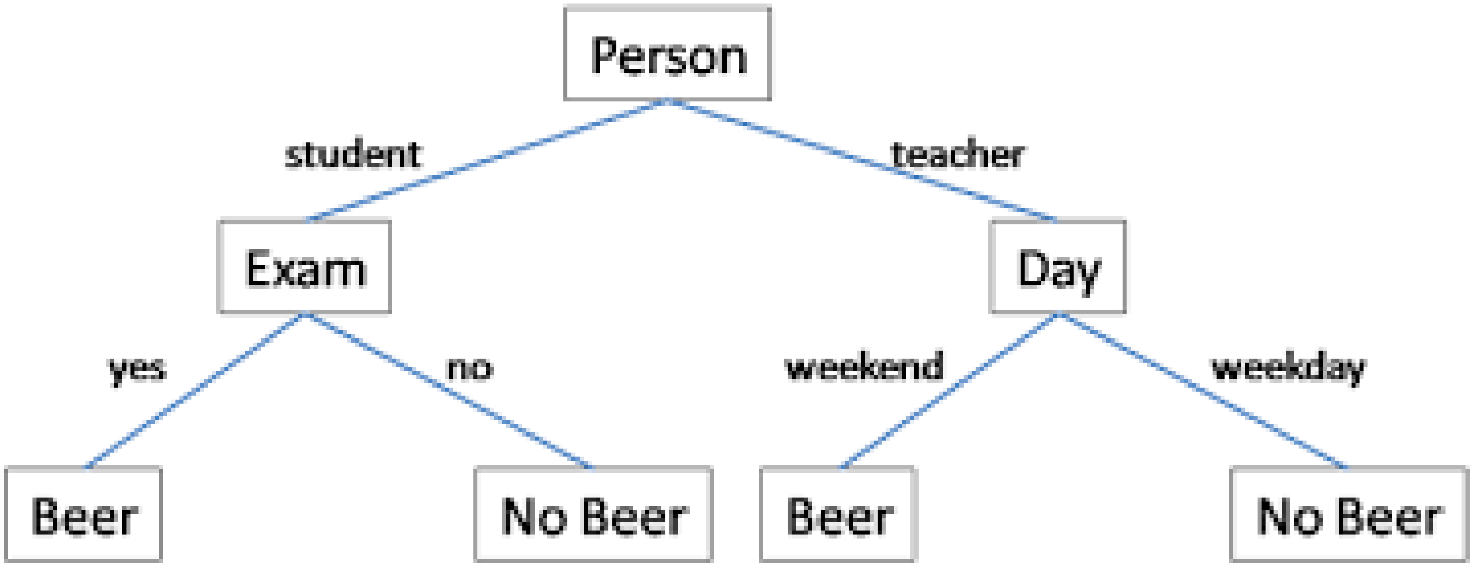
Mobile Cloud Computing

-

Decision Trees



Prof. Dr. Ch. Reich

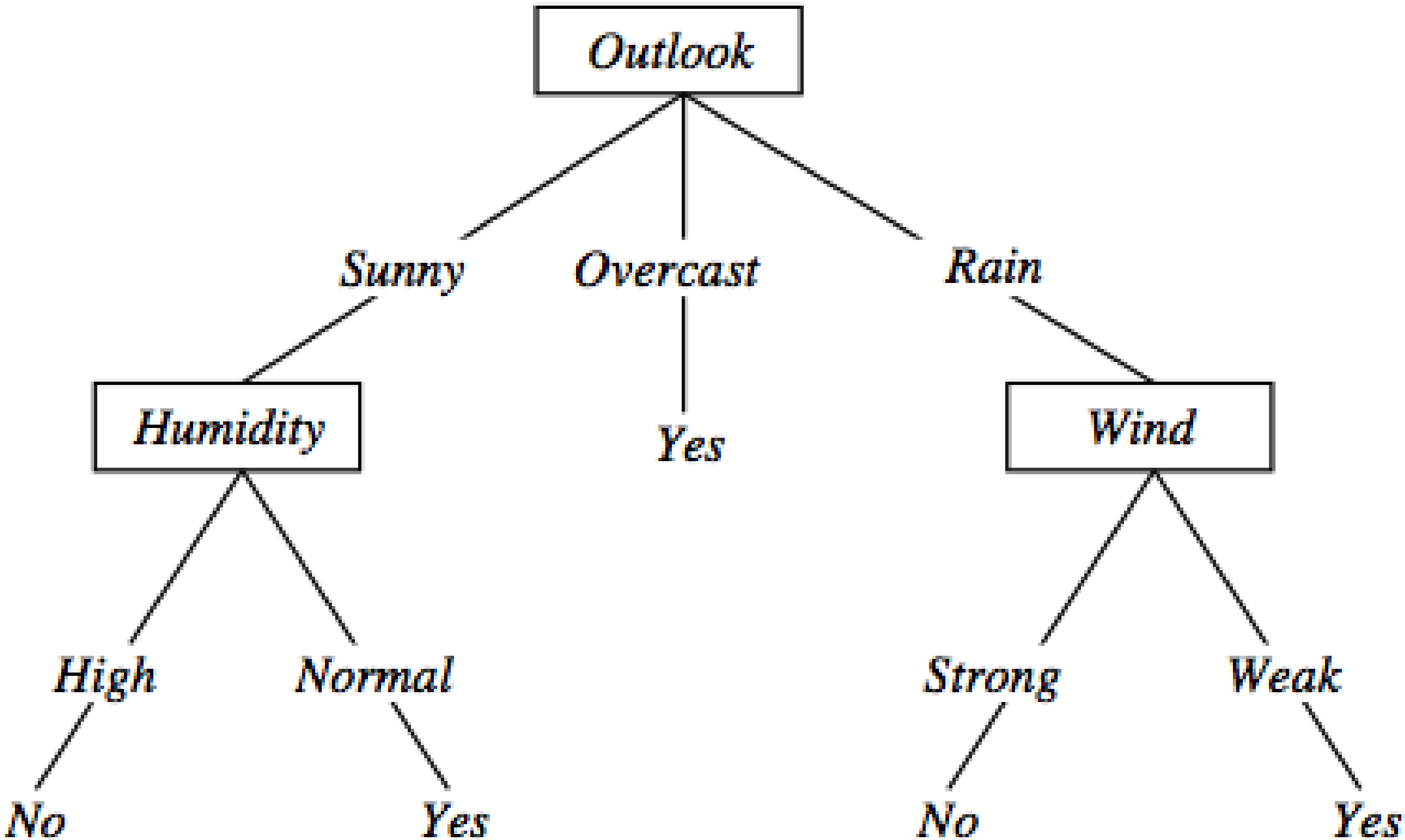
rch@hs-furtwangen.de

http://wolke.hs-furtwangen.de

What is a Decision Tree?

- An inductive learning task
  - Use particular facts to make more generalized conclusions
- Decision Trees is one of the most widely used and practical methods of inductive inference
- Features
  - Method for approximating discrete-valued functions (including boolean)
  - Learned functions are represented as decision trees (or if-then-else rules)
  - Expressive hypotheses space, including disjunction
  - Robust to noisy data

Decision tree representation (PlayTennis)

