

# **Machine Learning 10-715, Fall 2018**

## **Introduction, Admin, Course Overview**

Lecture 1, 08/27/2018

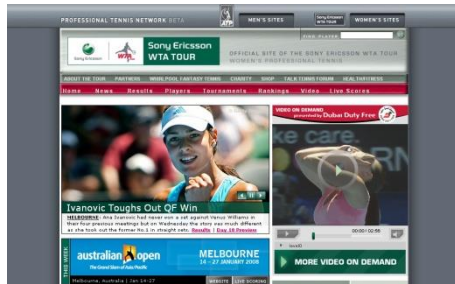
Maria-Florina (Nina) Balcan

# Machine Learning

Image Classification



Document Categorization



Speech Recognition

Protein Classification

Spam Detection

Branch Prediction

Fraud Detection

Natural Language Processing

Playing Games

Computational Advertising

# Machine Learning is Changing the World

“Machine learning is the hot new thing”  
(John Hennessy, President, Stanford)



“A breakthrough in machine learning would be worth ten  
Microsofts” (Bill Gates, Microsoft)

“Web rankings today are mostly a matter of machine learning”  
(Prabhakar Raghavan, VP Engineering at Google)

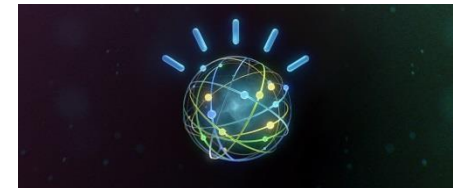


SMARTER THAN YOU THINK

Aiming to Learn as We Do, a Machine Teaches Itself



Jeff Swensen for The New York Times



# The COOLEST TOPIC IN SCIENCE

- “A breakthrough in machine learning would be worth ten Microsofts” (Bill Gates, Chairman, Microsoft)
- “Machine learning is the next Internet” (Tony Tether, Director, DARPA)
- Machine learning is the hot new thing” (John Hennessy, President, Stanford)
- “Web rankings today are mostly a matter of machine learning” (Prabhakar Raghavan, Dir. Research, Yahoo)
- “Machine learning is going to result in a real revolution” (Greg Papadopoulos, CTO, Sun)
- “Machine learning is today’s discontinuity” (Jerry Yang, CEO, Yahoo)

# This course: advanced introduction to machine learning.

- Cover (some of) the most commonly used machine learning paradigms and algorithms.
  - Sufficient amount of details on their mechanisms: explain why they work, not only how to use them.
  - Applications.

## This course: advanced introduction to machine learning.

- Cover (some of) the most commonly used machine learning paradigms and algorithms.
  - Sufficient amount of details on their mechanisms: explain why they work, not only how to use them.
  - Applications.
- 10-715 intended for PhD students in MLD.
- **Fastest-paced and most mathematical of the Intro to ML courses.**
- In addition to the goals listed above, 10-715 is intended to prepare students to write research papers that rely on and contribute to ML.
- PhD students from other departments (e.g., CSD or RI) might consider this course if their research depends strongly on and contributes to machine learning. MS students in MLD have the option of taking 10-715 or 10-701.

# **What is Machine Learning?**

**Examples of important machine learning paradigms.**

# **Supervised Classification**

**from data to discrete classes**



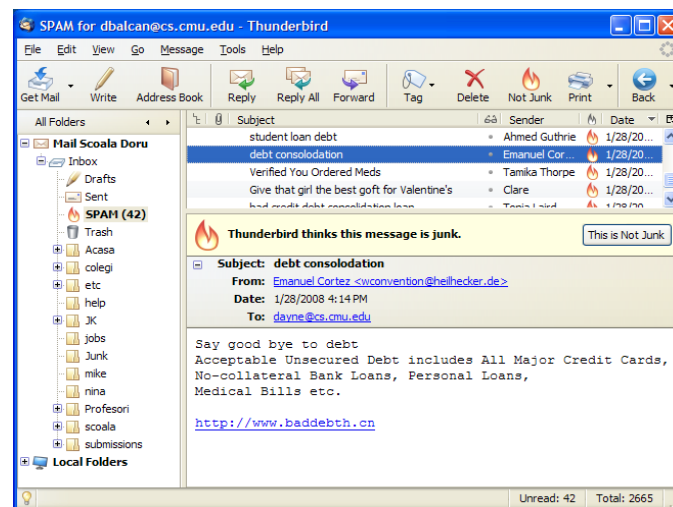
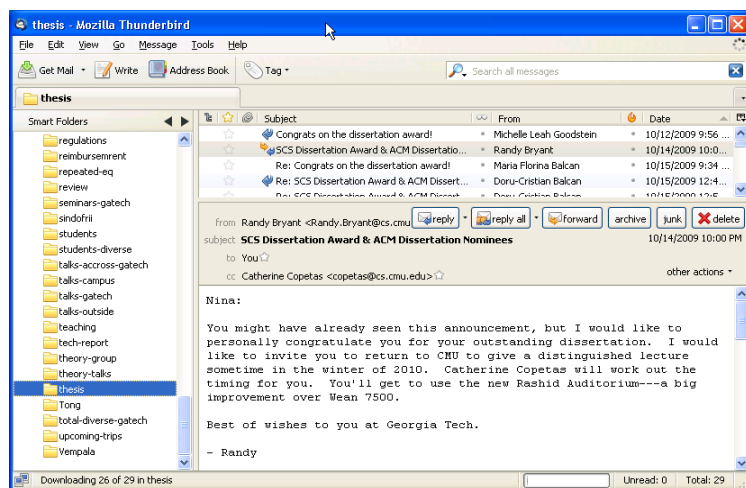
# Supervised Classification. Example: Spam Detection

Decide which emails are spam and which are important.

Not spam

Supervised classification

spam



**Goal:** use emails seen so far to produce good prediction rule for **future** data.

# Supervised Classification. Example: Spam Detection

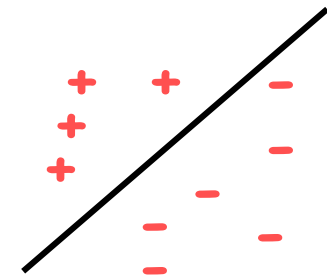
Represent each message by features. (e.g., keywords, spelling, etc.)

	"money"	"pills"	"Mr."	bad spelling	known-sender	spam?	
	Y	N	Y	Y	N	Y	
	N	N	N	Y	Y	N	
	N	Y	N	N	N	Y	
example	Y	N	N	N	Y	N	label
	N	N	Y	N	Y	N	
	Y	N	N	Y	N	Y	
	N	N	Y	N	N	N	

Reasonable RULES:

Predict SPAM if unknown AND (money OR pills)

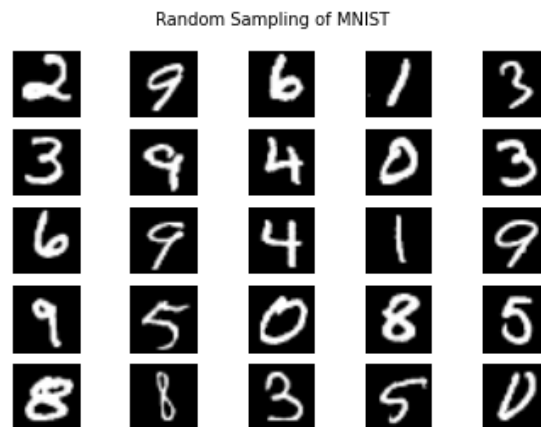
Predict SPAM if  $2\text{money} + 3\text{pills} - 5\text{known} > 0$



Linearly separable

# Supervised Classification. Example: Image classification

- Handwritten digit recognition  
(convert hand-written digits to characters 0..9)



- Face Detection and Recognition



# Supervised Classification. Many other examples

- Weather prediction



- Medicine:
  - diagnose a disease
    - input: from symptoms, lab measurements, test results, DNA tests, ...
    - output: one of set of possible diseases, or “none of the above”
    - examples: audiology, thyroid cancer, diabetes, ...
      - or: response to chemo drug X
      - or: will patient be re-admitted soon?
- Computational Economics:
  - predict if a stock will rise or fall
  - predict if a user will click on an ad or not
    - in order to decide which ad to show

# Regression. Predicting a numeric value

Stock market



Weather prediction



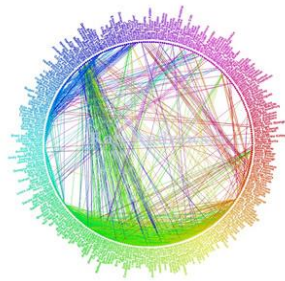
Temperature  
72° F

Predict the temperature at any given location

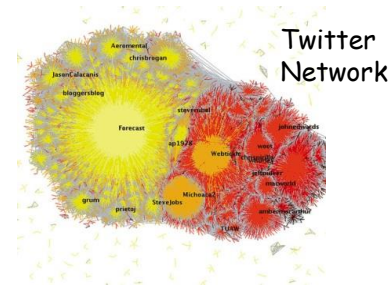
# Other Machine Learning Paradigm

**Clustering: discovering structure in data (only unlabeled data)**

- E.g, cluster users of social networks by interest (community detection).



