Introduction to Decision Trees

Abdus Salam Azad

What Is Machine Learning

- Tom Mitchell - Improving with experience at some task

- Wikipedia - **Machine learning** is a scientific discipline that explores the construction and study of algorithms that can ***learn from data***

- **Algorithms that can improve their performance using data**

When should I play Tennis ????

Can You Give Us Some Examples !!!!!!!!

Classification

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Classification

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Seems Like I really enjoy it when the outlook is overcast!!!!!

Day	Outlook				PlayTennis
D1	Sunny				No
D2	Sunny			ong	No
D3	Overcast	Ho		Weak	Yes
					Yes
					Yes
					No
					Yes
					No
					Yes
					Yes
					Yes
					Yes
					Yes
D14	Rain	Mild	High	Strong	No

Classific

Day	Outlook	Temperature			
D1	Sunny	Hot			
D2	Sunny	Hot	High		
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Classification

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Classification

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

Classification

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

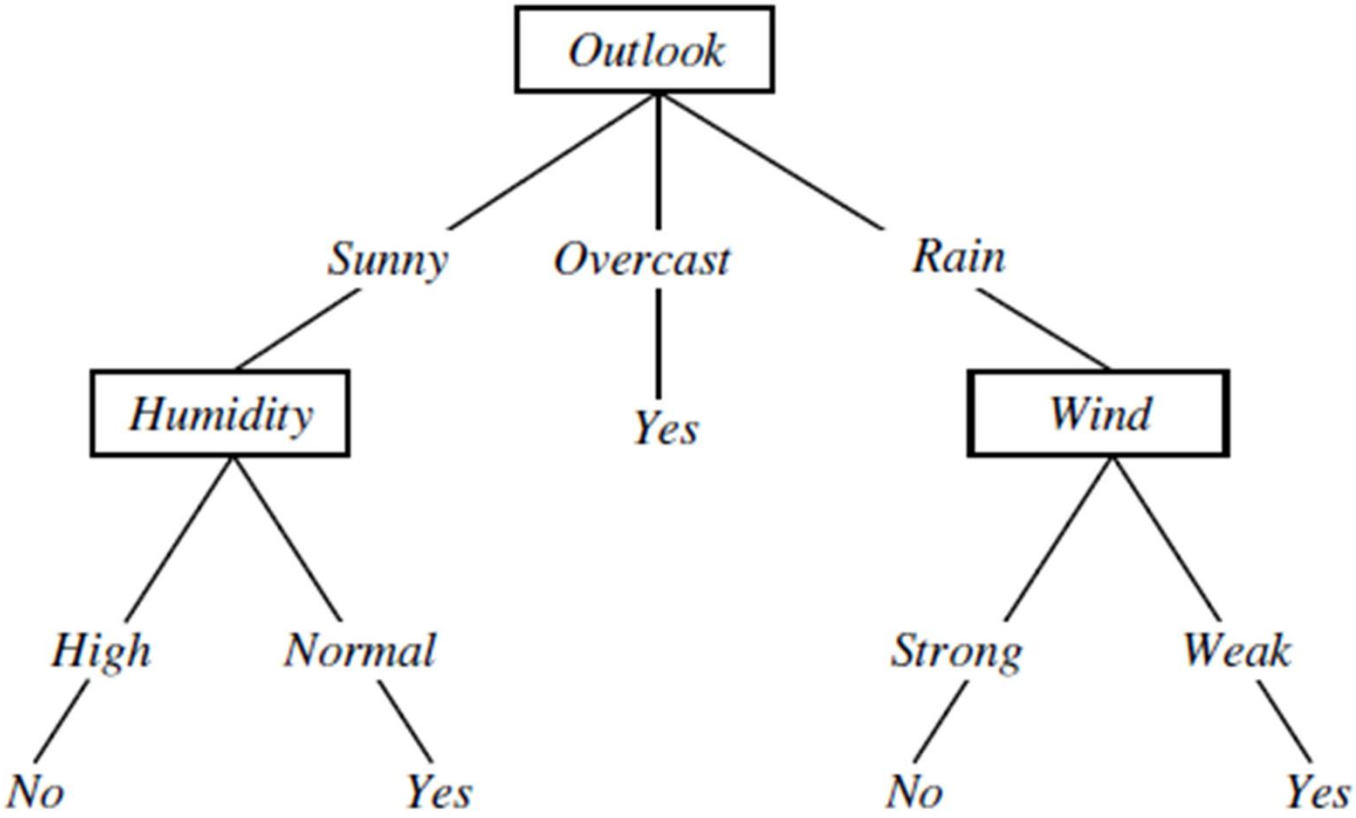
So, Its Humidity !!!!!!!!

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D14	Rain	Mild	High	Strong	No

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D14	Rain	Mild	High	Strong	No

Its Wind !

Learned Tree for Prediction



Thank you ML, Engineers !!!!!!

Feature Vector

Class Label

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

ID3(*Examples*, *Target_attribute*, *Attributes*)

Examples are the training examples. Target_attribute is the attribute whose value is to be predicted by the tree. Attributes is a list of other attributes that may be tested by the learned decision tree. Returns a decision tree that correctly classifies the given Examples.

