08.10.2024

Statistical Methods in AI (CS7.403)

Lecture-18: Decision Tree Learning-2

Ravi Kiran (ravi.kiran@iiit.ac.in)

https://ravika.github.io

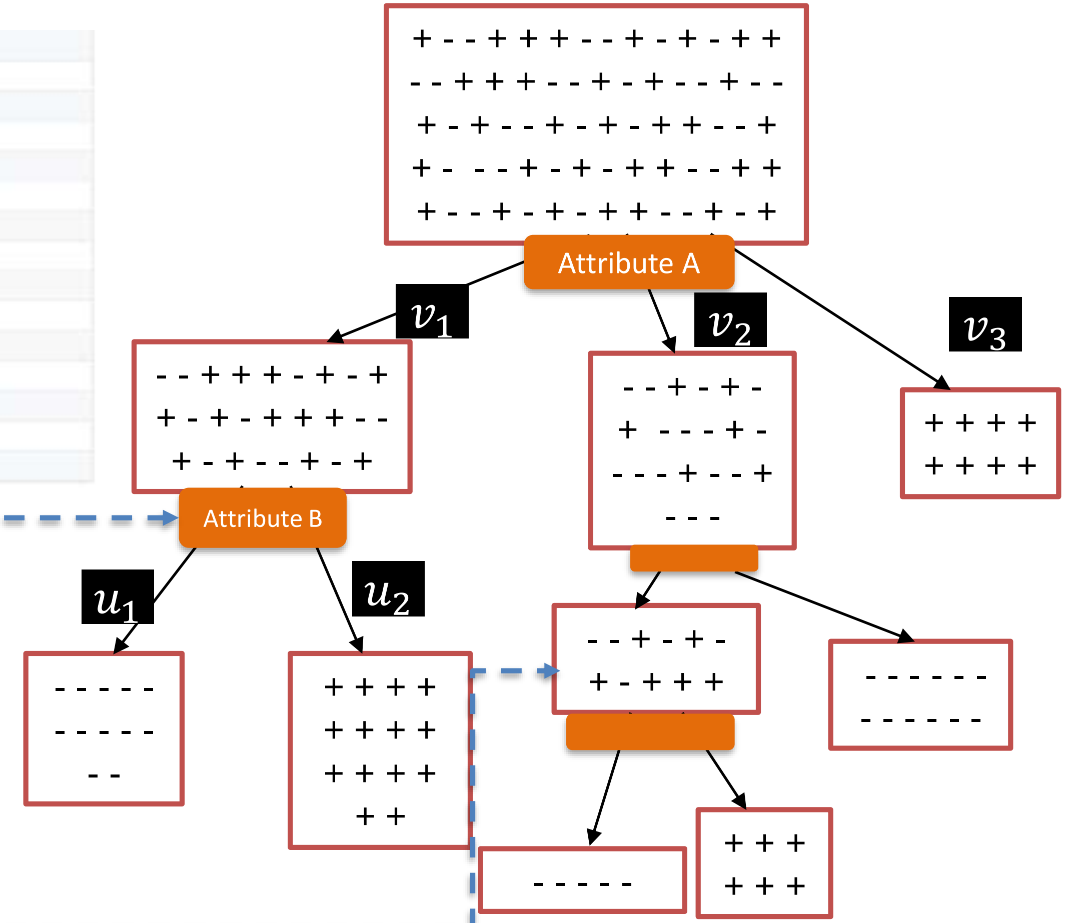
@vikataravi

Center for Visual Information Technology (CVIT)

IIIT Hyderabad

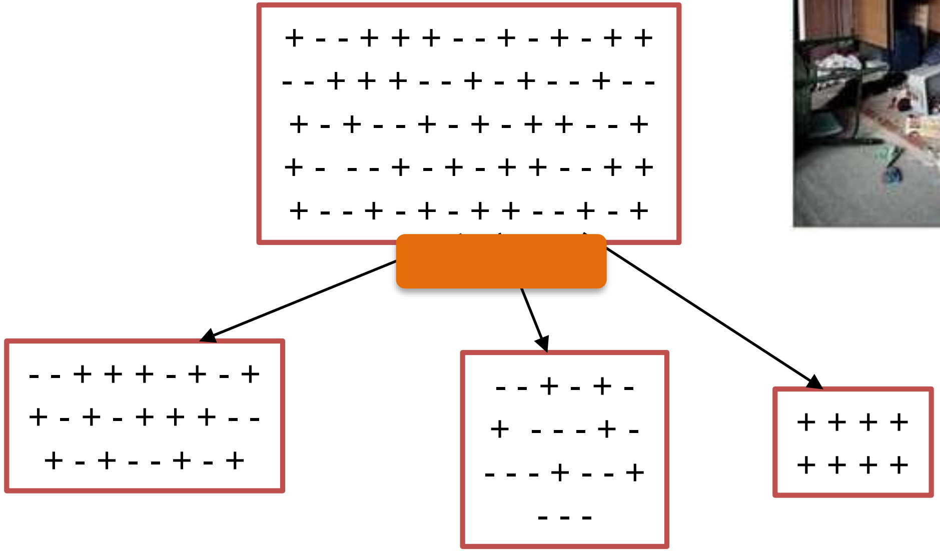
Decision Tree

1	outlook	temp	humidity	windy	play
2	sunny	hot	high	false	no
3	sunny	hot	high	true	no
4	overcast	hot	high	false	yes
5	rainy	mild	high	false	yes
6	rainy	cool	normal	false	yes
7	rainy	cool	normal	true	no
8	overcast	cool	normal	true	yes
9	sunny	mild	high	false	no
10	sunny	cool	normal	false	yes
11	rainy	mild	normal	false	yes
12	sunny	mild	normal	true	yes
13	overcast	mild	high	true	yes
14	overcast	hot	normal	false	yes
15	rainy	mild	high	true	no



Ideal attribute aka pure node

How much ‘impurity’ does this attribute decrease ?



Step-1: Compute impurity score of training label distribution

Day	Temperature	Outlook	Humidity	Windy	Play Golf?
07-05	hot	sunny	high	false	no
07-06	hot	sunny	high	true	no
07-07	hot	overcast	high	false	yes
07-09	cool	rain	normal	false	yes
07-10	cool	overcast	normal	true	yes
07-12	mild	sunny	high	false	no
07-14	cool	sunny	normal	false	yes
07-15	mild	rain	normal	false	yes
07-20	mild	sunny	normal	true	yes
07-21	mild	overcast	high	true	yes
07-22	hot	overcast	normal	false	yes
07-23	mild	rain	high	true	no
07-26	cool	rain	normal	true	no
07-30	mild	rain	high	false	yes

Entropy: $i(V) = -(q \log q + (1-q)\log(1-q))$

$$E(S) = -(\frac{9}{14}log(\frac{9}{14}) + \frac{5}{14}log(\frac{5}{14})) = 0.94$$

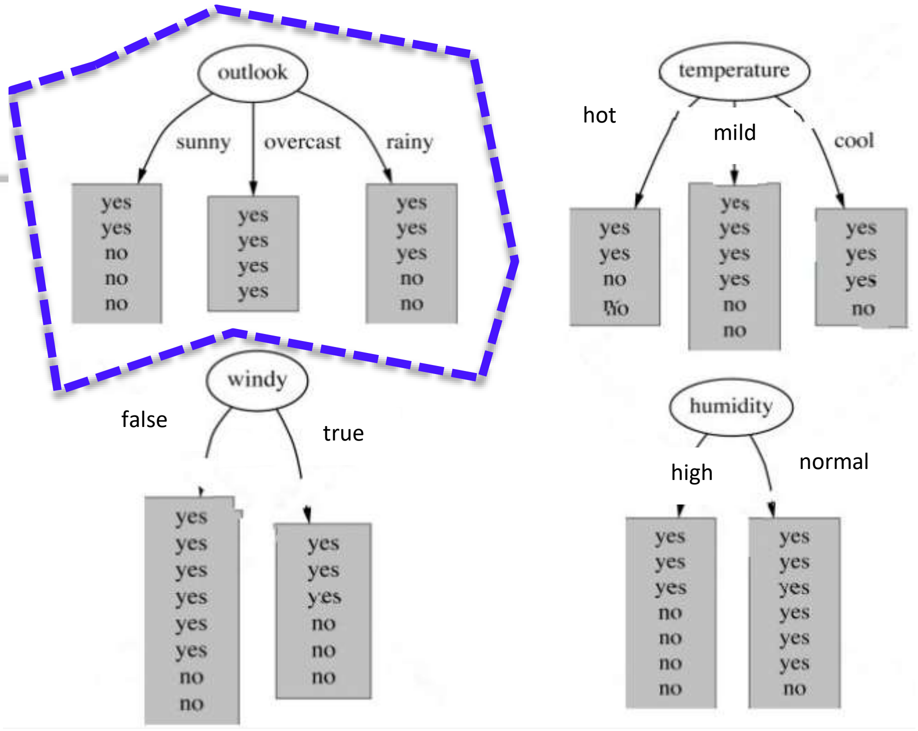
Step-2: Compute impurity score for each unique value of candidate attributes