Facebook network



Twitter Network

**Semi-Supervised Learning: learning with labeled & unlabeled data**

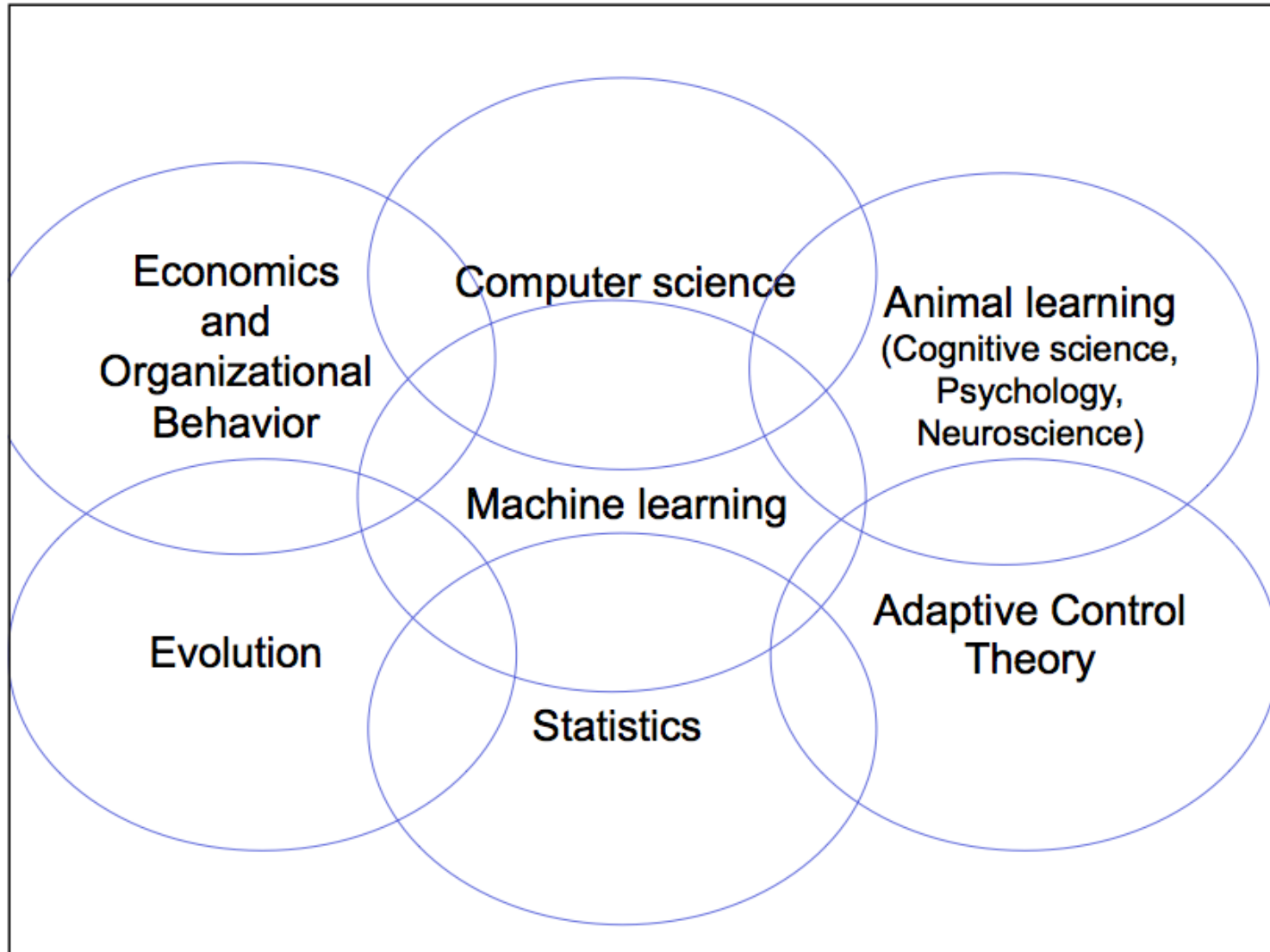
**Active Learning: learns pick informative examples to be labeled**

**Reinforcement Learning (acommodates indirect or delayed feedback)**

**Dimensionality Reduction**

**Collaborative Filtering (Matrix Completion), ...**

# Many communities relate to ML



Admin, Logistics, Grading



# Brief Overview

- Meeting Time: Mon, Wed, GHC 4307, 10:30 – 11:50
- Course Staff
  - Instructors:
    - Maria Florina (Nina) Balcan ([ninamf@cs.cmu.edu](mailto:ninamf@cs.cmu.edu))
  - TAs:
    - Paul Pu Liang ([paul.liangpu@gmail.com](mailto:paul.liangpu@gmail.com))
    - Barun Patra ([bpatra@andrew.cmu.edu](mailto:bpatra@andrew.cmu.edu))
    - Qizhe Xie ([qizhex@gmail.com](mailto:qizhex@gmail.com))

# Brief Overview

- Course Website

<http://www.cs.cmu.edu/~10715-f18/>

- See website for:
  - Syllabus details
    - All the lecture slides and homeworks
    - Additional useful resources.
  - Office hours
  - Recitation sessions
  - Grading policy
  - Honesty policy
  - Late homework policy
  - Piazza pointers
- Will use Piazza for discussions.

# Prerequisites. What do you need to know now?

- No official prerequisites. However, we will assume **mathematical and programming maturity**.
  - Calculus (multivariate)
  - Probability/statistics
  - Algorithms. Big O notation.
  - Linear algebra (matrices and vectors)
  - Programming:
    - You will implement some of the algorithms and apply them to datasets
    - Assignments will be in Python or Matlab/Octave (play with that now if you want)
    - Octave is open-source software clone of Matlab.

# Source Materials

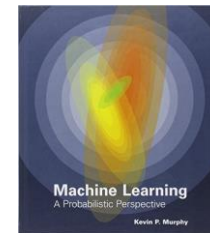
No textbook required. Will point to slides and freely available online material.

Useful textbooks:

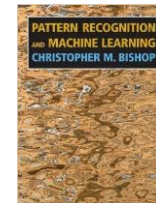
*Machine Learning*, Tom Mitchell, McGraw Hill, 1997.



*Machine Learning: a Probabilistic Perspective*,  
K. Murphy, MIT Press, 2012



*Pattern Recognition and Machine Learning*  
Christopher Bishop, Springer-Verlag 2006

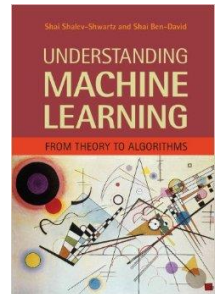


# Source Materials

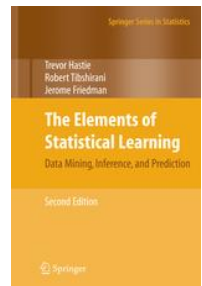
No textbook required. Will point to slides and freely available online material.

Useful textbooks:

*Understanding Machine Learning: From Theory to Algorithms*,  
S. Shalev-Shwartz & S. Ben-David, Cambridge University Press, 2014.



*The Elements of Statistical Learning: Data Mining, Inference and Prediction*, T. Hastie, R. Tibshirani, J. Friedman, Springer 2009.



# Grading

- 30% for homeworks.
- 20% for midterm
- 20% for final
- 25% for project
- 5% for class participation.
  - Piazza polls in class: bring a laptop or a phone
- At most 6 homeworks
  - Theory/math handouts
  - Programming exercises; applying/evaluating existing learners
  - Late assignments:
    - Up to 50% credit if it's less than 48 hrs late
    - You can drop your lowest assignment grade

# Collaboration policy

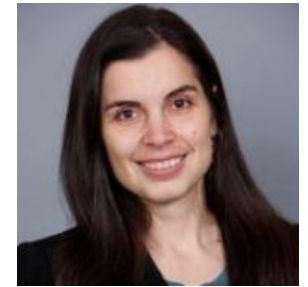
- Discussion of anything is ok...
- ...but the goal should be to *understand* better, not save work.
- So:
  - *no notes* of the discussion are allowed...the only thing you can take away is whatever's in your brain.
  - you should acknowledge who you got help from/did help in your homework