- Create a *Root* node for the tree
- If all *Examples* are positive, Return the single-node tree *Root*, with label = +
- If all *Examples* are negative, Return the single-node tree *Root*, with label = −
- If *Attributes* is empty, Return the single-node tree *Root*, with label = most common value of *Target_attribute* in *Examples*
- Otherwise Begin
  - $A \leftarrow$ the attribute from *Attributes* that best* classifies *Examples*
  - The decision attribute for *Root* $\leftarrow A$
  - For each possible value, v_i, of A,
    - Add a new tree branch below *Root*, corresponding to the test $A = v_i$
    - Let $Examples_{v_i}$ be the subset of *Examples* that have value v_i for A
    - If $Examples_{v_i}$ is empty
      - Then below this new branch add a leaf node with label = most common value of *Target_attribute* in *Examples*
      - Else below this new branch add the subtree ID3($Examples_{v_i}$, *Target_attribute*, *Attributes* − {A}))
- End
- Return *Root*

ID3(*Examples*, *Target_att*

Examples are th … *be*
predicted by the … *ed*
decision tree. Ret

- Create a *Root* node
- If all *Examples* are p
- If all *Examples* are negative, … with label = −
- If *Attributes* is empty, Return the sin… ode … with label = most common value of *Target_attribute* in *Examples*
- Otherwise Begin
  - *A* ← the attribute from *Attributes* that best* classifies *Examples*
  - The decision attribute for *Root* ← *A*

…corresponding to the test $A = v_i$
…*mples* that have value v_i for A

… add a leaf node with label = most common
…*Examples*
…ld the subtree
…*tribute*, *Attributes* − {*A*}))

• How to choose the BEST ???
•More importantly what is the BEST ????

Entropy

- S is a sample of training examples
- $p_{\oplus}$ is the proportion of positive examples in S
- $p_{\ominus}$ is the proportion of negative examples in S
- Entropy measures the impurity of S

$$Entropy(S) \equiv -p_{\oplus} \log_2 p_{\oplus} - p_{\ominus} \log_2 p_{\ominus}$$

Entropy

- S is a sample of training examples
- $p_{\oplus}$ is the proportion of positive examples in S
- $p_{\ominus}$ is the proportion of negative examples in S
- Entropy measures the impurity of S

$$Entropy(S) \equiv -p_{\oplus} \log_2 p_{\oplus} - p_{\ominus} \log_2 p_{\ominus}$$

- Entropy Is **Zero (Minimum)**, When all the examples belong to **same class**!!!
- Entropy Is **1(Maximum)**, When there are **equal** number of positive and negative examples!

Entropy

- Entropy Is Zero (Minimum), When all the examples belong to same class!!!
- Entropy Is 1(Maximum), When there are equal number of positive and negative examples!

To illustrate, suppose S is a collection of 14 examples of some boolean concept, including 9 positive and 5 negative examples (we adopt the notation [9+, 5−] to summarize such a sample of data). Then the entropy of S relative to this boolean classification is

$$\begin{aligned} Entropy([9+, 5-]) &= -(9/14)\log_2(9/14) - (5/14)\log_2(5/14) \\ &= 0.940 \end{aligned} \tag{3.2}$$

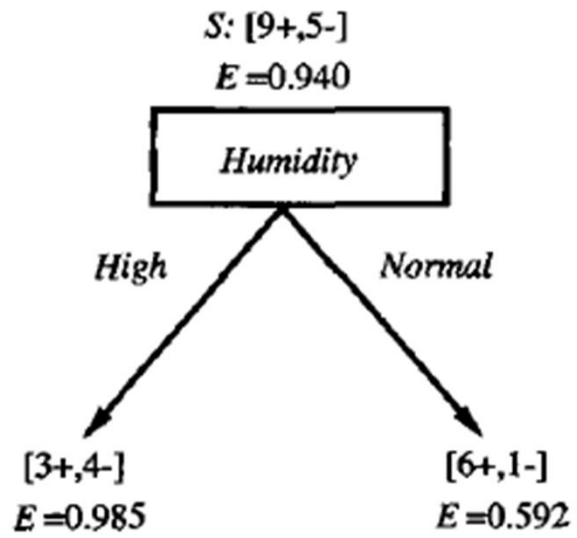
$$Entropy(S) \equiv \sum_{i=1}^{c} -p_i \log_2 p_i$$

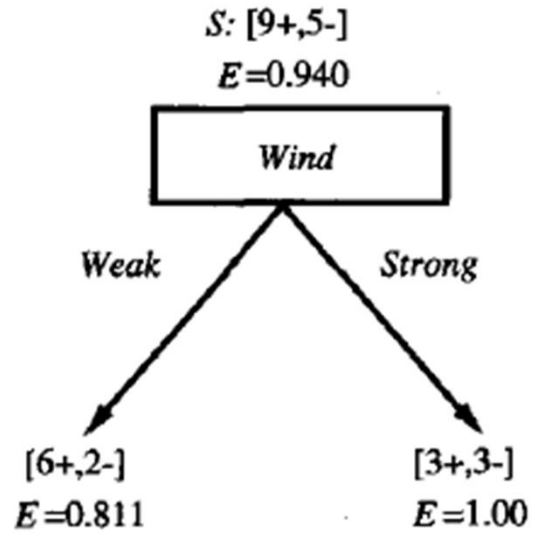
Information Gain

$$Gain(S, A) \equiv Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

$$\begin{aligned}
Values(Wind) &= Weak, Strong \\
S &= [9+, 5-] \\
S_{Weak} &\leftarrow [6+, 2-] \\
S_{Strong} &\leftarrow [3+, 3-] \\
Gain(S, Wind) &= Entropy(S) - \sum_{v \in \{Weak, Strong\}} \frac{|S_v|}{|S|} Entropy(S_v) \\
&= Entropy(S) - (8/14) Entropy(S_{Weak}) \\
&\quad - (6/14) Entropy(S_{Strong}) \\
&= 0.940 - (8/14)0.811 - (6/14)1.00 \\
&= 0.048
\end{aligned}$$

Which attribute is the best classifier?





