
10-601 Introduction to Machine Learning

Machine Learning Department
School of Computer Science
Carnegie Mellon University

# Decision Trees (Part I)

Matt Gormley
Lecture 2
Aug. 28, 2019

# Q&A

**Q:** Are we using Canvas?

**A:** No.

# Q&A

## **Q:** How will I earn the 5% Participation points?

**A:** Good question! One way is by filling out the **required** poll on what WIFI enabled devices you have. (You must sign in to Google using your **Andrew ID** to fill out the form.)
https://forms.gle/rAp7FJ4A6T3u2uny7

Other points will be earned (most likely) through in-class polls, some "free polls", and other opportunities to gain participation points.

Starting next week, please come to class with a WIFI enabled smartphone or tablet. We'll announce on Piazza what to do if you don't have such a device.

# Reminders

- **Homework 1: Background**
  - **Out: Wed, Aug. 28 (2nd lecture)**
  - **Due: Wed, Sep. 04 at 11:59pm**
  - Two parts:
    1. written part to Gradescope,
    2. programming part to Autolab
  - unique policy for this assignment:
    1. **two submissions** for written (see writeup for details)
    2. **unlimited submissions** for programming (i.e. keep submitting until you get 100%),
  - unique policy for this assignment: we will grant (essentially) any and all extension requests

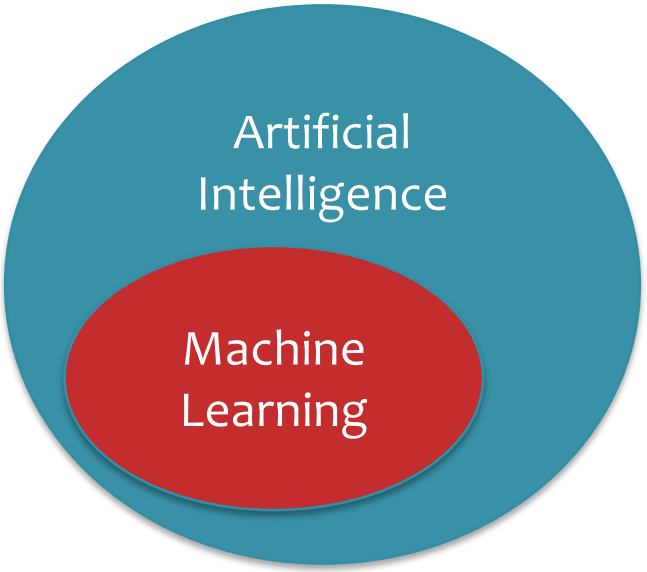
# WHAT IS MACHINE LEARNING?

# Artificial Intelligence

The basic goal of AI is to develop intelligent machines.

This consists of many sub-goals:

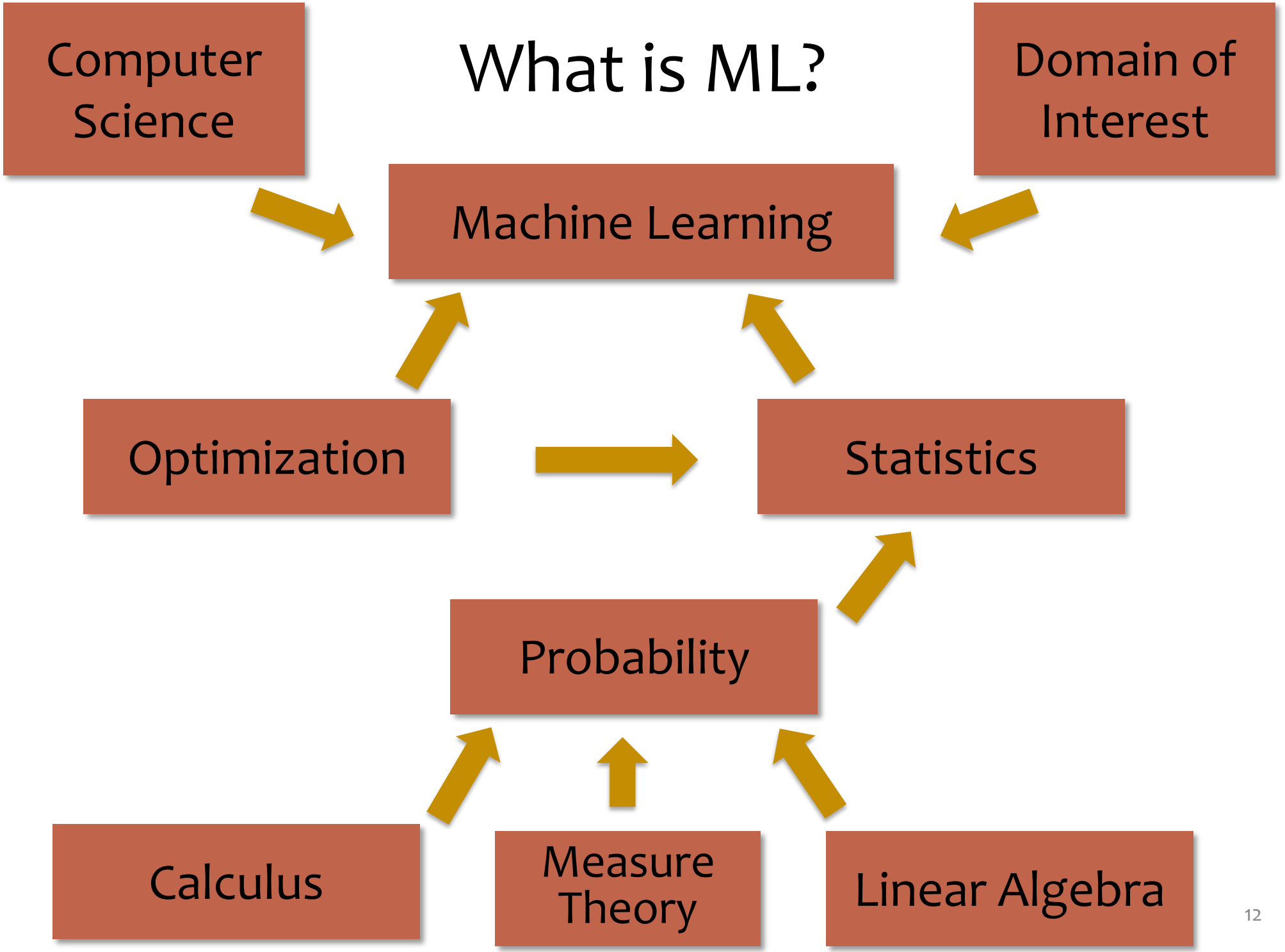
- Perception
- Reasoning
- Control / Motion / Manipulation
- Planning
- Communication
- Creativity
- Learning



