



INDIAN INSTITUTE OF
INFORMATION
TECHNOLOGY

Decision Tree Learning and Decision Theory

Dr. Animesh Chaturvedi

Assistant Professor: IIIT Dharwad

Young Researcher: Heidelberg Laureate Forum

Postdoc: King's College London & The Alan Turing Institute

PhD: IIT Indore MTech: IIITDM Jabalpur



Indian Institute of Technology Indore
भारतीय प्रौद्योगिकी संस्थान इंदौर



PDPM
Indian Institute of Information Technology,
Design and Manufacturing, Jabalpur

The
Alan Turing
Institute

Decision Trees

- Convenient Representation
 - Developed with learning in mind
 - Deterministic
 - Comprehensible output
- Expressive
 - Equivalent to propositional Disjunctive Normal Form (DNF)
 - Handles discrete and continuous parameters
- Simple learning algorithm
 - Handles noise well
 - Classify
 - Constructive (build DT by adding nodes)

Concept Learning

- E.g., Learn concept
 - Target Function has two values: T or F
- Represent concepts as **decision trees**
- Use **hill climbing search** space of **decision trees**
 - Start with simple concept
 - Refine it into a complex concept as needed

A concept can be thought of as a conjunction of boolean functions whose values categorize data into different classes.

Example: “Good day for tennis”

- Attributes of instances
 - **Outlook** = {rainy (r), overcast (o), sunny (s)}
 - **Temperature** = {cool (c), medium (m), hot (h)}
 - **Humidity** = {normal (n), high (h)}
 - **Wind** = {weak (w), strong (s)}
- Class value
 - **Play Tennis?** = {don't play (n), play (y)}
- Feature = attribute with one value
 - E.g., **outlook** = *sunny*
- Sample instance
 - **outlook**=*sunny*, **temp**=*hot*, **humidity**=*high*, **wind**=*weak*

Experience: “Good day for tennis”

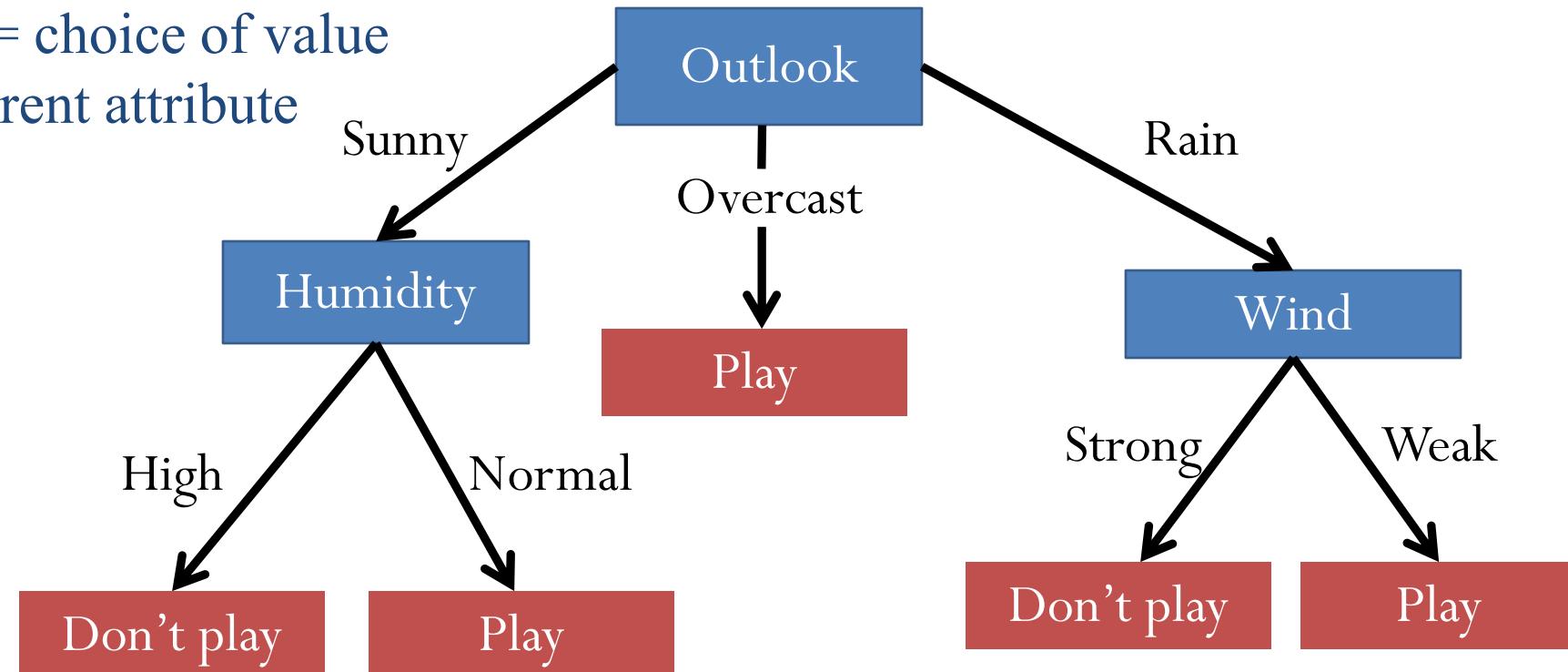
Day	Outlook	Temp	Humid	Wind	PlayTennis?
d1	s	h	h	w	n
d2	s	h	h	s	n
d3	o	h	h	w	y
d4	r	m	h	w	y
d5	r	c	n	w	y
d6	r	c	n	s	n
d7	o	c	n	s	y
d8	s	m	h	w	n
d9	s	c	n	w	y
d10	r	m	n	w	y
d11	s	m	n	s	y
d12	o	m	h	s	y
d13	o	h	n	w	y
d14	r	m	h	s	n

Decision Tree Representation

Good day for tennis?

Leaves = classification

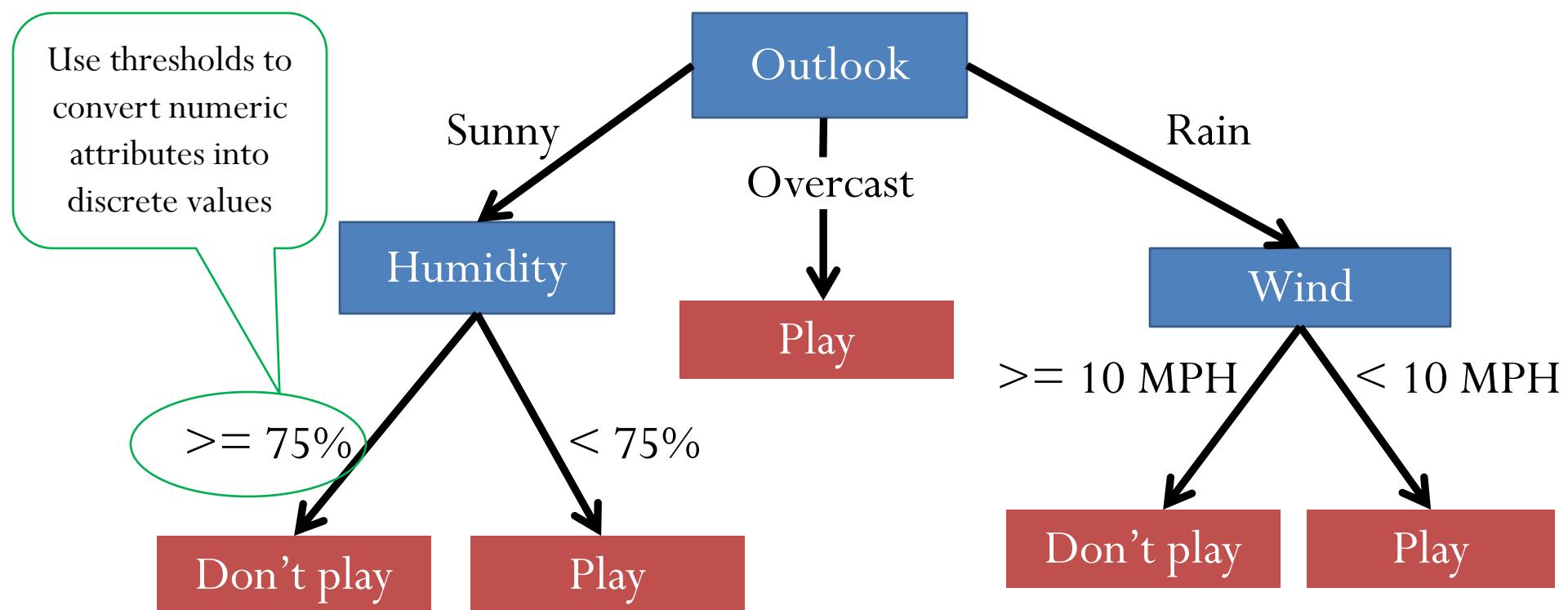
Arcs = choice of value
for parent attribute



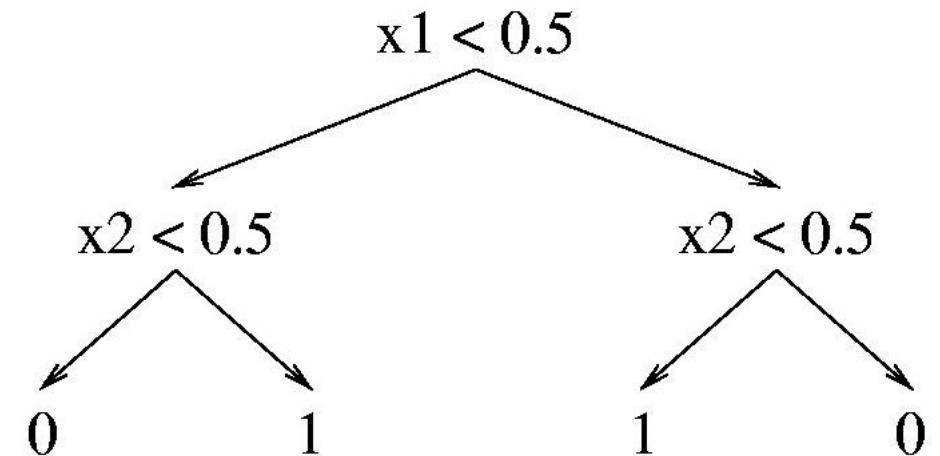
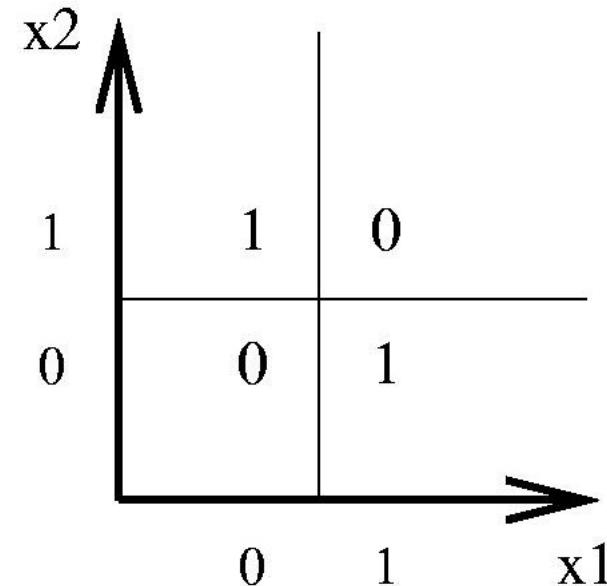
Decision tree is equivalent to logic in disjunctive normal form

$$\text{Play} \Leftrightarrow (\text{Sunny} \wedge \text{Normal}) \vee \text{Overcast} \vee (\text{Rain} \wedge \text{Weak})$$

Numeric Attributes



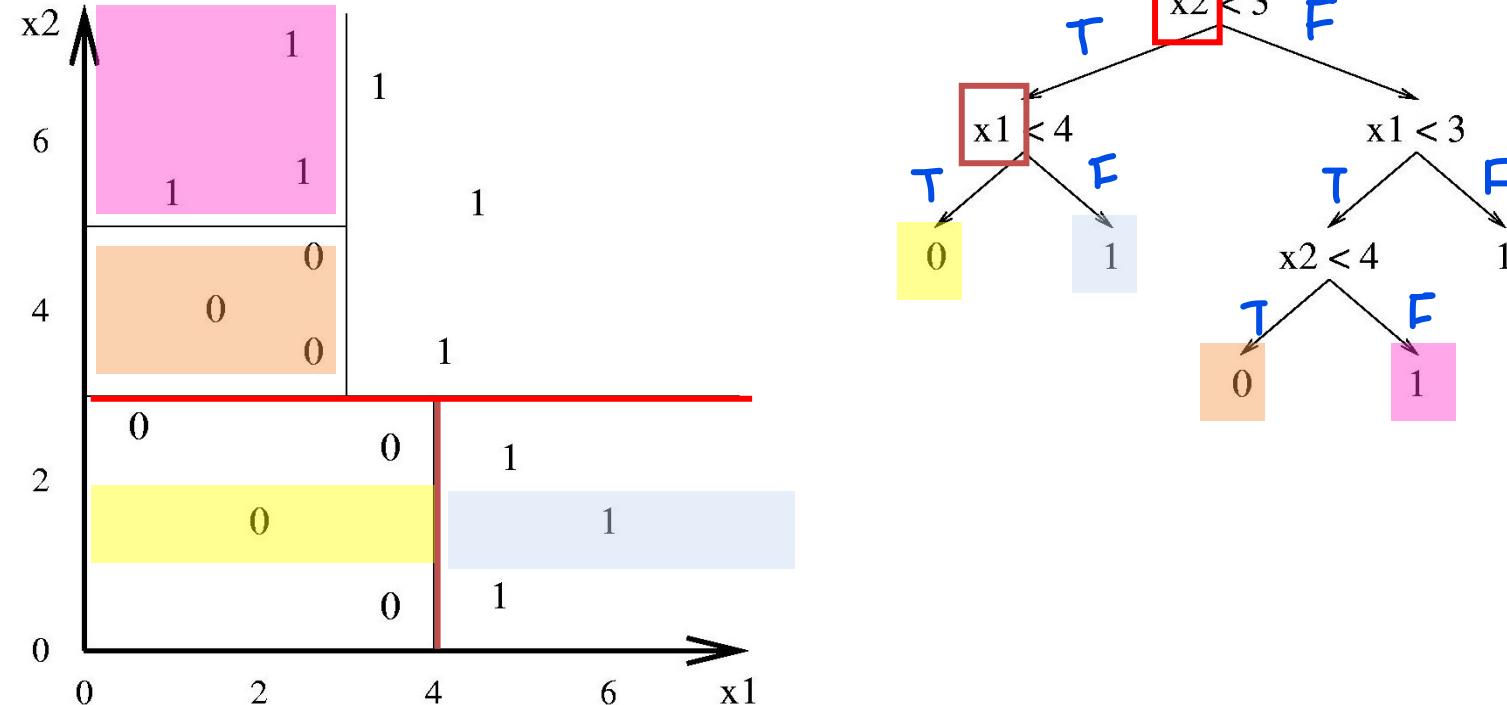
Boolean function can represents Decision tree



The tree will in the worst case require exponentially many nodes, however.

