Machine Learning 10-601 B, Spring 2016

Introduction, Admin, Course Overview

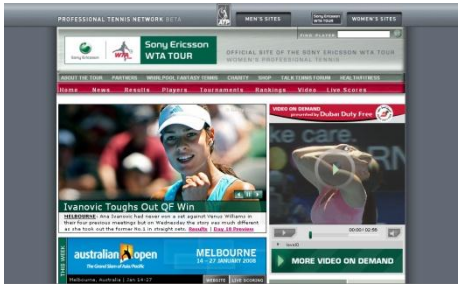
Lecture 1, Jan. 11th 2016

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

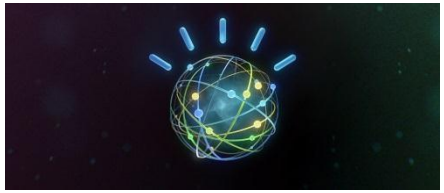
Computational Advertising

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)



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: 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.

What is Machine Learning?

Examples of important machine learning paradigms.

Supervised Classification

from data to discrete classes

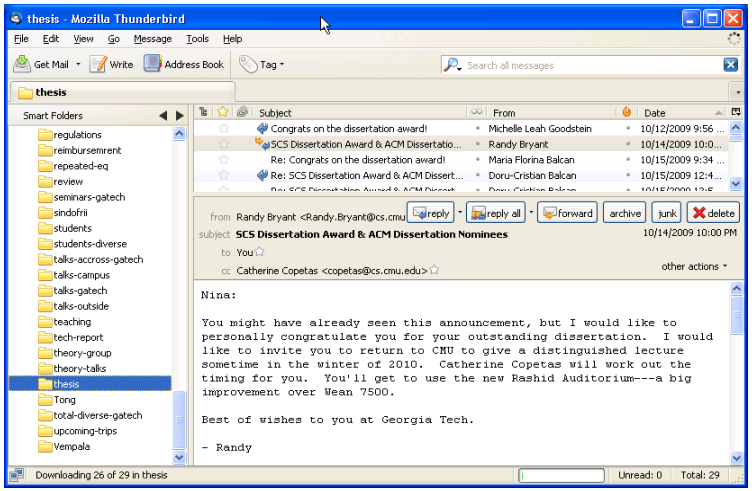
Supervised Classification. Example: Spam Detection

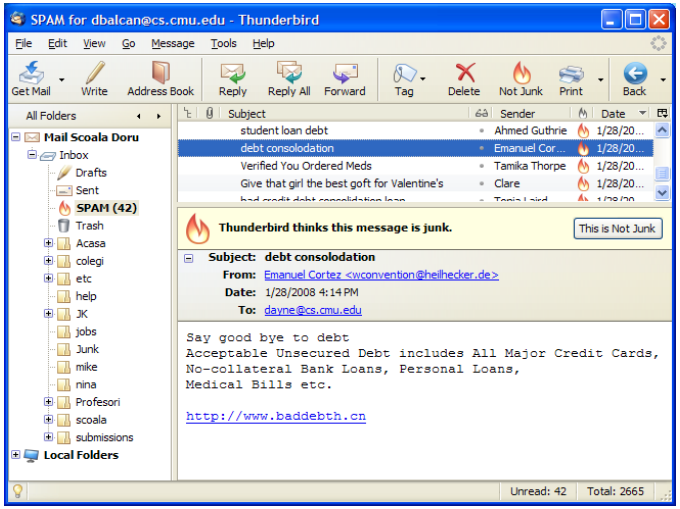
Decide which emails are spam and which are important.

Supervised classification

Not spam

spam





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

Supervised Classification. Example: Spam Detection

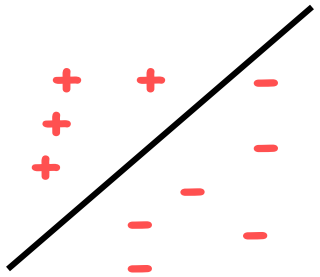
Represent each message by features. (e.g., keywords, spelling, etc.)

	"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)

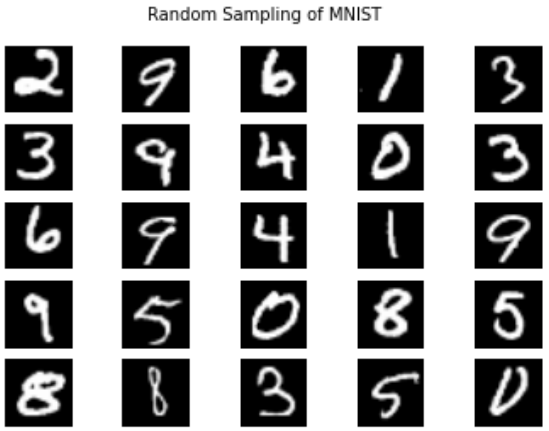
Predict SPAM if 2money + 3pills –5 known > 0



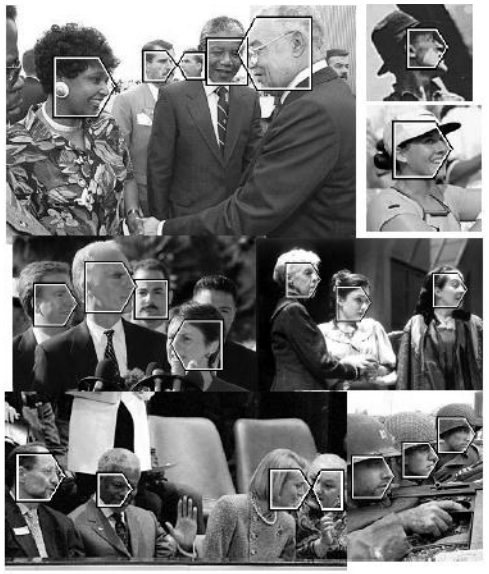
Linearly separable

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

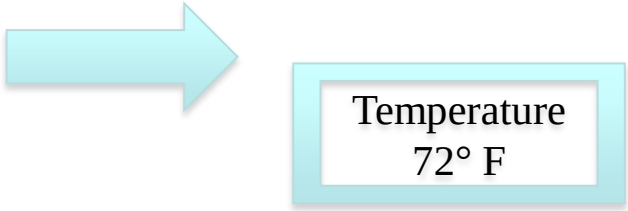
- Medicine:
  - diagnose a disease
    - input: from symptoms, lab measurements, test results, DNA tests, …
    - output: one of set of possible diseases, or “none of the above”
    - examples: audiology, thyroid cancer, diabetes, …
      - or: response to chemo drug X
      - or: will patient be re-admitted 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

