# Introduction to Machine Learning

*Knowing is not enough; we must apply.*
*Willing is not enough; we must do.*

*Johann Wolfgang von Goethe*

Acknowledgement: Eric Eaton, www.seas.upenn.edu/~cis519

Robot Image Credit: Viktoriya Sukhanova © 123RF.com

# What is Machine Learning?

"Learning is any process by which a system improves performance from experience."

- Herbert Simon

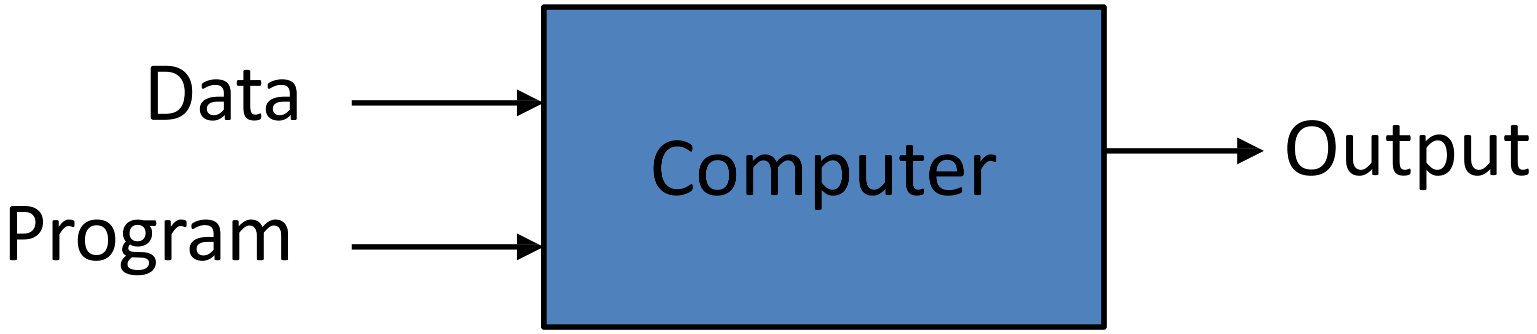
Definition by Tom Mitchell (1998):

Machine Learning is the study of algorithms that

- improve their performance $P$
- at some task $T$
- with experience $E$.

A well-defined learning task is given by <$P$, $T$, $E$>.

## Traditional Programming



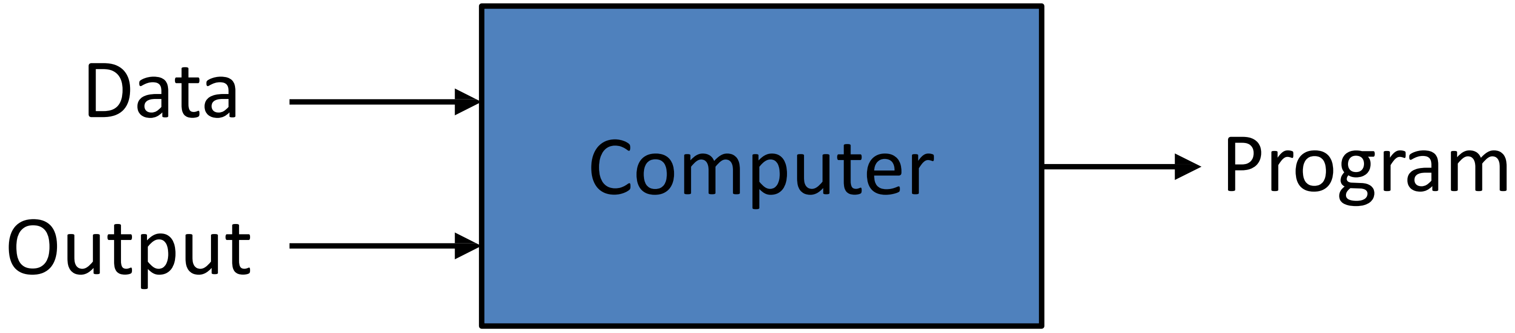
Data

Program

Computer

Output

## Machine Learning



Data

Output

Computer

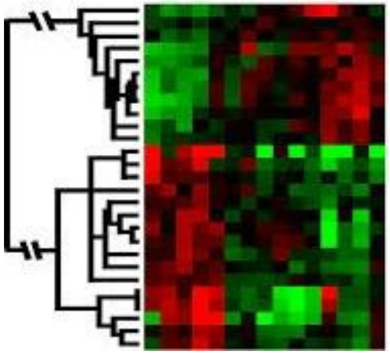
Program

Slide credit: Pedro Domingos

# When Do We Use Machine Learning?

ML is used when:

- Human expertise does not exist (navigating on Mars)
- Humans can’t explain their expertise (speech recognition)
- Models must be customized (personalized medicine)
- Models are based on huge amounts of data (genomics)

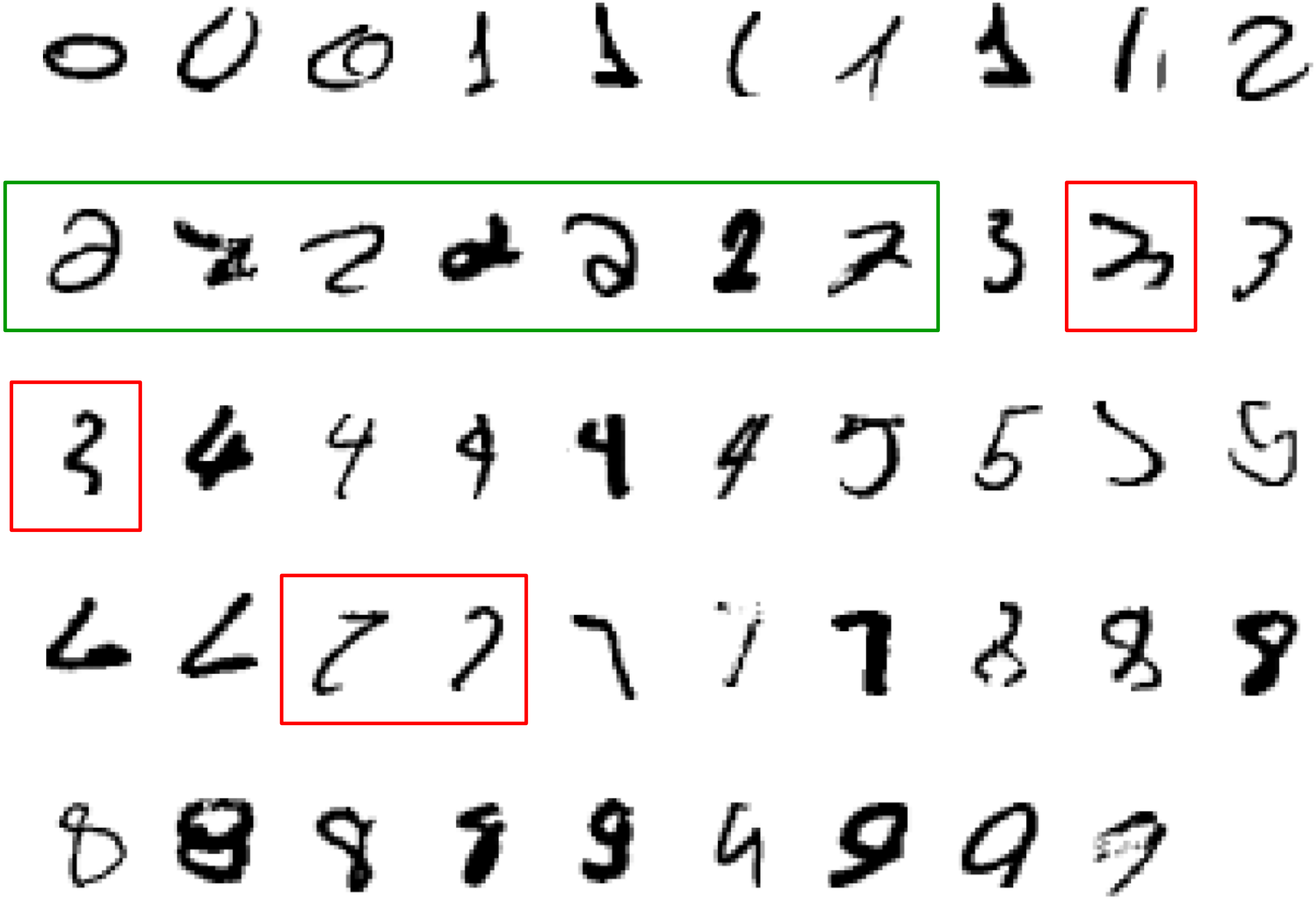


Learning isn’t always useful:

