# Introduction to Machine Learning

Stephen Scott

University of Nebraska Lincoln

# What is Machine Learning?

- Building machines that automatically **learn** from experience
  - Sub-area of artificial intelligence
- (Very) small sampling of applications:
  - Detection of fraudulent credit card transactions
  - Filtering spam email
  - Autonomous vehicles driving on public highways
  - Self-customizing programs: Web browser that learns what you like/where you are and adjusts
  - Applications we can’t program by hand: E.g., speech recognition
- You’ve used it today already ☺

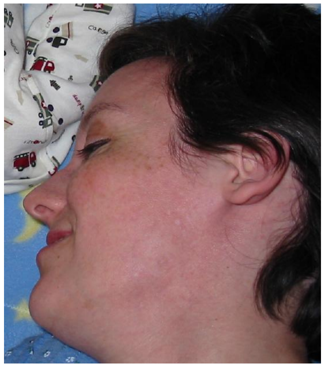
# What is Learning?

- Many different answers, depending on the field you’re considering and whom you ask
  - Artificial intelligence vs. psychology vs. education vs. neurobiology vs. …

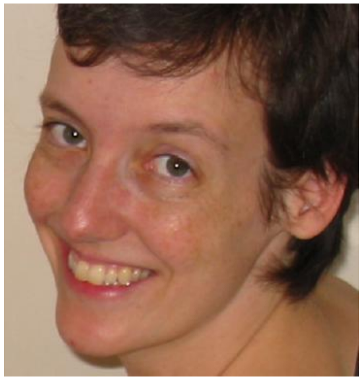
# Does Memorization = Learning?

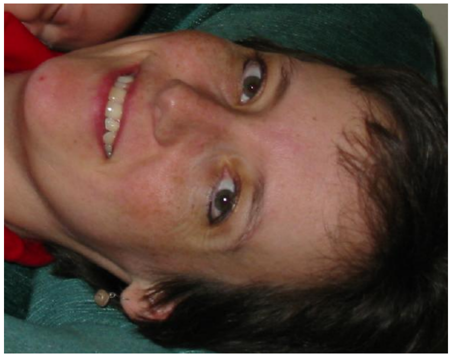
- Test #1: Thomas learns his mother's face

Sees:



But will he recognize:



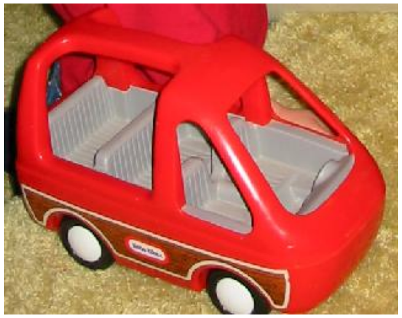


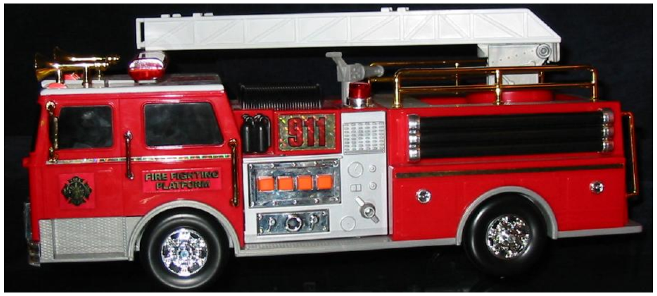
Thus he can generalize beyond what he’s seen!

# Does Memorization = Learning? (cont'd)

- Test #2: Nicholas learns about trucks

Sees:





But will he recognize others?