Gain (S, Humidity)

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

Gain (S, Wind)

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

Day	*Outlook*	*Temperature*	*Humidity*	*Wind*	*PlayTennis*
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

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

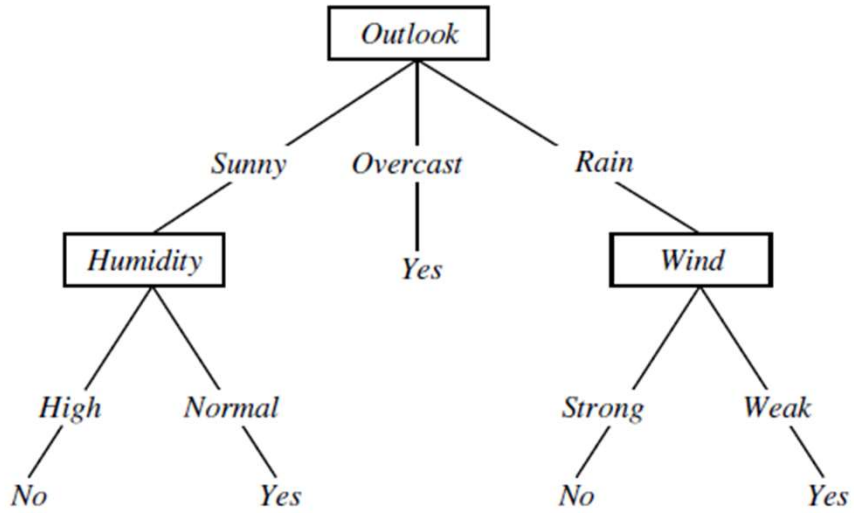
$$Gain(S, Humidity) = 0.151$$

$$Gain(S, Wind) = 0.048$$

$$Gain(S, Temperature) = 0.029$$

Predicting new examples

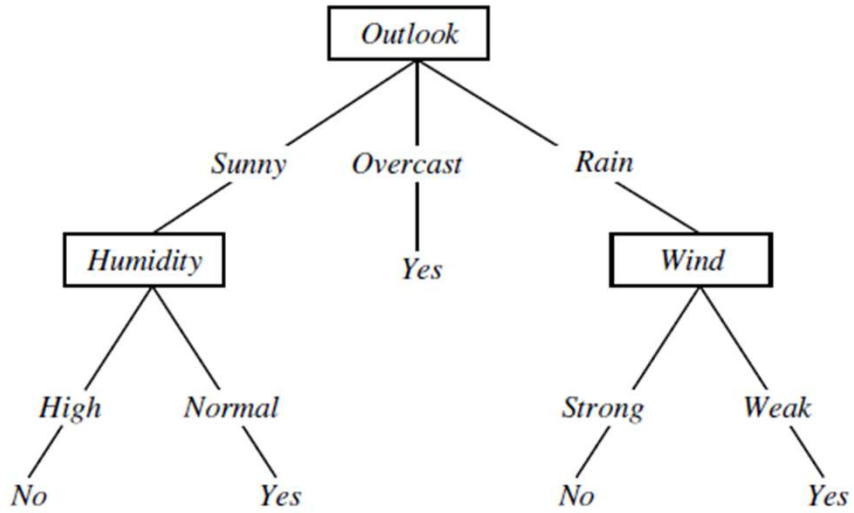
Day	*Outlook*	*Temperature*	*Humidity*	*Wind*	*PlayTennis*
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No



(Outlook, Temperature, Humidity, Wind)
(Sunny, Hot, Normal, Weak) == ?

Predicting new examples

Day	*Outlook*	*Temperature*	*Humidity*	*Wind*	*PlayTennis*
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No



(Outlook, Temperature, Humidity, Wind)
(Sunny, Cool, Normal, Weak) == ?

Evaluating the learned model

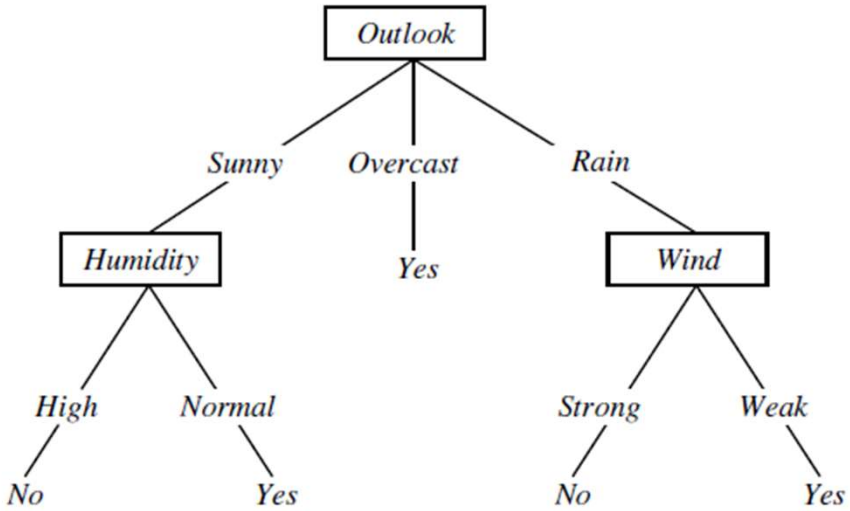
- Divide the data we have, in two sets
  - Training
  - Test

- We learn on training data
- We evaluate the performance on test data
  - Called validation
  - Test data is unseen for the model, as it has not seen this during training phase

- If the model has generalized well, it will perform “well” on the test data

Decision Tree Representation

- A disjunction (OR) of conjunctions (ANDs)



$$(Outlook = Sunny \wedge Humidity = Normal)$$
$$\vee \quad (Outlook = Overcast)$$
$$\vee \quad (Outlook = Rain \wedge Wind = Weak)$$