# Brief Overview

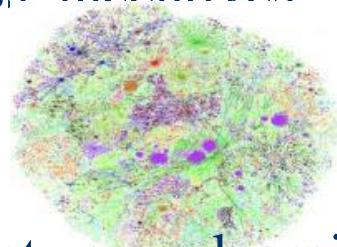
- Meeting Time: Mon, Wed, GHC 4307, 10:30 – 11:50
- Course Staff
  - Instructors:
    - Maria Florina (Nina) Balcan ([ninamf@cs.cmu.edu](mailto:ninamf@cs.cmu.edu))
  - TAs:
    - Paul Pu Liang ([paul.liangpu@gmail.com](mailto:paul.liangpu@gmail.com))
    - Barun Patra ([bpatra@andrew.cmu.edu](mailto:bpatra@andrew.cmu.edu))
    - Qizhe Xie ([qizhex@gmail.com](mailto:qizhex@gmail.com))



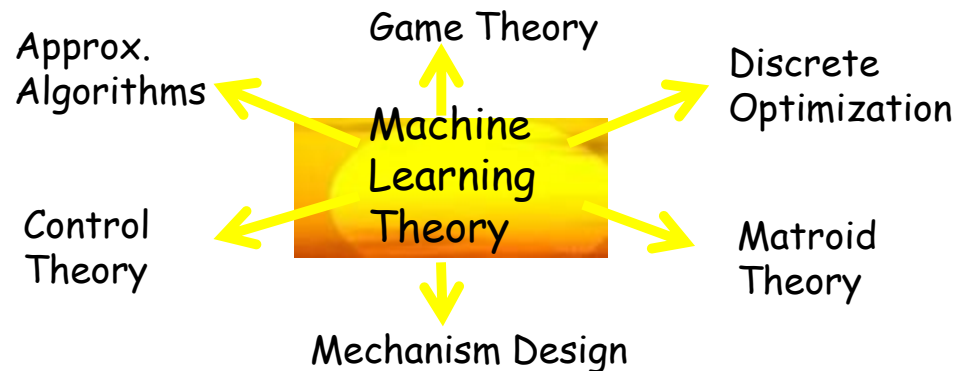
# Maria-Florina Balcan: Nina



- Foundations for Modern Machine Learning
  - E.g., deep learning, interactive, distributed, life-long learning; privacy, fairness incentives



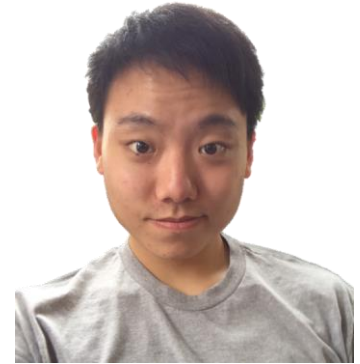
- Connections between learning & other fields (algorithms, algorithmic game theory)



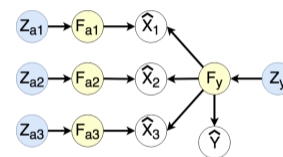
- Program Committee Chair for ICML 2016, COLT 2014

# Paul Liang

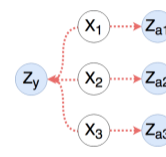
- MLD PhD student advised by LP Morency and Russ Salakhutdinov
- Research Interests:
  - Theory and applications of multimodal machine learning
  - Multimodal self-supervised learning, transfer learning, generative learning, reinforcement learning



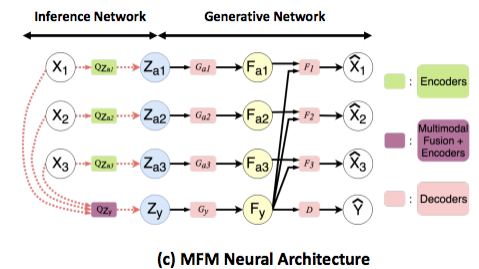
Language: *And he I don't think he got mad when hah I don't know maybe.* *Too much too fast, I mean we basically just get introduced to this character...* *All I can say is he's a pretty average guy.*



(a) MFM Generative Network



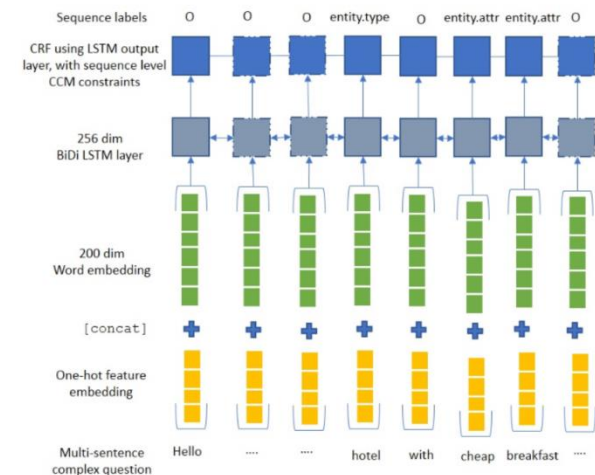
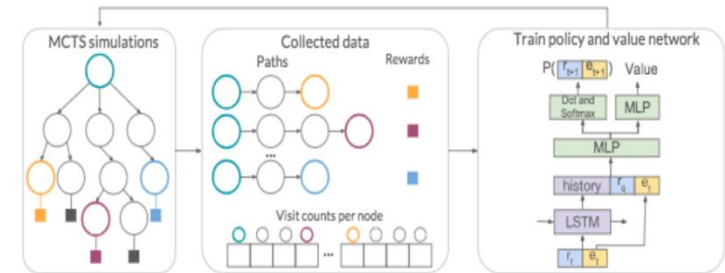
(b) MFM Inference Network



[www.cs.cmu.edu/~плиang/](http://www.cs.cmu.edu/~плиang/)

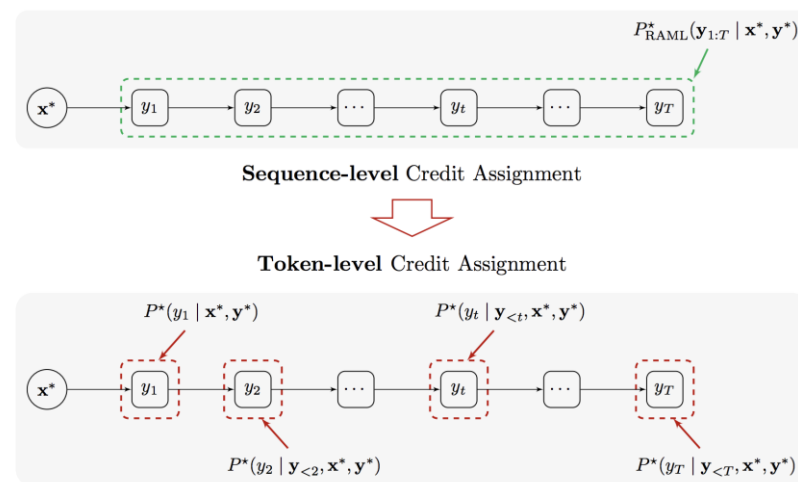
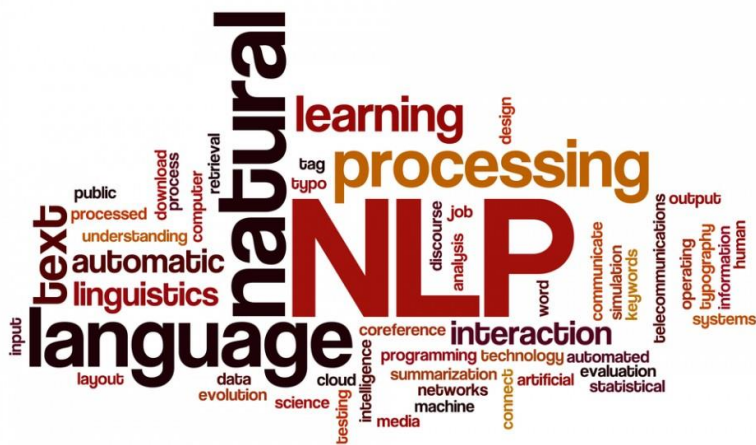
# Barun Patra

- 2<sup>nd</sup> Year ML Masters Student
- Advised by Prof. Matt Gormley
- Semi-supervised MT
- Latent Variable Models for QA
- KB Based Question Answering



# Qizhe Xie

- PhD student
- Research interests:
  - Deep learning
    - Semi-supervised learning
    - Generative model
  - Natural language processing



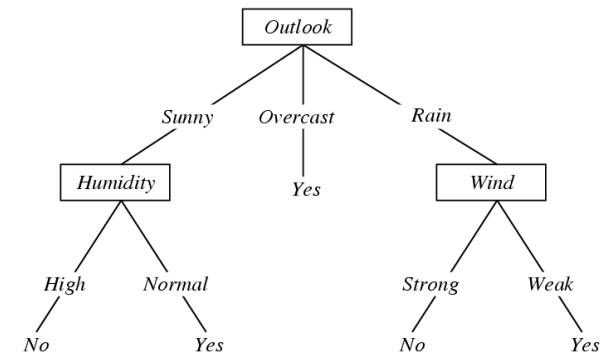
# Learning Decision Trees.