- So learning involves **ability to generalize** from labeled examples
- In contrast, memorization is trivial, especially for a computer

# What is Machine Learning? (cont'd)

- When do we use machine learning?
  - Human expertise does not exist (navigating on Mars)
  - Humans are unable to explain their expertise (speech recognition; face recognition; driving)
  - Solution changes in time (routing on a computer network; web search suggestions; driving)
  - Solution needs to be adapted to particular cases (biometrics; speech recognition; spam filtering)
- In short, when one needs to generalize from experience in a non-obvious way

# What is Machine Learning? (cont’d)

- When do we **not** use machine learning?
  - Calculating payroll
  - Sorting a list of words
  - Web server
  - Word processing
  - Monitoring CPU usage
  - Querying a database
- When we can definitively specify how all cases should be handled

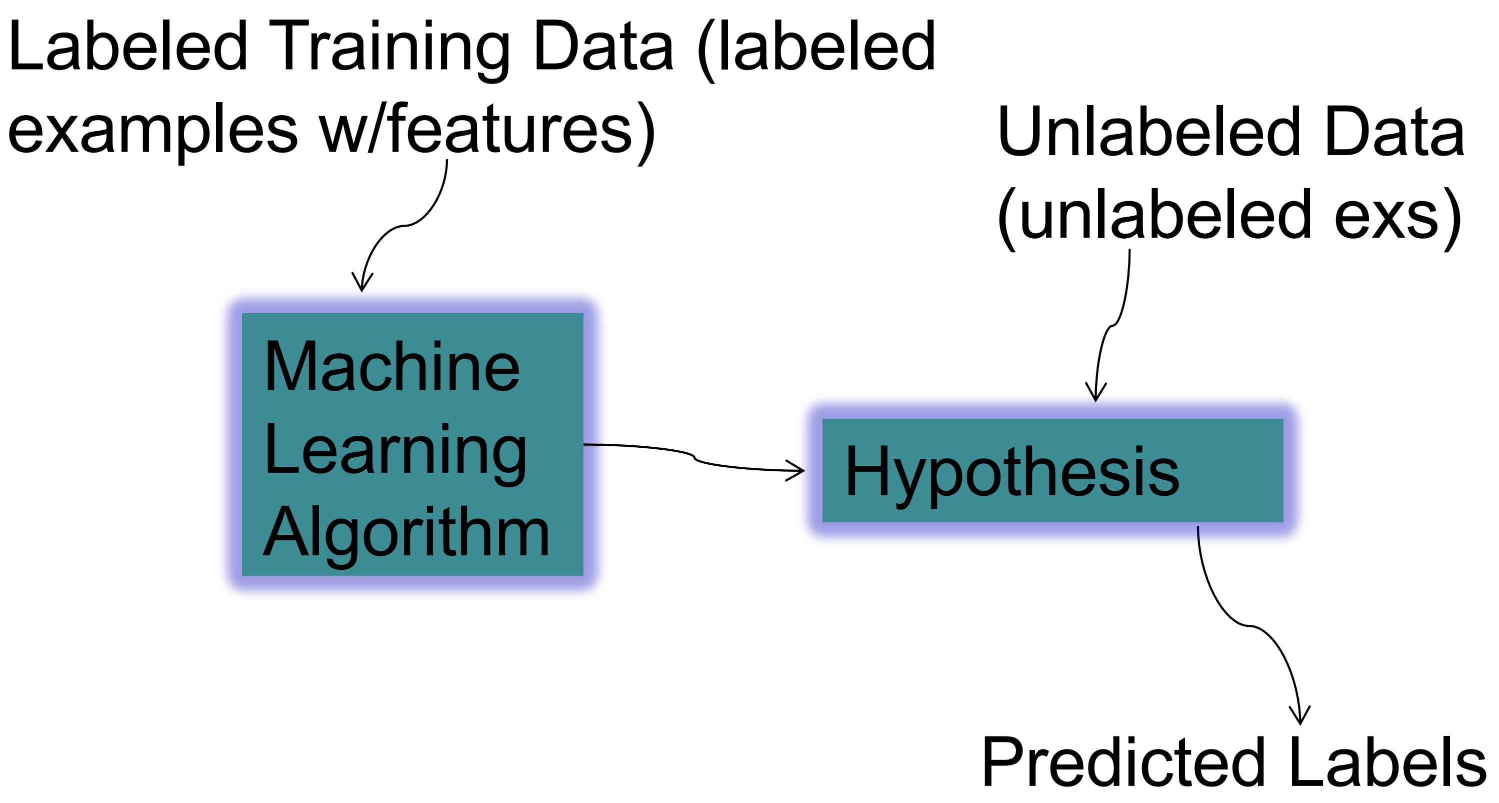
# More Formal Definition

- From Tom Mitchell's 1997 textbook:
  - *"A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P if its performance at tasks in T, as measured by P, improves with experience E."*
- Wide variations of how *T*, *P,* and *E* manifest