# What is Machine Learning?

**The goal of this course is to provide you with a toolbox:**

# What is ML?


Computer Science
Domain of Interest
Machine Learning
Optimization
Statistics
Probability
Calculus
Measure Theory
Linear Algebra

# What is ML?

## Speech Recognition

1. Learning to recognize spoken words

| THEN | NOW |
| --- | --- |
| "…the SPHINX system (e.g. Lee 1989) learns speaker-specific strategies for recognizing the primitive sounds (phonemes) and words from the observed speech signal…neural network methods…hidden Markov models…" (Mitchell, 1997) |  |

Source: https://www.stonetemple.com/great-knowledge-box-showdown/#VoiceStudyResults

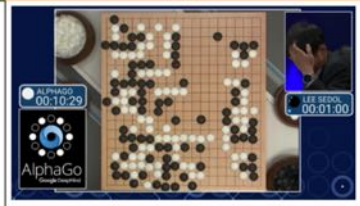
## Robotics

2. Learning to drive an autonomous vehicle

| THEN | NOW |
| --- | --- |
| "…the ALVINN system (Pomerleau 1989) has used its learned strategies to drive unassisted at 70 miles per hour for 90 miles on public highways among other cars…" (Mitchell, 1997) |  waymo.com |

## Games / Reasoning

3. Learning to beat the masters at board games

| THEN | NOW |
| --- | --- |
| "…the world's top computer program for backgammon, TD-GAMMON (Tesauro, 1992, 1995), learned its strategy by playing over one million practice games against itself…" (Mitchell, 1997) |   |

## Computer Vision

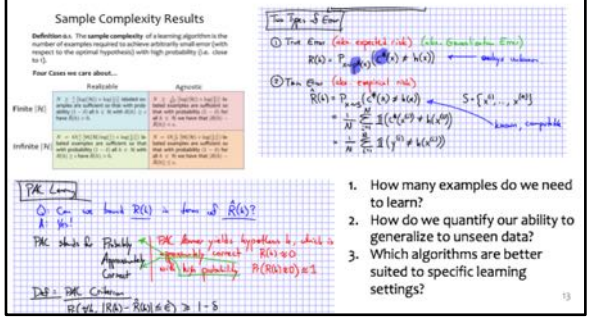
4. Learning to recognize images

| THEN | NOW |
| --- | --- |
| "…The recognizer is a convolution network that can be spatially replicated. From the network output, a hidden Markov model produces word scores. The entire system is globally trained to minimize word-level errors…." (LeCun et al., 1995) |   |

Images from https://blog.openai.com/generative-models/

## Learning Theory

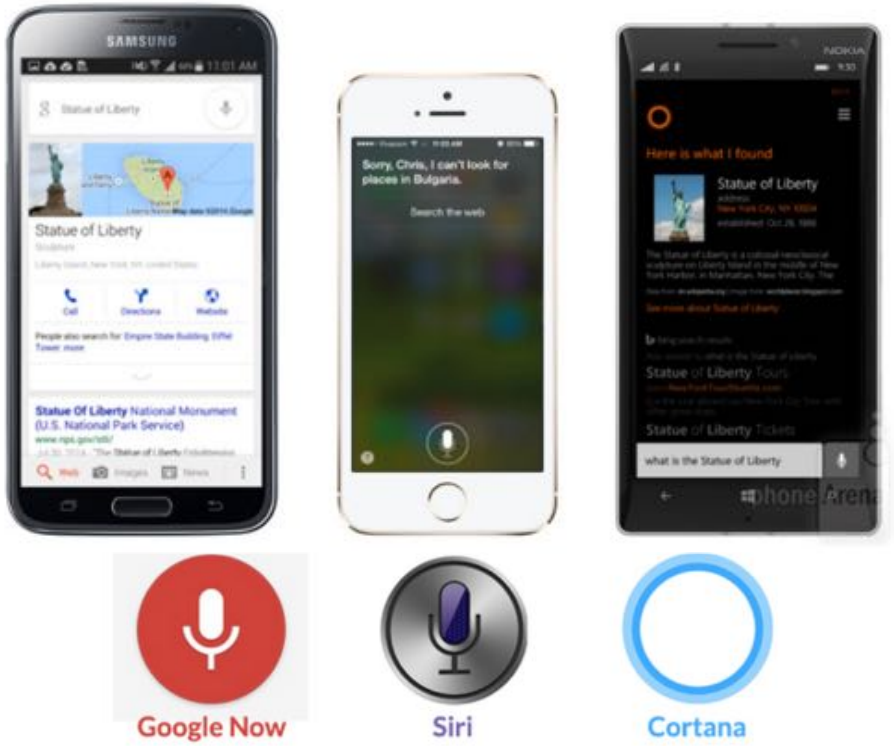
- 5. In what cases and how well can we learn?