Hypothesis Space

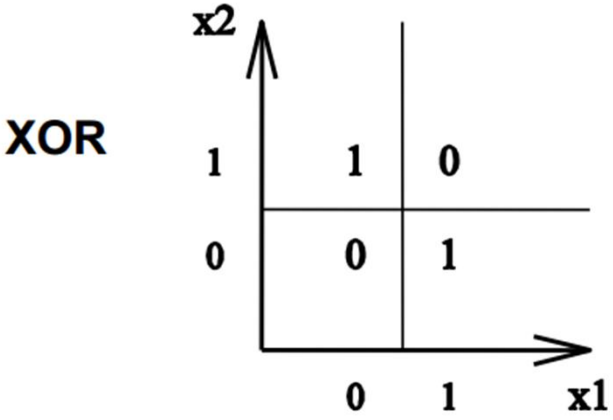
- There are many-many hypothesis
- It does a *simple to complex*, ***hill-climbing(greedy) search*** on the hypothesis space
  - The information gain guides the search
- The ID3 algorithm selects one of the hypotheses that correctly classifies the **training data**

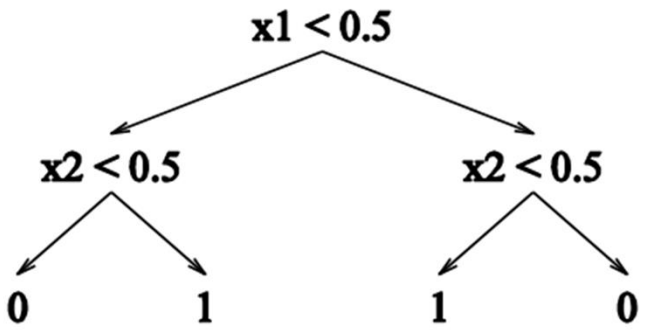
Some Insights on Decision Tree Learning using ID3

- Can only work with discrete valued attributes and classes/targets

- The hypothesis space is a complete space of finite-discrete valued function
  - Can represent any finite-discrete valued function

Decision Trees Can Represent Any Boolean Function





- If a target Boolean function has *n* inputs, there always exists a decision tree representing that target function.
- However, in the worst case, exponentially many nodes will be needed (why?)
  - 2^n possible inputs to the function
  - In the worst case, we need to use one leaf node to represent each possible input

Some Insights on Decision Tree Learning using ID3

- Maintains a single current hypothesis
  - There might be multiple “good” trees
  - ID3 will provide only one of them
  - Can not suggest alternatives

- Hill Climbing Search
  - No backtracking
  - Can converge in local minima

Some Insights on Decision Tree Learning using ID3

- Places the Attributes with more information gain towards the root


Why Reduce Entropy
???
Download from

Why Reduce Entropy ???

- Reducing entropy can lead to shorter trees (approximately)

- Inductive Bias of ID3:
  - Prefer shorter trees over taller ones
  - Places the Attributes with more information gain towards the root


How & Why am I preferring shorter trees ?
Download from

ID3 prefers shorter trees

- Occam's Razor: Prefer the simplest hypothesis that fits the data

- ID3 considers shorter hypotheses simpler than taller ones

Why Simpler?

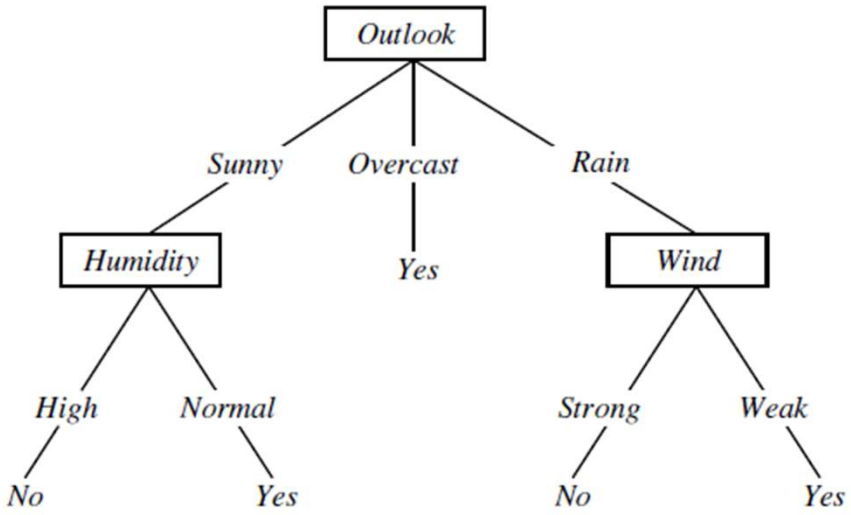
- No proof that simplest is the best
- One argument can be
  - Fewer Shorter Hypothesis than Taller Hypothesis
  - Less chance of co-incidence

Overfitting

Consider adding noisy training example #15:

$$Sunny,\ Hot,\ Normal,\ Strong,\ PlayTennis = No$$

Day	*Outlook*	*Temperature*	*Humidity*	*Wind*	*PlayTennis*
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No



Overfitting – A not so exact definition

- Say we have trained a model and named it A
- A has 99% accuracy on train data and
  60% accuracy on test data
- Consider, there can be found another model on the same training data, B with the following property
- B has 70% accuracy on train data and
  80% accuracy on test data
- We say that A has overfit the training data