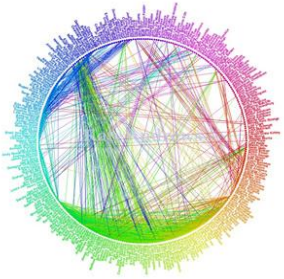


Predict the temperature at any given location

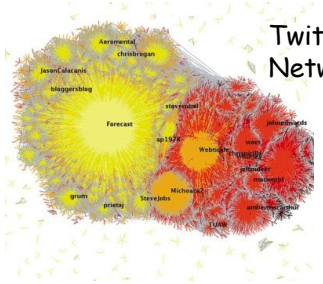
Other Machine Learning Paradigm

Clustering: discovering structure in data (only unlabeled data)

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



Facebook network



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

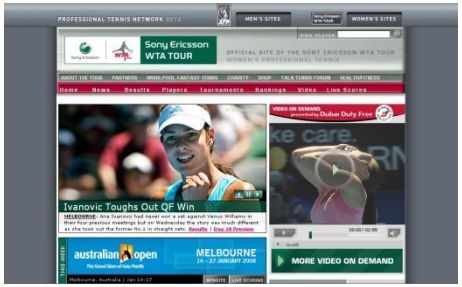
Collaborative Filtering (Matrix Completion), …

Machine Learning

ML is the preferred method to solve problems in many areas:

Vision/Image Classification

Document Categorization



Speech Recognition

Protein Classification

Spam Detection

Branch Prediction

Fraud Detection

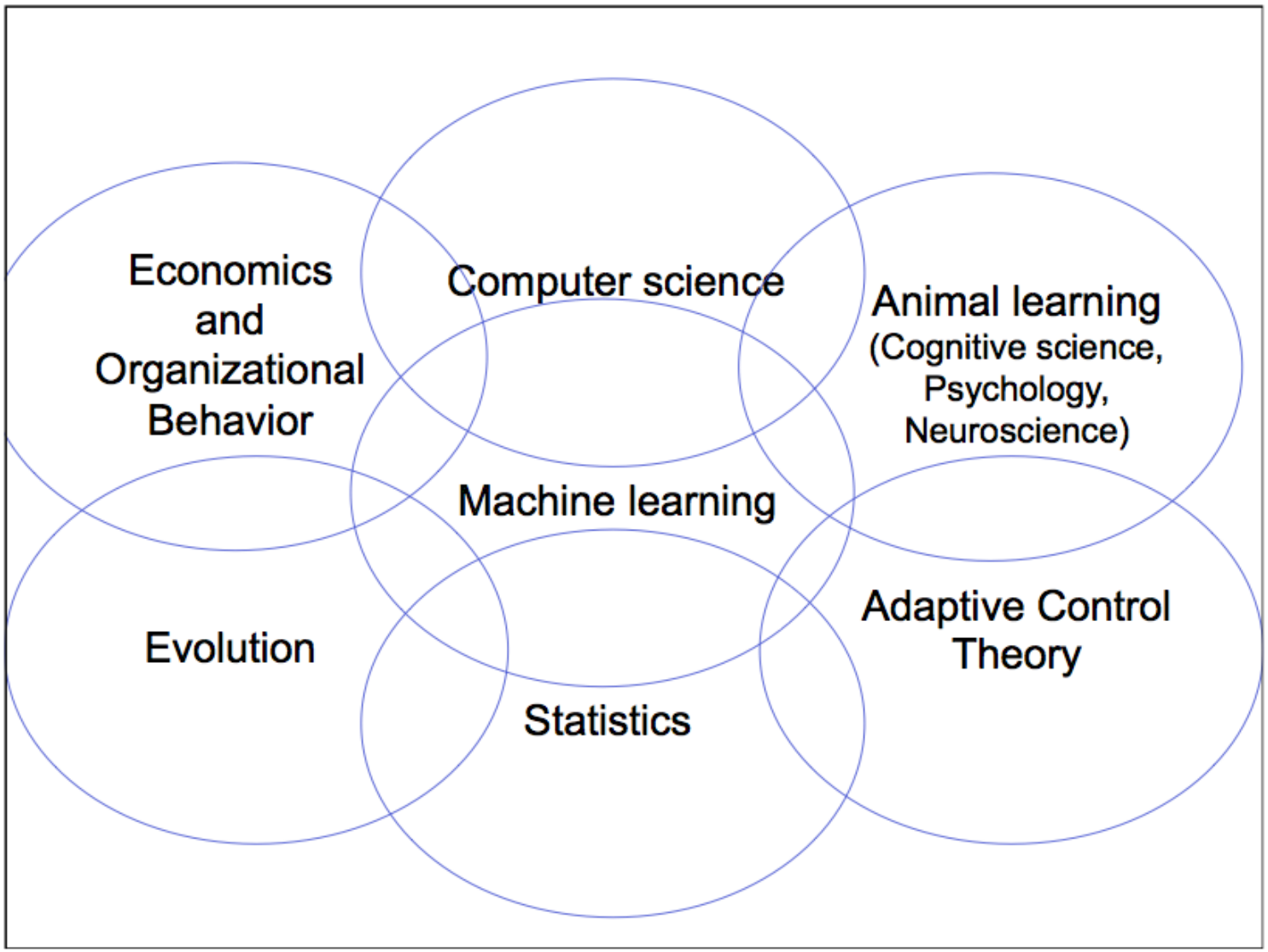
Natural Language Processing

Playing Games

Computational Advertising

Robot Control

Many communities relate to ML



Admin, Logistics, Grading

Brief Overview

- Meeting Time: Mon, Wed, GHC 4401, 10:30 – 11:50
- Course Staff:
  - Instructors:
    - Maria Florina (Nina) Balcan (ninamf@cs.cmu.edu)
    - William Cohen (wcohen@cs.cmu.edu)
  - TAs:
    - Travis Dick (tdick@cs.cmu.edu)
    - William Herlands (herlands@cmu.edu)
    - Renato Negrinho (negrinho@cs.cmu.edu)
    - María De Arteaga (mdeartea@andrew.cmu.edu)
    - Tianshu Ren (tren@andrew.cmu.edu)
    - Zichao Yang (zichaoy@andrew.cmu.edu)
    - Han Zhao (han.zhao@cs.cmu.edu)