# One Type of Task *T:* Supervised Learning

- Given several **labeled examples** of a **learning problem**
  - E.g., trucks vs. non-trucks (binary); height (real)
  - This is the experience *E*
- Examples are described by **features**
  - E.g., number-of-wheels (int), relative-height (height divided by width), hauls-cargo (yes/no)
- A supervised machine learning algorithm uses these examples to create a **hypothesis** (or **model**) that will **predict** the label of new (previously unseen) examples

# Supervised Learning (cont'd)


Labeled Training Data (labeled examples w/features)
Unlabeled Data (unlabeled exs)
Machine Learning Algorithm
Hypothesis
Predicted Labels

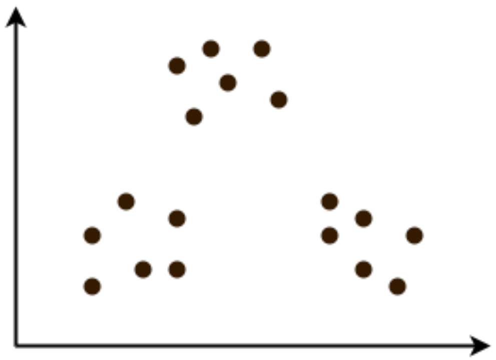
- Hypotheses can take on many forms

# Another Type of Task *T:* Unsupervised Learning

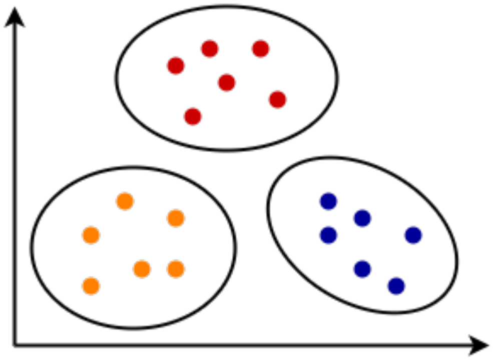
- *E* is now a set of **unlabeled examples**
- Examples are still described by **features**
- Still want to infer a model of the data, but instead of predicting labels, want to understand its **structure**
- E.g., **clustering, density estimation, feature extraction**