Example: Attribute Outlook

Entropy: $i(V) = -(q \log q + (1-q)\log(1-q))$

- **Outlook = rainy** 3 examples yes, 2 examples no

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



Step-2: Compute impurity score for each unique value of candidate attributes

Example: Attribute Outlook

Entropy: $i(V) = -(q \log q + (1-q)\log(1-q))$

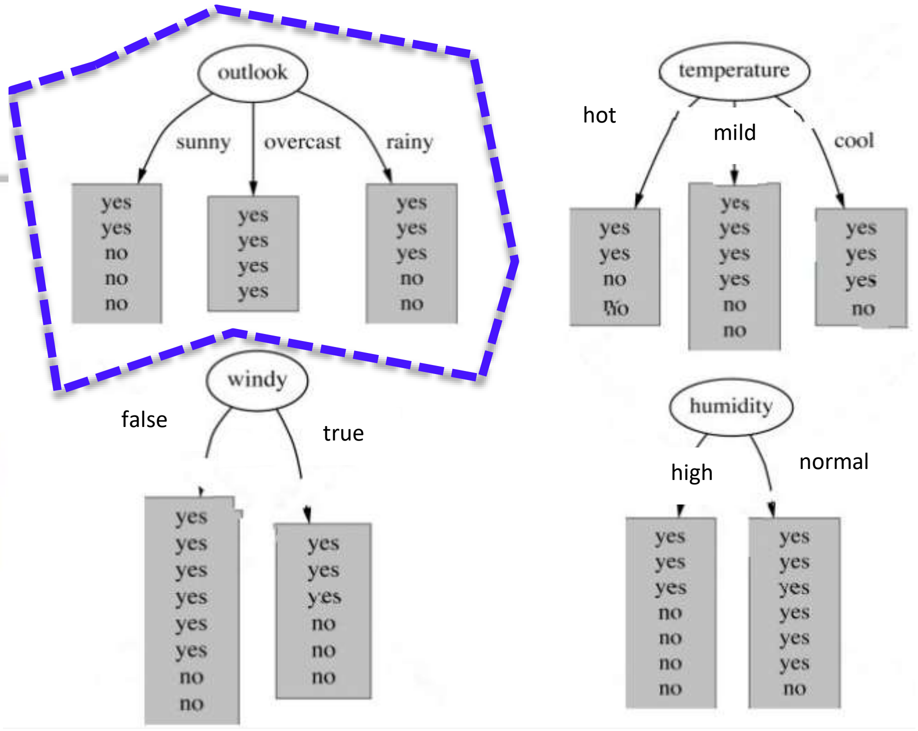
- **Outlook = rainy** 3 examples yes, 2 examples no

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

- Outlook = overcast: 4 examples yes, 0 examples no

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

Note: this is normally undefined. Here: = 0



Step-2: Compute impurity score for each unique value of candidate attributes

Example: Attribute Outlook

Entropy: $i(V) = -(q \log q + (1-q)\log(1-q))$

- **Outlook = rainy** 3 examples yes, 2 examples no

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

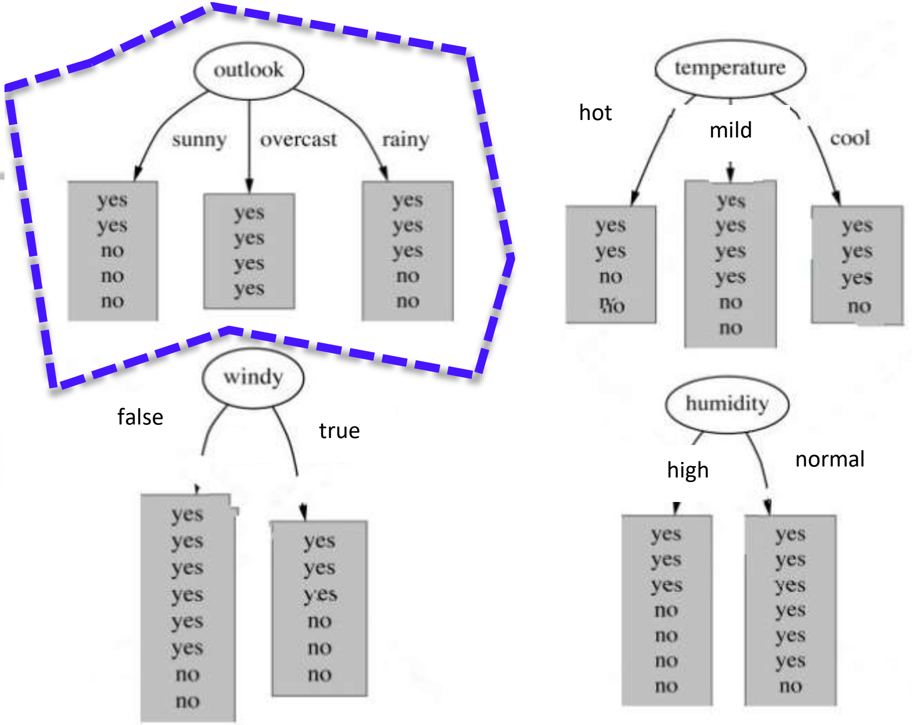
- Outlook = overcast: 4 examples yes, 0 examples no

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

Note: this is normally undefined. Here: = 0

- **Outlook = sunny** 2 examples yes, 3 examples no

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



Step-3: Compute impurity score for candidate attribute

- **Outlook = rainy** 3 examples yes, 2 examples no

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

- Outlook = overcast: 4 examples yes, 0 examples no

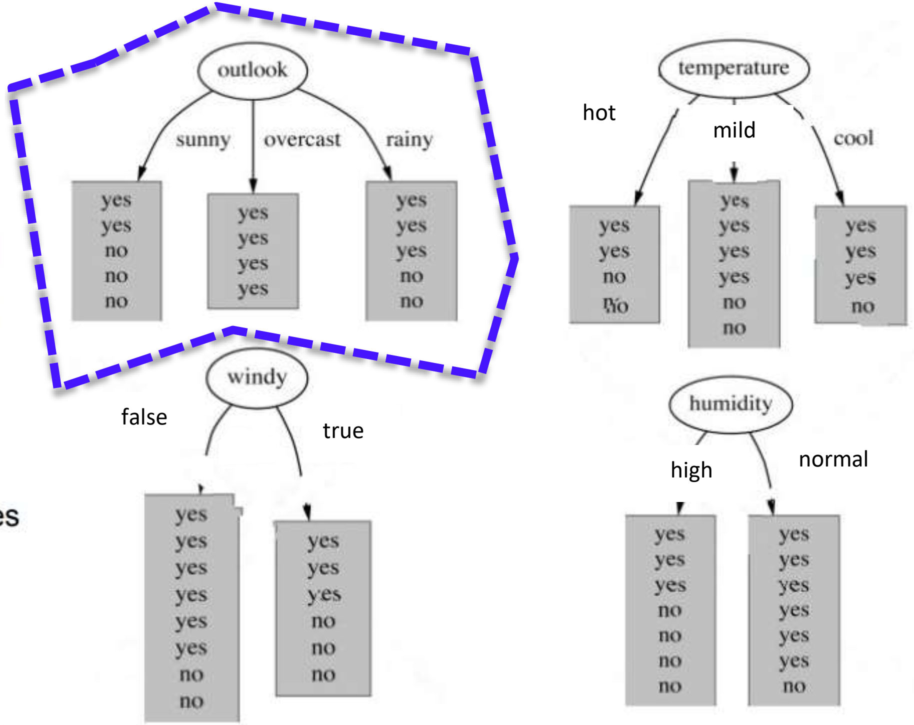
$$E(\text{Outlook}=\text{overcast}) = -1\log(1) - 0\log(0) = 0$$

Note: this is normally undefined. Here: = 0

- **Outlook = sunny** 2 examples yes, 3 examples no

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

- Entropy only computes the quality of a single (sub-)set of examples
  - corresponds to a single value
- How can we compute the quality of the entire split?
  - corresponds to an entire attribute



