovic @ Rutgers

Inductive learning – example C

Sometimes a consistent hypothesis is worse than an inconsistent one

overfitting



Example from V. Pavlovic @ Rutgers

Sky	Temp	Humid	Wind	Water	Forecst	EnjoySpt
Sunny	Warm	Normal	Strong	Warm	Same	Yes
Sunny	Warm	High	Strong	Warm	Same	Yes
Rainy	Cold	High	Strong	Warm	Change	No
Sunny	Warm	High	Strong	Cool	Change	Yes

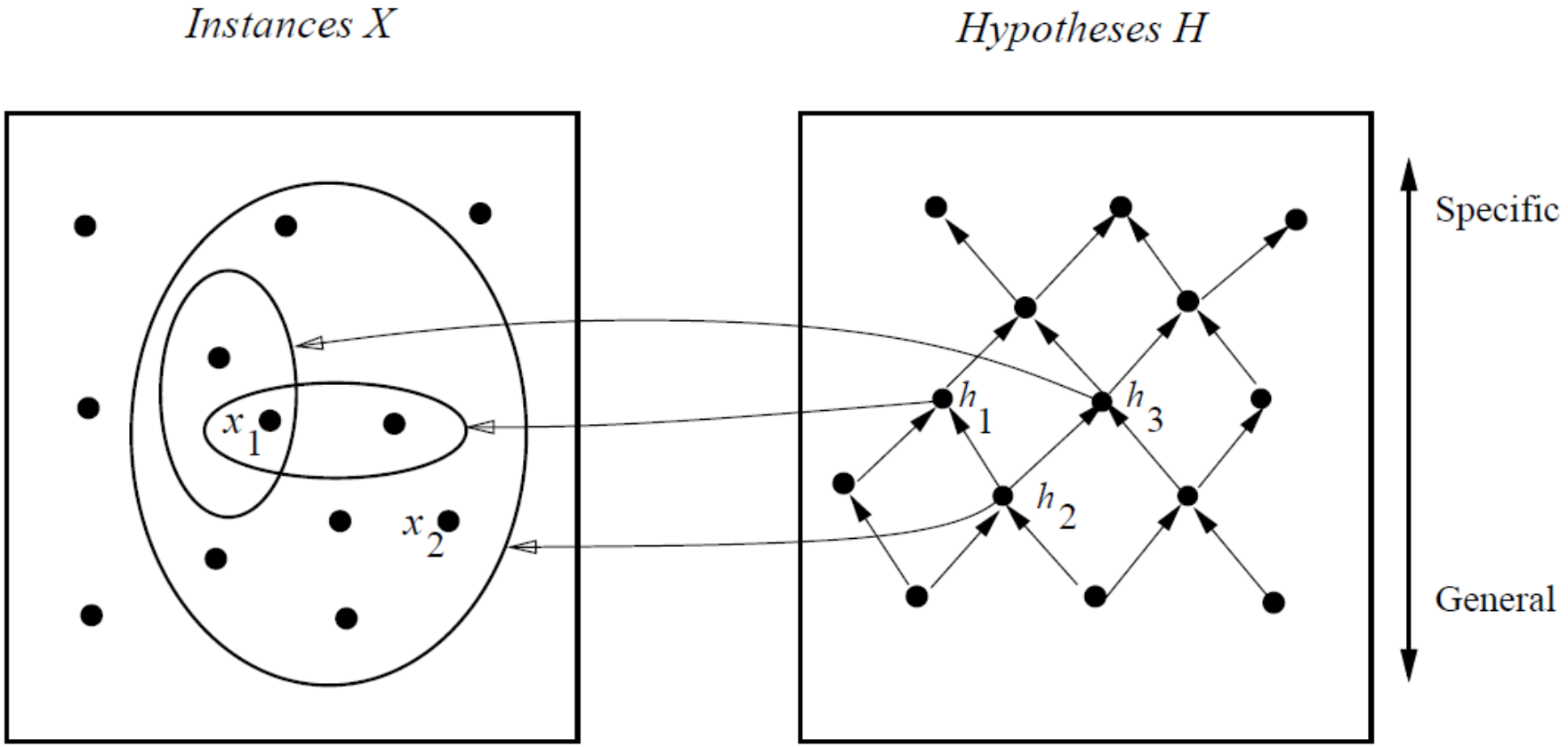
Target concept: *EnjoySport?*

How can we *represent* our *hypothesis*?

Conjunction of simple constraints on attributes:
a specific value *(Water=Warm)*
don't care *(Water=?)*
always false *(Water=Ø)*

<Sunny ? ? Strong ? Same> Yes?
No?

Slide adapted from T. Mitchell



x_1 = <Sunny, Warm, High, Strong, Cool, Same>

x_2 = <Sunny, Warm, High, Light, Warm, Same>

h_1 = <Sunny, ?, ?, Strong, ?, ?>

h_2 = <Sunny, ?, ?, ?, ?, ?>

h_3 = <Sunny, ?, ?, ?, Cool, ?>

Slide from T. Mitchell

Find-S Algorithm

(1) Initialize h to the most specific hypothesis in H

(2) For each positive training example x

For each attribute constraint a_i in h

(a) If the constraint a_i is satisfied by x

    do nothing

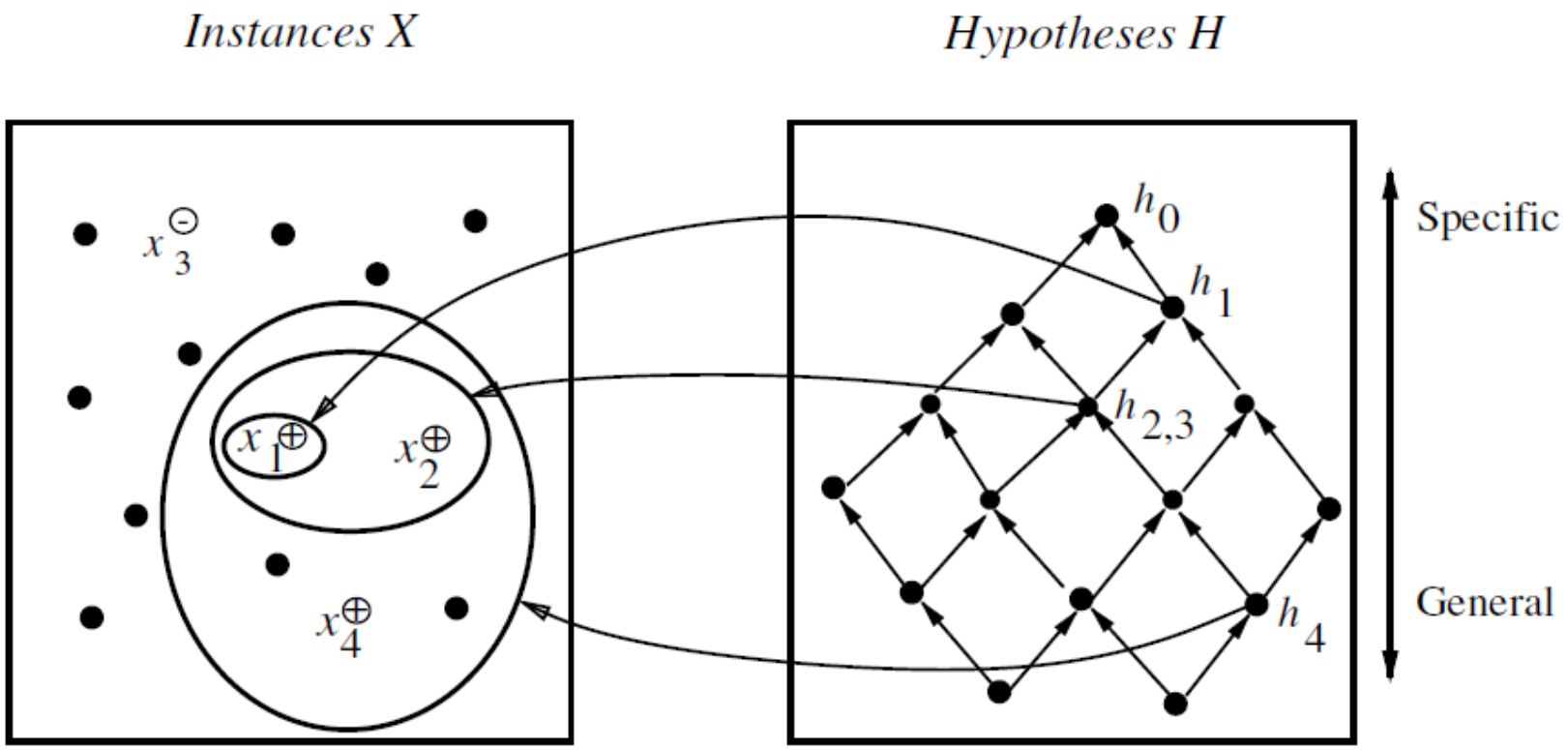
(a) Else

    replace a_i in h by the next more

    general constraint that is satisfied by x

(1) Output hypothesis h

Slide adapted from T. Mitchell



x_1 = <*Sunny Warm Normal Strong Warm Same*>, +

x_2 = <*Sunny Warm High Strong Warm Same*>, +

x_3 = <*Rainy Cold High Strong Warm Change*>, -

x_4 = <*Sunny Warm High Strong Cool Change*>, +

h_0 = <∅, ∅, ∅, ∅, ∅, ∅>

h_1 = <*Sunny Warm Normal Strong Warm Sam*

h_2 = <*Sunny Warm ? Strong Warm Same*>

h_3 = <*Sunny Warm ? Strong Warm Same*>

h_4 = <*Sunny Warm ? Strong ? ?* >

Slide from T. Mitchell

Problems

1. No idea how well the concept has been learned
   - do we need more training examples?

1. Cannot tell when training data is inconsistent
   - negative examples must be good for something

1. Picks maximally specific *h*
   - why is it better than any other one?
   - it is not even guaranteed to be unique

Slide adapted from T. Mitchell

Version Spaces

1. A hypothesis is consistent with a set of training examples *D* of target concept *c* iff *h(x)=c(x)* for each training example <*x,c(x)*> in *D*

$$Consistent(h, D) \equiv (\forall \langle x, c(x) \rangle \in D)\ h(x) = c(x)$$

1. The version space with respect to hypothesis space *H* and training examples *D*, VS_{HD}, is the subset of hypotheses from *H* that are consistent with all training examples in *D*

$$VS_{H,D} \equiv \{h \in H | Consistent(h, D)\}$$

Slide adapted from T. Mitchell

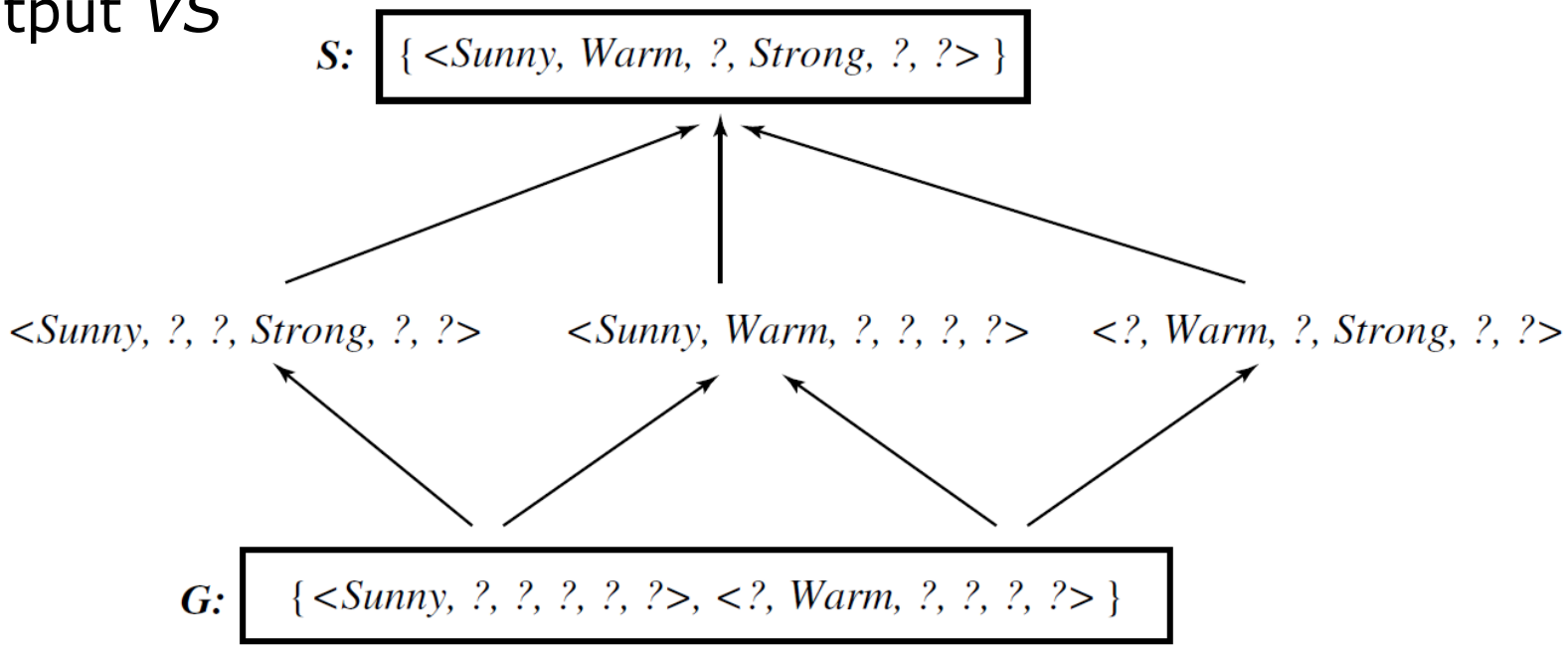
List-Then-Eliminate Algorithm

(1) Initialize *VS* = *H*

(2) For each training example <*x*,*c*(*x*)>

(3) remove from *VS* any hypothesis h for which

(4) Output *VS*



Slide adapted from T. Mitchell

Inductive Leap

	sky	temp	humid	wind	water	forecst
+	sunny	warm	normal	strong	cool	change
+	sunny	warm	normal	light	warm	same

S: <sunny warm normal ? ? ?>

What's the justification for this leap?

Why should we believe we can classify the unseen examples

<sunny warm normal strong warm same>

and

<sunny warm normal light cool change> ?

Slide adapted from T. Mitchell

An UNBIASED Learner

Choose *H* that is capable of expressing every teachable concept (i.e. *H* is the power set of *X*)

For example, allow disjunctions, conjunctions and negations over attribute constraints, e.g.

<sunny warm ? ? ? ?> $\vee$ <? ? ? ? ? ¬change>

+ <sunny warm normal strong cool change>

+ <sunny warm normal light warm same>

What is S and G?

Slide adapted from T. Mitchell

Inductive Bias

Consider

- concept learning algorithm L
- instances X, target concept c
- training examples $D_c = \{\langle x, c(x)\rangle\}$
- let $L(x_i, D_c)$ denote the classification assigned to the instance x_i by L after training on data D_c.

Definition:

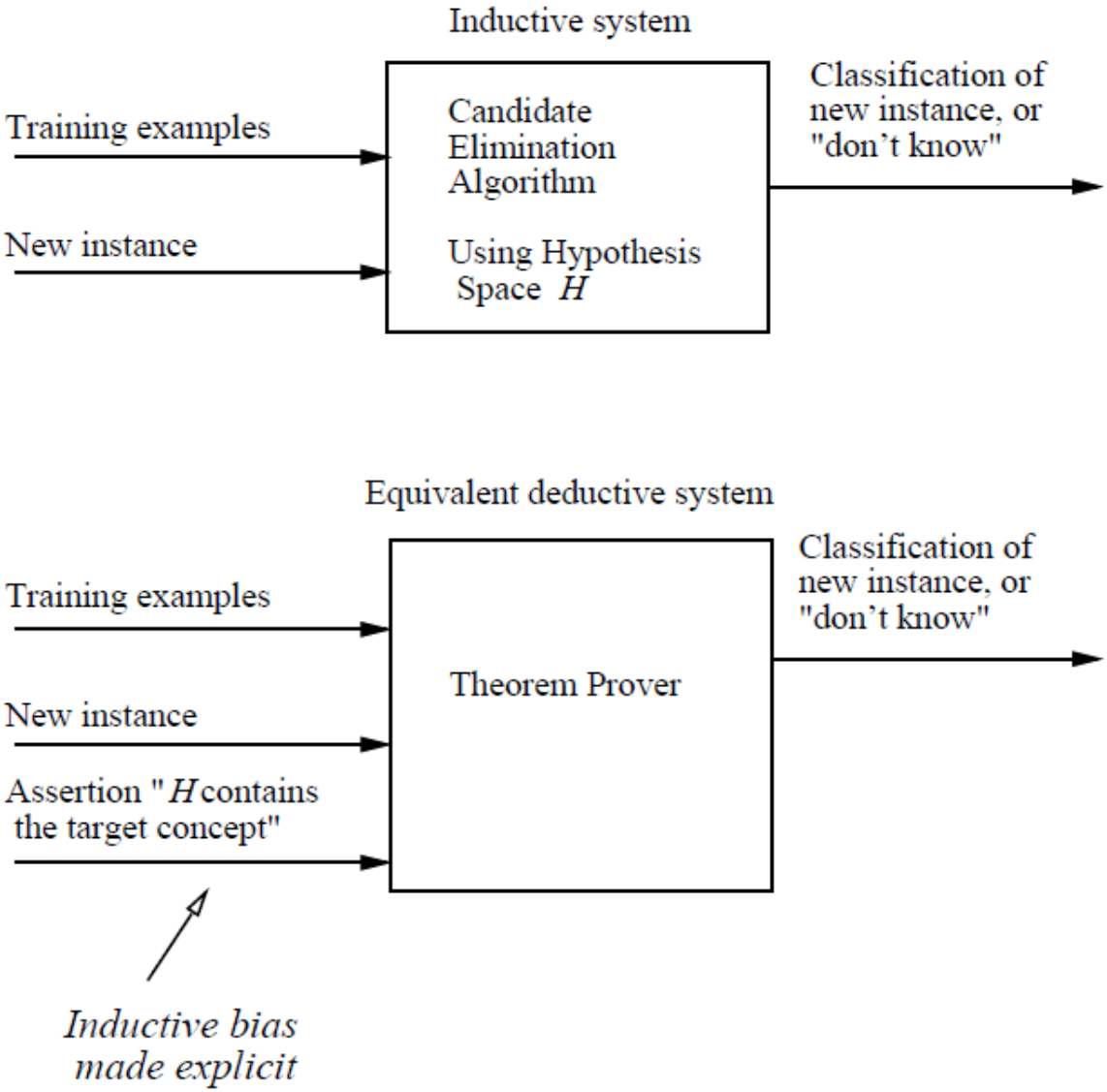
The **inductive bias** of L is any minimal set of assertions B such that for any target concept c and corresponding training examples D_c

$$(\forall x_i \in X)[(B \wedge D_c \wedge x_i) \vdash L(x_i, D_c)]$$

where $A \vdash B$ means A logically entails B

Slide from T. Mitchell

Inductive Bias



Slide from T. Mitchell

Learning problems

- The hypothesis takes a set of attribute values $\mathbf{x}$ as input
  - returns a "decision" $h(\mathbf{x})$
  - the predicted (estimated) output value
  - for the input $\mathbf{x}$.