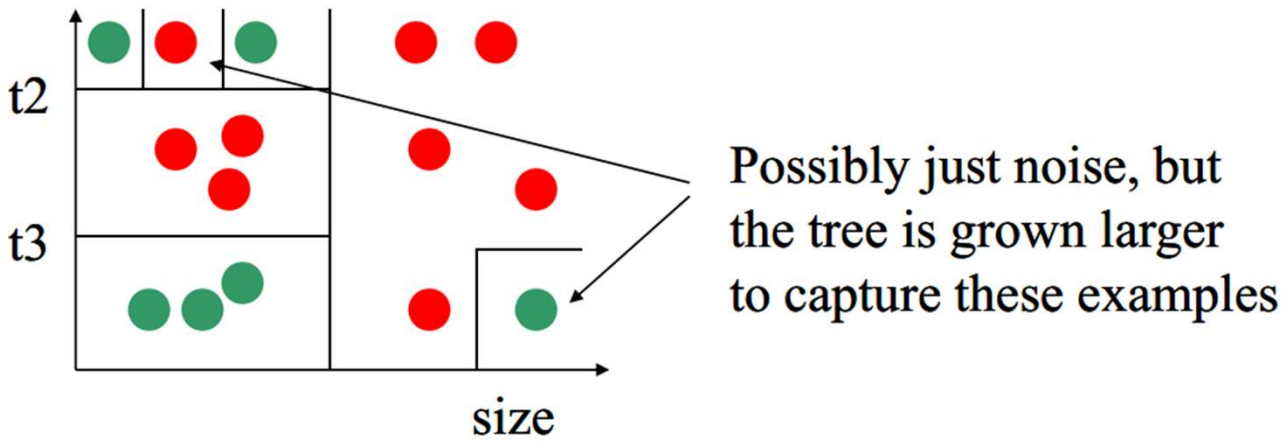
Overfitting

Definition: Given a hypothesis space H, a hypothesis $h \in H$ is said to **overfit** the training data if there exists some alternative hypothesis $h' \in H$, such that h has smaller error than h' over the training examples, but h' has a smaller error than h over the entire distribution of instances.

With increasing number of nodes, accuracy (on test set) decreases

Bigger trees can cause overfitting

- Decision tree has a very flexible hypothesis space
- As the nodes increase, we can represent arbitrarily complex decision boundaries – training set error is always zero



Overfitting

- Can happen when there is noise in data
- Not enough data to generalize the true target function
  - co-incidental regularities

Avoiding Overfitting in DT

- Stop Growing Tree Earlier
- Post Pruning the Tree

- approaches that stop growing the tree earlier, before it reaches the point where it perfectly classifies the training data,
- approaches that allow the tree to overfit the data, and then post-prune the tree.

Avoiding Overfitting in DT

- **Early Stop**: Stop Growing Tree Earlier
  - How to decide when stop learning
- **Post-Pruning**: Post Pruning the Tree
  - Shown better results

Validation Set Approach

- Take out some examples from the train set
  - We call this the “Validation Set”
  - The reduced-size Training Set is still called the training set
- Use the accuracy on the validation set as an estimate of the true accuracy (~accuracy on test set)
- The motivation is that the validation set might just not show the same random noise and co incidental regularities that the training set has

Early Stopping

- One simple approach can be

- Before adding each node in the tree
  - Compare the validation accuracy before adding this node
  - And after this node
  - If the accuracy is not increased do not add the node

Post pruning

- We will look into two approaches
  - Reduced Error Pruning
  - Rule Post Pruning

Reduced Error Pruning

- This method repeatedly prunes node from the tree one by one
- The process is stopped when the pruning results decrease in validation accuracy
- Removal of a node:
  - Remove the subtree rooted at that node
  - Add a leaf node with the most common label of that subtree (of the examples corresponding to that subtree)

Reduced Error Pruning

- Consider all node for pruning
- find the node n, removing which results in maximum validation accuracy
- if new validation accuracy is less than previous
  - stop pruning
- else
  - prune the node n,
  - repeat the procedure

Rule-Post Pruning

- A variant is used in C4.5 (an improved version of the basic ID3)
- Each Leaf gives one rule

1. Infer the decision tree from the training set, growing the tree until the training data is fit as well as possible and allowing overfitting to occur.
2. Convert the learned tree into an equivalent set of rules by creating one rule for each path from the root node to a leaf node.
3. Prune (generalize) each rule by removing any preconditions that result in improving its estimated accuracy.
4. Sort the pruned rules by their estimated accuracy, and consider them in this sequence when classifying subsequent instances.

Rule-Post Pruning

- The estimated accuracy is estimated on the examples that the rule is applicable to

1. Infer the decision tree from the training set, growing the tree until the training data is fit as well as possible and allowing overfitting to occur.
2. Convert the learned tree into an equivalent set of rules by creating one rule for each path from the root node to a leaf node.
3. Prune (generalize) each rule by removing any preconditions that result in improving its estimated accuracy.
4. Sort the pruned rules by their estimated accuracy, and consider them in this sequence when classifying subsequent instances.

Incorporating Continuous Attributes

- The continuous attribute A
- Dynamically select a boundary *c*
  - Select a boundary that most increases the information gain
- Add a new Attribute *Ac*
  - *Ac* = True if *Ac* < *c*
  - False otherwise

Incorporating Continuous Attributes

- For example, consider the temperature that Nadal gave was in Celsius
- We are considering a node where there is 6 examples

