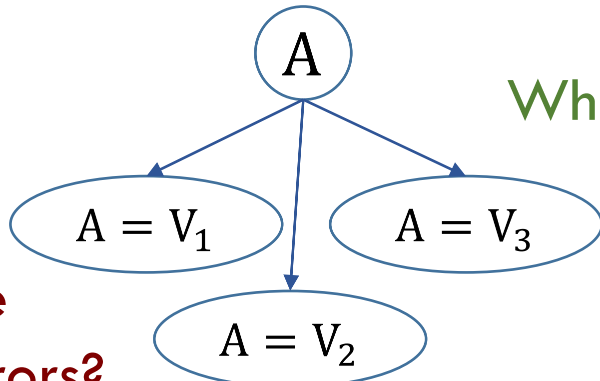
Review: 1R or 1-rule

- Simplest method to find classification rules.
- Provide "if condition then consequence" rules
- Decisions are made only based on the values of one attribute.
- We choose an attribute that have minimum total prediction error (minimum error on test data).



What is the majority class in each branch?

What are prediction errors?

location	size	pets	value
good	small	yes	high
good	big	no	high
good	big	no	high
bad	medium	no	medium
good	medium	only cats	medium
good	small	only cats	medium
bad	medium	yes	medium
bad	small	yes	low
bad	medium	yes	low
bad	small	no	low

- What is the error of the 1-R method splitting on the variable “size”?
  - What are the support and confidence of the following rule? “If size = small and pets = yes then value = high”

Decision Trees

Additional reading: Chapter 4 of Witten, Frank and Hall or Chapter 6 of Larose

Decision trees

UIC BUSINESS

- Decision trees generate “if-then” rules.
- Decision tree rules can easily be understood.
- In a lot of applications it is important to know how the model works. The ability to explain the reason for a decision is crucial. For example, a marketing professional would need complete descriptions of customer segments in order to launch a successful marketing campaign. The decision tree algorithm is ideal for these types of applications.*

* https://docs.oracle.com/cd/B28359_01/datamine.111/b28129/algo_decisiontree.htm#BABIDDDI

Example

Training examples *9yes*/*5no*

- Try to understand when to play
- Divide & Conquer
  - Split into subsets
  - Are they pure? (all yes or no)
  - If yes: stop
  - If not: repeat
- See which subset the new data falls into