- Discrete valued function $\Rightarrow$ classification
- Continuous valued function $\Rightarrow$ regression

Classification

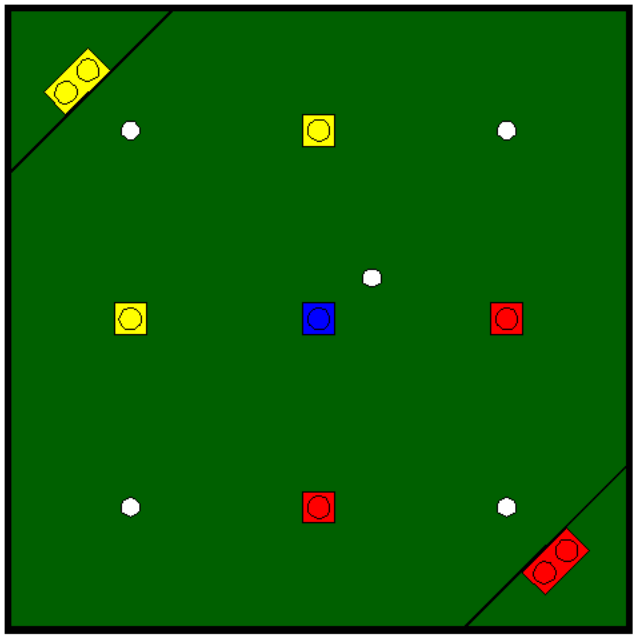
Order into one out of several classes

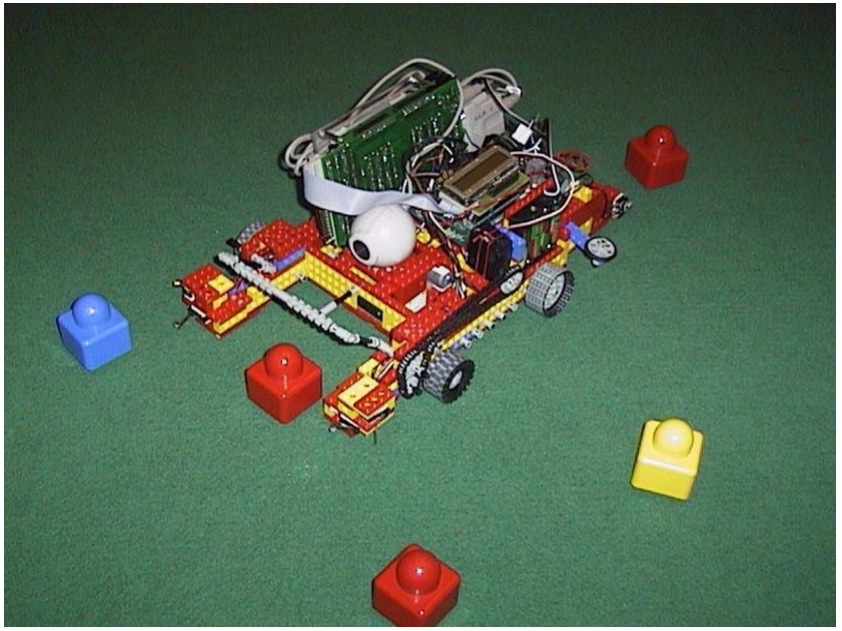
$$X^D \rightarrow C^K$$

Input space Output (category) space

$$\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_D \end{pmatrix} \in X^D \qquad \mathbf{c} = \begin{pmatrix} c_1 \\ c_2 \\ \vdots \\ c_K \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ \vdots \\ 0 \end{pmatrix} \in C^K$$

Example: Robot color vision, cancer detection, etc.





Classify the Lego pieces into *red*, *blue*, and *yellow.*
Classify *white* balls, *black* sideboard, and *green* carpet.
Input = pixel in image, output = category

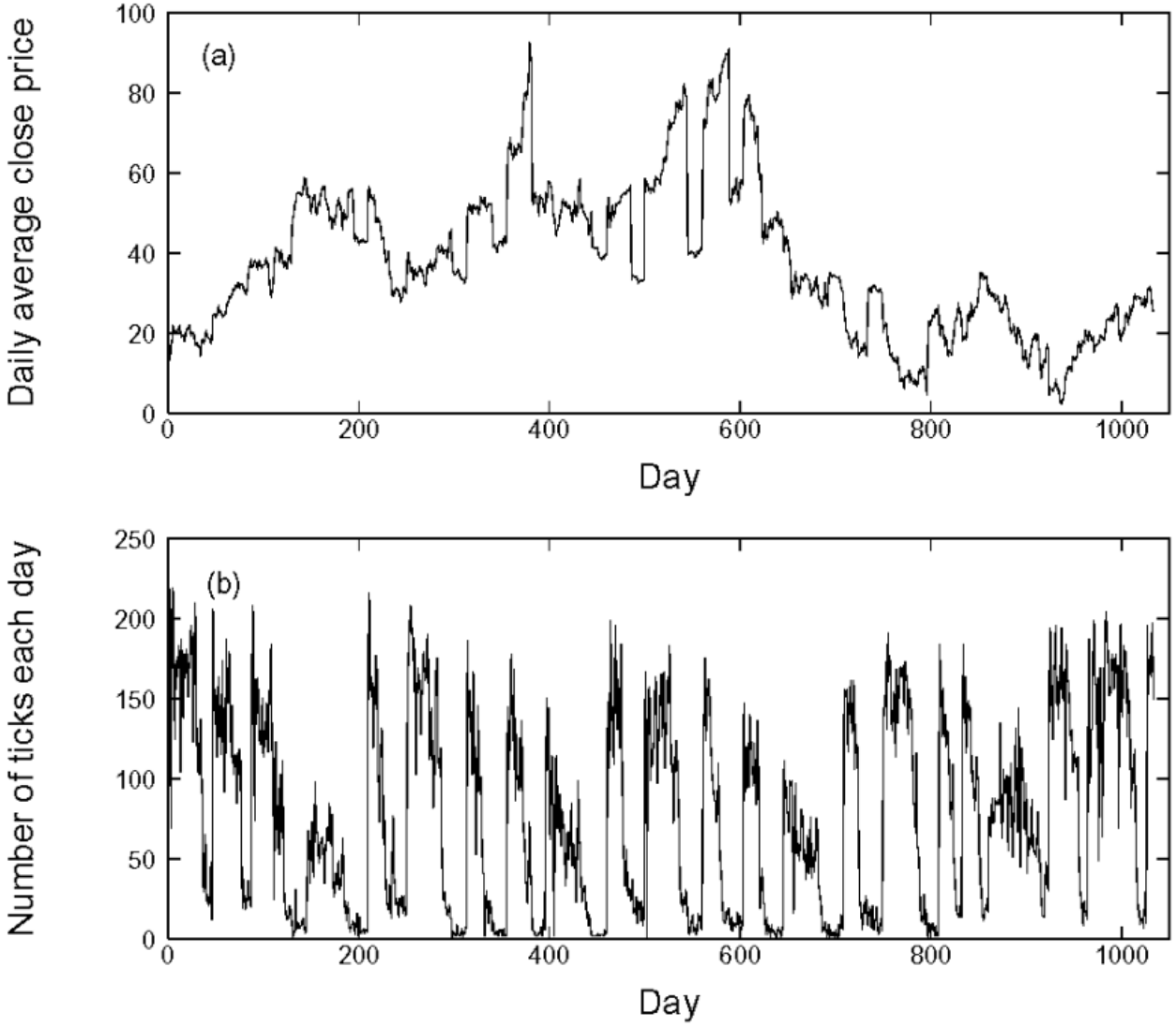
Regression

The “fixed regressor model”

$$f(\mathbf{x}) = g(\mathbf{x}) + \varepsilon$$

$\mathbf{x}$	Observed input
$f(\mathbf{x})$	Observed output
$g(\mathrm{x})$	True underlying function
ε	I.I.D noise process with zero mean

Example: Predict price for cotton futures

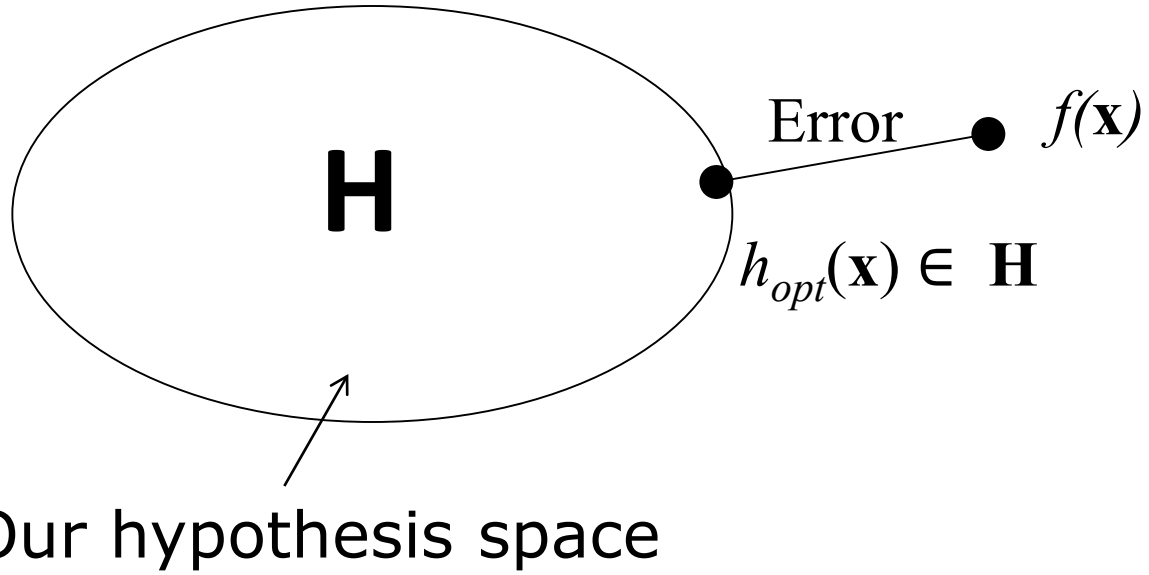


Input: Past history of closing prices, and trading volume

Output: Predicted closing price

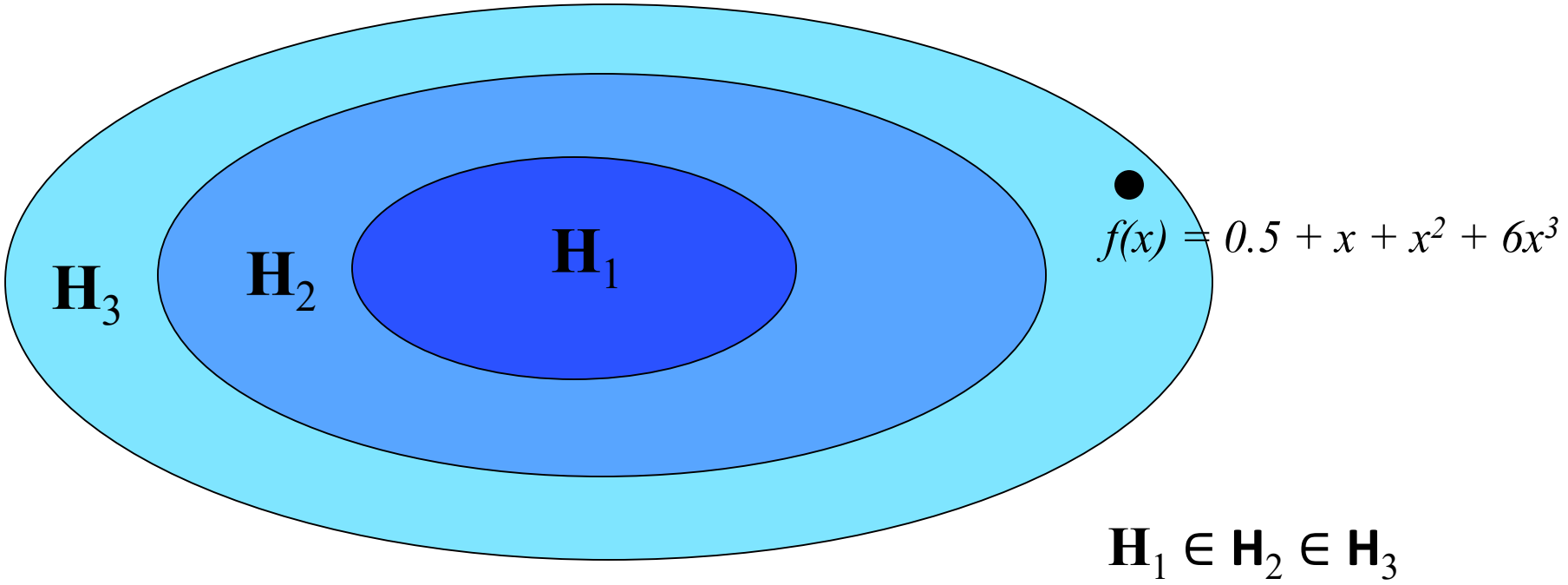
The idealized inductive learning problem

Find appropriate hypothesis space $\mathbf{H}$ and find $h(\mathbf{x}) \in \mathbf{H}$ with minimum "distance" to $f(\mathbf{x})$ ("error")



The learning problem is realizable if $f(\mathbf{x}) \in \mathbf{H}$.

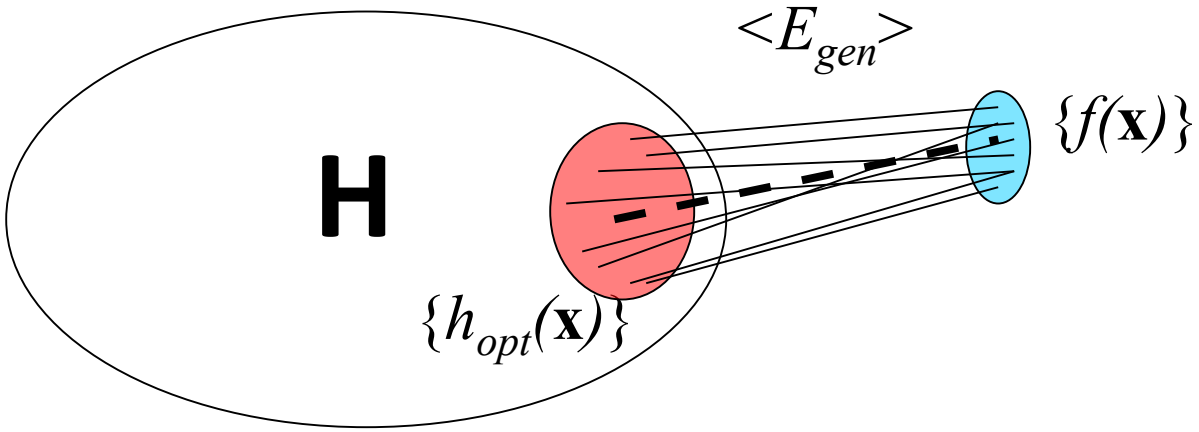
Hypothesis spaces (examples)



$\mathbf{H}_1 = \{a + bx\}$; $\mathbf{H}_2 = \{a + bx + cx^2\}$; $\mathbf{H}_3 = \{a + bx + cx^2 + dx^3\}$;
Linear; Quadratic; Cubic;

The real inductive learning problem

Find appropriate hypothesis space $\mathbf{H}$ and minimize the <u>expected</u> distance to $f(\mathbf{x})$ ("generalization error")



Data is never noise free and never available in infinite amounts, so we get variation in data and model.
The generalization error is a function of both the training data and the hypothesis selection method.

An example of classification Algorithm:

Decision Tree

What?

- Decision Trees (DTs) are a non-parametric supervised learning method used for classification.

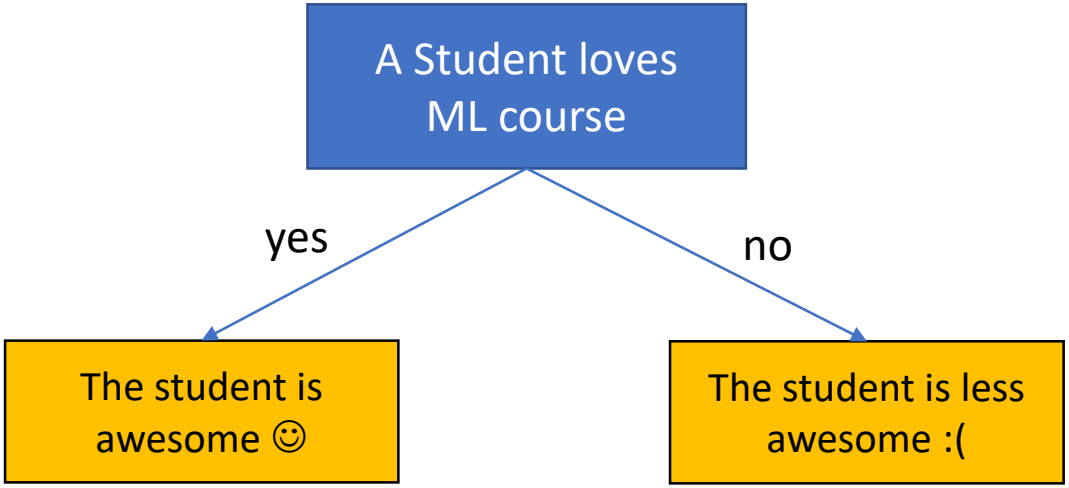
Why?

The key advantages of decision tree:

- Decision trees represent *rules*, which can be understood by humans and used in knowledge system such as database.
- Decision trees implicitly deploy feature selection or feature screening
- Decision trees need relatively little from users for data preparation
- Decision tree are not sensitive to nonlinear relationship between features

How it works?

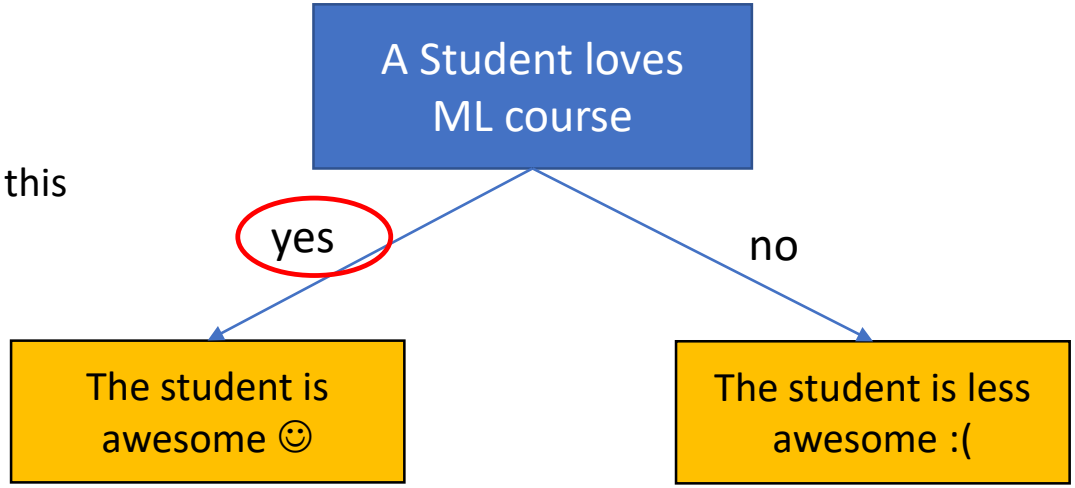
- A simple example of decision tree:



How it works?