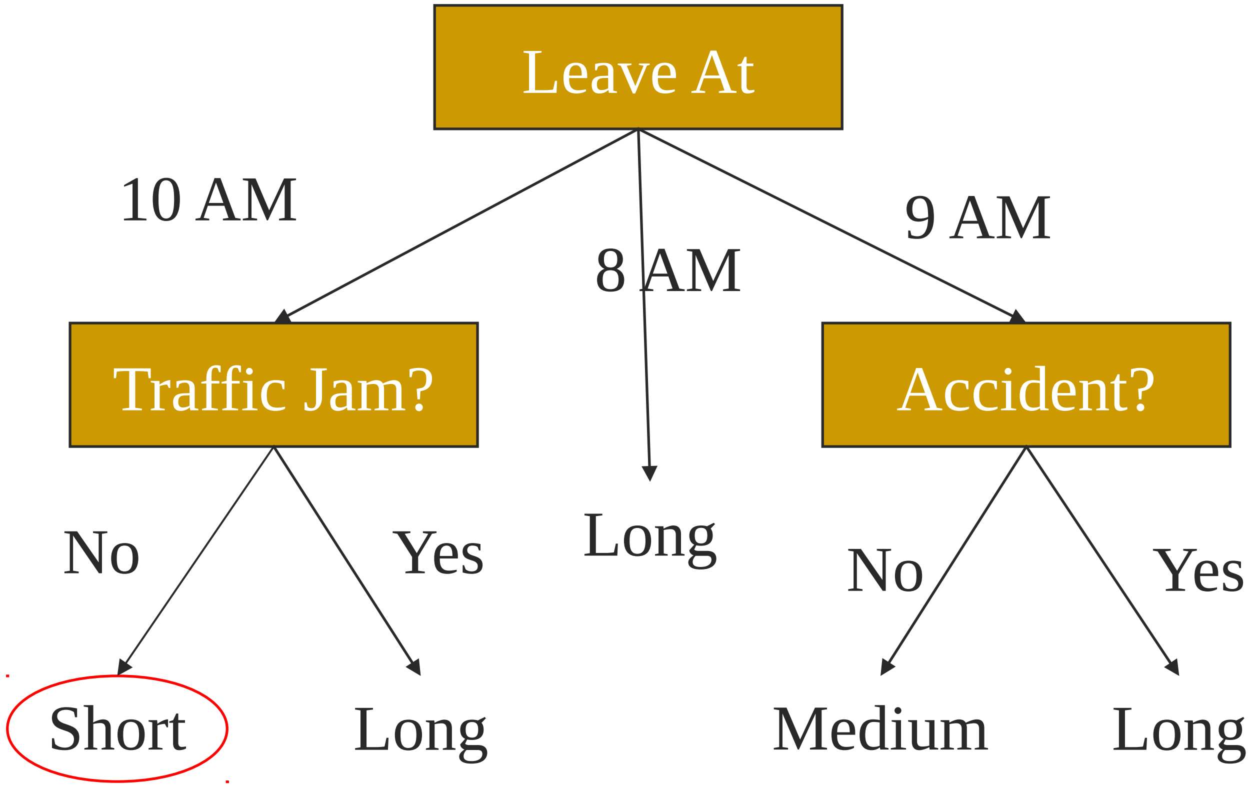


<Outlook=Sunny, Temp=Hot, Humidity=High, Wind=Strong> No

Decision trees expressivity

- Decision trees represent a disjunction (OR) of conjunctions (AND) on constraints on the value of attributes:
  - If ((Outlook = Sunny) AND (Humidity=Normal)) then play_tennis
  - If (Outlook = Overcast) then play_tennis
  - If ((Outlook = Rain) AND (Wind=Weak)) then play_tennis

Predicting Commute Time



If we leave at 10 AM and there are no cars stalled on the road, what will our commute time be?

Inductive Learning

- In this decision tree, we made a series of Boolean decisions and followed the corresponding branch
  - Did we leave at 10 AM?
  - Did a car stay because of traffic jam on the road?
  - Is there an accident on the road?

- By answering each of these yes/no questions, we then came to a conclusion on how long our commute might take

Decision Trees as Rules

- We did not have represent this tree graphically
- We could have represented as a set of rules. However, this may be much harder to read…

Decision Tree as a Rule Set

```
if hour == 8am
    commute time = long
else if hour == 9am
    if accident == yes
        commute time = long
    else
        commute time = medium
else if hour == 10am
    if trafficJam == yes
        commute time = long
    else
        commute time = short
```

- Notice that all attributes to not have to be used in each path of the decision.
- As we will see, all attributes may not even appear in the tree.

How to Create a Decision Tree

- We first make a list of attributes that we can measure
  - These attributes (for now) must be discrete
- We then choose a *target attribute* that we want to predict
- Then create an *experience table* that lists what we have seen in the past

Sample Experience Table

Example	Attributes				Target
	Hour	Weather	Accident	Traffic Jam	Commute
D1	8 AM	Sunny	No	No	Long
D2	8 AM	Cloudy	No	Yes	Long
D3	10 AM	Sunny	No	No	Short
D4	9 AM	Rainy	Yes	No	Long
D5	9 AM	Sunny	Yes	Yes	Long
D6	10 AM	Sunny	No	No	Short
D7	10 AM	Cloudy	No	No	Short
D8	9 AM	Rainy	No	No	Medium
D9	9 AM	Sunny	Yes	No	Long
D10	10 AM	Cloudy	Yes	Yes	Long
D11	10 AM	Rainy	No	No	Short
D12	8 AM	Cloudy	Yes	No	Long
D13	9 AM	Sunny	No	No	Medium

Choosing Attributes

- The previous experience decision table showed 4 attributes: hour, **weather**, accident and traffic jam
- But the decision tree only showed 3 attributes: hour, accident and traffic jam
- Why is that?