Brief Overview

- Course Website/the class wiki

  http://curtis.ml.cmu.edu/w/courses/index.php/Machine_Learning_10-601_in_Spring_2016

- See website for:
  - Syllabus details
    - All the lecture slides and homeworks
    - Additional useful resources.
  - Office hours
  - Recitation sessions (2 per week)
  - Grading policy
  - Honesty policy
  - Late homework policy
  - Piazza pointers
- Will use Piazza for discussions.
- For personal questions (e.g., extensions or prerequisites), use the mailing list: 10-601b-instructors@lists.andrew.cmu.edu.

Prerequisites. What do you need to know now?

- You should know how to do math and how to program:
  - 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 mostly in Matlab and/or Octave (play with that now if you want)
    - All CMU students can download Matlab free of charge from CMU software website. Octave is open-source software.
- We may review these things but we will **not** teach them
- Use HWK # 1 (out on Wed) as a self-assessment.

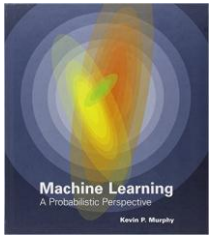
Source Materials

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

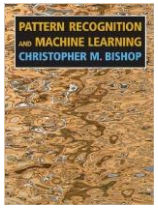
Useful textbooks:

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



Grading

- 50% for homeworks. There are 6 or 7 and you can drop 1.
- 20% for midterm
- 20% for final
- 10% for class participation.
    - Piazza polls in class: bring a laptop or a phone

- 6 or 7 weekly (mostly) 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 (see syllabus)

- 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

- This policy is stolen from Roni Rosenfeld.

- Just so you know: we will fail students, and CMU will expel them.

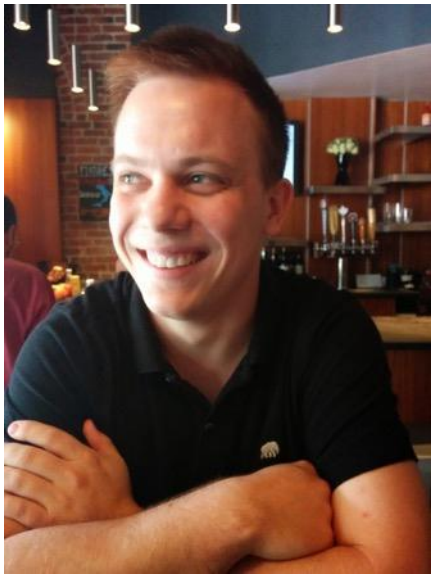
Who's teaching 10-601B?

- William Cohen: an old AI/ML guy:
  - First paper at ICML in 1988
  - Organized ICML in 1994, 2006, 2008
  - Past President IMLS
- Most cited work:
  - Representation and learning
    - Scalable rule learning
    - Similarity of names
  - Shallow NLP/IR related tasks:
    - Learning to rank
    - Text classification
    - NER

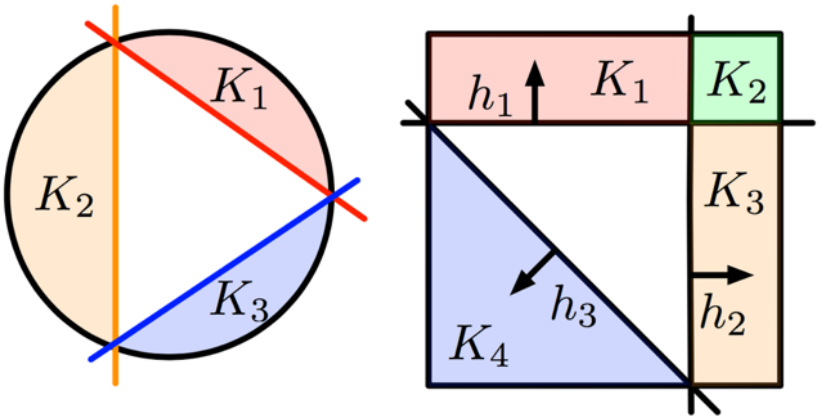
Who's TAing 10-601B?

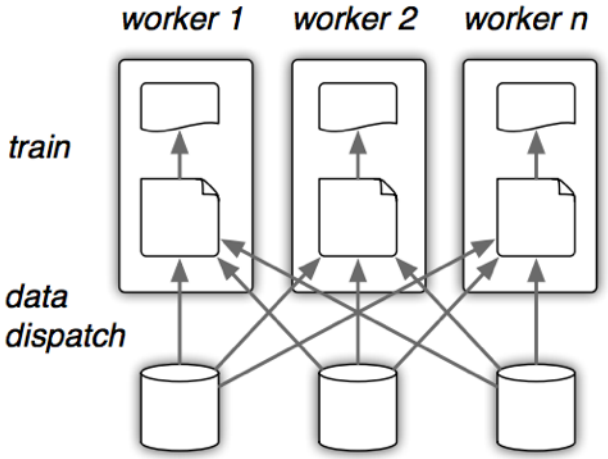
- **Maria De Arteaga**
  - Machine Learning & Public Policy PhD student
  - Member of the Auton Lab
  - Research interests:
    - Statistical machine learning, data mining, characterization models, causation.
    - Organized crime, nuclear detection, health.

Travis Dick



- Second year PhD student advised by Nina.
- Research projects include:
  - multiclass learning with unlabeled data;
  - distributed learning (i.e., using many computers to learn and predict);
  - privacy in Machine Learning.