## Supervised Classification.

Useful Readings:

- Mitchell, Chapter 3
- Bishop, Chapter 14.4
- “On the Boosting Ability Of Top-Down Decision Tree Learning Algorithms”.  
M Kearns, Y. Mansour, Journal of Computer and System Sciences 1999

DT learning: Method for learning discrete-valued target functions in which the function to be learned is represented by a decision tree.



# Supervised Classification: Decision Tree Learning

**Example:** learn concept **PlayTennis** (i.e., decide whether our friend will play tennis or not in a given day)

Simple  
Training  
Data Set

example

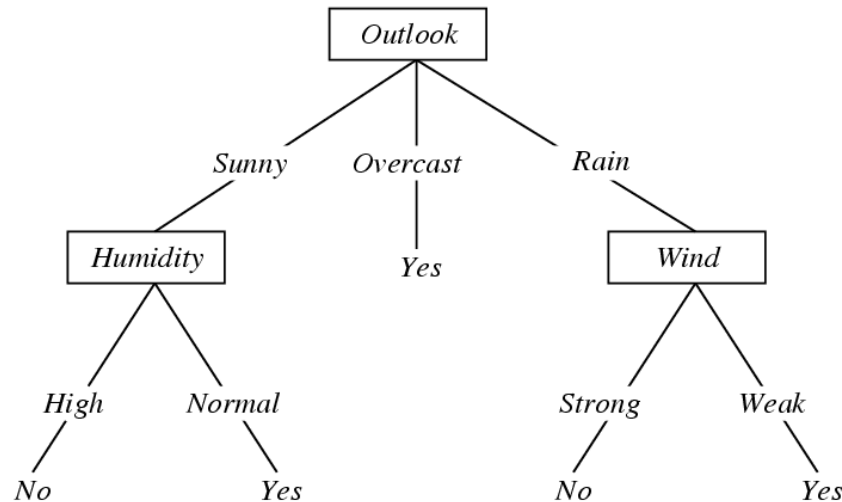
Day	Outlook	Temperature	Humidity	Wind	Play Tennis	
D1	Sunny	Hot	High	Weak	No	
D2	Sunny	Hot	High	Strong	No	
D3	Overcast	Hot	High	Weak	Yes	
D4	Rain	Mild	High	Weak	Yes	label
D5	Rain	Cool	Normal	Weak	Yes	
D6	Rain	Cool	Normal	Strong	No	
D7	Overcast	Cool	Normal	Strong	Yes	
D8	Sunny	Mild	High	Weak	No	
D9	Sunny	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	

# Supervised Classification: Decision Tree Learning

- Each internal node: test one (discrete-valued) attribute  $X_i$
- Each branch from a node: corresponds to one possible values for  $X_i$
- Each leaf node: predict  $Y$  (or  $P(Y=1|x \in \text{leaf})$ )

Example: A Decision tree for

$f: \langle \text{Outlook, Temperature, Humidity, Wind} \rangle \rightarrow \text{PlayTennis?}$



Day	Outlook	Temperature	Humidity	Wind	Play Tennis
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

E.g.,  $x=(\text{Outlook=sunny, Temperature=Hot, Humidity=Normal, Wind=High}), f(x)=\text{Yes}.$



# Supervised Classification: Problem Setting

**Input:** Training labeled examples  $\{(x^{(i)}, y^{(i)})\}$  of unknown target function  $f$

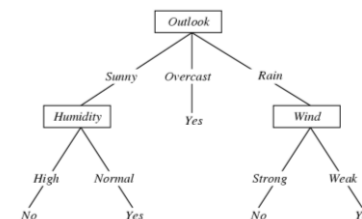
- Examples described by their values on some set of **features** or **attributes**

Day	Outlook	Temperature	Humidity	Wind	Play Tennis
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

- E.g. 4 attributes: *Humidity, Wind, Outlook, Temp*
  - e.g.,  $\langle \text{Humidity}=\text{High}, \text{Wind}=\text{weak}, \text{Outlook}=\text{rain}, \text{Temp}=\text{Mild} \rangle$
- Set of possible instances  $X$  (a.k.a instance space)
- Unknown target function  $f: X \rightarrow Y$ 
  - e.g.,  $Y=\{0,1\}$  label space
  - e.g., 1 if we play tennis on this day, else 0

**Output:** Hypothesis  $h \in H$  that (best) approximates target function  $f$

- Set of function hypotheses  $H=\{ h \mid h : X \rightarrow Y \}$ 
  - each hypothesis  $h$  is a decision tree





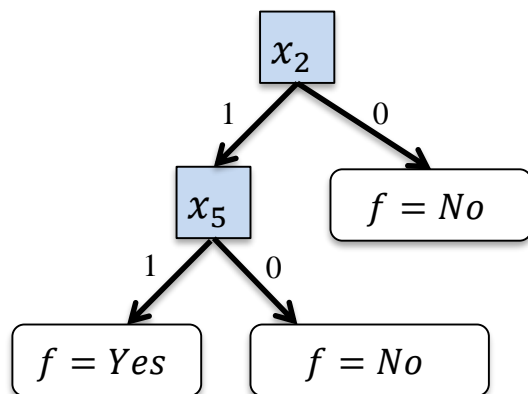
# Supervised Classification: Decision Trees

Suppose  $X = \langle x_1, \dots, x_n \rangle$

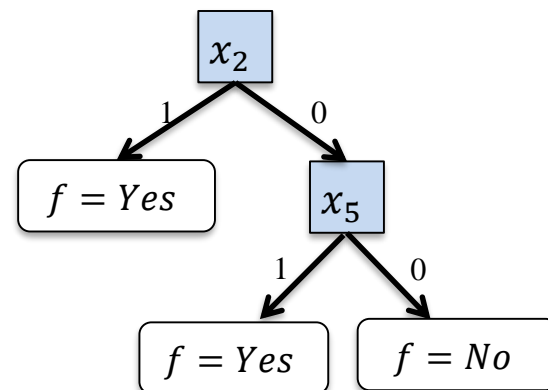
where  $x_i$  are boolean-valued variables

How would you represent the following as DTs?

$$f(x) = x_2 \text{ AND } x_5 ?$$



$$f(x) = x_2 \text{ OR } x_5$$



Hwk: How would you represent  $X_2 X_5 \vee X_3 X_4 (\neg X_1)$  ?

# Supervised Classification: Problem Setting

**Input:** Training labeled examples  $\{(x^{(i)}, y^{(i)})\}$  of unknown target function  $f$

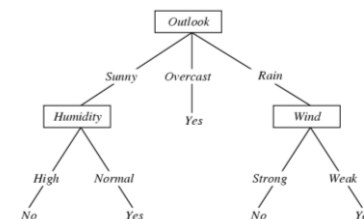
- Examples described by their values on some set of **features** or **attributes**

Day	Outlook	Temperature	Humidity	Wind	Play Tennis
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

- E.g. 4 attributes: *Humidity, Wind, Outlook, Temp*
  - e.g.,  $\langle \text{Humidity}=\text{High}, \text{Wind}=\text{weak}, \text{Outlook}=\text{rain}, \text{Temp}=\text{Mild} \rangle$
- Set of possible instances  $X$  (a.k.a instance space)
- Unknown target function  $f: X \rightarrow Y$ 
  - e.g.,  $Y=\{0,1\}$  label space
  - e.g., 1 if we play tennis on this day, else 0

**Output:** Hypothesis  $h \in H$  that (best) approximates target function  $f$

- Set of function hypotheses  $H=\{ h \mid h : X \rightarrow Y \}$ 
  - each hypothesis  $h$  is a decision tree



# Core Aspects in Decision Tree & Supervised Learning

How to automatically find a good hypothesis for training data?

- This is an **algorithmic** question, the main topic of computer science

When do we generalize and do well on unseen data?

- **Learning theory** quantifies ability to *generalize* as a function of the amount of training data and the hypothesis space
- **Occam's razor:** use the *simplest* hypothesis consistent with data!

Fewer short hypotheses than long ones

- a short hypothesis that fits the data is less likely to be a statistical coincidence
- highly probable that a sufficiently complex hypothesis will fit the data

# Core Aspects in Decision Tree & Supervised Learning

How to automatically find a good hypothesis for training data?

- This is an **algorithmic** question, the main topic of computer science

When do we generalize and do well on unseen data?