- There is no need to “learn” to calculate payroll

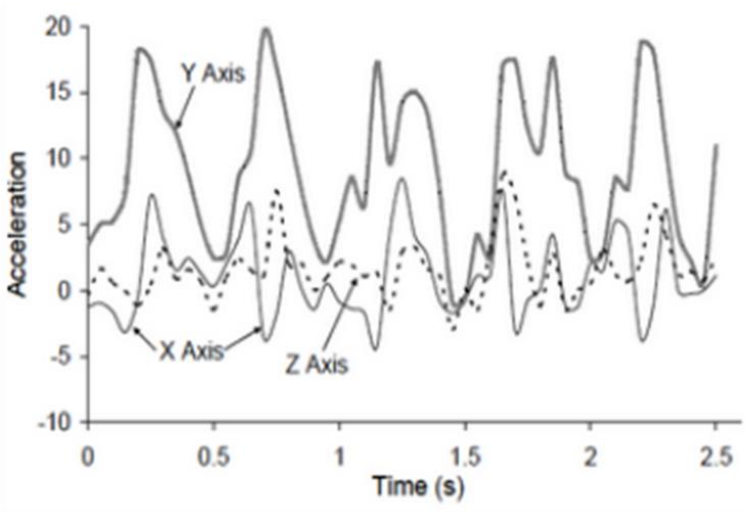
Based on slide by E. Alpaydin

# A classic example of a task that requires machine learning: It is very hard to say what makes a 2

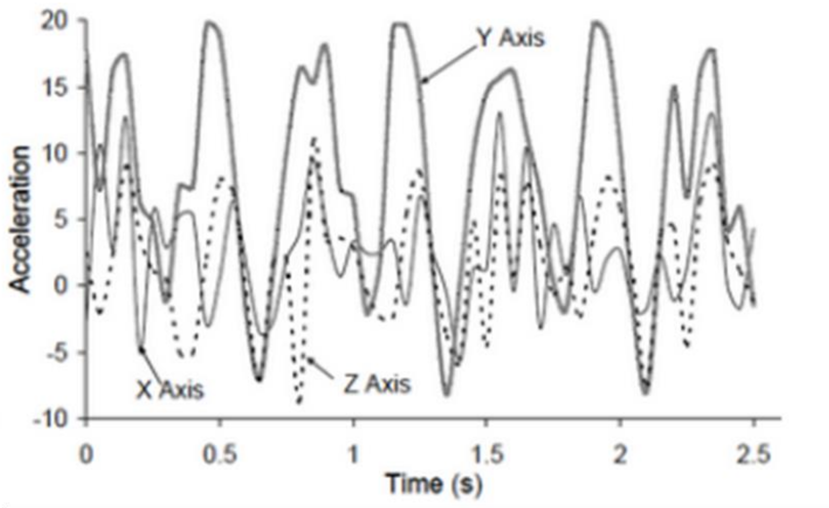


Slide credit: Geoffrey Hinton

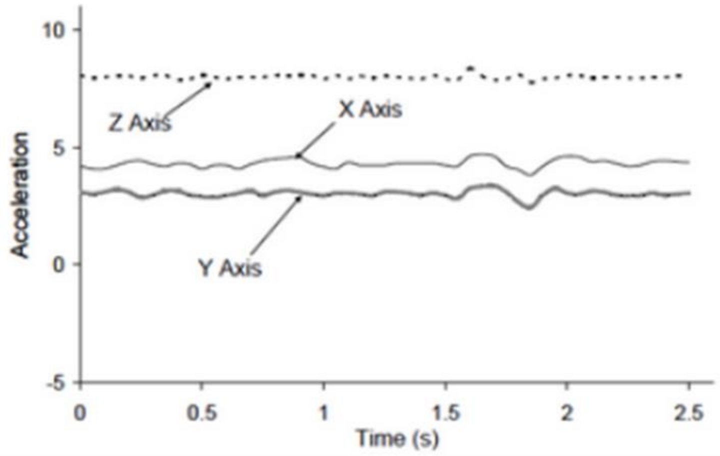
# Another Example: Activity Recognition



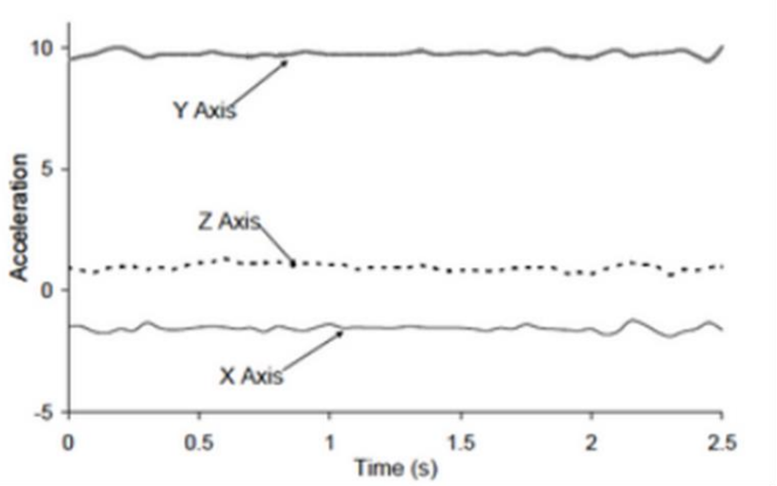
**Waking**



**Jogging**



**Sitting**



**Standing**

# Some more examples of tasks that are best solved by using a learning algorithm

- Recognizing patterns:
  - Facial identities or facial expressions
  - Handwritten or spoken words
  - Medical images
- Generating patterns:
  - Generating images or motion sequences
- Recognizing anomalies:
  - Unusual credit card transactions
  - Unusual patterns of sensor readings in a nuclear power plant
- Prediction:
  - Future stock prices or currency exchange rates

Slide credit: Geoffrey Hinton

# Sample Applications

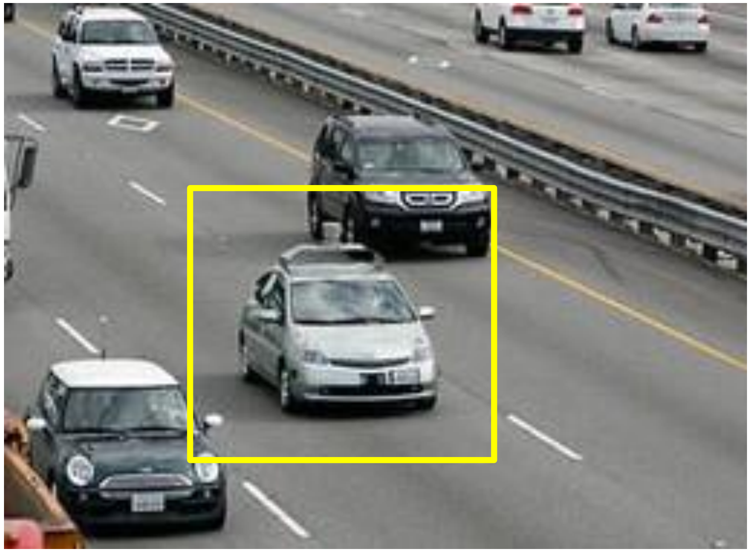
- Web search
- Computational biology
- Finance
- E-commerce
- Space exploration
- Robotics
- Information extraction
- Social networks
- Debugging software
- Many real-world problems that you are working on

Slide credit: Pedro Domingos

# Samuel’s Checkers-Player

“Machine Learning: Field of study that gives computers the ability to learn without being explicitly programmed.” -Arthur Samuel (1959)

# Autonomous Cars



- Nevada made it legal for autonomous cars to drive on roads in June 2011
- As of 2013, four states (Nevada, Florida, California, and Michigan) have legalized autonomous cars

Penn's Autonomous Car →
(Ben Franklin Racing Team)

# Speech Technology

您需要做什么?

打开手机中国

正在打开手机中国

Slide credit: Li Deng, MS Research

# Defining the Learning Task

Improve on task T, with respect to
performance metric P, based on experience E

T: Playing checkers
P: Percentage of games won against an arbitrary opponent
E: Playing practice games against itself

T: Recognizing hand-written words
P: Percentage of words correctly classified
E: Database of human-labeled images of handwritten words

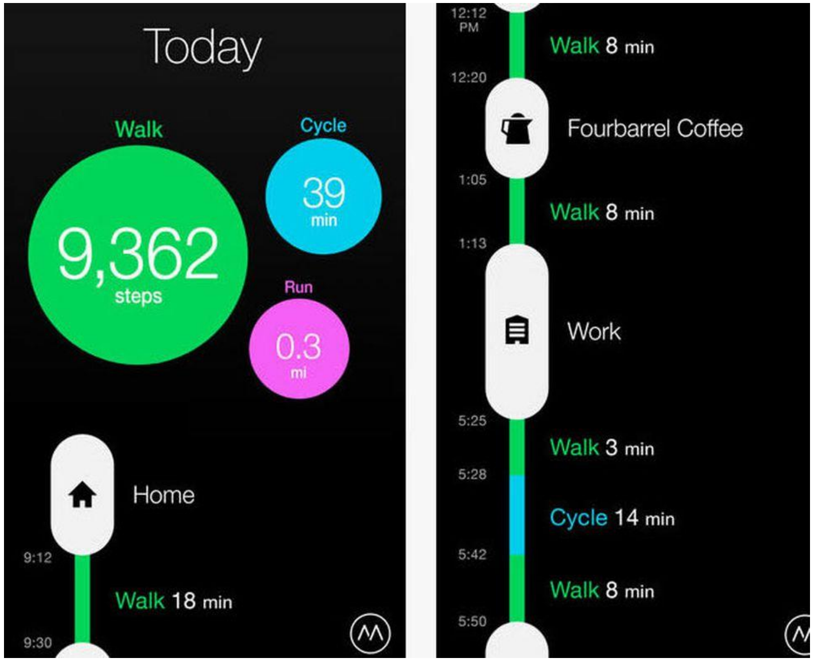
T: Driving on four-lane highways using vision sensors
P: Average distance traveled before a human-judged error
E: A sequence of images and steering commands recorded while observing a human driver.

T: Categorize email messages as spam or legitimate.
P: Percentage of email messages correctly classified.
E: Database of emails, some with human-given labels

Slide credit: Ray Mooney

# A Specific Example: Activity Recognition

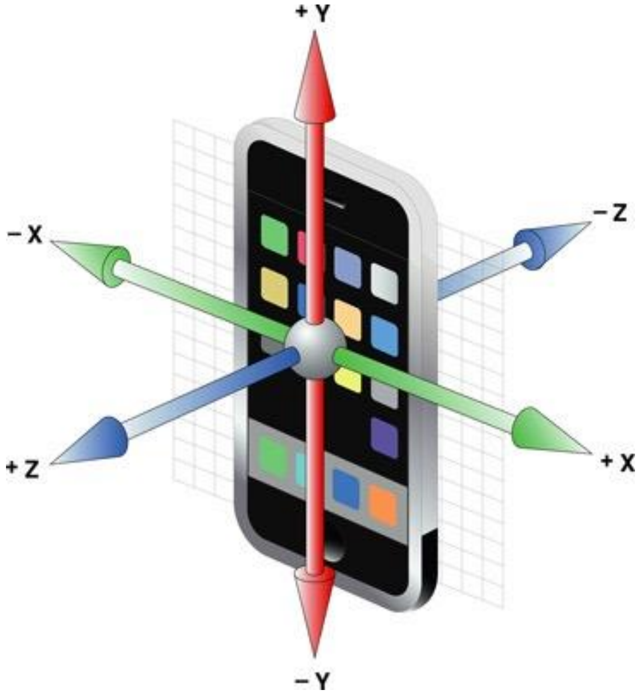
# Example: Mobile Activity Tracker

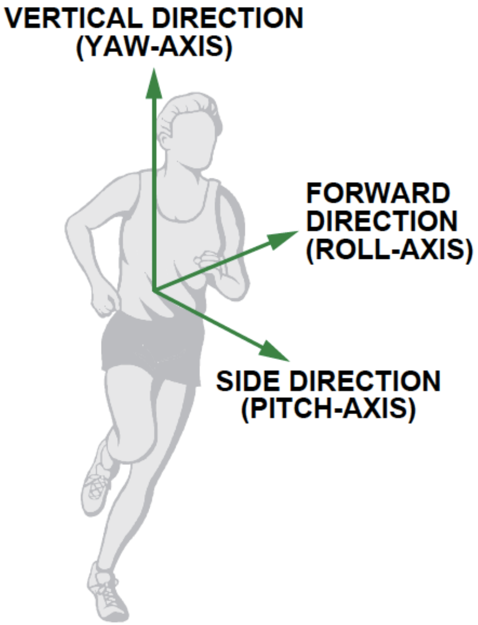


- Everyday exercise progress monitor and motivator
- Provide reliable feedback about how much they move. (People often overestimate!)
- Provide instant and constant feedback about activity levels.
- Gamify to encourage individuals to compete in getting fit and losing weight.

