Decision trees. Part II

Lecture 01.02

Decision tree induction algorithm

ID3 algorithm

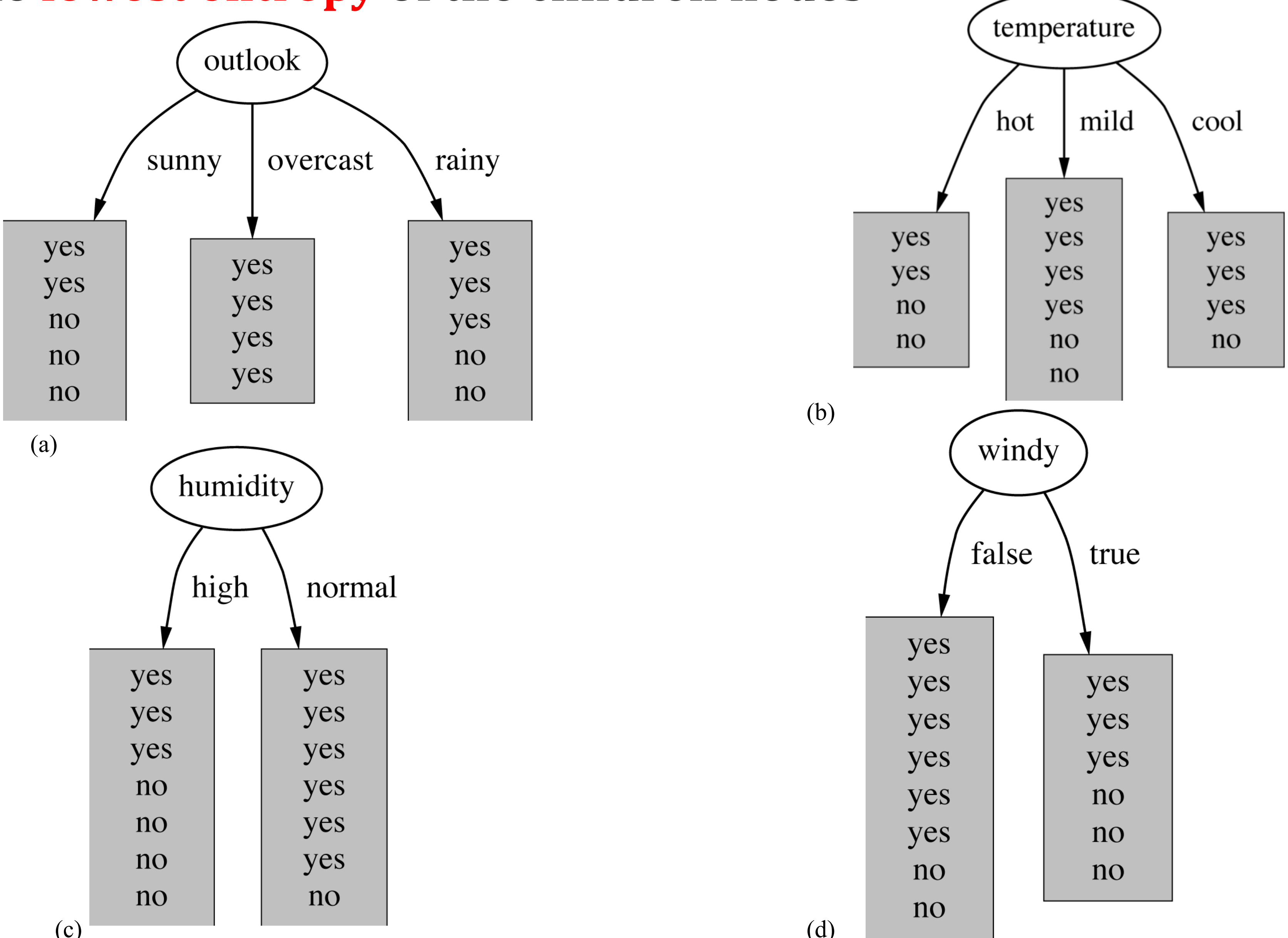
current set = all
parent entropy = entropy of *current set*

- **Step 1.**
For each attribute:
    compute entropy of a split on this *attribute*
    compute information gain vs. *parent entropy*
best attribute = attribute with maximum information gain
- **Step 2.**
create a node with *best attribute*
create branch for each possible attribute *value*
split instances into *subsets* according to the *value* of *best attribute*
- **Step 3.**
For each *subset* in *subsets*:
    If no split is possible then
        create leaf node
        mark it with the majority class
    Else
        current set = *subset*
        parent entropy = entropy of *current set*
        go to Step 1

Weather data example

Outlook	Temp	Humidity	Windy	Play
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

Choose attribute that results in the lowest entropy of the children nodes



Attribute “Outlook”

outlook=sunny

info([2,3]) = entropy(2/5,3/5) = -2/5*log(2/5,2) -3/5*log(3/5,2) = .971

outlook=overcast

info([4,0]) = entropy(4/4,0/4) = -1*log(1,2) -0*log(0,2) = 0

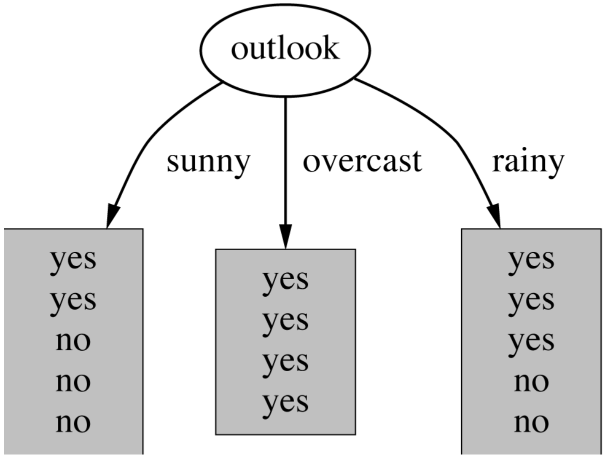
0*log(0) is normally not defined.

outlook=rainy

info([3,2]) = entropy(3/5,2/5) = -3/5*log(3/5,2)-2/5*log(2/5,2) = .971

average entropy:

.971*(5/14) + 0*(4/14) + .971*(5/14) = **.693**



Attribute "Temperature"

temperature=hot

info([2,2]) = entropy(2/4,2/4) = -2/4*log(2/4,2) -2/4*log(2/4,2) = 1

temperature=mild

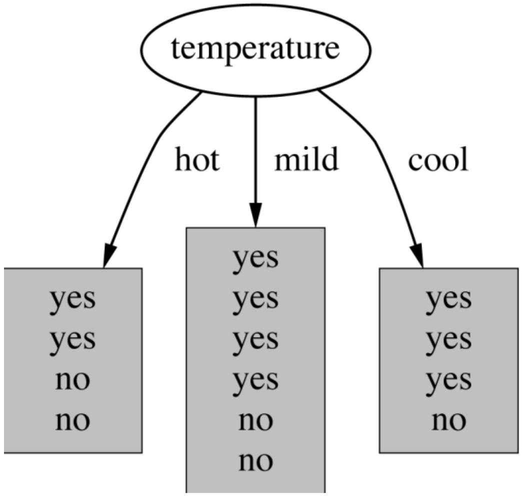
info([4,2]) = entropy(4/6,2/6) = -4/6*log(4/6,2) -2/6*log(2/6,2) = .92

temperature=cool

info([3,1]) = entropy(3/4,1/4) = -3/4*log(3/4,2)-1/4*log(1/4,2) = .811

average entropy:

1*(4/14) + .92*(6/14) + .811*(4/14) = **.91**



Attribute "Humidity"

humidity=high

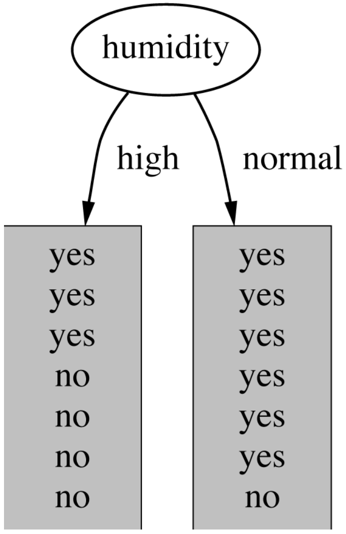
info([3,4]) = entropy(3/7,4/7) = -3/7*log(3/7,2) - 4/7*log(4/7,2) = .985

humidity=normal

info([6,1]) = entropy(6/7,1/7) = -6/7*log(6/7,2) - 1/7*log(1/7,2) = .592

average entropy:

.985*(7/14) + .592*(7/14) = **.788**



Attribute "Windy"

windy=false

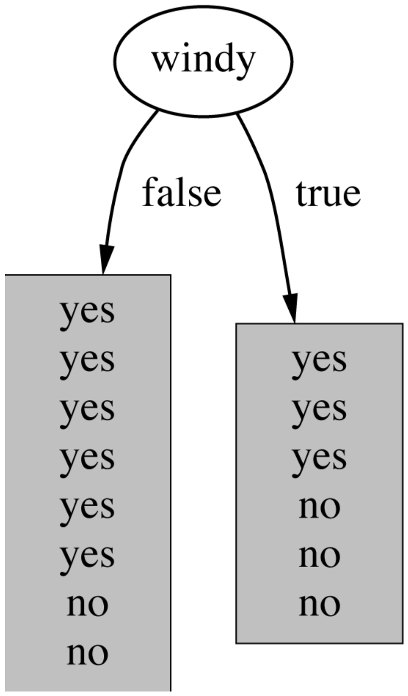
info([6,2]) = entropy(6/8,2/8) = -6/8*log(6/8,2) - 2/8*log(2/8,2) = .811

humidity=true

info([3,3]) = entropy(3/6,3/6) = -3/6*log(3/6,2) - 3/6*log(3/6,2) = 1

average entropy:

.811*(8/14) + 1*(6/14) = **.892**



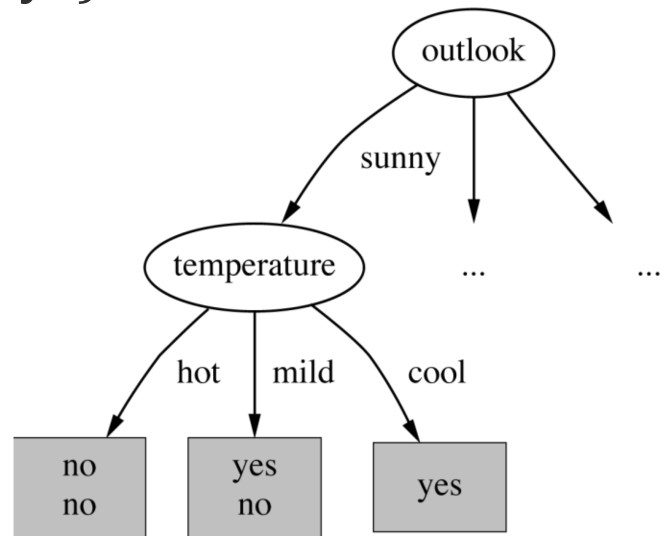
And the winner is...

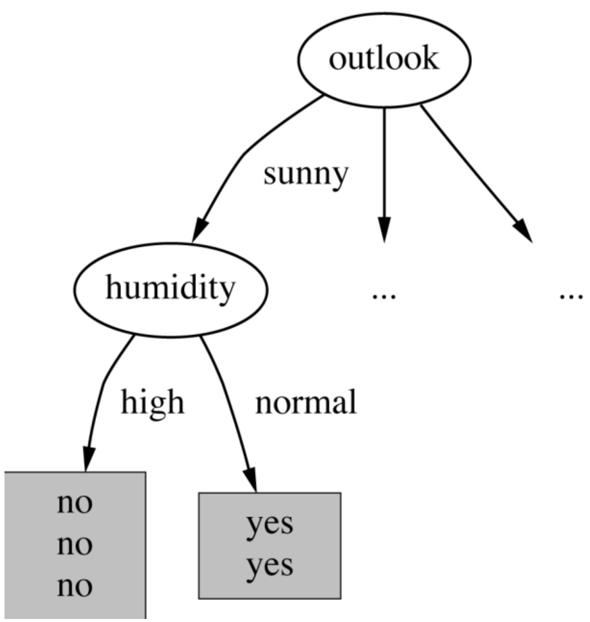
"Outlook"

...So, the root will be "Outlook"

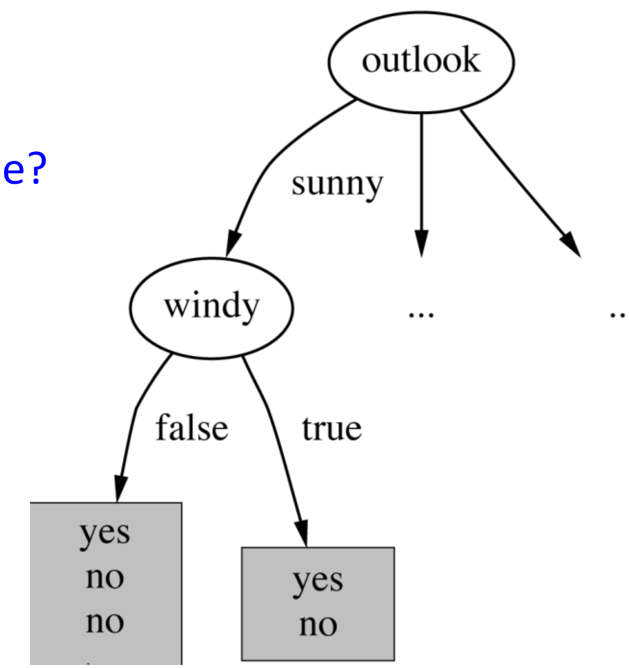
Continuing to split (for Outlook="Sunny")

Outlook	**Temp**	**Humidity**	**Windy**	**Play**
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

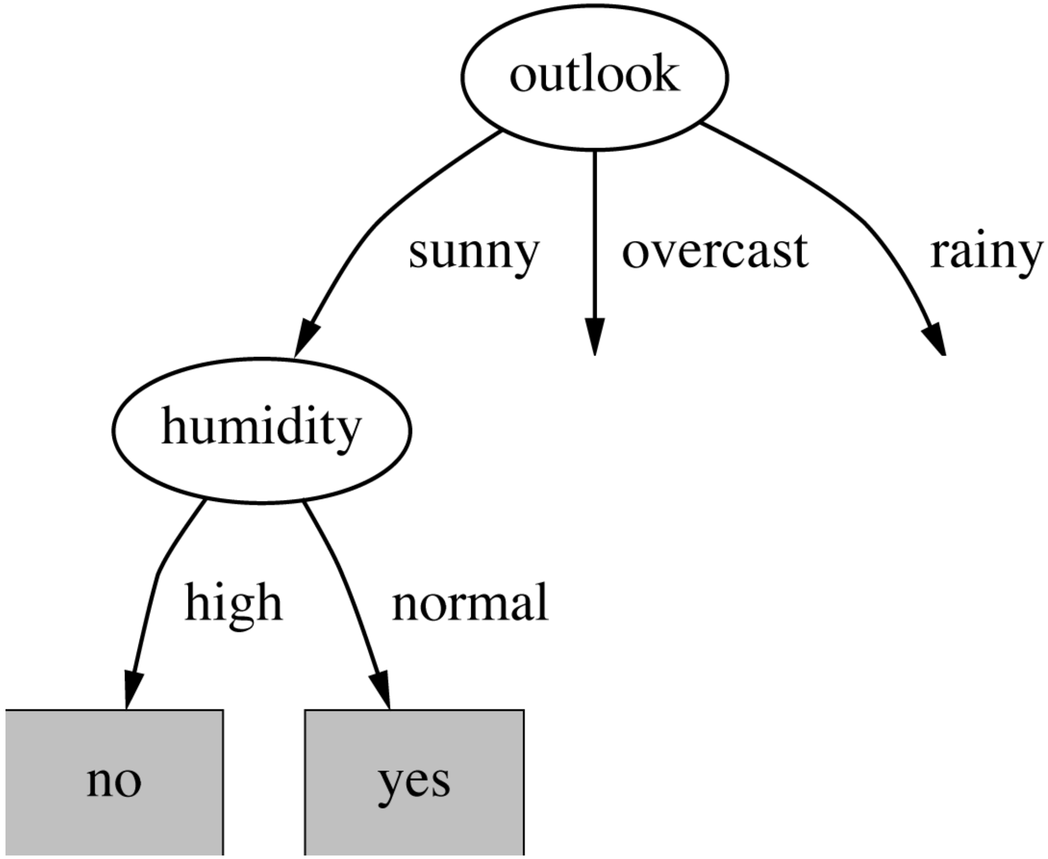




Which one to choose?