- **Occam's razor:** use the *simplest* hypothesis consistent with data!
- Decision trees: if we were able to find a **small decision tree** that explains data well, then good generalization guarantees.
  - NP-hard [Hyafil-Rivest'76]: unlikely to have a poly time algorithm
- Very nice practical heuristics; top down algorithms, e.g, ID3



# Top-Down Induction of Decision Trees

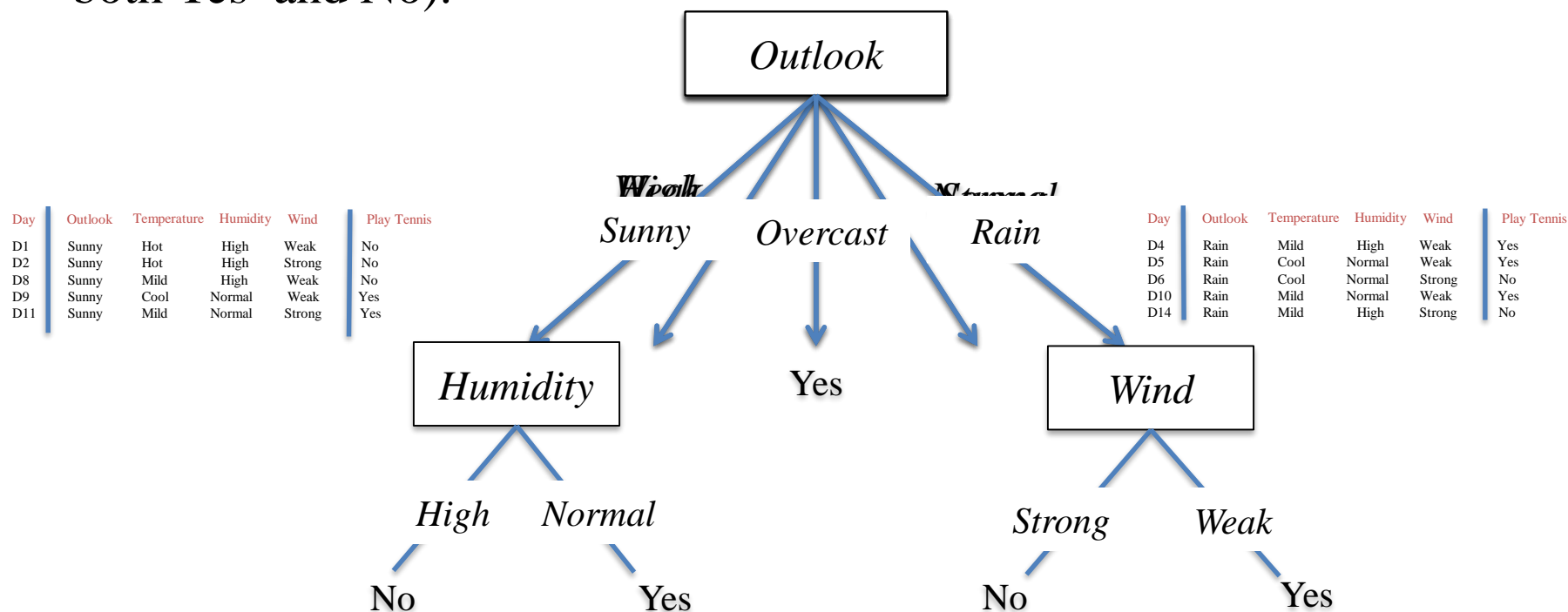
[ID3, C4.5, Quinlan]

ID3: Natural greedy approach to growing a decision tree top-down (from the root to the leaves by repeatedly replacing an existing leaf with an internal node.).

Algorithm:

- Pick “best” attribute to split at the root based on training data.
- Recurse on children that are impure (e.g, have both Yes and No).

Day	Outlook	Temperature	Humidity	Wind	Play Tennis
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



# Top-Down Induction of Decision Trees

[ID3, C4.5, Quinlan]

ID3: Natural greedy approaches where we grow the tree from the root to the leaves by repeatedly replacing an existing leaf with an internal node.

*node* = Root

Main loop:

1.  $A \leftarrow$  the “best” decision attribute for next *node*
2. Assign  $A$  as decision attribute for *node*
3. For each value of  $A$ , create new descendent of *node*
4. Sort training examples to leaf nodes
5. If training examples perfectly classified, Then STOP,  
Else iterate over new leaf nodes.



Key question: Which attribute is best?

# Top-Down Induction of Decision Trees

[ID3, C4.5, Quinlan]

ID3: Natural greedy approach to growing a decision tree top-down.

Algorithm:

- Pick “best” attribute to split at the root based on training data.
- Recurse on children that are impure (e., have both Yes and No).

Day	Outlook	Temperature	Humidity	Wind	Play Tennis
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

Key question: Which attribute is best?



ID3 uses a statistical measure called **information gain** (how well a given attribute separates the training examples according to the target classification)

# Top-Down Induction of Decision Trees

[ID3, C4.5, Quinlan]



Which attribute to select?

ID3: The attribute with highest information gain.

a statistical measure of how well a given attribute separates the training examples according to the target classification

**Information Gain** of **A** is the expected reduction in entropy of target variable **Y** for data sample **S**, due to sorting on variable **A**

$$Gain(S, A) = H_S(Y) - H_S(Y|A)$$

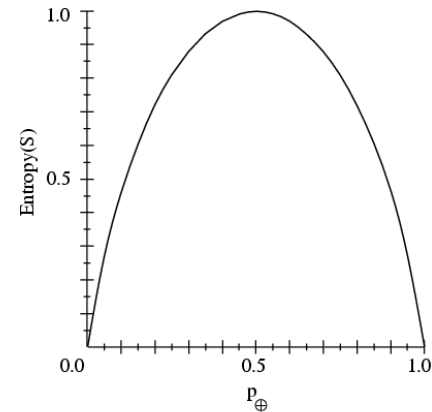
Entropy information theoretic measure that characterizes the impurity of a labeled set  $S$ .



# Sample Entropy of a Labeled Dataset

- $S$  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  $S$ .

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

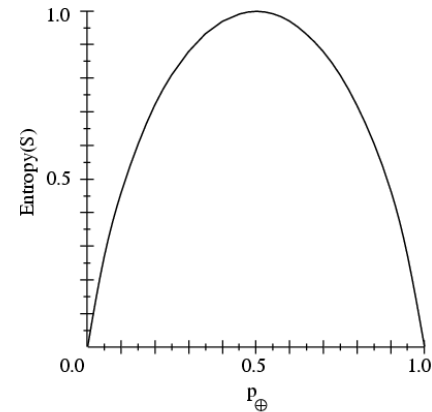


- E.g., if all negative, then entropy=0. If all positive, then entropy=0.
- If 50/50 positive and negative then entropy=1.
- If 14 examples with 9 positive and 5 negative, then entropy=.940

# Sample Entropy of a Labeled Dataset

- $S$  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  $S$ .

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



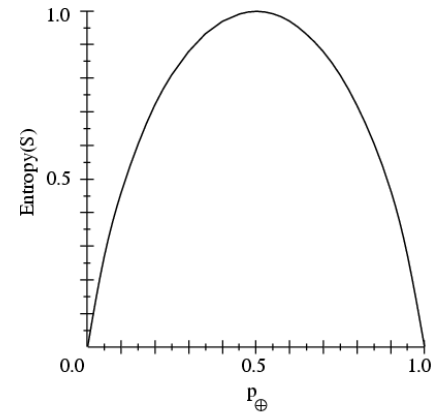
Interpretation from information theory: expected number of bits needed to encode label of a randomly drawn example in  $S$ .

- If  $S$  is all positive, receiver knows label will be positive, don't need any bits.
- If  $S$  is 50/50 then need 1 bit.
- If  $S$  is 80/20, then in a long sequence of messages, can code with less than 1 bit on average (assigning shorter codes to positive examples and longer codes to negative examples).

# Sample Entropy of a Labeled Dataset

- $S$  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  $S$ .

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



If labels not Boolean, then  $H(S) = \sum_{i \in Y} -p_i \log_2 p_i$

E.g., if  $c$  classes, all equally likely, then  $H(S) = \log_2 c$

# Information Gain

Given the definition of entropy, can define a measure of effectiveness of attribute in classifying training data:

**Information Gain** of **A** is the expected reduction in entropy of target variable **Y** for data sample **S**, due to sorting on variable **A**

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

entropy of original collection

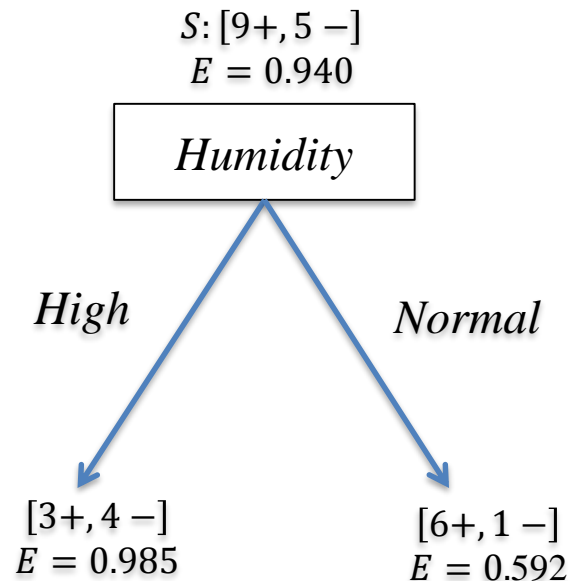
Expected entropy after S is partitioned using attribute A

sum of entropies of subsets  $S_v$  weighted by the fraction of examples that belong to  $S_v$ .

