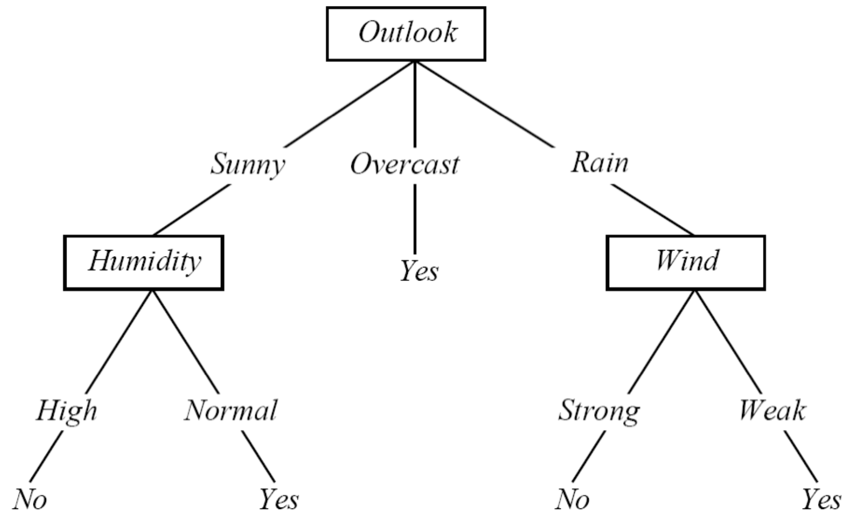
Play Tennis Example

Outlook	Temperature	Humidity	Windy	PlayTennis
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



Constructing DTs

- How do we choose the attributes and the order in which they appear in a DT?
  - Recursive partitioning of the original data table
  - Heuristic - each generated partition has to be “less random” (entropy reduction) than previously generated partitions

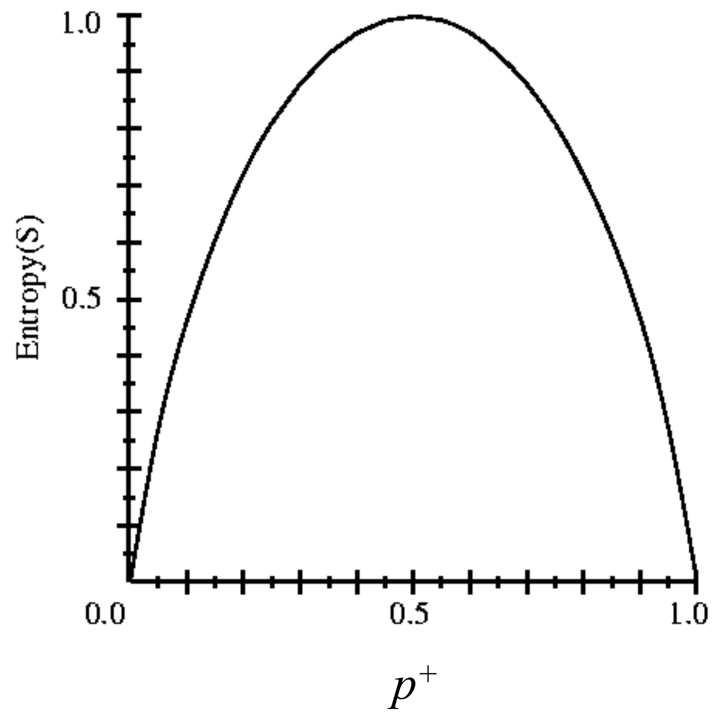
Entropy

S is a sample of training examples

p^+ is the proportion of positive examples in S

p^- is the proportion of negative examples in S

Entropy measures the impurity (randomness) of S



$$Entropy(S) \equiv -\, p^+ \log_2 p^+ - \, p^- \log_2 p^-$$

S

Outlook	Temperature	Humidity	Windy	PlayTennis
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