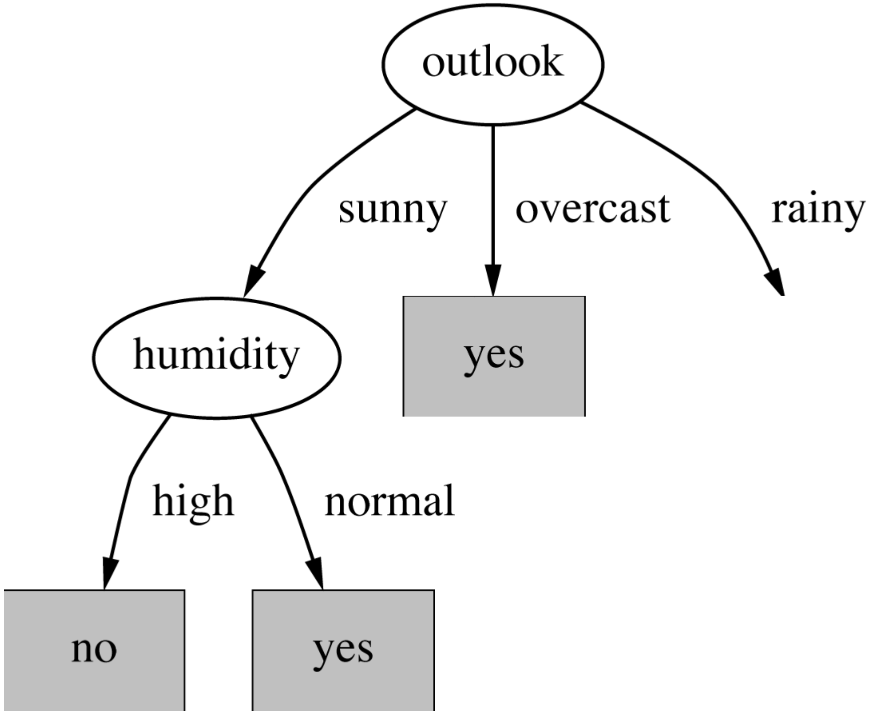
Tree so far



Continuing to split (for Outlook="Overcast")

Outlook	**Temp**	**Humidity**	**Windy**	**Play**
Overcast	Hot	High	False	Yes
Overcast	Cool	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes

- Nothing to split here, "play" is always "yes".



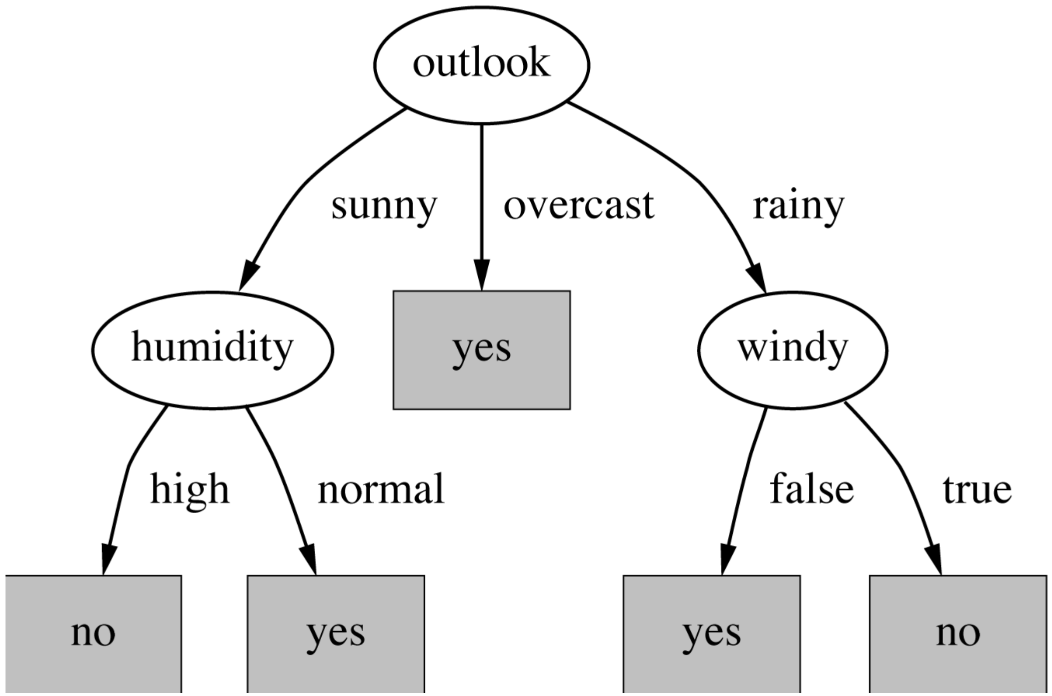
Tree so far

Continuing to split (for Outlook=“Rainy")

Outlook	**Temp**	**Humidity**	**Windy**	**Play**
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

- We see that "Windy" is the one to choose. (**Why?**)

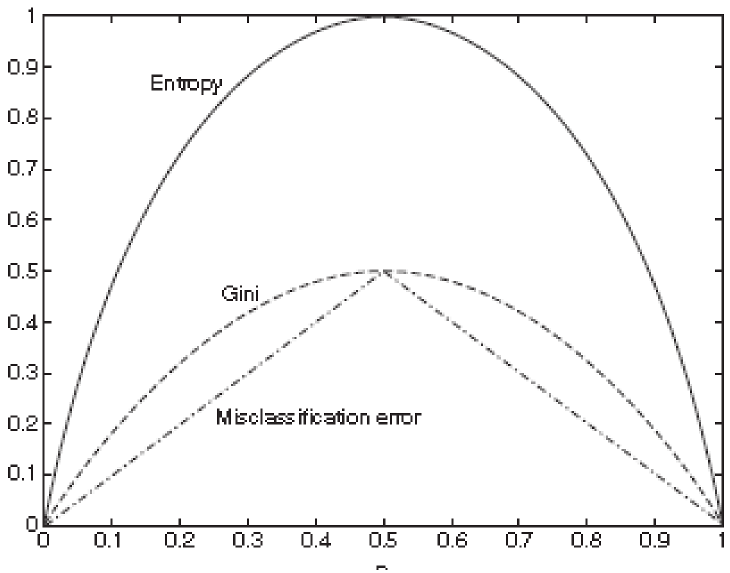
The final decision tree



- Note: not all leaves need to be pure; sometimes identical instances have different classes
- Splitting stops when data can't be split any further or there is no information gain

Split criteria

- The GINI score is maximized
  ⇔(1.0-GINI (GINI impurity) score is minimized)
- The entropy of a split is minimized
  ⇔(the information gain is maximized)



There are many other attribute selection criteria!
(But almost no difference in accuracy)

When to stop splitting

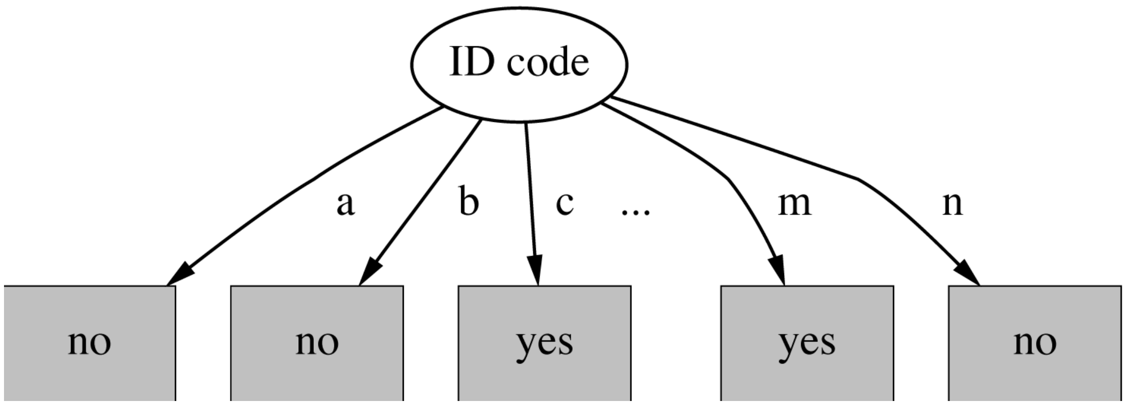
- Not to split: all records are of the same class
- Not to split: all records have the same attribute values
- Not to split: when there is no information gain or information gain is not significant
- In practice: when the number of records in the leaves is below predefined statistically significant number (30?)

The weather data with ID code

ID code	Outlook	Temp.	Humidity	Windy	Play
A	Sunny	Hot	High	False	No
B	Sunny	Hot	High	True	No
C	Overcast	Hot	High	False	Yes
D	Rainy	Mild	High	False	Yes
E	Rainy	Cool	Normal	False	Yes
F	Rainy	Cool	Normal	True	No
G	Overcast	Cool	Normal	True	Yes
H	Sunny	Mild	High	False	No
I	Sunny	Cool	Normal	False	Yes
J	Rainy	Mild	Normal	False	Yes
K	Sunny	Mild	Normal	True	Yes
L	Overcast	Mild	High	True	Yes
M	Overcast	Hot	Normal	False	Yes
N	Rainy	Mild	High	True	No

- ID3 algorithm
- Design issues
  - Split criteria
  - Stop criteria
  - Multi-valued attributes
- Applications
- Limitations
- Real-life examples
- Extracting rules from trees

The best split on ID code



- Entropy of split:

$$\text{info}(\text{"ID code"}) = \text{info}([0,1]) + \text{info}([0,1]) + \ldots + \text{info}([0,1]) = 0 \text{ bits}$$

$\Rightarrow$ Information gain is maximal for ID code (namely 0.940 bits)

However this tree is of no use for classification!

Highly-branching attributes

- ID3 algorithm
- Design issues
  - Split criteria
  - Stop criteria
  - Multi-valued attributes
- Applications
- Limitations
- Real-life examples
- Extracting rules from trees

- Subsets are more likely to be pure if there is a large number of values (pure but small)
  - Information gain is biased towards multi-valued attributes

My neighbor dataset

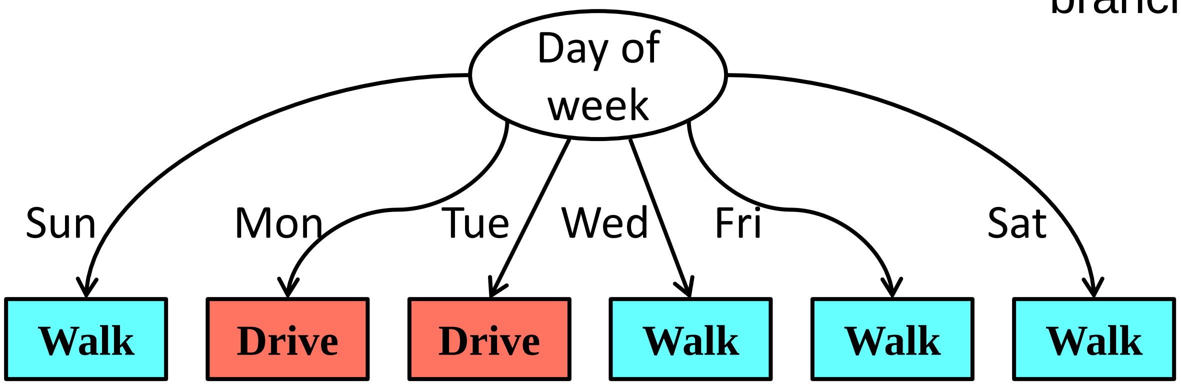
Temp	Precip	Day	Clothes	
22	None	Fri	Casual	**Walk**
3	None	Sun	Casual	**Walk**
10	Rain	Wed	Casual	**Walk**
30	None	Mon	Casual	**Drive**
20	None	Sat	Formal	**Drive**
25	None	Sat	Casual	**Drive**
-5	Snow	Mon	Casual	**Drive**
27	None	Tue	Casual	**Drive**
24	Rain	Mon	Casual	**?**

- ID3 algorithm
- Design issues
  - Split criteria
  - Stop criteria
  - Multi-valued attributes
- Applications
- Limitations
- Real-life examples
- Extracting rules from trees

The best attribute: day of week

Temp	Precip	Day	Clothes	
22	None	Fri	Casual	**Walk**
3	None	Sun	Casual	**Walk**
10	Rain	Wed	Casual	**Walk**
30	None	Mon	Casual	**Drive**
20	None	Sat	Formal	**Drive**
25	None	Sat	Casual	**Drive**
-5	Snow	Mon	Casual	**Drive**
27	None	Tue	Casual	**Drive**
24	Rain	Thu	Casual	**?**

No branch



- ID3 algorithm
- Design issues
  - Split criteria
  - Stop criteria
  - Multi-valued attributes
- Applications
- Limitations
- Real-life examples
- Extracting rules from trees

Solution: the gain ratio

- Intrinsic information: entropy (with respect to the attribute on focus) of the node to be split.
- Gain ratio: information gain divided by intrinsic information of the split