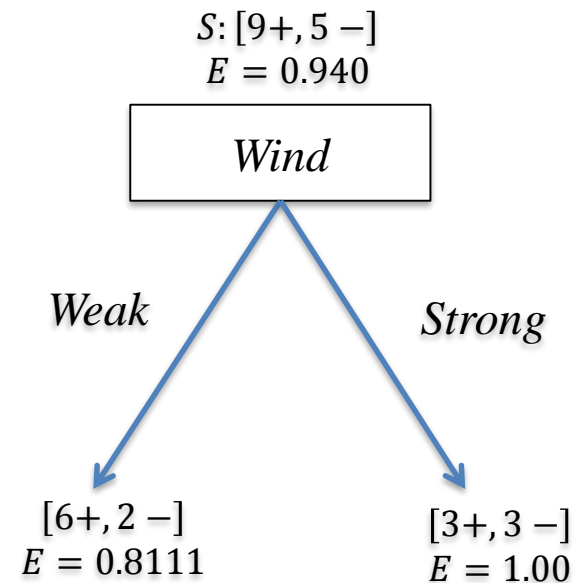
# Selecting the Next Attribute

Which attribute is the best classifier?

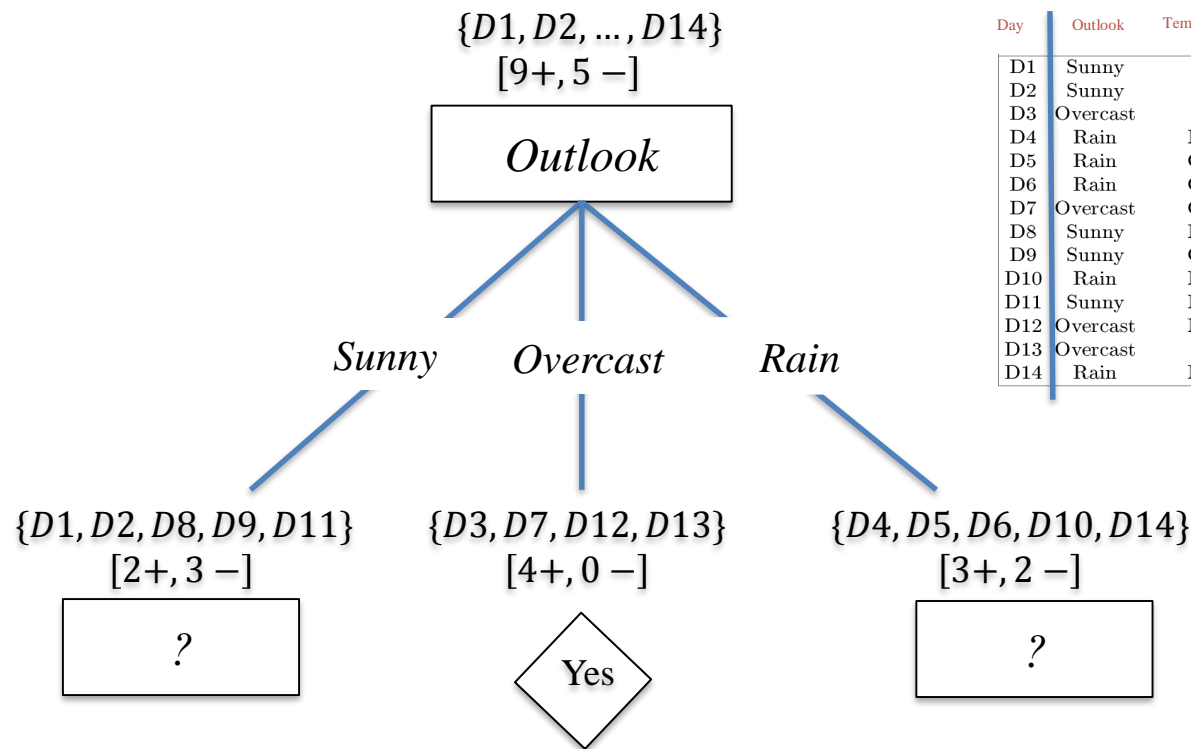
Day	Outlook	Temperature	Humidity	Wind	Play Tennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	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



$$\begin{aligned}
 \text{Gain}(S, \text{Humidity}) &= .940 - \left(\frac{7}{14}\right) \cdot .985 - \left(\frac{7}{14}\right) \cdot .592 \\
 &= .151
 \end{aligned}$$



$$\begin{aligned}
 \text{Gain}(S, \text{Wind}) &= .940 - \left(\frac{8}{14}\right) \cdot .8111 - \left(\frac{6}{14}\right) \cdot 1.0 \\
 &= .048
 \end{aligned}$$



Day	Outlook	Temperature	Humidity	Wind	Play Tennis
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

*Which attribute should be tested here?*

$$s_{\text{sunny}} = \{D1, D2, D8, D9, D11\}$$

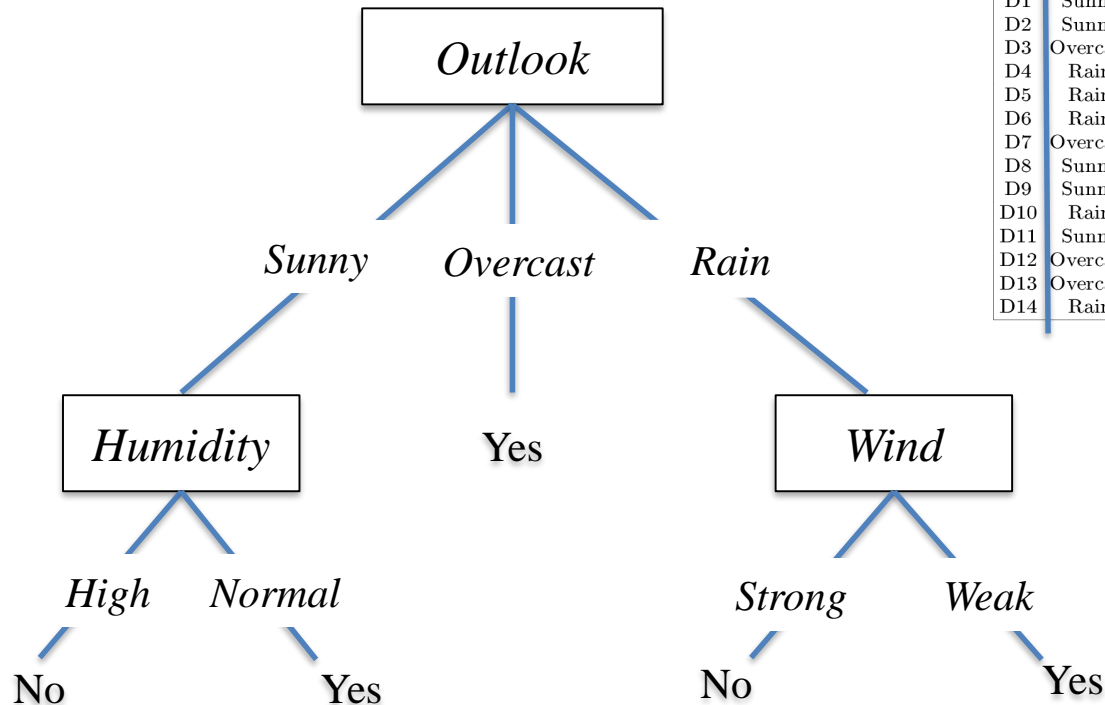
$$\text{Gain}(s_{\text{sunny}}, \text{Humidity}) = .970 - \left(\frac{3}{5}\right) 0.0 - \left(\frac{2}{5}\right) 0.0 = .970$$

$$\text{Gain}(s_{\text{sunny}}, \text{Temperature}) = .970 - \left(\frac{2}{5}\right) 0.0 - \left(\frac{2}{5}\right) 1.0 - \left(\frac{1}{5}\right) 0.0 = .570$$

$$\text{Gain}(s_{\text{sunny}}, \text{Wind}) = .970 - \left(\frac{2}{5}\right) 1.0 - \left(\frac{3}{5}\right) .918 = .019$$