Step-3: Compute impurity score for candidate attribute

- **Outlook = rainy** 3 examples yes, 2 examples no

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

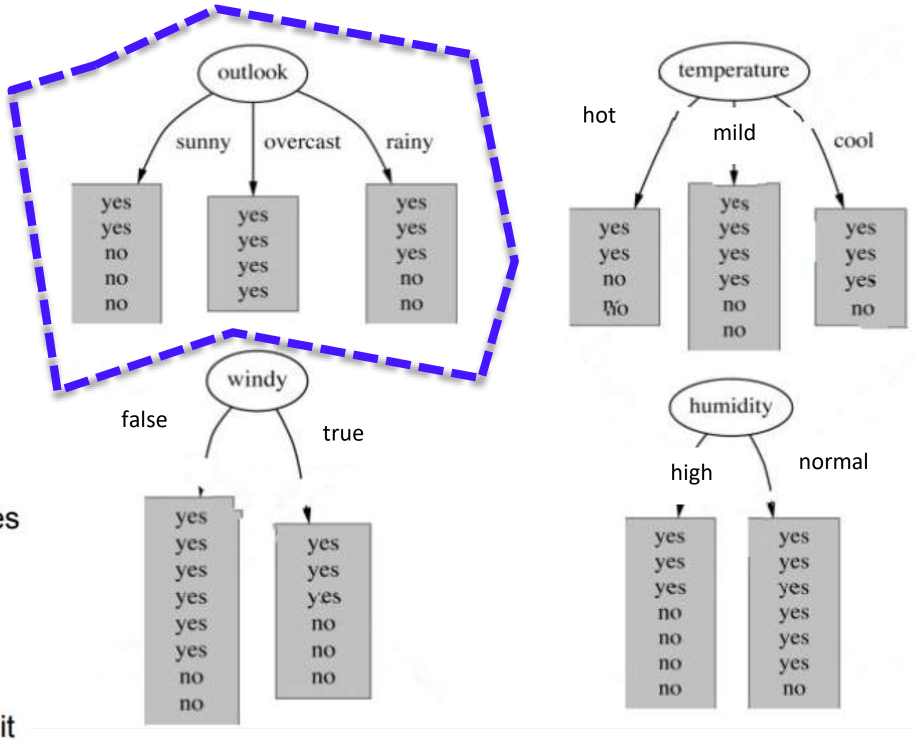
- Outlook = overcast: 4 examples yes, 0 examples no

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

Note: this is normally undefined. Here: = 0

- **Outlook = sunny** 2 examples yes, 3 examples no

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



- Entropy only computes the quality of a single (sub-)set of examples
  - corresponds to a single value
- How can we compute the quality of the entire split?
  - corresponds to an entire attribute

Solution:

- Compute the weighted average over all sets resulting from the split
  - weighted by their size

$$I(S, A) = \sum_i \frac{|S_i|}{|S|} \cdot E(S_i)$$

Step-3: Compute impurity score for candidate attribute

- **Outlook = rainy** 3 examples yes, 2 examples no

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

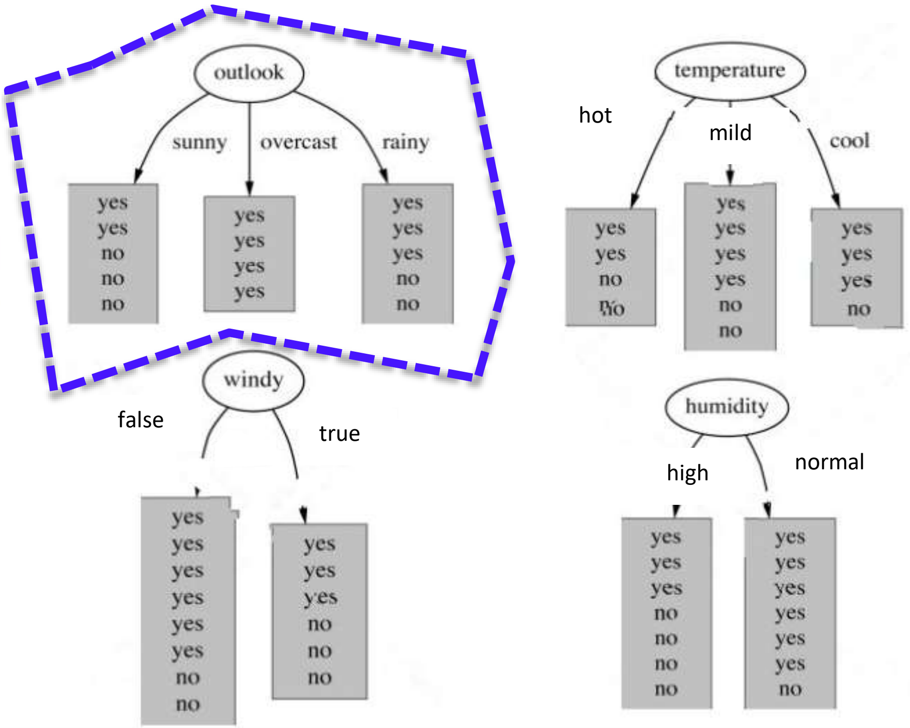
- Outlook = overcast: 4 examples yes, 0 examples no

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

Note: this is normally undefined. Here: = 0

- **Outlook = sunny** 2 examples yes, 3 examples no

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



$$I(S,A) = \sum_i \frac{|S_i|}{|S|} \cdot E(S_i)$$

- Average entropy for attribute *Outlook*:

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

Step-4: Compute Information Gain (reduction in impurity score) provided by candidate attribute

$$I(S,A)=\sum_i \frac{|S_i|}{|S|}\cdot E(S_i)$$

- Average entropy for attribute *Outlook*:

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

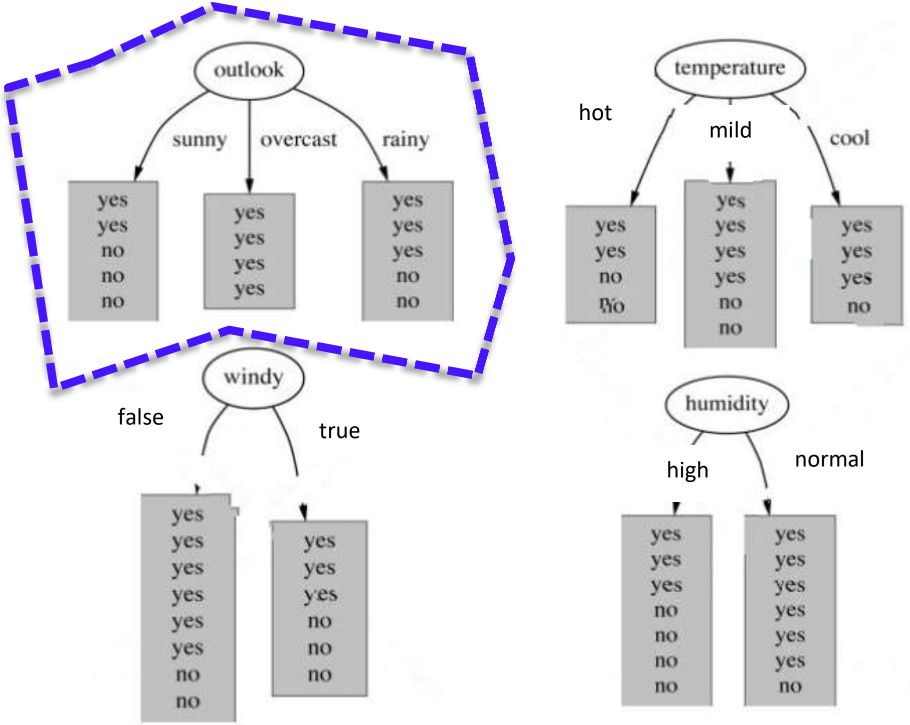
Entropy of root

$$E(S)=-\left(\frac{9}{14}log\left(\frac{9}{14}\right)+\frac{5}{14}log\left(\frac{5}{14}\right)\right)=0.94$$

Information Gain for Attribute A

$$Gain(S,A)=E(S)-I(S,A)=E(S)-\sum_i \frac{|S_i|}{|S|}\cdot E(S_i)$$

$$Gain(S,Outlook)=0.246$$



Step-5: Compare Information Gain provided by all candidates

Information Gain for Attribute A

$$Gain(S,A)=E(S)-I(S,A)=E(S)-\sum_i \frac{|S_i|}{|S|}\cdot E(S_i)$$

Gain (S, Humidity)

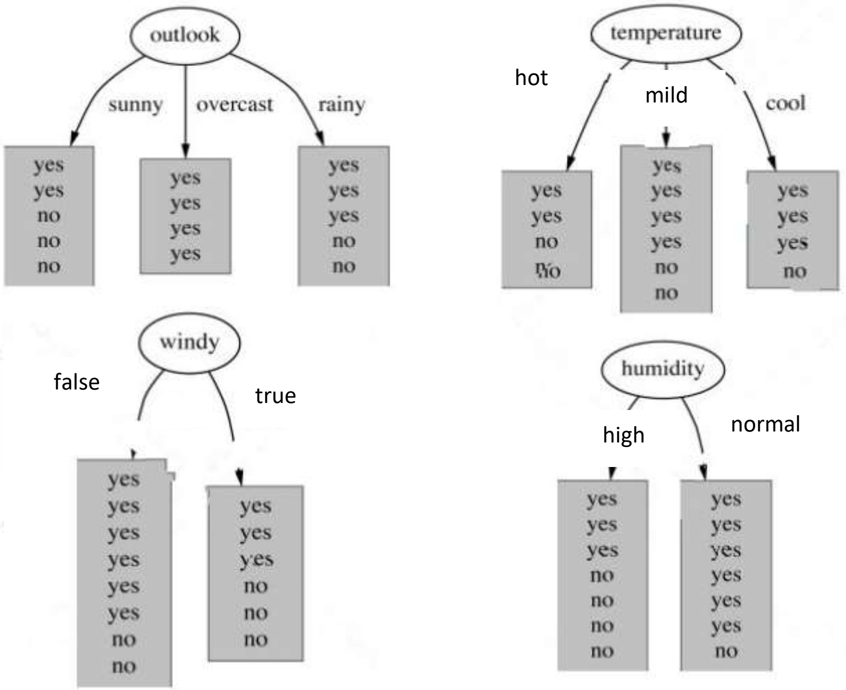
$= .940 - (7/14).985 - (7/14).592$
$= .151$