- A simple example of decision tree:

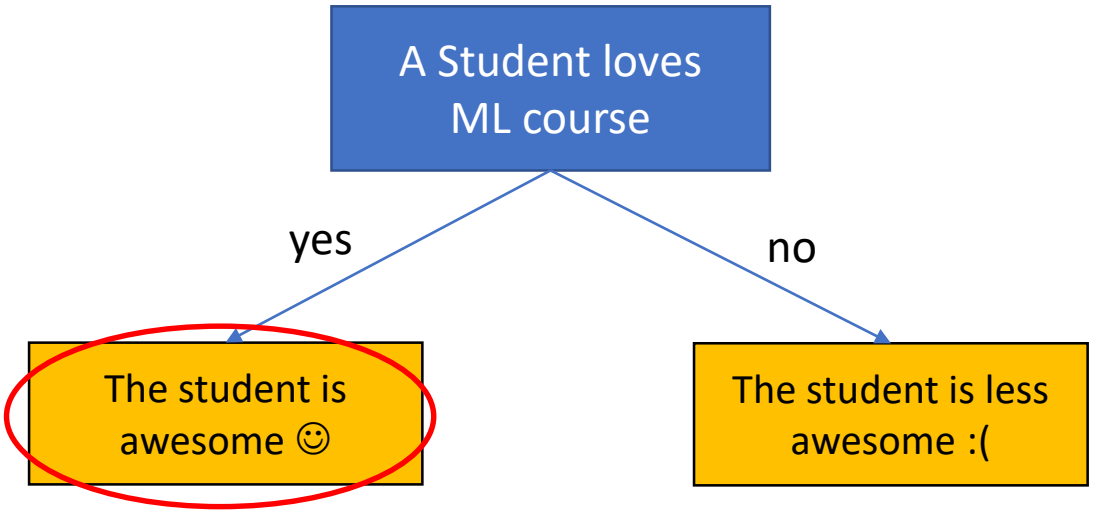
If a student loves this Course



How it works?

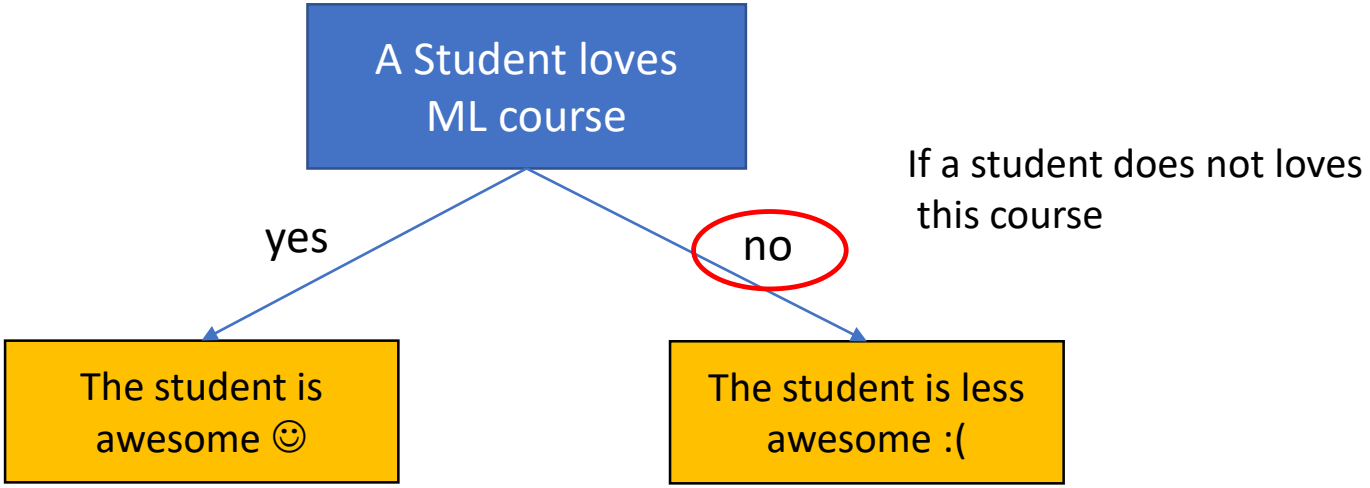
- A simple example of decision tree:

So, that student is awesome



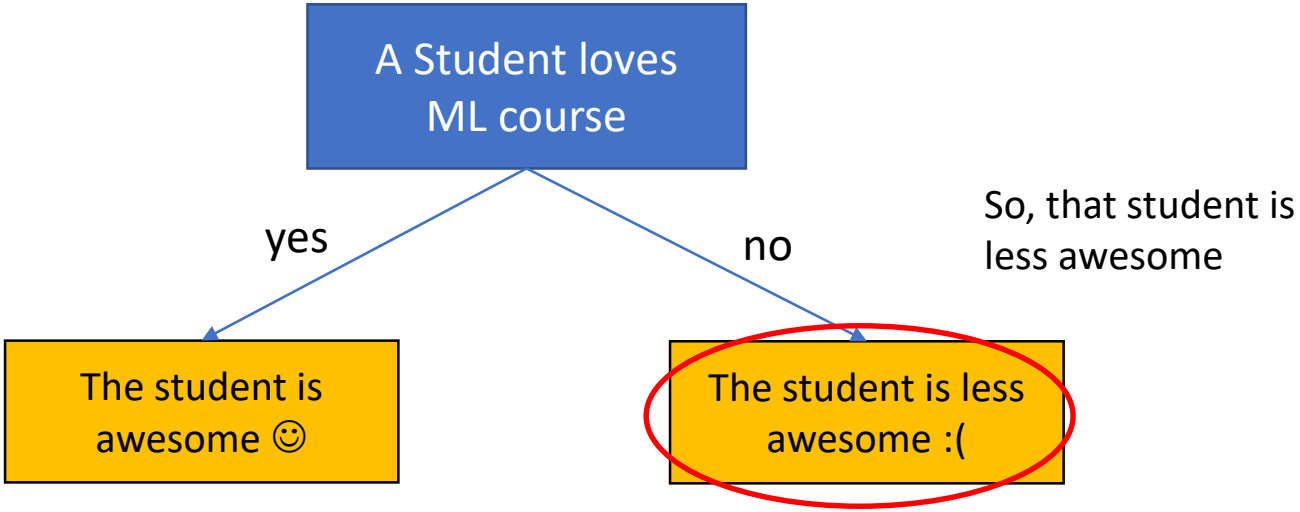
How it works?

- A simple example of decision tree:



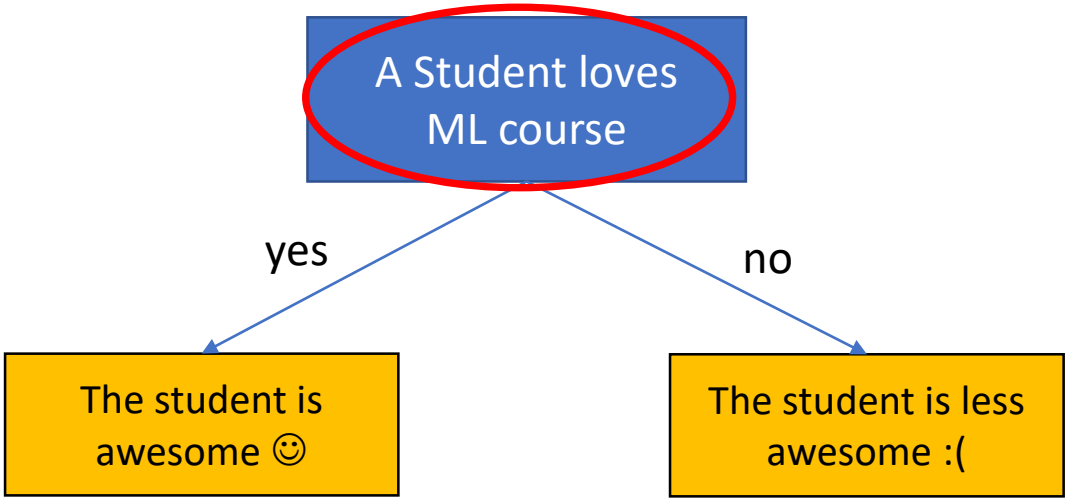
How it works?

- A simple example of decision tree:



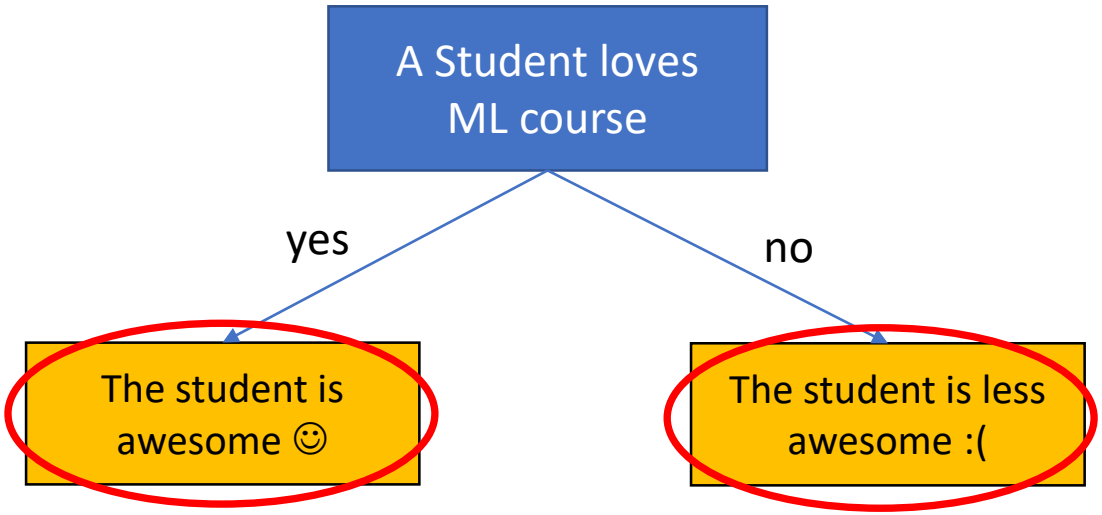
How it works?

- Generally, DT ask several question(s)



How it works?

- Generally, DT asks question(s)



- Then classifies the student w.r.t the given answers

How we can build DT from data?

Raw Date to DT?

We try to predict if Tom will play football given the new data (weather)

Day	Outlook	Humidity	Wind	Play
D1	Sunny	High	Weak	No
D2	Sunny	High	Strong	No
D3	**Overcast**	**High**	**Weak**	**Yes**
D4	**Rain**	**High**	**Weak**	**Yes**
D5	**Rain**	**Normal**	**Weak**	**Yes**
D6	Rain	Normal	Strong	No
D7	Overcast	Normal	Strong	Yes
D8	Sunny	High	Weak	No
D9	Sunny	Normal	Weak	Yes
D10	Rain	Normal	Weak	Yes
D11	Sunny	Normal	Strong	Yes
D12	Overcast	High	Strong	Yes
D13	Overcast	Normal	Weak	Yes
D14	Rain	High	Strong	No

- Training data contains:
  - 9 play and 5 not play

Raw Date to DT?

We try to predict if Tom will play football given the new data (weather)

Day	Outlook	Humidity	Wind	Play
D1	Sunny	High	Weak	No
D2	Sunny	High	Strong	No
D3	**Overcast**	**High**	**Weak**	**Yes**
D4	**Rain**	**High**	**Weak**	**Yes**
D5	**Rain**	**Normal**	**Weak**	**Yes**
D6	Rain	Normal	Strong	No
D7	Overcast	Normal	Strong	Yes
D8	Sunny	High	Weak	No
D9	Sunny	Normal	Weak	Yes
D10	Rain	Normal	Weak	Yes
D11	Sunny	Normal	Strong	Yes
D12	Overcast	High	Strong	Yes
D13	Overcast	Normal	Weak	Yes
D14	Rain	High	Strong	No
D15	Rain	High	Weak	?

- Training data contains:
  - 9 play and 5 not play
- New data:

D15 Rain High Weak ?

Raw Date to DT?

We try to predict if Tom will play football given the new data (weather)

Day	Outlook	Humidity	Wind	Play
D1	Sunny	High	Weak	No
D2	Sunny	High	Strong	No
D3	**Overcast**	**High**	**Weak**	**Yes**
D4	**Rain**	**High**	**Weak**	**Yes**
D5	**Rain**	**Normal**	**Weak**	**Yes**
D6	Rain	Normal	Strong	No
D7	Overcast	Normal	Strong	Yes
D8	Sunny	High	Weak	No
D9	Sunny	Normal	Weak	Yes
D10	Rain	Normal	Weak	Yes
D11	Sunny	Normal	Strong	Yes
D12	Overcast	High	Strong	Yes
D13	Overcast	Normal	Weak	Yes
D14	Rain	High	Strong	No
D15	Rain	High	Weak	?

- Training data contains:
  - 9 play and 5 not play
- New data:

D15 Rain High Weak ?

- Try to understand when Tom plays football
- Attribute selection and split into subsets
  - Stop when they are pure (all yes or no)
  - continue when they are not

Raw Date to DT?

We try to predict if Tom will play football given the new data (weather)

Day	Outlook	Humidity	Wind	Play
D1	Sunny	High	Weak	No
D2	Sunny	High	Strong	No
D3	**Overcast**	**High**	**Weak**	**Yes**
D4	**Rain**	**High**	**Weak**	**Yes**
D5	**Rain**	**Normal**	**Weak**	**Yes**
D6	Rain	Normal	Strong	No
D7	Overcast	Normal	Strong	Yes
D8	Sunny	High	Weak	No
D9	Sunny	Normal	Weak	Yes
D10	Rain	Normal	Weak	Yes
D11	Sunny	Normal	Strong	Yes
D12	Overcast	High	Strong	Yes
D13	Overcast	Normal	Weak	Yes
D14	Rain	High	Strong	No
D15	Rain	High	Weak	?

- Training data contains:
  - 9 play and 5 not play
- New data:

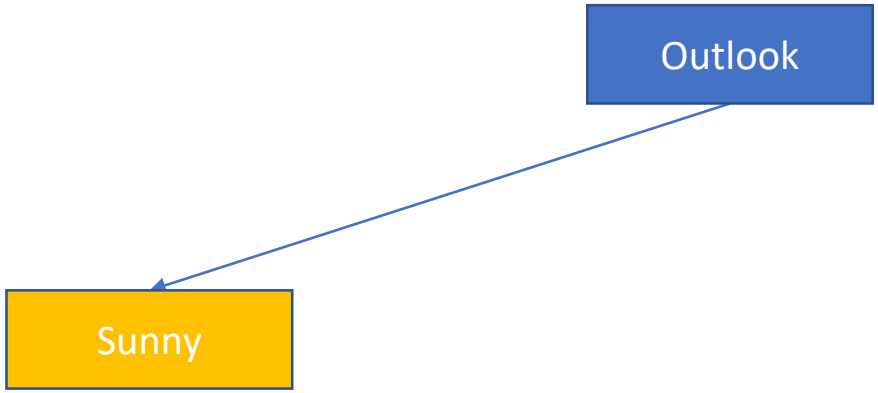
D15 Rain High Weak ?

- Try to understand when Tom plays football
- Attribute selection and split into subsets
- Stop when they are pure (all yes or no)
- continue when they are not

9 play: yes / 5 notplay: no

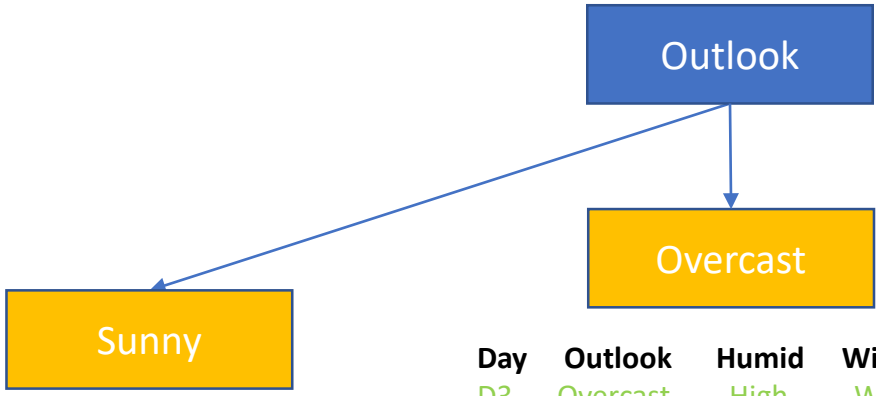
Outlook

9 play: yes / 5 notplay: no



Day	Outlook	Humid	Wind
D1	Sunny	High	Weak
D2	Sunny	High	Strong
D8	Sunny	High	Weak
D9	Sunny	Normal	Weak
D11	Sunny	Normal	Strong

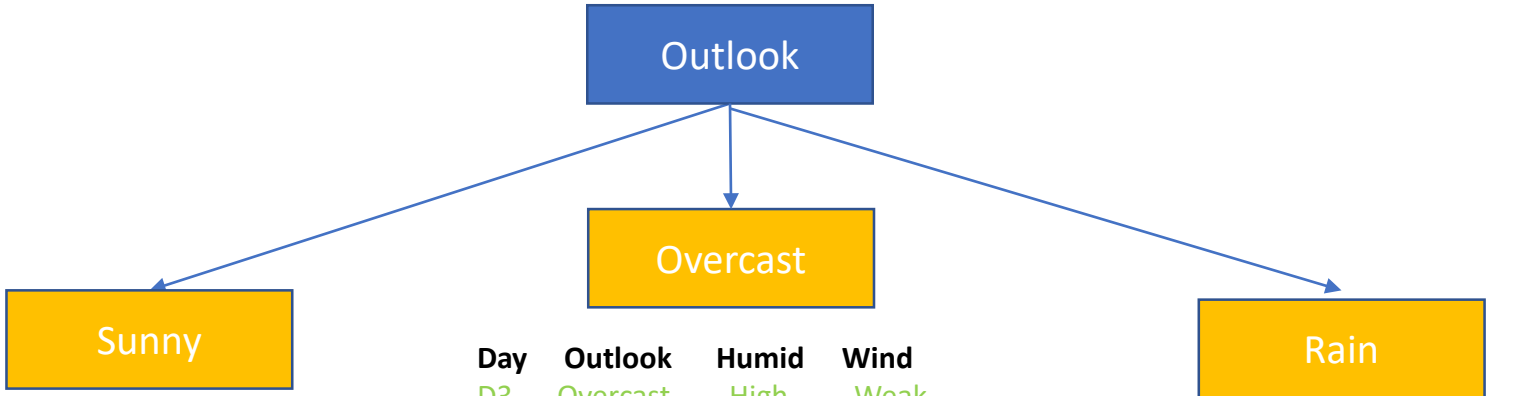
9 play: yes / 5 notplay: no



Day	Outlook	Humid	Wind
D3	Overcast	High	Weak
D7	Overcast	Normal	Strong
D12	Overcast	High	Srong
D13	Overcast	Normal	Weak