# Final Decision Tree for

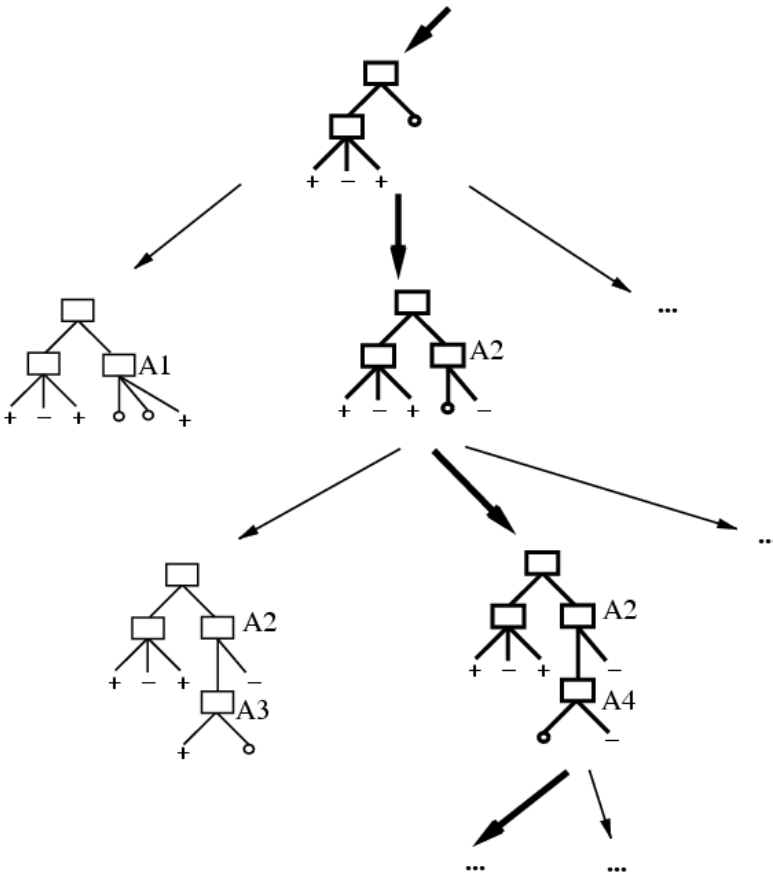
$f: \langle \text{Outlook, Temperature, Humidity, Wind} \rangle \rightarrow \text{PlayTennis?}$



Day	Outlook	Temperature	Humidity	Wind	Play Tennis
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

# Properties of ID3

- ID3 performs heuristic search through space of decision trees
- It stops at smallest acceptable tree. Why?



Occam's razor: prefer the simplest hypothesis that fits the data

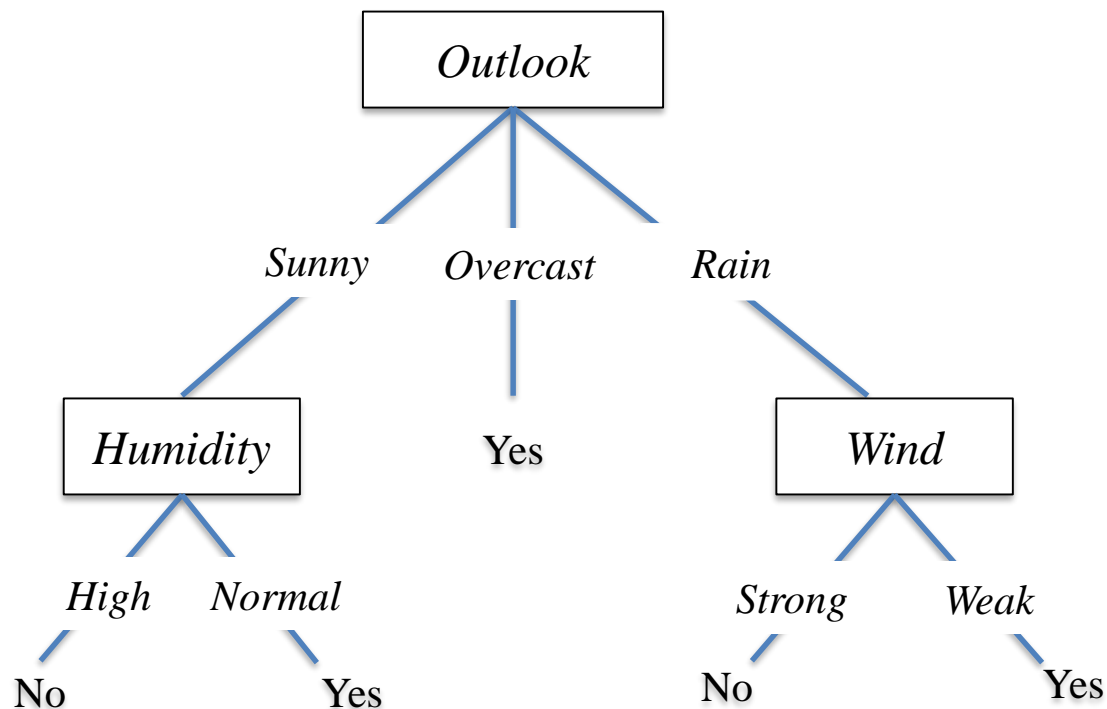


# Overfitting in Decision Trees

Consider adding noisy training example #15:

*Sunny, Hot, Normal, Strong, PlayTennis = No*

What effect on earlier tree?



# Properties of ID3

Overfitting could occur because of noisy data and because ID3 is not guaranteed to output a small hypothesis even if one exists.

Consider a hypothesis  $h$  and its

- Error rate over training data:  $error_{train}(h)$
- True error rate over all data:  $error_{true}(h)$

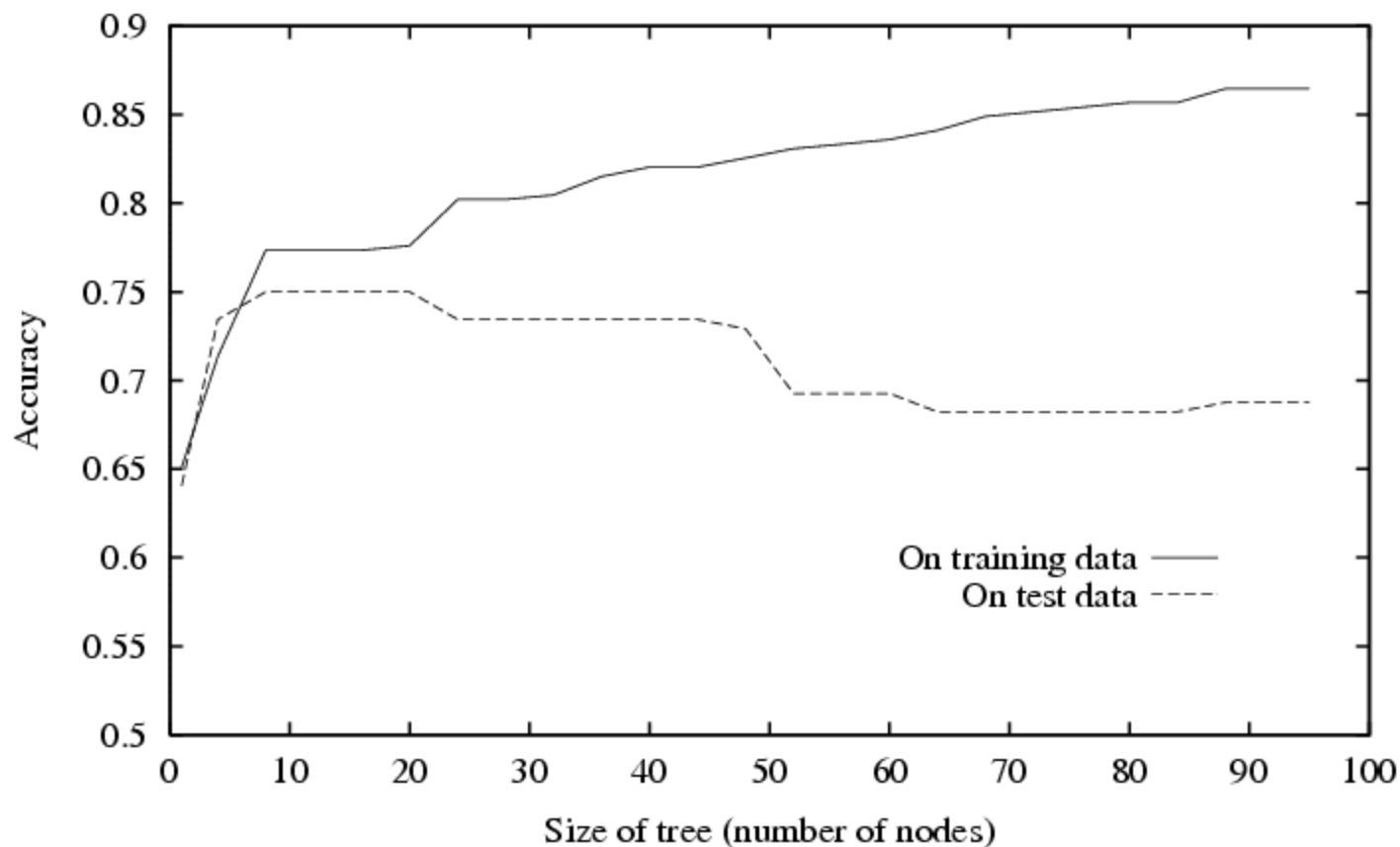
We say  $h$  overfits the training data if

$$error_{true}(h) > error_{train}(h)$$

$$\text{Amount of overfitting} = error_{true}(h) - error_{train}(h)$$

# Overfitting in Decision Tree Learning

---



# Avoiding Overfitting


How can we avoid overfitting?