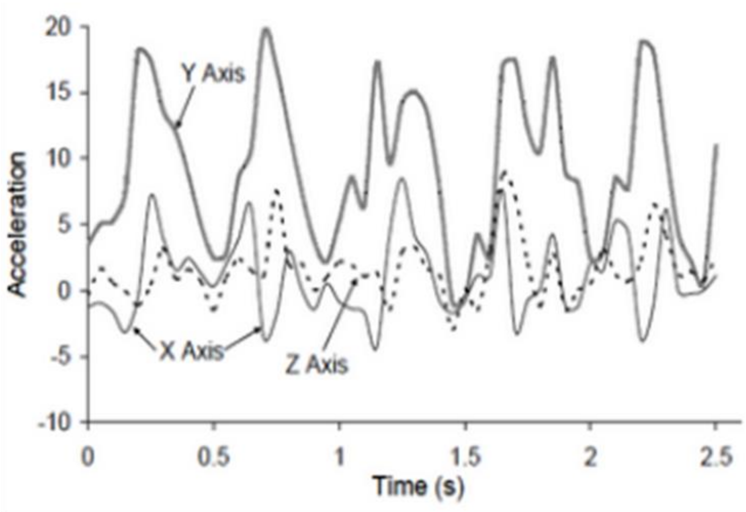
# Inertial Sensors: Accelerometer

- All commodity smartphones have accelerometers.
- Measure linear acceleration (m/s$^2$) in three different directions.

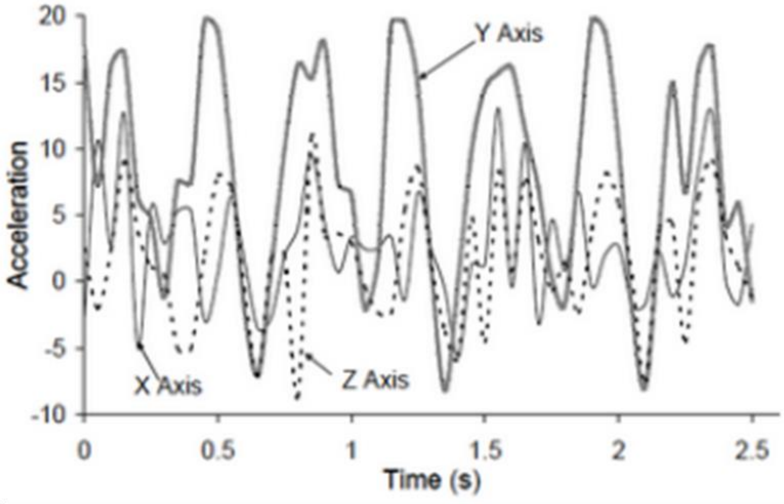




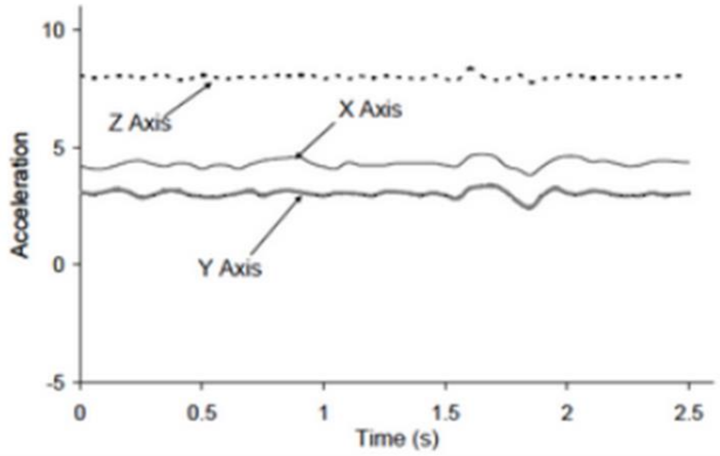
# Signal Patterns



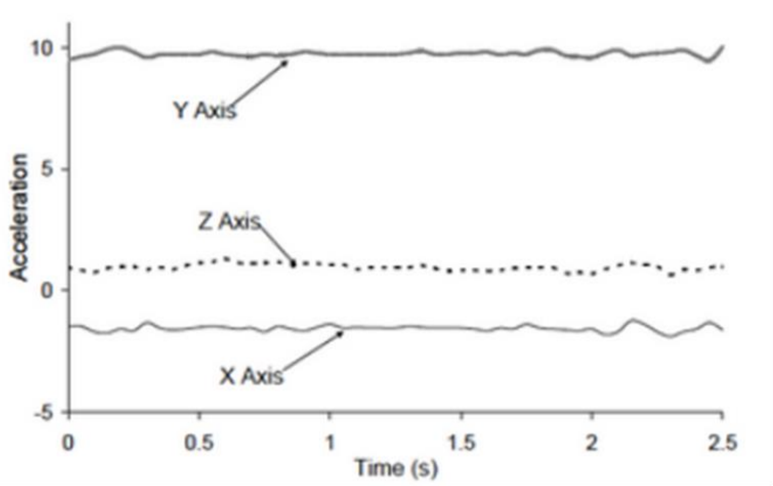
**Waking**



**Jogging**



**Sitting**



**Standing**

# Solution 1: Heuristic

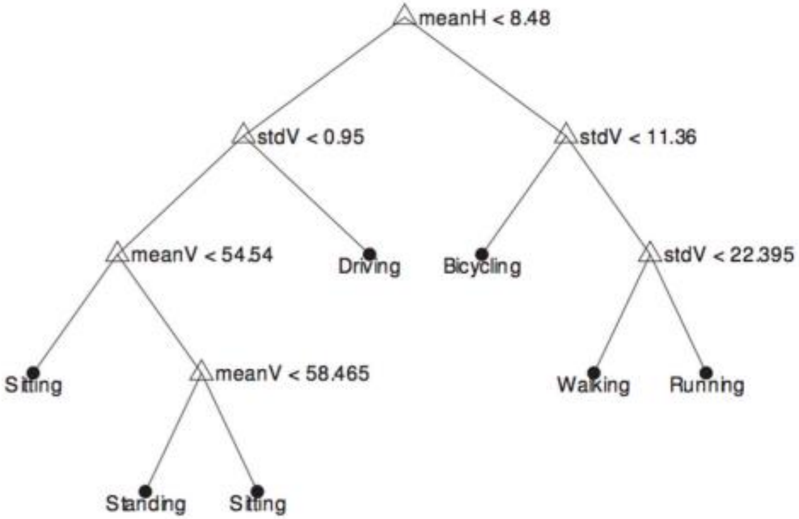
- If STDEV(y-axis samples) < $C_{Threshold1}$
  - If AVG(y-axis samples) > $C_{Threshold2}$
    - output standing
  - Else
    - output sitting
- Else
  - If FFT(y-axis samples) < $C_{Threshold3}$
    - output walking
  - Else
    - output jogging

# Are We Good?

- How do we determine good features and good thresholds?
  - ✓ How do we know STDEV is better than MAX?
  - ✓ How do we know AVG is better than Median?
  - ✓ How do we know the right values for $C_{threshold}$ ?
- What if a user puts her phone in her bag, not in her front pocket?
  - ✓ The Y-axis of the phone is not anymore the major axis of movement.
- How do we solve these problems? A better heuristic?

# Solution 2: Decision Tree

- A simple but effective ML classifier.
- This tree can be built by the C4.5 algorithm.
- Given sufficient training data, the algorithm can automatically determine the important features and their thresholds.



# How to Build a Decision Tree?

- Pseudocode

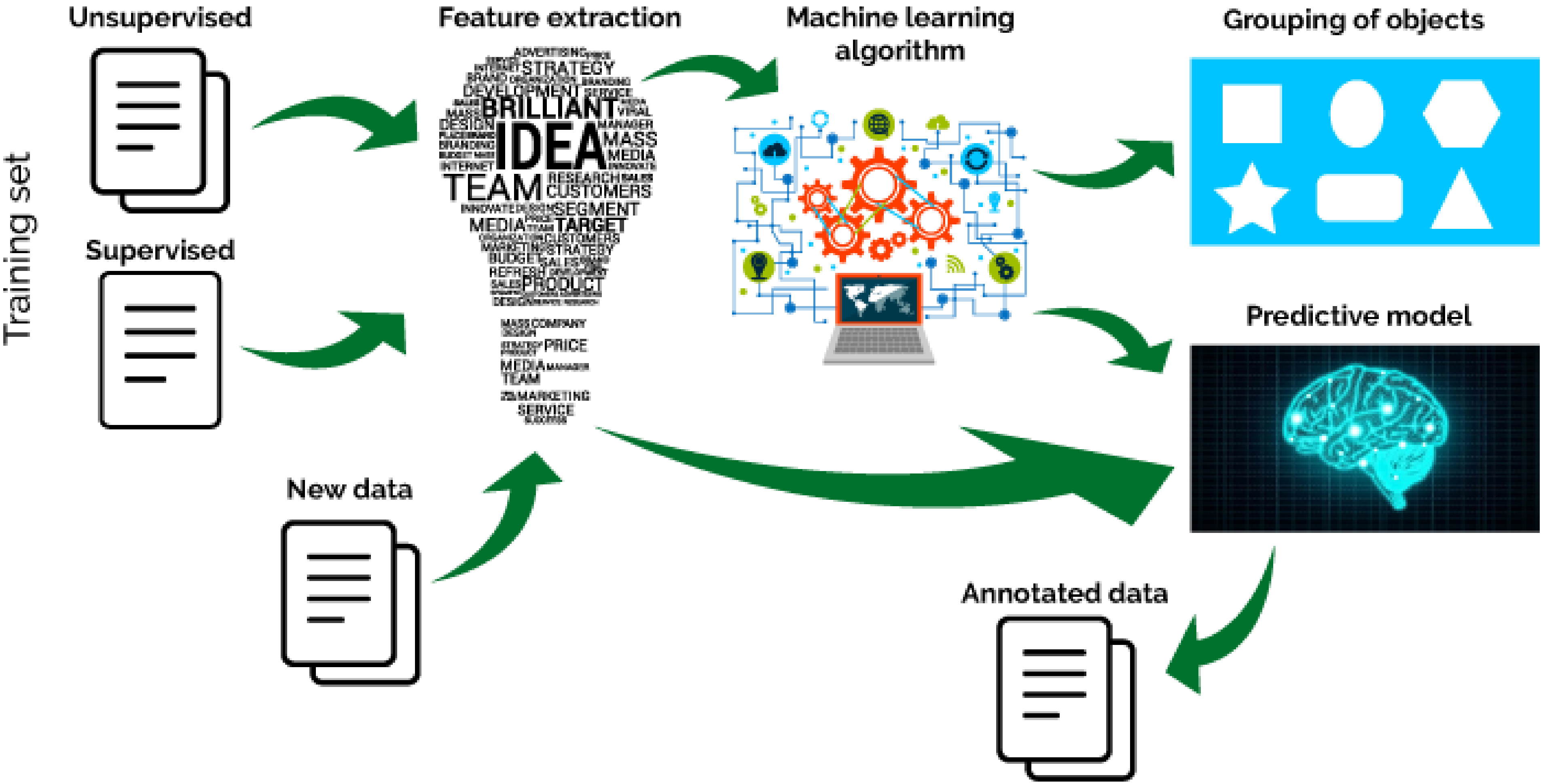
```
1. For each feature f, find the normalized information gain (a metric to effectively split data into classes) from  splitting on f
2. Let f_best  be the attribute with the highest normalized information gain
3. Create a decision node that splits on f_best
4. Recurse on the sublists obtained by splitting on f_best, and add those nodes as children of node
```

- More to be covered in Section 7 (Tree-based Approach) of our textbook.