# Clustering Examples

Flat

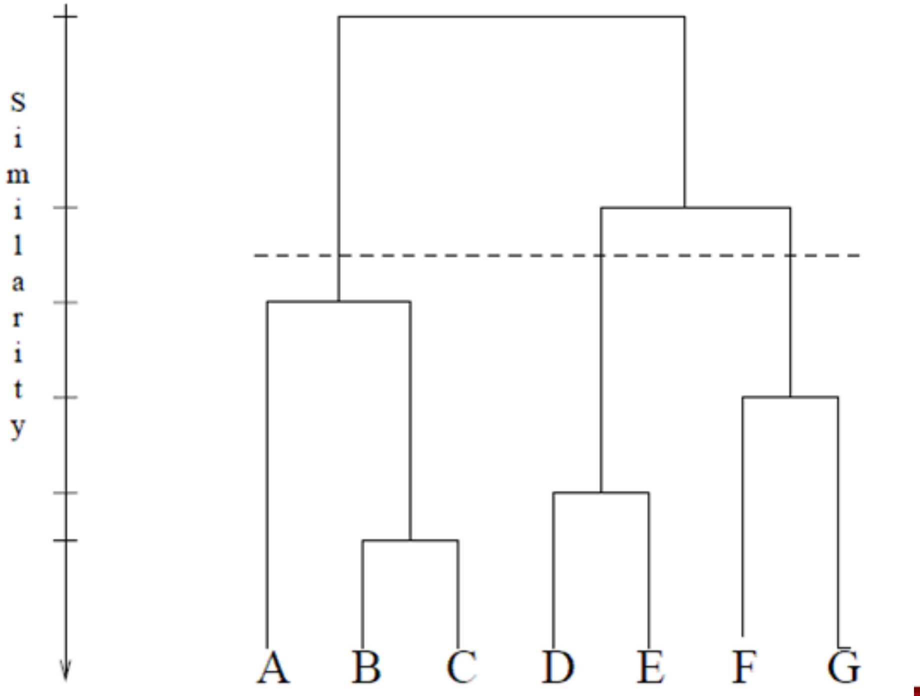


Before K-Means



After K-Means

Hierarchical



# Another Type of Task *T:* Semi-Supervised Learning

- *E* is now a mixture of both **labeled** and **unlabeled examples**
  - Cannot afford to label all of it (e.g., images from web)
- Goal is to infer a classifier, but leverage abundant unlabeled data in the process
  - **Pre-train** in order to **identify relevant features**
  - **Actively purchase** labels from small subset
- Could also use **transfer learning** from one task to another

# Another Type of Task *T:* Reinforcement Learning

- An **agent** *A* interacts with its **environment**
- At each step, *A* perceives the **state** *s* of its environment and takes **action** *a*
- Action *a* results in some **reward** *r* and changes state to *s'*
  - **Markov decision process (MDP)**
- Goal is to maximize **expected long-term reward**
- Applications: Backgammon, Go, video games, self-driving cars

# Reinforcement Learning (cont'd)

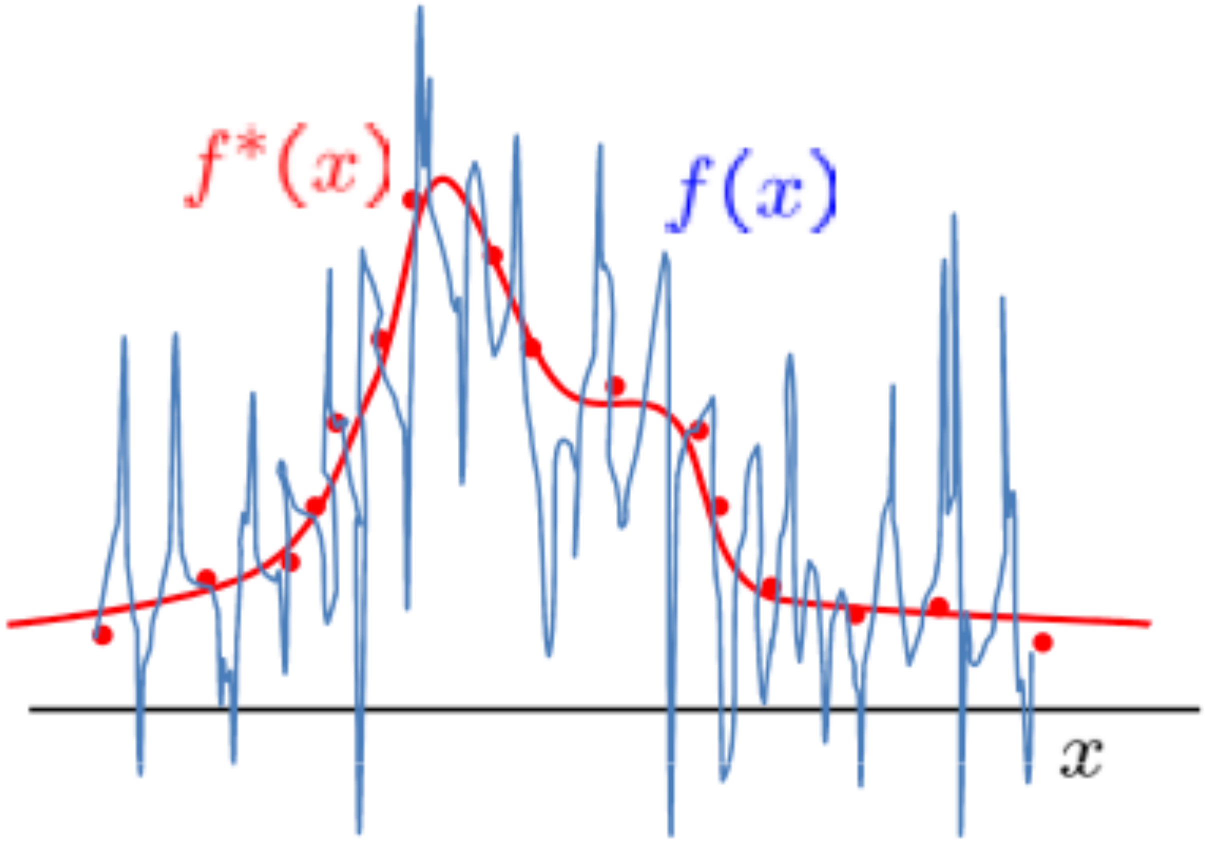
- RL differs from previous tasks in that the feedback (reward) is typically delayed
  - Often takes several actions before reward received
  - E.g., no reward in checkers until game ends
  - Need to decide how much each action contributed to final reward
    - **Credit assignment problem**