Gain (S, Wind)

$= .940 - (8/14).811 - (6/14)1.0$
$= .048$

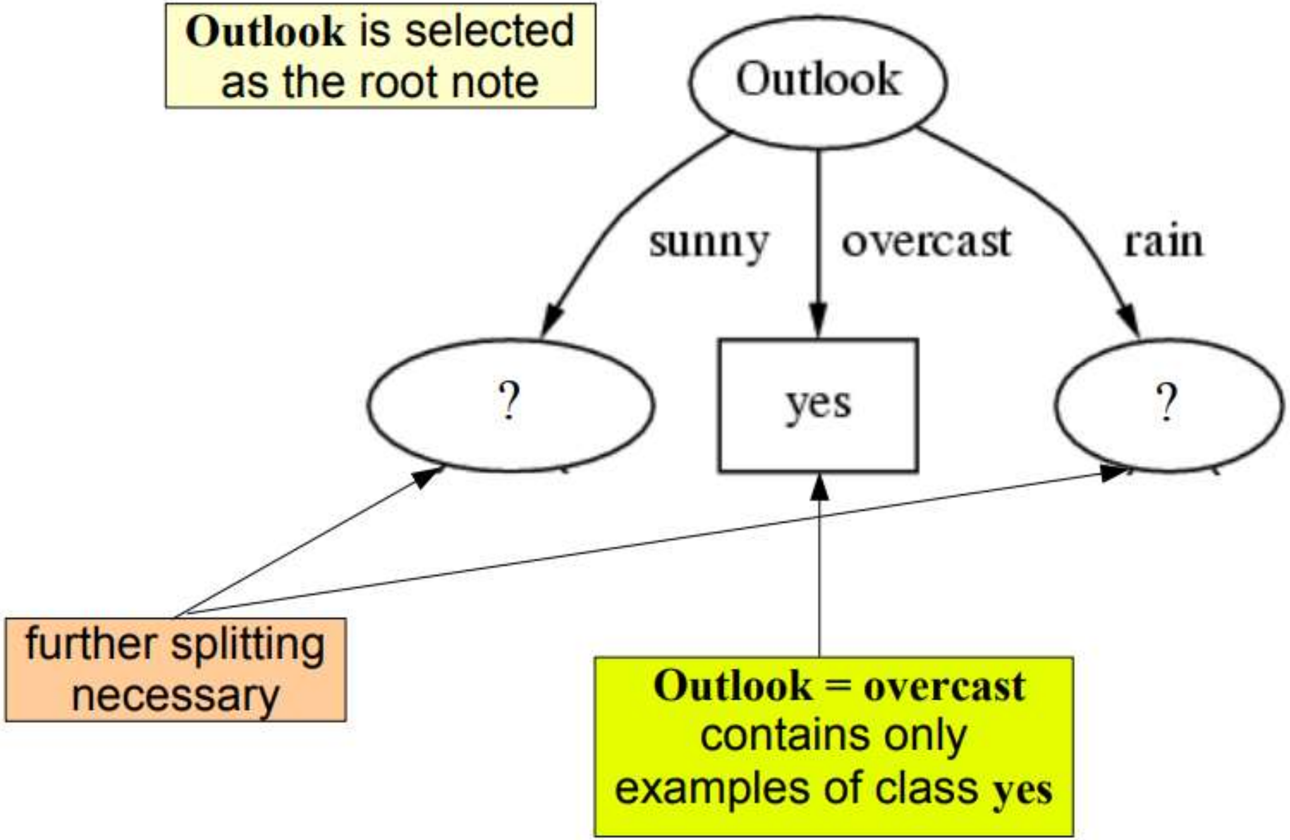
$Gain(S, Outlook) = \boxed{0.246}$

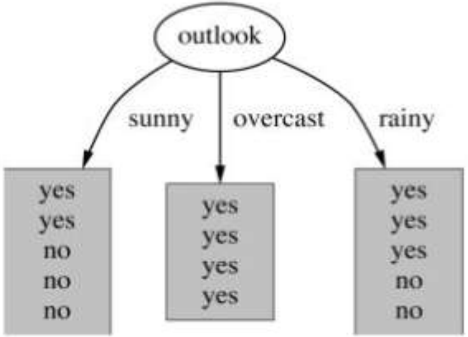
$Gain(S, Temperature) = 0.029$



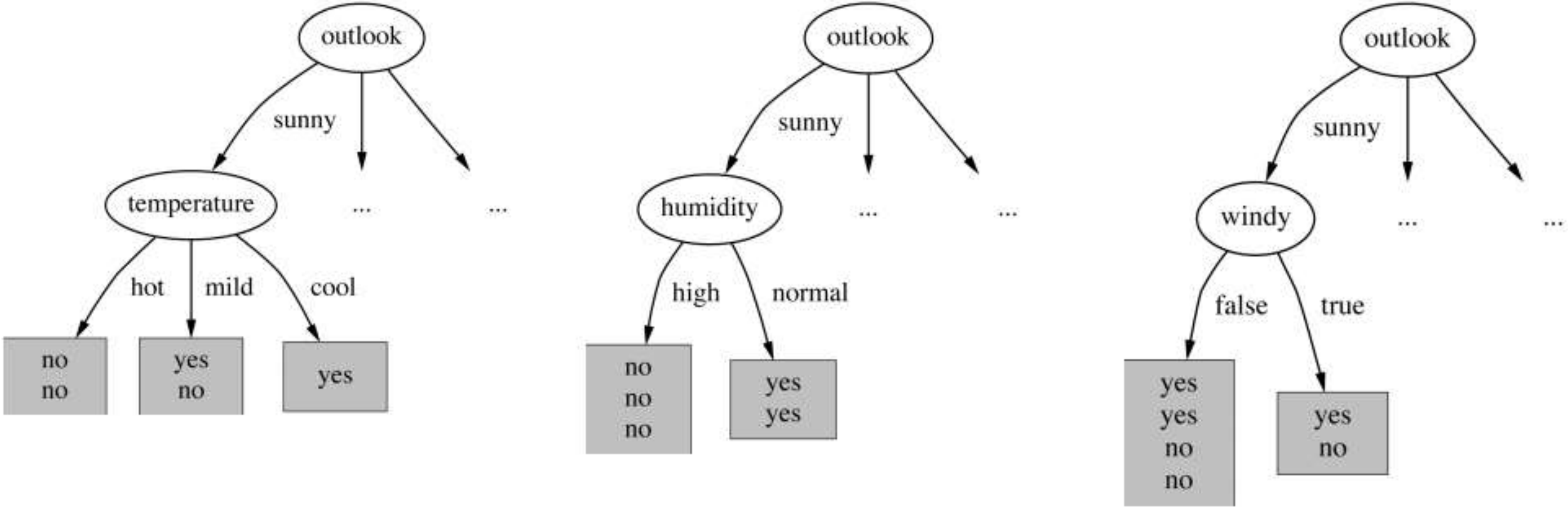
Select the attribute which provides largest 'impurity reduction'/Information Gain