Computing the gain ratio

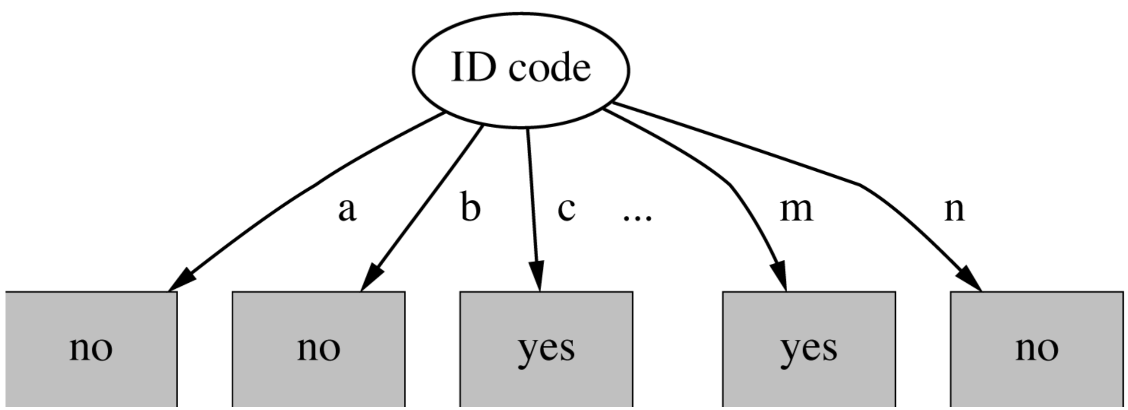
- Example: intrinsic information for ID code

$$\text{info}([1,1,\ldots,1]) = 14 \times (-1/14 \times \log 1/14) = 3.807 \text{ bits}$$

- Value of attribute decreases as intrinsic information gets larger
- Definition of gain ratio:

$$\text{gain\_ratio}(\text{"Attribute"}) = \frac{\text{gain}(\text{"Attribute"})}{\text{intrinsic\_info}(\text{"Attribute"})}$$

- Example: $\text{gain\_ratio}(\text{"ID\_code"}) = \frac{0.940 \text{ bits}}{3.807 \text{ bits}} = 0.246$



- ID3 algorithm
- Design issues
  - Split criteria
  - Stop criteria
  - Multi-valued attributes
- Applications
- Limitations
- Real-life examples
- Extracting rules from trees

Gain ratio vs. information gain

Temp	Precip	Day	Clothes	
Warm	None	Fri	Casual	**Walk**
Chilly	None	Sun	Casual	**Walk**
Chilly	Rain	Wed	Casual	**Walk**
Warm	None	Mon	Casual	**Drive**
Warm	None	Sat	Formal	**Drive**
Warm	None	Sat	Casual	**Drive**
Cold	Snow	Mon	Casual	**Drive**
Warm	None	Tue	Casual	**Drive**
Warm	Rain	Thu	Casual	**?**

All: Info(3,5)=0.95

Temp: 5/8 Info(1,4)+2/8 Info(2,0)+1/8 Info(1,0)=0.45

Precip: 6/8 Info(2,4)+ 1/8 Info(1.0) + 1/8 Info(1,0)=0.67

Day: $\boxed{0}$

Clothes: 7/8 Info(3,4)+1/8 Info (1,0)=0.86

- ID3 algorithm
- Design issues
  - Split criteria
  - Stop criteria
  - Multi-valued attributes
- Applications
- Limitations
- Real-life examples
- Extracting rules from trees

Gain ratio vs. information gain

Temp	Precip	Day	Clothes	
Warm	None	Fri	Casual	**Walk**
Chilly	None	Sun	Casual	**Walk**
Chilly	Rain	Wed	Casual	**Walk**
Warm	None	Mon	Casual	**Drive**
Warm	None	Sat	Formal	**Drive**
Warm	None	Sat	Casual	**Drive**
Cold	Snow	Mon	Casual	**Drive**
Warm	None	Tue	Casual	**Drive**
Warm	Rain	Thu	Casual	**?**

Attribute	Info gain	Intrinsic entropy	Gain ratio
Temp	0.50	Info(5,2,1)=1.29	0.54/1.29=0.38
Precip	0.28	Info(6,1,1)=1.06	0.28/1.06=0.26
Day	0.95	Info(1,1,1,2,2,1)=2.5	0.95/2.5=0.38
Clothes	0.09	Info(7,1)=0.54	0.09/0.54=0.17

- ID3 algorithm
- Design issues
  - Split criteria
  - Stop criteria
  - Multi-valued attributes
- Applications
- Limitations
- Real-life examples
- Extracting rules from trees

Learning algorithms: requirements

- For an algorithm to be useful in a wide range of real-world applications it must:
  - Permit numeric attributes
  - Allow missing values
  - Work in the presence of noise

Basic schemes need to be extended to fulfill these requirements

Weather data – temperature categories

Temp
Hot
Warm
Warm
Hot
Hot
Warm
Warm
Hot

In Canada

Temp
30
15
16
27
25
17
17
35

Temp
Hot
Chilly
Chilly
Warm
Warm
Chilly
Chilly
Hot

- ID3 algorithm
- Design issues
  - Split criteria
  - Stop criteria
  - Multi-valued attributes
  - Numeric attributes
  - Missing values
  - Overfitting
- Applications
- Limitations
- Real-life examples
- Extracting rules from trees

Weather data – temperature categories

Temp
Warm
Chilly
Chilly
Cold
Cold
Chilly
Chilly
Warm

In India
←

Temp
30
15
16
27
25
17
17
35

→

Temp
Hot
Chilly
Chilly
Warm
Warm
Chilly
Chilly
Hot

The weather *categories* are arbitrary.

Meaningful breakpoints in continuous attributes?

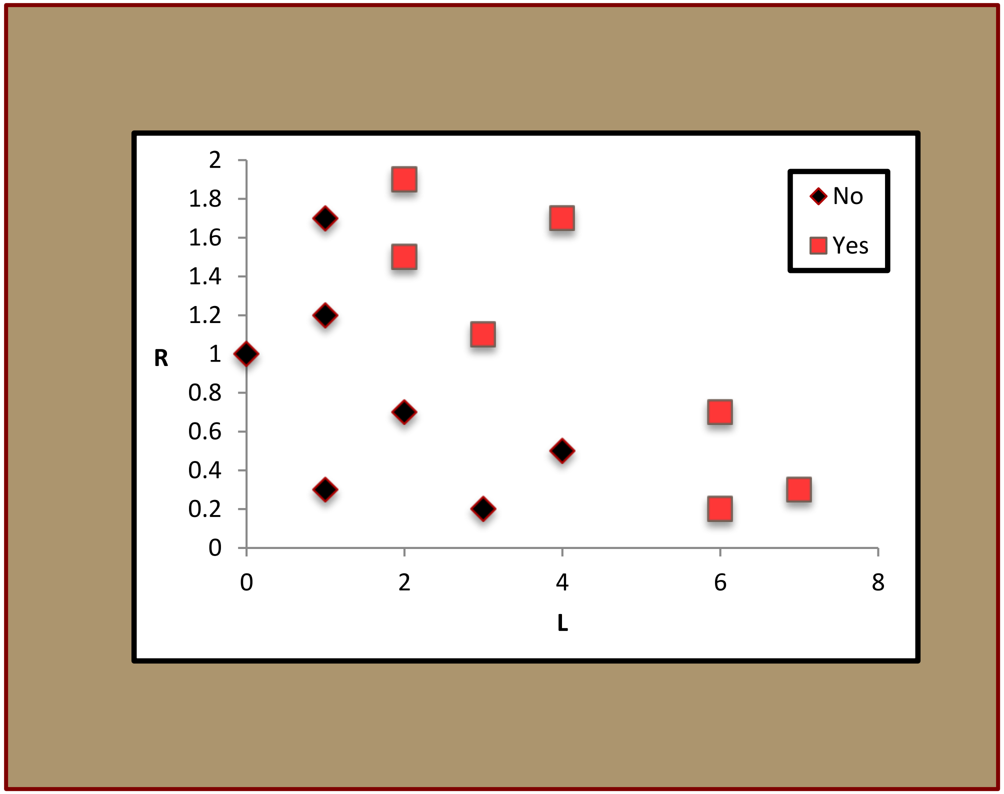
- ID3 algorithm
- Design issues
  - Split criteria
  - Stop criteria
  - Multi-valued attributes
  - Numeric attributes
  - Missing values
  - Overfitting
- Applications
- Limitations
- Real-life examples
- Extracting rules from trees

Numeric attributes: strategic goal

- Find numeric breakpoints which separate classes well
- Use the entropy of a split to evaluate each breakpoint

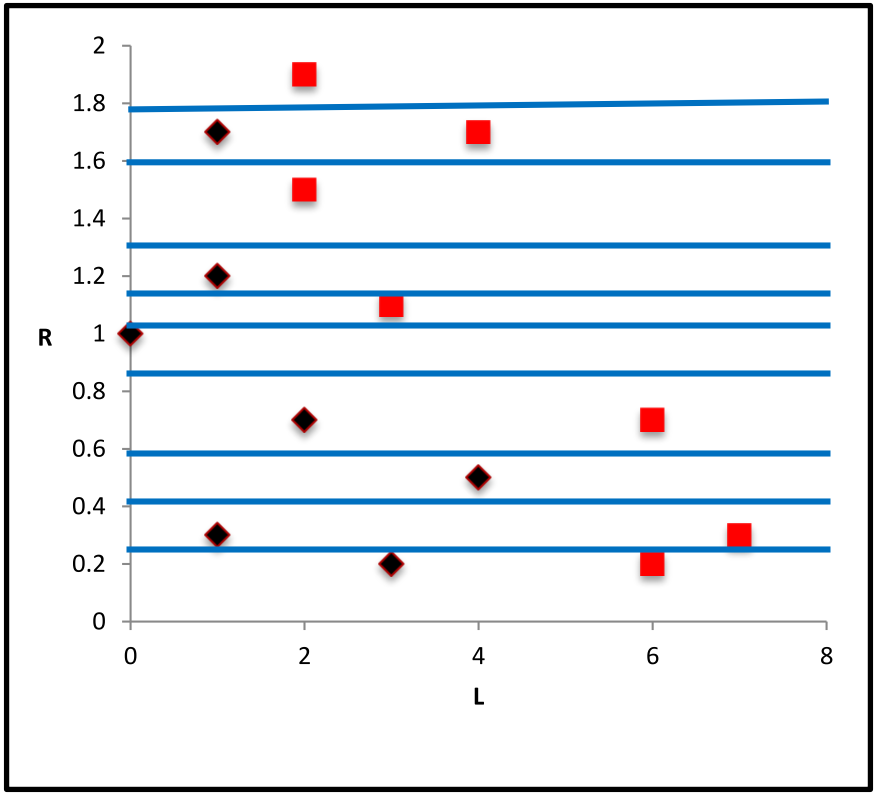
Bankruptcy example

# Late payments/ year (L)	Expenses/ income (R)	Bankruptcy (B)
3	0.2	No
1	0.3	No
4	0.5	No
2	0.7	No
0	1.0	No
1	1.2	No
1	1.7	No
6	0.2	Yes
7	0.3	Yes
6	0.7	Yes
3	1.1	Yes
2	1.5	Yes
4	1.7	Yes
2	1.9	Yes



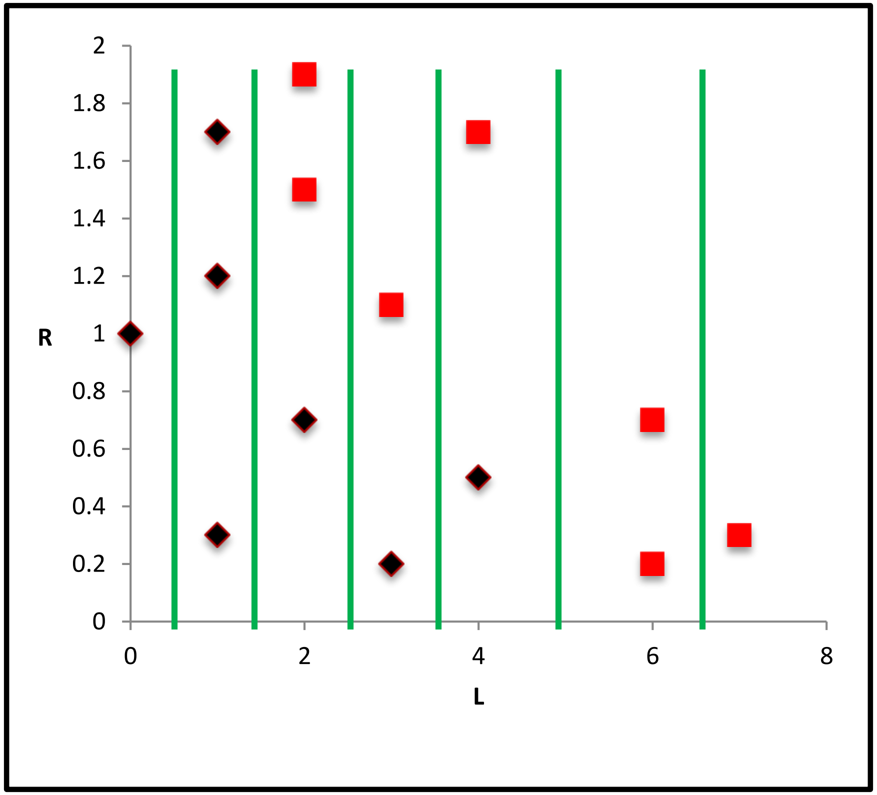
(Leslie Kaebling's example, MIT courseware)

Bankruptcy example



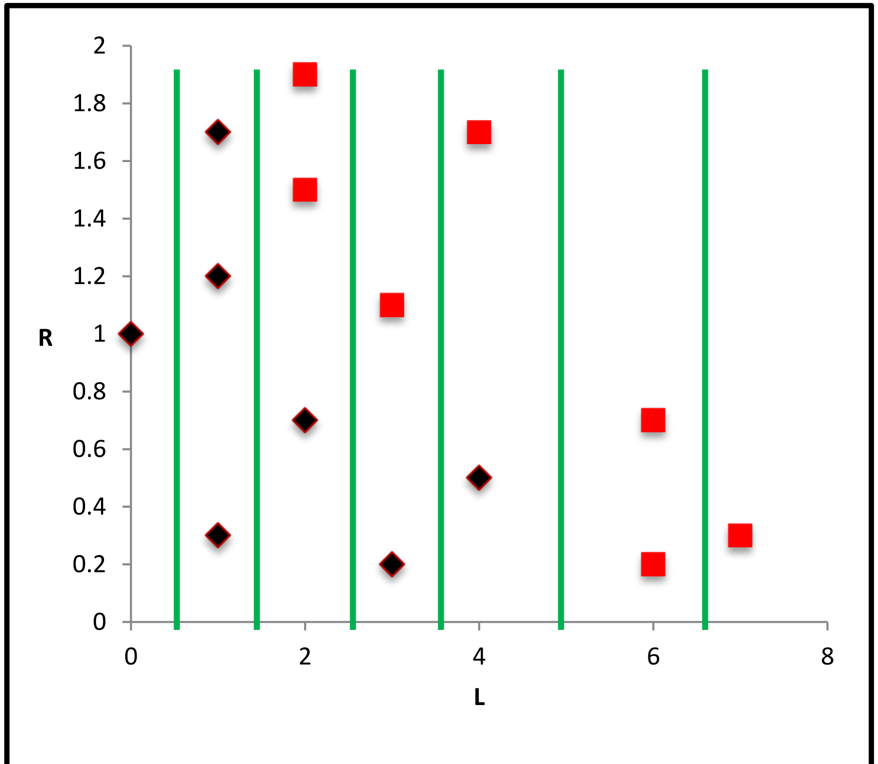
- Consider splitting (half-way) between each data point in each dimension.
- We have 9 different breakpoints in the R dimension