Decision tree to make Decision Boundaries

- Decision trees divide the feature space into axis-parallel rectangles, and label each rectangle with one of the K classes.



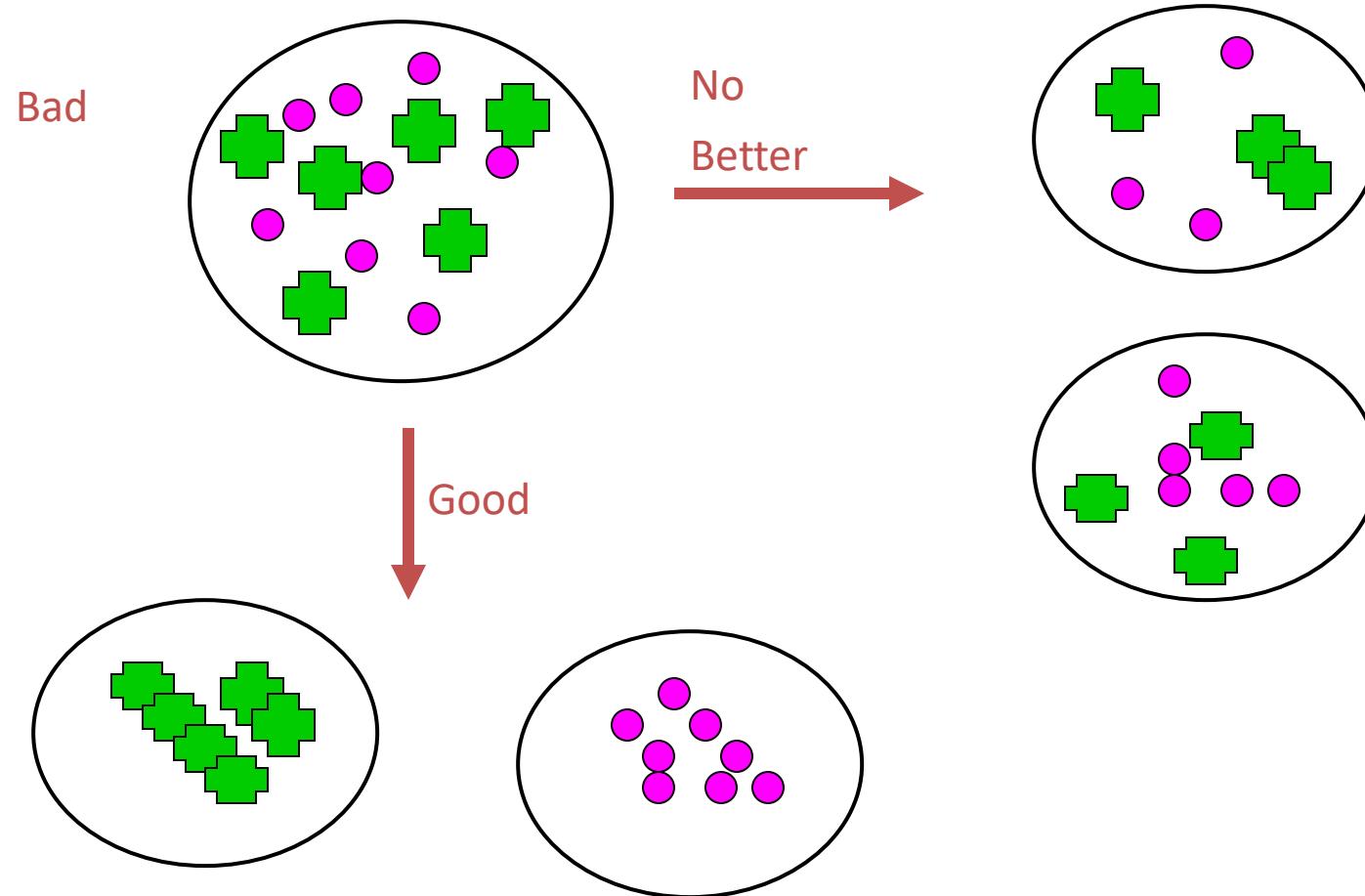
Depth of Decision tree and Boolean functions

Decision Trees Provide Variable-Size Hypothesis Space

As the number of nodes (or depth) of tree increases, the hypothesis space grows

- **depth 1** (“decision stump”) can represent any boolean function of one feature.
- **depth 2** Any boolean function of two features; some boolean functions involving three features (e.g., $(x_1 \wedge x_2) \vee (\neg x_1 \wedge \neg x_3)$)
- etc.

Disorder is bad Homogeneity is good



Which attribute should we use to split?

Decision Tree General Algorithm

BuildTree(TraingData)

 Split(TrainingData)

Split(D)

 If (all points in D are of the same class)

 Then Return

 For each attribute A

 Evaluate splits on attribute A

 Use best split to partition D into D1, D2

 Split (D1)

 Split (D2)

How to learn decision trees

- Constructing optimal binary decision trees is an NP Complete problem
 - Optimal tree is one which minimizes the expected number of tests required to identify the unknown object
 - NP-complete: belongs to both NP and NP-hard; easy to verify a solution to NP-complete, but hard to find a solution
- Often resort to heuristic algorithms
 - Build an empty decision tree → split → recurse (choosing a good attribute for splitting is important)
 - Some examples: ID3, C4.5, CART

There are 3 decision tree algorithms :- ID3, C4.5, CART

Features	ID3	C4.5	CART
Type of Data	Categorical	continuous & Categorical	Continuous and Nominal attribute data
Speed	Low	Faster than ID3	Average
Boosting	Doesn't support	Doesn't support	Supports
Pruning	No	Pruning	Post pruning
Missing Values	Can't handle	Can't handle	Can Handle
Formula	Information Entropy & Info Gain	Use split ratio and gain info	Use Gini diversity Index

* Pruning is when we selectively remove branches from a tree.
Goal is to remove unwanted branches & improve tree structure.

ID3

ID	Fever	Cough	Breathing issues	Infected
1	NO	NO	NO	NO
2	YES	YES	YES	YES
3	YES	YES	NO	NO
4	YES	NO	YES	YES
5	YES	YES	YES	YES
6	NO	YES	NO	NO
7	YES	NO	YES	YES
8	YES	NO	YES	YES
9	NO	YES	YES	YES
10	YES	YES	NO	YES
11	NO	YES	NO	NO
12	NO	YES	YES	YES
13	NO	YES	YES	NO
14	YES	YES	NO	NO

$$\text{Entropy}(\text{Fever}) = -\frac{6}{14} \log \frac{6}{14} - \frac{8}{14} \log \frac{8}{14} = 0.987$$

$$\text{Entropy}(\text{Fever}=\text{Yes}) = -\frac{6}{8} \log \frac{6}{8} - \frac{2}{8} \log \frac{2}{8} = 0.811$$

$$\text{Entropy}(\text{Fever}=\text{No}) = -\frac{4}{6} \log \frac{4}{6} - \frac{2}{6} \log \frac{2}{6} = 0.918$$

$$\begin{aligned}\text{Information Gain} &= \text{Entropy}(\text{Fever}) - \sum \frac{|S_v|}{|S|} \text{Entropy}(\text{Fever}=v) \\ &= 0.987 - \left(\frac{8}{14} (0.811) + \frac{6}{14} (0.918) \right) \\ &= 0.987 - (0.463 + 0.393) = 0.987 - 0.856 \\ &= \underline{\underline{0.131}}\end{aligned}$$

Similarly,

$$\text{Entropy}(\text{Cough}) = -\frac{6}{14} \log \frac{10}{14} - \frac{8}{14} \log \frac{4}{14} = 0.987$$

$$\text{Entropy}(\text{Cough}=\text{Yes}) = -\frac{5}{10} \log \frac{5}{10} - \frac{5}{10} \log \frac{5}{10} = 1$$

$$\text{Entropy}(\text{Cough}=\text{No}) = -\frac{1}{4} \log \frac{1}{4} - \frac{3}{4} \log \frac{3}{4} = 0.811$$

$$\text{Information Gain (cough)} = 0.987 - \frac{10}{14}(1) - \frac{4}{14}(0.011) = 0.207$$

$$\text{Entropy (Breathing Issue)} = 0.987$$

$$\text{Entropy (Breathing Issue = yes)} = -\frac{7}{8} \log \frac{7}{8} - \frac{1}{8} \log \frac{1}{8} = 0.543$$

$$\text{Entropy (Breathing Issue = no)} = -\frac{5}{6} \log \frac{5}{6} - \frac{1}{6} \log \frac{1}{6} = 0.650 \leftarrow$$

$$\text{Information Gain (Breathing)} = 0.987 - \frac{8}{14}(0.543) - \frac{6}{14}(0.650) =$$

Breathing Issue
yes / no

Fever
yes / no
YES
Can't deduce

ID	Fever	Cough	Breathing issues	Infected
1	YES	YES	YES	YES
2	YES	NO	YES	YES
3	YES	YES	YES	YES
4	YES	NO	YES	YES
5	YES	NO	YES	YES
6	NO	YES	YES	YES
7	NO	YES	YES	YES
8	NO	YES	YES	NO

$$= 0.398 \leftarrow \text{Highest IG}$$

$$\text{Entropy (Fever)} = -\frac{7}{8} \log \frac{7}{8} - \frac{1}{8} \log \frac{1}{8}$$

$$= 0.543$$

$$\text{Entropy (fever = yes)} = 0$$

$$\text{Entropy (fever = no)} = -\frac{2}{3} \log \frac{2}{3} - \frac{1}{3} \log \frac{1}{3}$$

$$\text{IG (fever)} = 0.543 - \frac{5}{8}(0) - \frac{3}{8}(0.918) = 0.199$$

Highest IG

$$\text{Entropy}(\text{cough}) = 0.543 \quad \text{Entropy}(\text{cough}=\text{yes}) = -\frac{4}{5} \log \frac{4}{5} - \frac{1}{5} \log \frac{1}{5} = 0.722$$

$$\text{Entropy}(\text{cough}=\text{no}) = 0 \quad \text{IG}(\text{cough}) = 0.543 - \frac{5}{8} \times (0.722) = 0.0918$$

ID	Fever	Cough	Breathing issues	Infected
1	NO	YES	YES	YES
2	NO	YES	YES	YES
3	NO	YES	YES	NO

Only the feature "Cough" is left but it gives no new information, thus we leave it here.

ID	Fever	Cough	Breathing issues	Infected
1	NO	NO	NO	NO
2	YES	YES	NO	NO
3	NO	YES	NO	NO
4	YES	YES	NO	YES
5	NO	YES	NO	NO
6	YES	YES	NO	NO

$$\text{Entropy}(\text{cough}) = -\frac{5}{6} \log \frac{5}{6} - \frac{1}{6} \log \frac{1}{6} = 0.650$$

$$\text{Entropy}(\text{cough}=\text{yes}) = -\frac{4}{5} \log \frac{4}{5} - \frac{1}{5} \log \frac{1}{5} = 0.722$$

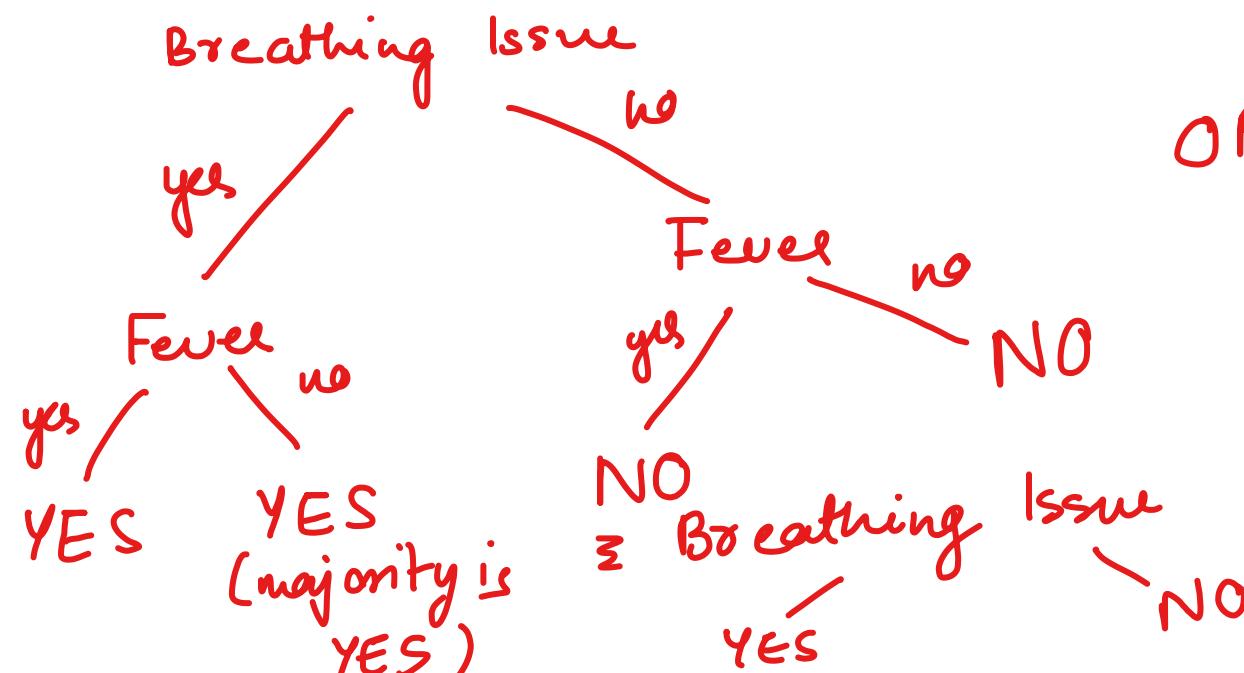
$$\text{Entropy}(\text{cough}=\text{no}) = 0$$