William Herlands

- Machine Learning and Public Policy

Gaussian processes
Kernel methods
Hawkes processes
Causal inference

Counterfactual modeling
Epidemiology
Crime prediction
Cybersecurity

- Work in NIPS, AISTATS, AAAI
- Email: herlands@cmu.edu
- Office: HBH 3030

Renato Negrinho

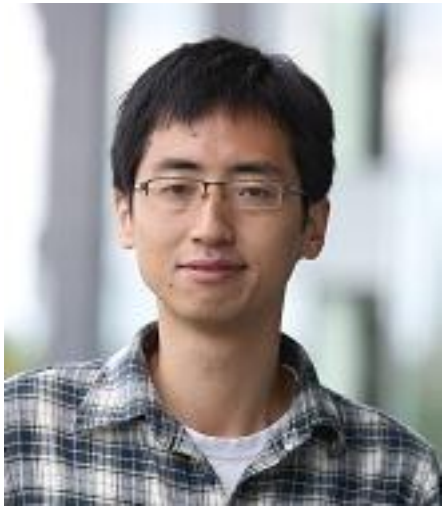
- 1st year ML PhD Student, advised by Geoff Gordon.
  - Previously, BSc+MSc (5 years total) on ECE on IST, Portugal; research on ML and NLP for 1 year; 2 internships (CMU, XRCE).
- Spectral algorithms, convex and combinatorial optimization, randomized and approximation algorithms, convex and discrete geometry; applications: NLP, CV, ML.
- At the intersection of theory and applications.
  - Start with a concrete problem; formulate it; search for connections in mathematics, computer science, optimization, ... Goal is understanding. Someone may have solved it.
  - "Mathematics is the art of giving the same name to different things" - Henri Poincaré

Tianshu Ren

- 1st-year ML Master
- Studied Statistics and Economics from University of North Carolina
- Experience in:
  - Video Summarization
  - Brain Image analysis with ML
- Interest:
  - Solving interesting questions with ML

Who's teaching 10-601B?

- Zichao Yang
  - Third year Phd student in CSD
  - Working with Alex Smola
  - Research interest: deep learning with applications to NLP and Computer Vision
  - Email: zichaoy@andrew.cmu.edu



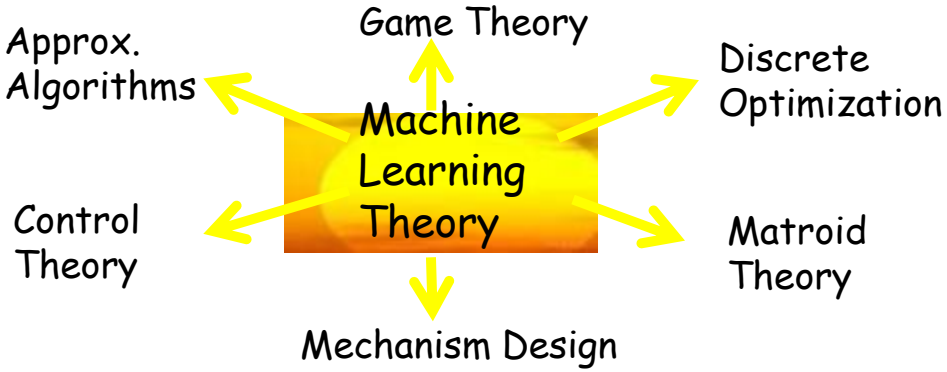
Who's TA of 10-601B?

- Han Zhao: PhD student @ MLD
  - Office: GHC 8007
  - Email: han.zhao@cs.cmu.edu
  - Research Interests:
    - Sequential Decision Making
    - Inference on PGMs

Maria-Florina Balcan: Nina

- Foundations for Modern Machine Learning
  - E.g., interactive, semi-supervised, distributed, multi-task, life-long learning

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



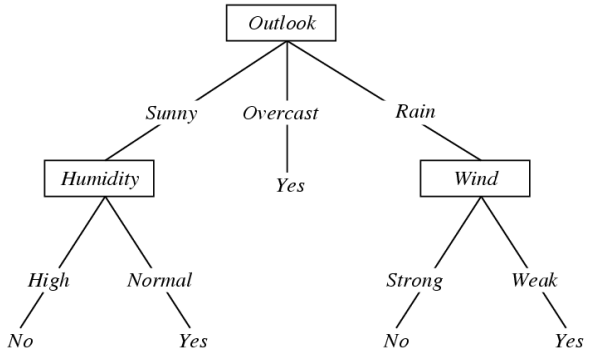
- Program Committee Chair for ICML 2016, COLT 2014

Learning Decision Trees.

Supervised Classification.

Useful Readings:

- Mitchell, Chapter 3
- Bishop, Chapter 14.4



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

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

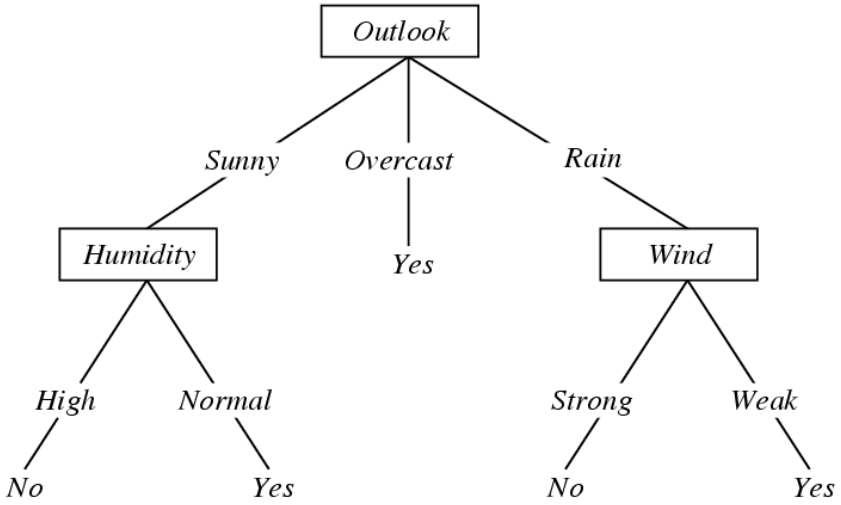
example

label

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 ∈ leaf))

Example: A Decision tree for
f: <Outlook, Temperature, Humidity, Wind> → 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=(Outlook=sunny, Temperature-Hot, Humidity=Normal,Wind=High), f(x)=Yes.