Bankruptcy example



- And there are another 6 possible breakpoints in the L dimension

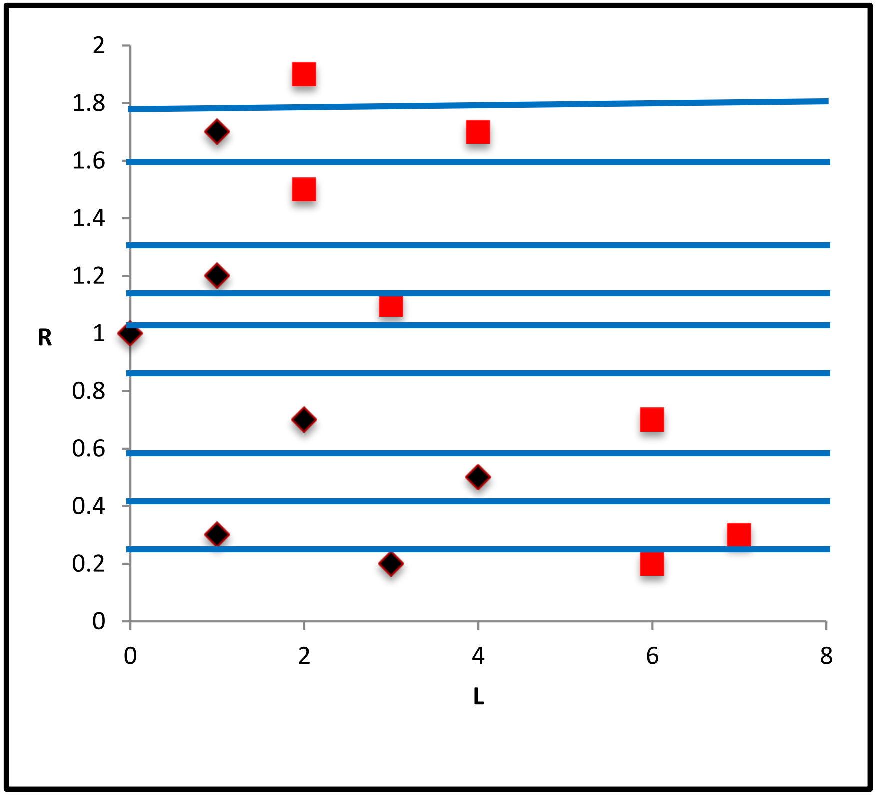
Evaluate entropy of a split on *L*



L<X	0.5	1.5	2.5	3.5	5.0	6.5
# Negative Left	1	4	5	6	7	7
# Positive Left	0	0	2	3	4	6
# Negative Right	6	3	2	1	0	0
# Positive Right	7	7	5	4	3	1
Entropy	0.93	0.63	0.86	0.85	0.74	0.92

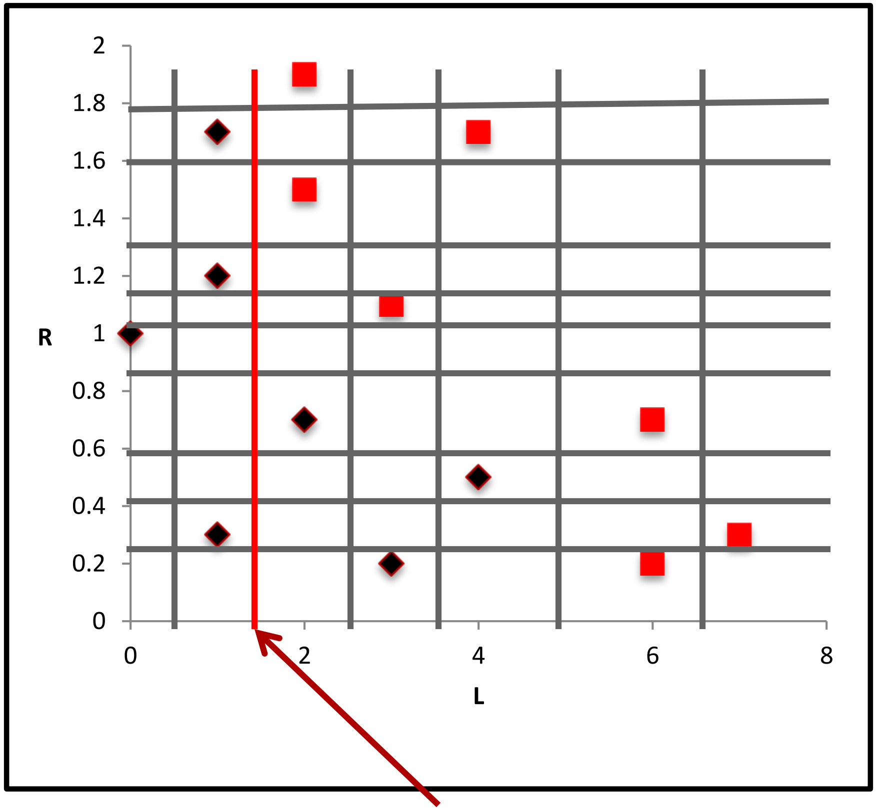
And on R

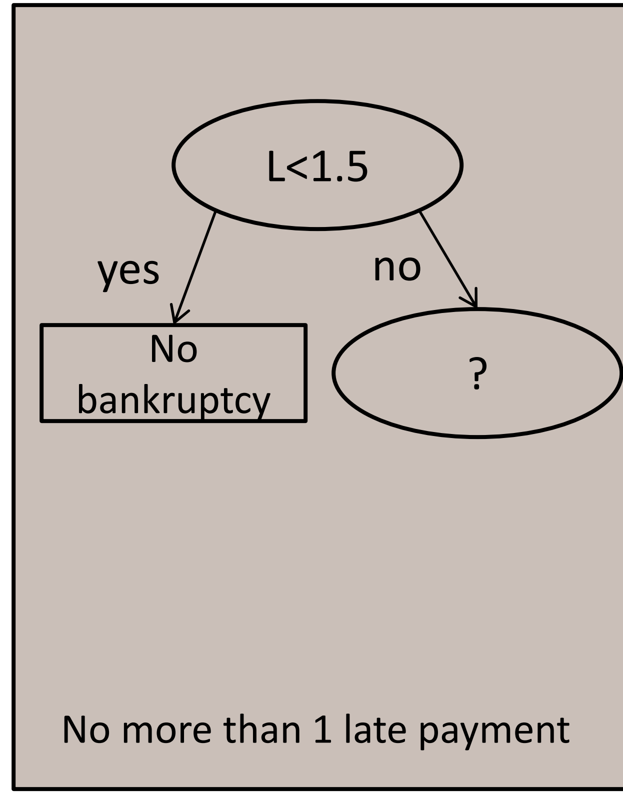
R<Y	Entropy
1.80	0.92
1.60	0.98
1.35	0.92
1.15	0.98
1.05	0.94
0.85	0.98
0.60	0.98
0.40	1.0
0.25	1.0



The best split point: min entropy

R<Y	Entropy
1.80	0.92
1.60	0.98
1.35	0.92
1.15	0.98
1.05	0.94
0.85	0.98
0.60	0.98
0.40	1.0
0.25	1.0



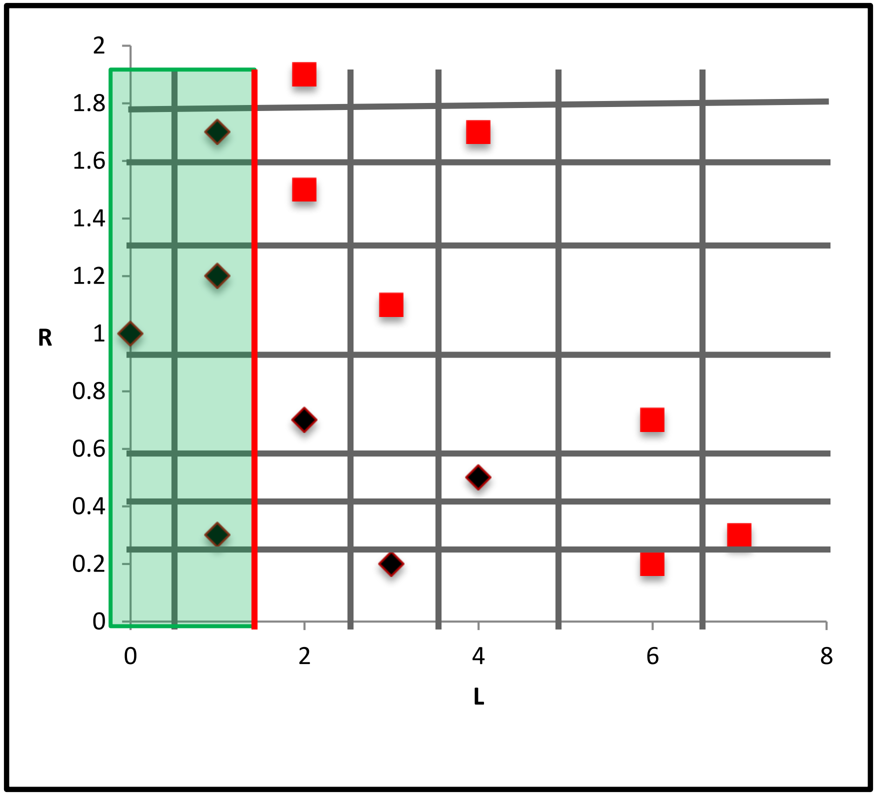


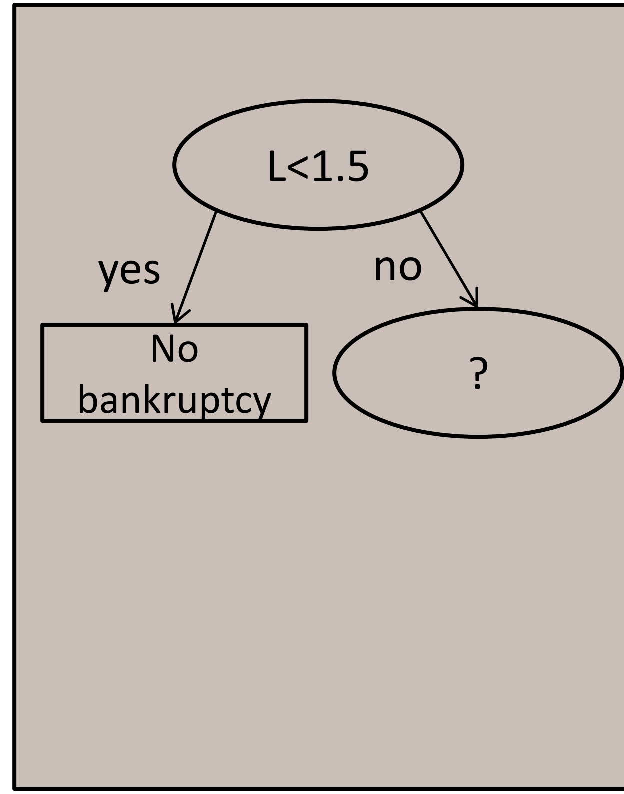
L<X	0.5	1.5	2.5	3.5	5.0	6.5
Entropy	0.93	0.63	0.86	0.85	0.74	0.92

- The best split: all the points with L not greater than 1.5 are of class 0, so we can make a leaf here.

Re-evaluate for the remaining points

R<Y	Entropy
1.80	0.92
1.60	0.98
1.30	0.92
0.90	0.60
0.60	0.79
0.40	0.88
0.25	0.85



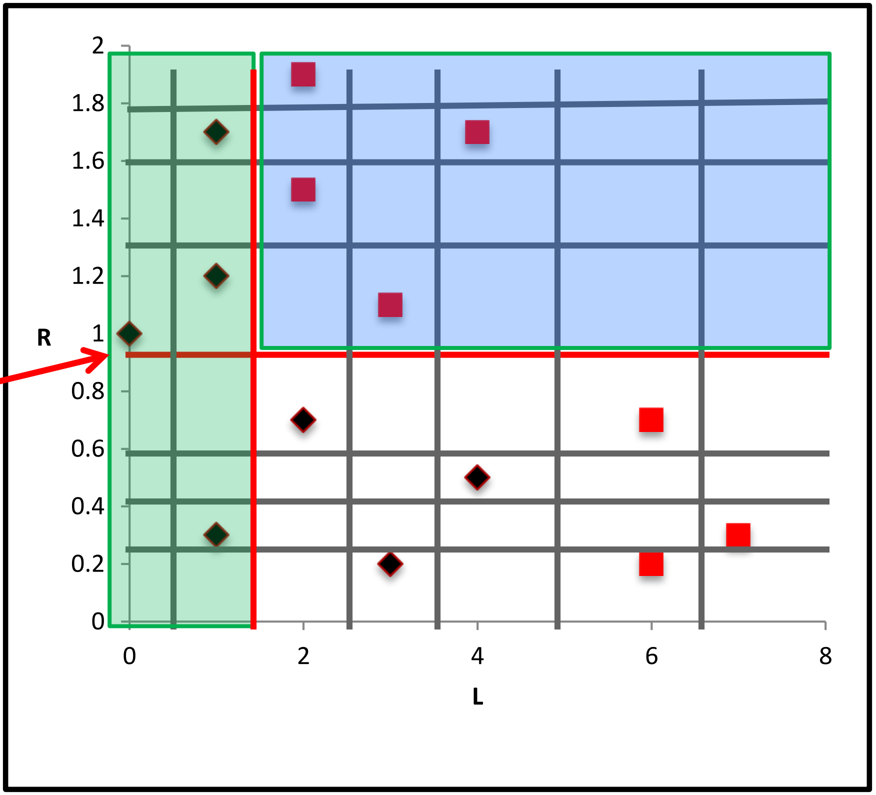


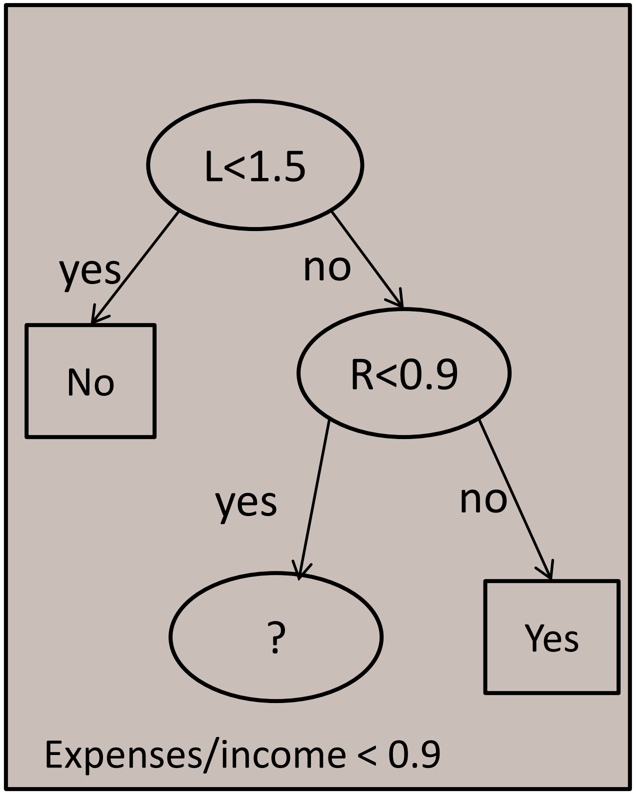
L<X	2.5	3.5	5.0	6.5
Entropy	0.88	0.85	0.69	0.83