Day	Outlook	Humid	Wind
D1	Sunny	High	Weak
D2	Sunny	High	Strong
D8	Sunny	High	Weak
D9	Sunny	Normal	Weak
D11	Sunny	Normal	Strong

9 play: yes / 5 notplay: no

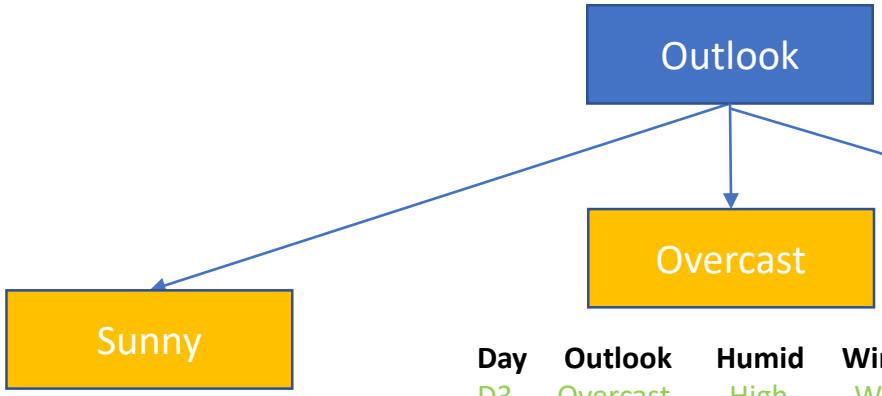


Day	Outlook	Humid	Wind
D1	Sunny	High	Weak
D2	Sunny	High	Strong
D8	Sunny	High	Weak
D9	Sunny	Normal	Weak
D11	Sunny	Normal	Strong

Day	Outlook	Humid	Wind
D3	Overcast	High	Weak
D7	Overcast	Normal	Strong
D12	Overcast	High	Srong
D13	Overcast	Normal	Weak

Day	Outlook	Humid	Wind
D4	Rain	High	Weak
D5	Rain	Normal	Weak
D6	Rain	Normal	Strong
D10	Rain	Normal	Weak
D14	Rain	High	Strong

9 play: yes / 5 notplay: no



Sunny

Day	Outlook	Humid	Wind
D1	Sunny	High	Weak
D2	Sunny	High	Strong
D8	Sunny	High	Weak
D9	Sunny	Normal	Weak
D11	Sunny	Normal	Strong

2 Yes and 3 No
We should split further

Overcast

Day	Outlook	Humid	Wind
D3	Overcast	High	Weak
D7	Overcast	Normal	Strong
D12	Overcast	High	Srong
D13	Overcast	Normal	Weak

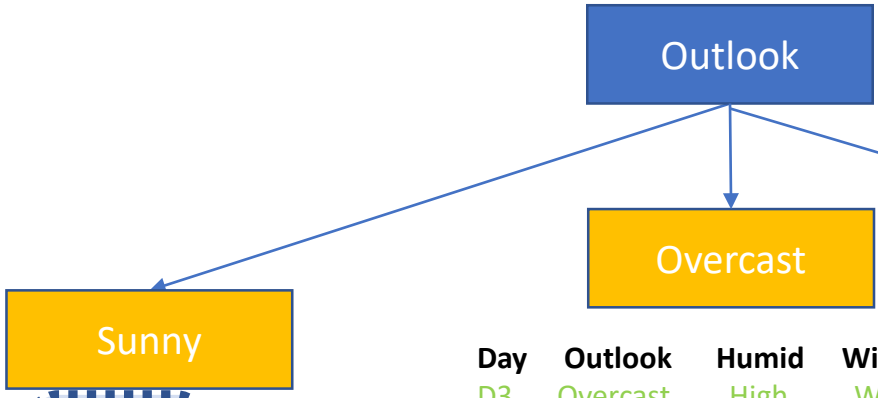
4 Yes and 0 No
Pure subset

Rain

Day	Outlook	Humid	Wind
D4	Rain	High	Weak
D5	Rain	Normal	Weak
D6	Rain	Normal	Strong
D10	Rain	Normal	Weak
D14	Rain	High	Strong

3 Yes and 2 No
We should split further

9 play: yes / 5 notplay: no



Day	Outlook	Humid	Wind
D1	Sunny	High	Weak
D2	Sunny	High	Strong
D8	Sunny	High	Weak
D9	Sunny	Normal	Weak
D11	Sunny	Normal	Strong

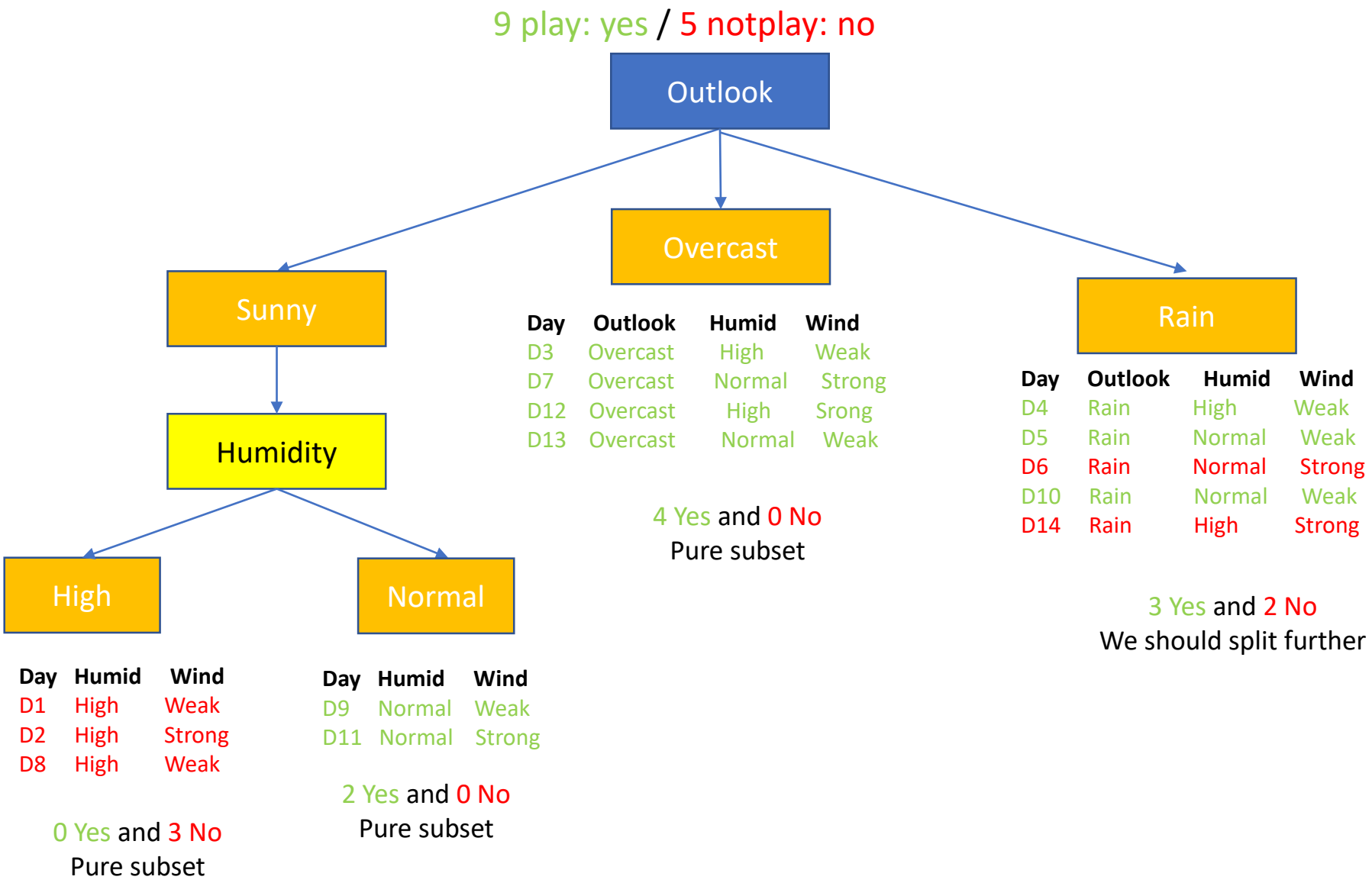
2 Yes and 3 No
We should split further

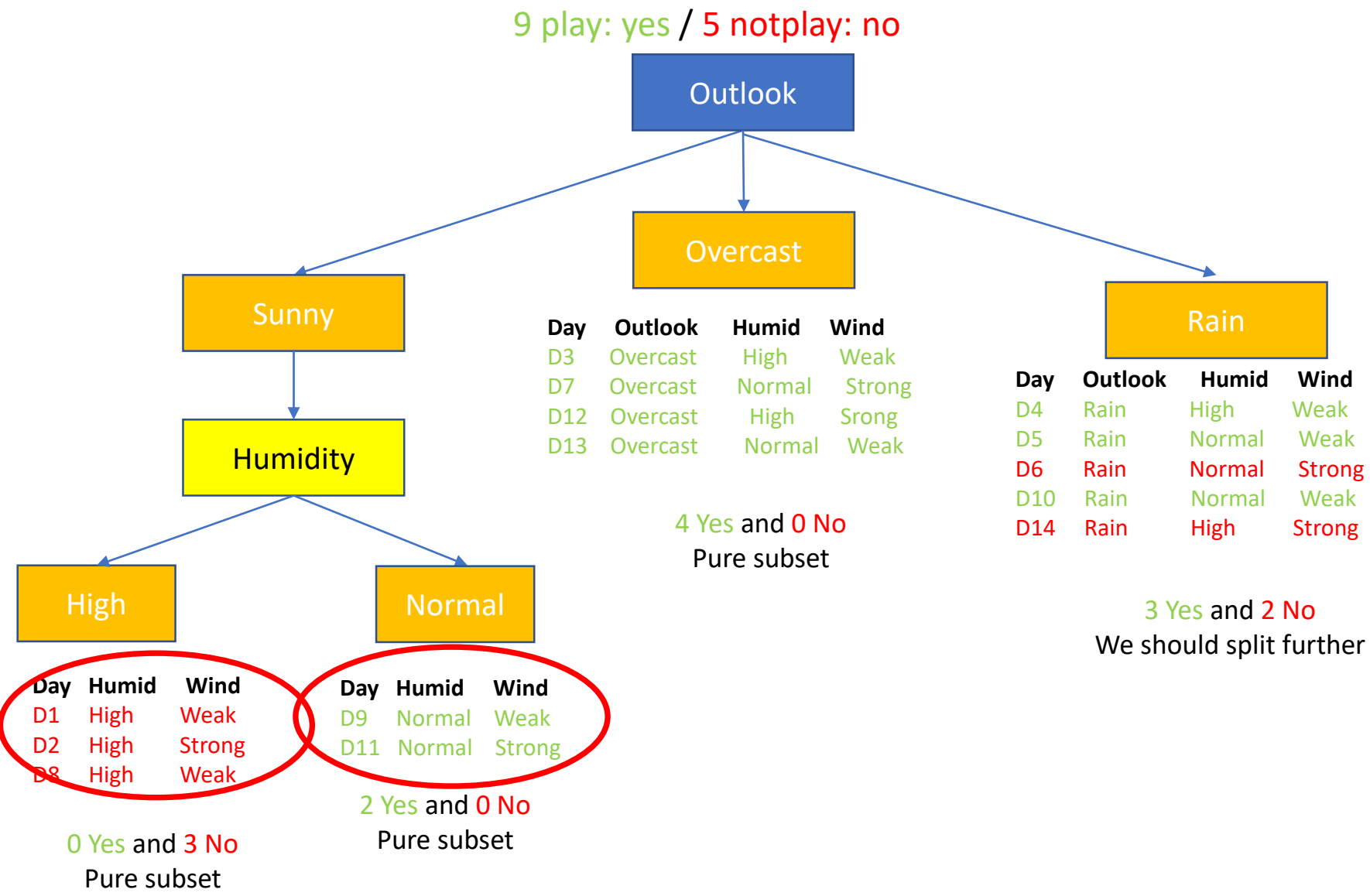
Day	Outlook	Humid	Wind
D3	Overcast	High	Weak
D7	Overcast	Normal	Strong
D12	Overcast	High	Srong
D13	Overcast	Normal	Weak

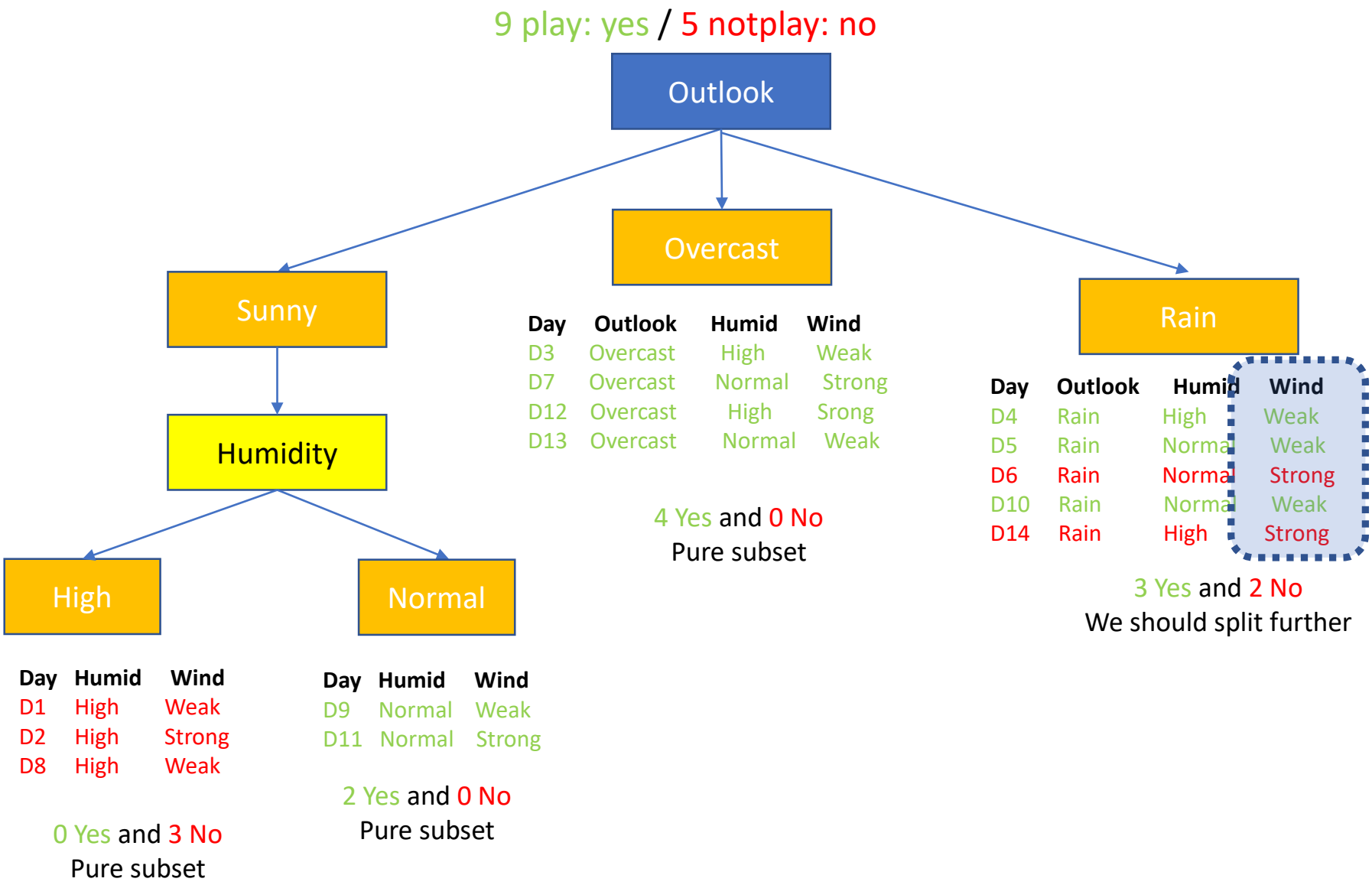
4 Yes and 0 No
Pure subset

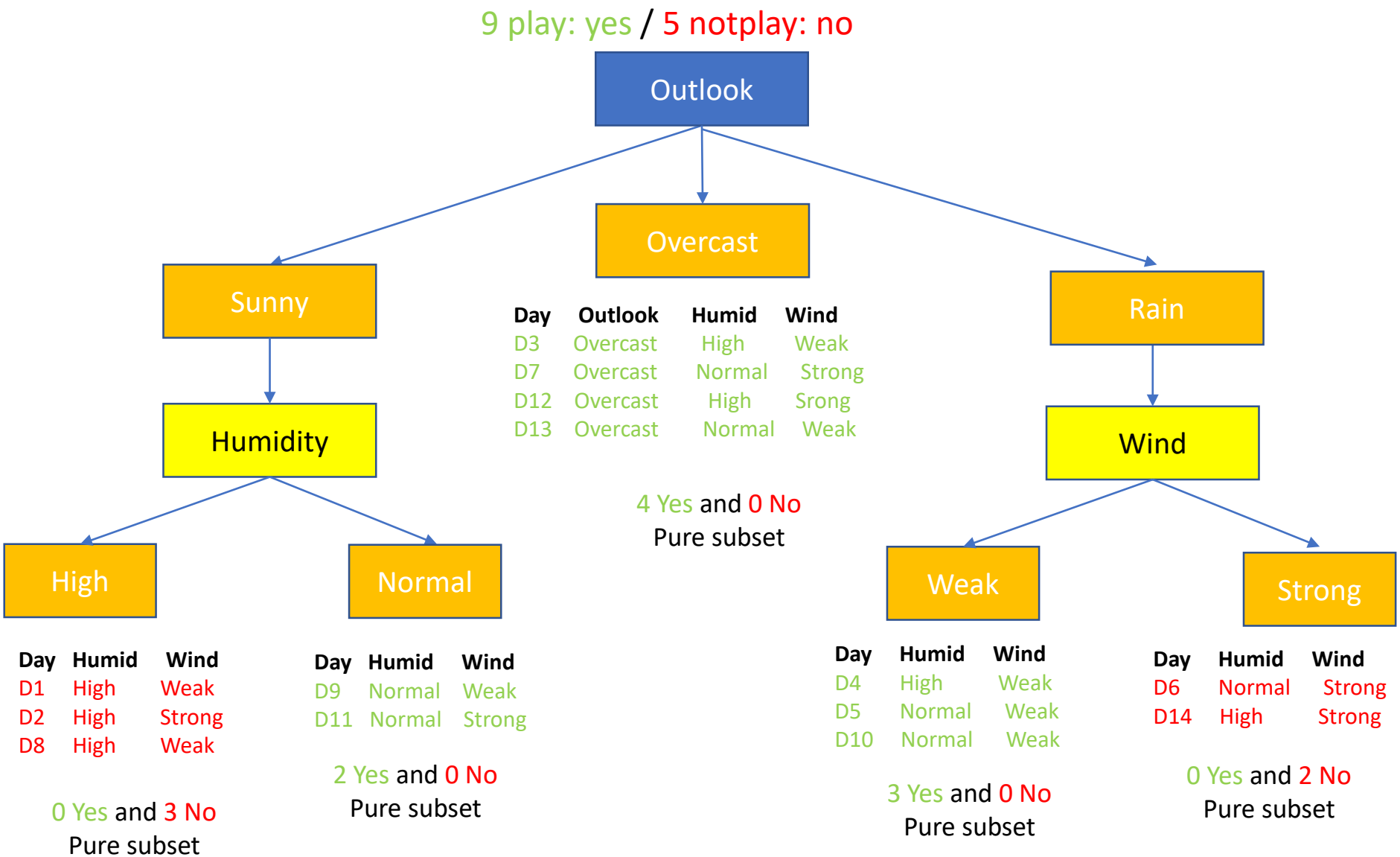
Day	Outlook	Humid	Wind
D4	Rain	High	Weak
D5	Rain	Normal	Weak
D6	Rain	Normal	Strong
D10	Rain	Normal	Weak
D14	Rain	High	Strong