$$\text{IG}(\text{cough}) = 0.650 - \frac{5}{6}(0.722) = 0.0483$$

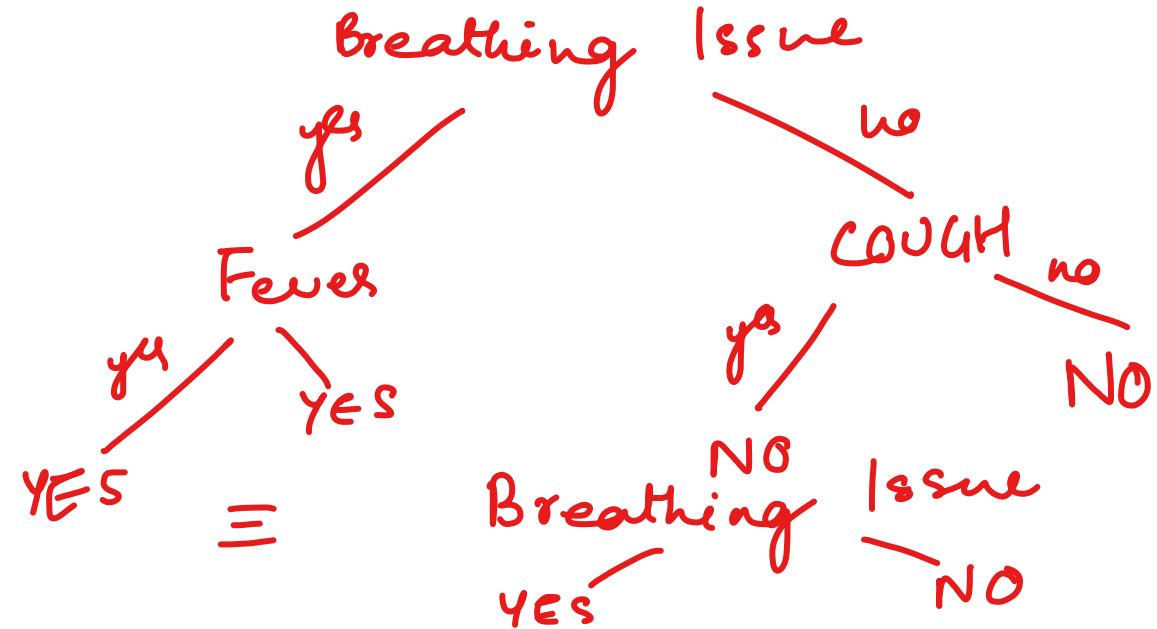
$$\text{Entropy}(\text{Fever}) = 0.650 \quad \text{Entropy}(\text{fever}=\text{yes}) = -\frac{2}{3} \log \frac{2}{3} - \frac{1}{3} \log \frac{1}{3} = 0.918$$

$$\text{IG}(\text{fever}) = 0.650 - \frac{1}{2}(0.918) = 0.191 \leftarrow \text{Highest IG}$$

$$\text{Entropy}(\text{fever}=\text{no}) = 0$$



OR



* ID3 is biased towards attributes with large value i.e. it is biased towards multi-variate attributes

C4.5

$$\text{Gain Ratio} = \frac{\text{Information Gain}(A)}{\text{SplitInfo}(A)}$$

$$\text{SplitInfo} = \sum_{j=1}^n \frac{|A_j|}{|A|} \log \frac{|A_j|}{|A|}$$

As we calculated Information Gains for attributes we can sense them - though no diff will be observed as all were bivariate attributes.

eg2

ID	Weather	Mood	Will Play?
1	Hot	Good	YES
2	Cold	Good	YES
3	Rainy	Bad	NO
4	Pleasant	Good	NO
5	Hot	Bad	YES
6	Pleasant	Bad	YES
7	Rainy	Bad	YES
8	Cold	Bad	NO

$$\text{Entropy}(\text{Weather}) = -\frac{5}{8} \log\left(\frac{5}{8}\right) - \frac{3}{8} \log\left(\frac{3}{8}\right) = 0.954$$

$$\text{Entropy}(\text{Hot}) = 0 \quad \text{Entropy}(\text{Weather} = \text{Cold}) = -\log\left(\frac{1}{2}\right) = 1$$

$$\text{Entropy}(\text{Weather} = \text{Rainy}) = 1 \quad \text{Entropy}(\text{Weather} = \text{Pleasant}) = 1$$

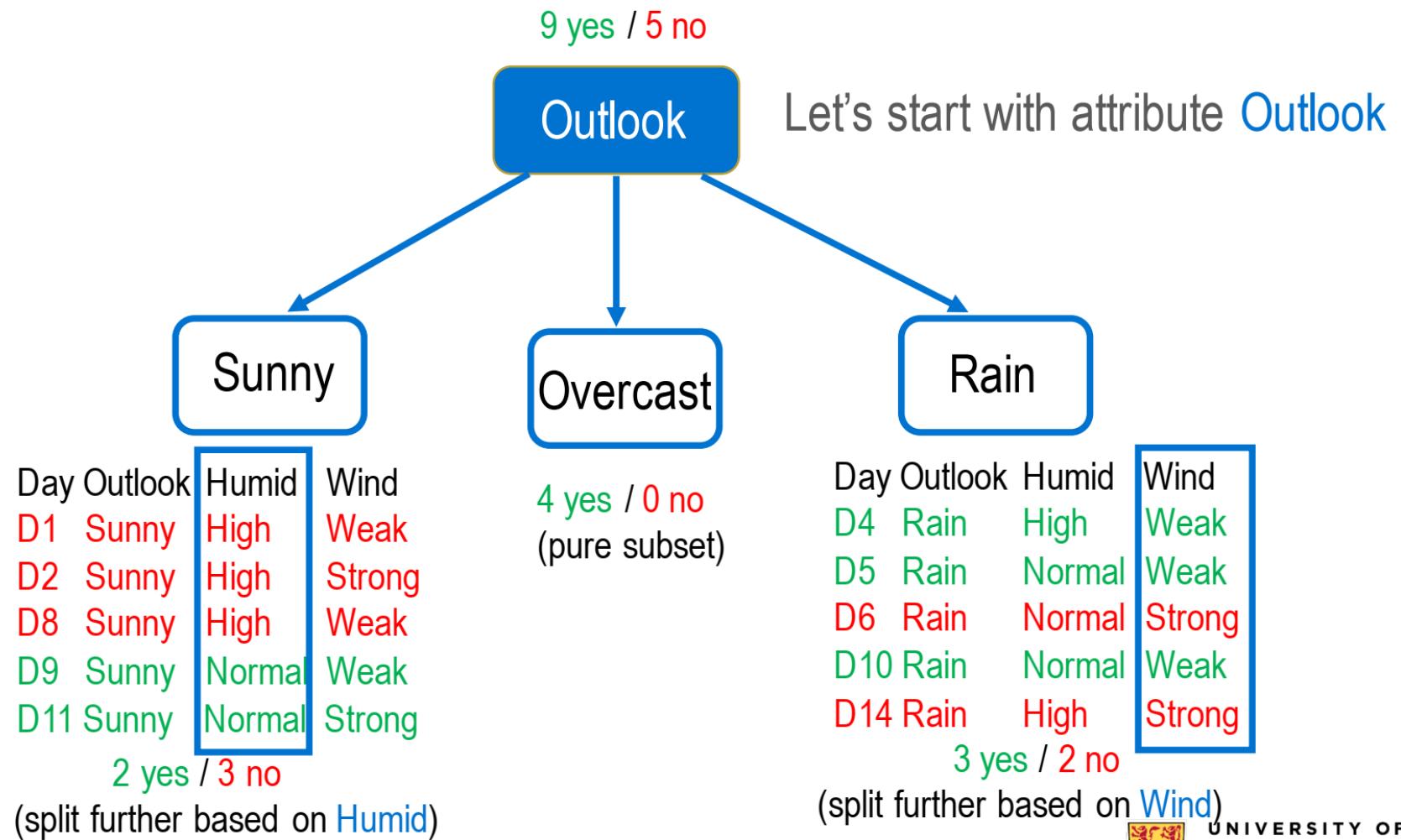
$$\text{IG}(\text{Weather}) = 0.954 - \frac{2}{8}(0) - \frac{2}{8}(1) - \frac{2}{8}(1) - \frac{2}{8}(1) = 0.204$$

$$\text{splitInfo} = \left[\frac{2}{8} \log\left(\frac{2}{8}\right) + \frac{2}{8} \log\left(\frac{2}{8}\right) + \frac{2}{8} \log\left(\frac{2}{8}\right) + \frac{2}{8} \log\left(\frac{2}{8}\right) \right] (-1) = -\log\left(\frac{2}{8}\right) = 0.602$$

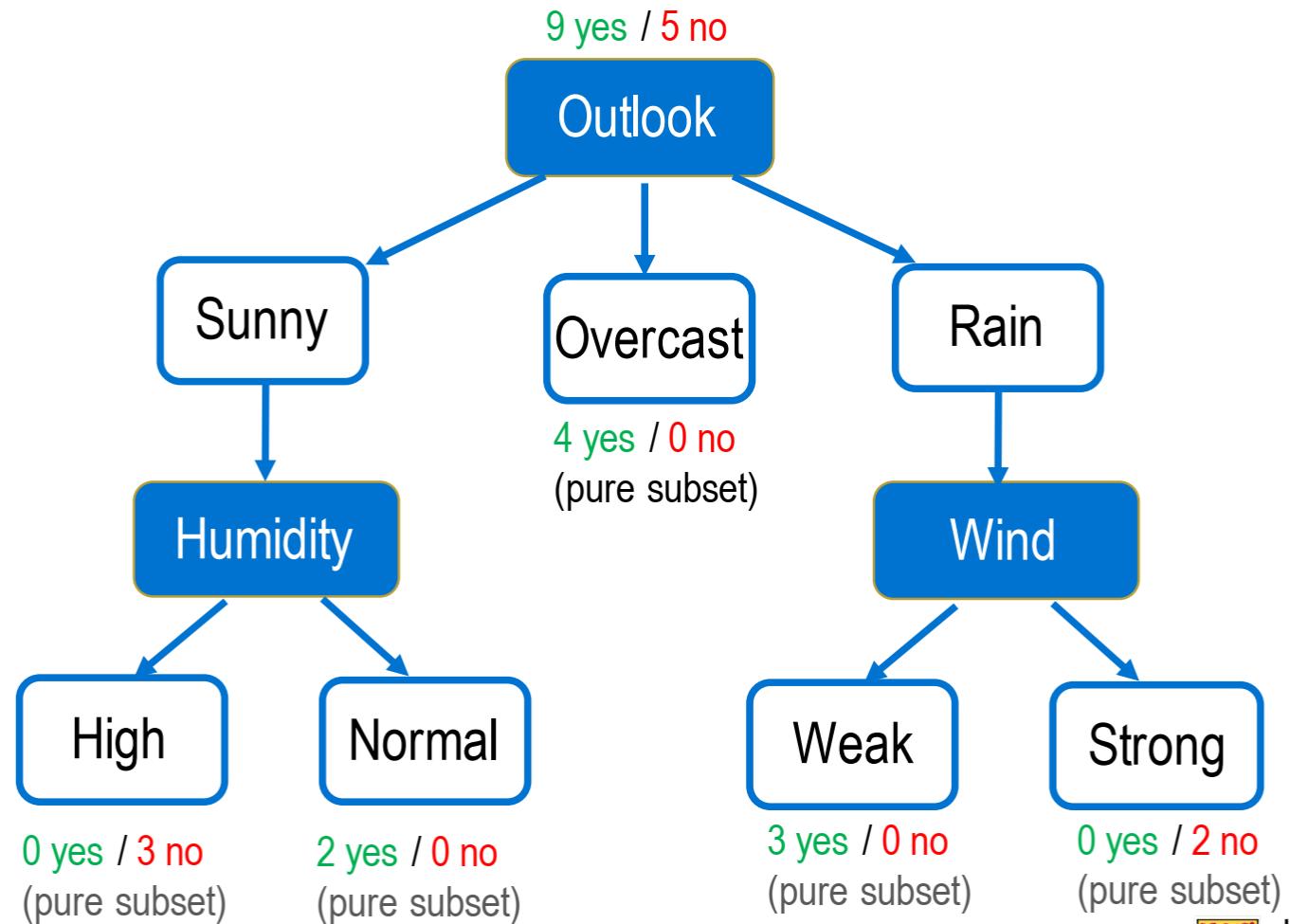
$$\text{Gain Ratio} = \frac{0.204}{0.602} = \underline{0.339}$$

(Rest Continued at End)