# Other ML Techniques

- Naïve Bayes classifier
- Decision tree
- Random forest
- Support vector machine
- Linear regression
- Hidden Markov model
- Gaussian mixture model
- ….

# ML Techniques Flow



Training set

Unsupervised

Supervised

New data

Feature extraction

Machine learning algorithm

Grouping of objects

Predictive model

Annotated data

# ML Techniques: Limitations

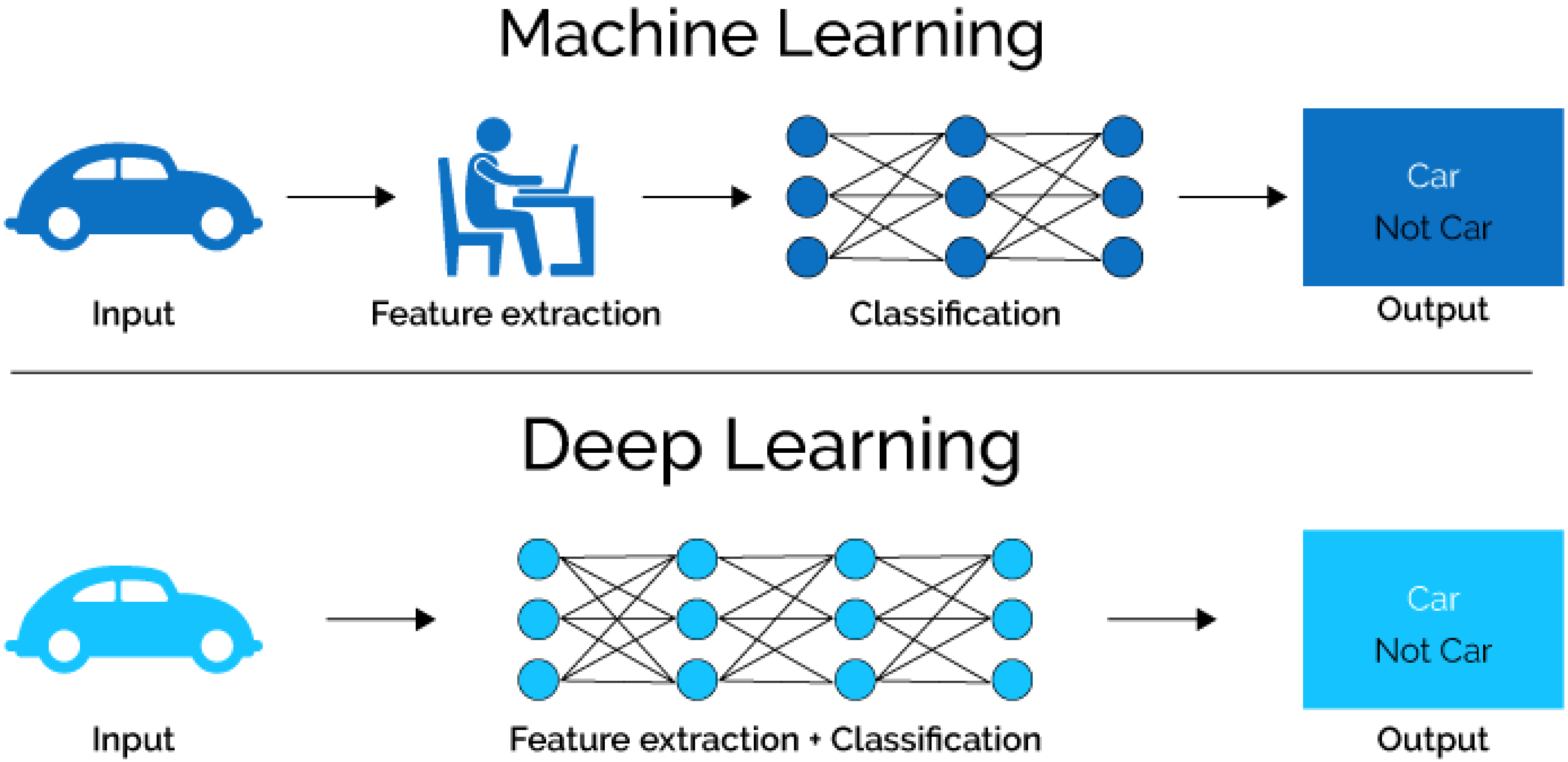
- Expert knowledge required for feature extraction
- Not easy to improve accuracy after a certain point (even with a large volume of data)
- Not easy to model non-linear relations between an input and output.

# ML Techniques: Limitations

- Linear regression?
  - ✓ Why is it linear?
- Bayesian?
  - ✓ What is the prior?
- SVM?
  - ✓ What are the features?
- Decision tree?
  - ✓ What are the nodes/variables?
- KNN?
  - ✓ Cluster on what features?

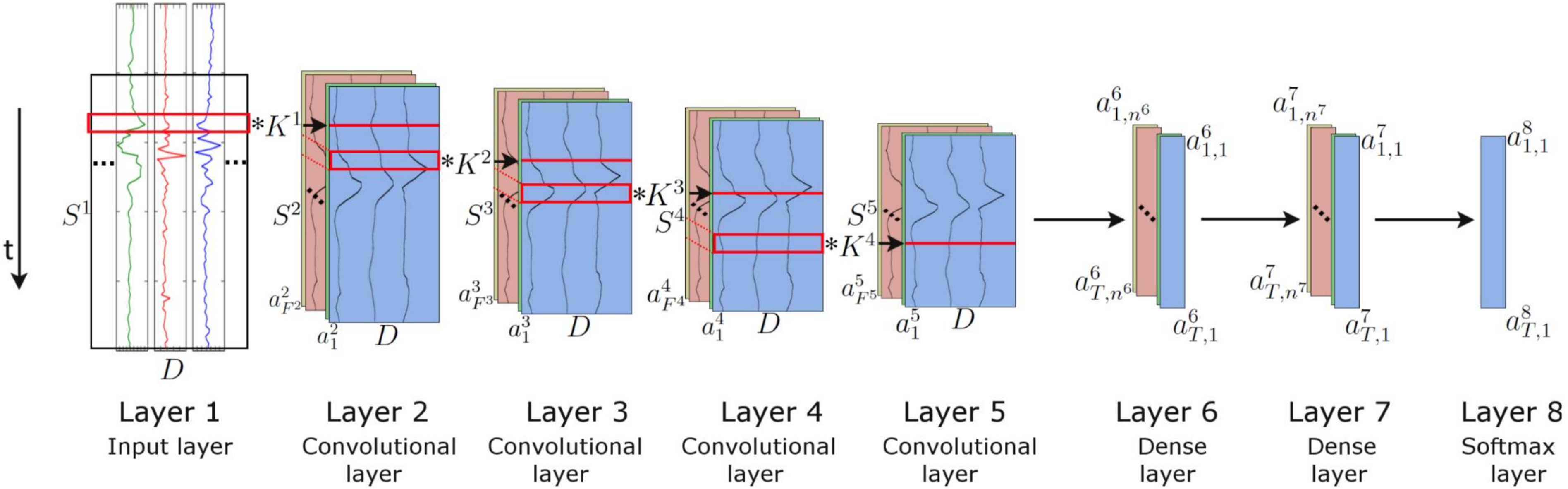
These methods do not suit well with very complex models.

# Deep Learning



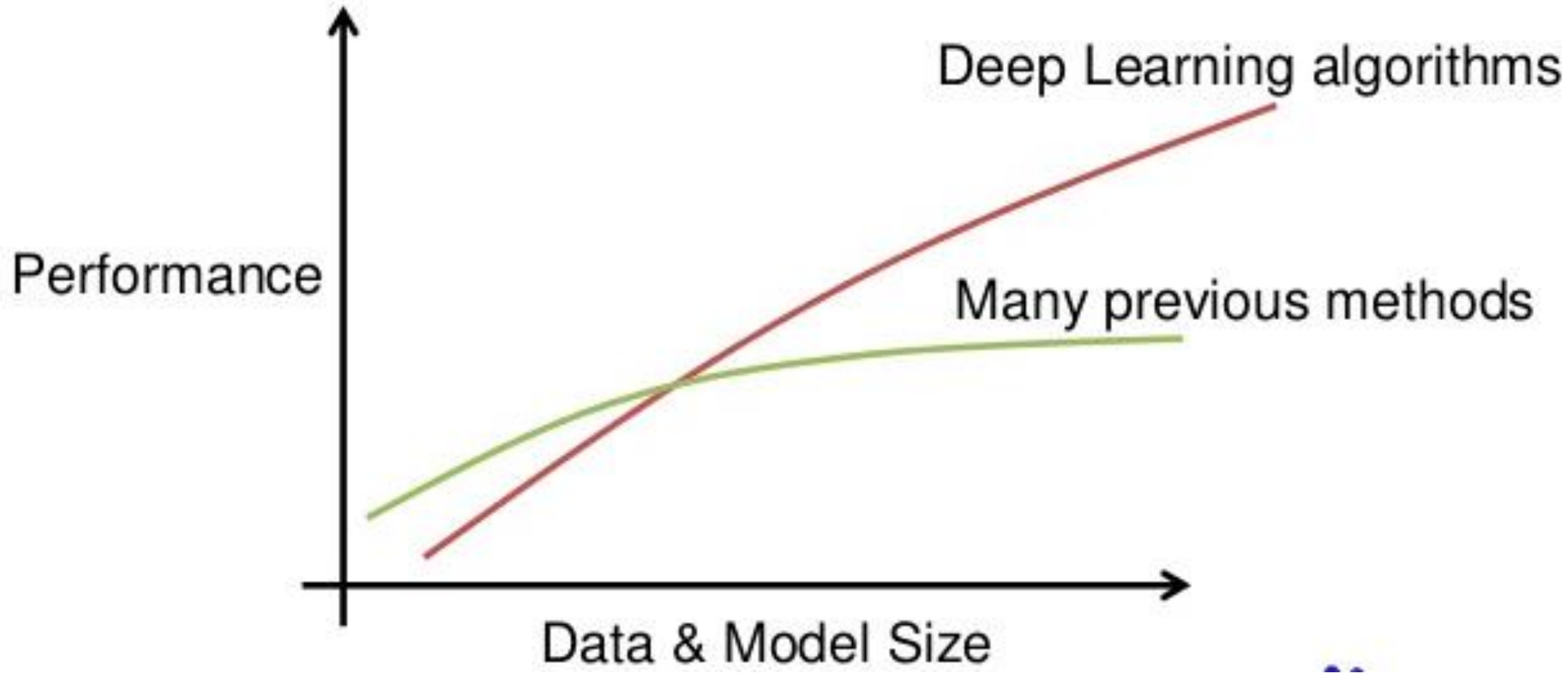
# Deep Learning for Activity Recognition