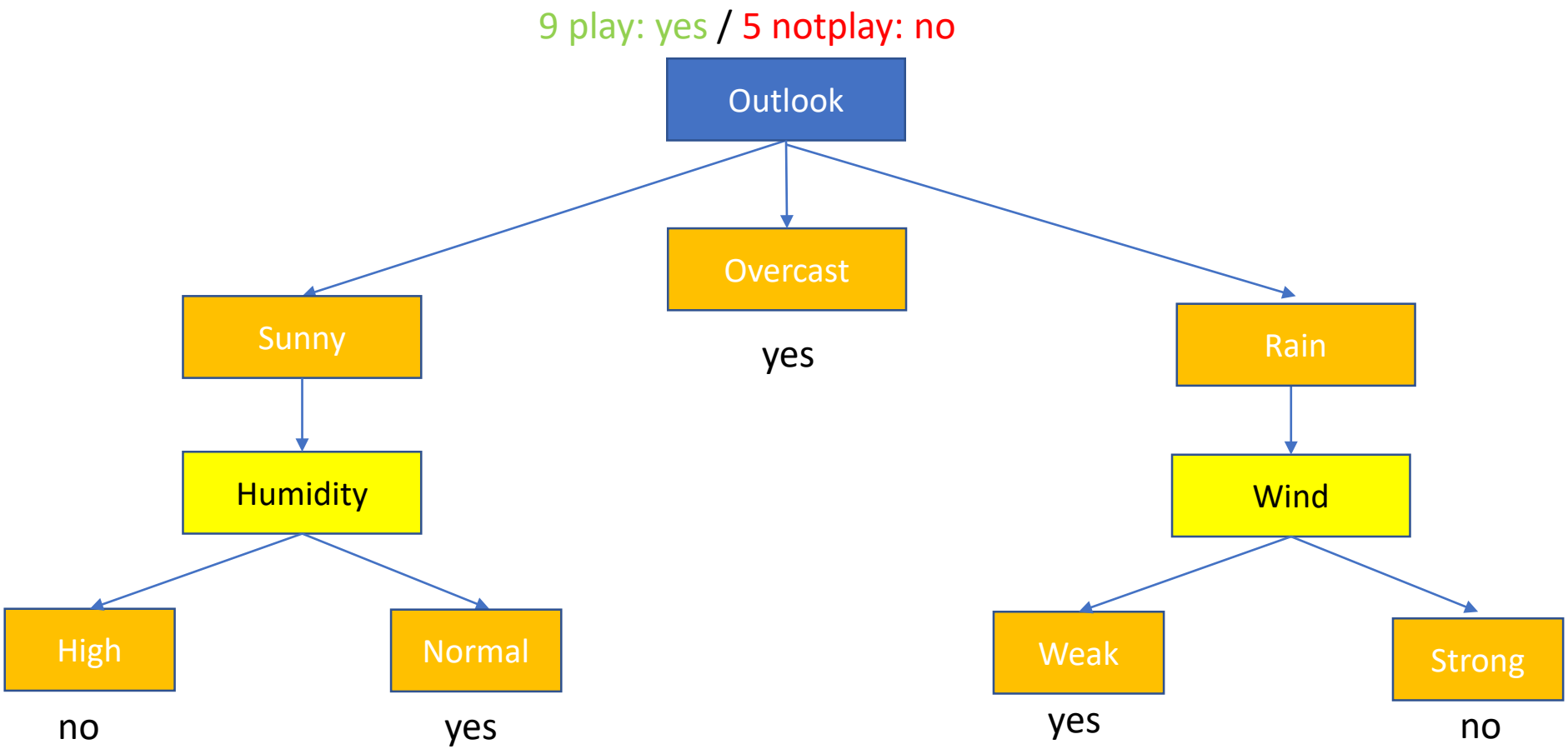
3 Yes and 2 No
We should split further

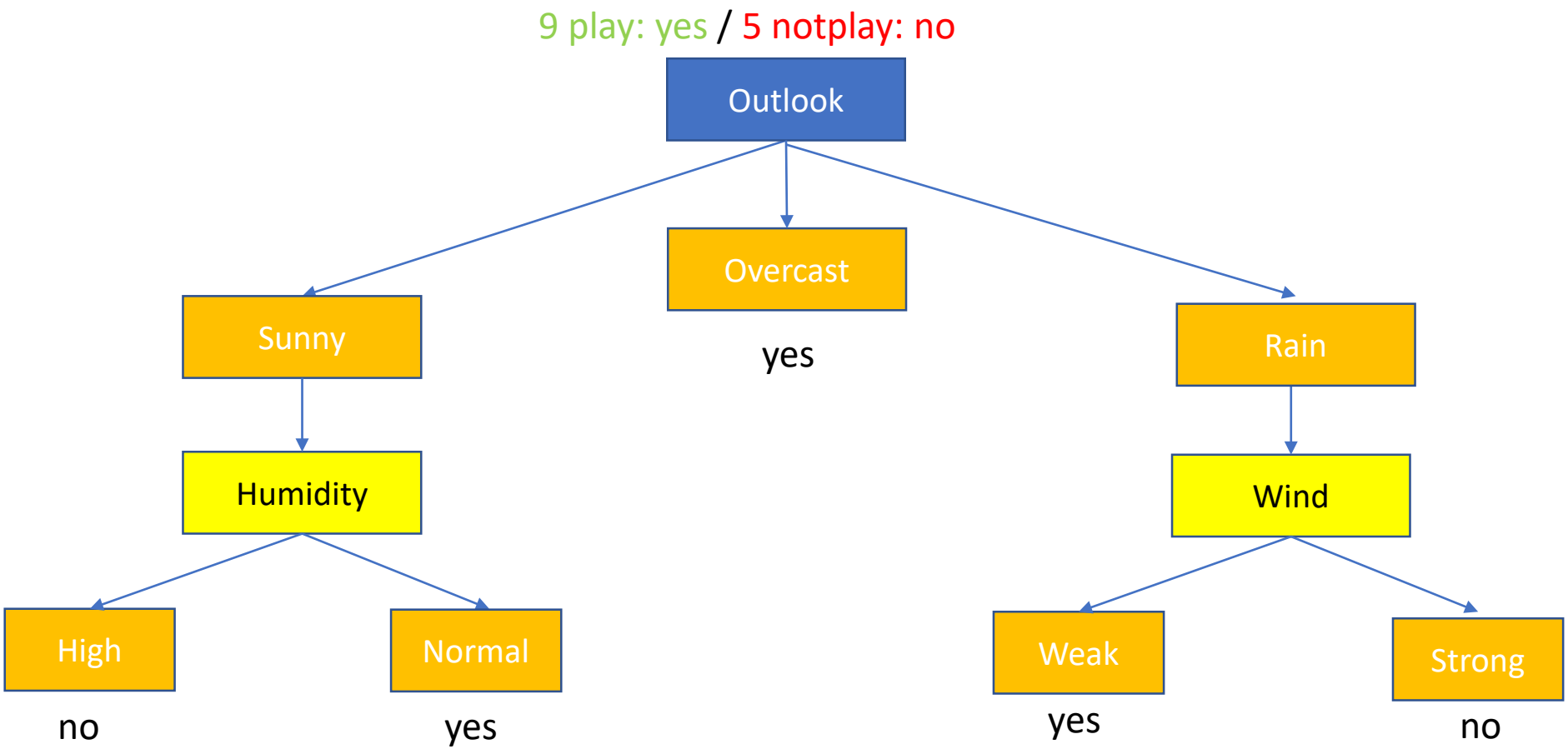

9 play: yes / 5 notplay: no
Outlook
Sunny
Overcast
Rain
Humidity
High
Normal
Day Outlook Humid Wind
D3 Overcast High Weak
D7 Overcast Normal Strong
D12 Overcast High Srong
D13 Overcast Normal Weak
4 Yes and 0 No
Pure subset
Day Outlook Humid Wind
D4 Rain High Weak
D5 Rain Normal Weak
D6 Rain Normal Strong
D10 Rain Normal Weak
D14 Rain High Strong
3 Yes and 2 No
We should split further
Day Humid Wind
D1 High Weak
D2 High Strong
D8 High Weak
0 Yes and 3 No
Pure subset
Day Humid Wind
D9 Normal Weak
D11 Normal Strong
2 Yes and 0 No
Pure subset


9 play: yes / 5 notplay: no
Outlook
Sunny
Overcast
Rain
Humidity
High
Normal
Day Outlook Humid Wind
D3 Overcast High Weak
D7 Overcast Normal Strong
D12 Overcast High Srong
D13 Overcast Normal Weak
4 Yes and 0 No
Pure subset
Day Outlook Humid Wind
D4 Rain High Weak
D5 Rain Normal Weak
D6 Rain Normal Strong
D10 Rain Normal Weak
D14 Rain High Strong
3 Yes and 2 No
We should split further
Day Humid Wind
D1 High Weak
D2 High Strong
D8 High Weak
0 Yes and 3 No
Pure subset
Day Humid Wind
D9 Normal Weak
D11 Normal Strong
2 Yes and 0 No
Pure subset

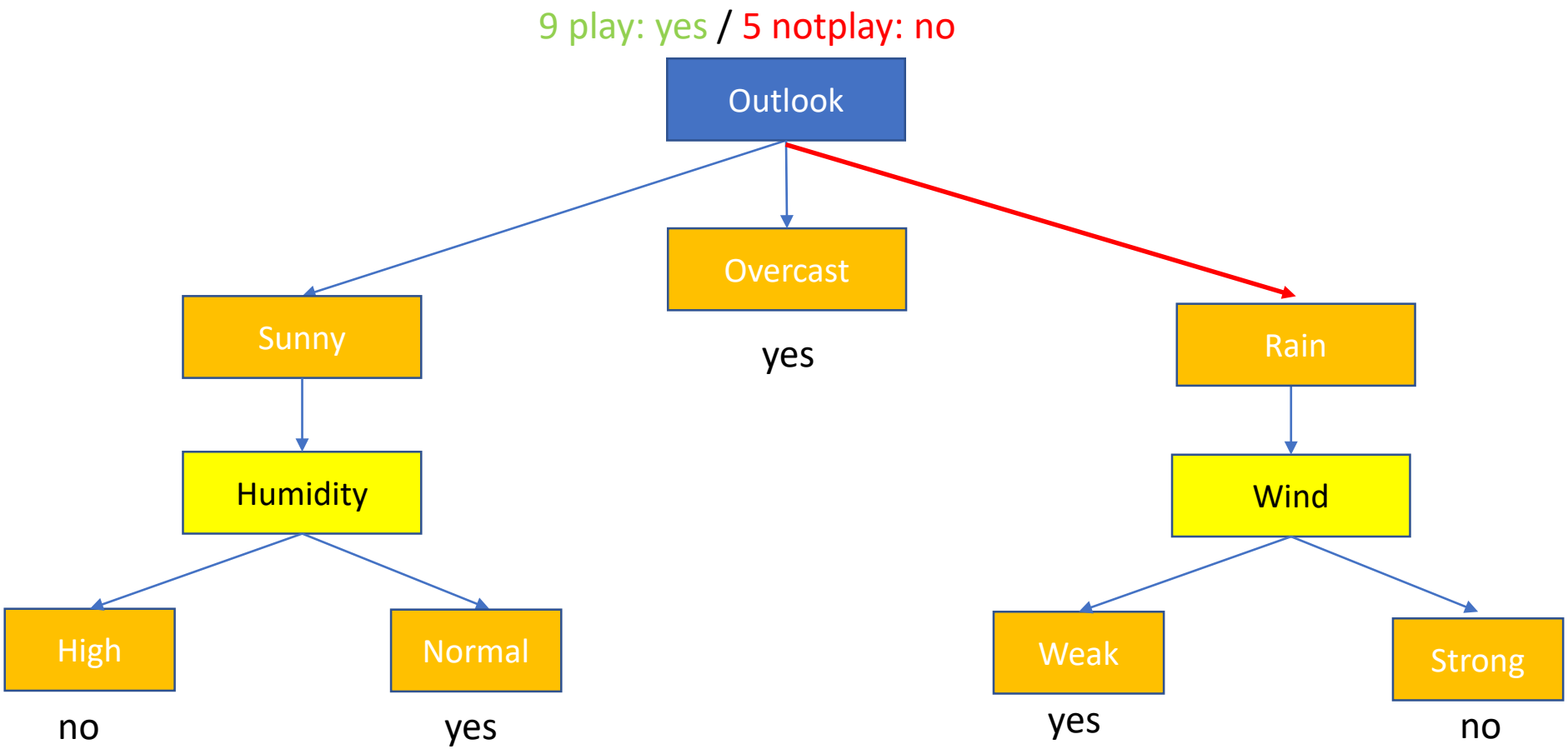

9 play: yes / 5 notplay: no
Outlook
Sunny
Overcast
Rain
Humidity
High
Normal
Day Outlook Humid Wind
D3 Overcast High Weak
D7 Overcast Normal Strong
D12 Overcast High Srong
D13 Overcast Normal Weak
4 Yes and 0 No
Pure subset
Day Outlook Humid Wind
D4 Rain High Weak
D5 Rain Normal Weak
D6 Rain Normal Strong
D10 Rain Normal Weak
D14 Rain High Strong
3 Yes and 2 No
We should split further
Day Humid Wind
D1 High Weak
D2 High Strong
D8 High Weak
0 Yes and 3 No
Pure subset
Day Humid Wind
D9 Normal Weak
D11 Normal Strong
2 Yes and 0 No
Pure subset


9 play: yes / 5 notplay: no
Outlook
Sunny
Overcast
Rain
Humidity
Wind
High
Normal
Weak
Strong
Day Outlook Humid Wind
D3 Overcast High Weak
D7 Overcast Normal Strong
D12 Overcast High Srong
D13 Overcast Normal Weak
4 Yes and 0 No
Pure subset
Day Humid Wind
D1 High Weak
D2 High Strong
D8 High Weak
0 Yes and 3 No
Pure subset
Day Humid Wind
D9 Normal Weak
D11 Normal Strong
2 Yes and 0 No
Pure subset
Day Humid Wind
D4 High Weak
D5 Normal Weak
D10 Normal Weak
3 Yes and 0 No
Pure subset
Day Humid Wind
D6 Normal Strong
D14 High Strong
0 Yes and 2 No
Pure subset

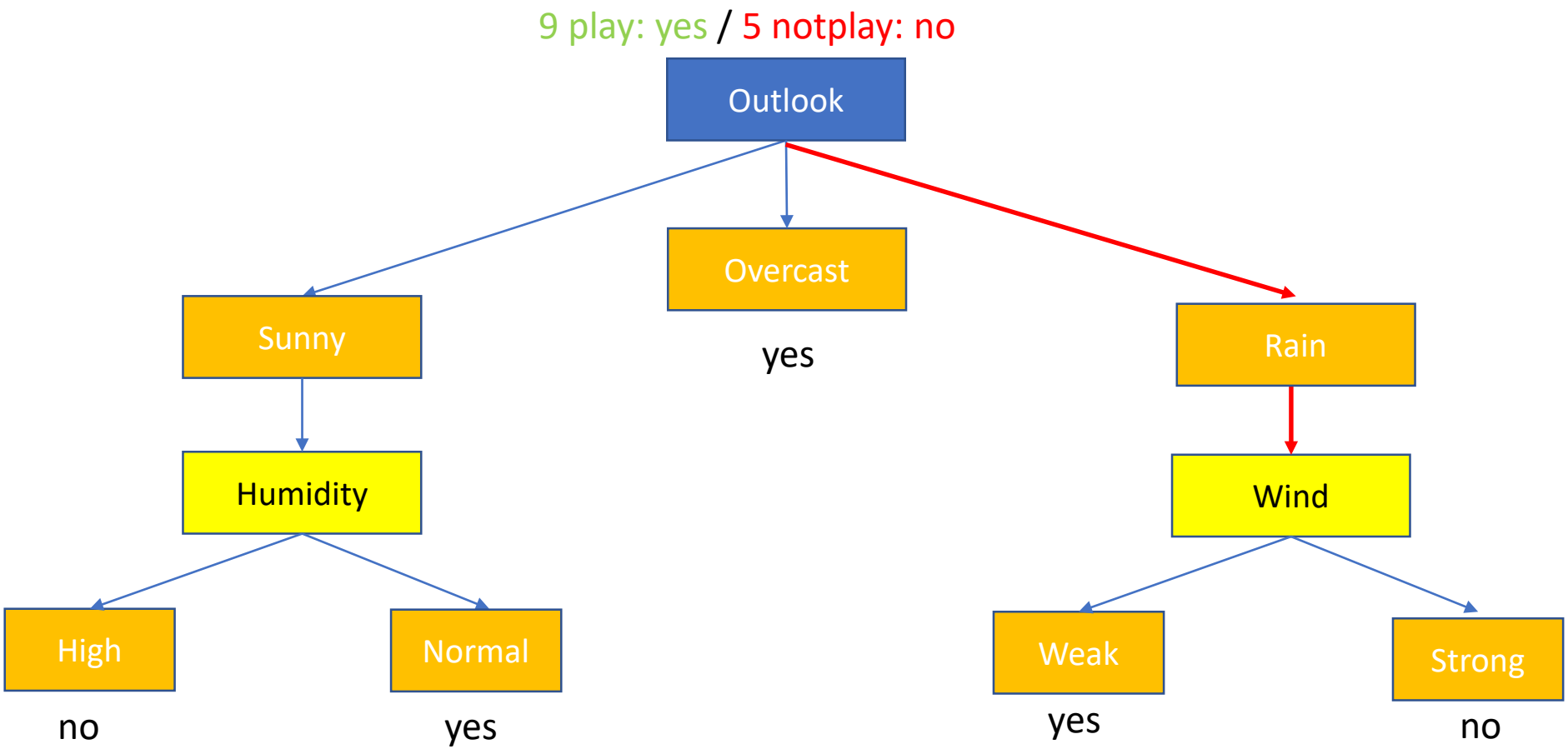

9 play: yes / 5 notplay: no
Outlook
Overcast
yes
Sunny
Rain
Humidity
Wind
High
Normal
Weak
Strong
no
yes
yes
no