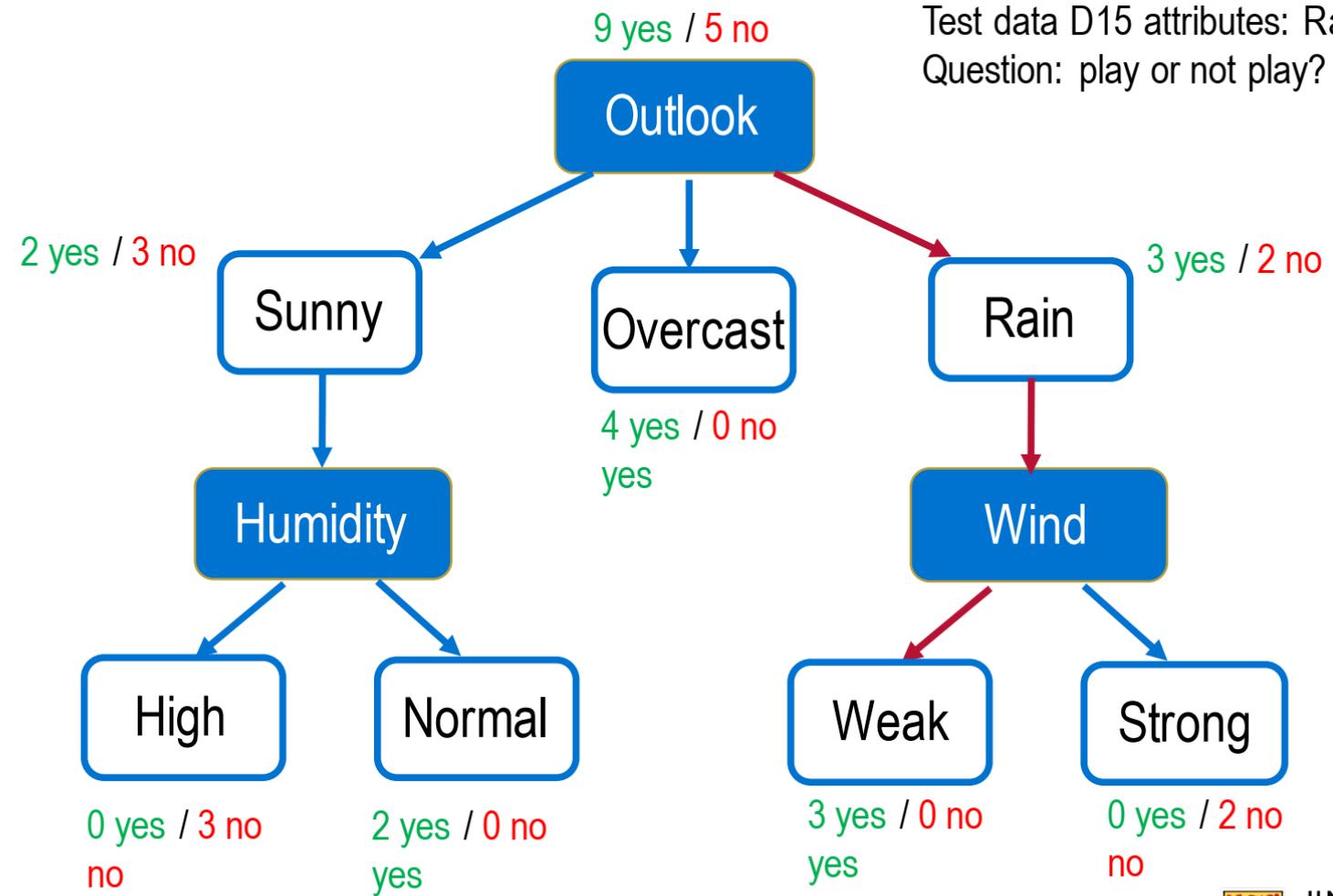
Split training data



Split training data



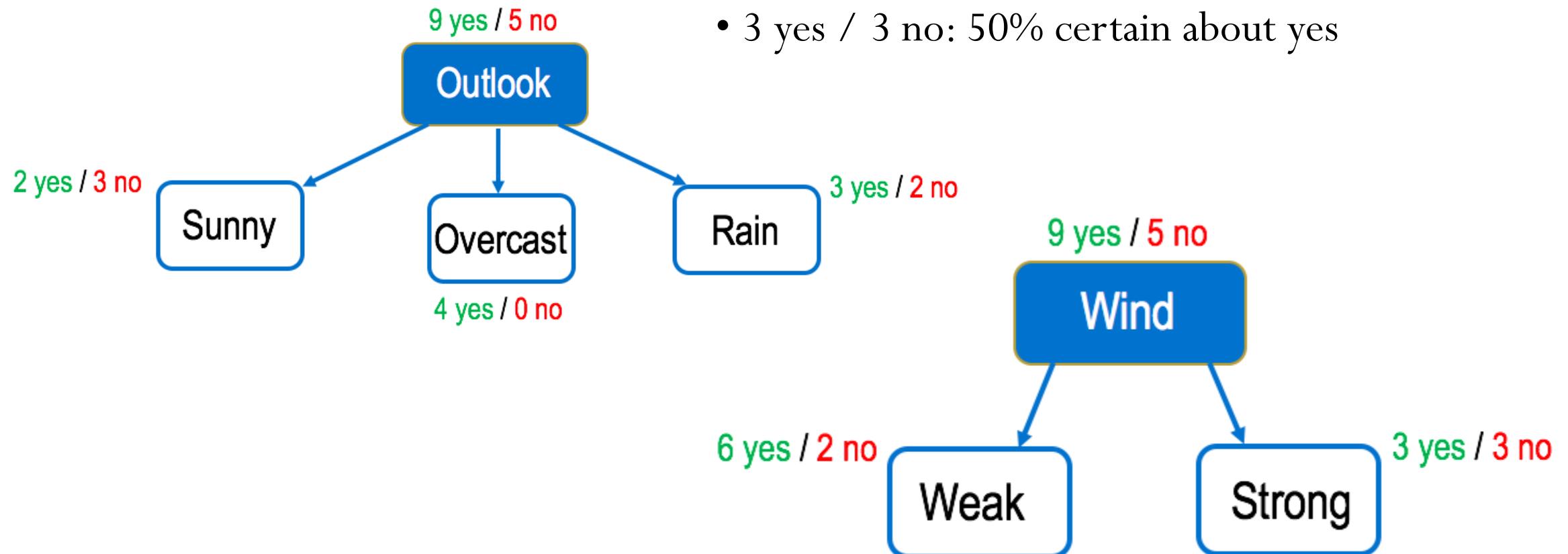
Split training data



How to select an attribute?

We hope that uncertainty can be reduced after the split

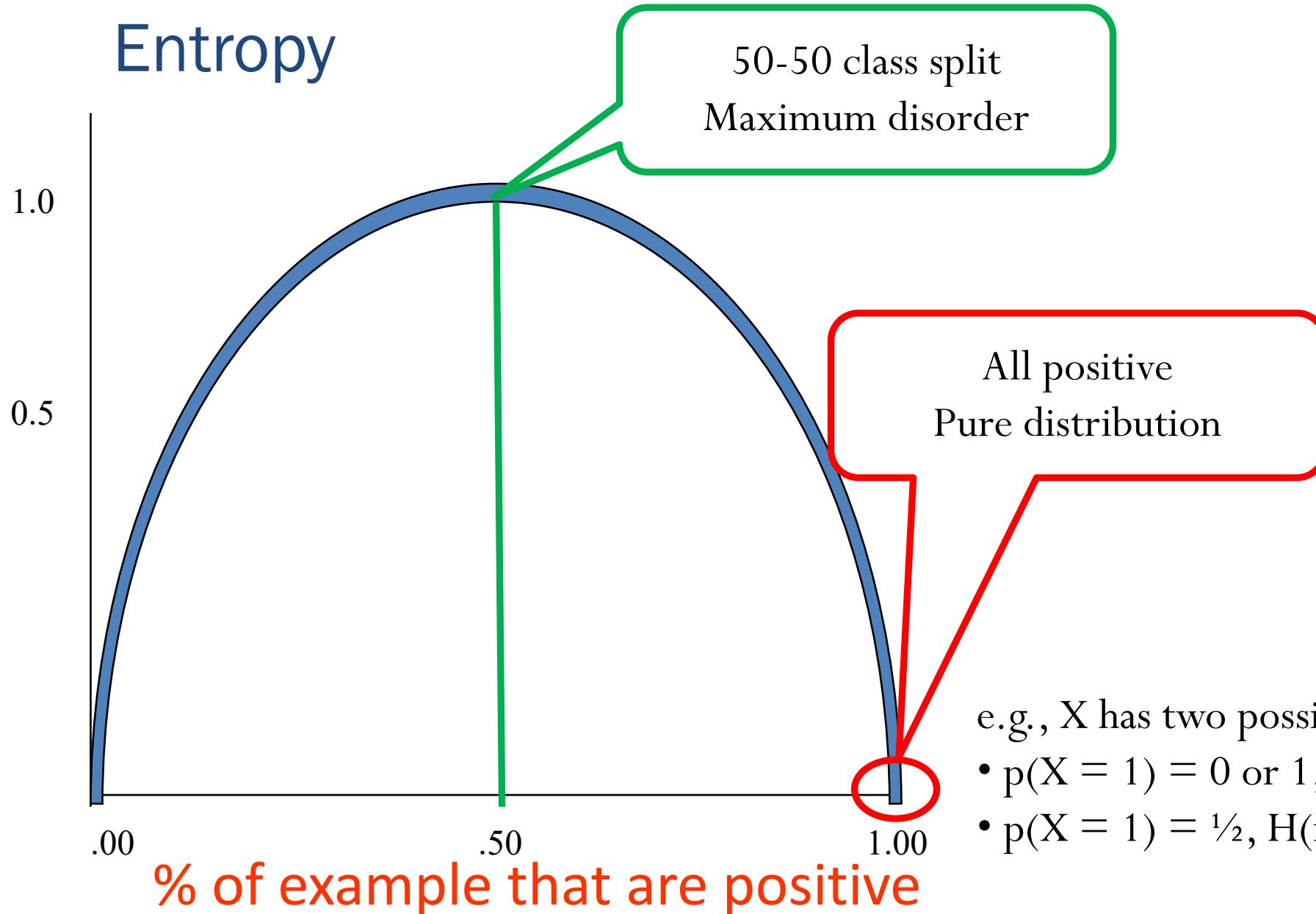
- 4 yes / 0 no: 100 % certain about yes
- 3 yes / 3 no: 50% certain about yes



ID3 (Iterative Dichotomiser 3) algorithm

- ID3 (node, {training data}) # Generate a DT
 1. Pick an attribute (A) with the maximum information gain for the considered training data
 2. For each value of A, create new child node
 3. Split training data to child nodes
 4. Check subset for each child node
 - If subset is pure: stop
 - Else: ID3 (child node, {subset data})

Entropy



- e.g., X has two possible values 0 and 1
- $p(X = 1) = 0$ or 1, $H(x) = 0$ bits (minimum)
- $p(X = 1) = \frac{1}{2}$, $H(x) = 1$ bits (maximum)

Entropy (disorder) is bad

Homogeneity is good

- Let S be a set of examples
- $\text{Entropy}(S) = -P \log_2(P) - N \log_2(N)$
 - P is proportion of pos example
 - N is proportion of neg examples
 - $0 \log 0 == 0$
- Example: S has 9 pos and 5 neg
$$\begin{aligned}\text{Entropy}([9+, 5-]) &= -(9/14) \log_2(9/14) - (5/14) \log_2(5/14) \\ &= 0.940\end{aligned}$$

Information gain

Expected drop in entropy after split

$$\text{Gain}(S, A) = H(S) - \sum_{V \in \text{Values}(A)} \frac{|S_V|}{|S|} H(S_V)$$

uncertainty before split

uncertainty after split

- A: attribute
- S: set of training examples
- V: possible values of attribute A
- S_V : set of training examples with the value of attribute A = V
- Subsets with more examples have a larger effect

Maximizing $\text{Gain}(S, A)$ is equivalent to minimizing uncertainty after split

Gain of Splitting on Wind

Values(wind)=weak, strong

$$S = [9+, 5-]$$

$$S_{\text{weak}} = [6+, 2-]$$

$$S_s = [3+, 3-]$$

Gain(S, wind)

$$= \text{Entropy}(S) - \sum (|S_v| / |S|) \text{Entropy}(S_v)$$

$$v \in \{\text{weak}, s\}$$

$$= \text{Entropy}(S) - 8/14 \text{Entropy}(S_{\text{weak}})$$

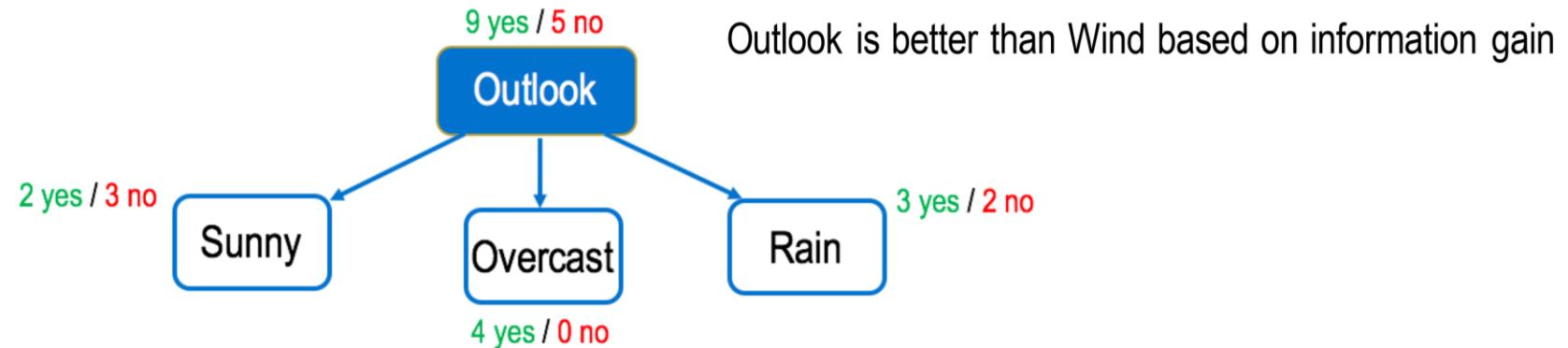
$$- 6/14 \text{Entropy}(S_s)$$

$$= 0.940 - (8/14) 0.811 - (6/14) 1.00$$

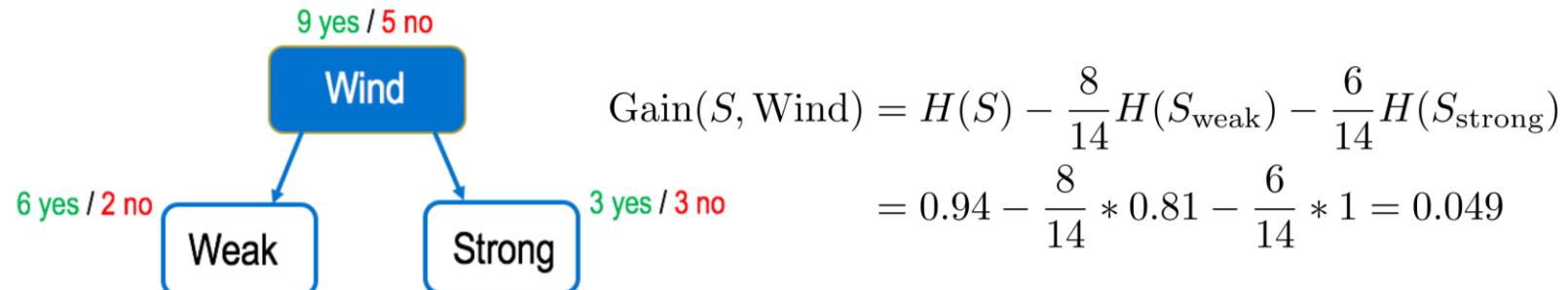
$$= .048$$

Day	Wind	Tennis?
d1	weak	n
d2	s	n
d3	weak	yes
d4	weak	yes
d5	weak	yes
d6	s	n
d7	s	yes
d8	weak	n
d9	weak	yes
d10	weak	yes
d11	s	yes
d12	s	yes
d13	weak	yes
d14	s	n