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

Outlook

9 yes / 5 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 / 3 no

Split further

Cloudy

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

4 yes / 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 / 2 no

Split further

Outlook

9 yes / 5 no

Sunny

Humidity

2 yes / 3 no

High

Day	Humid	Wind
D1	High	Weak
D2	High	Strong
D8	High	Weak

Normal

Day	Humid	Wind
D9	Normal	Weak
D11	Normal	Strong

Cloudy

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

4 yes / 0 yes

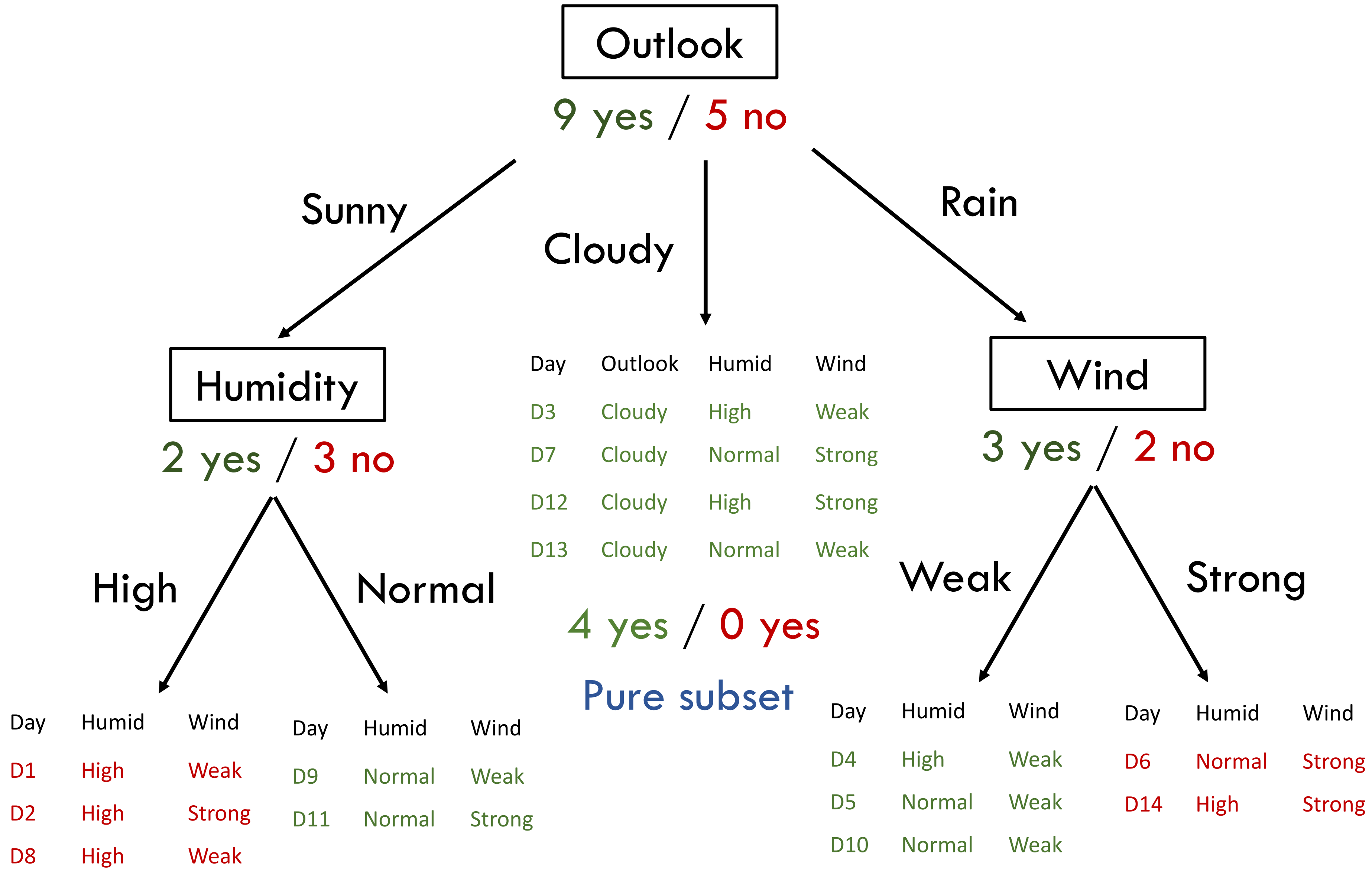
Pure subset

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 / 2 yes

Split further



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

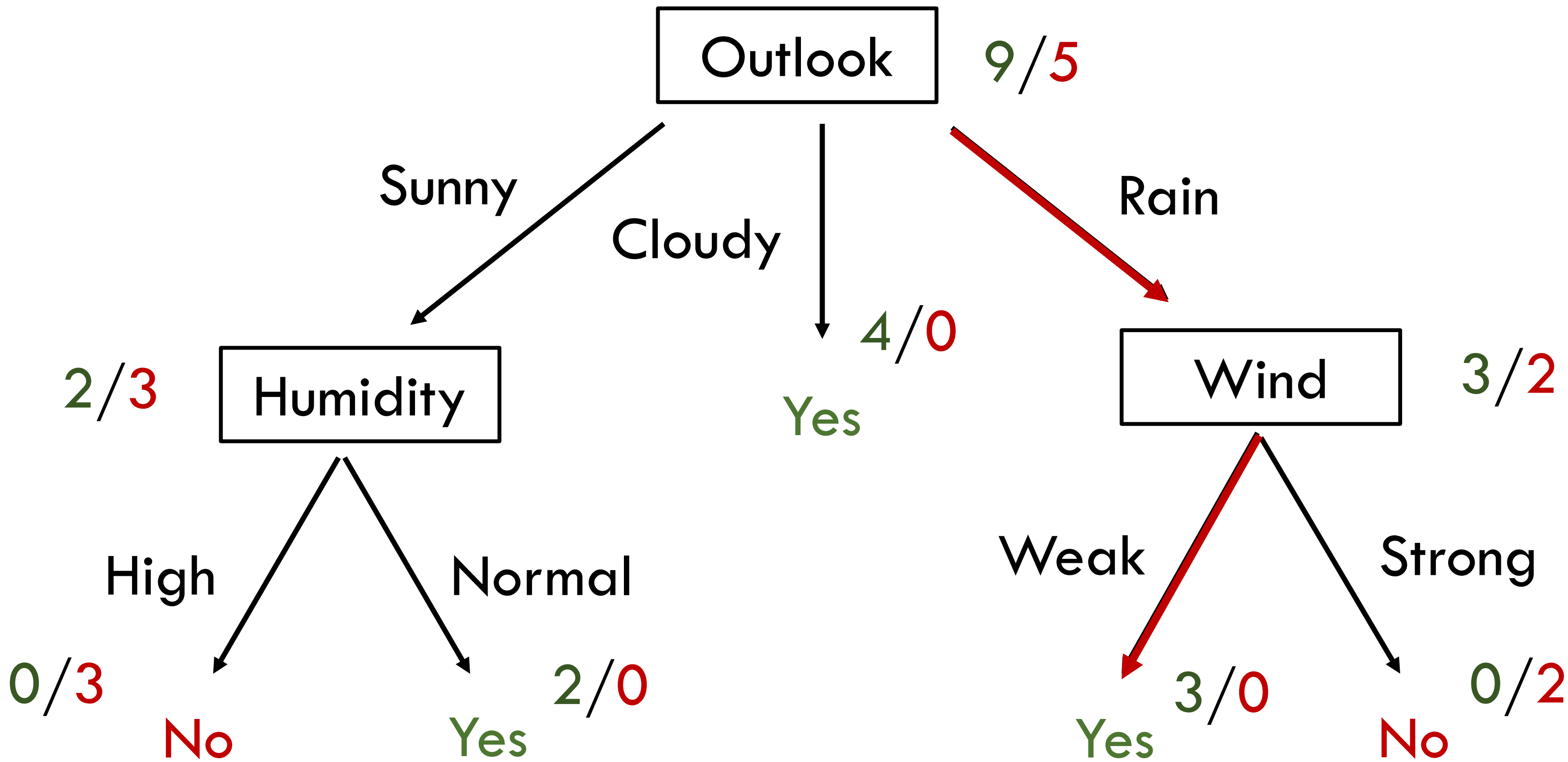
Day	Humid	Wind
D1	High	Weak
D2	High	Strong
D8	High	Weak

Day	Humid	Wind
D9	Normal	Weak
D11	Normal	Strong

Day	Humid	Wind
D4	High	Weak
D5	Normal	Weak
D10	Normal	Weak

Day	Humid	Wind
D6	Normal	Strong
D14	High	Strong

Example contd …

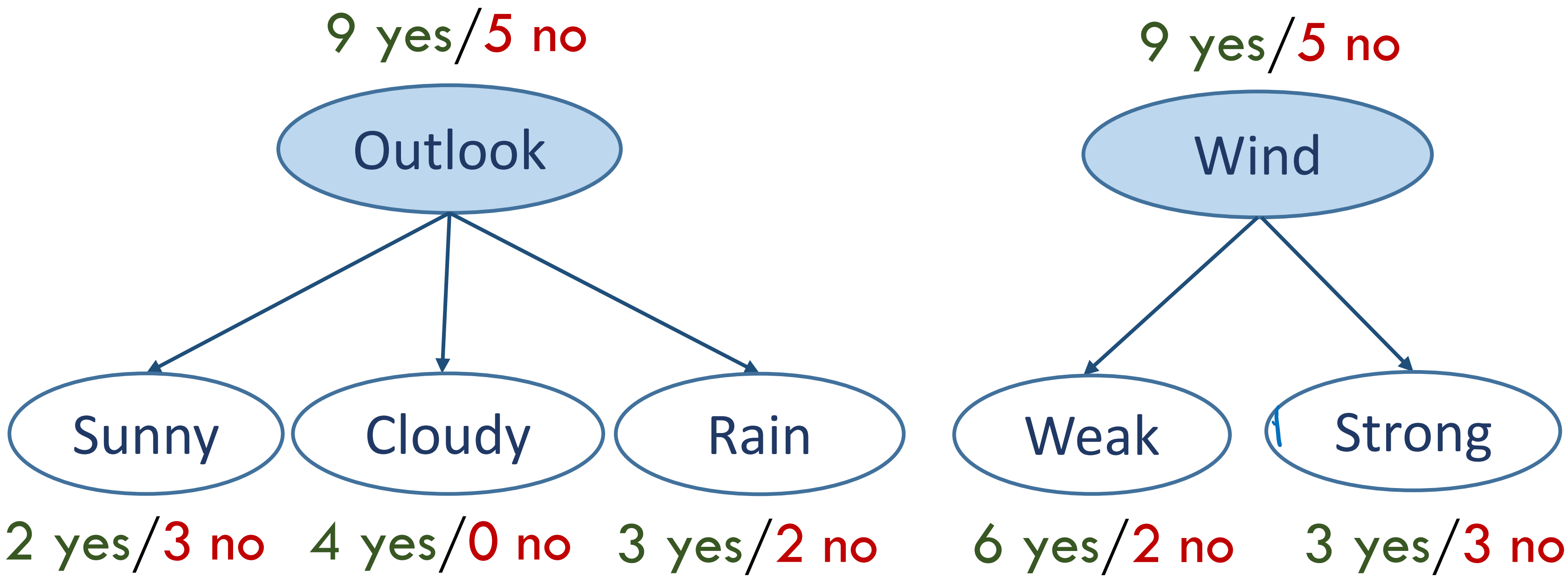


New data

Day	Outlook	Humidity	Wind	Play
D15	Rain	High	Weak	Yes

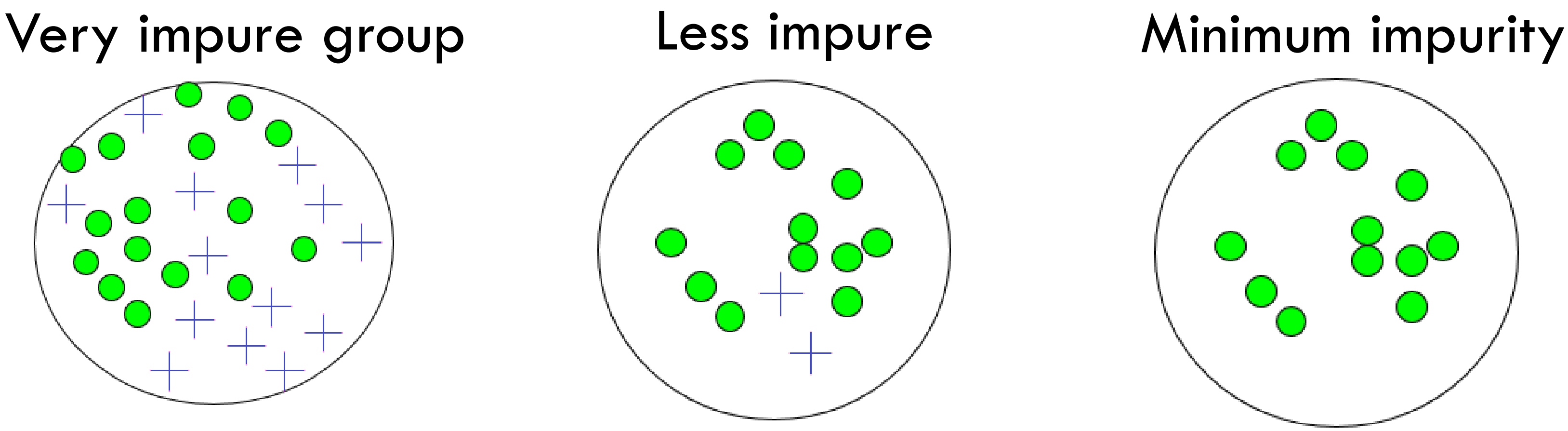
Which attribute to split on?