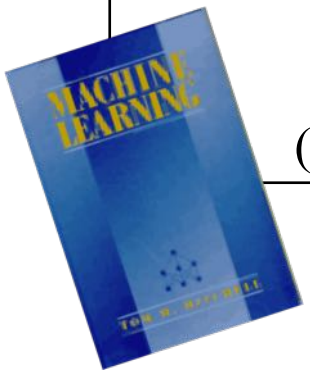



1. How many examples do we need to learn?
2. How do we quantify our ability to generalize to unseen data?
3. Which algorithms are better suited to specific learning settings?

# Speech Recognition

## 1. Learning to recognize spoken words

| THEN | NOW |
| --- | --- |
| "…the SPHINX system (e.g. Lee 1989) learns speaker-specific strategies for recognizing the primitive sounds (phonemes) and words from the observed speech signal…neural network methods…hidden Markov models…" (Mitchell, 1997) |   |




**Source:** https://www.stonetemple.com/great-knowledge-box-showdown/#VoiceStudyResults

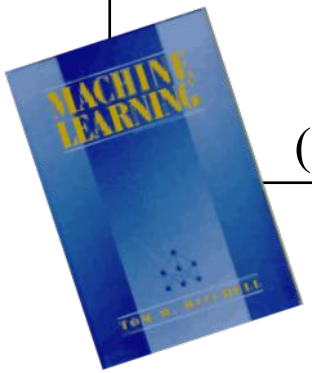
# Robotics

## 2. Learning to drive an autonomous vehicle

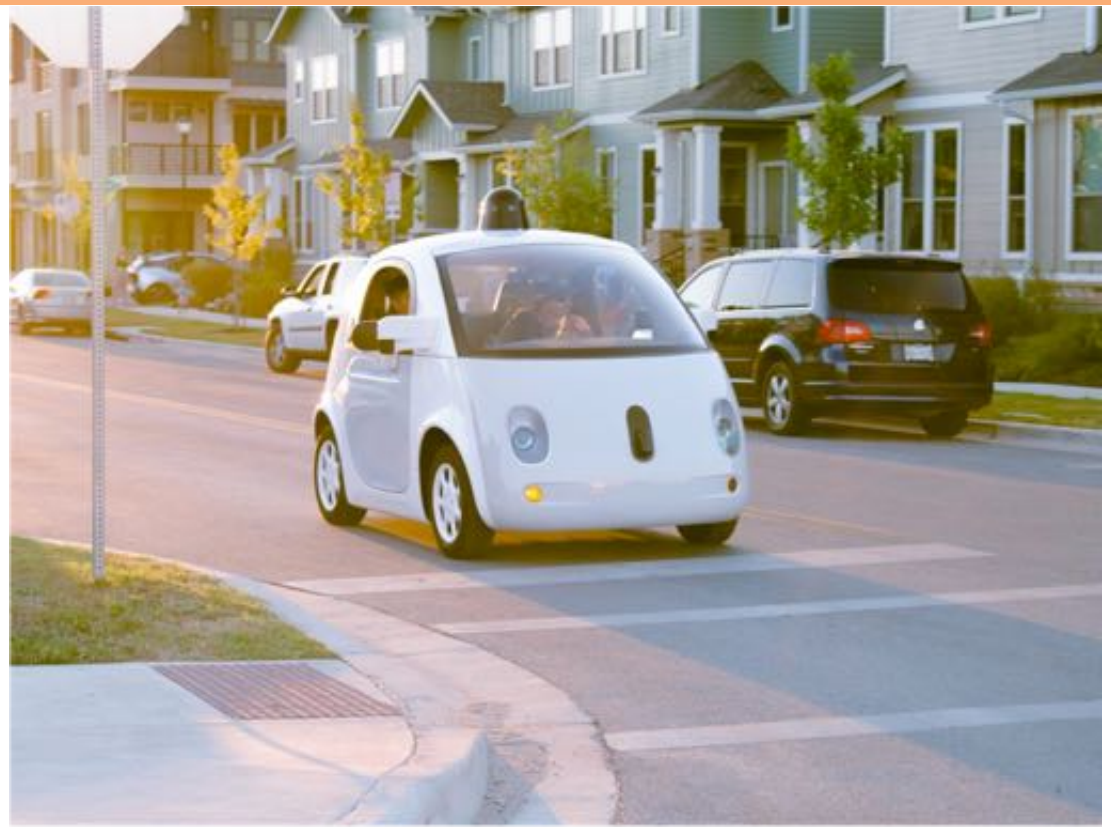
**THEN**

"…the ALVINN system (Pomerleau 1989) has used its learned strategies to drive unassisted at 70 miles per hour for 90 miles on public highways among other cars…"

(Mitchell, 1997)




**NOW**

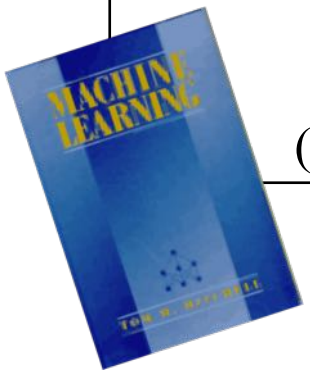


waymo.com

# Robotics

## 2. Learning to drive an autonomous vehicle

| THEN | NOW |
| --- | --- |
| "…the ALVINN system (Pomerleau 1989) has used its learned strategies to drive unassisted at 70 miles per hour for 90 miles on public highways among other cars…" |   |