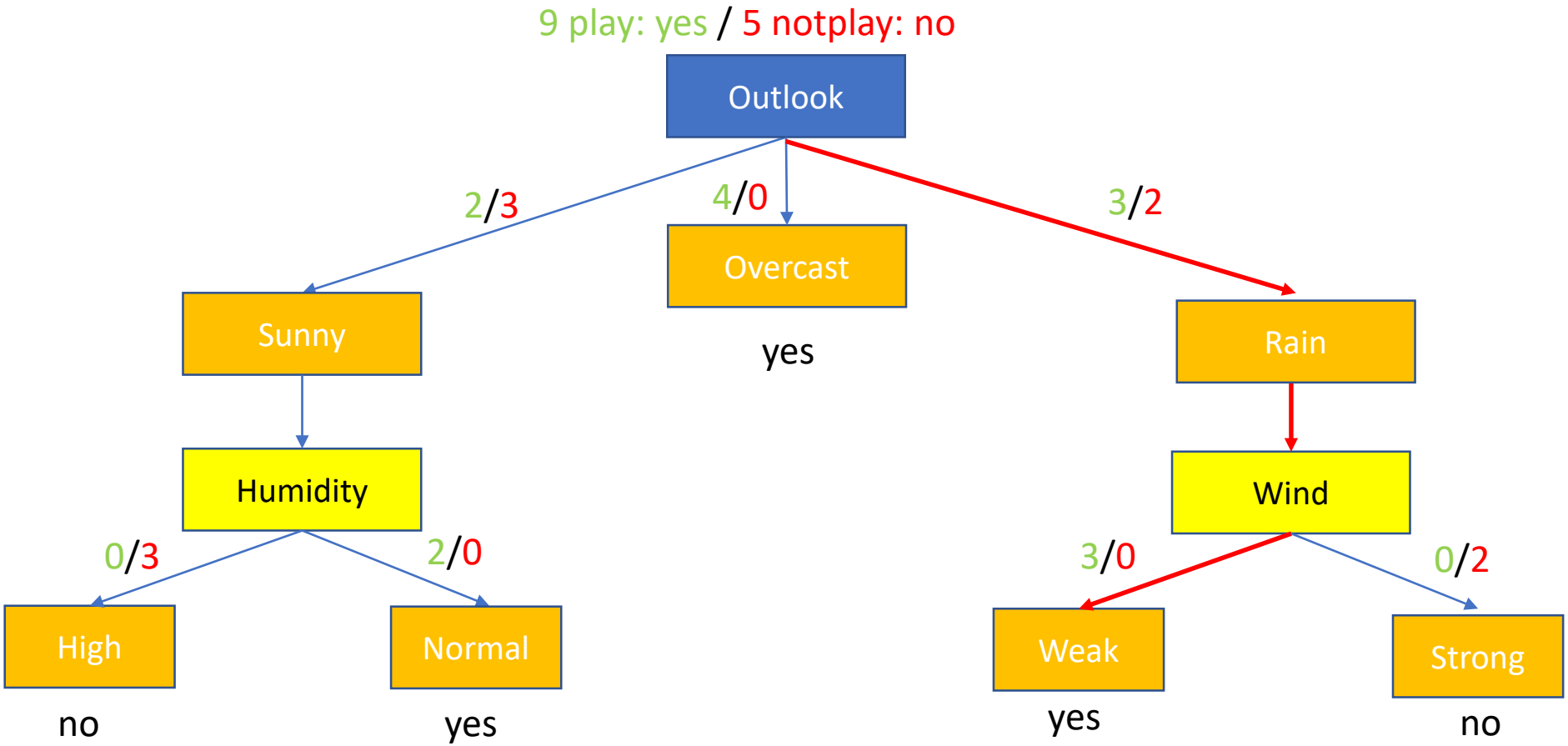
- New data: **Day** **Outlook** **Humid** **Wind**
  D15 Rain. High. Weak ?



- New data: **Day** **Outlook** **Humid** **Wind**
  D15 Rain. High. Weak ?



- New data: **Day** **Outlook** **Humid** **Wind**
  D15 Rain. High. Weak ?

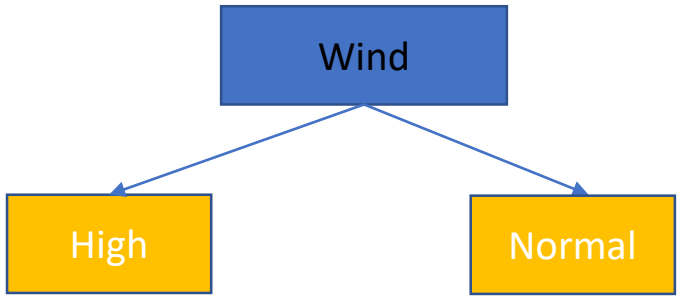


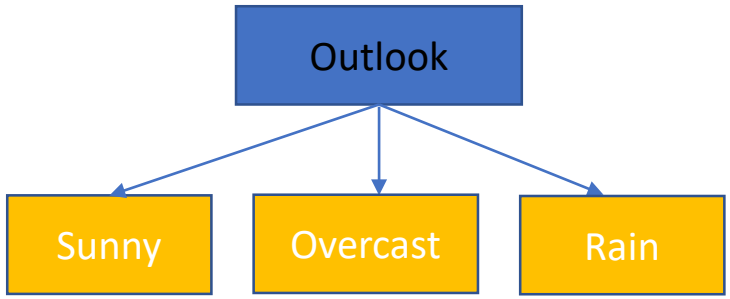
- New data: **Day** **Outlook** **Humid** **Wind**
  D15 Rain. High. Weak → Yes he will play

How we select the root node?

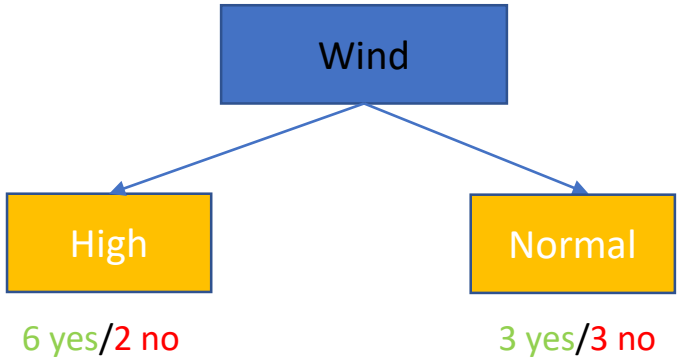
Which one of those attributes should be selected as the first attribute to split on?

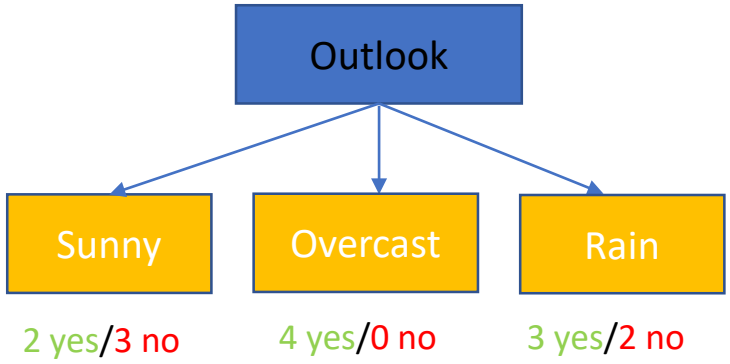
Which one is better ?





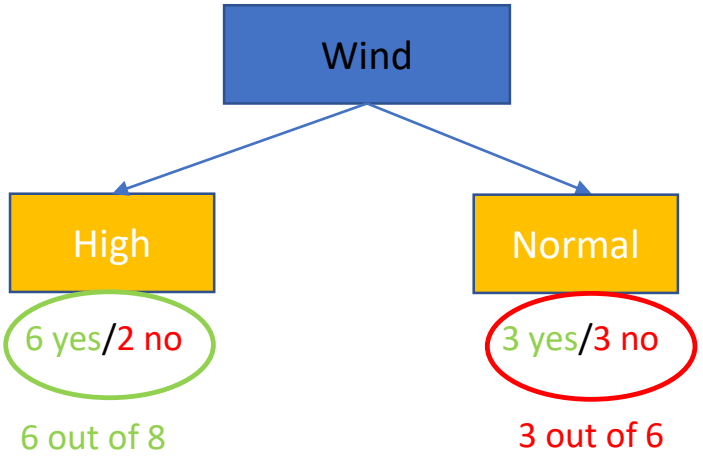
Which one is better ?

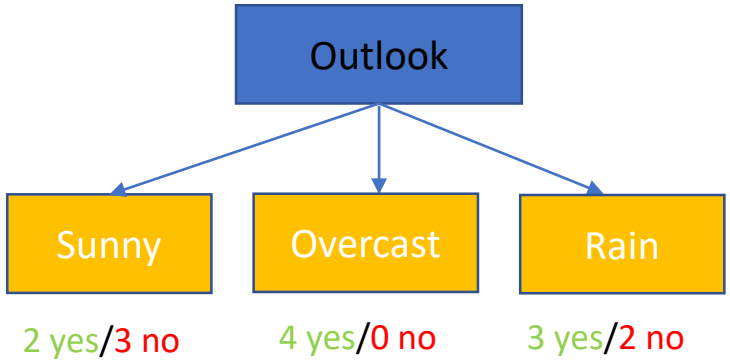




- We should calculate the purity of the split:
  - Being certain about yes and no after splitting
    - Impure: 50% chance → 3 yes and 3 no
    - Pure set: 100% sure -> 4 yes and 0 no

Which one is better ?

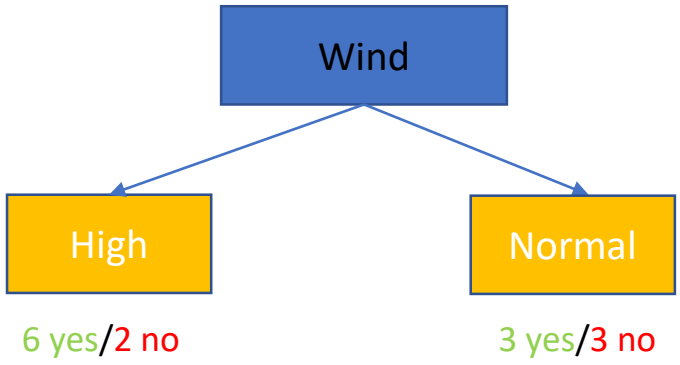


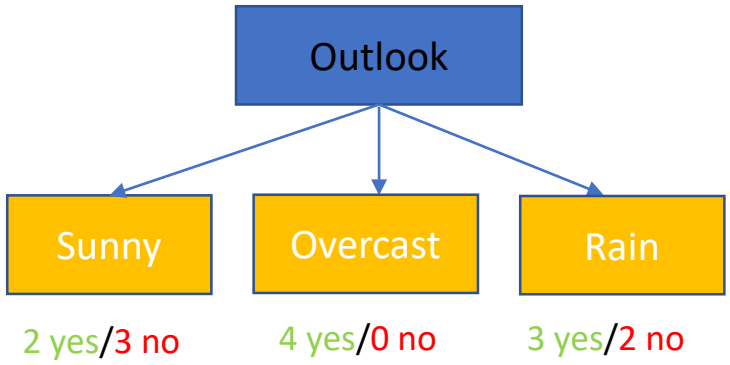


- We should calculate the purity of the split:
  - Being certain about yes and no after splitting
    - Impure: 50% chance → 3 yes and 3 no
    - Pure set: 100% sure -> 4 yes and 0 no

Calculating impurity:

Which one is better ?





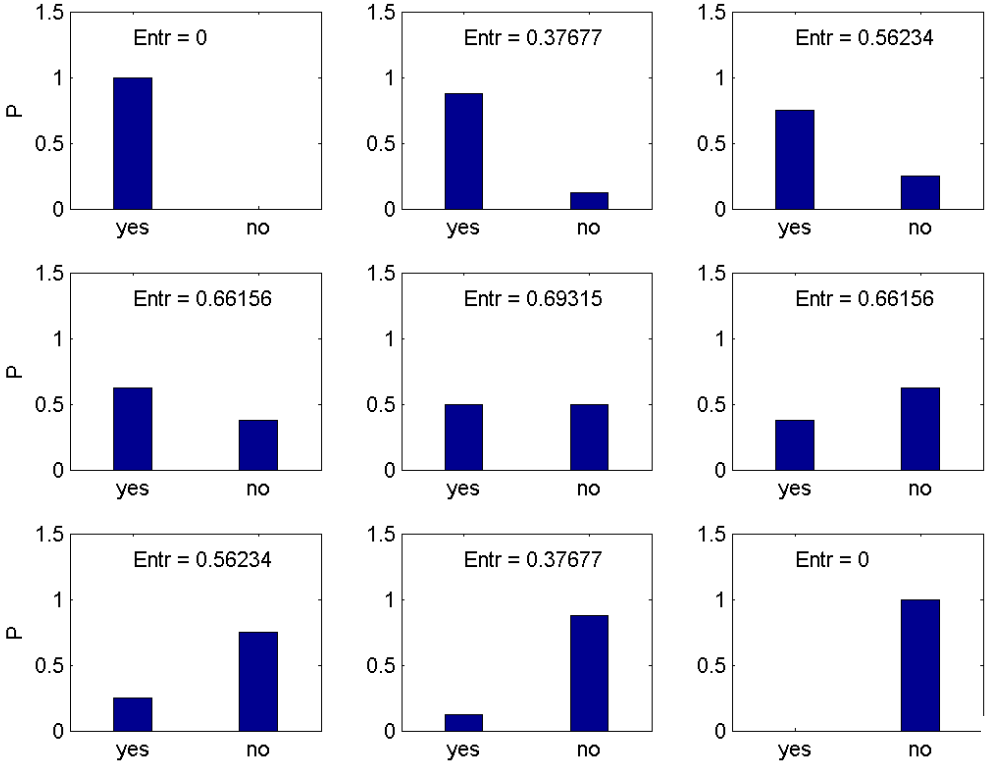
- There are different ways to calculate the impourity:
  - Gini and Entropy
- General form of:

$$\text{Gini (E)} = 1 - \sum_{j=1}^{c} p^2{}_j$$

Gini impurity: $1-(P(\text{yes}))^2 - (P(\text{yes}))^2$

$$\text{Entropy} = -\sum_i P(v_i) \ln\left[P(v_i)\right]$$

Entropy: $-P(\text{yes}) \ln[P(\text{yes})] - P(\text{no}) \ln[P(\text{no})]$



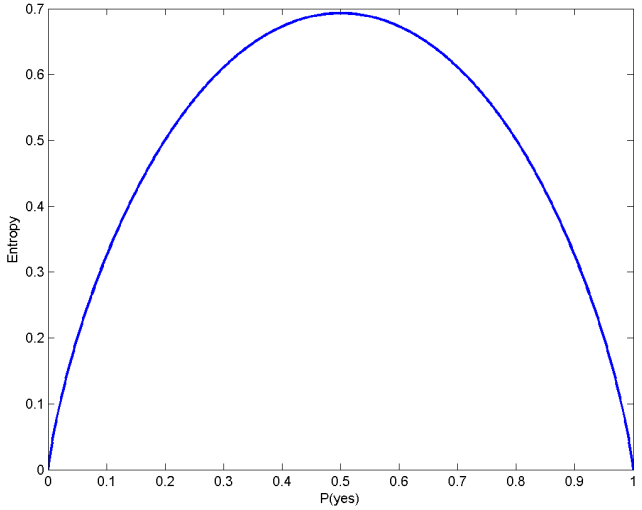
Entropy is a measure of ”order” in a system.

The second law of thermodynamics:
Elements in a closed system tend
to seek their most probable distribution;
in a closed system entropy always increases

The entropy is maximal when all possibilities are equally likely.

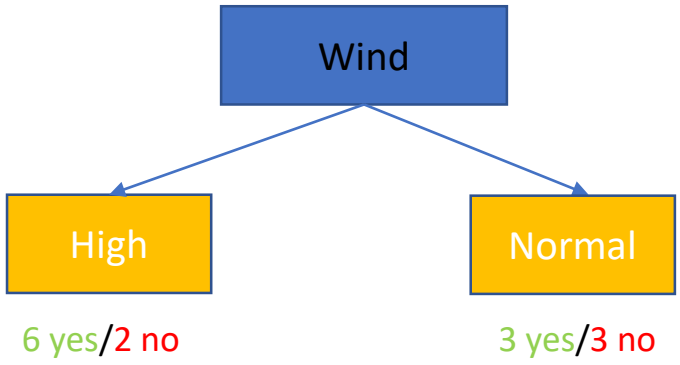
The goal of the decision tree is to decrease the entropy in each node.

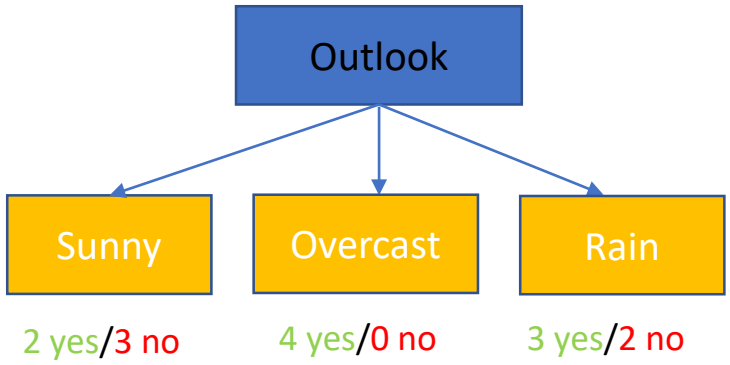
Entropy is zero in a pure ”yes” node (or pure ”no” node).



Calculating impurity:

Which one is better ?