UIC BUSINESS



Which attribute to select?

Impurity



- Want to measure “purity” of the split
  - More certain about Yes/No after split
    - Pure set (4 yes/0 no) ⇒ completely certain (100%)
    - Impure (3 yes/3 no) ⇒ completely uncertain (50%)
  - Can’t use P(“yes”| set):
    - Must be symmetric: 4 yes/0 no as pure as 0 yes/4 no

Tests for choosing best split

UIC BUSINESS

- Impurity (Diversity) Measures:
  - Entropy (information gain)
  - Information Gain Ratio
  - Gini (population diversity)
  - Chi-square Test

Entropy as a selection criterion

UIC BUSINESS

- Entropy:

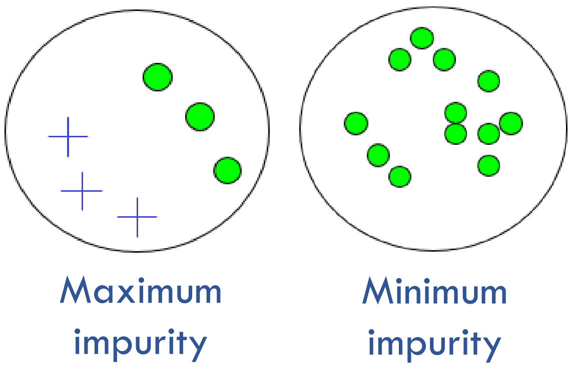
$$\text{info}(\mathrm{T}) = -p_+ \log_2 p_+ - p_- \log_2 p_-$$

  - T is the subset of training examples
  - p_+/p_- is the % of positive/negative examples in T
- Interpretation: assume item X belongs to T
  - How many bits need to tell if X positive or negative
- Impure (3 yes/ 3 no):

$$\text{info}(\mathrm{T}) = -\frac{3}{6}\log_2\frac{3}{6} - \frac{3}{6}\log_2\frac{3}{6} = 1$$

- Impure (4 yes/ 0 no):

$$\text{info}(\mathrm{T}) = -\frac{4}{4}\log_2\frac{4}{4} - \frac{0}{4}\log_2\frac{0}{4} = 0$$



Information gain

UIC BUSINESS

- Want many items in pure sets
- Expected drop in entropy after split

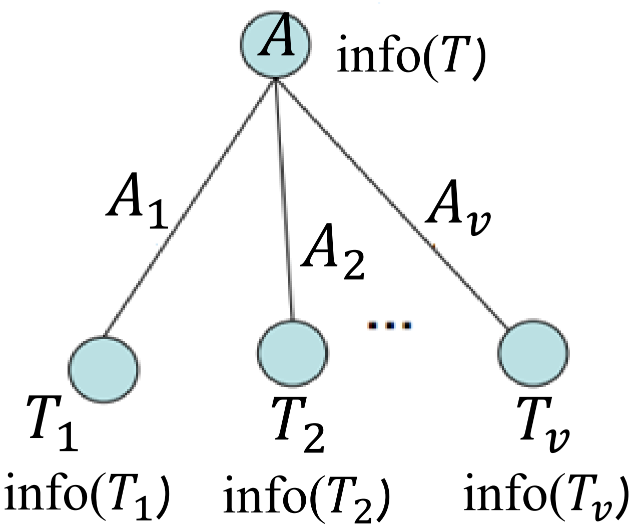
Information Gain = entropy(parent) – [average entropy(children)]

$$\mathrm{Gain}(\mathrm{T}, A) = \mathrm{info}(\mathrm{T}) - \sum_{v \in \mathrm{values}(A)} \frac{|T_v|}{|T|} \mathrm{info}(\mathrm{T}_v)$$

v : possible value in attribute A

T: current set of instances

T_v: set of all instances where $A = v$

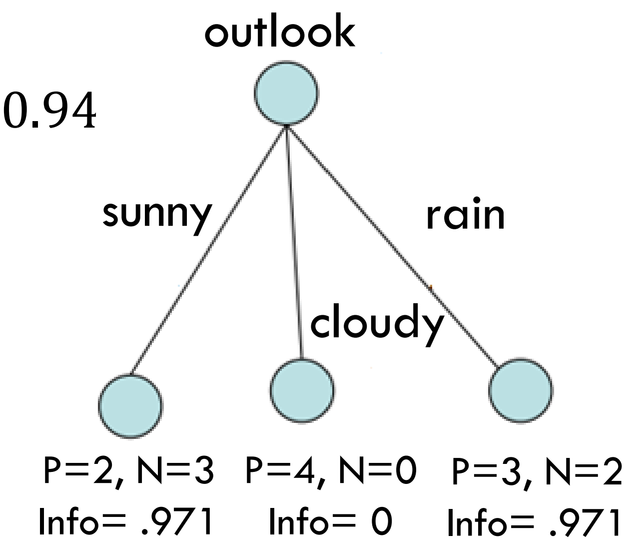


Example contd…

$$\text{Info}(T) = -\left(\frac{9}{14}\right)\log_2\left(\frac{9}{14}\right) - \left(\frac{5}{14}\right)\log_2\left(\frac{5}{14}\right) = 0.94$$

$$\text{Info}_{\text{outlook}}(T) = \left(\frac{5}{14}\right)(0.971) + \left(\frac{4}{14}\right)(0) + \left(\frac{5}{14}\right)(0.9) = 0.694$$