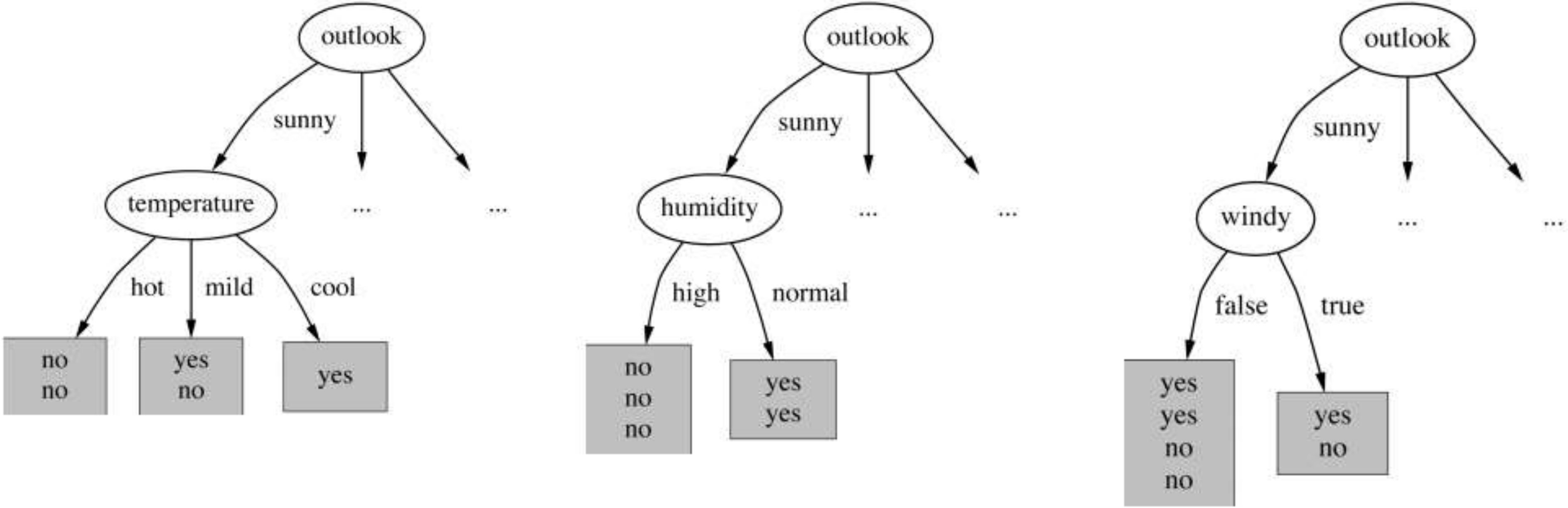
Step-6: Assign root node





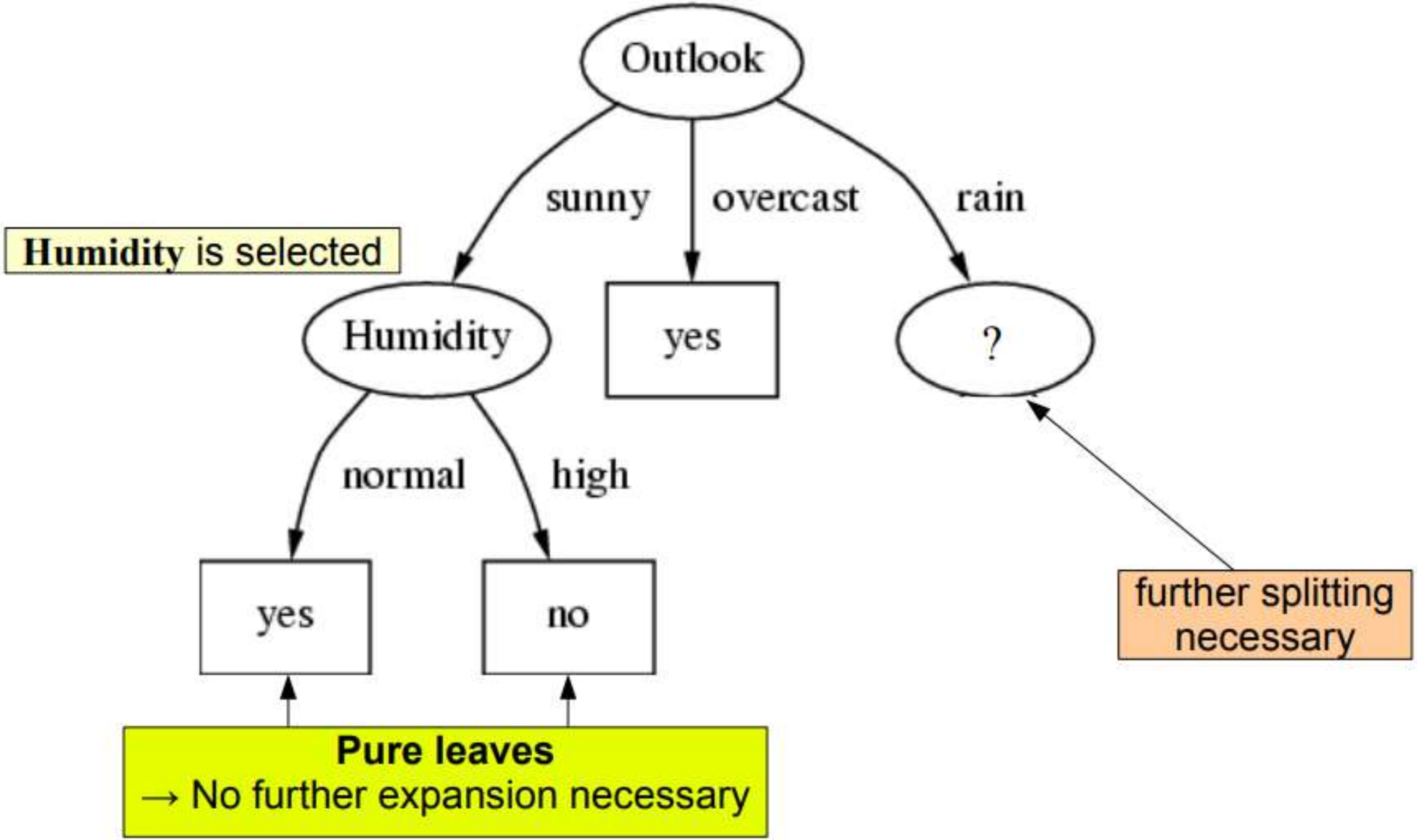
Recurse and repeat Steps 1-6





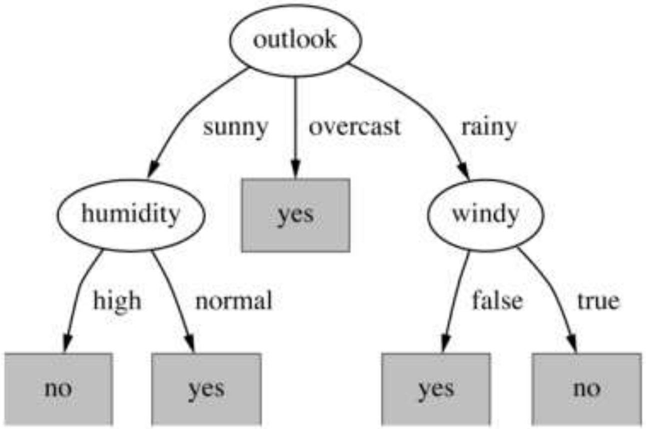
$$
\left.\begin{aligned}
\text{Gain}(\textit{Temperature}) &= 0.571 \text{ bits}\\
\text{Gain}(\textit{Humidity}) &= 0.971 \text{ bits}\\
\text{Gain}(\textit{Windy}) &= 0.020 \text{ bits}
\end{aligned}\right\}
$$

Humidity is selected


Outlook
sunny
overcast
rain
Humidity is selected
Humidity
yes
?
normal
high
yes
no
further splitting necessary
Pure leaves
→ No further expansion necessary

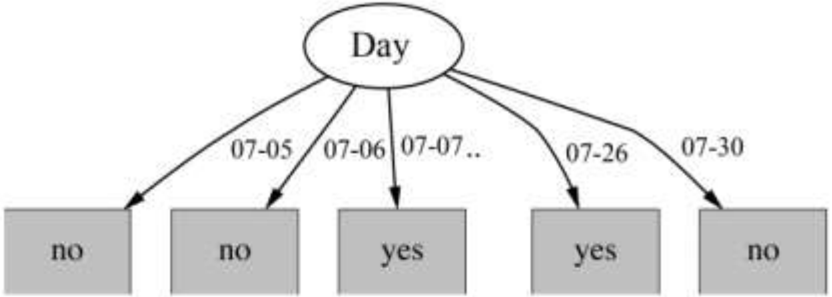
Final Decision Tree

Day	Temperature	Outlook	Humidity	Windy	Play Golf?
07-05	hot	sunny	high	false	no
07-06	hot	sunny	high	true	no
07-07	hot	overcast	high	false	yes
07-09	cool	rain	normal	false	yes
07-10	cool	overcast	normal	true	yes
07-12	mild	sunny	high	false	no
07-14	cool	sunny	normal	false	yes
07-15	mild	rain	normal	false	yes
07-20	mild	sunny	normal	true	yes
07-21	mild	overcast	high	true	yes
07-22	hot	overcast	normal	false	yes
07-23	mild	rain	high	true	no
07-26	cool	rain	normal	true	no
07-30	mild	rain	high	false	yes



- Problematic: attributes with a large number of values
  - extreme case: each example has its own value
    - e.g. example ID; Day attribute in weather data

- Problematic: attributes with a large number of values
  - extreme case: each example has its own value
    - e.g. example ID; Day attribute in weather data
- Subsets are more likely to be pure if there is a large number of different attribute values
  - Information gain is biased towards choosing attributes with a large number of values



- Entropy of split:

$$I(\text{Day}) = \frac{1}{14}\left(E([0,1]) + E([0,1]) + \ldots + E([0,1])\right) = 0$$