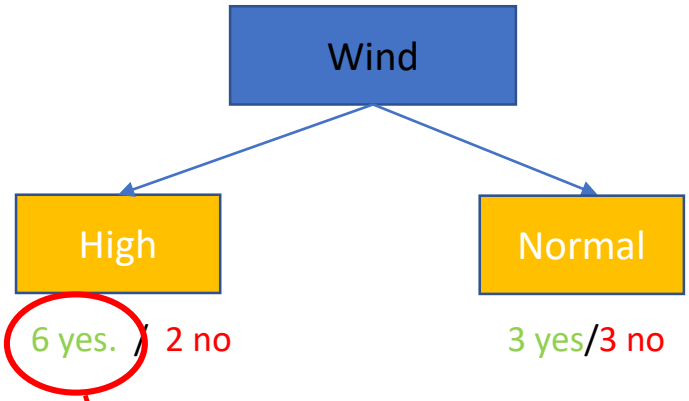


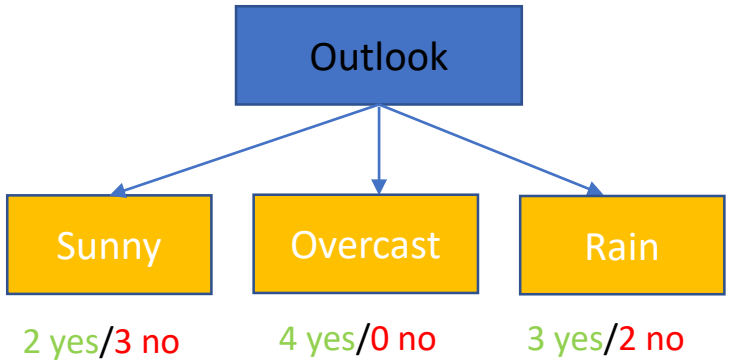


Entropy: -P(yes) ln[P(yes)]-P(no) ln[P(no)]

Calculating impurity:

Which one is better ?



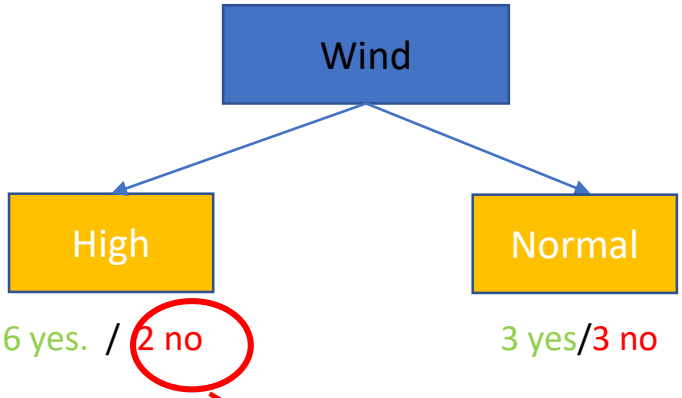


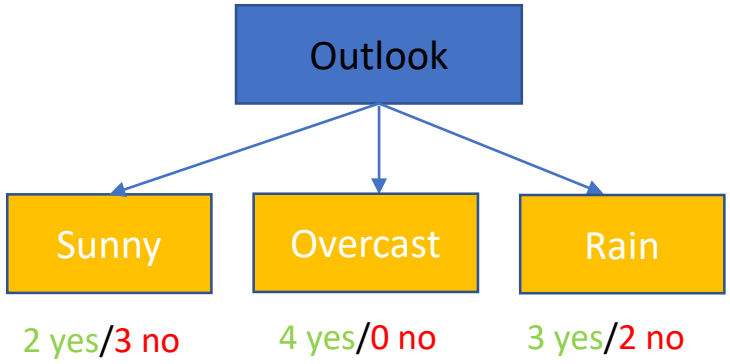
Entropy: -P(yes) ln[P(yes)]-P(no) ln[P(no)]

$$= -\left(\frac{6}{6+2}\right)\ln[\left(\frac{6}{6+2}\right)]$$

Calculating impurity:

Which one is better ?





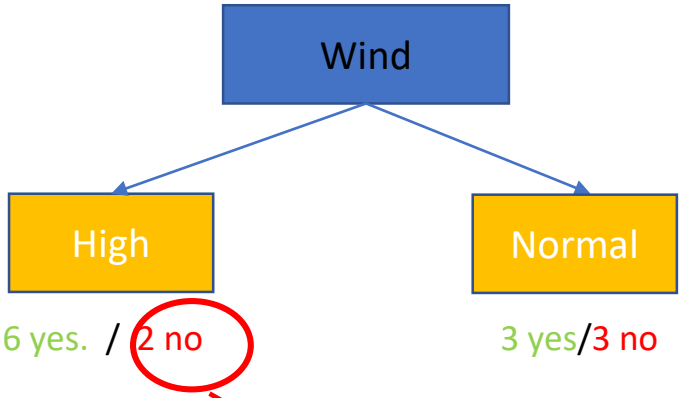
Entropy: -P(yes) ln[P(yes)]-P(no) ln[P(no)]

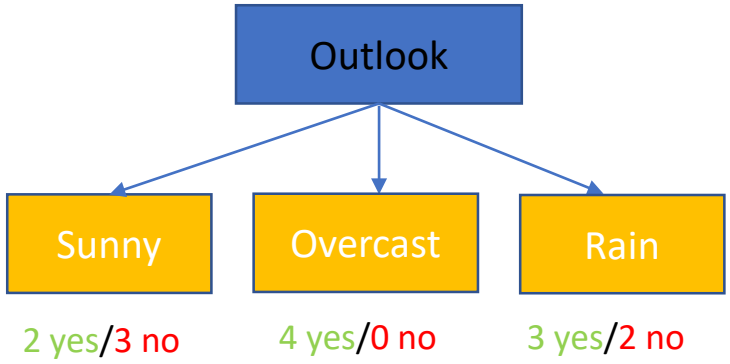
$$= -\left(\frac{6}{6+2}\right)\ln[\left(\frac{6}{6+2}\right)] - (\frac{2}{6+2})\ln[\left(\frac{2}{6+2}\right)]$$

= ?

Calculating impurity:

Which one is better ?





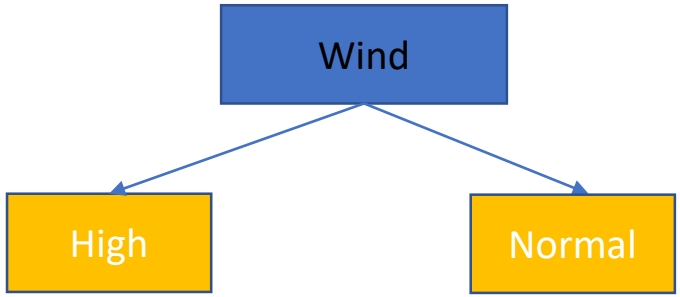
Entropy: -P(yes) ln[P(yes)]-P(no) ln[P(no)]

$$= -\left(\frac{6}{6+2}\right)\ln[\left(\frac{6}{6+2}\right)] - (\frac{2}{6+2})\ln[\left(\frac{2}{6+2}\right)]$$

= 0.80

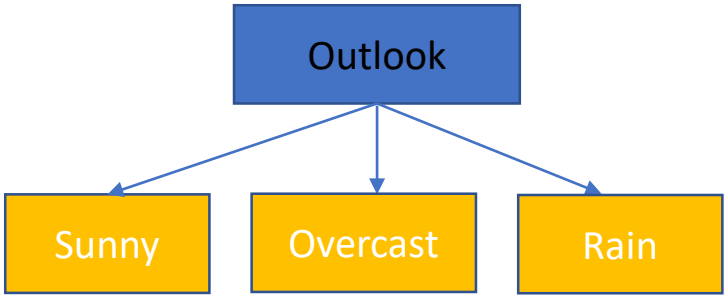
Calculating impurity:

Which one is better ?



6 yes. / 2 no

3 yes/3 no



2 yes/3 no

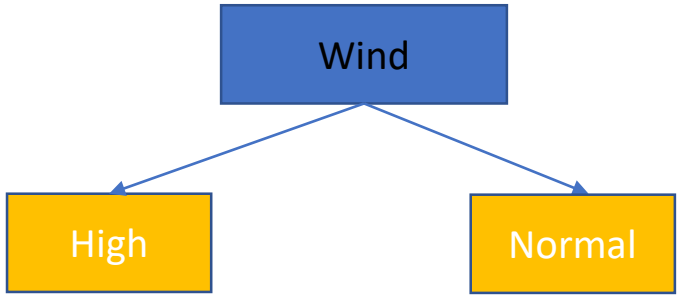
4 yes/0 no

3 yes/2 no

Entropy: 0.80

Calculating impurity:

Which one is better ?

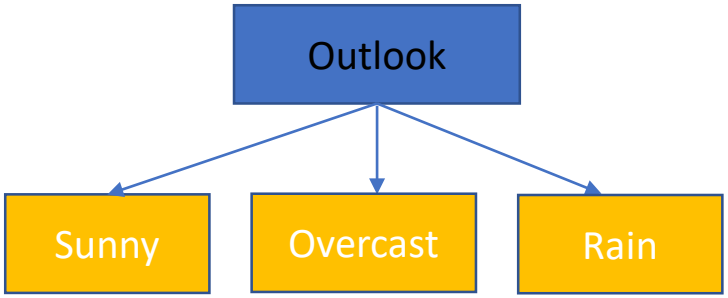


6 yes. / 2 no

Entropy: 0.80

3 yes /. 3 no

Entropy: ?



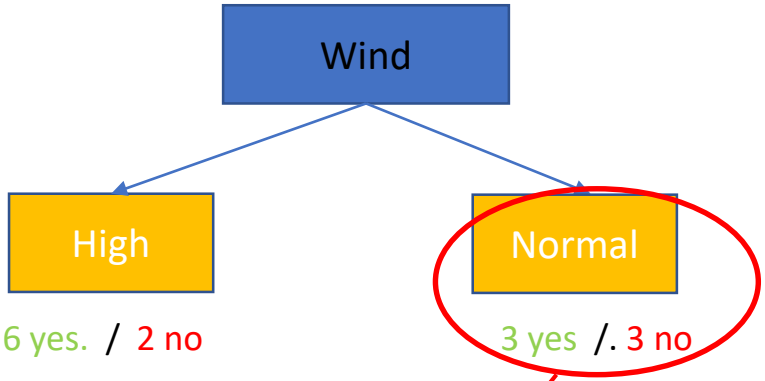
2 yes/3 no

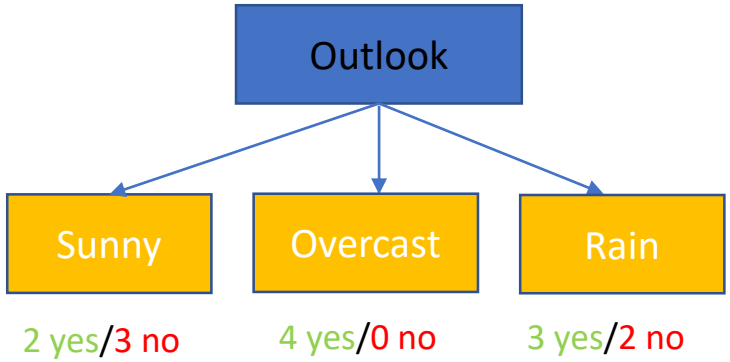
4 yes/0 no

3 yes/2 no

Calculating impurity:

Which one is better ?





Entropy: 0.80

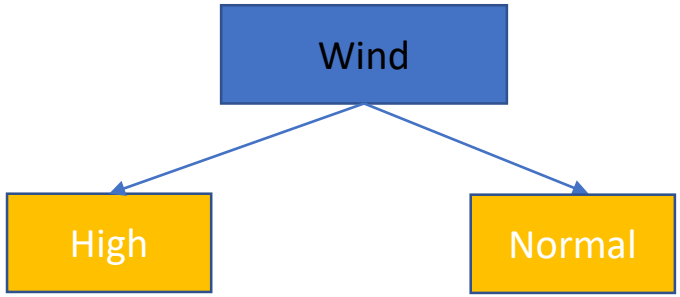
Entropy: -P(yes) ln[P(yes)]-P(no) ln[P(no)]

$$= -\left(\frac{3}{3+3}\right)\ln[\left(\frac{3}{3+3}\right)] - (\frac{3}{3+3})\ln[\left(\frac{3}{3+3}\right)]$$

= 1

Calculating impurity:

Which one is better ?

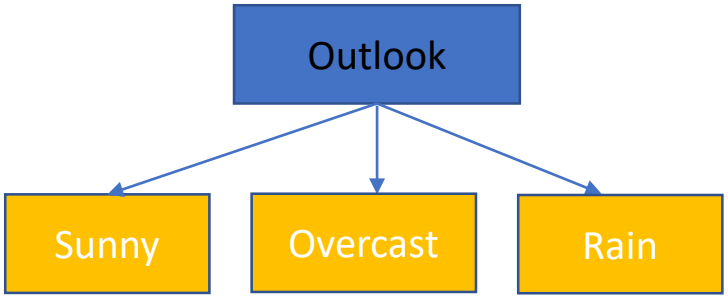


6 yes. / 2 no

Entropy: 0.80

3 yes /. 3 no

Entropy: 1



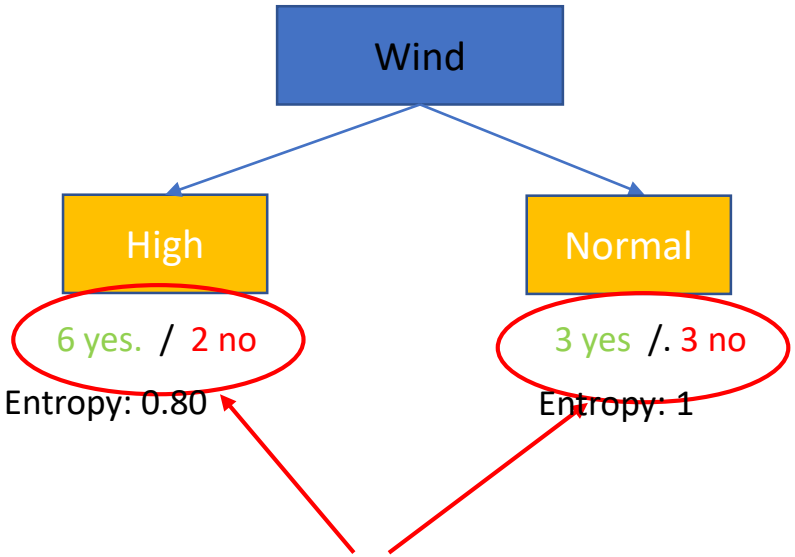
2 yes/3 no

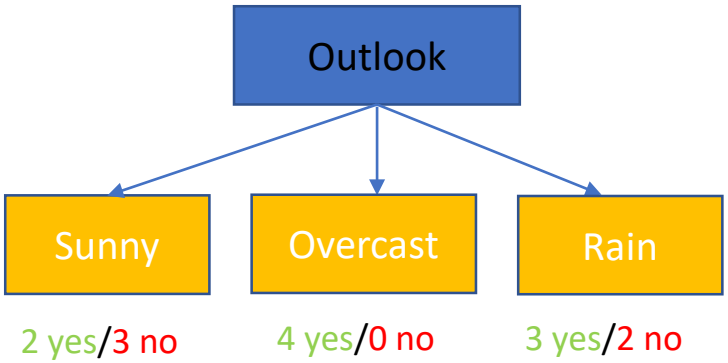
4 yes/0 no

3 yes/2 no

Calculating impurity:

Which one is better ?





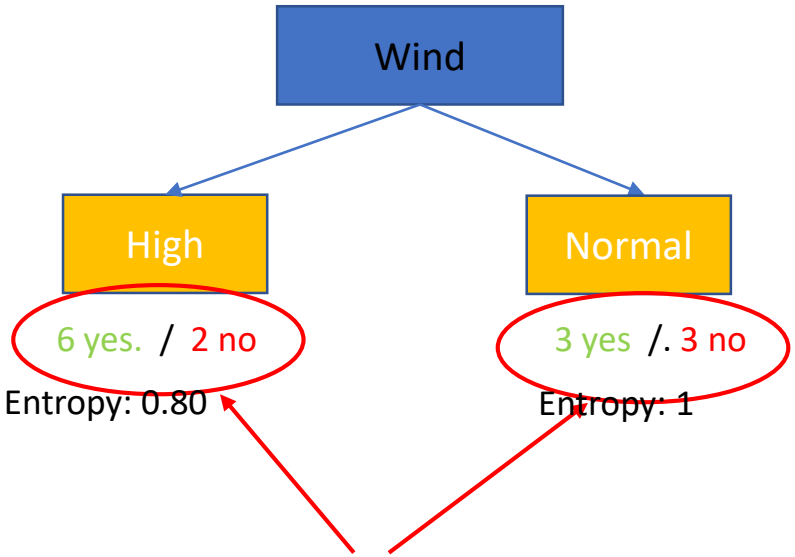
- Since the sampels are different,
  - Weighted average is needed to be calculated

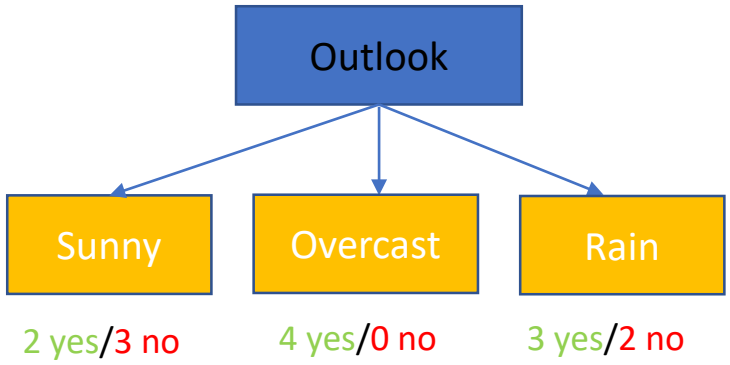
Weighted average of Entropy for Wind:

$$= \left(\frac{total\ number\ \ in\ the\ first\ node}{total\ numbers\ in\ all\ nodes}\right) Entropy + \left(\frac{total\ number\ \ in\ the\ second\ node}{total\ numbers\ in\ all\ nodes}\right) Entropy$$

Calculating impurity:

Which one is better ?





- Since the sampels are different,
  - Weighted average is needed to be calculated

Weighted average of Gini impurity for Wind:

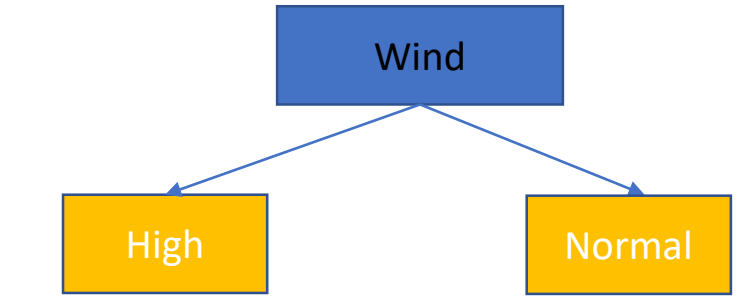
$$= \left(\frac{total\ number\ \ in\ the\ first\ node}{total\ numbers\ in\ all\ nodes}\right) Entropy + \left(\frac{total\ number\ \ in\ the\ second\ node}{total\ numbers\ in\ all\ nodes}\right) Entropy$$

$$= \left(\frac{8}{8+6}\right) 0.80 + \left(\frac{6}{8+6}\right) 1$$

= 0.856

Calculating impurity:

Which one is better ?