$$\text{Gain(outlook)} = 0.940 - 0.694 = 0.246$$

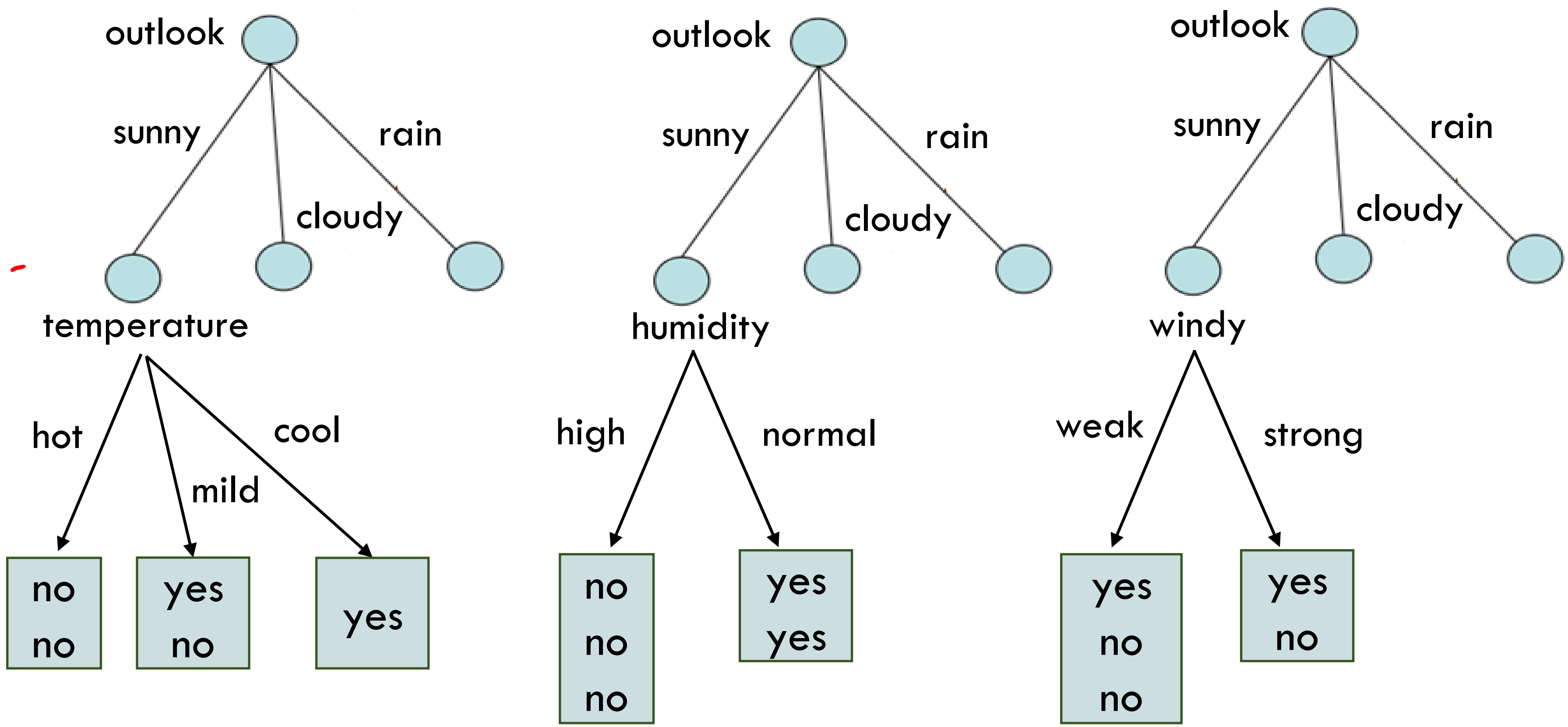


Similarly,
Gain (temp) = 0.029
Gain (humidity) = 0.151
Gain (windy) = 0.049

Select maximal gain attribute for split: outlook

Example contd…

- Once the first decision attribute has been chosen, we repeat the operation with the other nodes



Gain(temperature) = 0.57

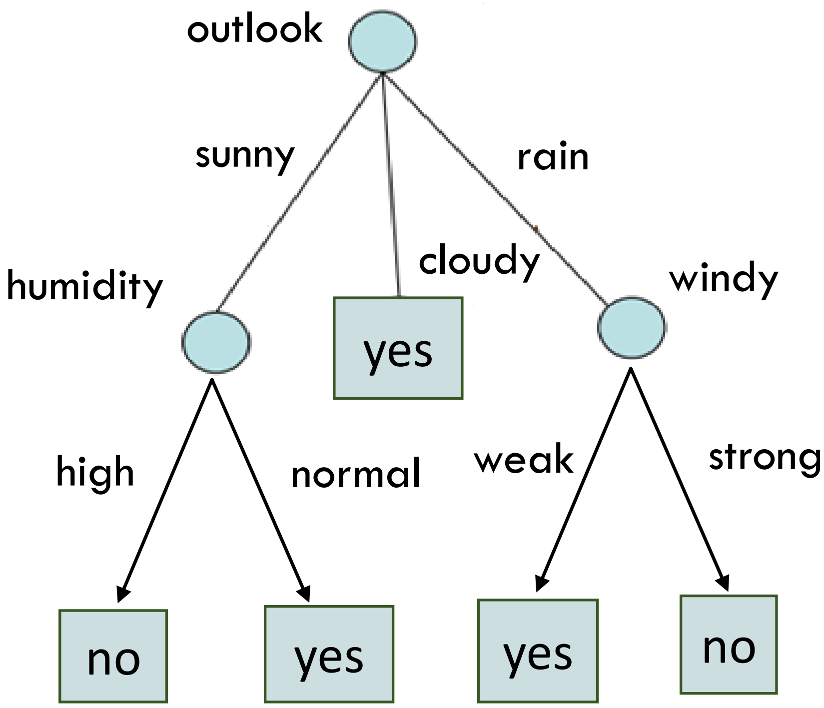
$\boxed{\text{Gain(humidity) = 0.971}}$

Gain(windy) = 0.020

Example contd…

UIC BUSINESS

- Final result



- With real examples, a certain level of impurity in each leaf node is usually tolerated.
  - Trying to learn the training data perfectly, will likely to lead overfitting.

What if we have a nominal target?

UIC BUSINESS

- Suppose the target variable has K classes.
- Entropy:

$$\text{info}(\mathrm{T}) = -\sum_{i=1}^{K} p_i \log_2 p_i$$

where p_i is the probability of class $i \in \{1, \dots, K\}$

Information gain disadvantage

UIC BUSINESS

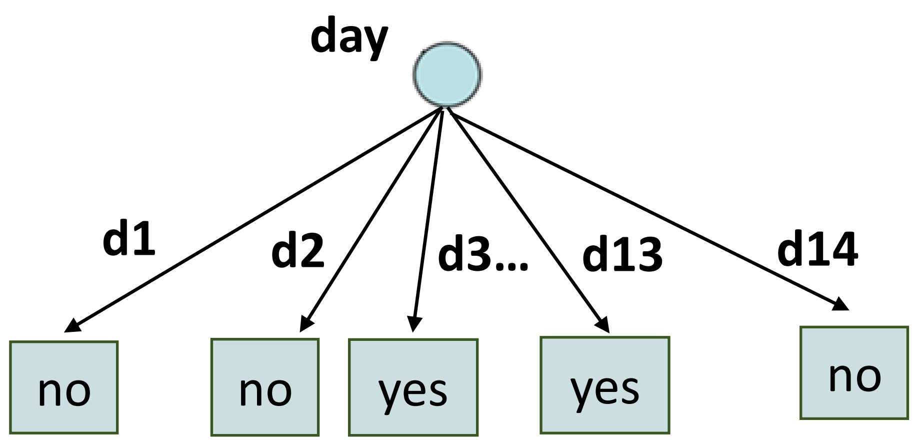
- Let's consider the attribute "Day" from tennis data set

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
d1	sunny	hot	high	weak	no
d2	sunny	hot	high	strong	no
d8	sunny	mild	high	weak	no
d9	sunny	cool	normal	weak	yes
d11	sunny	mild	normal	strong	yes
d3	cloudy	hot	high	weak	yes
d7	cloudy	cool	normal	strong	yes
d12	cloudy	mild	high	strong	yes
d13	cloudy	hot	normal	weak	yes
d4	rain	mild	high	weak	yes
d5	rain	cool	normal	weak	yes
d6	rain	cool	normal	strong	no
d10	rain	mild	normal	weak	yes
d14	rain	mild	high	strong	no

Information gain disadvantage

UIC BUSINESS

- This attribute gives us a perfect (and useless) classification!!!
- Its info gain is 0.940 bits, i.e., all the information needed to solve the problem:



- Entropy of split = 0 (since each leaf node is "pure", having only one case). Therefore, the information gain is maximal.

Gain ratio

UIC BUSINESS

- The solution to this problem is to somehow penalize the attributes that lead to a very high number of branches.
- One option is to take into account the number and size of the children nodes, regardless of what classes they contain.
- Instead of using "gain" to determine which attribute to use for the next branch, we shall use "gain ratio" which we shall define as the division between gain and the information value of the attribute.

$$\text{GainRatio(T)} = \text{Gain (T)}/\text{SplitInfo(T)}$$

Where $\text{SplitInfo(T)} = -\Sigma\left(\frac{|Ti|}{|T|}\right)\log_2\left(\frac{|Ti|}{|T|}\right)$

Example (tennis dataset)