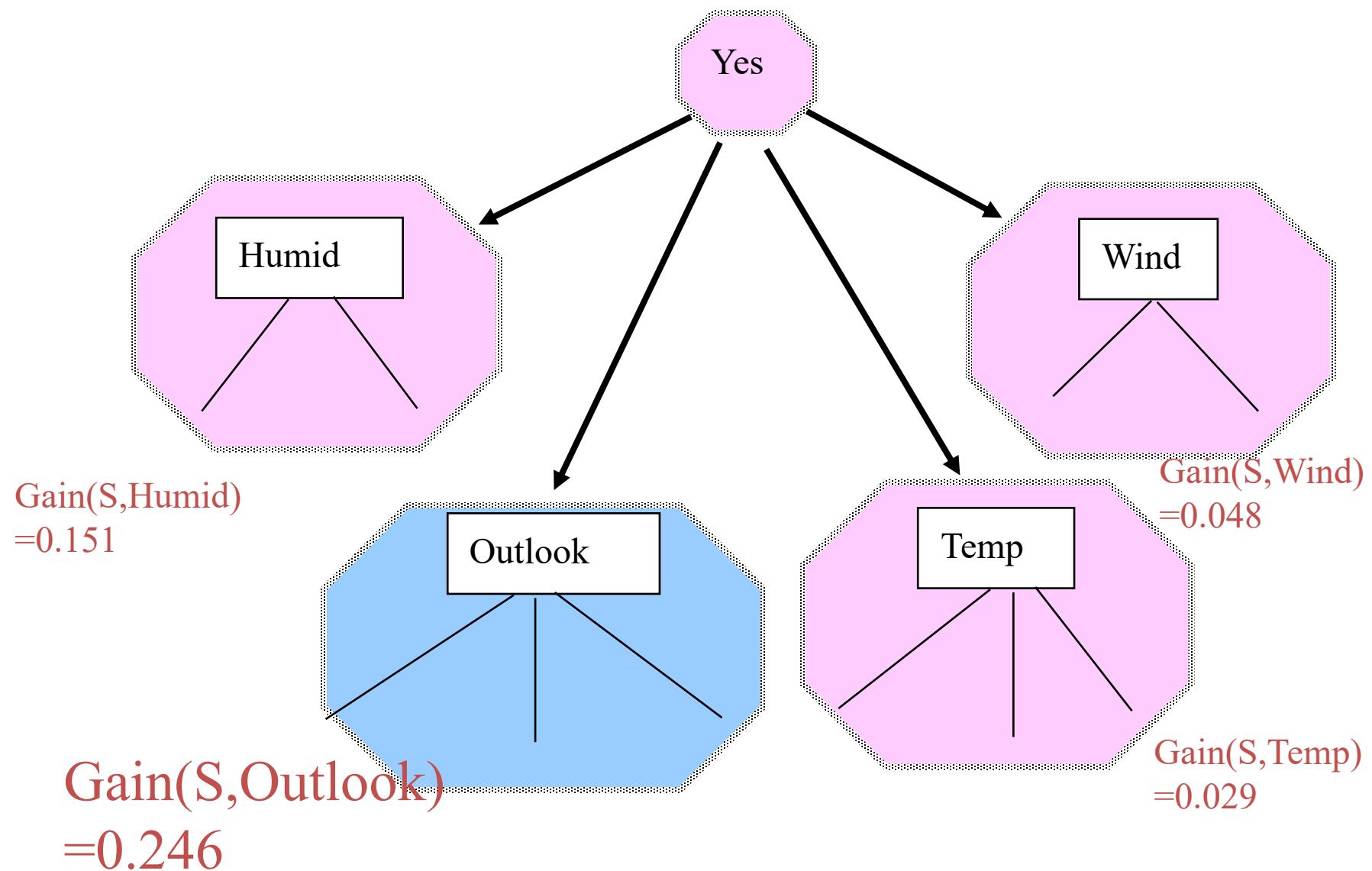
Information gain



$$\begin{aligned} \text{Gain}(S, \text{Outlook}) &= H(S) - \frac{5}{14}H(S_{\text{sunny}}) - \frac{4}{14}H(S_{\text{overcast}}) - \frac{5}{14}H(S_{\text{rain}}) \\ &= 0.94 - \frac{5}{14} * 0.97 - \frac{4}{14} * 0 - \frac{5}{14} * 0.97 = 0.25 \end{aligned}$$



Evaluating Attributes



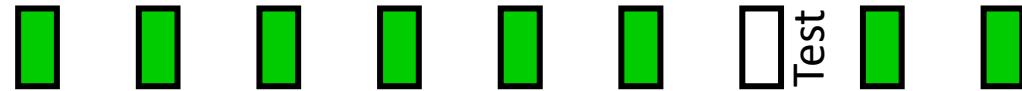
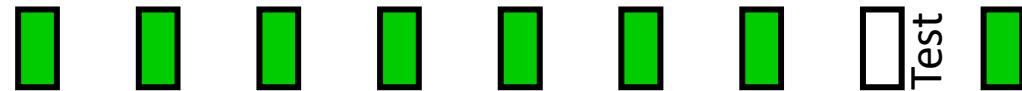
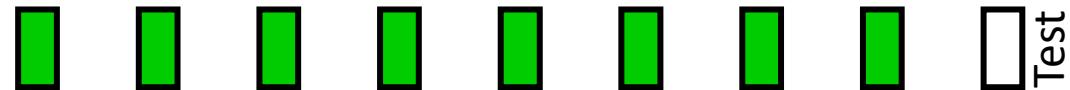
Issues

- Missing data — *cannot handle*
- Real-valued attributes — *Deals only with categorical data*
- Many-valued features — *biased to multi-variate features*
- Evaluation
- Overfitting

Evaluation: Cross Validation

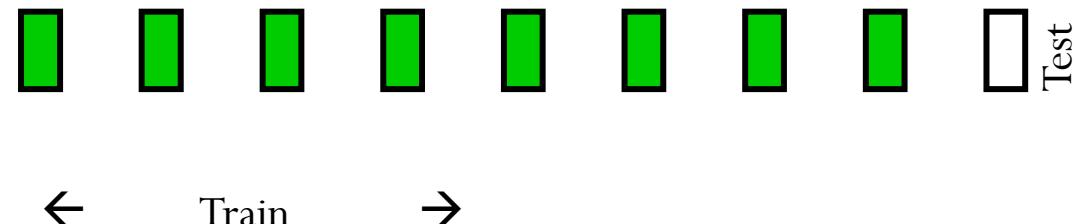
- Partition examples into k disjoint sets
- Now create k training sets
 - Each set is union of all equiv classes *except one*
 - So each set has $(k-1)/k$ of the original training data

← Train →



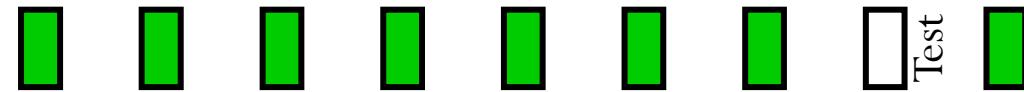
Cross validation

- Partition examples into k disjoint sets
- Now create k training sets
 - Each set is union of all equiv classes *except one*
 - So each set has $(k-1)/k$ of the original training data



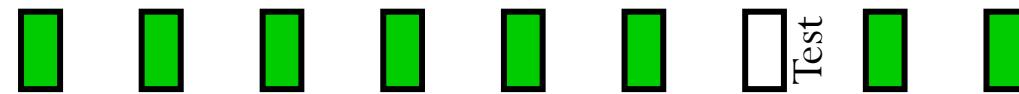
Cross Validation

- Partition examples into k disjoint sets
- Now create k training sets
 - Each set is union of all equiv classes *except one*
 - So each set has $(k-1)/k$ of the original training data



Cross Validation

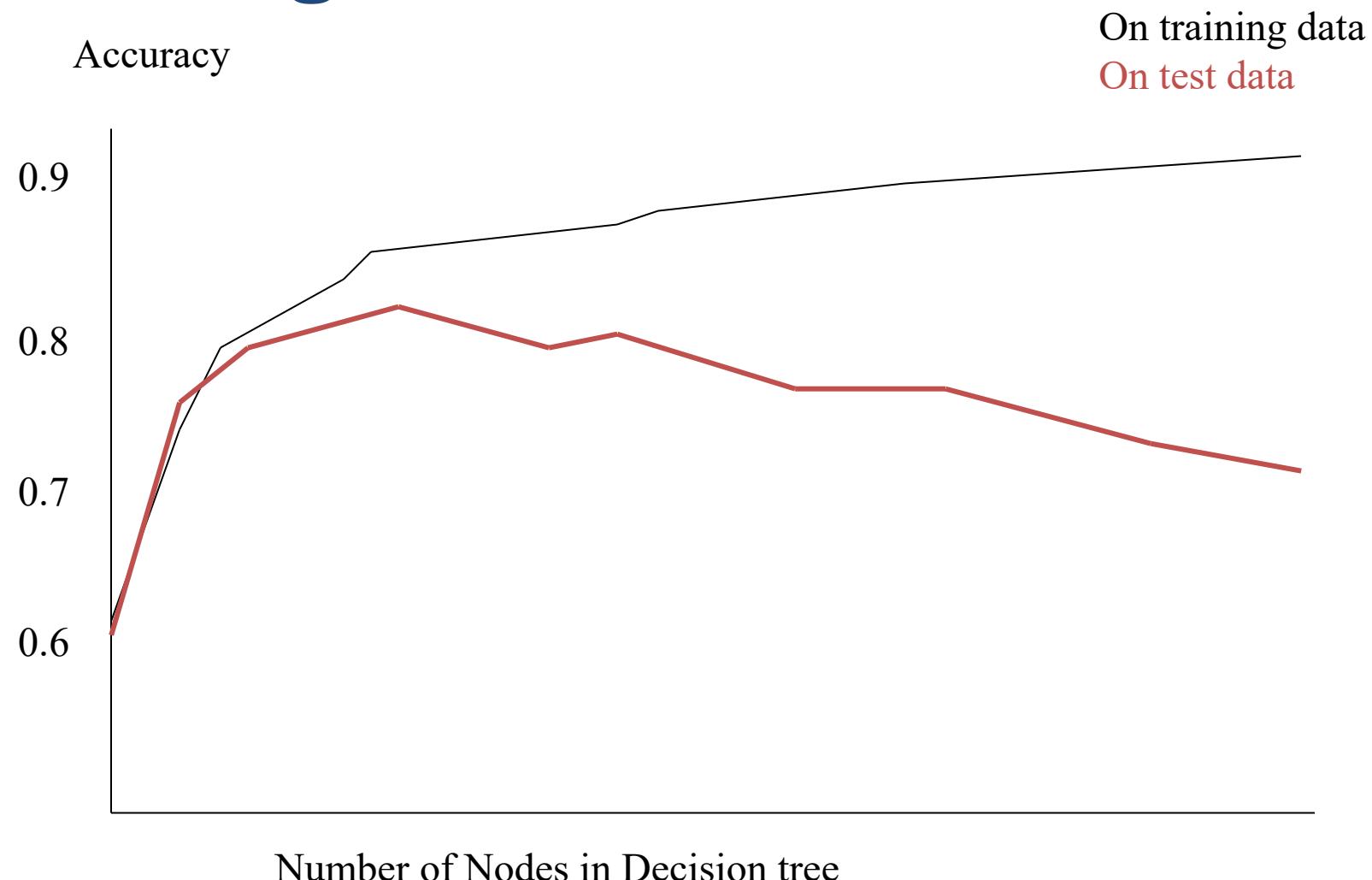
- Partition examples into k disjoint sets
- Now create k training sets
 - Each set is union of all equiv classes *except one*
 - So each set has $(k-1)/k$ of the original training data



Cross-Validation (2)

- Training and validation sets
 - training set is used to build the tree
 - a separate validation set is used to evaluate the accuracy over subsequent data, and to evaluate the impact of pruning
 - justification: validation set is unlikely to exhibit the same noise and spurious correlation
 - rule of thumb: 2/3 to the training set, 1/3 to the validation set
- Leave-one-out
 - Use if < 100 examples (rough estimate)
 - Hold out one example, train on remaining examples
- M of N fold
 - Repeat M times
 - Divide data into N folds, do N fold cross-validation

Overfitting



Overfitting Definition

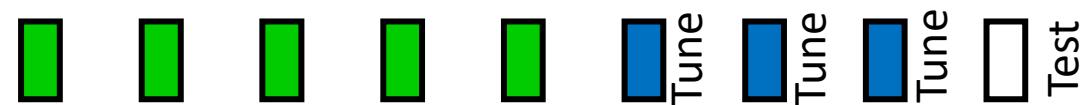
- DT is *overfit* when exists another DT' and
 - DT has *smaller* error on training examples, but
 - DT has *bigger* error on test examples
- Causes of overfitting
 - Noisy data, or
 - Training set is too small
- Solutions
 - Reduced error pruning
 - Early stopping
 - Rule post pruning

Avoid Overfitting

- How to avoid overfitting?
 - Stop growing the tree
 - before it perfectly classifies the training data
 - when data split is not statistically significant
 - Allow overfitting, but post-prune the tree
 - Grow full tree, then post-prune
 - Acquire more training data
 - Remove irrelevant attributes (manual process – not always possible)
- How to select “best” tree:
 - Measure performance over training data
 - Measure performance over separate validation data set
 - Add complexity penalty to performance measure (heuristic: simpler is better)

Reduced Error Pruning

- Split data into train and validation set



- Repeat until pruning is harmful
 - Remove each subtree and replace it with majority class and evaluate on validation set
 - Remove subtree that leads to largest gain in accuracy