Choosing Attributes

- Methods for selecting attributes (which will be described later) show that weather is not a discriminating attribute
- We use the principle of *Occam's Razor*: Given a number of competing hypotheses, the simplest one is preferable

Choosing Attributes

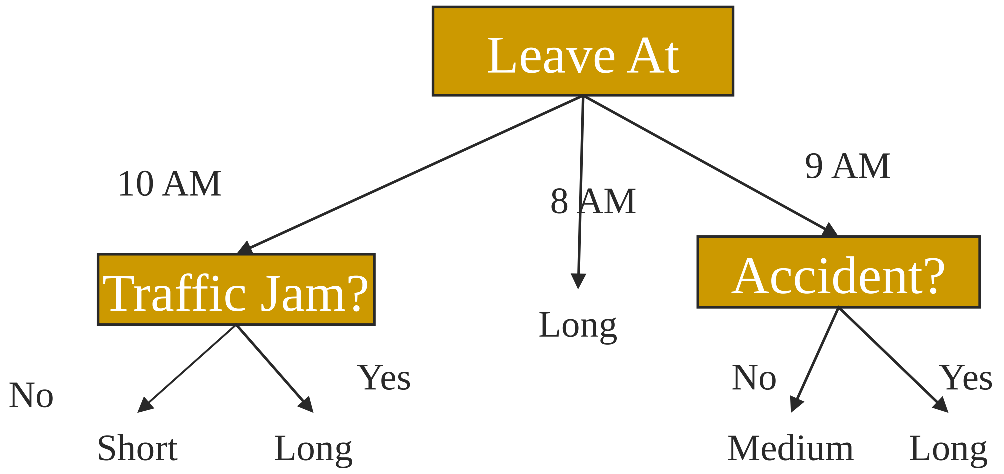
- The basic structure of creating a decision tree is the same for most decision tree algorithms
- The difference lies in how we select the attributes for the tree
- We will focus on the ID3 algorithm developed by Ross Quinlan in 1975

Decision Tree Algorithms

- The basic idea behind any decision tree algorithm is as follows:
  - Choose the *best* attribute(s) to split the remaining instances and make that attribute a decision node
  - Repeat this process for recursively for each child
  - Stop when:
    - All the instances have the same target attribute value
    - There are no more attributes
    - There are no more instances

Identifying the Best Attributes

- Refer back to our original decision tree



- How did we know to split on *leave at* and then on *Traffic Jam* and a*ccident* and not *weather*?

ID3 Heuristic

- To determine the best attribute, we look at the ID3 heuristic
- ID3 splits attributes based on their *entropy*.
- Entropy is the measure of disinformation…

Entropy

- Entropy is minimized when all values of the target attribute are the same.
  - If we know that commute time will always be *short*, then entropy = 0
- Entropy is maximized when there is an equal chance of all values for the target attribute (i.e. the result is random)
  - If commute time = short in 3 instances, medium in 3 instances and long in 3 instances, entropy is maximized

Entropy

- Calculation of entropy
  - Entropy(S) = $\sum_{(i=1 \text{ to } n)} -|S_i|/|S| * \log_2(|S_i|/|S|)$
    - S = set of examples
    - S_i = subset of S with value v_i under the target attribute
    - n = size of the range of the target attribute