- Consider only the remaining points. The entropy is recalculated, since the numbers have changed and the breakpoints moved (only 7 out of 9 for R)

The next best split

R<Y	Entropy
1.80	0.92
1.60	0.98
1.30	0.92
0.90	0.60
0.60	0.79
0.40	0.88
0.25	0.85

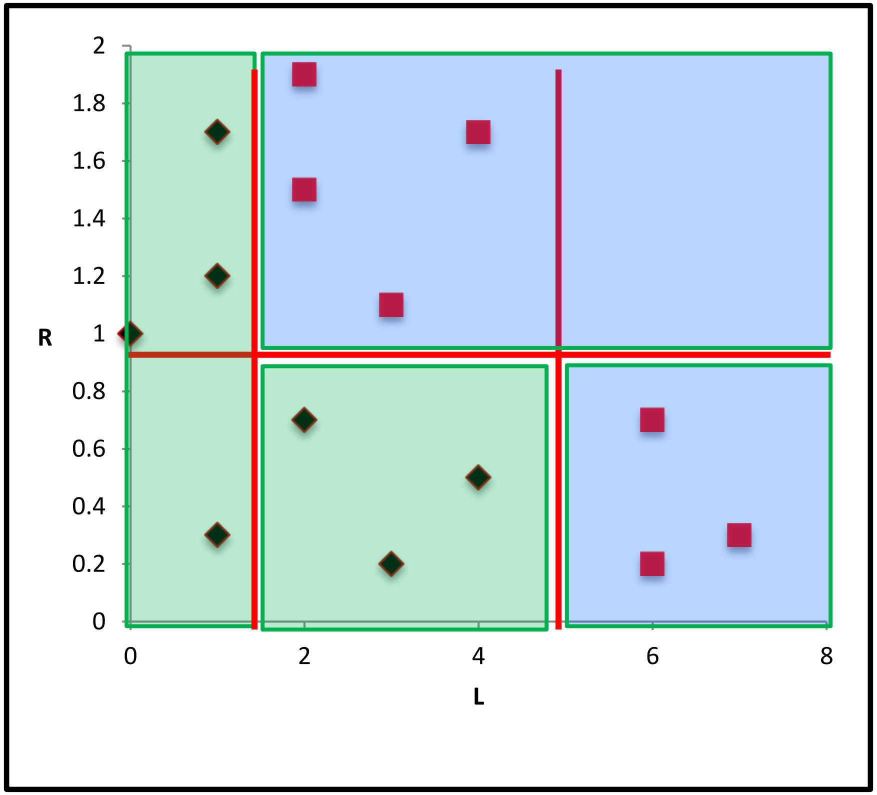


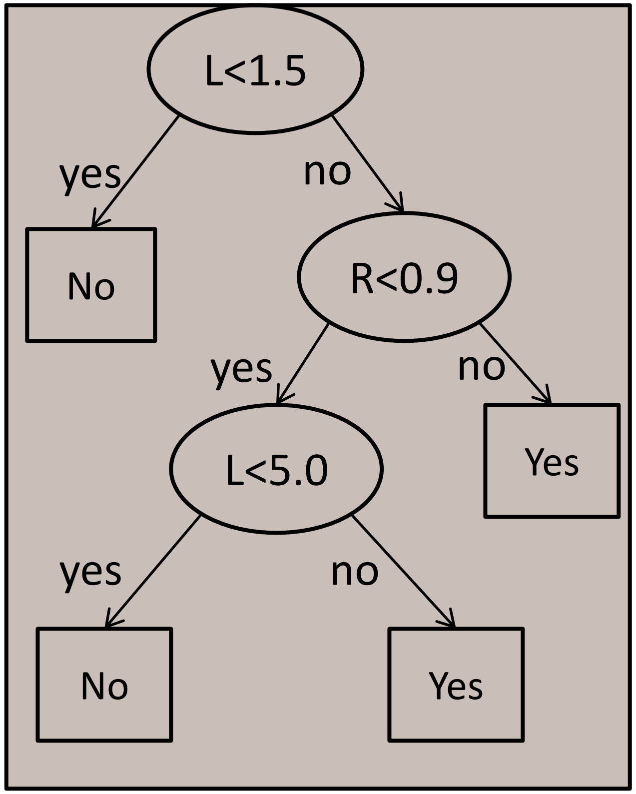


L<X	2.5	3.5	5.0	6.5
Entropy	0.88	0.85	0.69	0.83

- Split on R<0.9 and continue working with the remaining points

The final tree





Numeric target attribute: numeric class

- When the target attribute is numeric, the split should reduce the *variance* of the class values
- Variance – the deviation of the population values from the mean:

  the mean of the sums of the squared deviations from the mean:

  Variance=average [(x_i-mean (X))^2]

  for each numeric value x_i in set X

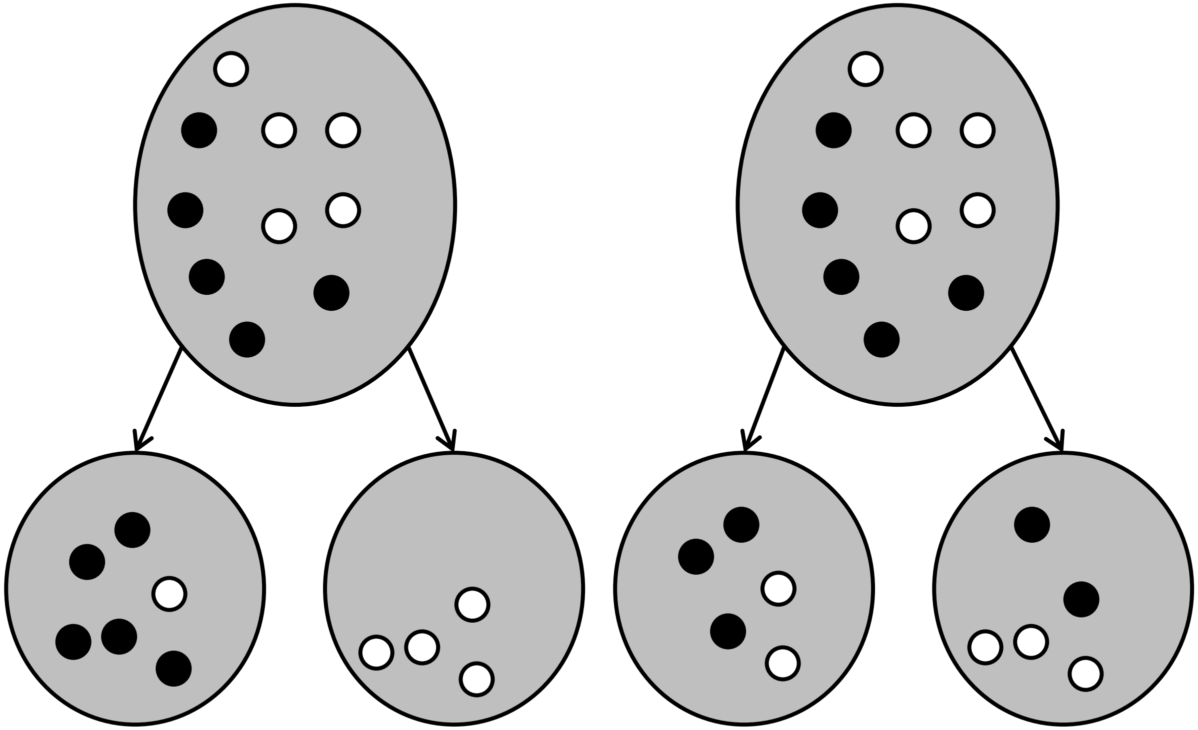
Actual formula for a sample population used in the examples (var In Excel):

$$\frac{\sum_{i=1}^{N}(x_i - \bar{x})^2}{N-1}$$

Illustration: simplified

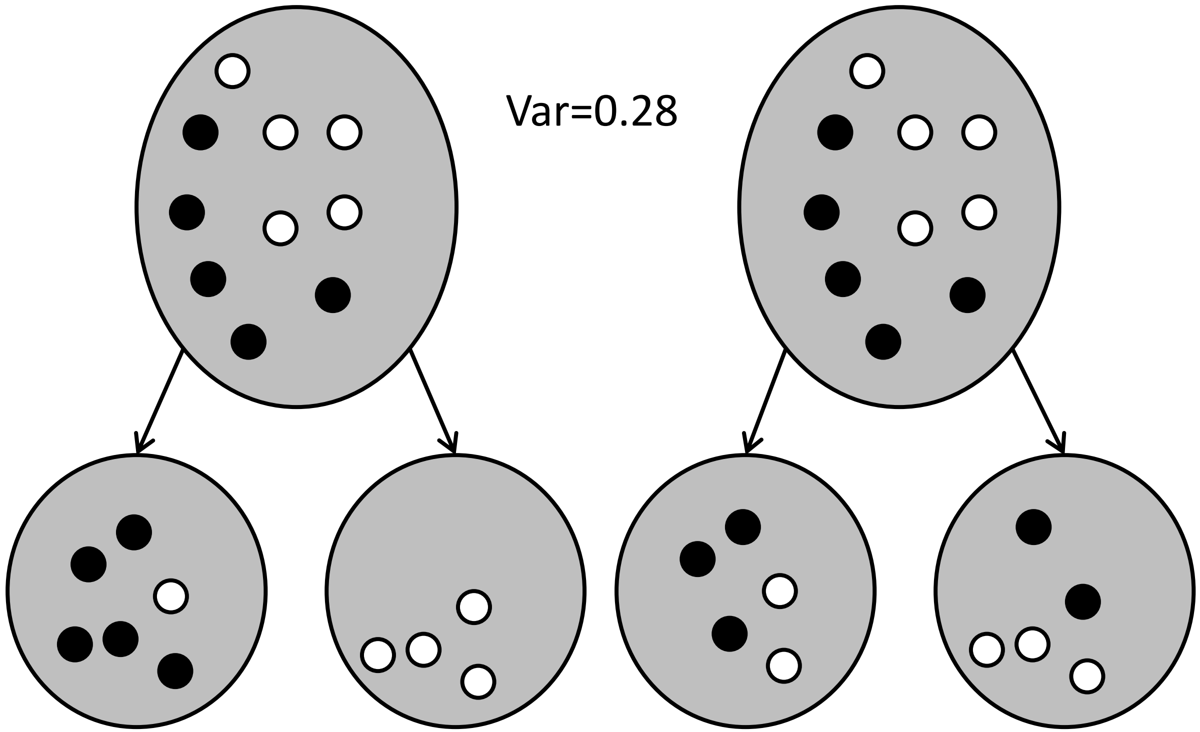
○ Represents value 0.0

● Represents value 1.0



- ID3 algorithm
- Design issues
  - Split criteria
  - Stop criteria
  - Multi-valued attributes
  - Numeric attributes
  - Missing values
  - Overfitting
- Applications
- Limitations
- Real-life examples
- Extracting rules from trees

Split based on variance



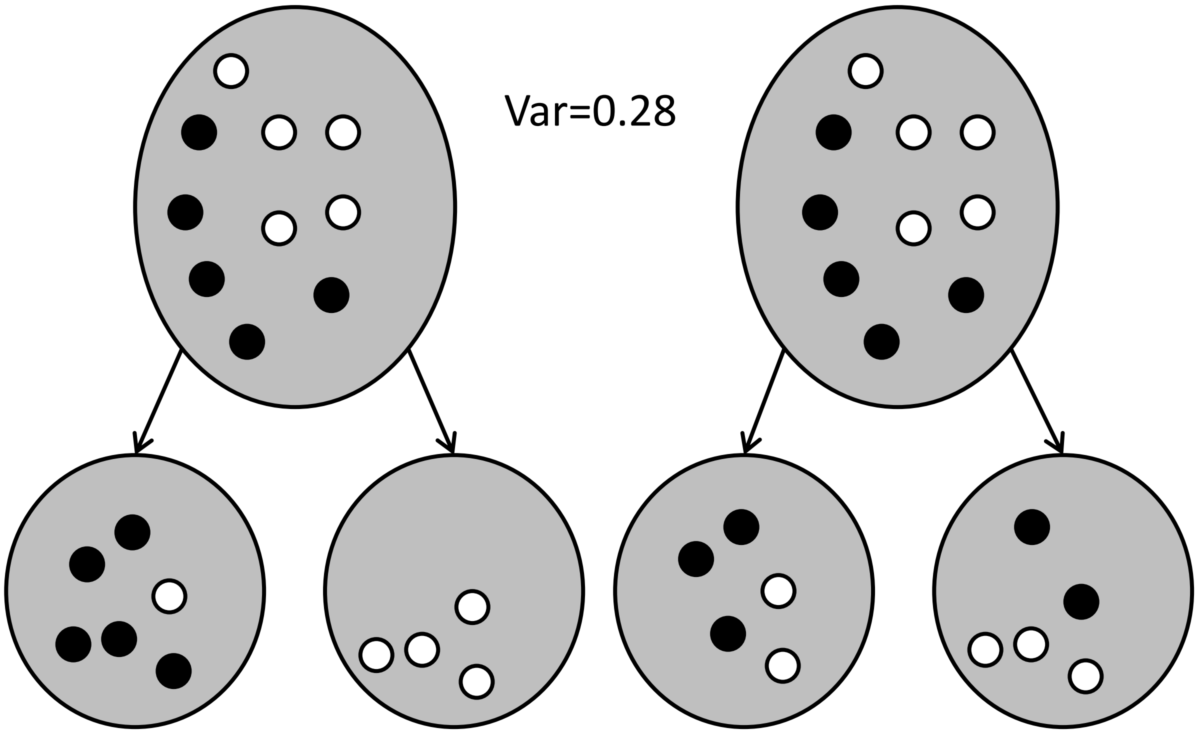
Mean=0.83
Variance=0.17

Mean=0.0
Variance=0.0

Variance of the split=6/10*0.17+4/10*0=0.10

- ID3 algorithm
- Design issues
  - Split criteria
  - Stop criteria
  - Multi-valued attributes
  - Numeric attributes
  - Missing values
  - Overfitting
- Applications
- Limitations
- Real-life examples
- Extracting rules from trees