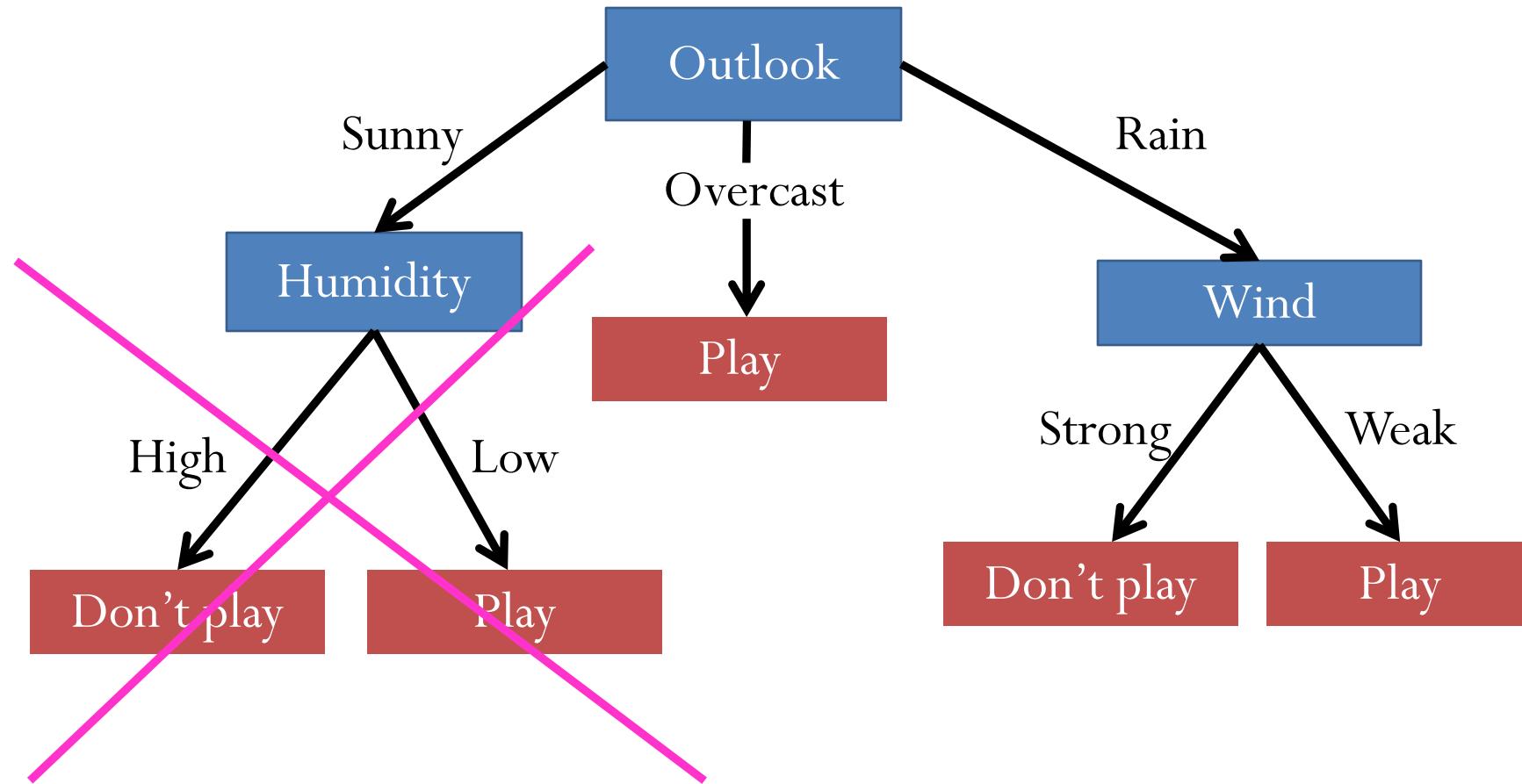
Reduced-Error Pruning

Split data into *training* and *validation* set

Do until further pruning is harmful:

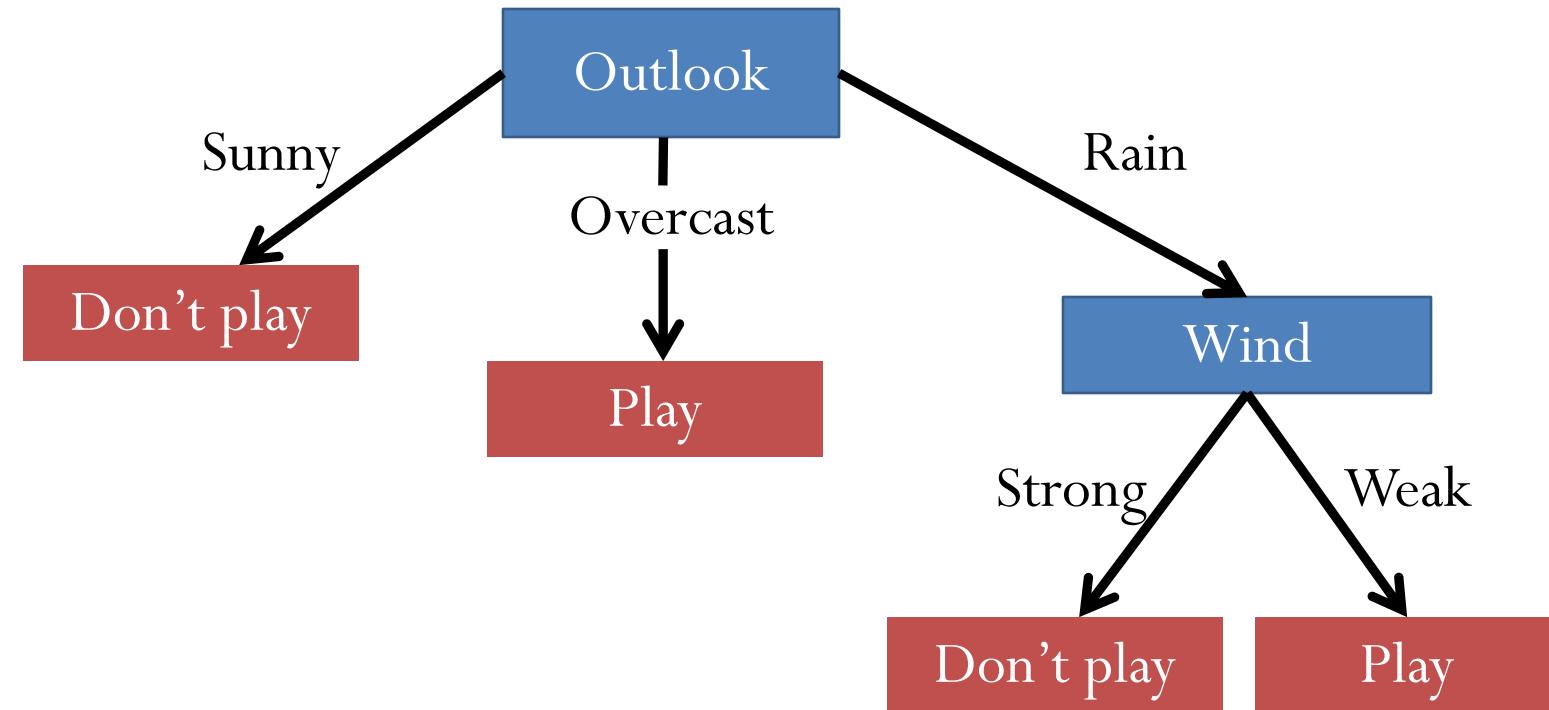
1. Evaluate impact on *validation* set of pruning each possible node (plus those below it)
2. Greedily remove the one that most improves *validation* set accuracy