Entropy(*S*) = *Entropy*([9+,5-]) = *.94*

Recursive Partitioning – ID3

ID3($\mathbf{D}$,$\mathbf{X}$) =
- Let T be a new tree
- If all instances in $\mathbf{D}$ have same class c
  - Label(T) = c; Return T
- If $\mathbf{X} = \varnothing$ or no attribute has positive information gain
  - Label(T) = most common class in $\mathbf{D}$; return T
- $X \leftarrow$ attribute with highest information gain
- Label(T) = X
- For each value x of X
  - $\mathbf{D}_x \leftarrow$ instances in $\mathbf{D}$ with $X = x$
  - If $\mathbf{D}_x$ is empty
    - Let T_x be a new tree
    - Label(T_x) = most common class in $\mathbf{D}$
  - Else
    - $T_x = \text{ID3}(\mathbf{D}_x, \mathbf{X} - \{X\})$
  - Add a branch from T to T_x labeled by x
- Return T

Recursive Partitioning

Partition(*Examples*, *TargetAttribute*, *Attributes*)

Examples are the training examples. TargetAttribute is a binary (+/-) *categorical dependent variable and Attributes is the list of independent variables which are available for testing at this point. This function returns a decision tree.*

- Create a *Root* node for the tree.
- If all *Examples* are positive then return *Root* as a leaf node with label = +.
- Else if all *Examples* are negative then return *Root* as a leaf node with label = -.
- Else if *Attributes* is empty then return *Root* as a leaf node with label = most common value of TargetAttribute in Examples.
- Otherwise
    - A := the attribute from *Attributes* that reduces entropy the most on the *Examples*.
    - *Root* := A
    - For each $v \in values(A)$
        - Add a new branch below the *Root* node with value $A = v$
        - Let $Examples_v$ be the subset of *Examples* where $A = v$
        - If $Examples_v$ is empty then add new leaf node to branch with label = most common value of *TargetAttribute* in *Examples*.
        - Else add new subtree to branch Partition($Examples_v$, *TargetAttribute*, $Attributes - \{A\}$)
- Return Root

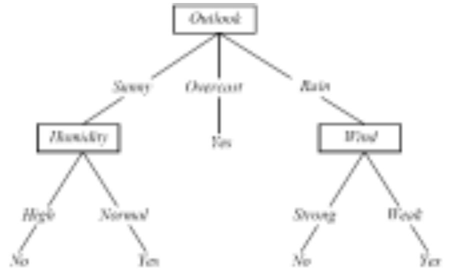
Based on material from the book: "Machine Learning", Tom M. Mitchell. McGraw-Hill, 1997.

Decision Tree Learning

Outlook	Temperature	Humidity	Windy	PlayTennis
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