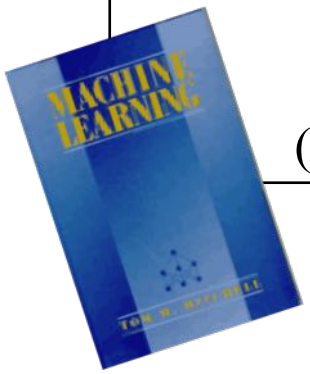

(Mitchell, 1997)

https://www.geek.com/wp-content/uploads/2016/03/uber.jpg

# Robotics

## 2. Learning to drive an autonomous vehicle

| THEN | NOW |
| --- | --- |
| "…the ALVINN system (Pomerleau 1989) has used its learned strategies to drive unassisted at 70 miles per hour for 90 miles on public highways among other cars…" |  |

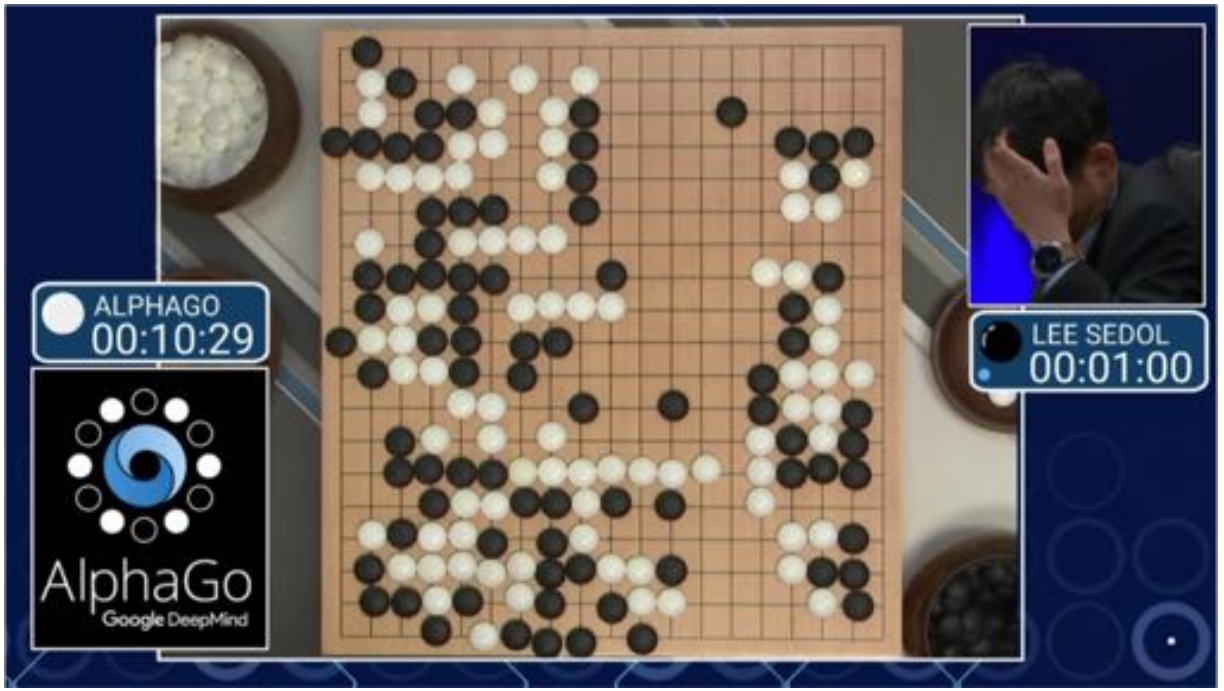



(Mitchell, 1997)

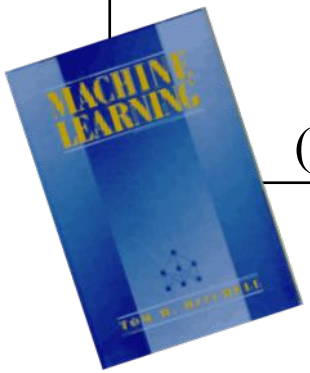
https://www.argo.ai/

# Games / Reasoning

## 3. Learning to beat the masters at board games

| THEN | NOW |
| --- | --- |
| "…the world's top computer program for backgammon, TD-GAMMON (Tesauro, 1992, 1995), learned its strategy by playing over one million practice games against itself…" |  |

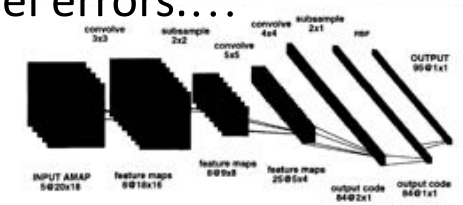
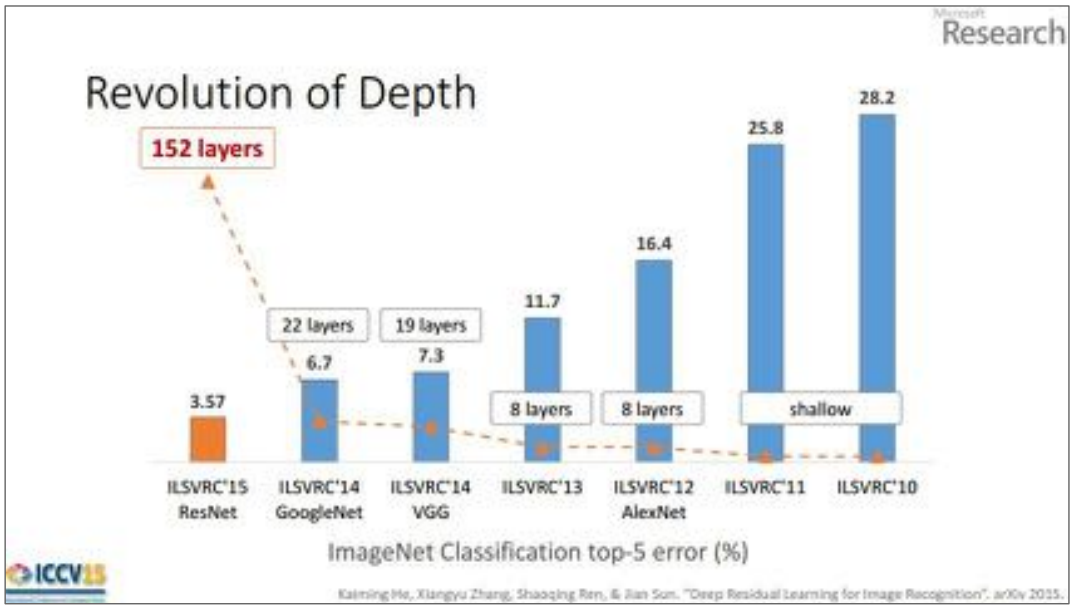