- Example of applying a convolutional neural network



# Machine Learning vs. Deep Learning

- Deep learning: the more data, the higher accuracy



# Types of Learning

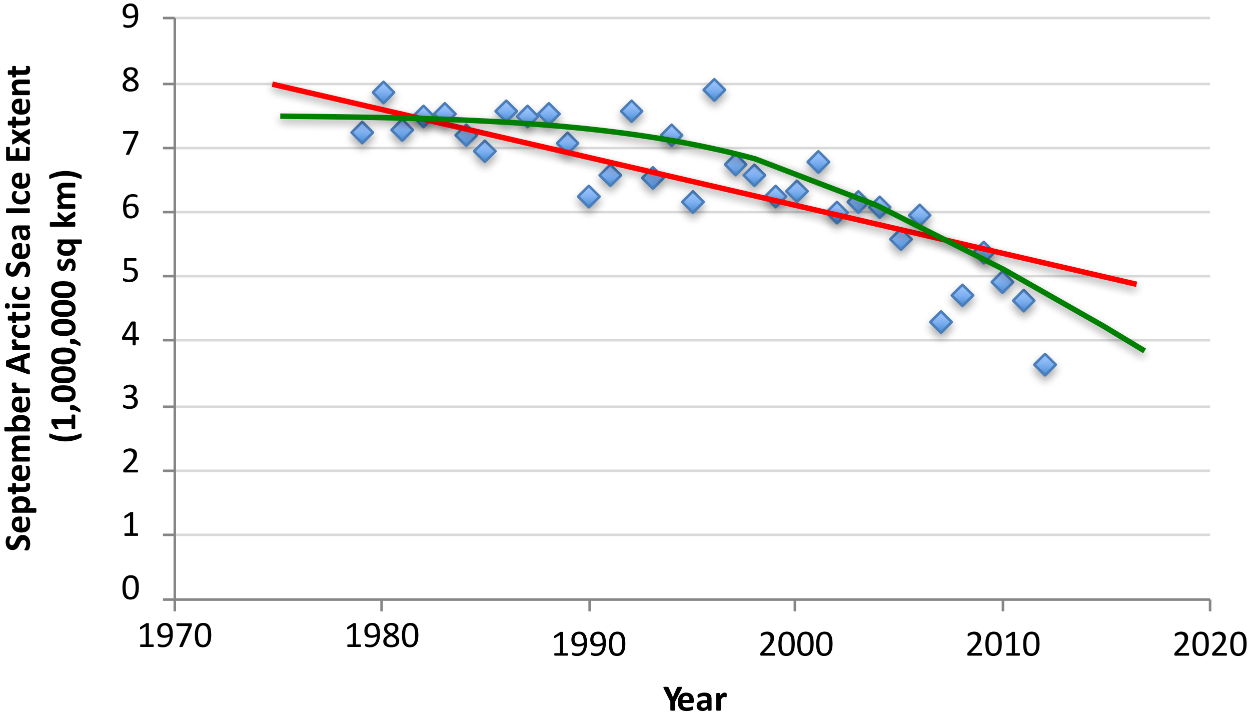
# Types of Learning

- **Supervised (inductive) learning**
  - Given: training data + desired outputs (labels)
- **Unsupervised learning**
  - Given: training data (without desired outputs)
- **Semi-supervised learning**
  - Given: training data + a few desired outputs
- **Reinforcement learning**
  - Rewards from sequence of actions

Based on slide by Pedro Domingos

# Supervised Learning: Regression

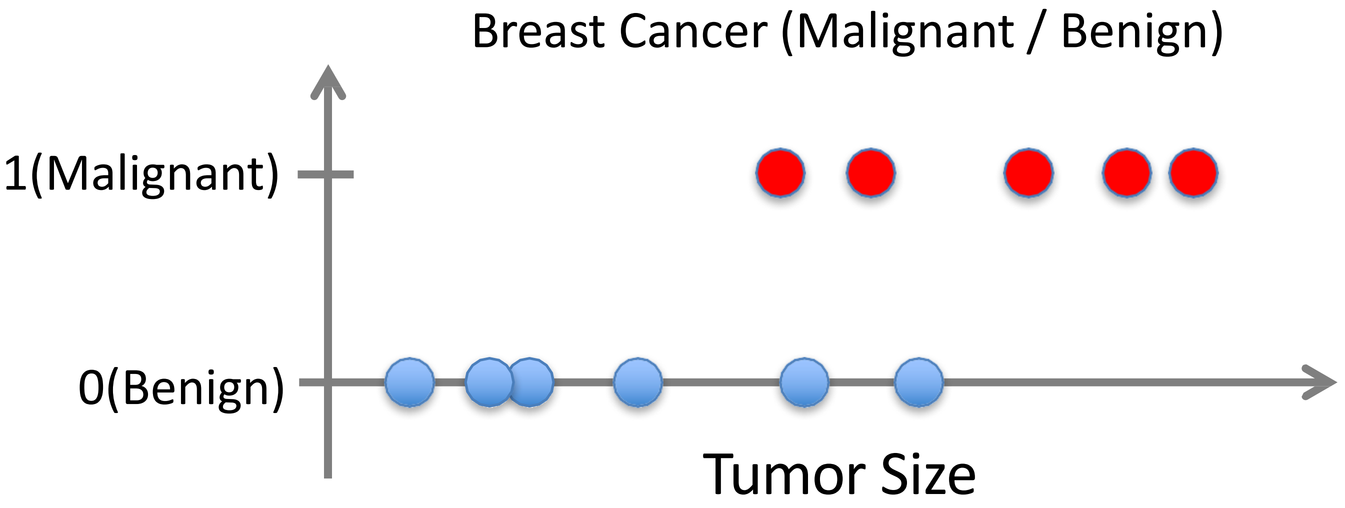
- Given $(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)$
- Learn a function $f(x)$ to predict $y$ given $x$
  - $y$ is real-valued == regression



Data from G. Witt. Journal of Statistics Education, Volume 21, Number 1 (2013)

# Supervised Learning: Classification

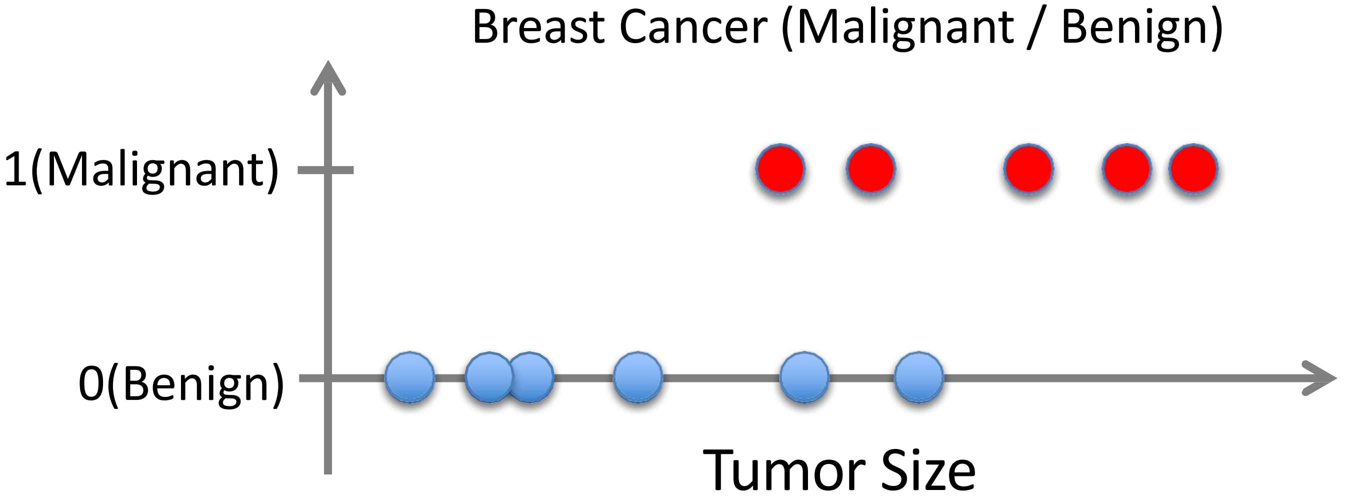
- Given ($x_1$, $y_1$), ($x_2$, $y_2$), ..., ($x_n$, $y_n$)
- Learn a function $f(x)$ to predict $y$ given $x$
  - $y$ is categorical == classification