Facts or Observations

Induction

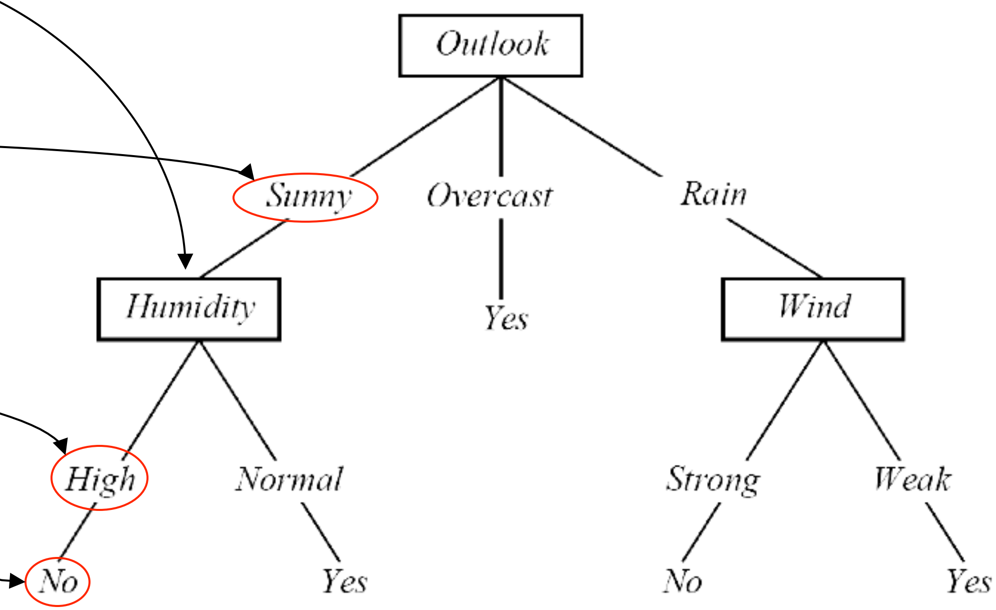


Theory

Interpreting a DT

DT ≡ Decision Tree

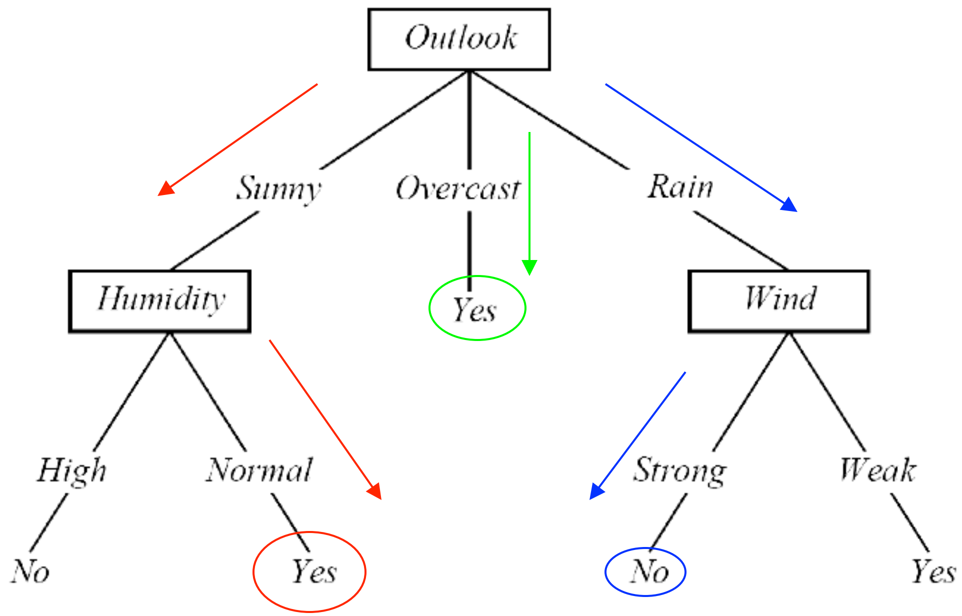
Outlook	Temperature	Humidity	Windy	PlayTennis
Sunny	Hot	High	False	No
Sunny	Hot	High	True	No
Overcast	Hot	High	False	Yes
Rainy	Mild	High	False	Yes
Rainy	Cool	Normal	False	Yes
Rainy	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Sunny	Mild	High	False	No
Sunny	Cool	Normal	False	Yes
Rainy	Mild	Normal	False	Yes
Sunny	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Rainy	Mild	High	True	No



⇒ A DT uses the <u>attributes</u> of an observation table as nodes and the <u>attribute values</u> as links.
⇒ <u>All</u> attribute values of a particular attribute need to be represented as links.
⇒ The target attribute is special - its values show up as <u>leaf nodes</u> in the DT.

Interpreting a DT

Each path from the root of the DT to a leaf can be interpreted as a decision rule.