Supervised Classification: Problem Setting

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

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

- Examples described by their values on some set of features or attributes
  - E.g. 4 attributes: *Humidity, Wind, Outlook, Temp*
    - e.g., <*Humidity=High, Wind=weak, Outlook=rain, Temp=Mild*>
  - 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

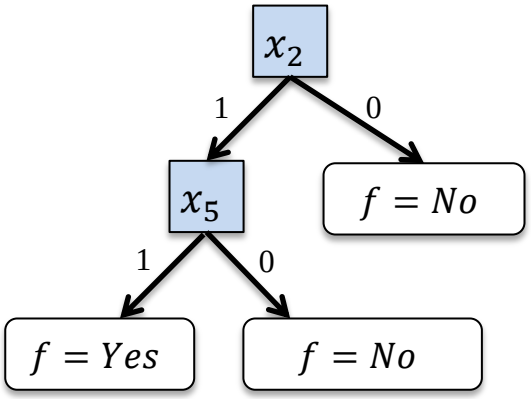
Supervised Classification: Decision Trees

Suppose $X = <x_1, \ldots x_n>$

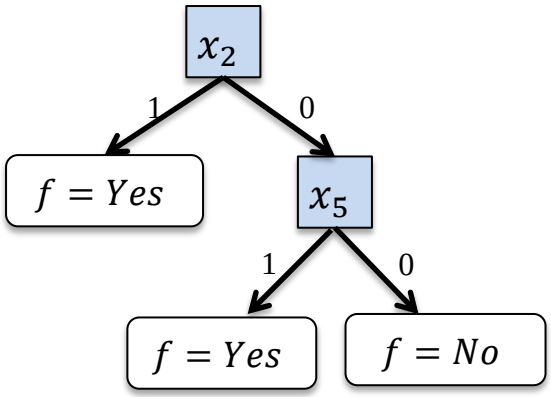
where x_i are boolean-valued variables

How would you represent the following as DTs?

$f(x) = x_2$ *AND* x_5 ?



$f(x) = x_2$ *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

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

- Examples described by their values on some set of features or attributes
  - E.g. 4 attributes: *Humidity, Wind, Outlook, Temp*
    - e.g., <*Humidity=High, Wind=weak, Outlook=rain, Temp=Mild*>
  - 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

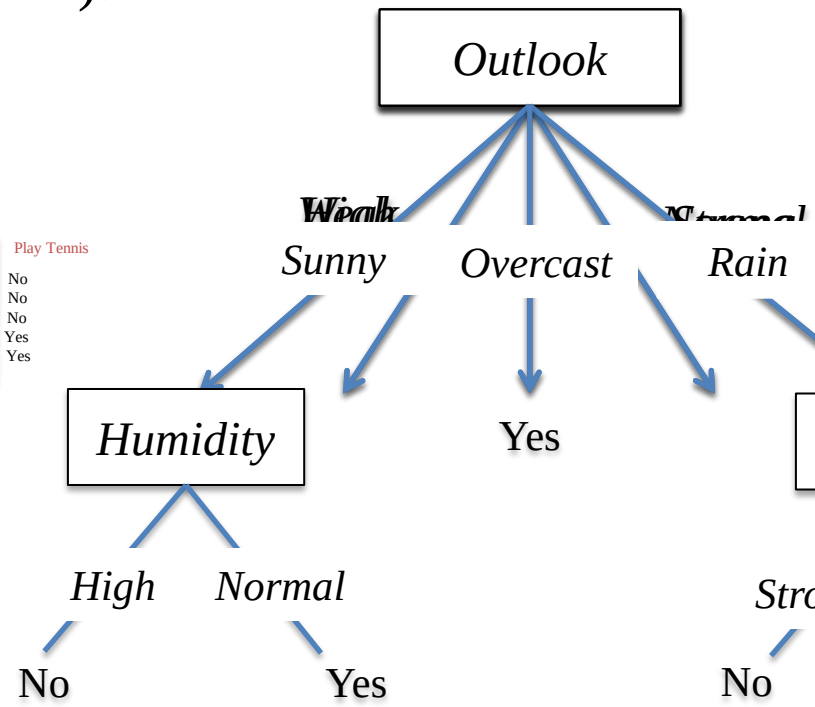
Top-Down Induction of Decision Trees [ID3, C4.5, Quinlan]

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



Day	Outlook	Temperature	Humidity	Wind	Play Tennis
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

Day	Outlook	Temperature	Humidity	Wind	Play Tennis
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D10	Rain	Mild	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

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

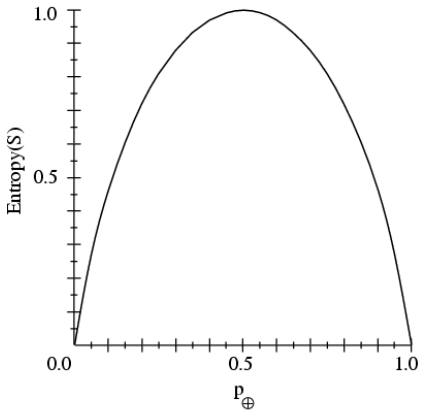
Information Gain of A is the expected reduction in entropy of target variable Y for data sample S, due to sorting on variable A

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

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.

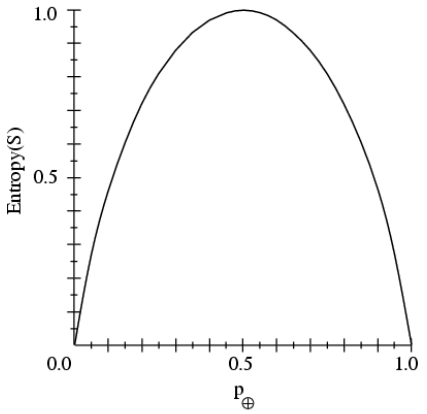


$$H(S) \equiv -p_{\oplus} \log_2 p_{\oplus} - p_{\ominus} \log_2 p_{\ominus}$$

- 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.