UIC BUSINESS

- For our Day attribute:

$$\text{SplitInfo} = -\frac{1}{14} \times \log \frac{1}{14} \times 14 = 3.907$$

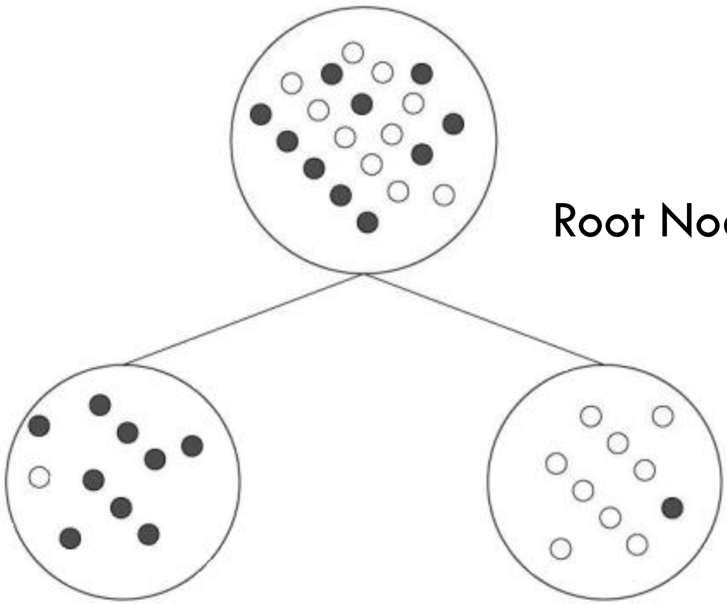
- For our Day attribute: gain ratio is $0.940/3.907 = 0.246$
- For the initial branch:

Outlook		Temperature		Humidity		Windy	
Info	0.693	Info	0.911	Info	0.788	Info	0.892
Gain	0.247	Gain	0.029	Gain	0.152	Gain	0.048
Split info	1.577	Split info	1.362	Split info	1.000	Split info	0.985
Gain ratio	0.157	Gain ratio	0.019	Gain ratio	0.152	Gain ratio	0.049

Gini index

UIC BUSINESS

- The Gini measure of a node is one minus the sum of the squares of the proportions of the classes.



Gini: $1 - P_+^2 - P_-^2$

Root Node: $1 - \left(0.5^2 + 0.5^2 \right) = 0.5$

(even balance)

Leaf Node: $1 - \left(0.1^2 + 0.9^2\right) = 0.18$

(close to pure)

- Gini Impurity can be understood as a criterion to minimize the probability of misclassification
- Gini score for subtree: weighted (by split) sum of Gini for sub-nodes
- The attribute providing smallest gini index is chosen to split the node.

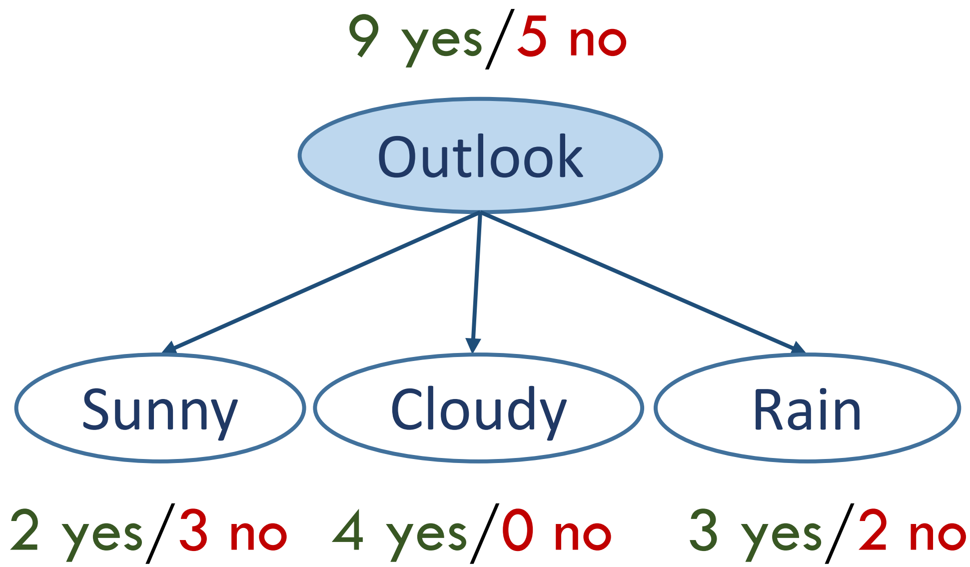
Example (tennis dataset again)

UIC BUSINESS

- Shall I play tennis?

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
d1	sunny	hot	high	weak	no
d2	sunny	hot	high	strong	no
d8	sunny	mild	high	weak	no
d9	sunny	cool	normal	weak	yes
d11	sunny	mild	normal	strong	yes
d3	cloudy	hot	high	weak	yes
d7	cloudy	cool	normal	strong	yes
d12	cloudy	mild	high	strong	yes
d13	cloudy	hot	normal	weak	yes
d4	rain	mild	high	weak	yes
d5	rain	cool	normal	weak	yes
d6	rain	cool	normal	strong	no
d10	rain	mild	normal	weak	yes
d14	rain	mild	high	strong	no

Example



$$\text{Gini(sunny)} = 1 - \left(\left(\frac{2}{5}\right)^2 + \left(\frac{3}{5}\right)^2\right) = 0.48$$

$$\text{Gini(Cloudy)} = 1 - \left(\left(\frac{4}{4}\right)^2 + \left(\frac{0}{4}\right)^2\right) = 0$$

$$\text{Gini(Rain)} = 1 - \left(\left(\frac{3}{5}\right)^2 + \left(\frac{2}{5}\right)^2\right) = 0.48$$

$$\text{Gini(Outlook)} = \left(\frac{5}{14}\right) \times 0.48 + \left(\frac{4}{14}\right) \times 0 + \left(\frac{5}{14}\right) \times 0.48 = 0.3428$$

Practical notices

UIC BUSINESS

- In practice both Gini and Entropy typically yield very similar results.
- Empirical studies show that in only 2% of the cases it matters whether you use Gini impurity or entropy.

  Introduction to Data Mining: *"Studies have shown that the choice of impurity measure has little effect on the performance of decision tree induction algorithms. This is because many impurity measures are quite consistent with each other …"*
- Entropy might be a little slower to compute (because it has to compute the logarithm).

Chi-Square based selection

UIC BUSINESS

- Chi-Square – measures probability that observations are due to sampling variation only.
- Chi-square for a node: sum of squares of standardized differences between observed and expected frequencies of all classes

For 2 classes C_1 and C_2, Sum of square of differences:

$$\left[n_{C_1} - E\left(n_{C_1}\right)\right]^2 + \left[n_{C_2} - E\left(n_{C_2}\right)\right]^2$$

n_{C_i} is the proportion of the class C_i in a node

$E(n_{C_i})$ is the proportion of the class in the parent node

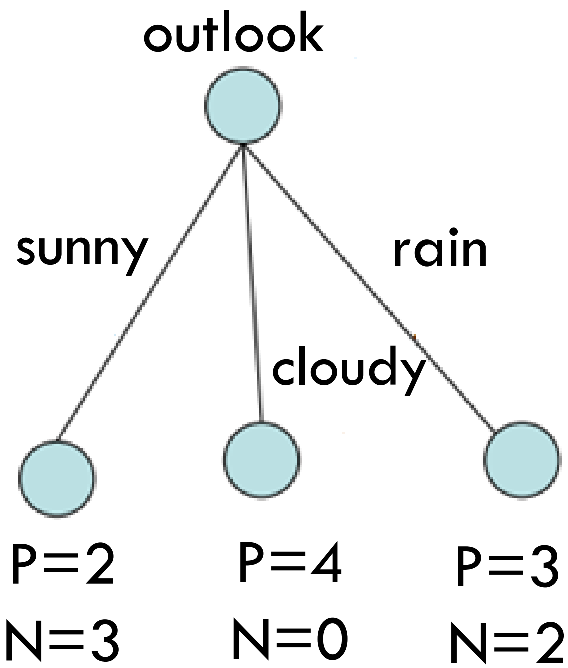
Sum of square of standardized differences:

$$\frac{\left[n_{C_1} - E\left(n_{C_1}\right)\right]^2}{E\left(n_{C_1}\right)} + \frac{\left[n_{C_2} - E\left(n_{C_2}\right)\right]^2}{E\left(n_{C_2}\right)}$$

Example

UIC BUSINESS

- Does Outlook affect the decision on whether to go out and play (P/N)?