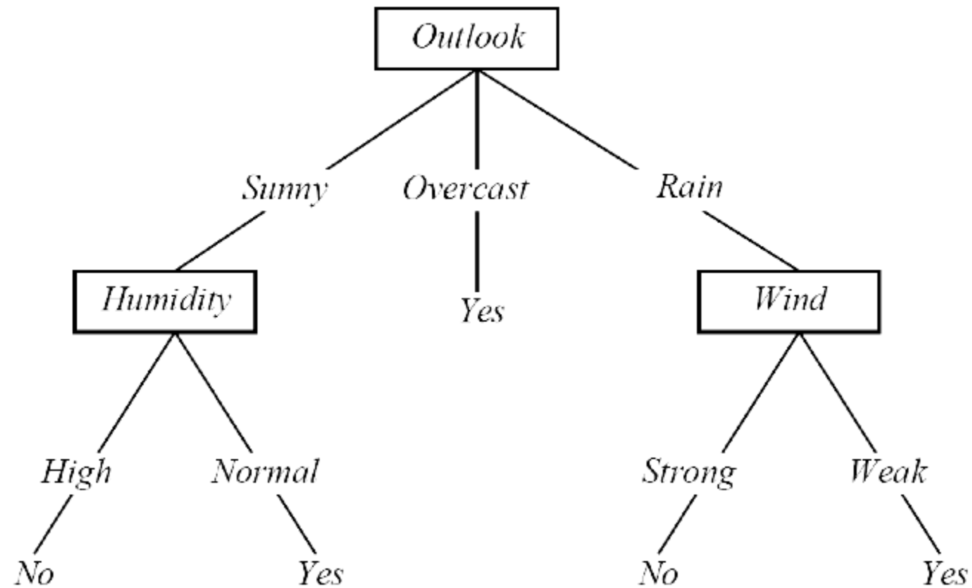
IF *Outlook* = *Sunny* AND *Humidity* = *Normal* THEN *Playtennis* = *Yes*

IF *Outlook* = *Overcast* THEN *Playtennis* =*Yes*

IF *Outlook* = *Rain* AND *Wind* = *Strong* THEN *Playtennis* = *No*

DT: Explanation & Prediction

Outlook	Temperature	Humidity	Windy	PlayTennis
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



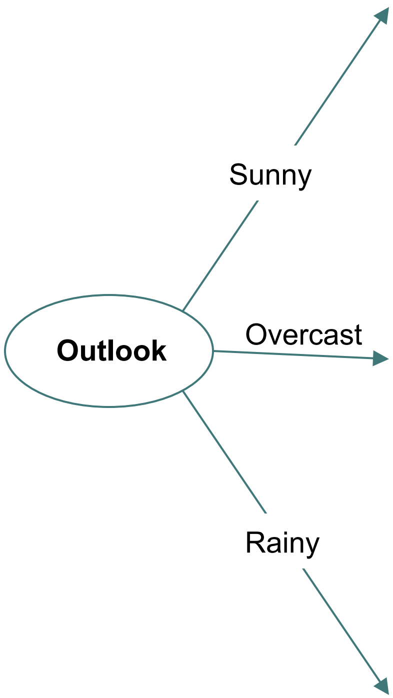
Explanation: the DT summarizes (explains) all the observations in the table perfectly $\Rightarrow$ 100% Accuracy

Prediction: once we have a DT (or model) we can use it to make predictions on observations that are not in the original training table, consider:

Outlook = Sunny, Temperature = Mild, Humidity = Normal, Windy = False, Playtennis = ?

$$\text{AvgEntropy}(S,A) = \sum_{v \in Values(A)} \frac{|S_v|}{|S|} E(S_v) \quad \text{(weighted average)}$$

Partitioning the Data Set



Outlook	Temperature	Humidity	Windy	PlayTennis
Sunny	Hot	High	False	No
Sunny	Hot	High	True	No
Sunny	Mild	High	False	No
Sunny	Cool	Normal	False	Yes
Sunny	Mild	Normal	True	Yes

E = .97

Outlook	Temperature	Humidity	Windy	PlayTennis
Overcast	Hot	High	False	Yes
Overcast	Cool	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes

E = 0

Outlook	Temperature	Humidity	Windy	PlayTennis
Rainy	Mild	High	False	Yes
Rainy	Cool	Normal	False	Yes
Rainy	Cool	Normal	True	No
Rainy	Mild	Normal	False	Yes
Rainy	Mild	High	True	No