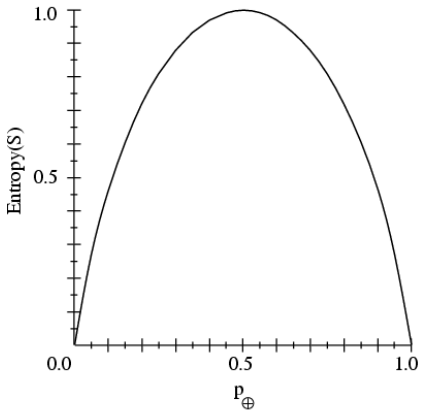
$$H(S) \equiv -p_{\oplus} \log_2 p_{\oplus} - p_{\ominus} \log_2 p_{\ominus}$$

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

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

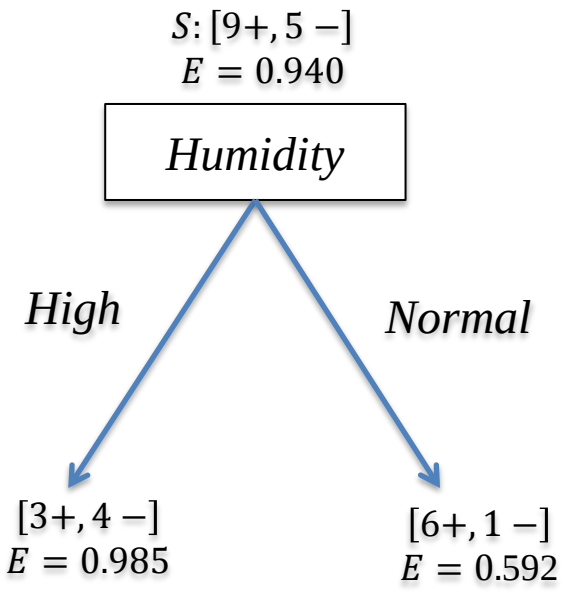
entropy of original collection

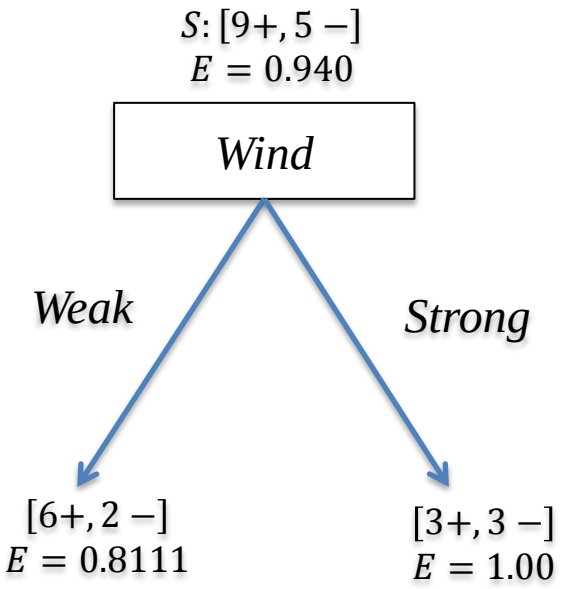
sum of entropies of subsets S_v weighted by the fraction of examples that belong to S_v.

Selecting the Next Attribute

Which attribute is the best classifier?

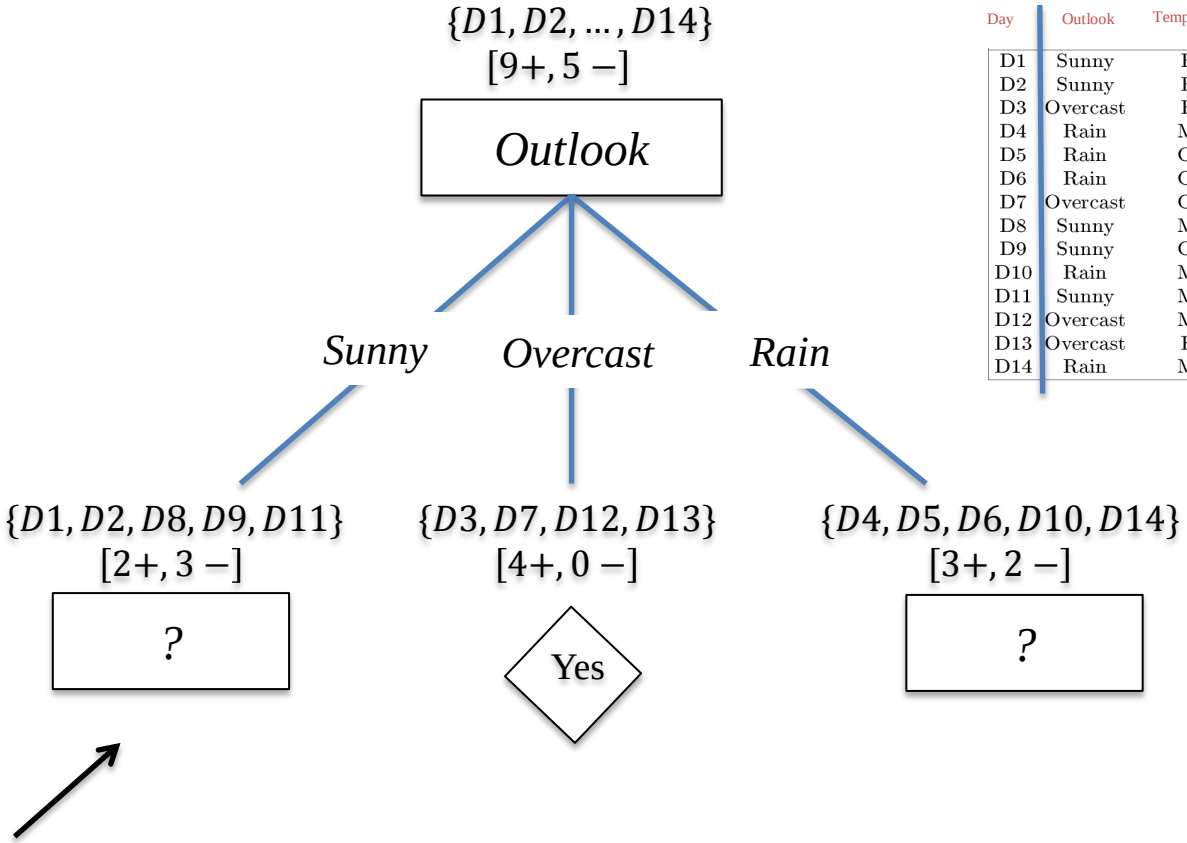
Day	Outlook	Temperature	Humidity	Wind	Play Tennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal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





$$Gain(S, Humidity) = .940 - \left(\frac{7}{14}\right).985 - \left(\frac{7}{14}\right).592 = .151$$

$$Gain(S, Wind) = .940 - \left(\frac{8}{14}\right).811 - \left(\frac{6}{14}\right)1.0 = .048$$



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?