If outlook is irrelevant to Play/Not Play decision, then expected proportion of P and N cases in sub-nodes should be in same proportion as the number of cases in the sub-nodes.

Observations	P	N	Totals
Sunny	2	3	5
Cloudy	4	0	4
Rain	3	2	5
Totals	9	5	14

Expected value of

$$[\text{Sunny, P}] = \frac{9 \times 5}{14}$$

Observations	P	N	Totals
Sunny	2 (3.21)	3 (1.79)	5
Cloudy	4 (2.57)	0 (1.43)	4
Rain	3 (3.21)	2 (1.79)	5
Totals	9	5	14

Expected value of

$$[\text{Sunny, N}] = \frac{5 \times 5}{14}$$

Observations	P	N	Totals
Sunny	2 (3.21)	3 (1.79)	5
Cloudy	4 (2.57)	0 (1.43)	4
Rain	3 (3.21)	2 (1.79)	5
Totals	9	5	14

$(\text{Obs-Exp})^2/\text{Exp}$	P	N	Totals
Sunny	0.46	0.82	1.27
Cloudy	0.79	1.43	2.23
Rain	0.01	0.03	0.04
Totals	1.27	2.27	$\boxed{3.54}$

χ^2

$$\boxed{\text{Chi-Sq} = [c_{ij} - E(c_{ij})]^2 / E(c_{ij}))}$$

Chi-Square based selection

UIC BUSINESS

- Higher Chi-square implies more significant differences so we choose an attribute which has the highest chi-square to split on.
- This method can be only used on categorical variables.
- Stops splitting when differences are no longer significant (Pre-pruning)
- CHAID – Chi-square Automated Interaction Detection

Some decision tree algorithms

UIC BUSINESS

- CHAID (Chi-Square Automatic Interaction Detection)
  - chi-square impurity test
  - used for predicting categorical variables
- CART or C&R (Classification and Regression Trees)
  - Gini impurity test
  - mostly used for binary splits
- C4.5 / C5.0 (earlier, ID3 – Iterative Dichotomizer)
  - entropy, information gain based impurity test

Numeric attributes

UIC BUSINESS

Outlook	Temperature	Humidity	Windy	Play Tennis
Sunny	Hot	High	Weak	No
Sunny	Hot	High	Strong	No
Cloudy	Hot	High	Weak	Yes
Rainy	Mild	Normal	Weak	Yes
…	…	…	…	…

If outlook = sunny and humidity = high then play = no
If outlook = rainy and windy = strong then play = no
If outlook = cloudy then play = yes
If humanity = normal then play yes
If none of the above then play yes

Numeric attributes

UIC BUSINESS

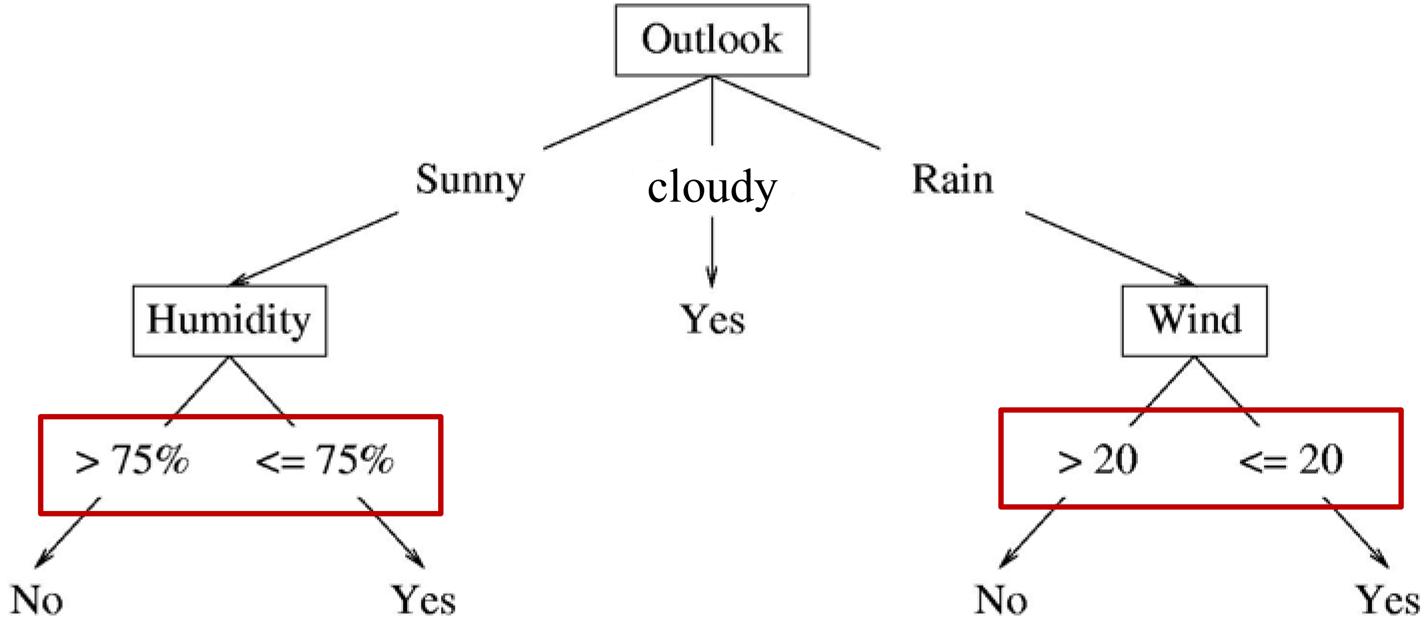
Outlook	Temperature	Humidity	Windy	Play Tennis
Sunny	85	85	Weak	No
Sunny	80	90	Strong	No
Cloudy	83	86	Weak	Yes
Rainy	75	80	Weak	Yes
…	…	…	…	…

If outlook = sunny and humidity > 83 then play = no
If outlook = rainy and windy = strong then play = no
If outlook = cloudy then play = yes
If humanity < 85 then play yes
If none of the above then play yes

Continuous (numeric) attributes

UIC BUSINESS

- If the features are continuous, internal nodes may test the value of a feature against a threshold.



How to split a numerical attribute

UIC BUSINESS

- To split on temperature attribute,
  - Check every possible cut points
  - Choose the one with the best information gain

64	65	68	69	70	71	72	72	75	75	80	81	83	85
Y	N	Y	Y	Y	N	N	Y	Y	Y	N	Y	Y	N

Place split points halfway between values

Example: temperature < 71.5: 4 yes, 2 no

Temperature ≥ 71.5: 5 yes, 3 no

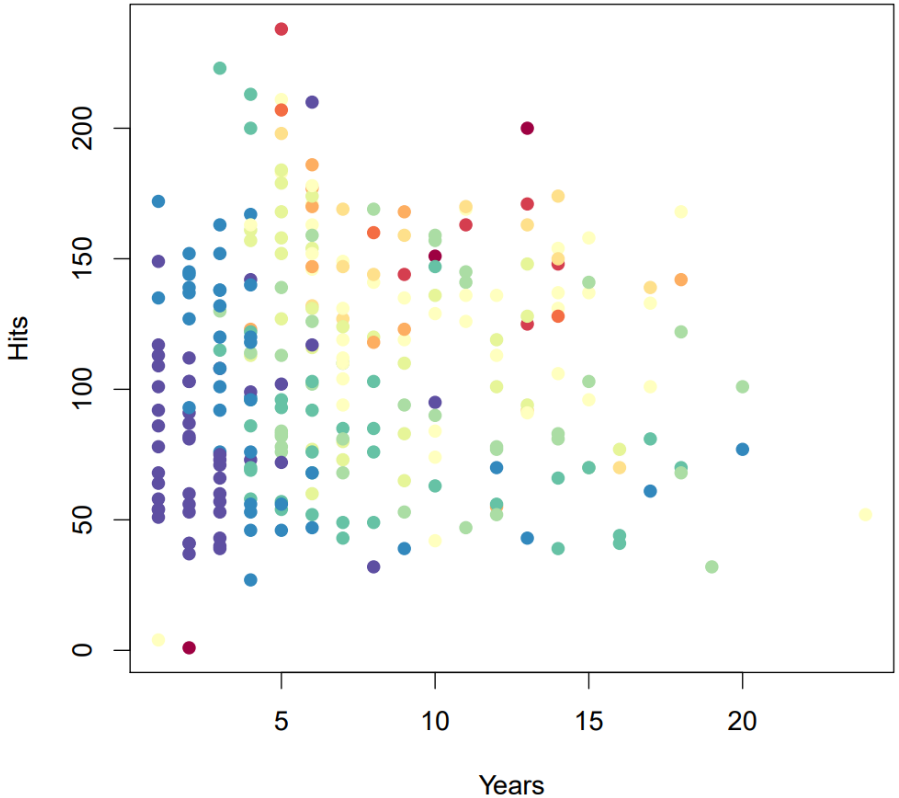
Info([4,2], [5,3]) = 6/14 Info([4,2]) + 8/14 Info([5,3]) = 0.939