(Mitchell, 1997)

# Computer Vision

## 4. Learning to recognize images

| THEN | NOW |
| --- | --- |
| "…The recognizer is a convolution network that can be spatially replicated. From the network output, a hidden Markov model produces word scores. The entire system is globally trained to minimize word-level errors…."  (LeCun et al., 1995) |    |

Images from https://blog.openai.com/generative-models/

# Learning Theory

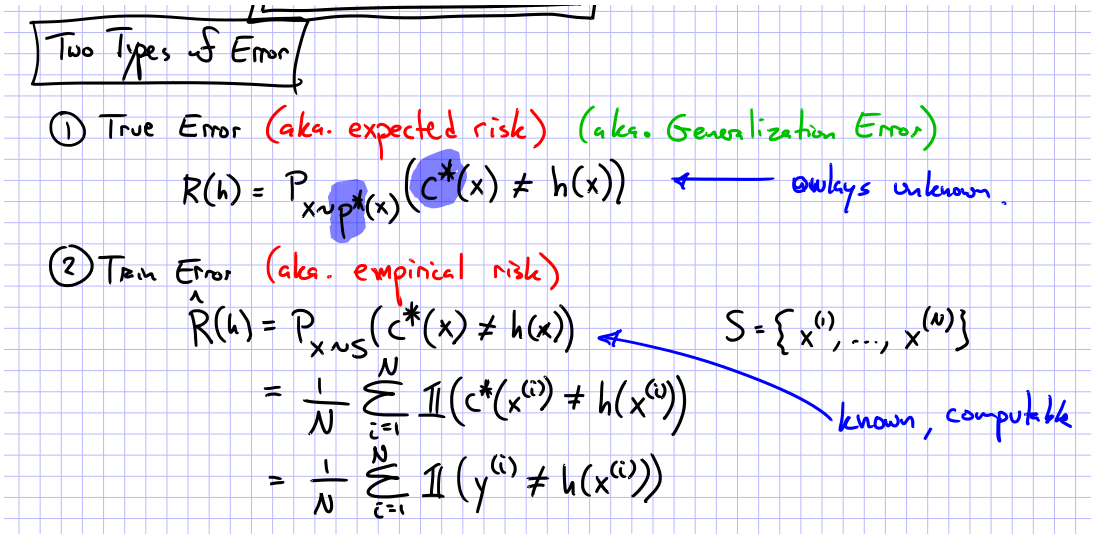
- 5. In what cases and how well can we learn?

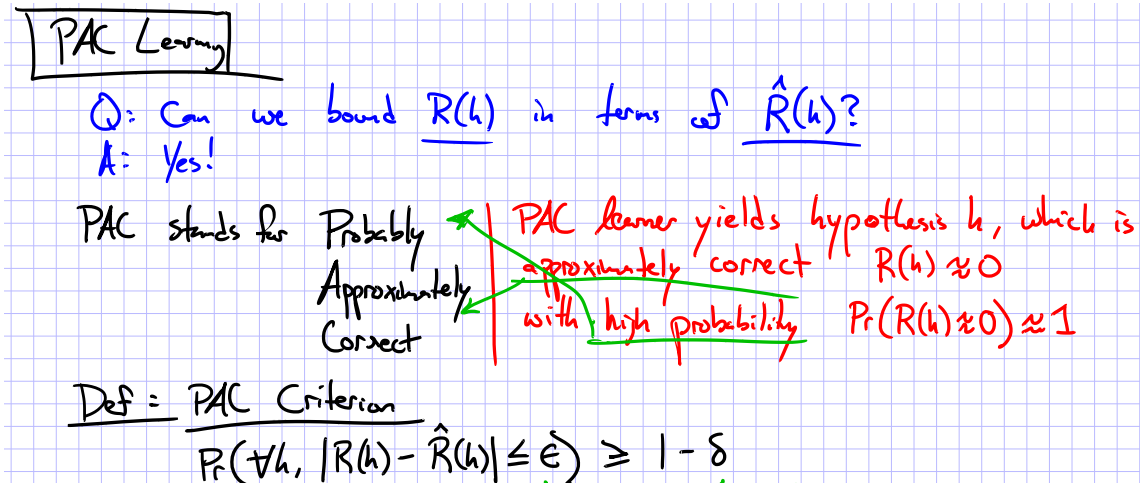
## Sample Complexity Results

**Definition 0.1.** The **sample complexity** of a learning algorithm is the number of examples required to achieve arbitrarily small error (with respect to the optimal hypothesis) with high probability (i.e. close to 1).