Reduced Error Pruning Example



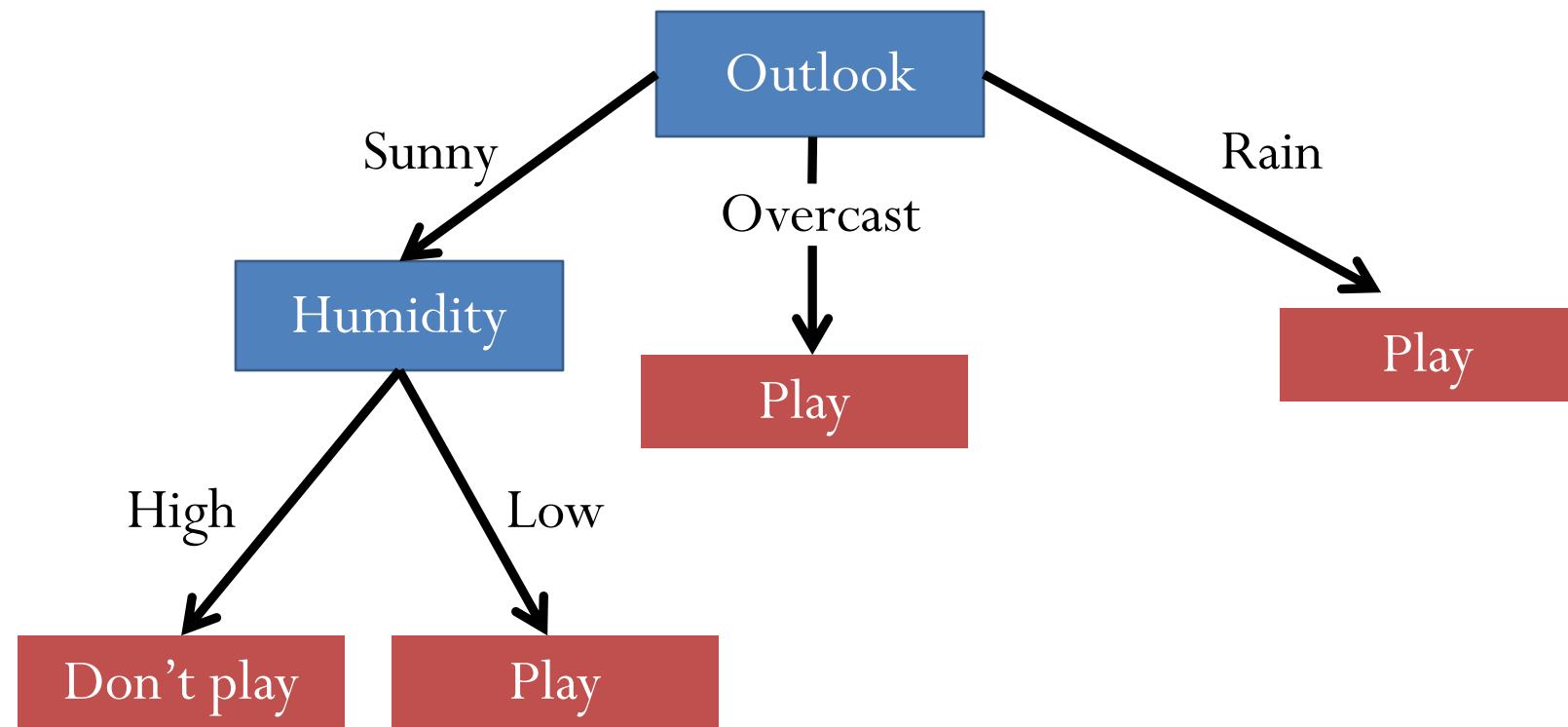
Validation set accuracy = 0.75

Reduced Error Pruning Example



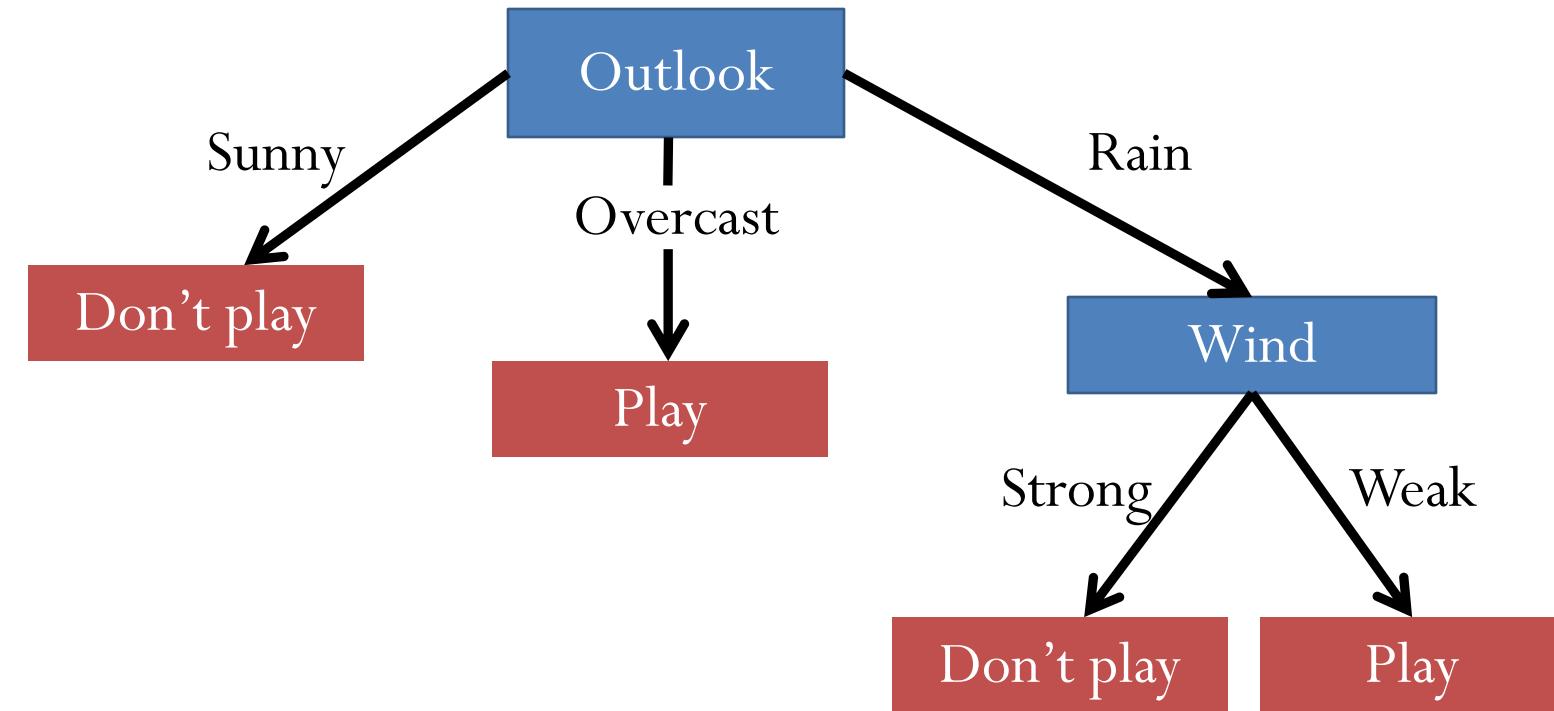
Validation set accuracy = 0.80

Reduced Error Pruning Example



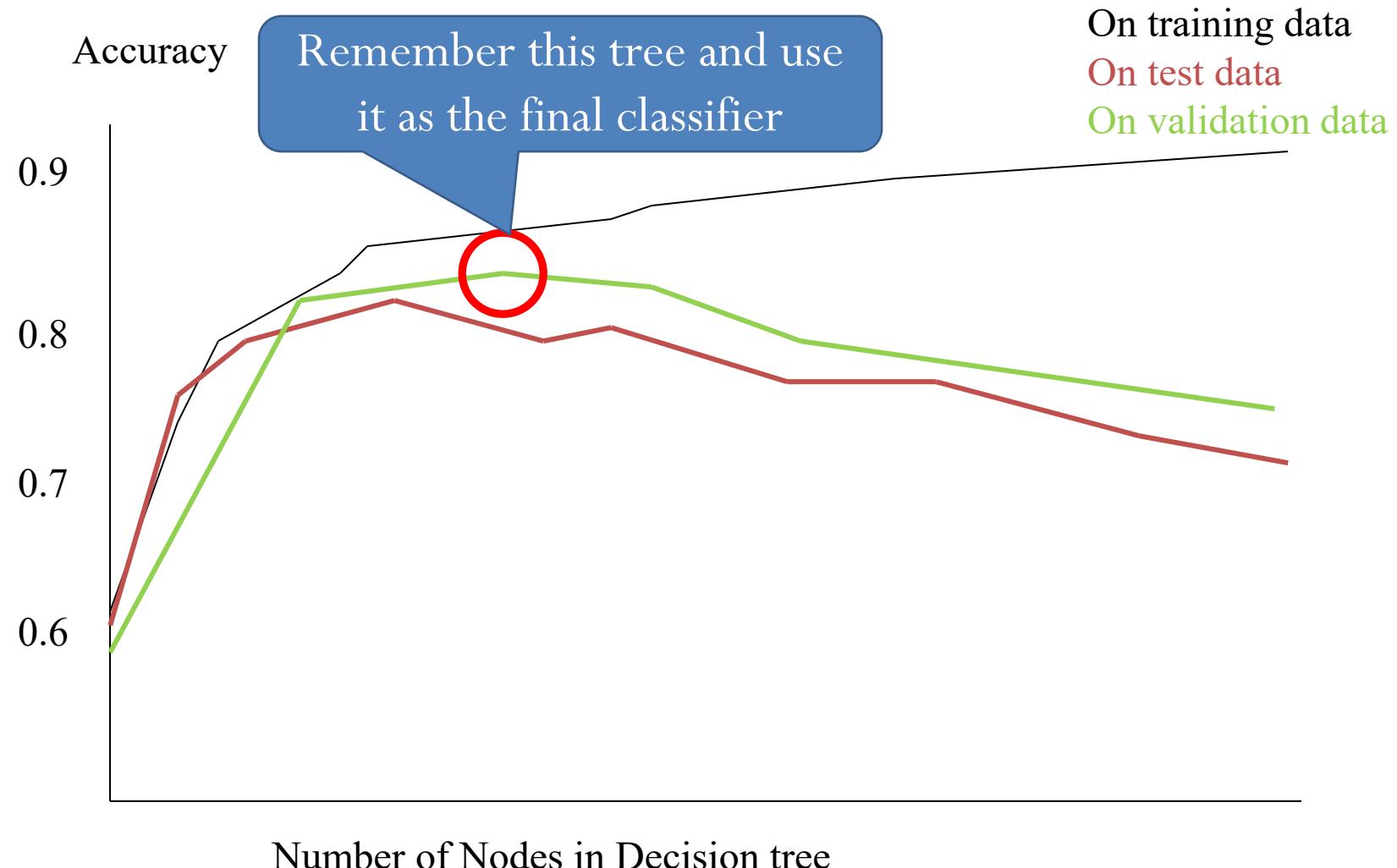
Validation set accuracy = 0.70

Reduced Error Pruning Example



Use this as final tree

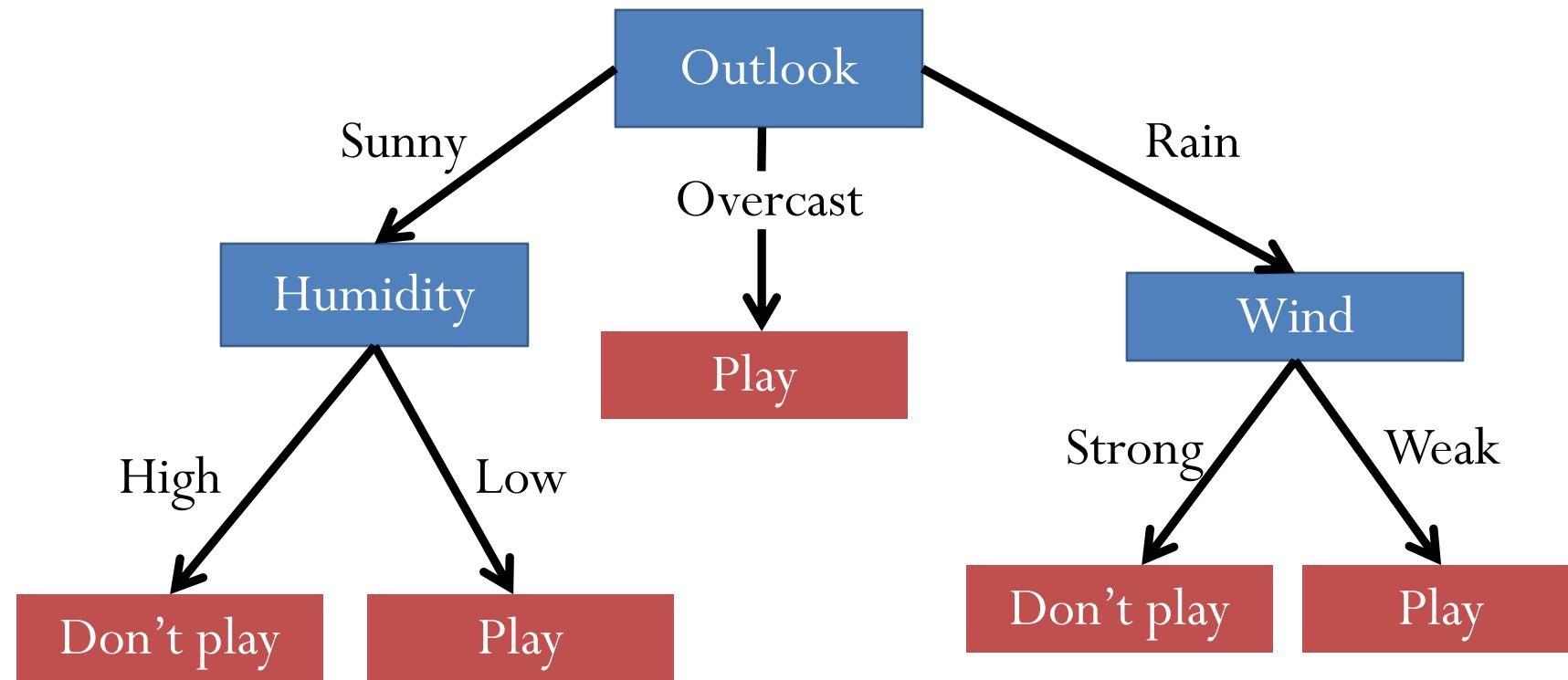
Early Stopping



Post Rule Pruning

- Split data into train and validation set
- Prune each rule independently
 - Remove each pre-condition and evaluate accuracy
 - Pick pre-condition that leads to largest improvement in accuracy
- Note: ways to do this using training data and statistical tests
 1. Convert tree to equivalent set of rules
 2. Prune each rule independently of others
 3. Sort final rules into desired sequence for use

Conversion to Rule



$\text{Outlook} = \text{Sunny} \wedge \text{Humidity} = \text{High} \Rightarrow \text{Don't play}$

$\text{Outlook} = \text{Sunny} \wedge \text{Humidity} = \text{Low} \Rightarrow \text{Play}$

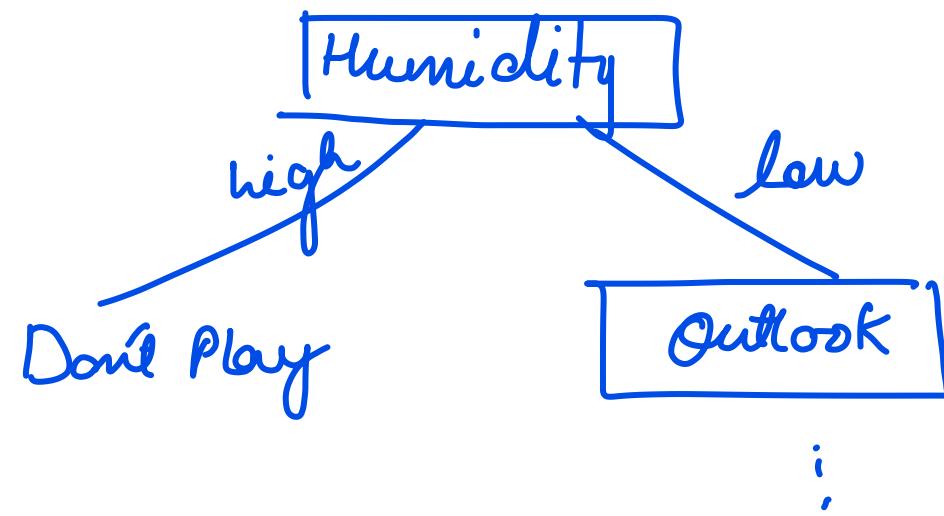
$\text{Outlook} = \text{Overcast} \Rightarrow \text{Play}$

...

IF $(Outlook = Sunny) \text{ AND } (Humidity = High)$
THEN $PlayTennis = No$

IF $(Outlook = Sunny) \text{ AND } (Humidity = Normal)$
THEN $PlayTennis = Yes$

...



Example

$\text{Outlook} = \text{Sunny} \wedge \text{Humidity} = \text{High} \Rightarrow \text{Don't play}$

Validation set accuracy = 0.68

$\rightarrow \text{Outlook} = \text{Sunny} \Rightarrow \text{Don't play}$ Validation set accuracy = 0.65

$\rightarrow \text{Humidity} = \text{High} \Rightarrow \text{Don't play}$

Validation set accuracy = 0.75

Thus, we remove the pre-condition of $\text{outlook} = \text{Sunny}$

Keep this rule

Overfitting 2

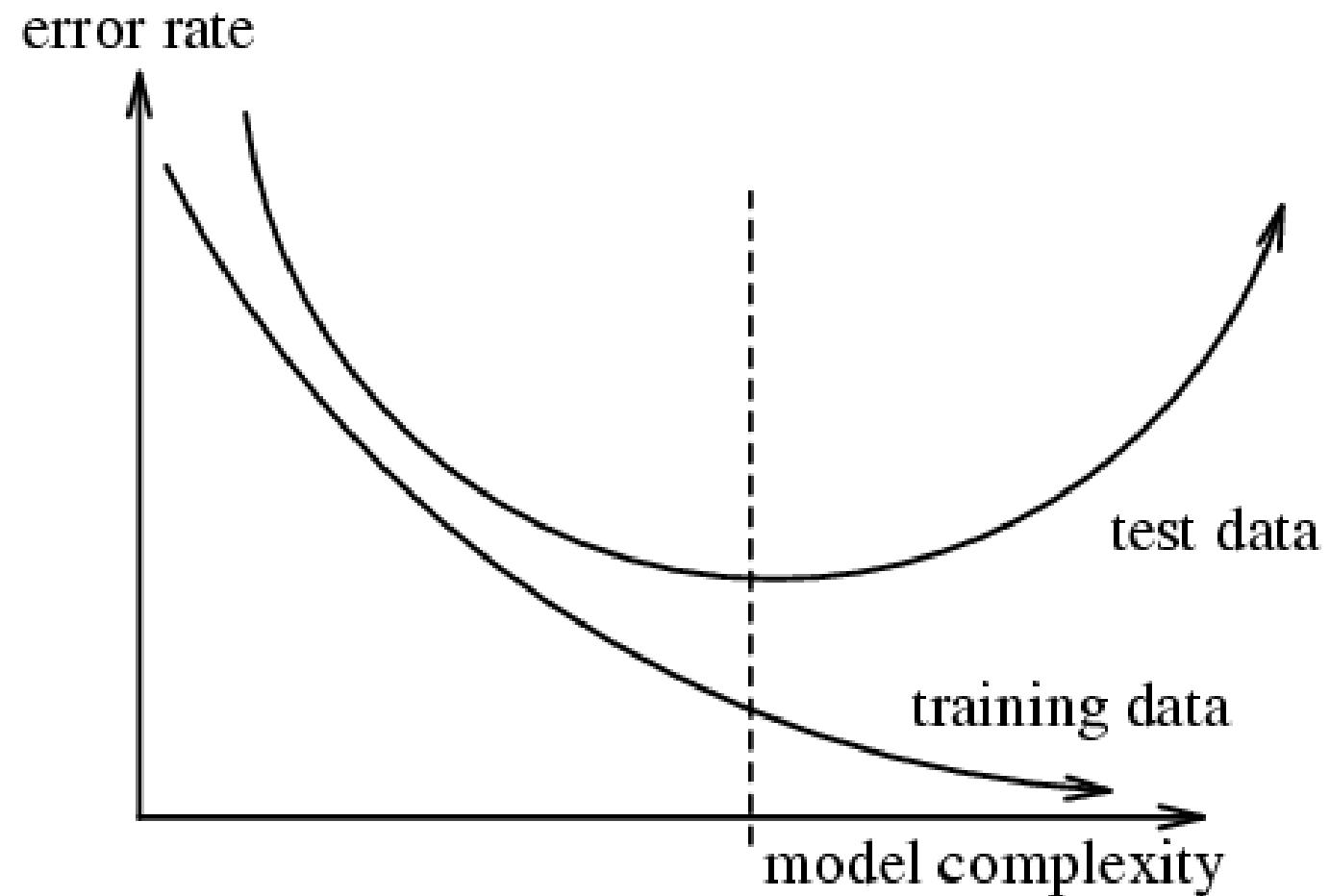
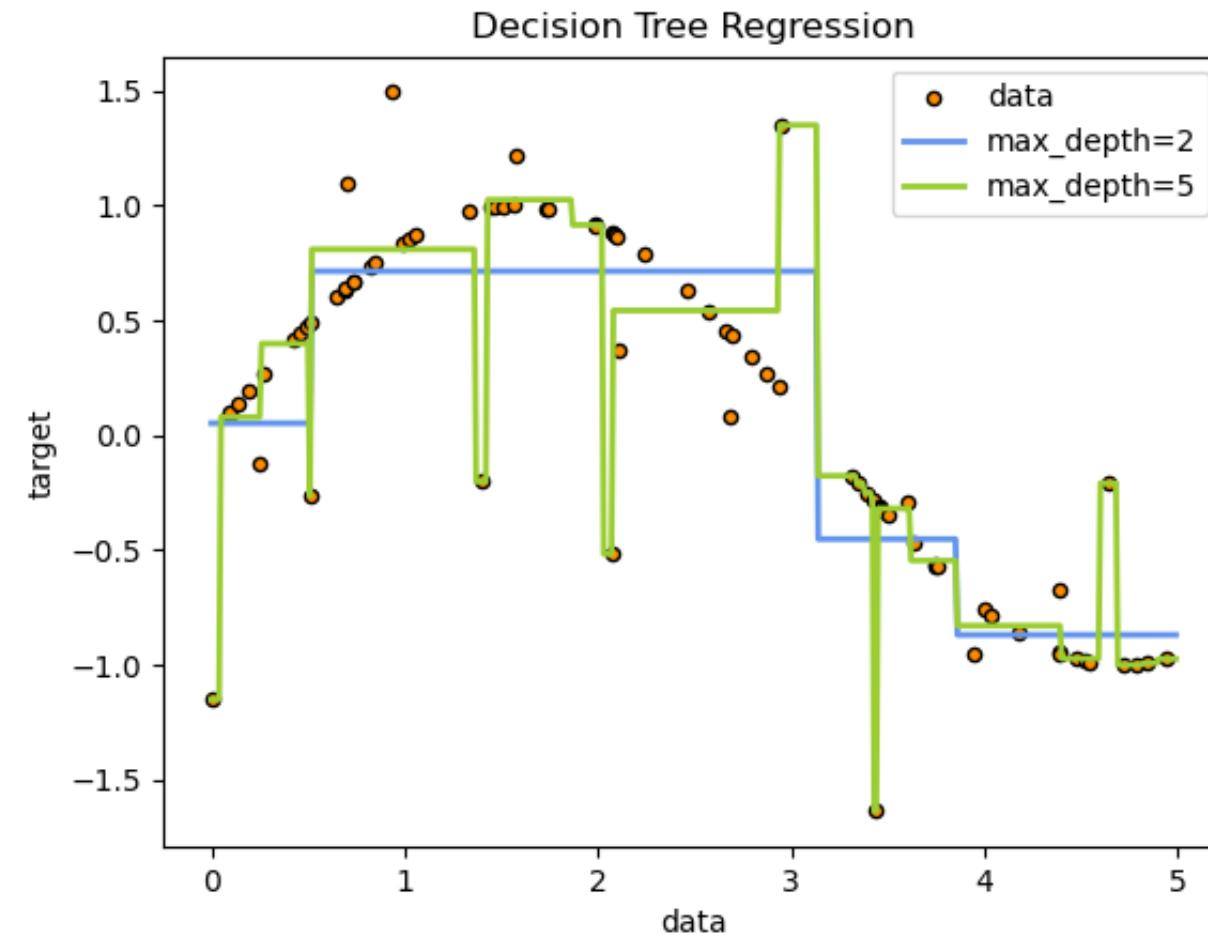