Regression tree

UIC BUSINESS

Baseball salary data: how would you stratify it?

- Salary is color-coded from low (blue, green) to high (yellow,red)



Source: https://lagunita.stanford.edu/c4x/HumanitiesScience/StatLearning/asset/trees.pdf

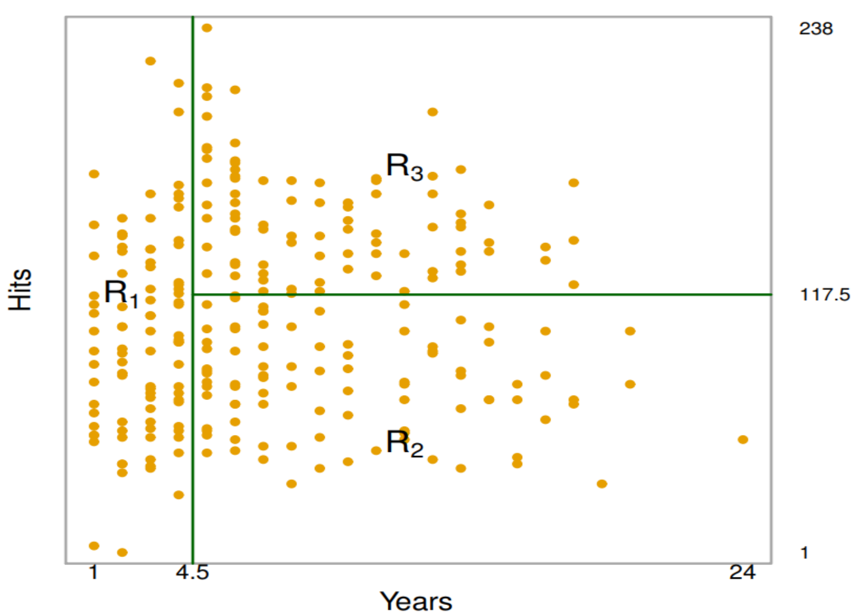
Decision tree for this data set

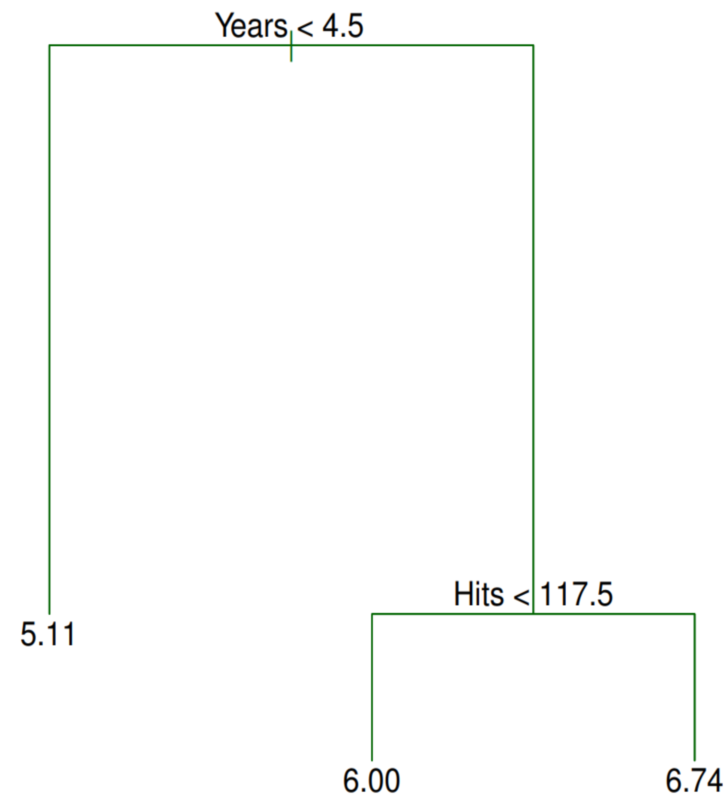
UIC BUSINESS

Overall, the tree stratifies or segments the players into three regions of predictor space:

$R1 = \{X \mid \text{Years} < 4.5\}$,

$R2 = \{X \mid \text{Years} \geq 4.5, \text{Hits} = 4.5, \text{Hits} \geq 117.5\}$.





Regression tree

UIC BUSINESS

- We divide the predictor space - that is, the set of possible values for $X_1, X_2, \ldots, X_p$ - into J distinct and non-overlapping regions, $R_1, R_2, \ldots, R_J$.

- For every observation that falls into the region R_j , we make the same prediction, which is simply the mean of the response values for the training observations in R_j .

- The goal is to find boxes $R_1, R_2, \ldots, R_J$ that minimize the RSS, given by

$$\sum_{j=1}^{J} \sum_{i \in R_j} \left(y_i - \bar{y}_{R_j}\right)^2$$

where $\bar{y}_{R_j}$ is the mean response for the training observations within the jth box.

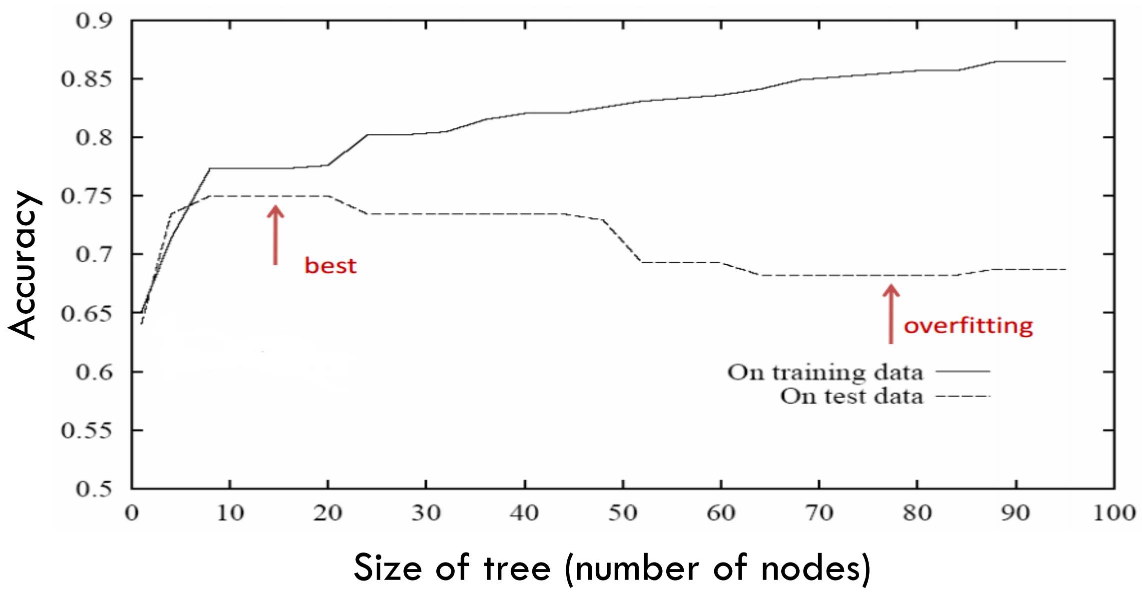
Regression tree construction

- We first select the numerical predictor X_j and the cut-point s such that splitting the predictor space into the regions $\{X|X_j < s\}$ and $\{X|X_j \geq s\}$ leads to the greatest possible reduction in RSS (Same approach works for categorical inputs).
- Next, we repeat the process, looking for the best predictor and best cut-point in order to split the data further so as to minimize the RSS within each of the resulting regions.
- The process continues until a stopping criterion is reached.
- We predict the response for a given test observation using the mean of the training observations in the region to which that test observation belongs.