- Stop growing when data split not statistically significant
- Grow full tree, then post-prune

# Decision Tree Learning, Further Guarantees on ID3

"On the Boosting Ability Of Top-Down Decision Tree Learning Algorithms". M Kearns, Y. Mansour, Journal of Computer and System Sciences 1999

# Top Down Decision Trees Algorithms

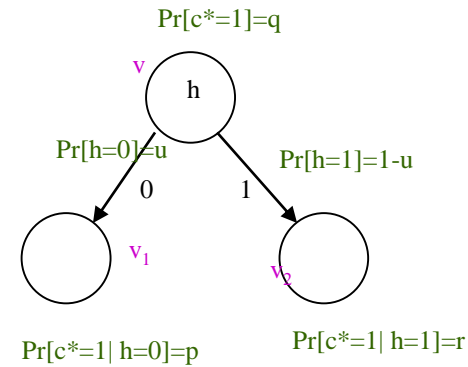
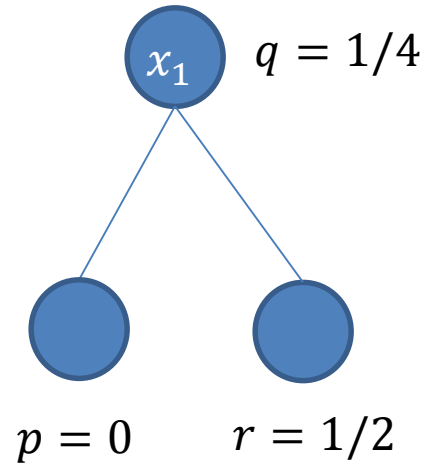
- Decision trees: if we were able to find a small decision tree consistent with the data, then good generalization guarantees.
  - NP-hard [Hyafil-Rivest'76] 
- Very nice practical heuristics; top down algorithms, e.g, ID3
- Natural greedy approaches where we grow the tree from the root to the leaves by repeatedly replacing an existing leaf with an internal node.
  - Key point: splitting criterion.
  - ID3: split the leaf that decreases the entropy the most.
- Why not split according to error rate --- this is what we care about after all?
  - There are examples where we can get stuck in local minima!!!



# Entropy as a better splitting measure

$$f(x) = x_1 \wedge x_2$$

0	0	0	—
0	0	1	—
0	1	0	—
0	1	1	—
1	0	0	—
1	0	1	—
1	1	0	+
1	1	1	+



Initial error rate is  $1/4$  (25% positive, 75% negative)

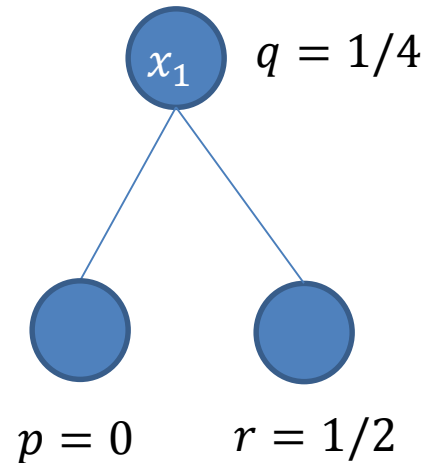
Error rate after split is  $0.5 * 0 + 0.5 * 0.5 = 1/4$   
(left leaf is 100% negative; right leaf is 50/50)

Overall error doesn't decrease!

# Entropy as a better splitting measure

$$f(x) = x_1 \wedge x_2$$

0	0	0	—
0	0	1	—
0	1	0	—
0	1	1	—
1	0	0	—
1	0	1	—
1	1	0	+
1	1	1	+




Initial entropy is  $\frac{1}{4}(\log_2 4) + \frac{3}{4}(\log_2 \frac{4}{3}) = 0.81$

Entropy after split is  $\frac{1}{2} * 0 + \frac{1}{2} * 1 = 0.5$

Entropy decreases!

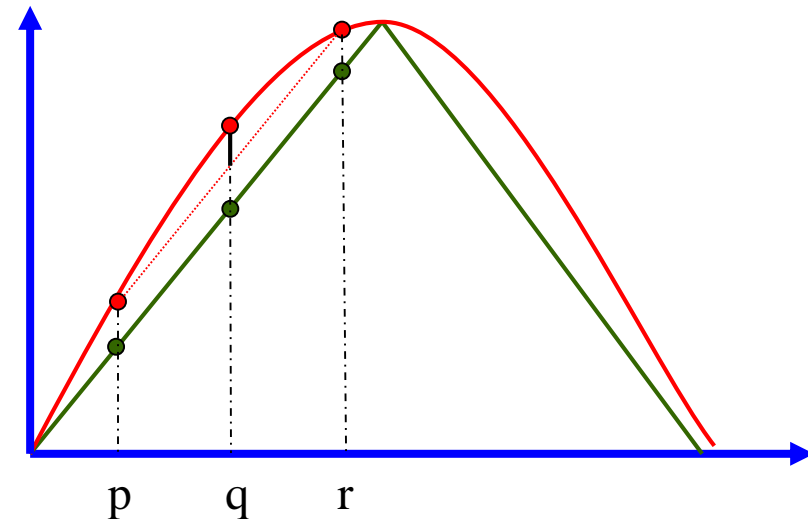
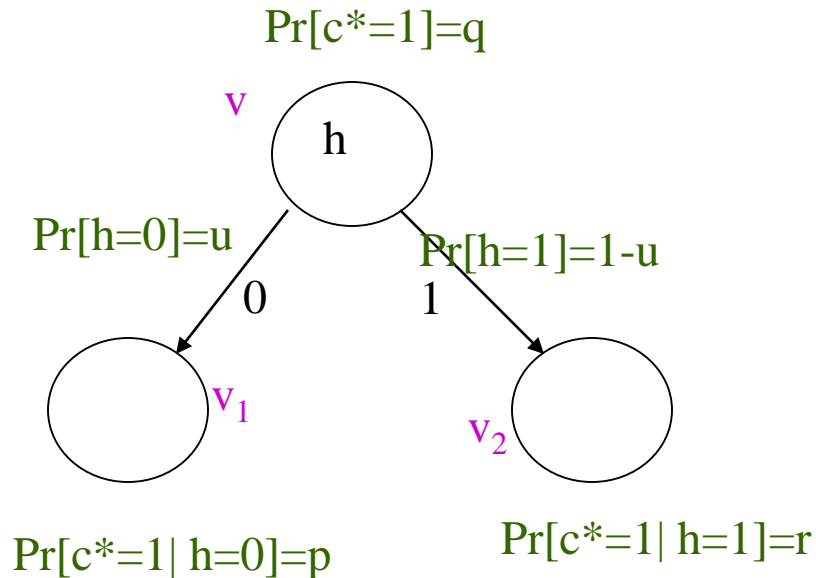


# Top Down Decision Trees Algorithms

- Natural greedy approaches where we grow the tree from the root to the leaves by repeatedly replacing an existing leaf with an internal node.
  - Key point: **splitting criterion**.
  - ID3: split the leaf that decreases the entropy the most.
-  Why not split according to error rate --- this is what we care about after all?
  - There are examples where you can get stuck!!!
- [Kearns-Mansour'96]: if measure of progress is entropy, we can always guarantees success under some formal relationships between the class of splits and the target (the class of splits can weakly approximate the target function).
  - Provides a way to think about the effectiveness of various top down algos.

# Top Down Decision Trees Algorithms

- Key: strong concavity of the splitting criterion



- $q = up + (1-u)r$ . Want to lower bound:  $G(q) - [uG(p) + (1-u)G(r)]$
- If:  $G(q) = \min(q, 1-q)$  (error rate), then  $G(q) = uG(p) + (1-u)G(r)$
- If:  $G(q) = H(q)$  (entropy), then  $G(q) - [uG(p) + (1-u)G(r)] > 0$  if  $r-p > 0$  and  $u \neq 1, u \neq 0$  (this happens under the weak learning assumption)

# Key Issues in Machine Learning

- How can we gauge the accuracy of a hypothesis on unseen data?
  - **Occam's razor**: use the *simplest* hypothesis consistent with data!  
This will help us avoid overfitting.
  - **Learning theory** will help us quantify our ability to **generalize** as a function of the amount of training data and the hypothesis space
- How do we find the best hypothesis?
  - This is an **algorithmic** question, the main topic of computer science
- How do we choose a hypothesis space?
  - Often we use **prior knowledge** to guide this choice
- How to model applications as machine learning problems?  
(engineering challenge)