Based on example by Andrew Ng

# Supervised Learning: Classification

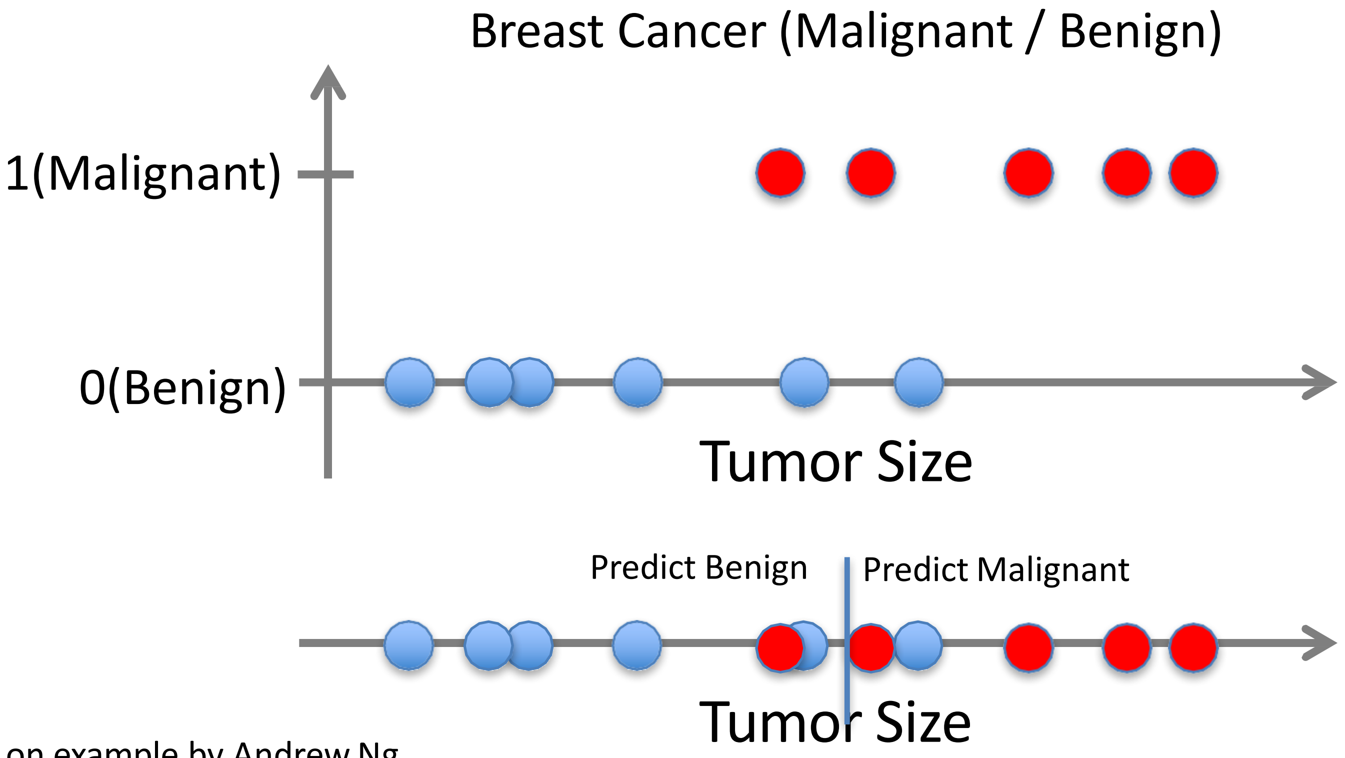
- Given $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$
- Learn a function $f(x)$ to predict $y$ given $x$
  - $y$ is categorical == classification



Based on example by Andrew Ng

# Supervised Learning: Classification

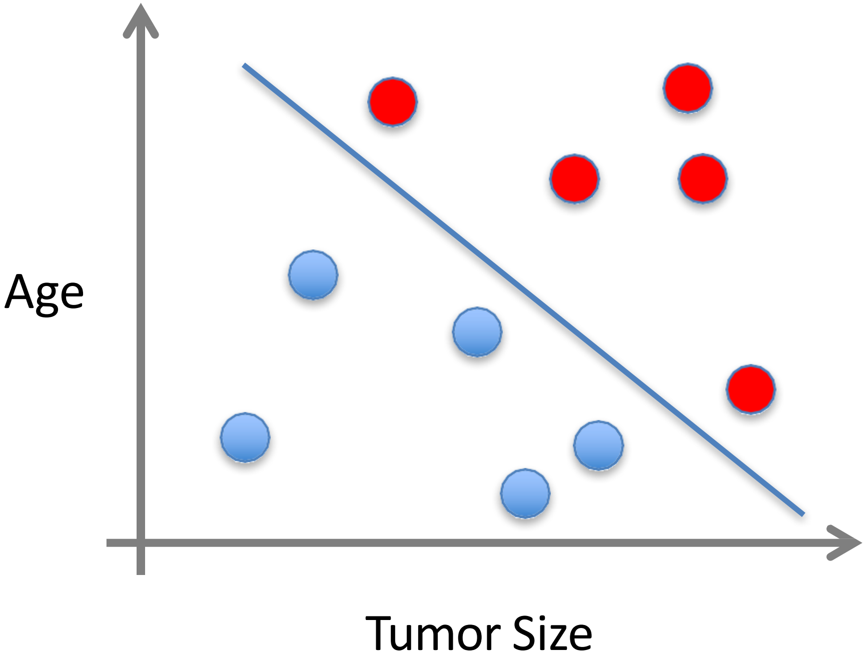
- Given $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$
- Learn a function $f(x)$ to predict $y$ given $x$
  - $y$ is categorical == classification



Based on example by Andrew Ng

# Supervised Learning

- $x$ can be multi-dimensional
  – Each dimension corresponds to an attribute



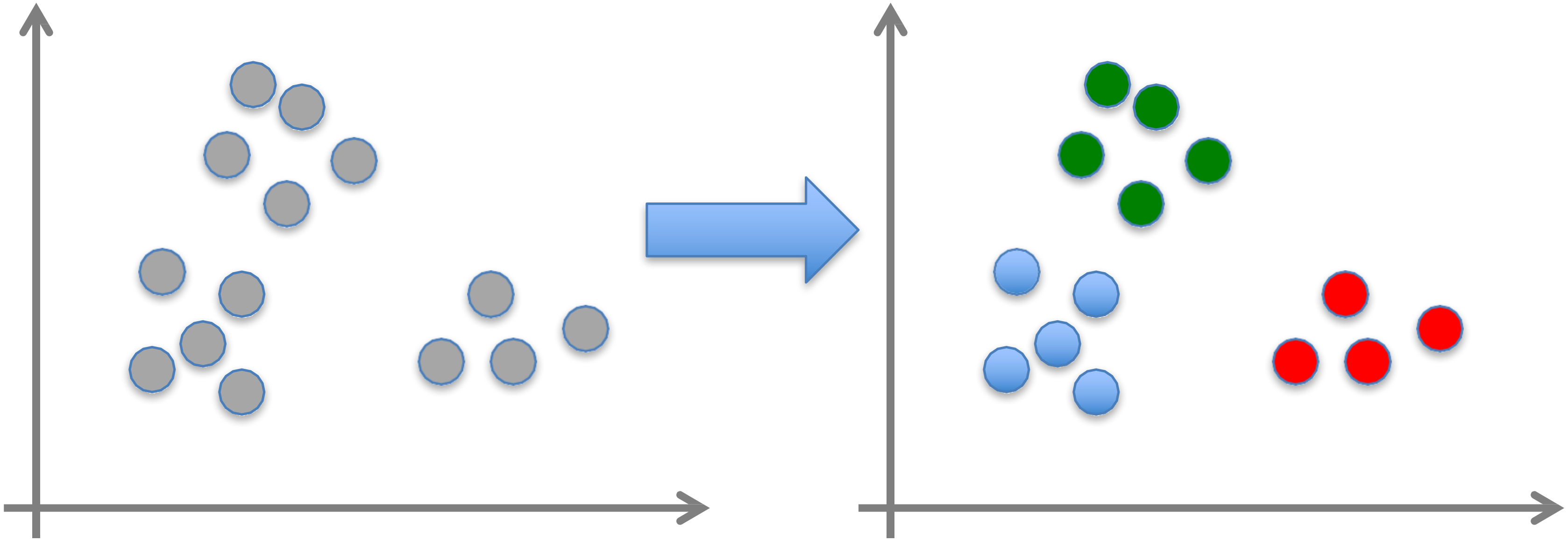
- Clump Thickness
- Uniformity of Cell Size
- Uniformity of Cell Shape

…

Based on example by Andrew Ng

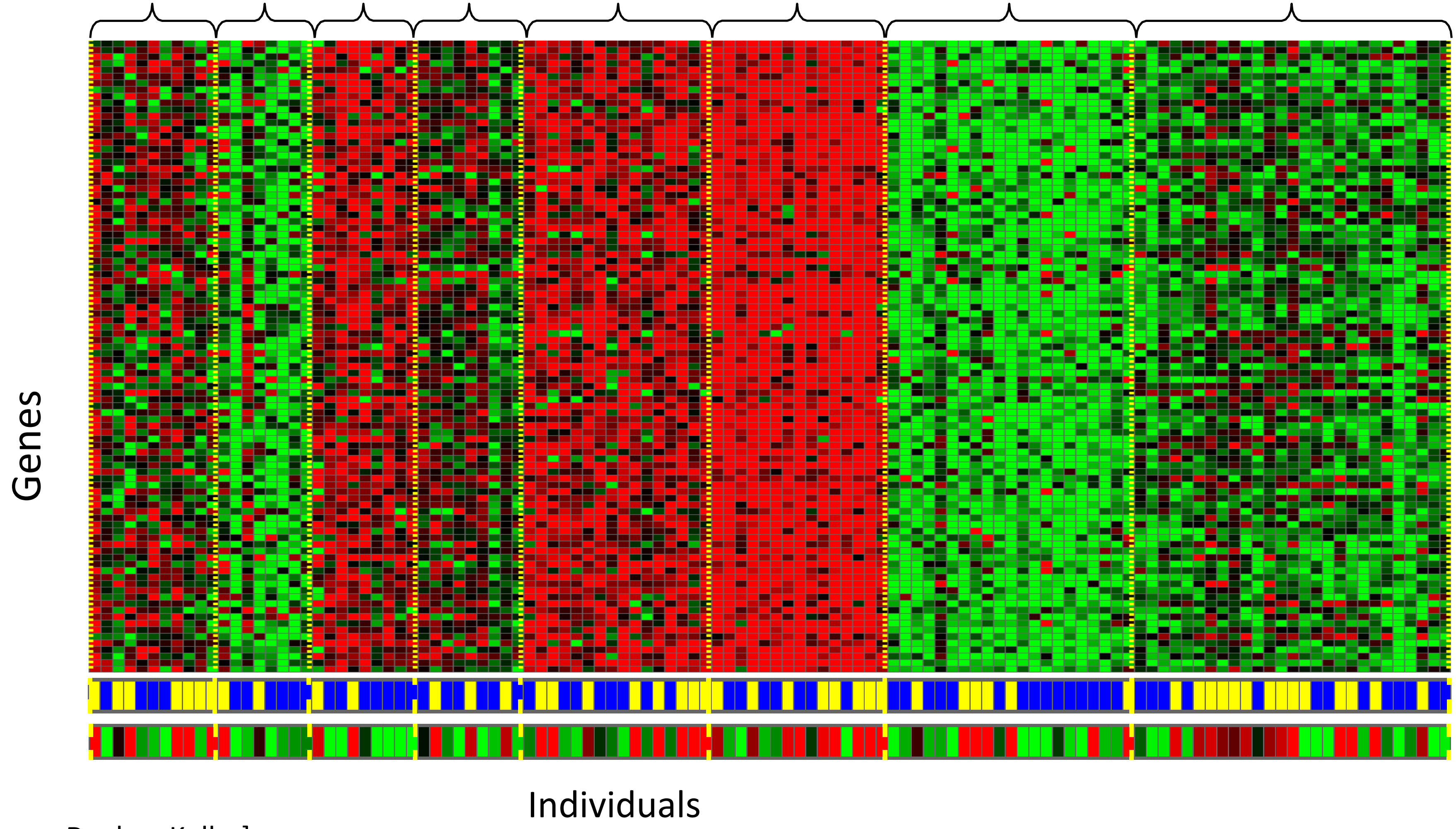
# Unsupervised Learning

- Given $x_1$, $x_2$, ..., $x_n$ (without labels)
- Output hidden structure behind the $x$'s
  - E.g., clustering