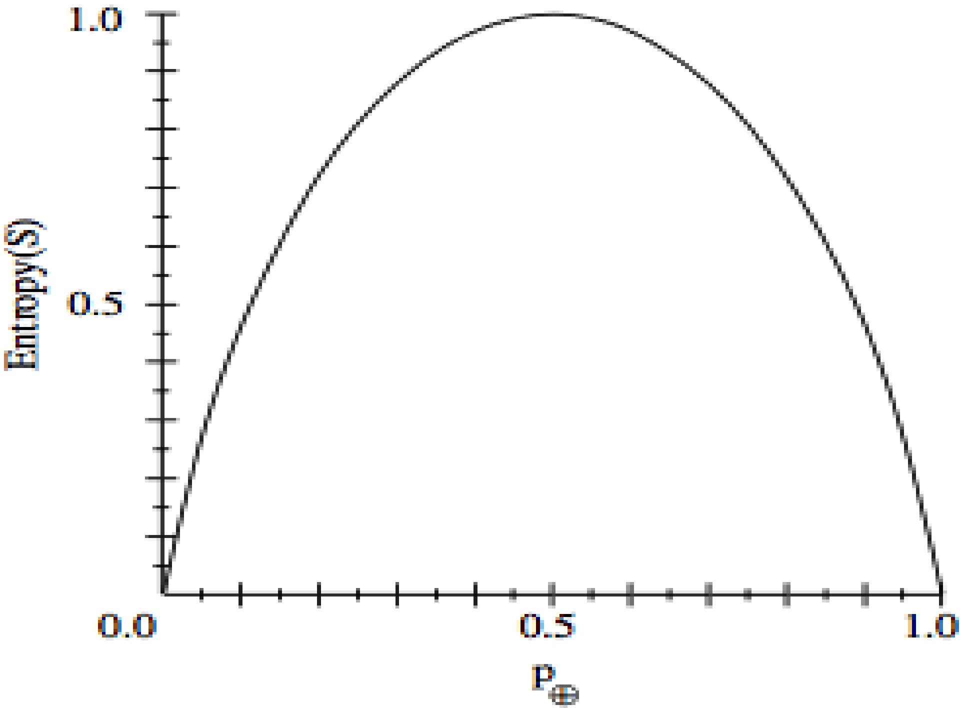
Example: 3 yes / 3 no

Entropy (S) = -3/6*log(3/6)-3/6*log(3/6) = 1

Example: 4yes / 0 no

Entropy (S) = -4/4*log(4/4) – 0/4*log(0/4) = 0

Entropy



ID3

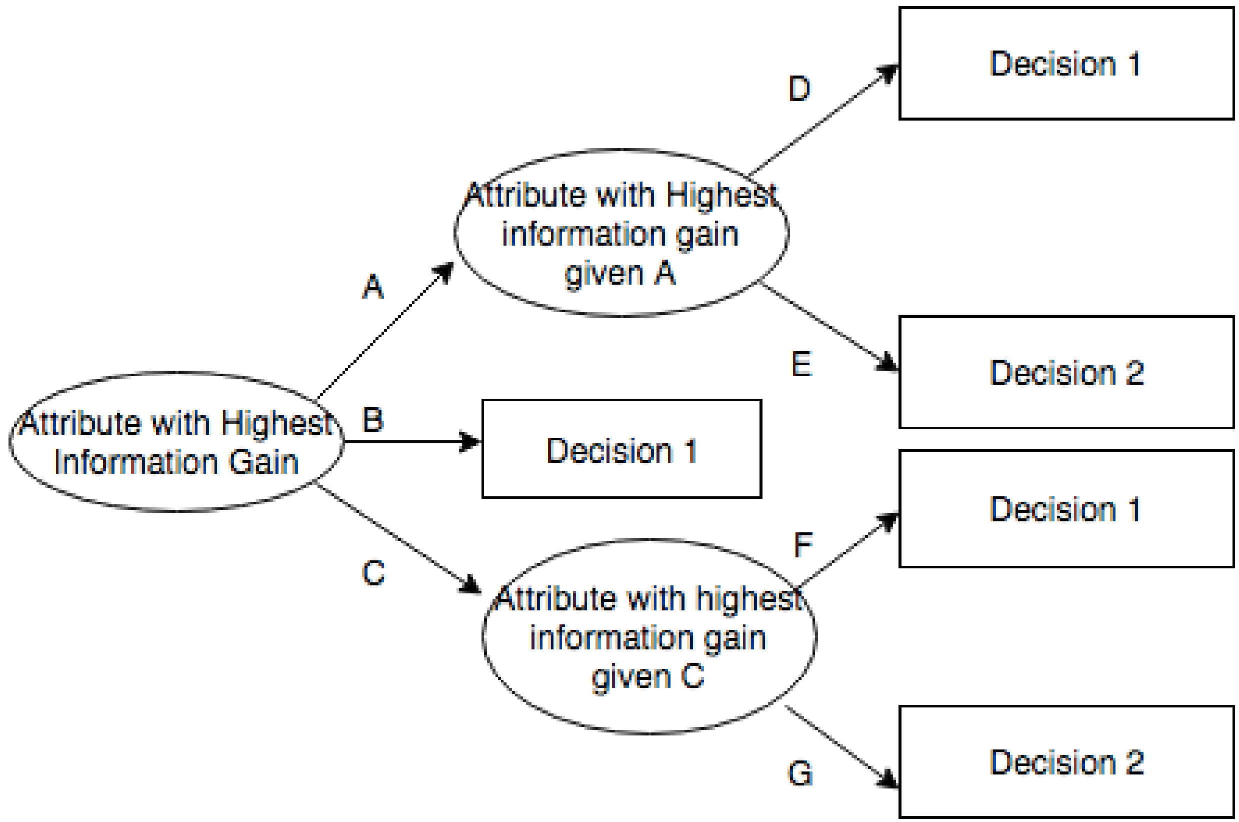
- ID3 splits on attributes with the lowest entropy
- We calculate the entropy for all values of an attribute as the weighted sum of subset entropies as follows:
  - $\sum_{(i = 1 \text{ to } k)} |S_i|/|S| \text{ Entropy}(S_i)$, where k is the range of the attribute we are testing
- We can also measure **information gain** (which is inversely proportional to entropy) as follows:
  - $\text{Entropy}(S) - \sum_{(i = 1 \text{ to } k)} |S_i|/|S| \text{ Entropy}(S_i)$