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

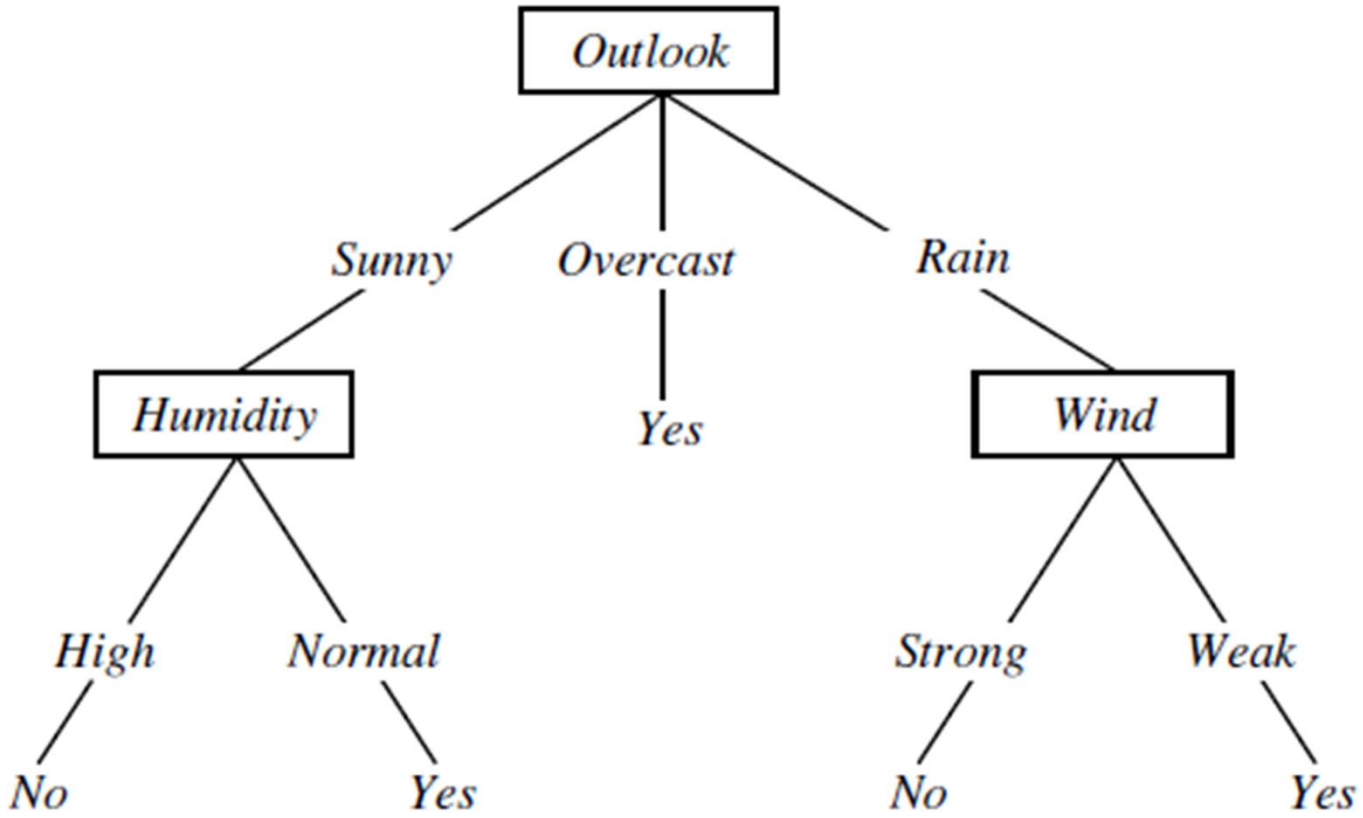
Incorporating Continuous Attributes

- Sort the examples based on the attribute

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

- Identify the label transitions
  - 48->60, 80->90
- Set of candidate thresholds
  - 54, 85
- Find the gain for both threshold, take the maximum (54 in this case)

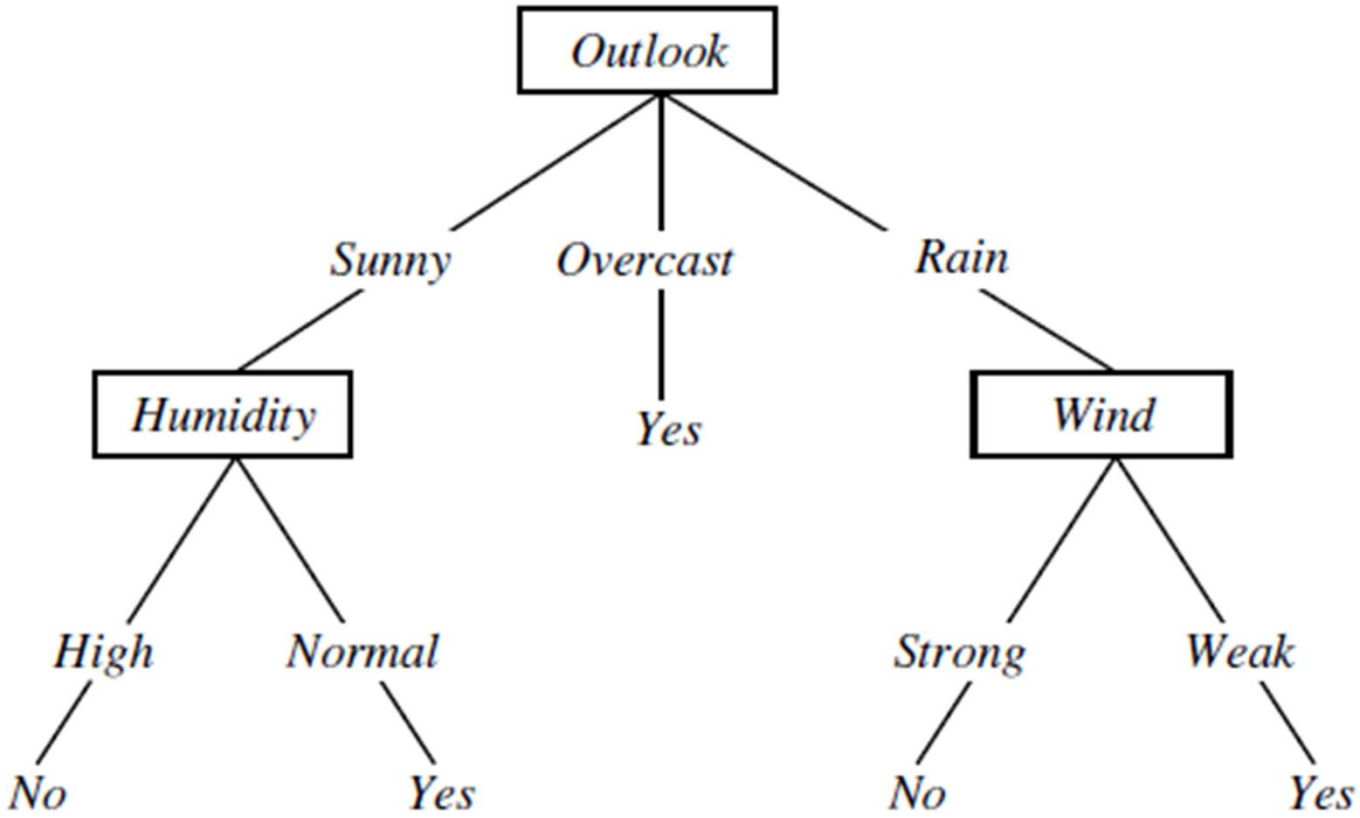
The learned model is easily interpretable



When the Outlook is Sunny and Humidity is High -> Don't play tennis

\+ When the Outlook is Sunny and Humidity is Normal -> Play tennis ….