# How do ML algorithms work?

- ML boils down to searching a space of functions (models) to optimize an **objective function**
  - – Objective function quantifies goodness of model relative to performance measure *P* on experience *E*
    - Often called “loss” in supervised learning
  - – Objective function also typically depends on a measure of **model complexity** to mitigate **overfitting** training data
    - Called a **regularizer**

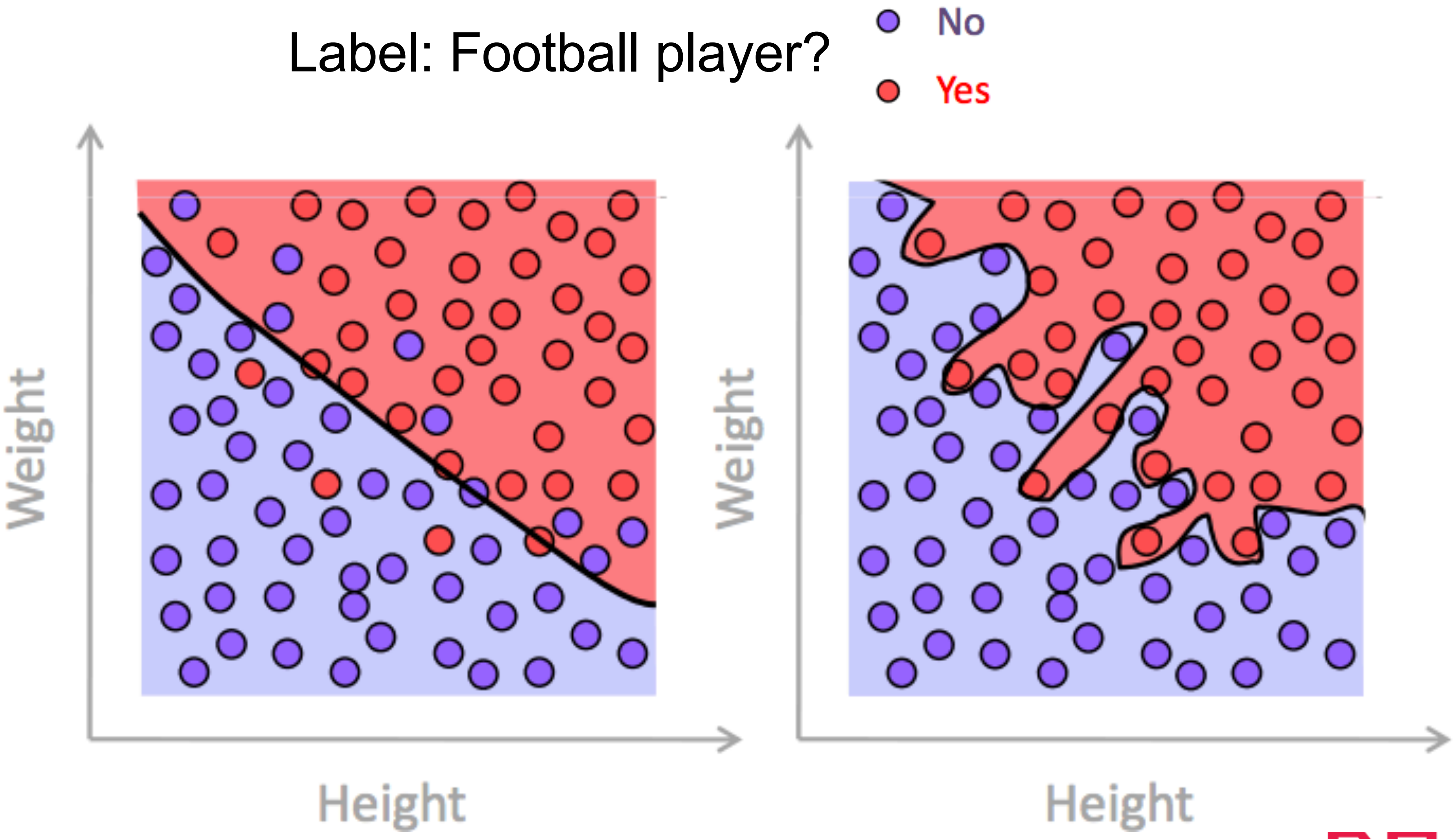
# Model Complexity

- In classification and regression, possible to find hypothesis that perfectly classifies training data
  - But should we necessarily use it?



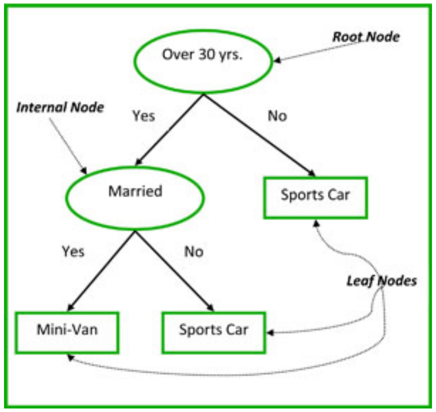
# Model Complexity (cont'd)

Label: Football player?



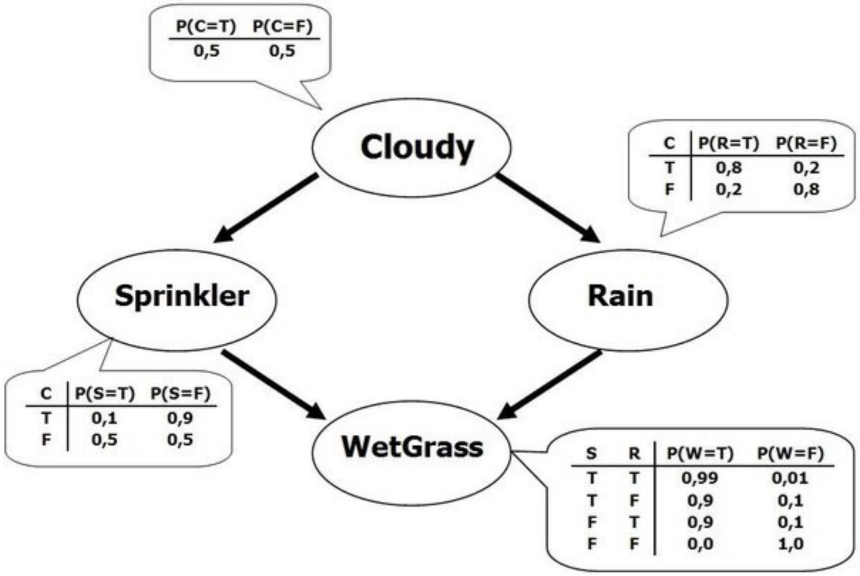
➔ To generalize well, need to balance **training performance** with **simplicity**