**Four Cases we care about…**

| | Realizable | Agnostic |
|---|---|---|
| Finite $\lvert\mathcal{H}\rvert$ | $N \geq \frac{1}{\epsilon}\left[\log(\lvert\mathcal{H}\rvert) + \log(\frac{1}{\delta})\right]$ labeled examples are sufficient so that with probability $(1-\delta)$ all $h \in \mathcal{H}$ with $R(h) \geq \epsilon$ have $\hat{R}(h) > 0$. | $N \geq \frac{1}{2\epsilon^2}\left[\log(\lvert\mathcal{H}\rvert) + \log(\frac{2}{\delta})\right]$ labeled examples are sufficient so that with probability $(1-\delta)$ for all $h \in \mathcal{H}$ we have that $\lvert R(h) - \hat{R}(h)\rvert < \epsilon$. |
| Infinite $\lvert\mathcal{H}\rvert$ | $N = O(\frac{1}{\epsilon}\left[\text{VC}(\mathcal{H})\log(\frac{1}{\epsilon}) + \log(\frac{1}{\delta})\right])$ labeled examples are sufficient so that with probability $(1-\delta)$ all $h \in \mathcal{H}$ with $R(h) \geq \epsilon$ have $\hat{R}(h) > 0$. | $N = O(\frac{1}{\epsilon^2}\left[\text{VC}(\mathcal{H}) + \log(\frac{1}{\delta})\right])$ labeled examples are sufficient so that with probability $(1-\delta)$ for all $h \in \mathcal{H}$ we have that $\lvert R(h) - \hat{R}(h)\rvert \leq \epsilon$. |




1. How many examples do we need to learn?
2. How do we quantify our ability to generalize to unseen data?
3. Which algorithms are better suited to specific learning settings?

# What is ML?

## Speech Recognition

1. Learning to recognize spoken words

| THEN | NOW |
|---|---|
| "…the SPHINX system (e.g. Lee 1989) learns speaker-specific strategies for recognizing the primitive sounds (phonemes) and words from the observed speech signal…neural network methods…hidden Markov models…" (Mitchell, 1997) |   Source: https://www.stonetemple.com/great-knowledge-box-showdown/#VoiceStudyResults |

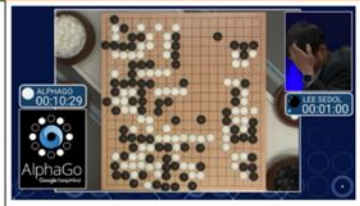
## Robotics

2. Learning to drive an autonomous vehicle

| THEN | NOW |
|---|---|
| "…the ALVINN system (Pomerleau 1989) has used its learned strategies to drive unassisted at 70 miles per hour for 90 miles on public highways among other cars…" (Mitchell, 1997) |  waymo.com |

## Games / Reasoning

3. Learning to beat the masters at board games

| THEN | NOW |
|---|---|
| "…the world's top computer program for backgammon, TD-GAMMON (Tesauro, 1992, 1995), learned its strategy by playing over one million practice games against itself…" (Mitchell, 1997) |   |

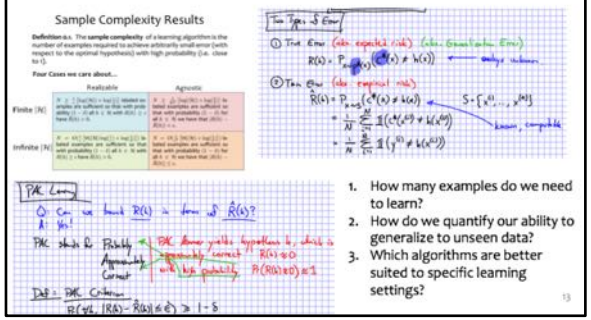
## Computer Vision

4. Learning to recognize images

| THEN | NOW |
|---|---|
| "…The recognizer is a convolution network that can be spatially replicated. From the network output, a hidden Markov model produces word scores. The entire system is globally trained to minimize word-level errors…."  (LeCun et al., 1995) |   Images from https://blog.openai.com/generative-models/ |

## Learning Theory

- 5. In what cases and how well can we learn?




1. How many examples do we need to learn?
2. How do we quantify our ability to generalize to unseen data?
3. Which algorithms are better suited to specific learning settings?

# What is Machine Learning?

**The goal of this course is to provide you with a toolbox:**


**To solve all the problems above and more**

# Topics

- Foundations
  - Probability
  - MLE, MAP
  - Optimization
- Classifiers
  - KNN
  - Naïve Bayes
  - Logistic Regression
  - Perceptron
  - SVM
- Regression
  - Linear Regression
- Important Concepts
  - Kernels
  - Regularization and Overfitting
  - Experimental Design
- Unsupervised Learning
  - K-means / Lloyd's method
  - PCA
  - EM / GMMs
- Neural Networks
  - Feedforward Neural Nets
  - Basic architectures
  - Backpropagation
  - CNNs, LSTMs
- Graphical Models
  - Bayesian Networks
  - HMMs
  - Learning and Inference
- Learning Theory
  - Statistical Estimation (covered right before midterm)
  - PAC Learning
- Other Learning Paradigms
  - Matrix Factorization
  - Reinforcement Learning
  - Information Theory

# ML Big Picture

**Learning Paradigms:**

*What data is available and when? What form of prediction?*

- supervised learning
- unsupervised learning
- semi-supervised learning
- reinforcement learning
- active learning
- imitation learning
- domain adaptation
- online learning
- density estimation
- recommender systems
- feature learning
- manifold learning
- dimensionality reduction
- ensemble learning
- distant supervision
- hyperparameter optimization

**Theoretical Foundations:**

*What principles guide learning?*

- ❑ probabilistic
- ❑ information theoretic
- ❑ evolutionary search
- ❑ ML as optimization