The learned model is easily interpretable



You know how the model takes the decision.

Train, Test, & Validation Test

- Which set(s) has/have the training algorithm seen?
  - Train
  - Test
  - Validation

Train, Test, & Validation Test

- Which set(s) has/have the training algorithm seen?
  - Train
  - Test
  - Validation

- The training algorithm only sees the Training dataset directly

Train, Test, & Validation Test

- Which set(s) has/have impact on the trained model?
  - Train
  - Test
  - Validation

Train, Test, & Validation Test

- Which set(s) has/have impact on the trained model?
  - Train
  - Test
  - Validation

- The model is built based on training dataset
- Using validation set, we tuned the model (eg. pruning)
  - Hence, it also has impact on the learned model

Train, Test, & Validation Test

- On which sets, the accuracy is closest to the true accuracy?
  - Train
  - Test
  - Validation

Train, Test, & Validation Test

- On which sets, the accuracy is closest to the true accuracy?
  - Train
  - Test
  - Validation

- The testset is completely unseen by the algorithm

Important Concepts to learn

- Feature Vector and feature space
- Over Fitting
- Generalization
- Training, Test, and Validation Data
- Noise in training data
- Hypothesis, Hypothesis Space
- Etc..


Why didn’t I just classify over the attribute “Day” ???
Download from

What will be the information gain for the attribute day ??

Day	*Outlook*	*Temperature*	*Humidity*	*Wind*	*PlayTennis*
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

What will be the information gain for the attribute day ??

Day	*Outlook*	*Temperature*	*Humidity*	*Wind*	*PlayTennis*
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Date has so many possible values!!

- Date has many possible values
- Typically it will divide the training set into a large number of small subsets
  - Typically it will have a high information gain
- How to change the measure so that attributes such as dates are not used??

Gain Ratio

- Entropy of S w.r.to the values of an attribute

$$SplitInformation(S, A) \equiv -\sum_{i=1}^{c} \frac{|S_i|}{|S|} \log_2 \frac{|S_i|}{|S|}$$

- Gain Ratio

$$GainRatio(S, A) \equiv \frac{Gain(S, A)}{SplitInformation(S, A)}$$

- Split Information discourages Attributes which uniformly distributes data

Split Information

- Attributes like date (one example per value)
  - SI: $\log_2 n$; n = number of examples
  - Large Split Information
  - Small Gain Ratio
- Attributes the cut the data in half
  - One

Split Information

- Attributes that most examples for one value and very less for the others
  - A = a => 127
  - A = ~a => 1
  - SI: Large Value

One practical consideration

- If the split information becomes very small
  - Gain can be overflowed!!

Handling Missing Values

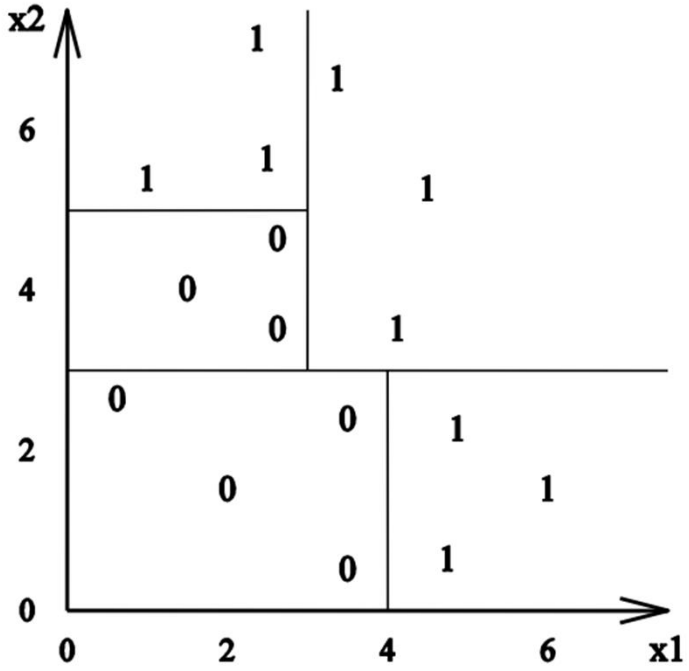
- Nominal Values
  - Most Common Value
  - Most Common Value for that class
  - Using Probability Estimation
- Numeric Values
  - Average
  - Average for that class
  - Probability, etc

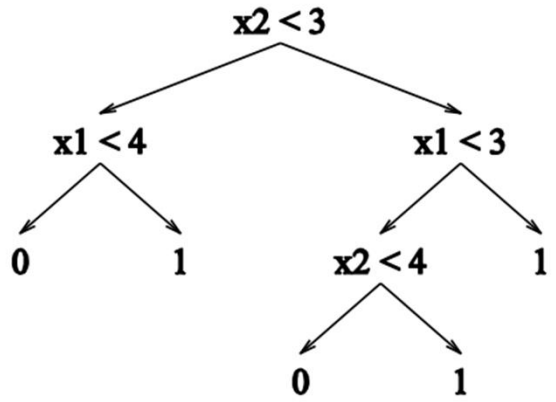
Handling Attributes with Differing Costs