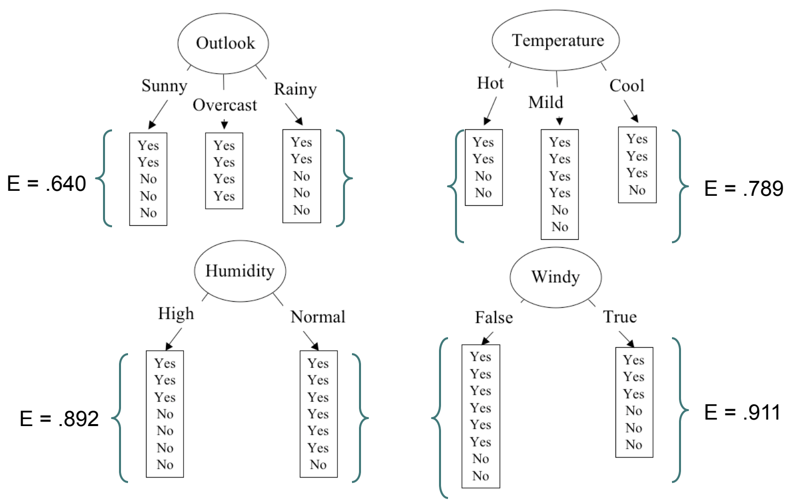
E = .97

Average Entropy = .64

(weighted .69)

Partitioning in Action

Outlook	Temperature	Humidity	Windy	PlayTennis
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



Recursive Partitioning

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

Outlook

Sunny	Hot	High	False	No
Sunny	Hot	High	True	No
Sunny	Mild	High	False	No
Sunny	Cool	Normal	False	Yes
Sunny	Mild	Normal	True	Yes

Overcast	Hot	High	False	Yes
Overcast	Cool	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes

Rainy	Mild	High	False	Yes
Rainy	Cool	Normal	False	Yes
Rainy	Cool	Normal	True	No
Rainy	Mild	Normal	False	Yes
Rainy	Mild	High	True	No

Recursive Partitioning

Outlook

Sunny	Hot	High	False	No
Sunny	Hot	High	True	No
Sunny	Mild	High	False	No
Sunny	Cool	Normal	False	Yes
Sunny	Mild	Normal	True	Yes

Overcast	Hot	High	False	Yes
Overcast	Cool	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes

Rainy	Mild	High	False	Yes
Rainy	Cool	Normal	False	Yes
Rainy	Cool	Normal	True	No
Rainy	Mild	Normal	False	Yes
Rainy	Mild	High	True	No

Recursive Partitioning

Outlook

Sunny	Hot	High	False	No
Sunny	Hot	High	True	No
Sunny	Mild	High	False	No
Sunny	Cool	Normal	False	Yes
Sunny	Mild	Normal	True	Yes

Overcast	Hot	High	False	Yes
Overcast	Cool	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes

Rainy	Mild	High	False	Yes
Rainy	Cool	Normal	False	Yes
Rainy	Cool	Normal	True	No
Rainy	Mild	Normal	False	Yes
Rainy	Mild	High	True	No

Humidity

Sunny	Hot	High	False	No
Sunny	Hot	High	True	No
Sunny	Mild	High	False	No

Sunny	Cool	Normal	False	Yes
Sunny	Mild	Normal	True	Yes

Recursive Partitioning

Outlook

Sunny	Hot	High	False	No
Sunny	Hot	High	True	No
Sunny	Mild	High	False	No
Sunny	Cool	Normal	False	Yes
Sunny	Mild	Normal	True	Yes

Overcast	Hot	High	False	Yes
Overcast	Cool	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes

Rainy	Mild	High	False	Yes
Rainy	Cool	Normal	False	Yes
Rainy	Cool	Normal	True	No
Rainy	Mild	Normal	False	Yes
Rainy	Mild	High	True	No

Humidity

Sunny	Hot	High	False	No
Sunny	Hot	High	True	No
Sunny	Mild	High	False	No

Sunny	Cool	Normal	False	Yes
Sunny	Mild	Normal	True	Yes

Windy

Rainy	Mild	High	False	Yes
Rainy	Cool	Normal	False	Yes
Rainy	Mild	Normal	False	Yes

Rainy	Cool	Normal	True	No
Rainy	Mild	High	True	No

Continuous-Valued Attributes

Consider:

Temperature:	40	48	60	72	80	90
PlayTennis:	No	No	Yes	Yes	Yes	No

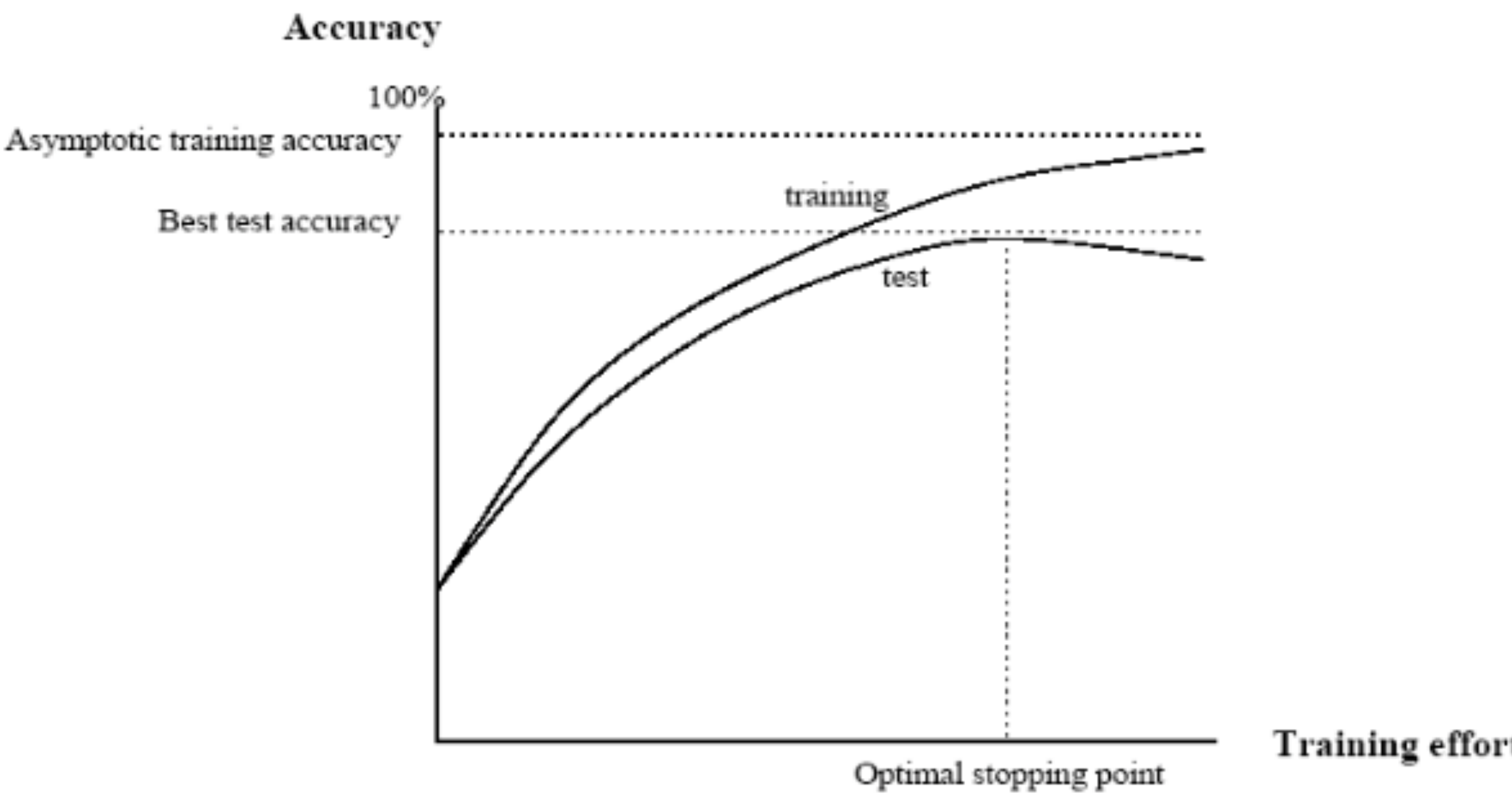
(48+60)/2
= 54

(80+90)/2
= 85

Highest Gain: Temperature > 54

- Sort instances according to the attribute values
- Find "Splits" where the classes change
- Select the split that gives you the highest gain