  - Information gain is maximal for Day (0.940 bits)

$Gain(S, Humidity)$
$= .940 - (7/14).985 - (7/14).592$
$= .151$

$Gain(S, Wind)$
$= .940 - (8/14).811 - (6/14)1.0$
$= .048$

$Gain(S, Temperature) = 0.029$

$Gain(S, Outlook) = 0.246$

Attributes with large # of values

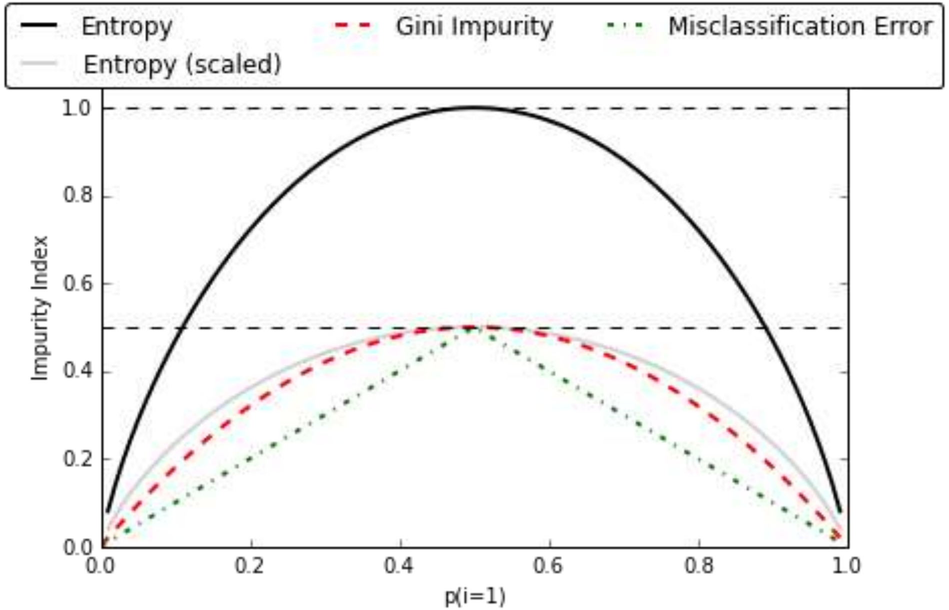
- This may cause several problems:
  - *Overfitting*
    - selection of an attribute that is non-optimal for prediction
  - *Fragmentation*
    - data are fragmented into (too) many small sets

Impurity function: candidates

Entropy: $i(V) = -\big(q \log q + (1-q)\log(1-q)\big)$

Gini index: $i(V) = 2q(1-q)$

Misclassification rate: $i(V) = \min(q, 1-q)$



Attributes with large # of values – measure

- Intrinsic information of a split
  - entropy of distribution of instances into branches
  - i.e. how much information do we need to tell which branch an instance belongs to

$$IntI(S, A) = -\sum_i \frac{|S_i|}{|S|} \log\left(\frac{|S_i|}{|S|}\right)$$

- Example:
  - Intrinsic information of Day attribute:

$$IntI(\mathrm{Day}) = 14 \times \left(-\tfrac{1}{14} \cdot \log\left(\tfrac{1}{14}\right)\right) = 3.807$$

- Observation:
  - Attributes with higher intrinsic information are less useful

Handling numerical attributes – some optimizations

- Assume a numerical attribute for Temperature
- First step:
  - Sort all examples according to the value of this attribute
  - Could look like this:

64	65	68	69	70	71	72	72	75	75	80	81	83	85
Yes	No	Yes	Yes	Yes	No	No	Yes	Yes	Yes	No	Yes	Yes	No

Handling numerical attributes – some optimizations

- Assume a numerical attribute for Temperature
- First step:
  - Sort all examples according to the value of this attribute
  - Could look like this:

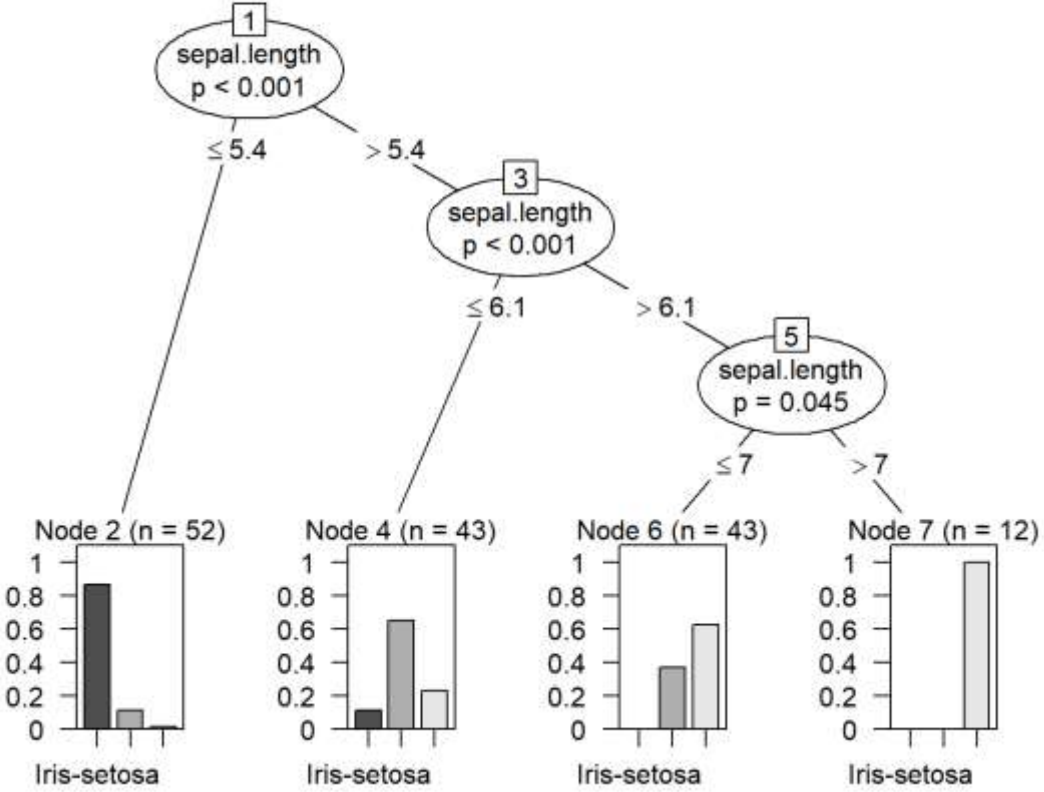
64	65	68	69	70	71	72	72	75	75	80	81	83	85
Yes	No	Yes	Yes	Yes	No	No	Yes	Yes	Yes	No	Yes	Yes	No

- One split between each pair of values
  - E.g. Temperature < 71.5: yes/4, no/2
    Temperature ≥ 71.5: yes/5, no/3

$$I(\text{Temperature} @ 71.5) = \frac{6}{14} \cdot E(\text{Temperature} < 71.5) + \frac{8}{14} E(\text{Temperature} \geq 71.5) = 0.939$$