Split based on variance



Mean=0.60
Variance=0.30

Mean=0.40
Variance=0.30

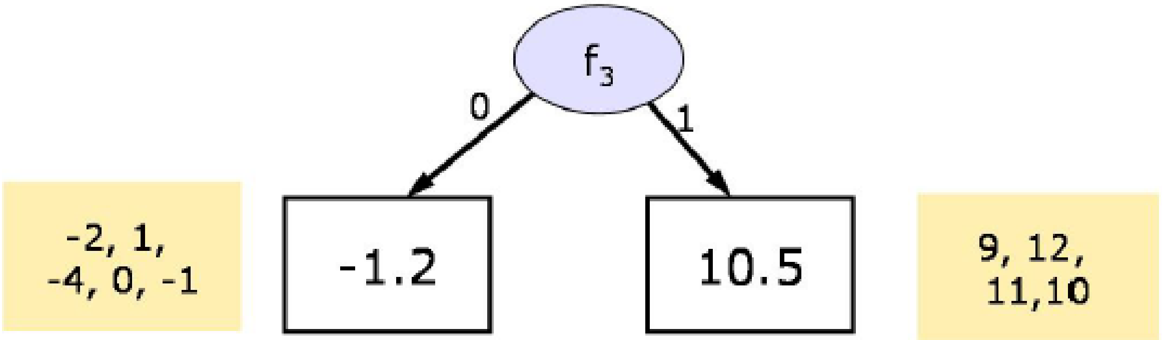
Variance of the split= $\boxed{0.10}$

Variance of the split=0.30

Choose the left split: variance reduction 0.18

- ID3 algorithm
- Design issues
  - Split criteria
  - Stop criteria
  - Multi-valued attributes
  - Numeric attributes
  - Missing values
  - Overfitting
- Applications
- Limitations
- Real-life examples
- Extracting rules from trees

Regression tree



- Stop when the variance at the leaf is small.
- Set the value at the leaf to be the mean of the class values

Missing values: possible causes

1. Malfunctioning measuring equipment
2. Changes in the experimental design
3. Survey - may refuse to answer certain questions (age or income)
4. Archeological skull may be damaged
5. Merging similar but not identical datasets

Missing values: possible solutions

1. Consider *null* to be a possible value with its own branch: “not reported”

   People who leave many traces in the customers database are more likely to be interested in the promotion offer than those whose lifestyle leaves most of the fields *null*

2. Impute missing value based on the value in records most similar to the current record

3. Follow all the branches of the tree with the weighted contribution

Missing values: both branches

A1	A2	A3	Class
1	0	1	yes
1	0	1	yes
0	1	1	yes
0	0	1	no
1	0	0	no

- To test the split on attribute A3:
  - If we know the value, we treat it with probability 1.0 (100%):

Info (instances (A3=1))=Entropy (3/4,1/4)

Info (instances (A3=0))=Entropy (0/1, 1/1)

Missing values: both branches

A1	A2	A3	Class
1	0		yes
1	0	1	yes
0	1	1	yes
0	0	1	no
1	0	0	no

- To test the split on attribute A3:
  - If the value is missing we estimate it based on the popularity of this value:

    it might be 1 with probability 0.75

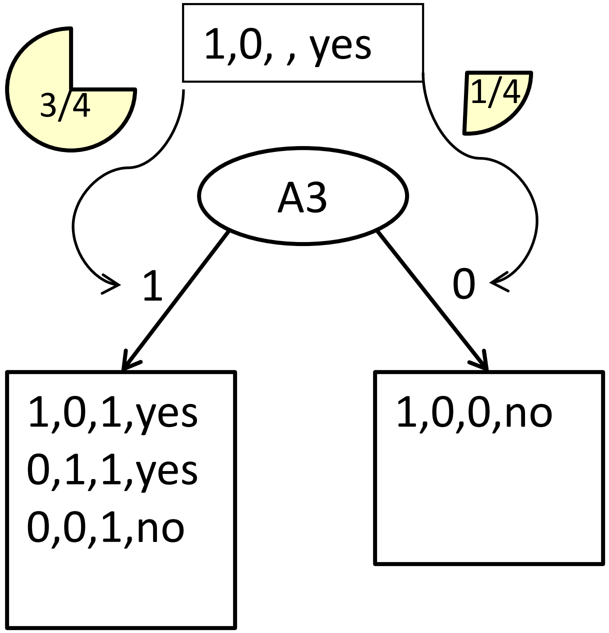
    it might be 0 with probability 0.25

    we count it in both branches:

Missing values: both branches

A1	A2	A3	Class
1	0		yes
1	0	1	yes
0	1	1	yes
0	0	1	no
1	0	0	no

Distribute between both branches

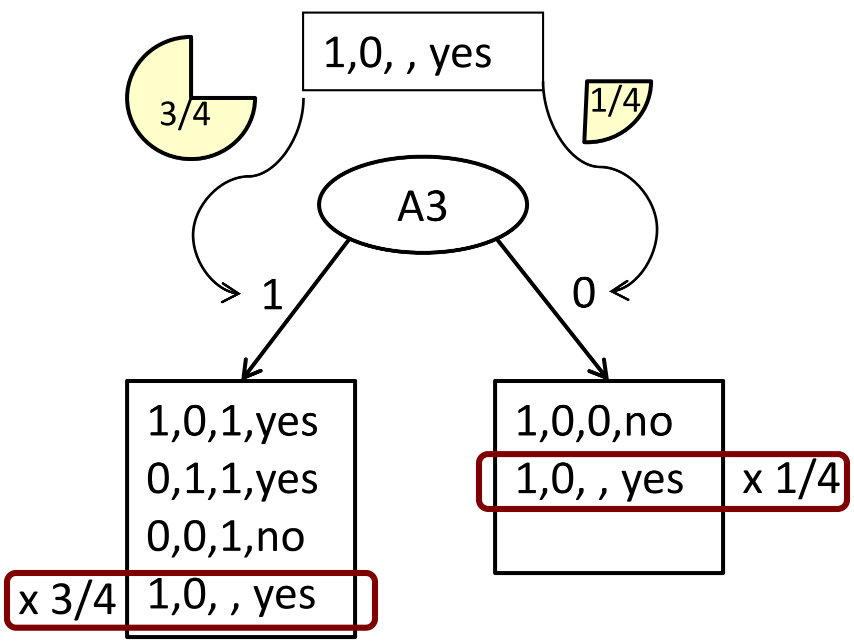


- ID3 algorithm
- Design issues
  - Split criteria
  - Stop criteria
  - Multi-valued attributes
  - Numeric attributes
  - Missing values
  - Overfitting
- Applications
- Limitations
- Real-life examples
- Extracting rules from trees

Missing values: both branches

A1	A2	A3	Class
1	0		yes
1	0	1	yes
0	1	1	yes
0	0	1	no
1	0	0	no

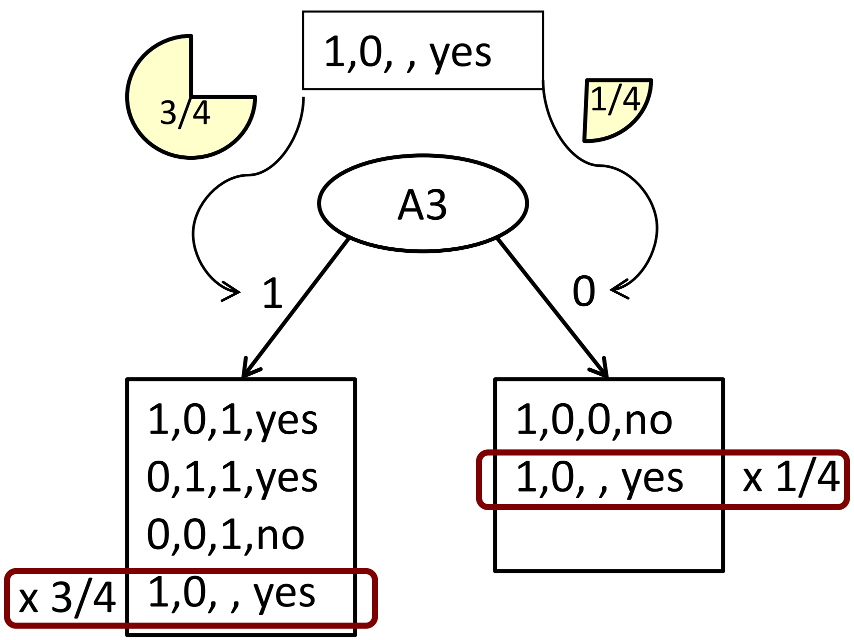
Distribute between both branches



- ID3 algorithm
- Design issues
  - Split criteria
  - Stop criteria
  - Multi-valued attributes
  - Numeric attributes
  - Missing values
  - Overfitting
- Applications
- Limitations
- Real-life examples
- Extracting rules from trees

Missing values: entropy update

A1	A2	A3	Class
1	0		yes
1	0	1	yes
0	1	1	yes
0	0	1	no
1	0	0	no



Info (instances (A3=1))= Entropy(2.75/3.75, 1.0/3.75)

Info (instnces (A3=0))= Entropy(0.25/1.25, 1.0/1.25)

- ID3 algorithm
- Design issues
  - Split criteria
  - Stop criteria
  - Multi-valued attributes
  - Numeric attributes
  - Missing values
  - Overfitting
- Applications
- Limitations
- Real-life examples
- Extracting rules from trees

Missing values: compare

A1	A2	A3	Class
1	0	1	yes
1	0	1	yes
0	1	1	yes
0	0	1	no
1	0	0	no

Info (instances (A3=1))=Entropy (3/4,1/4)

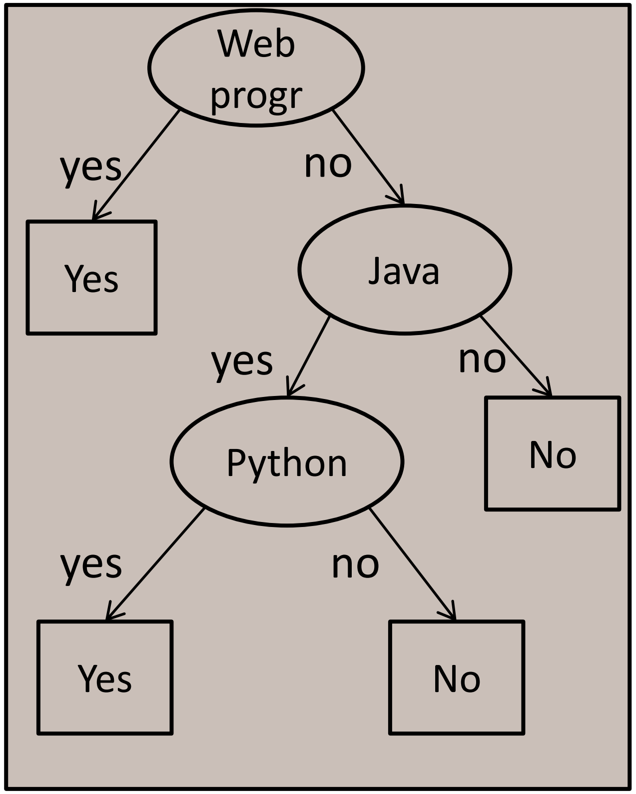
Info (instances (A3=0))=Entropy (0/1, 1/1)

A1	A2	A3	Class
1	0		yes
1	0	1	yes
0	1	1	yes
0	0	1	no
1	0	0	no

Info (instances (A3=1))= Entropy(2.75/3.75, 1.0/3.75)

Info (instances (A3=0))= Entropy(0.25/1.25, 1.0/1.25)

Error rate in training and validation sets



In a validation set: If *N* records arrive at a leaf, and *E* of them are classified incorrectly, then the error rate at that node is *E*/*N*.

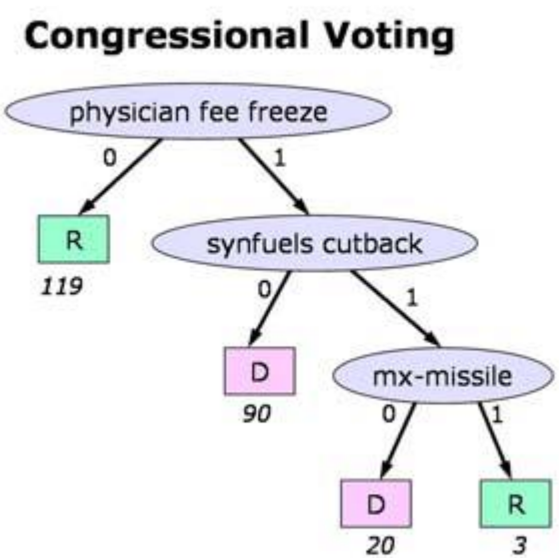
Class label:

interested in building web ML apps?

- Error rate of the training set (built on 4 instances): 0
- Error rate on validation set: ?

- ID3 algorithm
- Design issues
  - Split criteria
  - Stop criteria
  - Multi-valued attributes
  - Numeric attributes
  - Missing values
  - Overfitting
- Applications
- Limitations
- Real-life examples
- Extracting rules from trees

Overfitting: too confident prediction



- Attempt to fit all the training data. When the number of records in each splitting subset is small, the probability of splitting on noise grows
- The tree is making predictions that are more confident that what can be really deduced from the data

Handling overfitting: main strategies

- *Post-pruning* - take a fully-grown decision tree and discard unreliable parts
- *Pre-pruning* - stop growing a branch when information becomes unreliable

Post-pruning preferred in practice—pre-pruning can "stop too early"

Pre-pruning

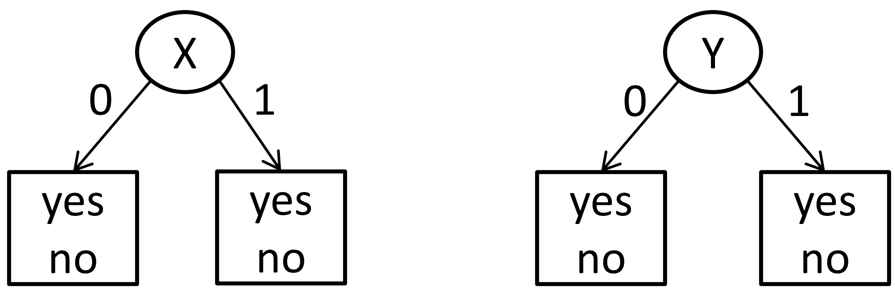
- Stop splitting when the number of instances is below the threshold (< 30)
- Stop splitting when information gain is below the threshold

  -Dangerous: the algorithm is based on the local optimization: there is no information gain in the current split, but may be a big gain at the next level!