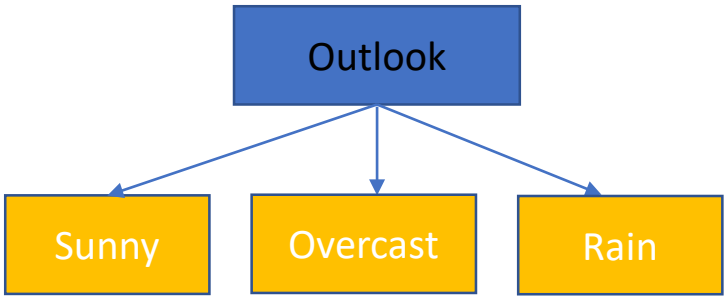


6 yes. / 2 no

Entropy: 0.80

3 yes /. 3 no

Entropy: 1



2 yes/3 no

Entropy: 0.96

4 yes/0 no

Entropy: 0

3 yes/2 no

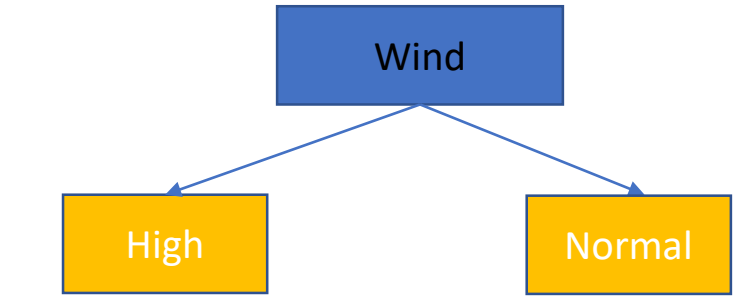
Entropy: 0.96

Weighted average of Entropy for Wind:

=0.856

Calculating impurity:

Which one is better ?

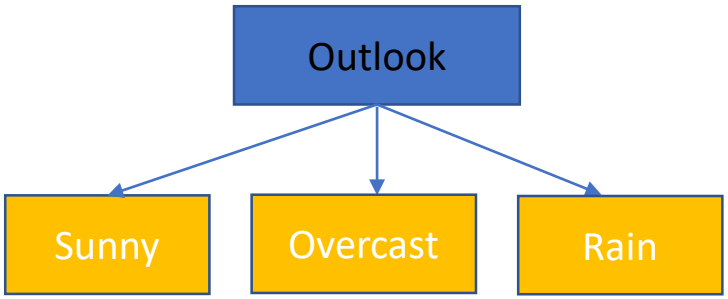


6 yes. / 2 no

Entropy: 0.35

3 yes /. 3 no

Entropy: 1



2 yes/3 no

Entropy: 0.96

4 yes/0 no

Entropy: 0

3 yes/2 no

Entropy: 0.96

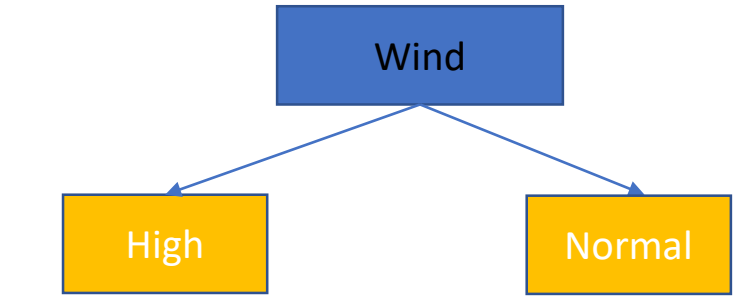
Weighted average of Entropy for Wind:

=0.856

Weighted average of Entropy for Outlook:

=0.684

Calculating impurity:

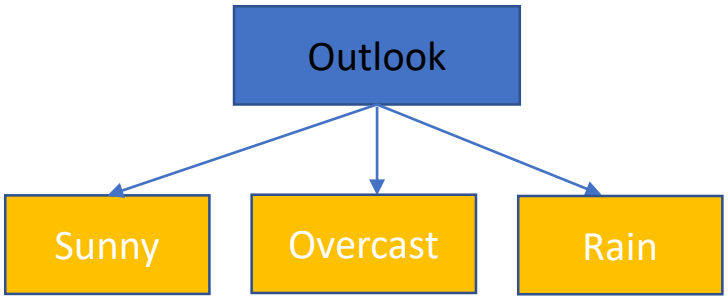


6 yes. / 2 no

Entropy: 0.35

3 yes /. 3 no

Entropy: 1



2 yes/3 no

Entropy: 0.96

4 yes/0 no

Entropy: 0

3 yes/2 no

Entropy: 0.96

Weighted average of Entropy for Wind:

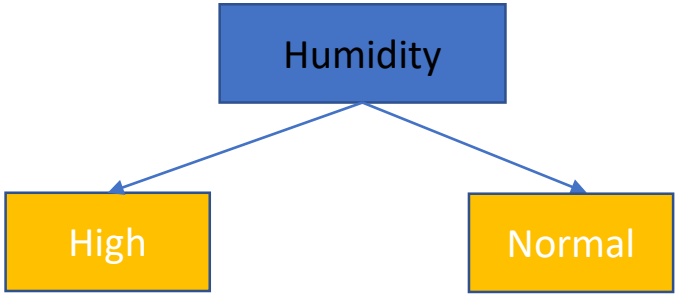
=0.856

Weighted average of Entropy for Outlook:

=0.684

A simple quiz!?

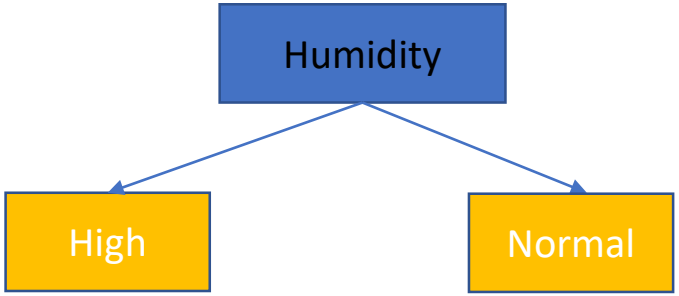
Calculate the impurity (Entropy and Gini) for this feature and then compare with the other two impurity values?



?

A simple quiz!?

Calculate the impurity (Entropy and Gini) for this feature and then compare with the other two impurity values?



?

Decision Tree Weakness:

- The key weakness of decision tree:
  - Decision tree is not a suitable algorithm for continues data
  - Decision tree performs poorly with limited data and multiple classes
  - Computationally expensive
    - In terms of training the data, by splitting each node ..

How do we know it is correct?

How do we know that $h \square f$?
(Hume's Problem of Induction)

Try h on a new (test) set of examples
(ones not used during training)

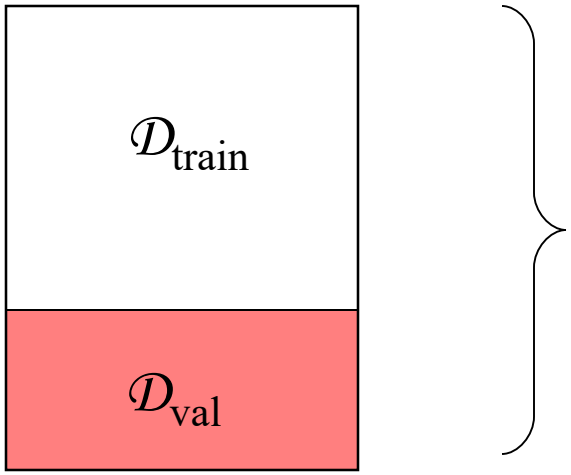
...and assume the ”principle of uniformity”, i.e. the result we get on this test data should be indicative of results on future data. Causality is constant.

Inspired by a slide by V. Pavlovic

Cross-validation

Use a “validation set”.

$$E_{generalisation} \approx E_{validation}$$

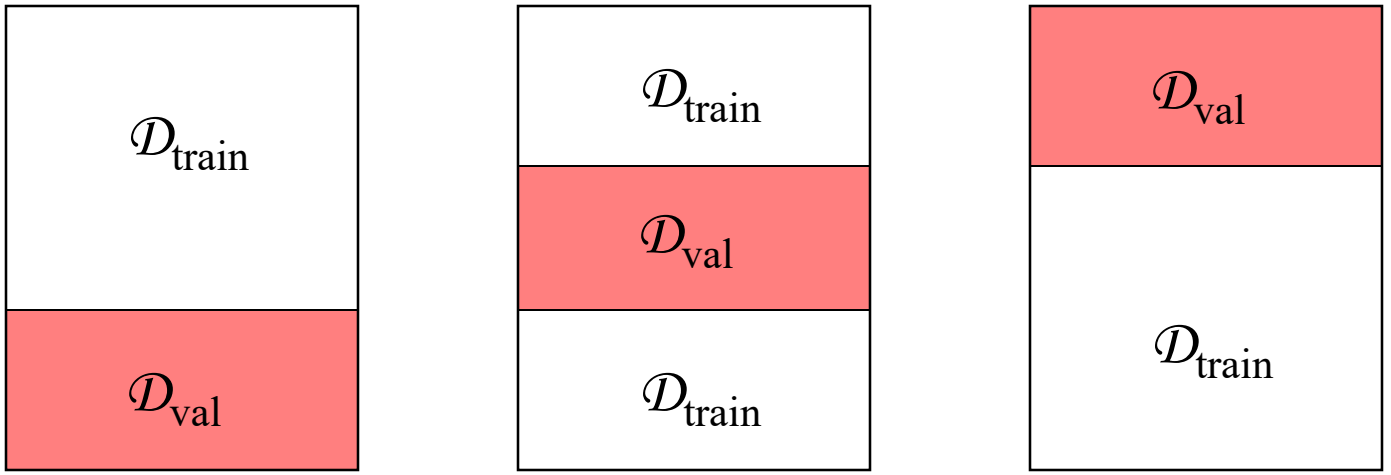


Split your data set into two parts, one for training your model and the other for validating your model.

K-fold Cross-Validation

More accurate than using only one validation set.

$$E_{gen} \approx \langle E_{val} \rangle = \frac{1}{K} \sum_{k=1}^{K} E_{val}(k)$$



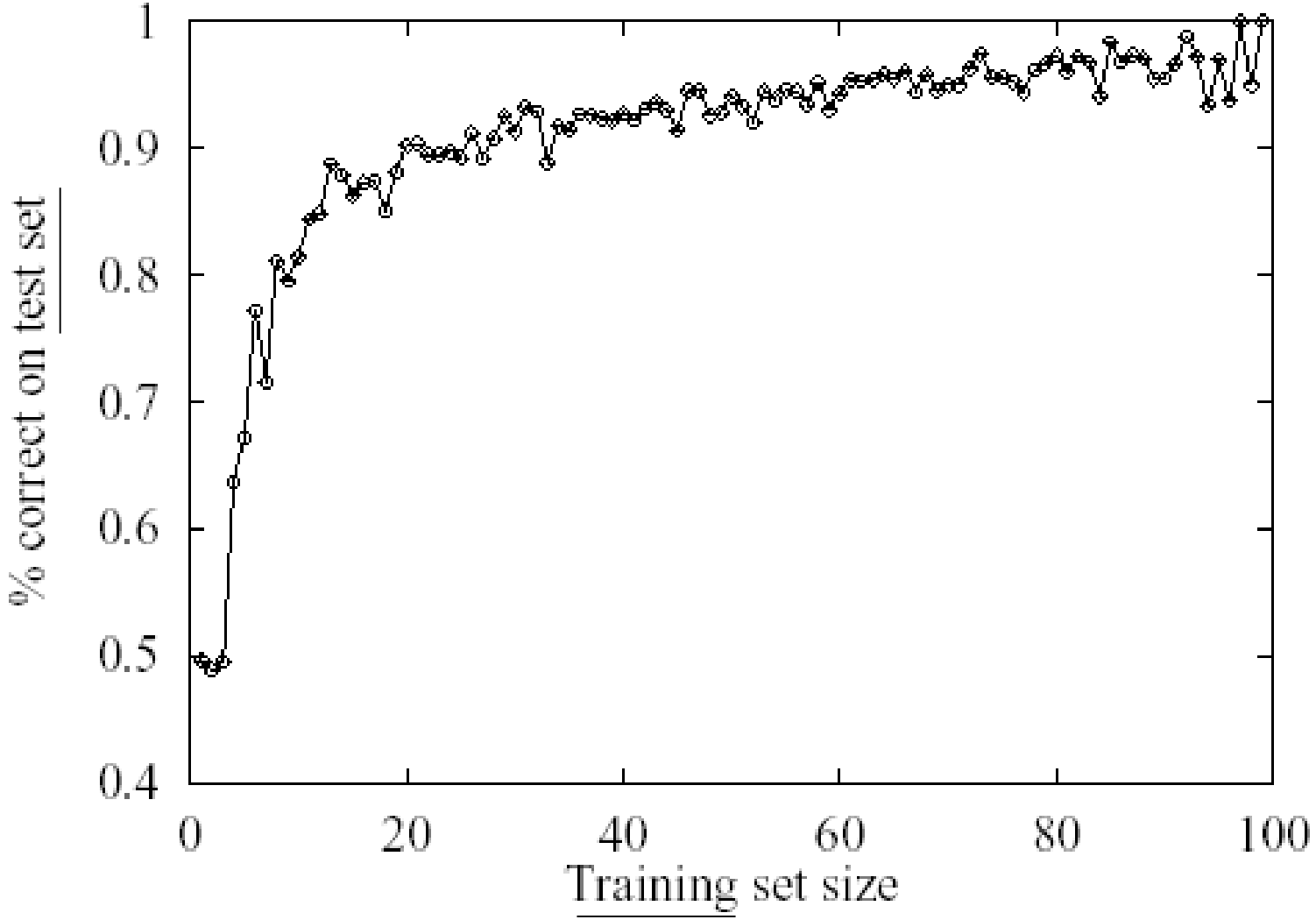
$E_{\text{val}}(1)$ $E_{\text{val}}(2)$ $E_{\text{val}}(3)$

PAC

- Any hypothesis that is consistent with a sufficiently large set of training (and test) examples is unlikely to be seriously wrong; it is **probably approximately correct (PAC)**.

- The error should be < epsilon

Learning curve for the decision tree algorithm on 100 randomly generated examples in the restaurant domain.
The graph summarizes 20 trials.



The error

$\mathbf{X}$ = the set of all possible examples (instance space).

D = the distribution of these examples.

$\mathbf{H}$ = the hypothesis space ($h \square \mathbf{H}$).

N = the number of training data.

$$\text{error}(h) = P[h(\mathbf{x}) \neq f(\mathbf{x}) \mid \mathbf{x} \text{ drawn from } D]$$

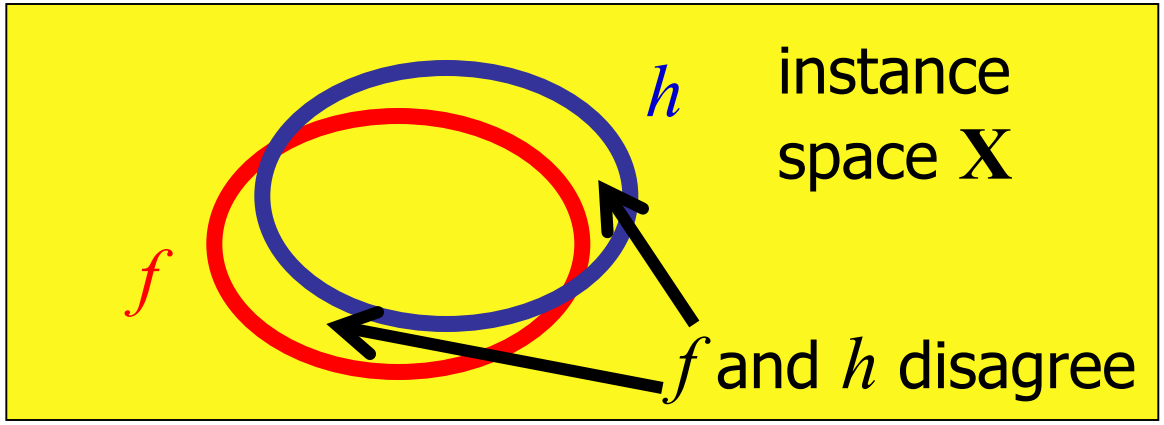


Image adapted from F. Hoffmann @ KTH

Probability for bad hypothesis

Suppose we have a bad hypothesis h with $\text{error}(h) > \square$.

What is the probability that it is consistent with N samples?

- Probability for being inconsistent with one sample = $\text{error}(h) > \square$.
- Probability for being consistent with one sample = $1 - \text{error}(h) < 1 - \square$.
- Probability for being consistent with N independently drawn samples $< (1 - \square)^N$.

Probability for bad hypothesis

What is the probability that the set $\mathbf{H}_{\text{bad}}$ of bad hypotheses with $\text{error}(h) > \varepsilon$ contains a consistent hypothesis?

A measure of the number of bad models

$$P(h \text{ consistent} \wedge \text{error}(h) > \varepsilon) \leq \left|\mathbf{H}_{\text{bad}}\right|(1-\varepsilon)^N \leq \left|\mathbf{H}\right|(1-\varepsilon)^N$$

Probability for bad hypothesis

What is the probability that the set $\mathbf{H}_{\text{bad}}$ of bad hypotheses with $\text{error}(h) > \square$ contains a consistent hypothesis?

$$P(h \text{ consistent} \wedge \text{error}(h) > \varepsilon) \leq \left|\mathbf{H}_{\text{bad}}\right|(1-\varepsilon)^N \leq \left|\mathbf{H}\right|(1-\varepsilon)^N$$

If we want this to be less than some constant $\square$, then

$$\left|\mathbf{H}\right|\left(1-\varepsilon\right)^N < \delta \Rightarrow \ln\left|\mathbf{H}\right| + N\ln(1-\varepsilon) < \ln\delta$$

Probability for bad hypothesis

What is the probability that the set $\mathbf{H}_{\text{bad}}$ of bad hypotheses with $\text{error}(h) > \varepsilon$ contains a consistent hypothesis?

$$P(h \text{ consistent} \wedge \text{error}(h) > \varepsilon) \leq |\mathbf{H}_{\text{bad}}|(1-\varepsilon)^N \leq |\mathbf{H}|(1-\varepsilon)^N$$

If we want this to be less than some constant δ, then

$$N > \frac{\ln(|\mathbf{H}|) - \ln(\delta)}{-\ln(1-\varepsilon)} \approx \frac{\ln(|\mathbf{H}|) - \ln(\delta)}{\varepsilon}$$

Don't expect to learn very well if $\mathbf{H}$ is large

How to make learning work?

- Use simple hypotheses
  - Always start with the simple ones first
- Constrain $\mathbf{H}$ with priors
  - Do we know something about the domain?
  - Do we have reasonable a priori beliefs on parameters?
- Use many observations
  - Easy to say...
- Always report validation results

Summary

- In learning context we have Classification and Regression
  - Classification tries to predict descrete data e.g., label
  - Regression tries to predict continuse data e.g., quantity
- To build a hypotheis we should start from a simple
- Decision tree is a simple, but a powerful classifier
  - For classification and regresssion problems
- To validate the algorithm we can use cross validation
- PAC is a theoretical framework and its goal is to build a hypotheis that has high prabobility and approximatly correct.