**Problem Formulation:**

*What is the structure of our output prediction?*

| | |
|---|---|
| boolean | Binary Classification |
| categorical | Multiclass Classification |
| ordinal | Ordinal Classification |
| real | Regression |
| ordering | Ranking |
| multiple discrete | Structured Prediction |
| multiple continuous | (e.g. dynamical systems) |
| both discrete & cont. | (e.g. mixed graphical models) |

**Application Areas**

*Key challenges?*

NLP, Speech, Computer Vision, Robotics, Medicine, Search

**Facets of Building ML Systems:**

*How to build systems that are robust, efficient, adaptive, effective?*

1. *Data prep*
2. *Model selection*
3. *Training (optimization / search)*
4. *Hyperparameter tuning on validation data*
5. *(Blind) Assessment on test data*

**Big Ideas in ML:**

*Which are the ideas driving development of the field?*

- *inductive bias*
- *generalization / overfitting*
- *bias-variance decomposition*
- *generative vs. discriminative*
- *deep nets, graphical models*
- *PAC learning*
- *distant rewards*

# DEFINING LEARNING PROBLEMS

# Well-Posed Learning Problems

**Three components** <*T*,*P*,*E*>**:**

1. Task, *T*
2. Performance measure, *P*
3. Experience, *E*

**Definition of learning:**

A computer program **learns** if its performance at tasks in *T*, as measured by *P*, improves with experience *E*.

Definition from (Mitchell, 1997)

# Example Learning Problems

3. Learning to beat the masters at **chess**
    1. Task, *T*:
    2. Performance measure, *P*:
    3. Experience, *E*:

# Example Learning Problems

4. Learning to **respond to voice commands (Siri)**
   1. Task, *T*:
   2. Performance measure, *P*:
   3. Experience, *E*:

# Capturing the Knowledge of Experts


**Solution #1: Expert Systems**

- Over 20 years ago, we had rule based systems
- Ask the expert to
  1. Obtain a PhD in Linguistics
  2. Introspect about the structure of their native language
  3. Write down the rules they devise

| Give me directions to Starbucks |
| --- |
| `If: "give me directions to X"`<br>`Then: directions(here, nearest(X))` |

| How do I get to Starbucks? |
| --- |
| `If: "how do i get to X"`<br>`Then: directions(here, nearest(X))` |

| Where is the nearest Starbucks? |
| --- |
| `If: "where is the nearest X"`<br>`Then: directions(here, nearest(X))` |

# Capturing the Knowledge of Experts


**Solution #1: Expert Systems**

- Over 20 years ago, we had rule based systems
- Ask the expert to
  1. Obtain a PhD in Linguistics
  2. Introspect about the structure of their native language
  3. Write down the rules they devise

I need directions to Starbucks

```
If: "I need directions to X"
Then: directions(here, nearest(X))
```

Starbucks directions

```
If: "X directions"
Then: directions(here, nearest(X))
```

Is there a Starbucks nearby?

```
If: "Is there an X nearby"
Then: directions(here, nearest(X))
```

# Capturing the Knowledge of Experts


**Solution #2: Annotate Data and Learn**

- Experts:
  - **Very good at** answering questions about specific cases
  - **Not very good at** telling **HOW** they do it
- 1990s: So why not just have them tell you what they do on **SPECIFIC CASES** and then let **MACHINE LEARNING** tell you how to come to the same decisions that they did

# Capturing the Knowledge of Experts


## Solution #2: Annotate Data and Learn

1. Collect raw sentences $\{x_1, \ldots, x_n\}$
2. Experts annotate their meaning $\{y_1, \ldots, y_n\}$

| $x_1$: How do I get to Starbucks? |
| --- |
| $y_1$: `directions(here, nearest(Starbucks))` |

| $x_2$: Show me the closest Starbucks |
| --- |
| $y_2$: `map(nearest(Starbucks))` |

| $x_3$: Send a text to John that I'll be late |
| --- |
| $y_3$: `txtmsg(John, I'll be late)` |

| $x_4$: Set an alarm for seven in the morning |
| --- |
| $y_4$: `setalarm(7:00AM)` |

# Example Learning Problems

4. Learning to **respond to voice commands (Siri)**
    1. Task, *T*:
       **predicting action from speech**
    2. Performance measure, *P*:
       **percent of correct actions taken in user pilot study**
    3. Experience, *E*:
       **examples of (speech, action) pairs**

# Problem Formulation

- Often, the same task can be formulated in more than one way:
- Ex: Loan applications
  - creditworthiness/score (regression)
  - probability of default (density estimation)
  - loan decision (classification)

**Problem Formulation:**
*What is the structure of our output prediction?*

| | |
|---|---|
| boolean | Binary Classification |
| categorical | Multiclass Classification |
| ordinal | Ordinal Classification |
| real | Regression |
| ordering | Ranking |
| multiple discrete | Structured Prediction |
| multiple continuous | (e.g. dynamical systems) |
| both discrete & cont. | (e.g. mixed graphical models) |

# Well-posed Learning Problems

**In-Class Exercise**

1. Select a **task**, T
2. Identify **performance measure**, P
3. Identify **experience**, E
4. Report ideas back to rest of class

**Example Tasks**

- Identify objects in an image
- Translate from one human language to another
- Recognize speech
- Assess risk (e.g. in loan application)
- Make decisions (e.g. in loan application)
- Assess potential (e.g. in admission decisions)
- Categorize a complex situation (e.g. medical diagnosis)
- Predict outcome (e.g. medical prognosis, stock prices, inflation, temperature)
- Predict events (default on loans, quitting school, war)
- Plan ahead under perfect knowledge (chess)
- Plan ahead under partial knowledge (Poker, Bridge)