# Examples of Types of Models



## Decision trees

## Probabilistic graphical models

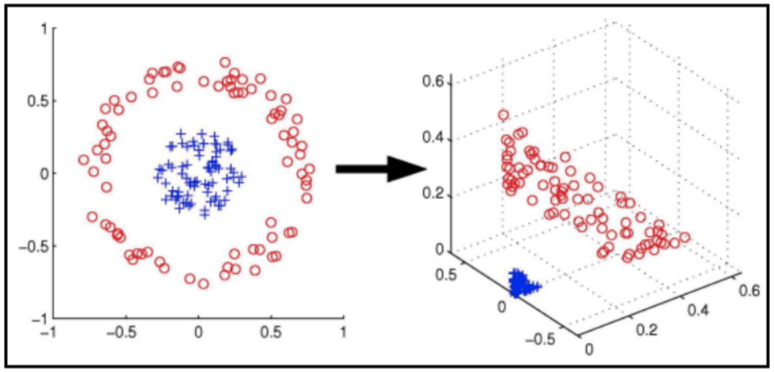


| P(C=T) | P(C=F) |
|---|---|
| 0,5 | 0,5 |

| C | P(R=T) | P(R=F) |
|---|---|---|
| T | 0,8 | 0,2 |
| F | 0,2 | 0,8 |

| C | P(S=T) | P(S=F) |
|---|---|---|
| T | 0,1 | 0,9 |
| F | 0,5 | 0,5 |

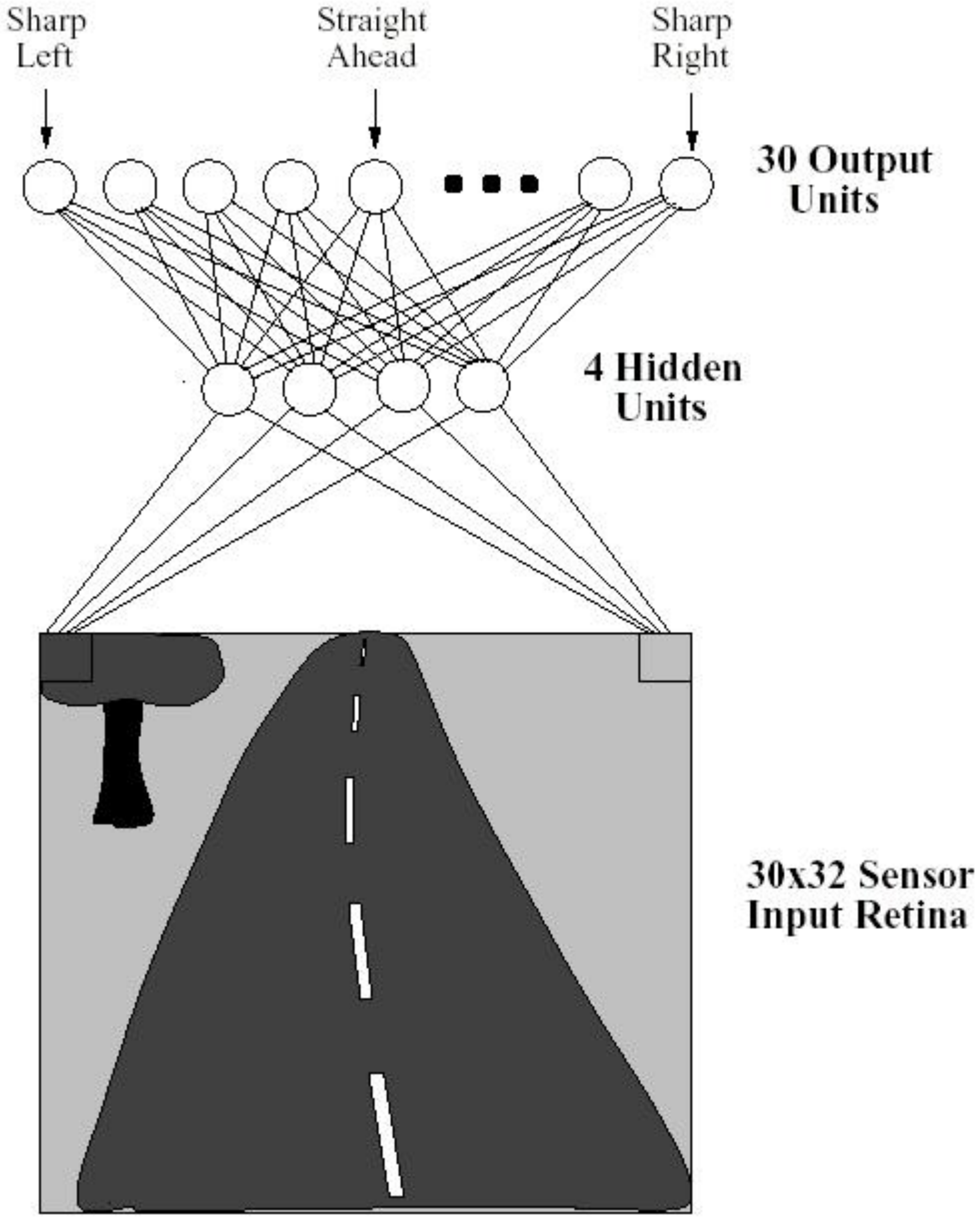
| S | R | P(W=T) | P(W=F) |
|---|---|---|---|
| T | T | 0,99 | 0,01 |
| T | F | 0,9 | 0,1 |
| F | T | 0,9 | 0,1 |
| F | F | 0,0 | 1,0 |