Overfitting

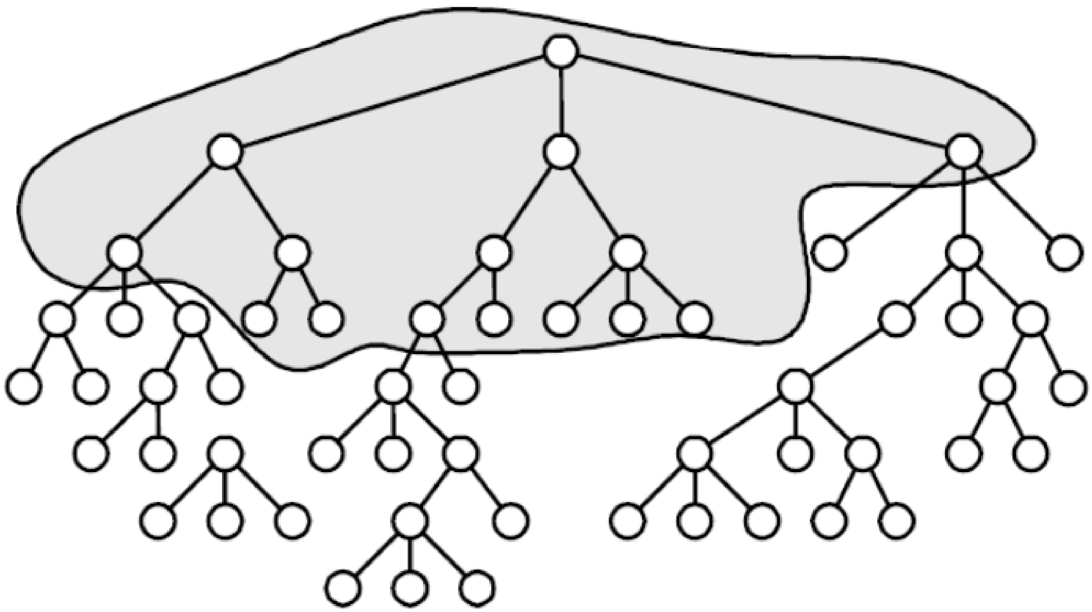
UIC BUSINESS

- Can always classify training examples perfectly?
  - Keep splitting until each node contains 1 example
  - Singleton = pure
- Does not work on new data – training error does not provide a good estimate of how well the tree will perform on unseen records



Avoid overfitting: Pruning

UIC BUSINESS



- Overfitting results in decision trees that are more complex than necessary
- Less complex trees can yield more stable models.
- To avoid overfitting do not include branches that fit data too specifically

Pruning

UIC BUSINESS

Two pruning strategies

- Pre-pruning: the process is done during the construction of the tree. There is some criteria to stop expanding the nodes (allowing a certain level of "impurity" in each node).
- Post-pruning: the process is done after the construction of the tree. Branches are removed from the bottom up to a certain limit. It uses similar criteria to pre-pruning.

* Post-pruning preferred in practice – pre-pruning can “stop too early”

Pre-pruning

UIC BUSINESS

- Possible conditions:
  - Stop if number of instances is less than some user-specified threshold
  - Stop if expanding the current node does not improve impurity measures by at least some threshold
  - Stop growing the tree when there is no *statistically significant* association between any attribute and the class at a particular node (Most popular test to check if there is any significant association is chi-squared test)
  - ID3/C5.0 used chi-squared test in addition to information gain. Only statistically significant attributes were allowed to be selected by information gain procedure

Problem: Early stopping! Pre-pruning may stop the growth process prematurely. The structure will become visible only in fully expanded tree.

Post-pruning

UIC BUSINESS

- Grow decision tree to its entirety (fully-grown tree shows all attribute interactions)
- Trim the nodes of the decision tree in a bottom-up fashion
- If generalization error estimate (error on test data) improves after trimming, replace sub-tree by a leaf-node.
- Class label of leaf node is determined from majority class of instances in the sub-tree

Estimating generalization error

UIC BUSINESS

Validation set – withhold a subset (~1/3) of training data to use for pruning

- Note: you should randomize the order of training examples

- Classify examples in validation set – some might have errors
- For each node:
  - Sum the errors over entire subtree
  - Calculate error on same example if converted to a leaf with majority class label
- Prune node with highest reduction in error
- Repeat until error no longer reduced

Estimating generalization error

UIC BUSINESS

Pessimistic error estimate -- Suppose $e(t)$ and $e'(t)$ indicate the training error and testing error in set t, respectively.

- For each leaf node compute: $e'(t) = e(t) + 0.5$
- Then the total errors is $e'(T) = e(T) + N \times 0.5$ (N: number of leaf nodes)

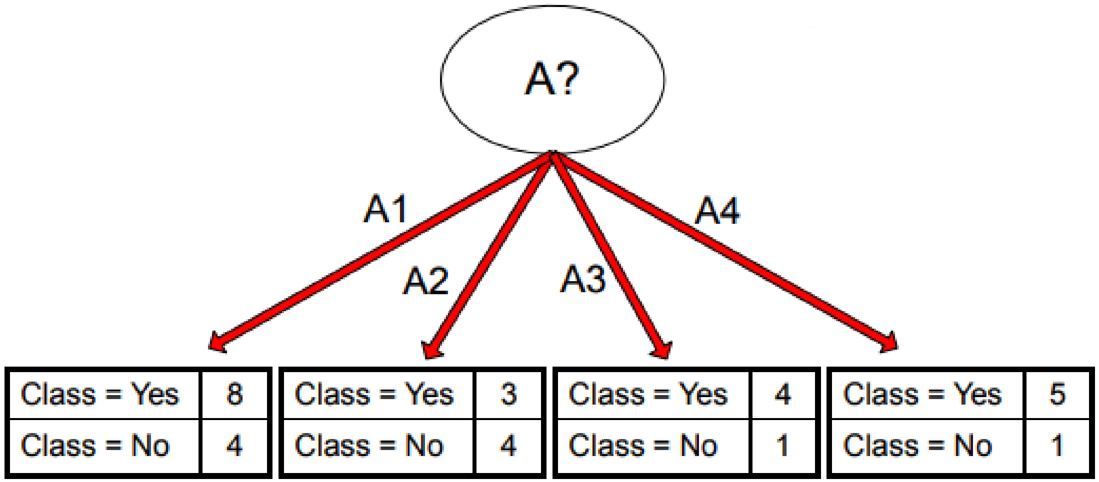
Example: For a tree with 30 nodes and 10 errors on training (out of 1000 instances):

Training error = 10/1000 = 1%

Generalization error = $(10+30 \times 0.5)/1000$ = 2.5%.

Example of post-pruning

Class = Yes	20
Class = No	10
Error = 10/30	



Training error before splitting: 10/30

Pessimistic error: (10 + 0.5)/30 = 10.5/30

Training error after splitting: 9/30

Pessimistic error after splitting: (9+4×0.5)/30 = 11/30

Decision: Prune!

https://www.ismll.uni-hildesheim.de/lehre/ml-08w/skript/decision_trees2.pdf