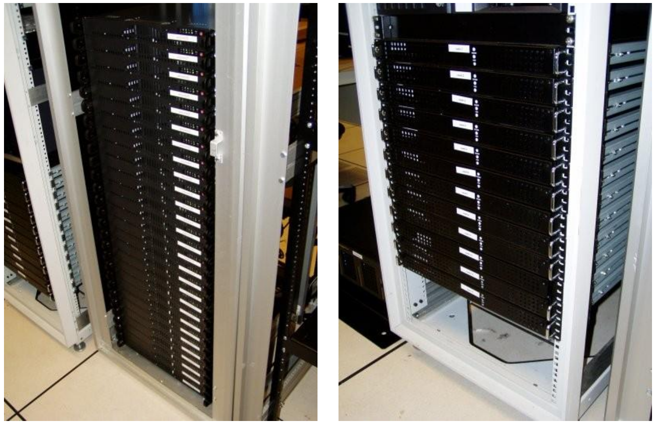
# Unsupervised Learning

Genomics application: group individuals by genetic similarity

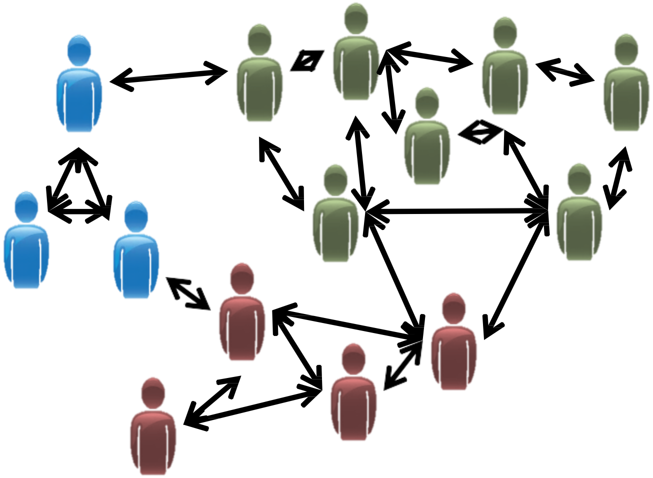


[Source: Daphne Koller]

# Unsupervised Learning



Organize computing clusters



Social network analysis

Market segmentation

Image credit: NASA/JPL-Caltech/E. Churchwell (Univ. of Wisconsin, Madison)

Astronomical data analysis

Slide credit: Andrew Ng

# Unsupervised Learning

- Independent component analysis – separate a combined signal into its original sources

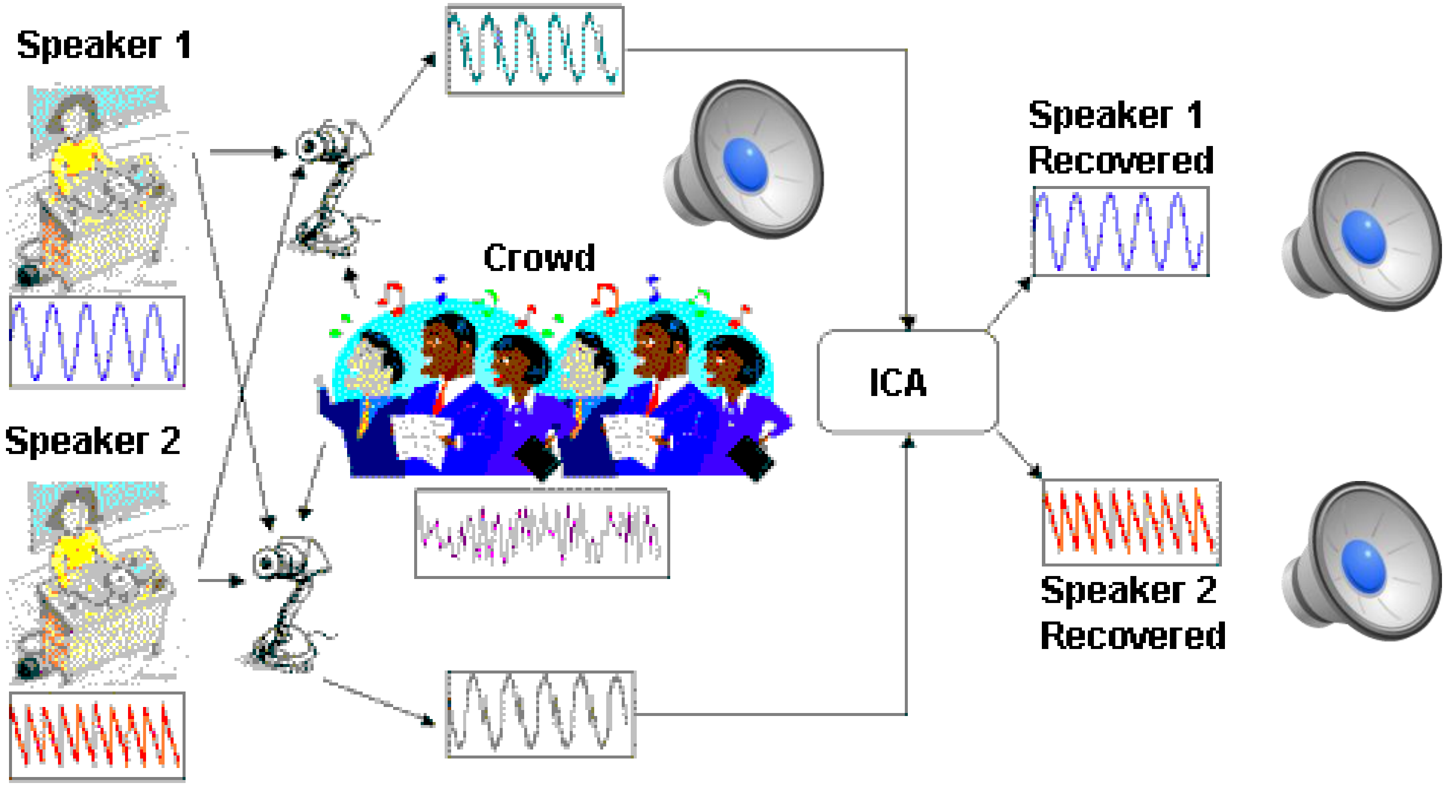


Image credit: statsoft.com Audio from http://www.ism.ac.jp/~shiro/research/blindsep.html

# Unsupervised Learning

- Independent component analysis – separate a combined signal into its original sources

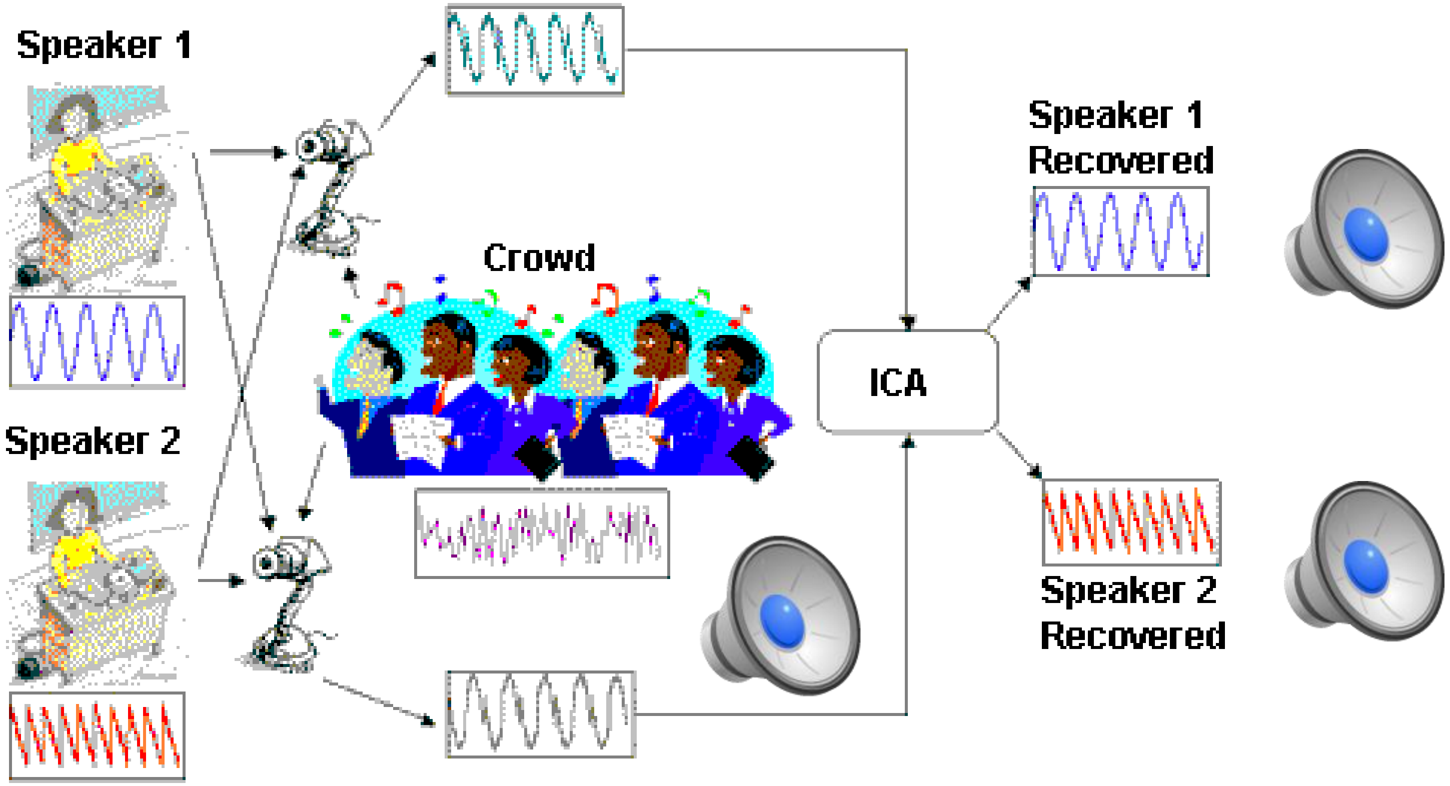
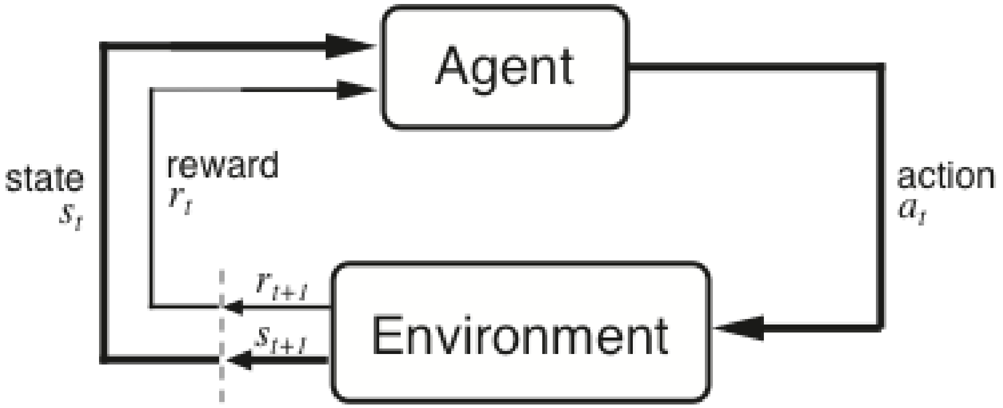