Figure from w.w.cohen

Scikit Learn on Decision Trees

- Regression



Scikit Learn on Decision Trees

Given training vectors $x_i \in R^n$, $i=1, \dots, l$ and a label vector $y \in R^l$, a decision tree recursively partitions the feature space such that the samples with the same labels or similar target values are grouped together.

Let the data at node m be represented by Q_m with n_m samples. For each candidate split $\theta = (j, t_m)$ consisting of a feature j and threshold t_m , partition the data into $Q_m^{left}(\theta)$ and $Q_m^{right}(\theta)$ subsets

$$Q_m^{left}(\theta) = \{(x, y) | x_j \leq t_m\}$$
$$Q_m^{right}(\theta) = Q_m \setminus Q_m^{left}(\theta)$$

The quality of a candidate split of node m is then computed using an impurity function or loss function $H()$, the choice of which depends on the task being solved (classification or regression)

$$G(Q_m, \theta) = \frac{n_m^{left}}{n_m} H(Q_m^{left}(\theta)) + \frac{n_m^{right}}{n_m} H(Q_m^{right}(\theta))$$

Select the parameters that minimises the impurity

$$\theta^* = \operatorname{argmin}_{\theta} G(Q_m, \theta)$$

Recurse for subsets $Q_m^{left}(\theta^*)$ and $Q_m^{right}(\theta^*)$ until the maximum allowable depth is reached, $n_m < \min_{samples}$ or $n_m = 1$.

Scikit Learn on Decision Trees

- Gini and Log Loss or Entropy:

If a target is a classification outcome taking on values $0, 1, \dots, K-1$, for node m , let

$$p_{mk} = \frac{1}{n_m} \sum_{y \in Q_m} I(y = k)$$

be the proportion of class k observations in node m . If m is a terminal node, `predict_proba` for this region is set to p_{mk} . Common measures of impurity are the following.

Gini:

$$H(Q_m) = \sum_k p_{mk}(1 - p_{mk})$$

Log Loss or Entropy:

$$H(Q_m) = - \sum_k p_{mk} \log(p_{mk})$$

what we saw
in ID3 as well

→ CART Algorithm (Classification And Regression Tree)

Cart Algorithms can be used for both classification and regression tasks. It uses Gini Index to divide a node into its subnodes. It only gives 2 branches for each decision node i.e. it creates Binary Tree.

$$\text{Gini Index} = \sum_{i=0}^c P_i (1 - P_i) = 1 - \sum_{i=1}^c P_i^2 \quad \text{where } P_i \text{ is the relative frequency of class } i.$$

→ this GI is for the initial dataset

After splitting the dataset on the basis of a feature and a threshold, we calculate gini-index for that attribute as

$$\text{gini}_A(D) = \frac{|D_1|}{|D|} \text{gini}(D_1) + \frac{|D_2|}{|D|} \text{gini}(D_2)$$

$$\text{Reduction in Impurity} = \text{gini}(D) - \text{gini}_A(D) = \Delta \text{gini}(A)$$

The attribute that maximizes reduction in impurity or in other words attribute with minimum $gini_A(D)$ is selected for splitting.

ID	Weather	Mood	Will Play?
1	Hot	Good	YES
2	Cold	Good	YES
3	Rainy	Bad	NO
4	Pleasant	Good	NO
5	Hot	Bad	YES
6	Pleasant	Bad	YES
7	Rainy	Bad	YES
8	Cold	Bad	NO

$$gini(D) = 1 - \left[\left(\frac{5}{8} \right)^2 + \left(\frac{3}{8} \right)^2 \right] = \frac{64 - 25 - 9}{64} = \frac{30}{64}$$

Weather = { Hot, Cold, Rainy, Pleasant }

S1 { Hot or Cold or Rainy }

$$gini_1(D_1) = \frac{6}{8} \left(1 - \left(\frac{1}{6} \right)^2 - \left(\frac{2}{6} \right)^2 \right) + \frac{2}{8} \left(1 - \left(\frac{1}{2} \right)^2 - \left(\frac{1}{2} \right)^2 \right) \\ = 0.458$$

S2 { Hot or Cold or Pleasant }

$$gini_2(D_1) = \frac{6}{8} \left(1 - \left(\frac{4}{6} \right)^2 - \left(\frac{2}{6} \right)^2 \right) + \frac{2}{8} \left(\frac{1}{2} \right) \\ = 0.458$$

S3 { Hot or Rainy or Pleasant }

$$gini_3(D_1) = \frac{6}{8} \left(1 - \left(\frac{4}{6} \right)^2 - \left(\frac{2}{6} \right)^2 \right) + \frac{2}{8} \left(\frac{1}{2} \right) = 0.458$$

$$S_4 \quad \{ \text{cold or Rainy or Pleasant} \}$$

$$\text{gini}_4(D_1) = \frac{6}{8} \left(1 - \left(\frac{3}{6}\right)^2 - \left(\frac{3}{6}\right)^2 \right) + \frac{2}{8} (0) = 0.375 \quad \text{gini minimum}$$

$$S_5 \quad \{ \text{Hot or Cold} \} \quad \text{gini}_5(D_1) = \frac{4}{8} \left(1 - \left(\frac{3}{4}\right)^2 - \left(\frac{1}{4}\right)^2 \right) + \frac{4}{8} \left(1 - \left(\frac{2}{4}\right)^2 - \left(\frac{2}{4}\right)^2 \right)$$

$$= 0.4375$$

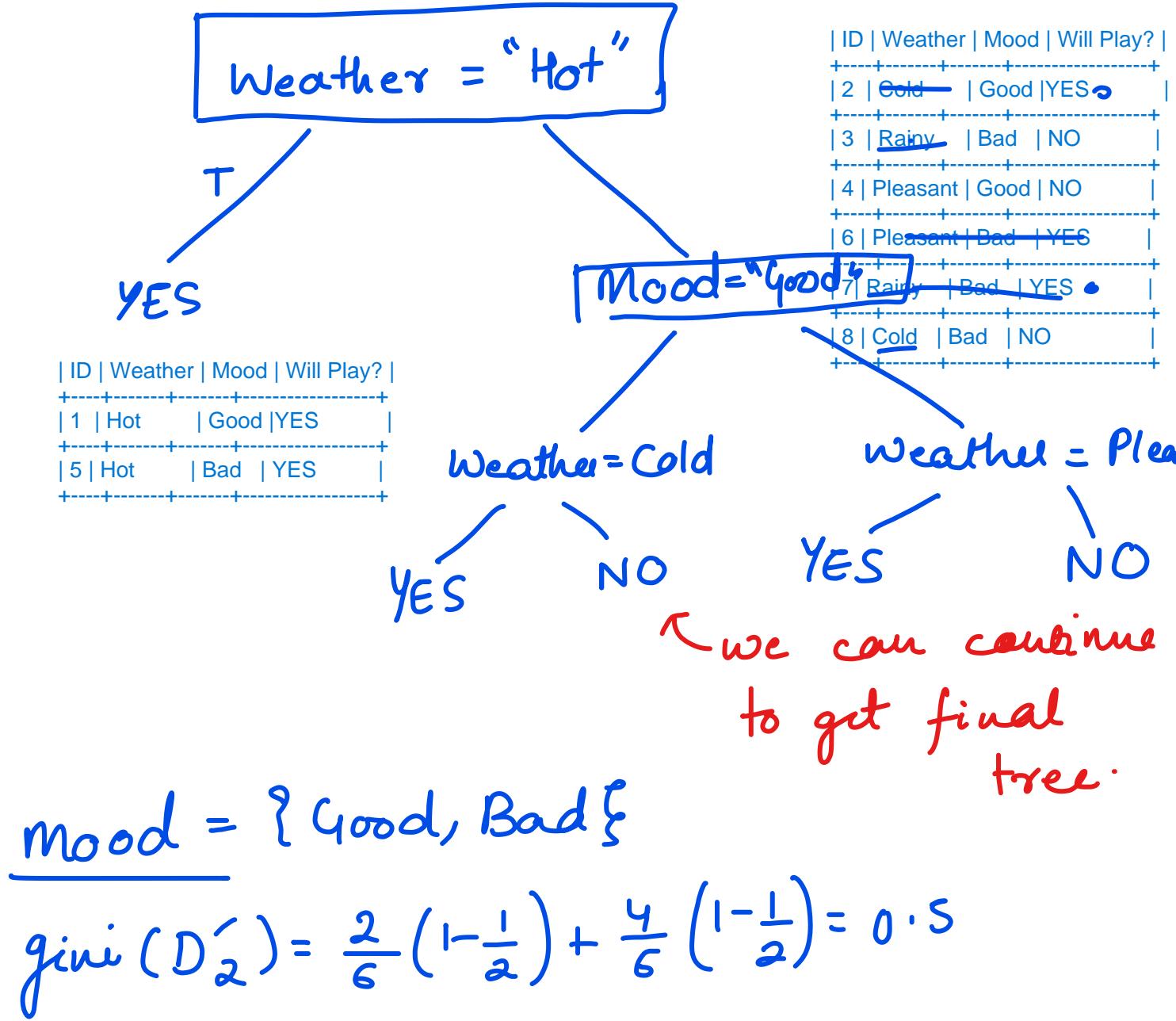
$$S_6 \quad \{ \text{Hot or Rainy} \} \quad g_6(D_1) = 0.4375 \quad S_7 \quad \{ \text{Hot or Pleasant} \} = 0.4375$$

Mood $\{ \text{good, bad} \}$

$$\text{gini}(D_2) = \frac{3}{8} \left(1 - \left(\frac{2}{3}\right)^2 - \left(\frac{1}{3}\right)^2 \right) + \frac{5}{8} \left(1 - \left(\frac{2}{5}\right)^2 - \left(\frac{3}{5}\right)^2 \right)$$

$$= 0.467$$

Thus, we choose Weather = {Cold, Rainy, Pleasant} as the required decision node.



new dataset / subset

Weather = {Cold, Rainy, Pleasant}

$S_1 = \{ \text{Cold or Rainy} \}$

$$\text{gini}_1(D'_1) = \frac{4}{6} \left(1 - \frac{2}{4} - \frac{2}{4} \right) + \frac{2}{6} \left(1 - \frac{1}{2} - \frac{1}{2} \right) = 0.5$$

$S_2 = \{ \text{cold or pleasant} \}$

$$gini_2(D'_1) = \frac{4}{6} \left(1 - \frac{1}{2}\right) + \frac{2}{6} \left(1 - \frac{1}{2}\right)$$

$s_3 = \{ \text{Rainy or Pleasant} \}^{-0.5}$

$$\text{gini}_3(D_1') = 0.5$$

- * For a particular attribute A with 'v' values, we will have to calculate gini Index for $\frac{2^v - 2}{2} = 2^{v-1} - 1$ subsets.
- * For continuous and numerical values, mean of 2 consecutive values (when arranged in a sorted order) is calculated & gini Index is calculated on the basis of this mean.

References

References

- Dietterich, T. G., (1998). Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms. *Neural Computation*, 10 (7) 1895-1924
- Densar, J., (2006). Demsar, Statistical Comparisons of Classifiers over Multiple Data Sets. *The Journal of Machine Learning Research*, pages 1-30.
- Machine Learning, Jesse Davis, jdavis@cs.washington.edu
<https://courses.cs.washington.edu/courses/cse573/08au/slides/>
- Daniel S. Weld <https://www.cs.washington.edu/people/faculty/weld>
- CS489/698: Intro to ML, Lecture 19: Decision Tree, Instructor: Sun Sun (17/18/18)
- “Introductory Applied Machine Learning” by Victor Lavrenko and Nigel Goddard
University of Edinburgh.
- <https://scikit-learn.org/stable/modules/tree.html>

ଖୁବଖୁବ

Thai

ଧନ୍ୟବାଦ:

Sanskrit

Ευχαριστώ

Greek

Спасибо

Russian

شُكْرًا

Arabic

多謝

Traditional

Chinese

多谢

Simplified

Chinese

Japanese

תודה רבה

Italian

Grazie

Hebrew

ಧನ್ಯವಾದಗಳು

Kannada

Thank You

English

Gracias

Spanish

Obrigado

Portuguese

Merci

French

Danke

German

ଧନ୍ୟବାଦ

Hindi

நன்றி

Tamil

Tamil

ありがとうございました

감사합니다

Korean

<https://sites.google.com/site/animeshchaturvedi07>