Overfitting



Patterns in Data

- True pattern in domain

  - present in large amounts of data

  - generalizes to unseen instances

- Spurious pattern in training set

  - “noise in the data”

  - present in small amounts of data

  - does not generalize

ID3 Learning Process

- Beginning

  - lots of data

  - discovers true patterns in data

- Later

  - small amount of data

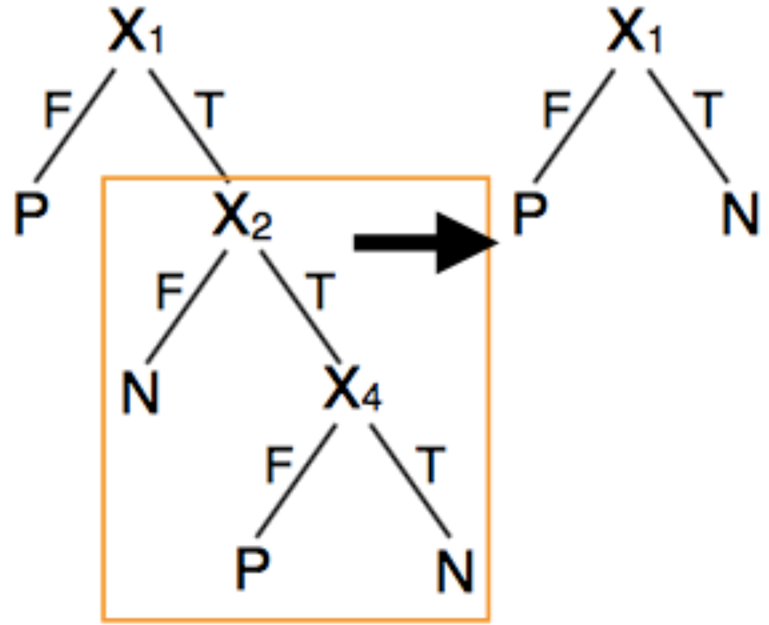
  - likely to learn spurious patterns

Pruning

- Remove a subtree
- Replace with a leaf
- Class of leaf is most common class of data in subtree

Pruning Example

X_1	X_2	X_3	X_4	C
F	F	F	F	P
F	F	T	T	P
F	T	F	T	P
T	T	T	F	P
T	F	F	F	N
T	T	T	T	N
T	T	T	F	N



When to Prune

- Pre-pruning

  - prune the tree before you grow it

  - adds a termination condition to ID3

  - saves computation time

- Post-pruning

  - grow complete tree and prune it afterward

  - takes into account more information

Pruning Methods

- Minimum number of instances in leaves
- (quasi) Statistical approaches

Minimum Instances

- Each leaf has to have a minimum number of instances
- Don't split if the required number is too low
- Instead return majority label

Deviation

- Before splitting, ask:

  - Is the pattern found in the data after splitting statistically significant?

- Pre-pruning method

  - only need distribution of classes at node

Deviation

- Prune if deviation is small

Proposed Split

- Given instances **D**
- Propose split on X_i
- Notation
  - N_c = number of instances with class c
  - $\mathbf{D}_x$ = data set with value x for attribute X_i
  - N_x = number of instances in $\mathbf{D}_x$
  - N_{xc} = number of instances in $\mathbf{D}_x$ with class c

Absence of Pattern

- Null hypothesis: X_i is irrelevant
- In **D**, proportion with class c: N_c/N
- If null hypothesis is true, we expect on average the number of instances in $\mathbf{D}_x$ with class c to be:

$$\hat{N}_{xc} = \frac{N_c}{N}\left|\mathbf{D}_x\right|$$

Deviation

- We don't expect to see *exactly* that many, even if null hypothesis is true
- We expect some deviation due to random chance
- Measure the deviation from total absence of pattern:

$$\mathrm{Dev} = \sum_{x} \sum_{c} \frac{(N_{xc} - \hat{N}_{xc})^2}{\hat{N}_{xc}}$$

→ Don't split if Dev is small