- Split points can be placed between values or directly at values

Decision Tree with numerical attribute

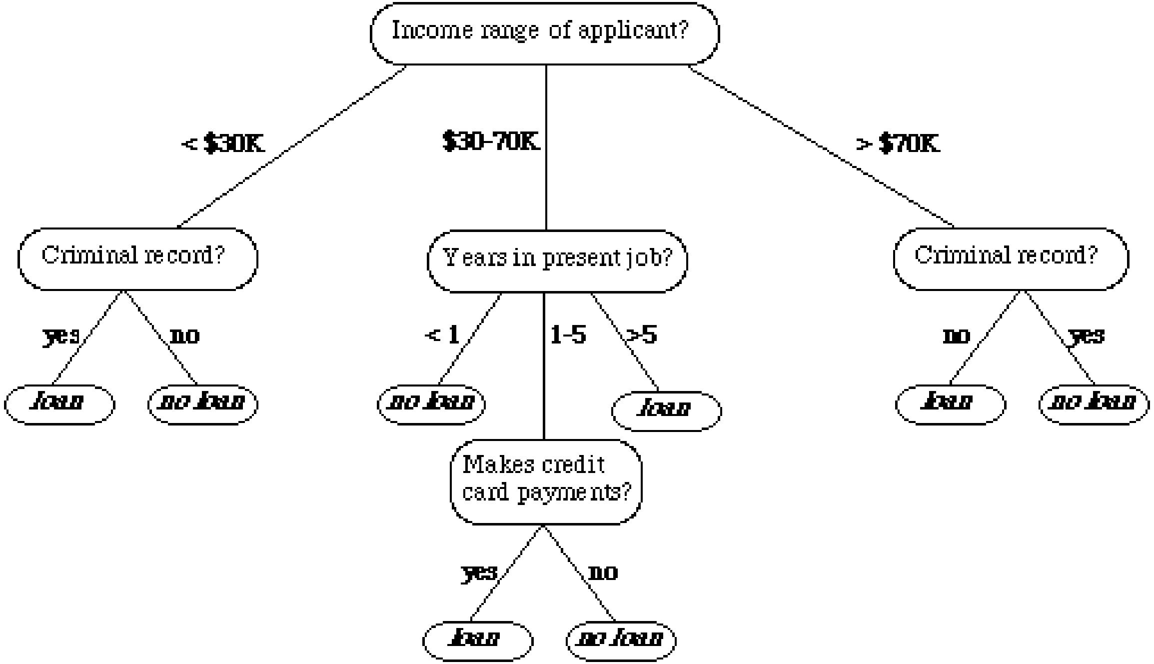


Handling numerical attributes

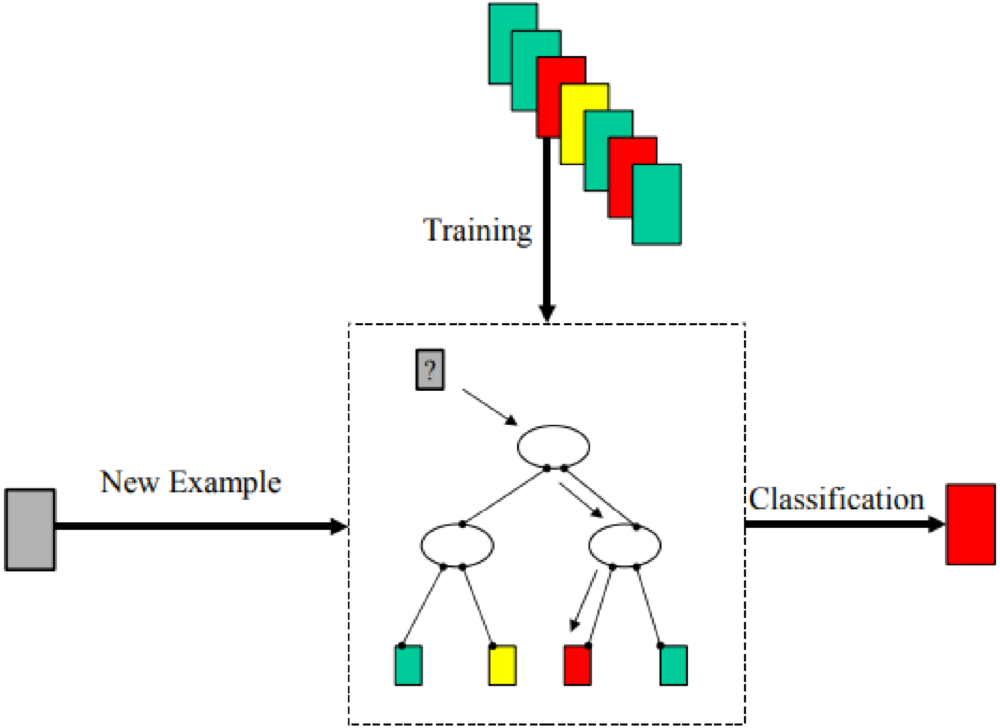
- Splitting (multi-way) on a nominal attribute exhausts all information in that attribute
  - Nominal attribute is tested (at most) once on any path in the tree
- Not so for binary splits on numerical attributes (why ?)
- ➔ Attribute may be tested multiple times in the tree
- → Tree may become hard to read

Handling numerical attributes

- Discretization / Clustering



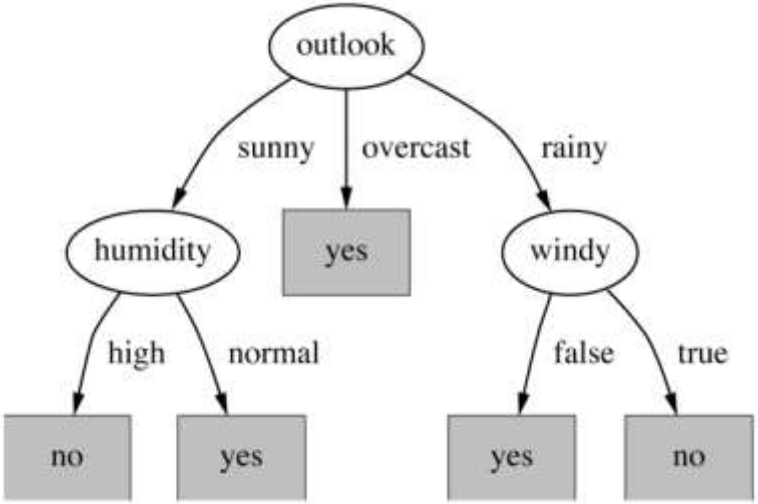
Deployment

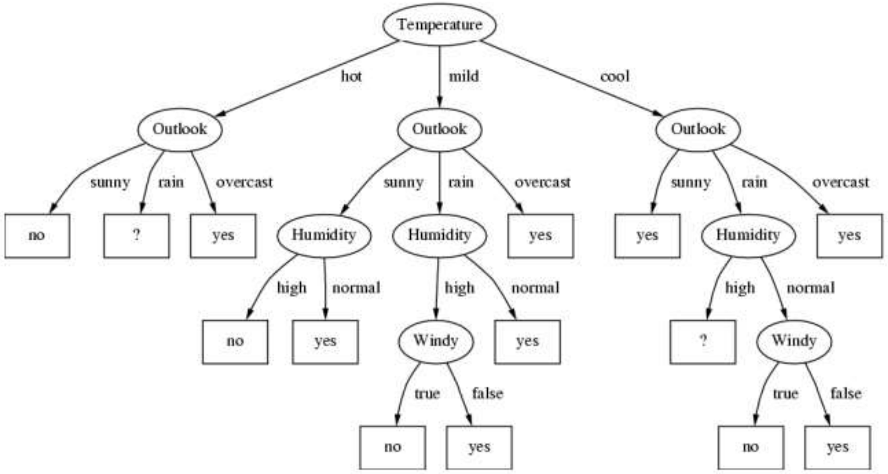


Other issues to address

- Missing attributes
- Attribute values not seen during tree induction (construction)
- Attribute missing in ‘test phase’
  - Divide into pieces etc.

Small is often better



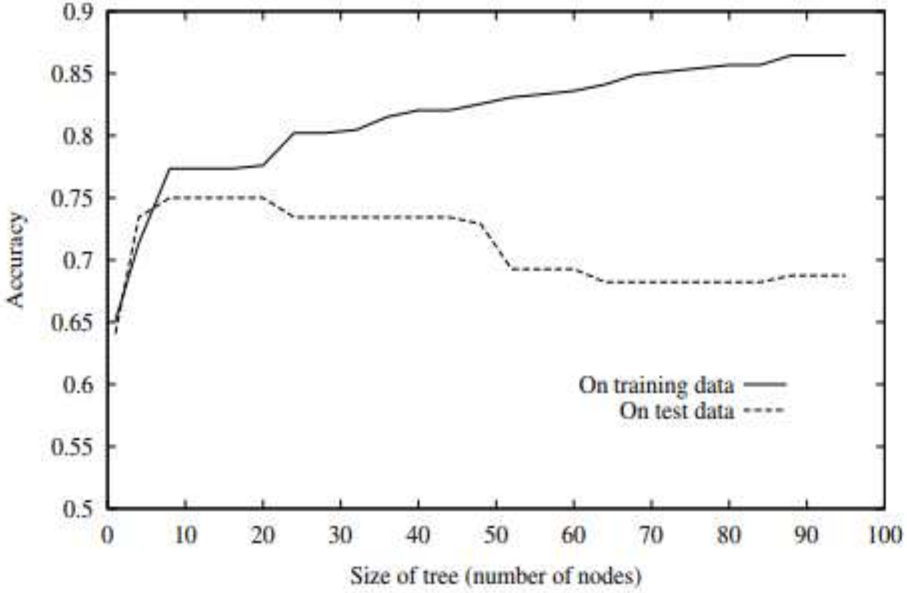


The Smallest Decision Tree

- Learning the smallest DT is NP-hard (Hyafil & Rivest ’76)
- Greedy Heuristic
  - Start from empty decision tree
  - Split on next best attribute (feature)
  - Recurse

Overfitting in Decision Trees

- Overfitting can occur with noisy training examples, and also when small numbers of examples are associated with leaf nodes (→ coincidental or accidental regularities)



Avoiding overfitting

- Pre-pruning : stop growing tree based on statistical tests of significance
- Post-pruning : Grow full tree, then prune

Reduced-Error Pruning

Split training data further into *training* and *validation* sets

Grow tree based on *training 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 node that most improves *validation set* accuracy