Image credit: statsoft.com Audio from http://www.ism.ac.jp/~shiro/research/blindsep.html

# Reinforcement Learning

- Given a sequence of states and actions with (delayed) rewards, output a policy
  - Policy is a mapping from states → actions that tells you what to do in a given state
- Examples:
  - Credit assignment problem
  - Game playing
  - Robot in a maze
  - Balance a pole on your hand

# The Agent-Environment Interface



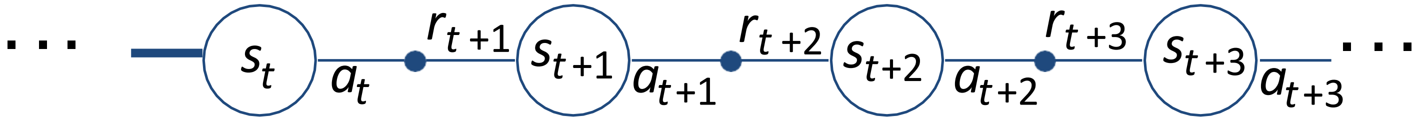
Agent and environment interact at discrete time steps : $t = 0, 1, 2, \ldots$

Agent observes state at step $t$ : $s_t \in S$

produces action at step $t$ : $a_t \in A(s_t)$

gets resulting reward : $r_{t+1} \in \Re$

and resulting next state : $s_{t+1}$



Slide credit: Sutton & Barto

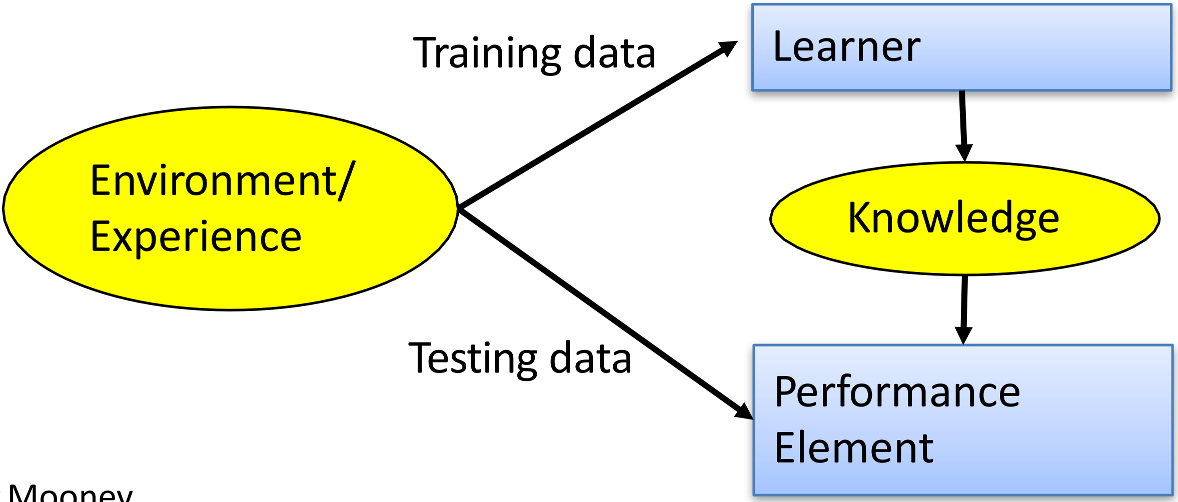
# Reinforcement Learning

https://www.youtube.com/watch?v=4cgWya-wjgY

# Framing a Learning Problem

# Designing a Learning System

- Choose the training experience
- Choose exactly what is to be learned
  - i.e. the ***target function***
- Choose how to represent the target function
- Choose a learning algorithm to infer the target function from the experience



Based on slide by Ray Mooney

# Training vs. Test Distribution

- We generally assume that the training and test examples are independently drawn from the same overall distribution of data
  - We call this “i.i.d” which stands for “independent and identically distributed”

- If examples are not independent, requires ***collective classification***

- If test distribution is different, requires ***transfer learning***

Slide credit: Ray Mooney

# ML in a Nutshell

- Tens of thousands of machine learning algorithms
  - Hundreds new every year

- Every ML algorithm has three components:
  - **Representation**
  - **Optimization**
  - **Evaluation**

Slide credit: Pedro Domingos

# Various Function Representations

- Numerical functions
  - Linear regression
  - Neural networks
  - Support vector machines
- Symbolic functions
  - Decision trees
  - Rules in propositional logic
  - Rules in first-order predicate logic
- Instance-based functions
  - Nearest-neighbor
  - Case-based
- Probabilistic Graphical Models
  - Naïve Bayes
  - Bayesian networks
  - Hidden-Markov Models (HMMs)
  - Probabilistic Context Free Grammars (PCFGs)
  - Markov networks

Slide credit: Ray Mooney

# Various Search/Optimization Algorithms

- Gradient descent
  - Perceptron
  - Backpropagation
- Dynamic Programming
  - HMM Learning
- Divide and Conquer
  - Decision tree induction
  - Rule learning
- Evolutionary Computation
  - Genetic Algorithms (GAs)
  - Genetic Programming (GP)
  - Neuro-evolution

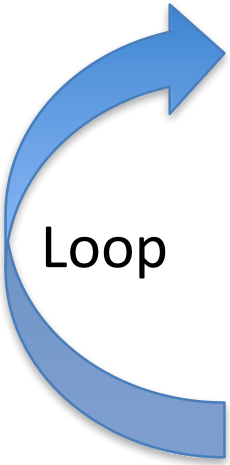
Slide credit: Ray Mooney

# Evaluation

- Accuracy
- Precision and recall
- Squared error
- Likelihood
- Posterior probability
- Cost / Utility
- Margin
- Entropy
- K-L divergence
- etc.

Slide credit: Pedro Domingos

# ML in Practice



- Understand domain, prior knowledge, and goals
- Data integration, selection, cleaning, pre-processing, etc.
- Learn models
- Interpret results
- Consolidate and deploy discovered knowledge

Based on a slide by Pedro Domingos

# Lessons Learned about Learning

- Learning can be viewed as using direct or indirect experience to approximate a chosen target function.
- Function approximation can be viewed as a search through a space of hypotheses (representations of functions) for one that best fits a set of training data.
- Different learning methods assume different hypothesis spaces (representation languages) and/or employ different search techniques.

Slide credit: Ray Mooney

# What We'll Cover in this Course

- **Supervised learning**
  - Decision tree induction
  - Linear regression
  - Logistic regression
  - Support vector machines & kernel methods
  - Model ensembles
  - Neural networks & deep learning

- **Unsupervised learning**
  - Clustering
  - Dimensionality reduction
- **Evaluation**
- **Applications**

Our focus will be on applying machine learning to real applications

# A Brief History of Machine Learning (Backup Slides)

# History of Machine Learning

- 1950s
  - Samuel’s checker player
  - Selfridge’s Pandemonium
- 1960s:
  - Neural networks: Perceptron
  - Pattern recognition
  - Learning in the limit theory
  - Minsky and Papert prove limitations of Perceptron
- 1970s:
  - Symbolic concept induction
  - Winston’s arch learner
  - Expert systems and the knowledge acquisition bottleneck
  - Quinlan’s ID3
  - Michalski’s AQ and soybean diagnosis
  - Scientific discovery with BACON
  - Mathematical discovery with AM

Slide credit: Ray Mooney

# History of Machine Learning (cont.)

- 1980s:
  - Advanced decision tree and rule learning
  - Explanation-based Learning (EBL)
  - Learning and planning and problem solving
  - Utility problem
  - Analogy
  - Cognitive architectures
  - Resurgence of neural networks (connectionism, backpropagation)
  - Valiant’s PAC Learning Theory
  - Focus on experimental methodology
- 1990s
  - Data mining
  - Adaptive software agents and web applications
  - Text learning
  - Reinforcement learning (RL)
  - Inductive Logic Programming (ILP)
  - Ensembles: Bagging, Boosting, and Stacking
  - Bayes Net learning

Slide credit: Ray Mooney

# History of Machine Learning (cont.)

- 2000s
  - Support vector machines & kernel methods
  - Graphical models
  - Statistical relational learning
  - Transfer learning
  - Sequence labeling
  - Collective classification and structured outputs
  - Computer Systems Applications (Compilers, Debugging, Graphics, Security)
  - E-mail management
  - Personalized assistants that learn
  - Learning in robotics and vision
- 2010s
  - Deep learning systems
  - Learning for big data
  - Bayesian methods
  - Multi-task & lifelong learning
  - Applications to vision, speech, social networks, learning to read, etc.
  - ???

Based on slide by Ray Mooney