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

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

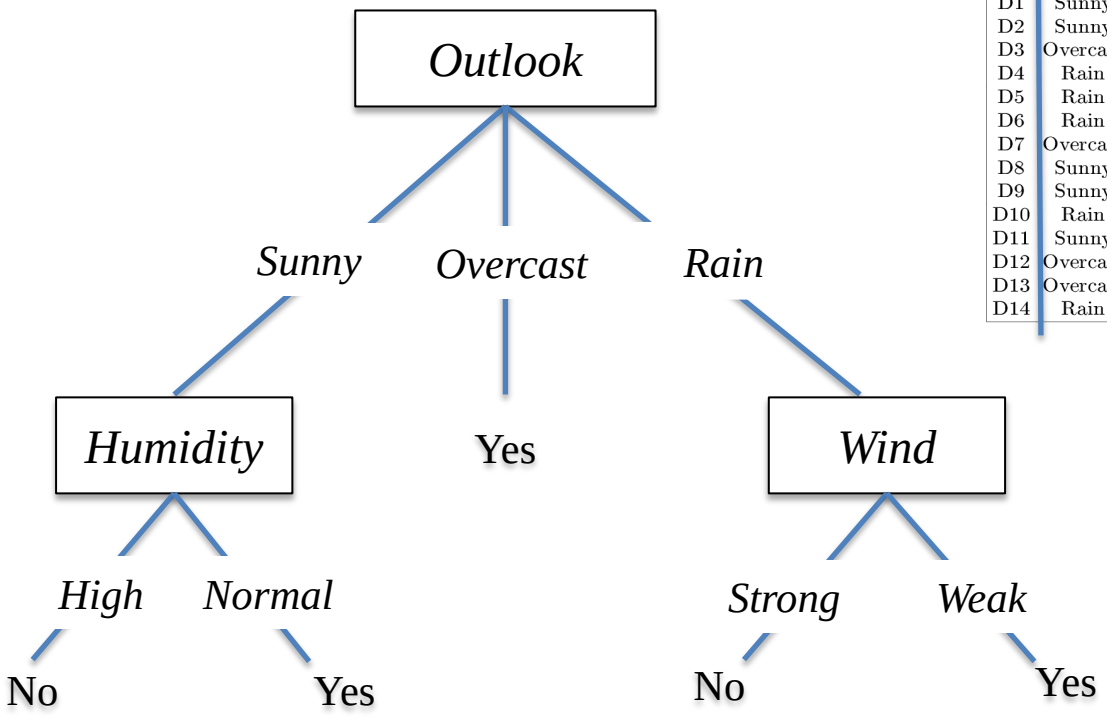
$$Gain(S_{sunny}, Temperature) = .970 - \left(\frac{2}{5}\right)0.0 - \left(\frac{2}{5}\right)1.0 - \left(\frac{1}{5}\right)0.0 = .570$$

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

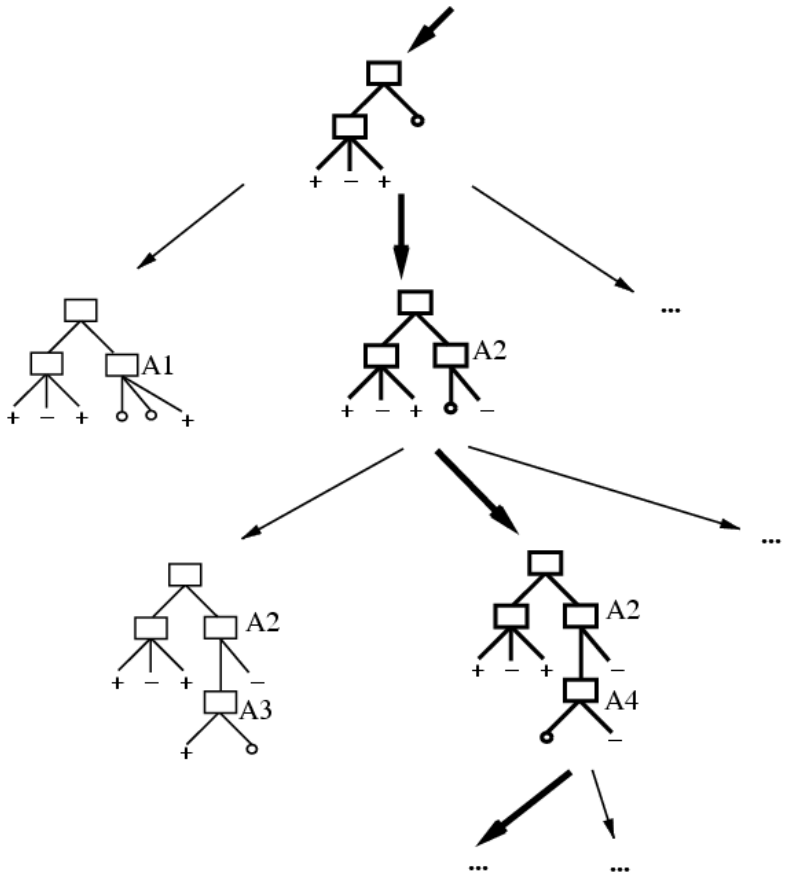
Final Decision Tree for

f: <Outlook, Temperature, Humidity, Wind> → 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?