ID3

- Given our commute time sample set, we can calculate the entropy of each attribute at the root node

Attribute	**Expected Entropy**	**Information Gain**
Hour	0.6511	0.768449
Weather	1.28884	0.130719
Accident	0.92307	0.496479
Traffic Jam	1.17071	0.248842

Potential ID3-generated decision tree.



Problems with ID3

- ID3 is not optimal
  - Uses *expected* entropy reduction, not actual reduction
- Must use discrete (or discretized) attributes
  - What if we left for work at 9:30 AM?
  - We could break down the attributes into smaller values…

Problems with Decision Trees

- While decision trees classify quickly, the time for building a tree may be higher than another type of classifier

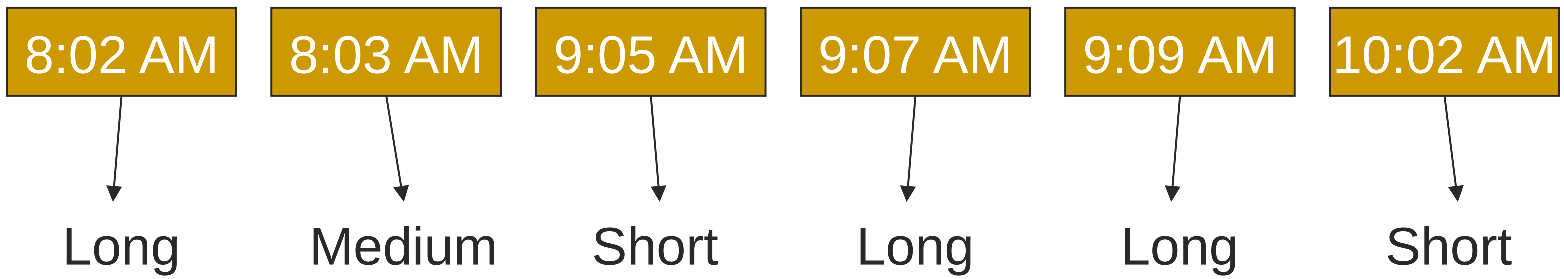
- Decision trees suffer from a problem of errors propagating throughout a tree
  - A very serious problem as the number of classes increases

Error Propagation

- Since decision trees work by a series of local decisions, what happens when one of these local decisions is wrong?
  - Every decision from that point on may be wrong
  - We may never return to the correct path of the tree

Problems with ID3

- If we broke down leave time to the minute, we might get something like this:



Since entropy is very low for each branch, we have n branches with n leaves. This would not be helpful for predictive modeling.

Problems with ID3

- We can use a technique known as discretization
- We choose *cut points*, such as 9AM for splitting continuous attributes
- These cut points generally lie in a subset of *boundary points*, such that a boundary point is where two adjacent instances in a sorted list have different target value attributes