Examples from Roni Rosenfeld

# Machine Learning & Ethics

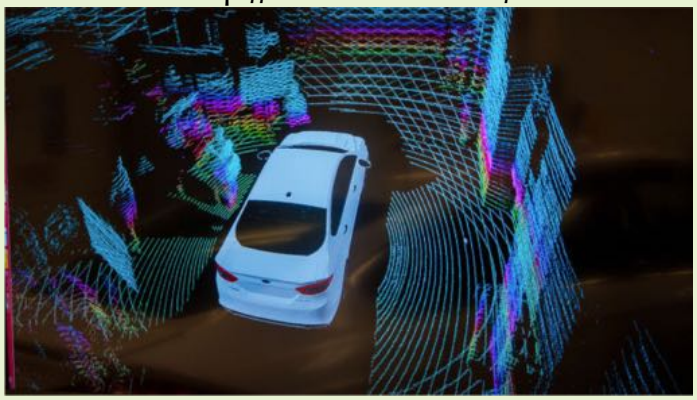
What ethical responsibilities do we have as machine learning experts?

Some topics that we won't cover are probably deserve an entire course

If our search results for news are optimized for ad revenue, might they reflect gender / racial / socio-economic biases?


http://bing.com/

http://arstechnica.com/



How do autonomous vehicles make decisions when all of the outcomes are likely to be negative?

Should restrictions be placed on intelligent agents that are capable of interacting with the world?


http://vizdoom.cs.put.edu.pl/

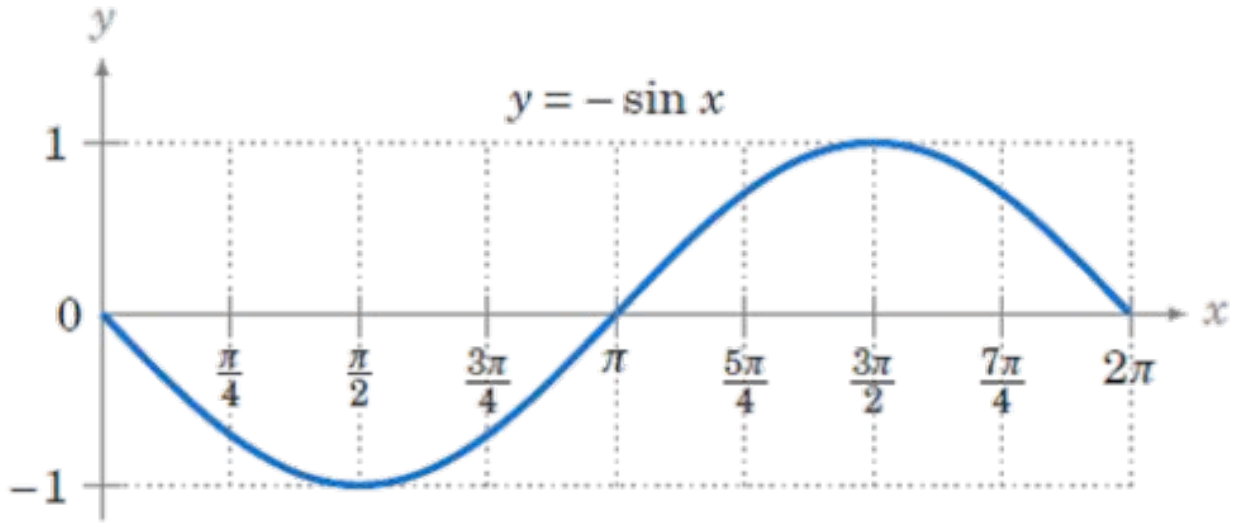
# Big Ideas

1. How to formalize a learning problem
2. How to learn an expert system (i.e. Decision Tree)
3. Importance of inductive bias for generalization
4. Overfitting

# FUNCTION APPROXIMATION

# Function Approximation

**Quiz:** Implement a simple function which returns sin(x).




A few constraints are imposed:

1. You can't call any other trigonometric functions
2. You *can* call an existing implementation of sin(x) a few times (e.g. 100) to test your solution
3. You only need to evaluate it for x in [0, 2*pi]

# Medical Diagnosis

- Setting:
  - Doctor must decide whether or not to prescribe a treatment
  - Looks at attributes of a patient to make a medical diagnosis
  - Prescribes treatment if diagnosis is positive
- Key problem area for Machine Learning
- Potential to reshape health care

# ML as Function Approximation

*Chalkboard*

- ML as Function Approximation
  - Problem setting
  - Input space
  - Output space
  - Unknown target function
  - Hypothesis space
  - Training examples

# DECISION TREES

# Decision Trees

*Chalkboard*

- Example: Medical Diagnosis
- Does memorization = learning?
- Decision Tree as a hypothesis
- Function approximation for DTs

# Tree to Predict C-Section Risk

Learned from medical records of 1000 women (Sims et al., 2000)

Negative examples are C-sections

```
[833+,167-] .83+ .17-
Fetal_Presentation = 1: [822+,116-] .88+ .12-
| Previous_Csection = 0: [767+,81-] .90+ .10-
| | Primiparous = 0: [399+,13-] .97+ .03-
| | Primiparous = 1: [368+,68-] .84+ .16-
| | | Fetal_Distress = 0: [334+,47-] .88+ .12-
| | | | Birth_Weight < 3349: [201+,10.6-] .95+ .
| | | | Birth_Weight >= 3349: [133+,36.4-] .78+
| | | Fetal_Distress = 1: [34+,21-] .62+ .38-
| Previous_Csection = 1: [55+,35-] .61+ .39-
Fetal_Presentation = 2: [3+,29-] .11+ .89-
Fetal_Presentation = 3: [8+,22-] .27+ .73-
```

Figure from Tom Mitchell