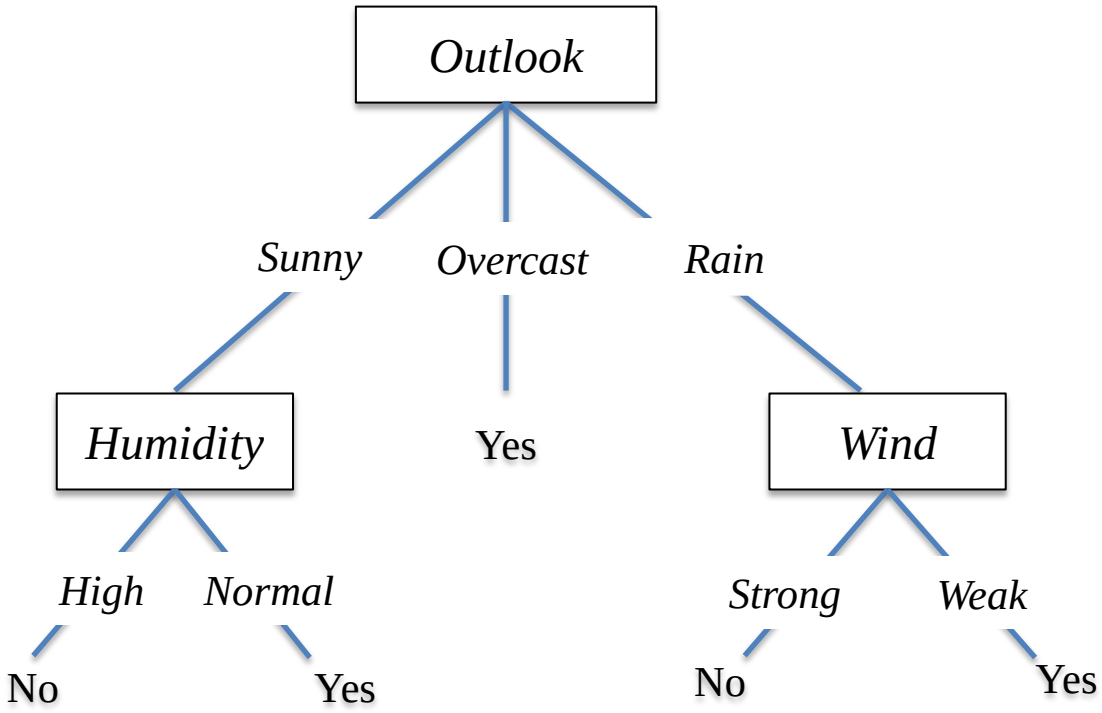
Occam's razor: prefer the simplest hypothesis that fits the data

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

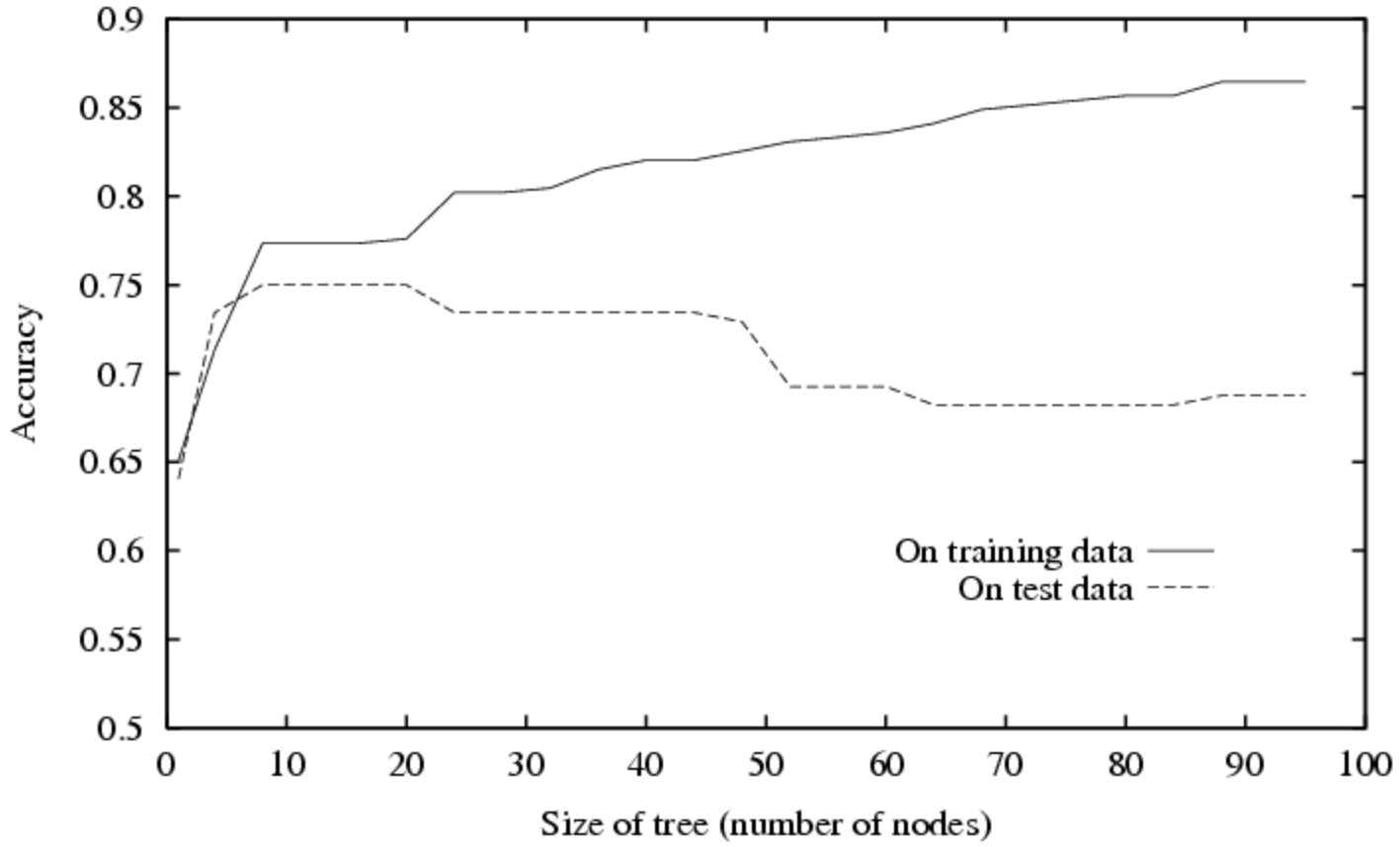
- Error rate over training data: $error_{train}(h)$
- True error rate over all data: $error_{true}(h)$

We say *h* overfits the training data if

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

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

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)