- Different attribute testing can have different cost

$$\frac{Gain^2(S, A)}{Cost(A)} \qquad \frac{2^{Gain(S, A)} - 1}{(Cost(A) + 1)^w}$$

Decision Trees divide the feature space into axis-parallel rectangles and label each rectangle with one of the K classes





Learning From Examples

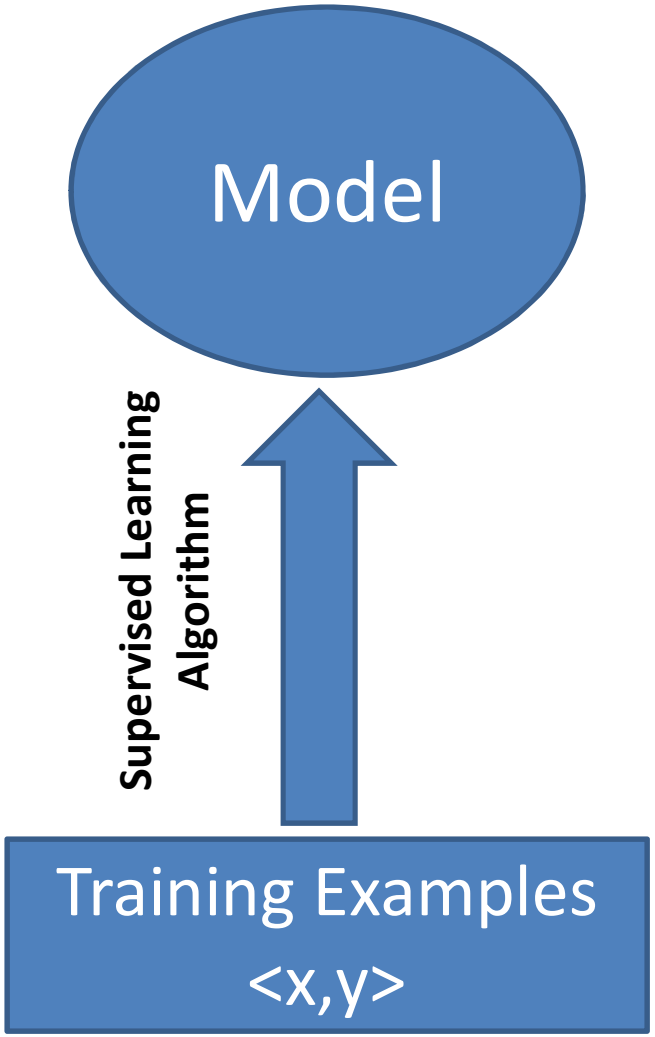
- The problem we just saw is a “Classification” problem
  - Which is a member of a broader class
    - Supervised Learning
- The approach we learned is a “decision tree classifier”
  - The algorithm name is ID3

Supervised Learning

- Given a **training set** of N example input-output pairs (x1, y1),(x2, y2), . . .(xN, yN)
  - each y_j was generated by an unknown function $y = f(x)$
  - x is a vector
  - Labeled dataset
- Discover a function h that approximates the true function f.
- The function h is called a **hypothesis**

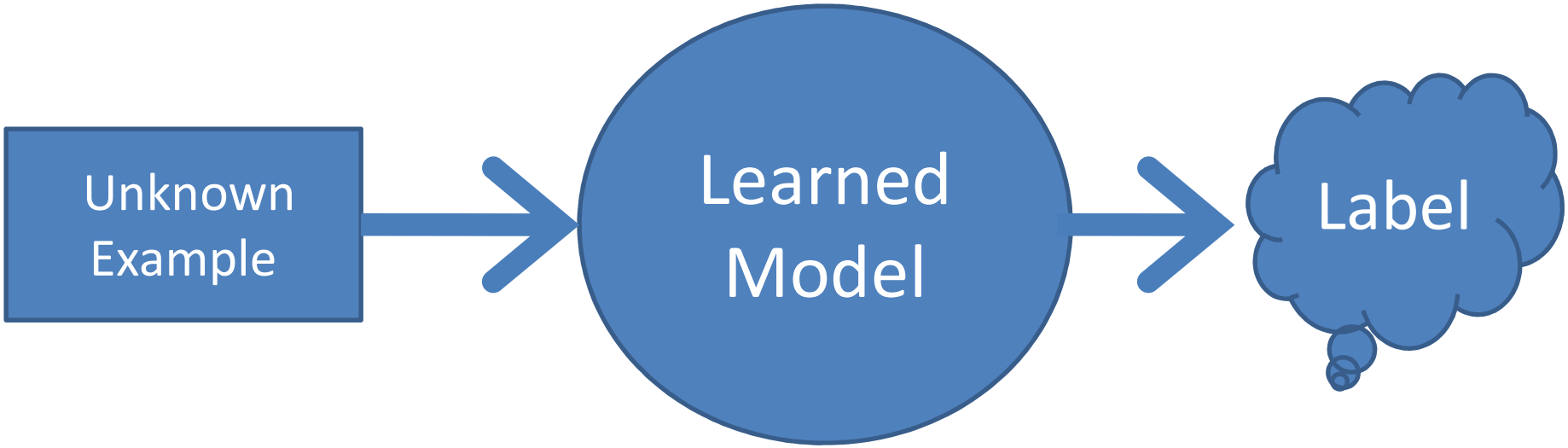
Supervised Learning - Training Phase

- Training Phase: The supervised learning algorithm first learn the model using the labeled example



Supervised Learning - Test Phase

Use the model to predict the label of an unknown example



Resource

- Decision Tree: Tom Mitchell, Chapter Three
- http://web.engr.oregonstate.edu/~xfern/classes/cs534/notes/decision-tree-7-11.pdf
- http://www.cs.cmu.edu/~awm/15781/slides/DTreesAndOverfitting-9-13-05.pdf
- Google