Pre-pruning gone bad: example

- The *exclusive-or* (XOR) problem

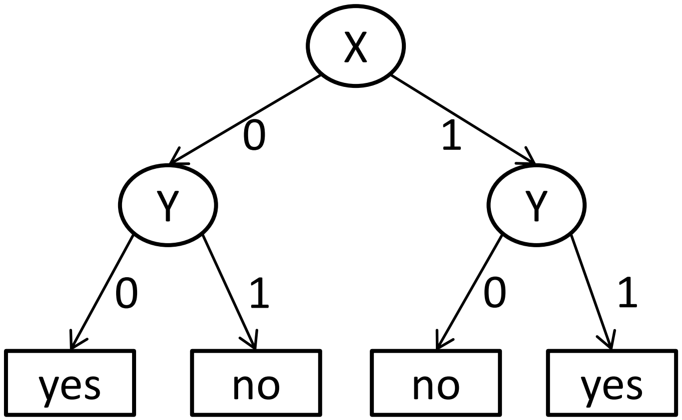
X	Y	Class
0	0	yes
0	1	no
1	0	no
1	1	yes



There is no information gain: the entropy is 1.0 for the root and for the both splits – so we must stop here

Pre-pruning gone bad: example

X	Y	Class
0	0	yes
0	1	no
1	0	no
1	1	yes



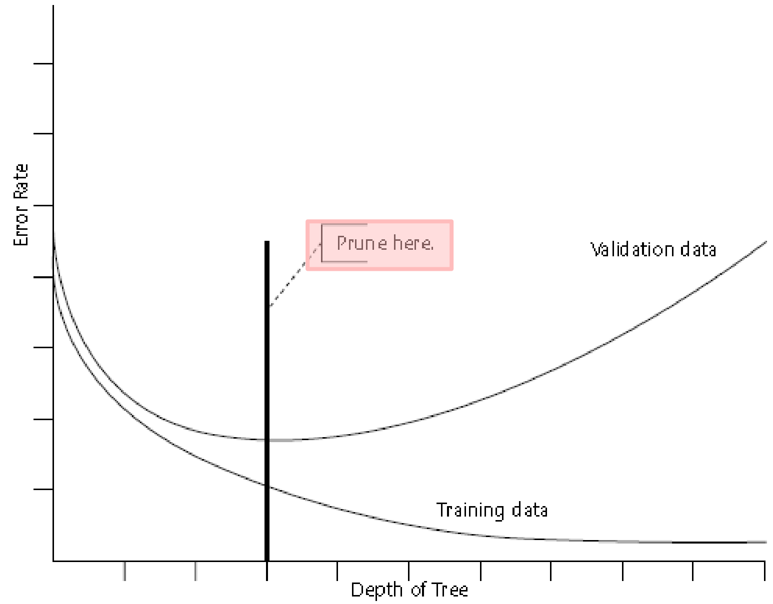
But the subsequent split produces completely pure nodes!

Structure is only visible in fully expanded tree

Post-pruning strategies

1. Use hold-out validation set.

If the validation error rate exceeds the statistically defined threshold, prune the sub-tree and replace it by the majority class



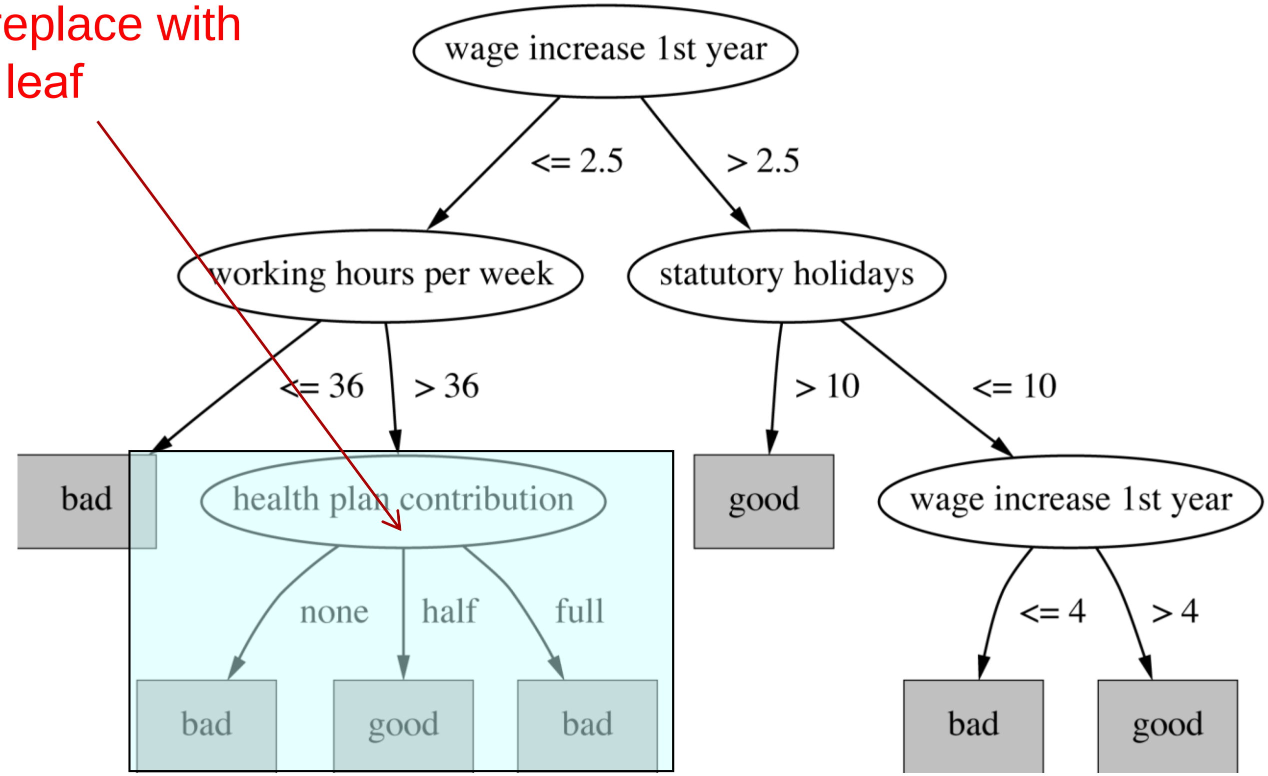
Post-pruning strategies

2. Consider the number of instances in the node for computing its error rate (the smaller the number, the greater the error rate).

If error rate of children is greater than that of the parent, the branches are pruned and replaced by the majority class.

Sub-tree pruning – bottom up

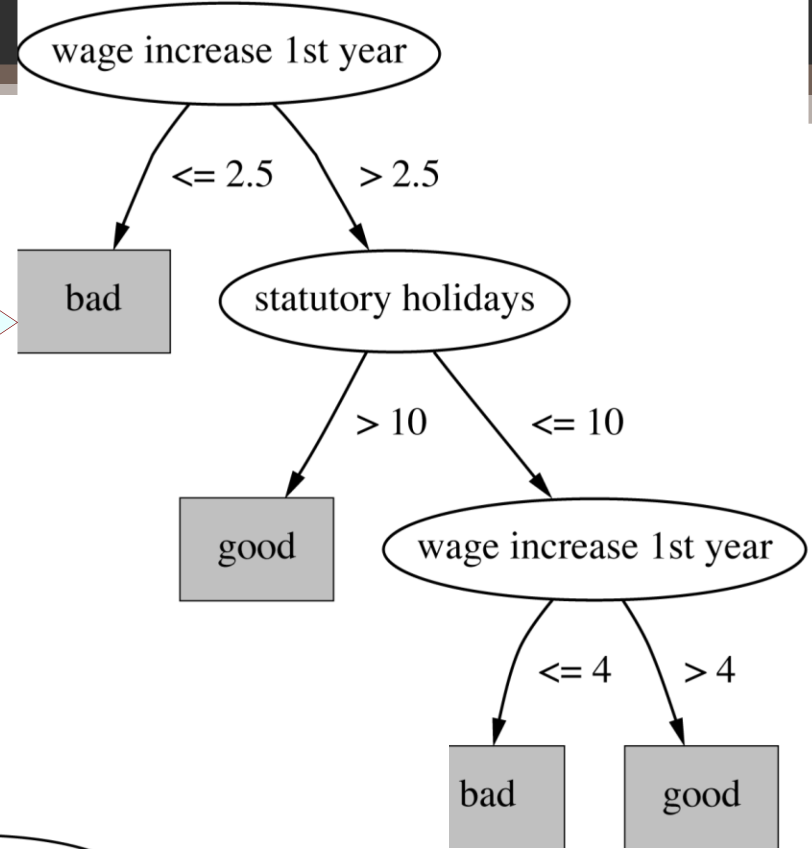
Large error rate on validation set: collapse the node and replace with 'bad' leaf

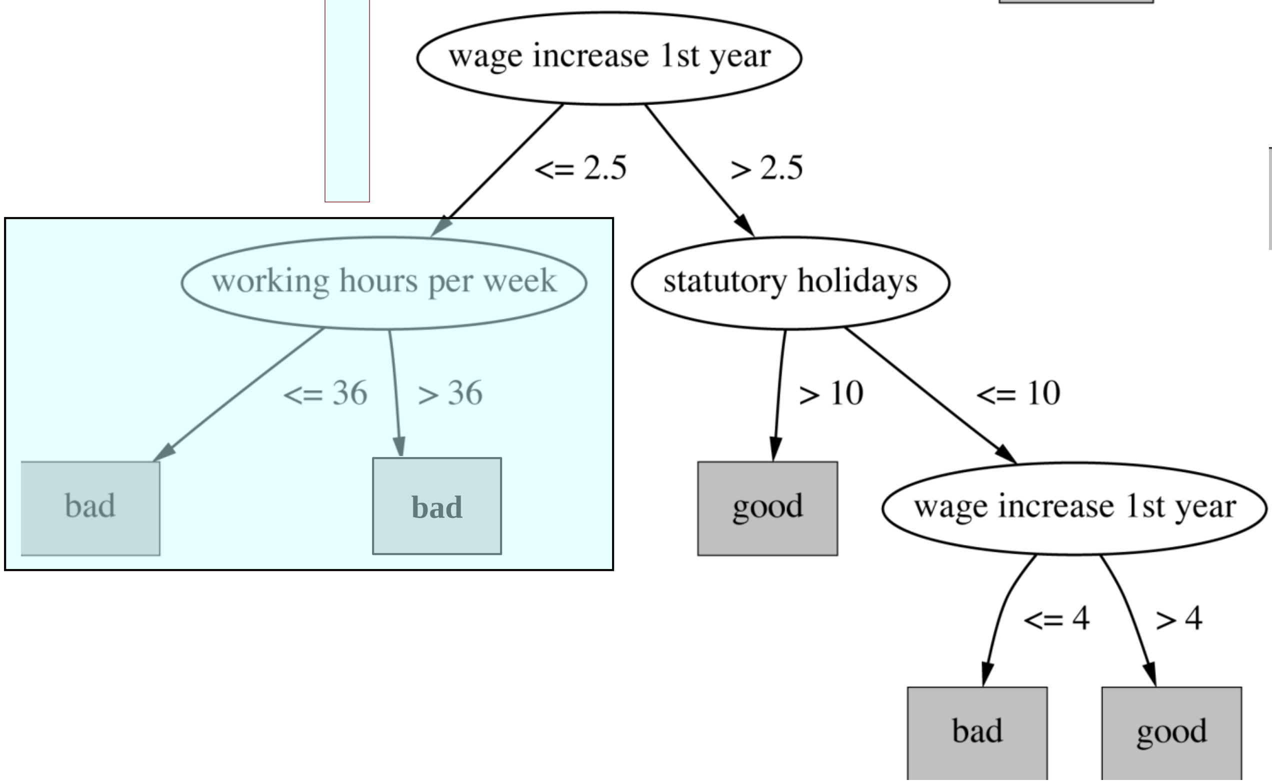


witten & eibe

Sub-tree pruning – bottom up

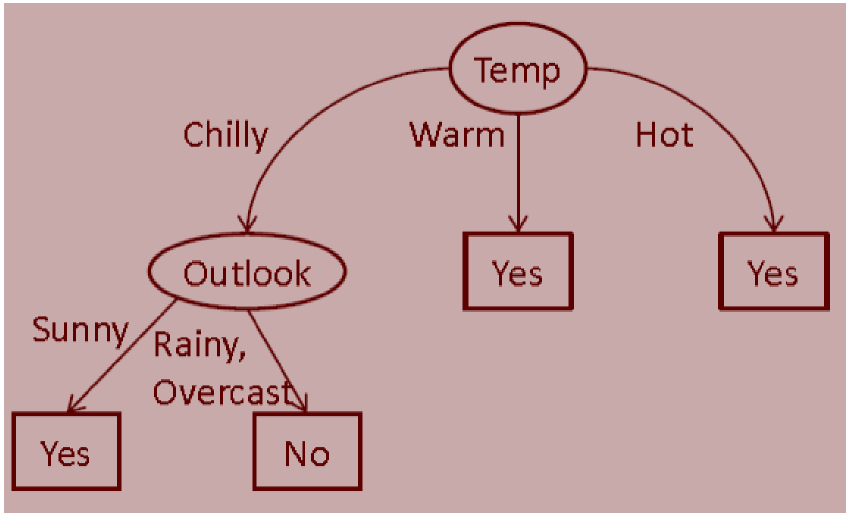
There is no more split on ‘working hours per week’ – collapse the sub-tree





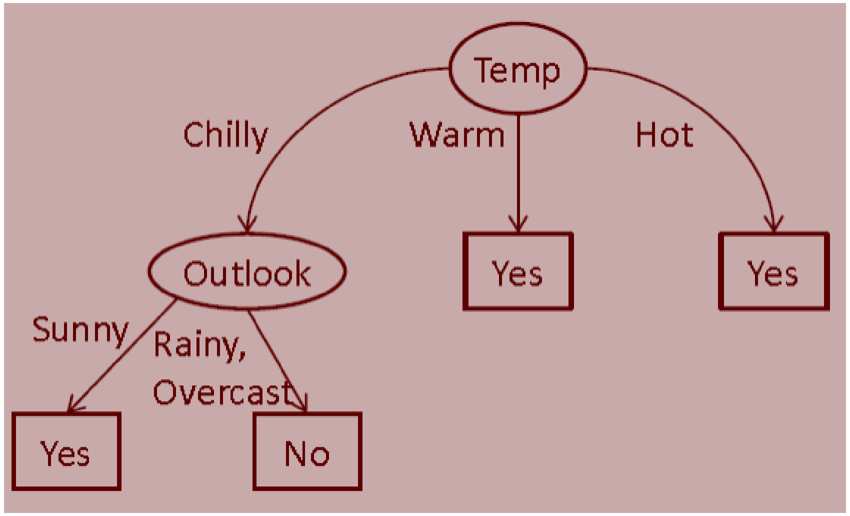
Decision trees for classification

- Classify and make transparent decision
- Each class leaf has its own rule path
- The same result by different reasons



Decision trees for data exploration

- The most important attributes are at the top of the tree
- Start each data mining project from exploring the most important attributes with decision trees



When (not) to use decision trees

Good performance (use decision trees)

- The factors of decision are not less important than the classification accuracy
- The goal is to assign each record to one of a few broad categories (Categorical attributes with low cardinality*)
- **You suspect that there is a set of objective rules underlying the data**

Not that good performance (use something else)