Decision Trees → Code

rec	Age	Income	Student	Credit_rating	Buys_computer(CLASS)
r1	<=30	High	No	Fair	No
r2	<=30	High	No	Excellent	No
r3	31…40	High	No	Fair	Yes
r4	>40	Medium	No	Fair	Yes
r5	>40	Low	Yes	Fair	Yes
r6	>40	Low	Yes	Excellent	No
r7	31…40	Low	Yes	Excellent	Yes
r8	<=30	Medium	No	Fair	No
r9	<=30	Low	Yes	Fair	Yes
r10	>40	Medium	Yes	Fair	Yes
r11	<=30	Medium	Yes	Excellent	Yes
r12	31…40	Medium	No	Excellent	Yes
r13	31…40	High	Yes	Fair	Yes
r14	>40	Medium	No	Excellent	No

IF *age* = "<=30" AND *student* = "*no*" THEN
buys_computer = "*no*"

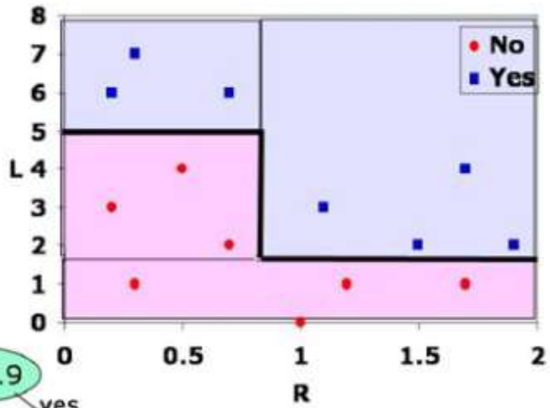
IF *age* = "<=30" AND *student* = "*yes*" THEN
buys_computer = "*yes*"

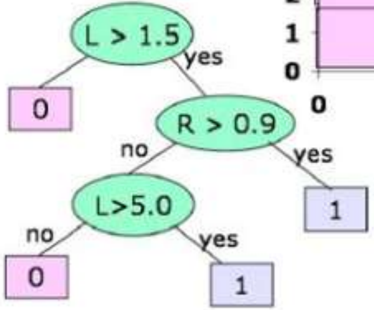
IF *age* = "31…40" THEN
buys_computer = "*yes*"

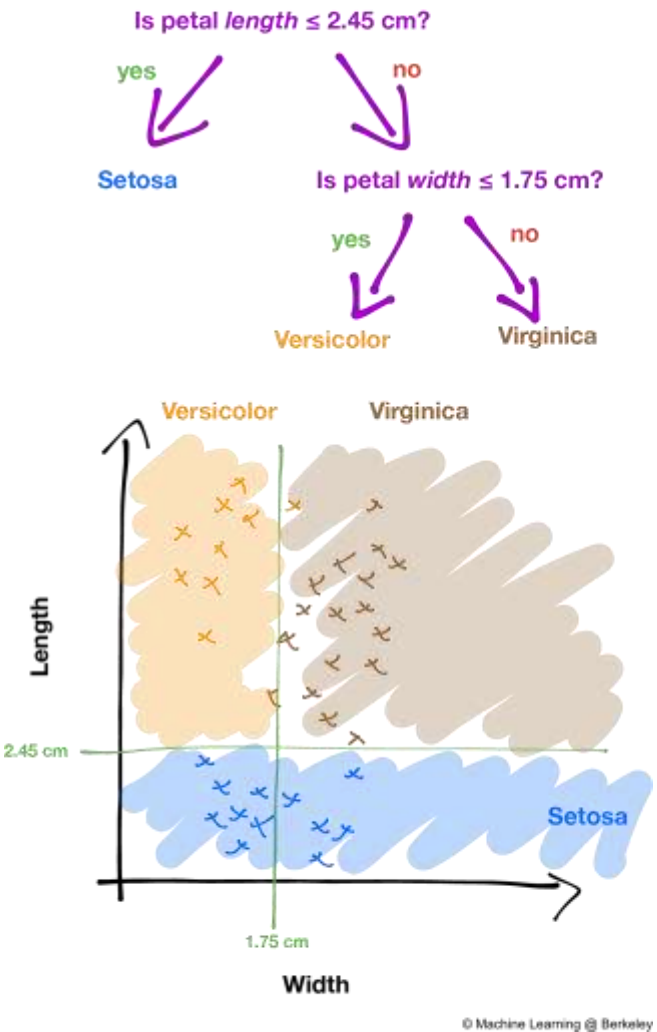
IF *age* = ">40" AND *credit_rating* = "*excellent*" THEN
buys_computer = "no"

IF *age* = ">40" AND *credit_rating* = "*fair*" THEN
buys_computer = "yes"

Decision Boundaries







Decision trees for classification

Some real examples (from Russell & Norvig, Mitchell)

- BP's GasOIL system for separating gas and oil on offshore platforms - decision trees replaced a hand-designed rules system with 2500 rules. C4.5-based system outperformed human experts and saved BP millions. (1986)
- learning to fly a Cessna on a flight simulator by watching human experts fly the simulator (1992)
- can also learn to play tennis, analyze C-section risk, etc.

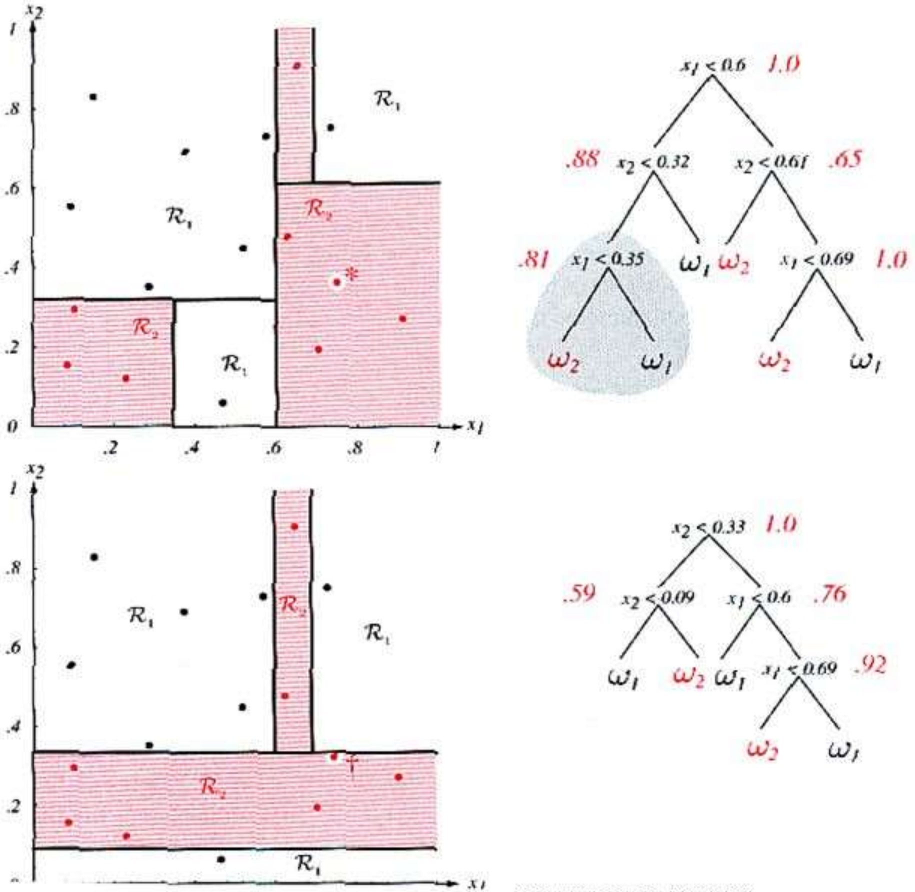
Advantages of DT

- Easy to use, understand
- Produce rules that are easy to interpret & implement
- Variable selection & reduction is automatic
- Do not require the assumptions of statistical models
- Can work without extensive handling of missing data

Disadvantages

- May not perform well where there is structure in the data that is not well captured by horizontal or vertical splits
- Since the process deals with one variable at a time, no way to capture interactions between variables

Decision Trees are not stable



Moving just one example slightly may lead to quite different trees and space partition!

Lack of stability against small perturbation of data.

Figure from Duda, Hart & Stork, Chap. 8

References and Reading

- https://en.wikipedia.org/wiki/Decision_tree_learning
- Cool demo: http://www.r2d3.us/visual-intro-to-machine-learning-part-1/
- Entropy
  - https://towardsdatascience.com/entropy-how-decision-trees-make-decisions-2946b9c18c8
  - https://plus.maths.org/content/information-surprise
  - In decision trees: https://bricaud.github.io/personal-blog/entropy-in-decision-trees/

- Textbook References
  - [TM] Machine Learning by Tom Mitchell (3.1 – 3.5, 3.7 – 3.8)
  - [PRML] Pattern Recognition and Machine Learning by Chris Bishop (1.2 (intro), 1.6)
  - [DHS] Duda and Hart (8.1 – 8.4)

- Code
  - https://scikit-learn.org/stable/modules/tree.html
  - https://scikit-learn.org/stable/auto_examples/tree/plot_unveil_tree_structure.html