## Support vector machines

# Artificial Neural Networks

- Designed to simulate brains
- "Neurons" (processing units) communicate via connections, each with a numeric weight
- Learning comes from adjusting the weights



# Artificial Neural Networks (cont'd)

- ANNs are basis of **deep learning**
- "Deep" refers to depth of the architecture
  - More layers => more processing of inputs
- Each input to a node is multiplied by a weight
- Weighted sum *S* sent through **activation function:**
  - **Rectified linear:** max(0, *S*)
  - **Convolutional + pooling:** Weights represent a (e.g.) 3x3 **convolutional kernel** to identify features in (e.g.) images
  - **Sigmoid:** tanh(*S*) or 1/(1+exp(-*S*))
- Often trained via **stochastic gradient descent**

# Example Performance Measures *P*

- Let *X* be a set of labeled instances
- **Classification error:** number of instances of *X* hypothesis *h* predicts correctly, divided by $|X|$
- **Squared error:** Sum $(y_i - h(x_i))^2$ over all $x_i$
  - If labels from {0,1}, same as classification error
  - Useful when labels are real-valued
- **Cross-entropy:** Sum over all $x_i$ from *X*:

  $y_i \ln h(x_i) + (1 – y_i) \ln (1 - h(x_i))$
  - Generalizes to > 2 classes
  - Effective when *h* predicts probabilities

# Other Variations

- Missing attributes
  - Must some how estimate values or tolerate them
- Sequential data, e.g., genomic sequences, speech
  - Hidden Markov models
  - Recurrent neural networks
- Have much unlabeled data and/or missing attributes, but can purchase some labels/attributes for a price
  - **Active learning** approaches try to minimize cost
- Outlier detection
  - E.g., intrusion detection in computer systems

# Relevant Disciplines

- Artificial intelligence: Learning as a search problem, using prior knowledge to guide learning
- Probability theory: computing probabilities of hypotheses
- Computational complexity theory: Bounds on inherent complexity of learning
- Control theory: Learning to control processes to optimize performance measures
- Philosophy: Occam's razor (everything else being equal, simplest explanation is best)
- Psychology and neurobiology: Practice improves performance, biological justification for artificial neural networks
- Statistics: Estimating generalization performance

# Summary

- Idea of intelligent machines has been around a long time
- Early on was primarily academic interest
- Past few decades, improvements in processing power plus very large data sets allows highly sophisticated (and successful!) approaches
- Prevalent in modern society
  - You’ve probably used it several times today
- No single “best” approach for any problem
  - Depends on requirements, type of data, volume of data