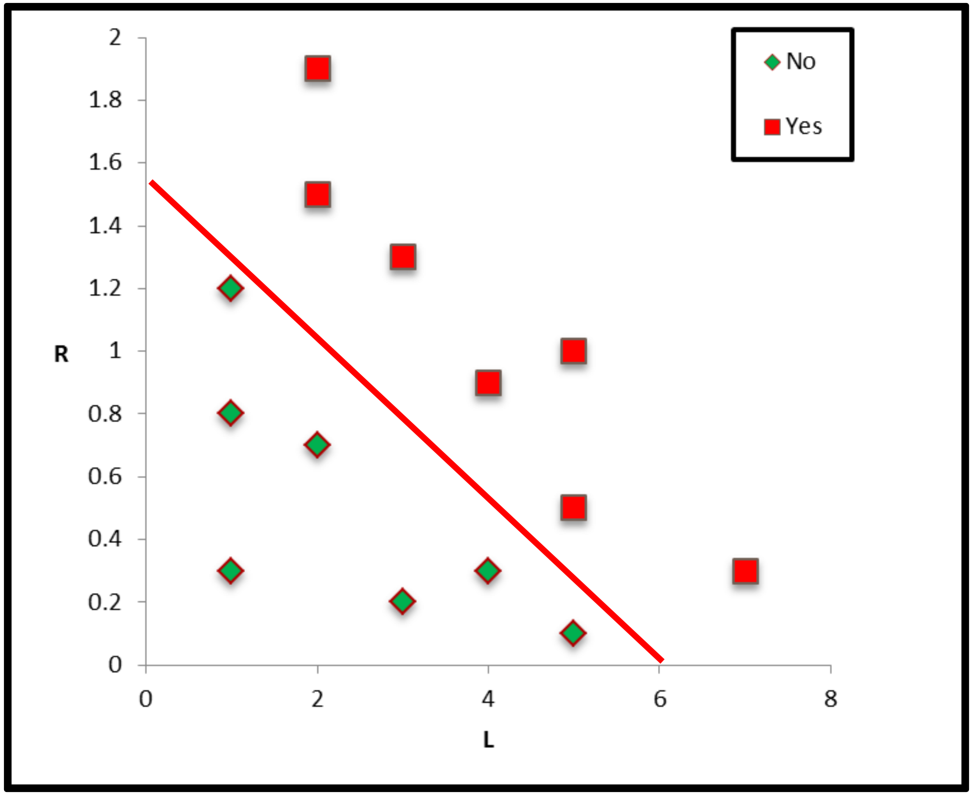
- Continuous numeric attributes, ordinal attributes
- Hierarchical relationships between classes
- High-cardinality attributes
- Numeric value prediction

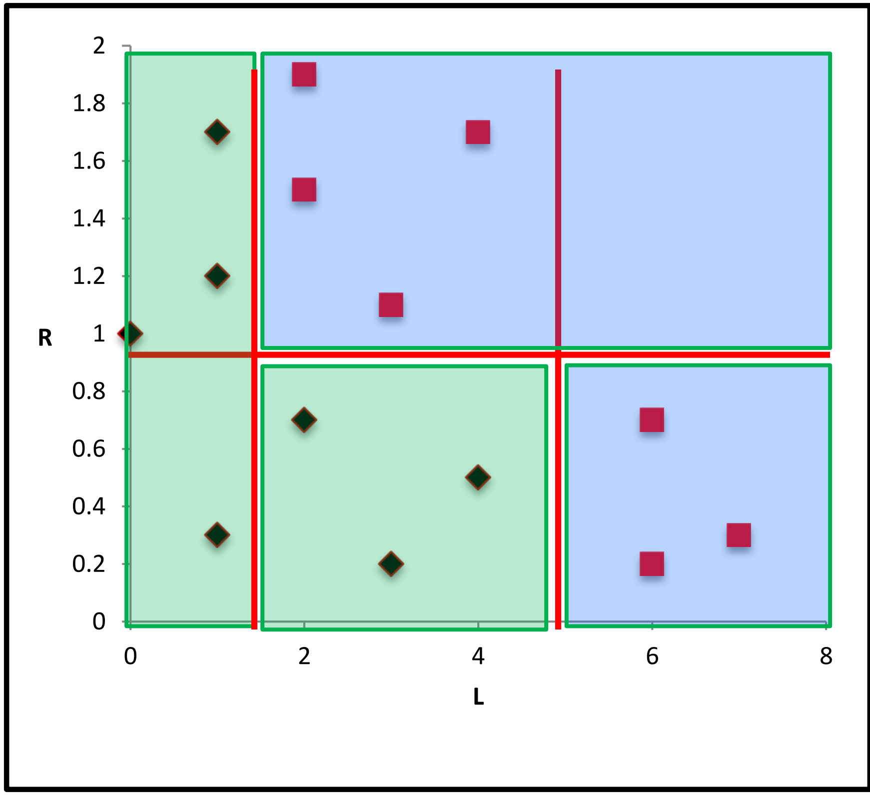
---

*cardinality - the number of possible distinct values

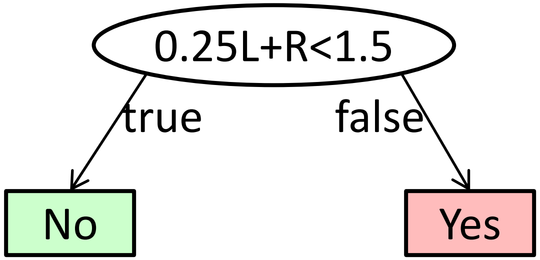
Limitations. Rectilinear decision boundaries

- Boolean split: the instances are divided by the boundaries which are parallel to the axes
- Solution: use all reasonable combinations of attributes.

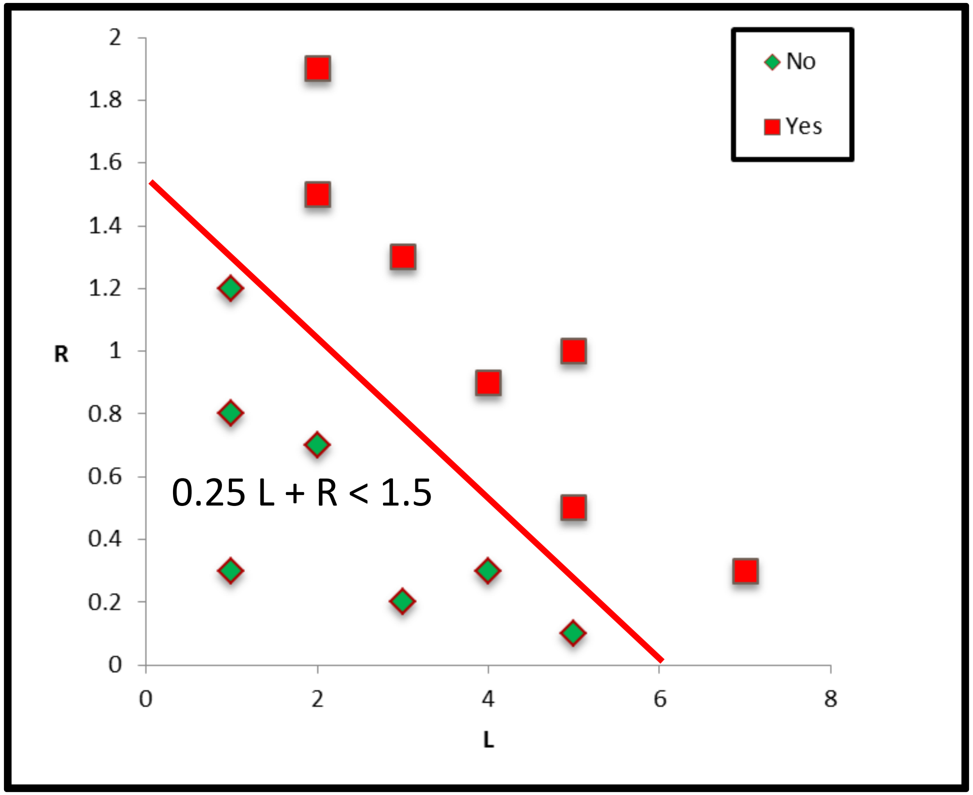


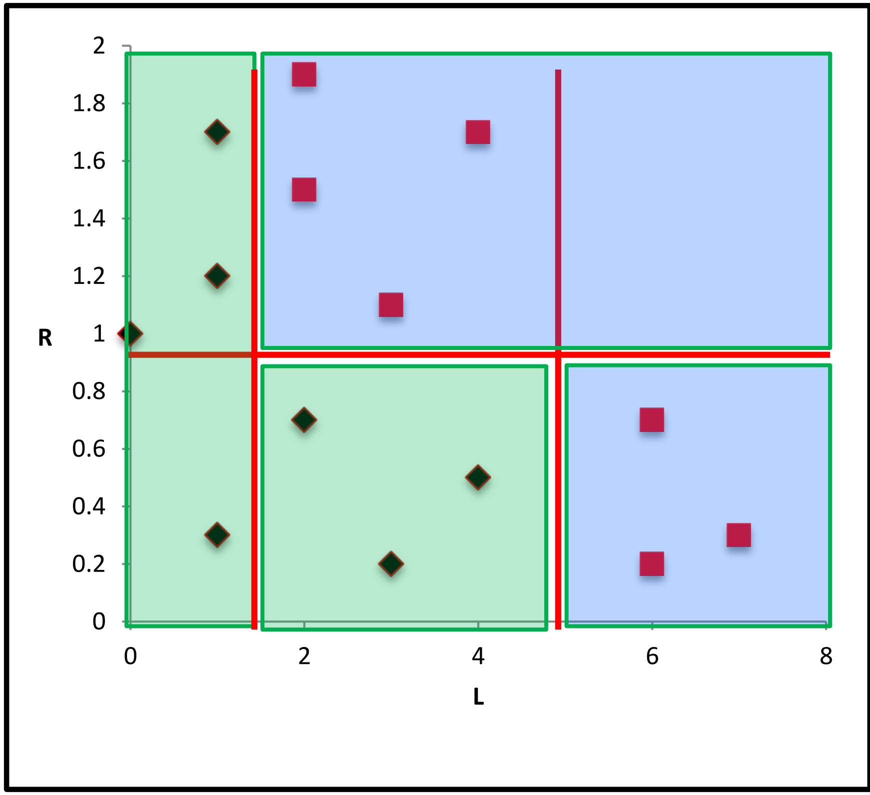


Non-rectilinear boundaries: attribute combinations



One-level decision tree





Decision trees in real life

- Selecting the most promising eggs for in-vitro fertilization – England, 2000
- Soybean disease classification – 1979, 97% accuracy vs. 72% by human expert
- Classification system for serial criminal patterns (CSSCP) - using three years' worth of data on armed robbery, the system was able to spot 10 times as many patterns as a team of experienced detectives with access to the same data.
- Screening potential terrorists and drug smugglers at border crossings

Border crossing: gross oversimplification

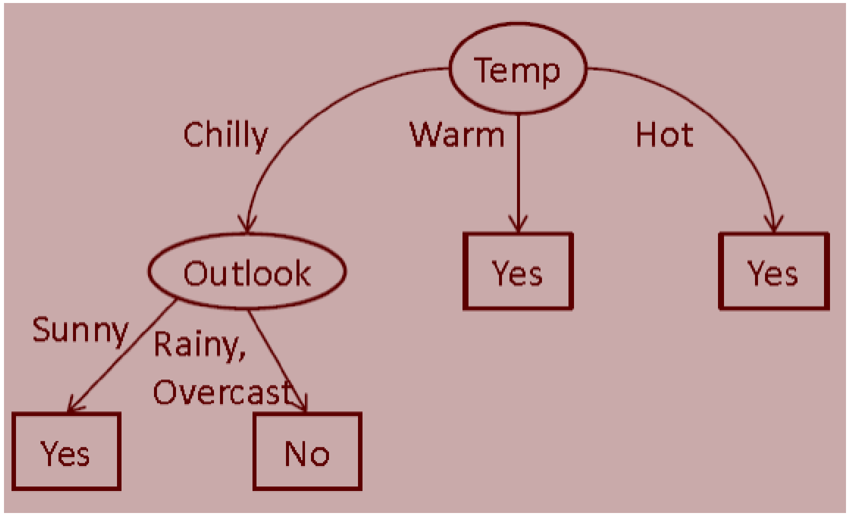
- Age: 20-25
- Gender: male
- Nationality: Saudi Arabia
- Country of residence: Germany
- Visa status: student
- University: unknown
- # times entering the country in the past year: 3
- Countries visited during the past 3 years: U.K., Pakistan
- Flying lessons: yes

Assessment: possible terrorist (probability 29%)
Action: detain and report

Carnival Booth: An Algorithm for Defeating the Computer-Assisted Passenger Screening System

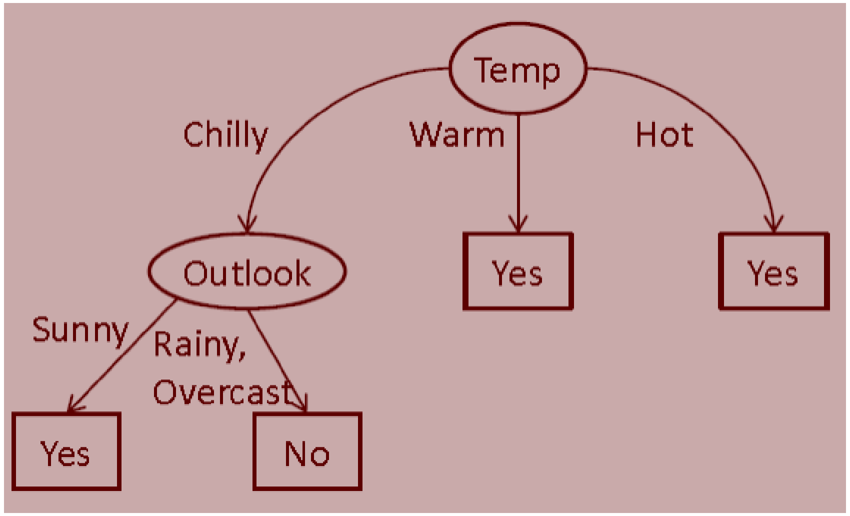
From trees to rules: how?

- How can we produce a set of rules from a decision tree?



From trees to rules – simple

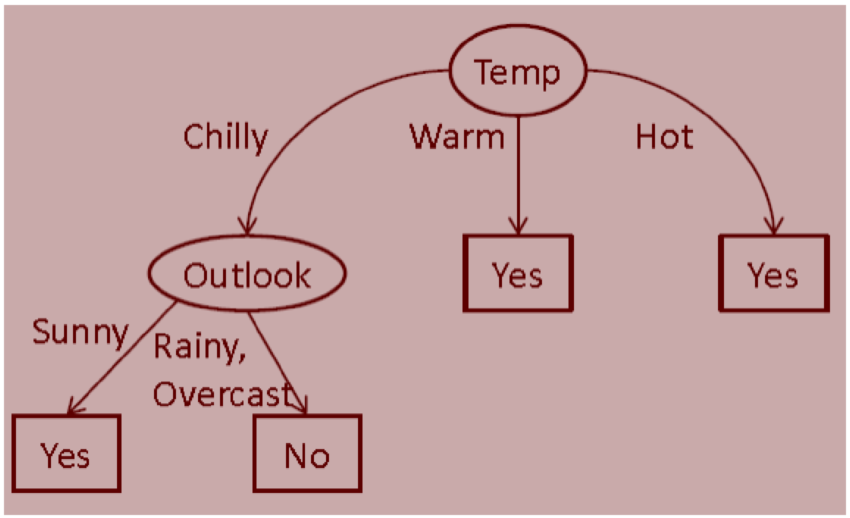
- One rule for each leaf



If Temp = “Warm” **then** play
If Temp = “Hot” **then** play
If Temp = “Chilly” and Outlook=“Sunny” **then** play
Default: no play

From trees to rules – simple

- The set of rules can be minimized



If Temp = “Chilly” and (Outlook=“Rainy” or Outlook = “Overcast”)
then no play
Default: play

Difference between decision trees and rules

- Rules are more readable than decision trees
- Decision trees describe the **general concept** extracted from the data, while each rule represents **a nugget of knowledge**
- Trees contain predictions for **all class variables**, while each rule predicts only **one class value**