Machine Learning

CSE 142

Xin (Eric) Wang

Monday, September 27, 2021

Today

- Introduction (cont.)
- Key machine learning concepts (Prologue)

Notes

- Weekly discussion sessions
  - 12:30-1:30pm on Tuesdays (it starts tomorrow!)
  - Recorded and posted on Canvas

- Sign up on Piazza if you haven't

What is a learning problem?

Review

- Learning involves improving performance
  - at some task ***T***
  - with experience ***E*** (i.e., data)
  - evaluated in terms of performance measure ***P***

- Example: learn to play checkers
  - Task ***T***: playing checkers well
  - Experience ***E***: game database, playing against itself
  - Performance ***P***: percent of games won against humans

- What exactly should be learned?
  - How might this be represented?
  - What specific algorithm(s) should be used?

Components of a learning problem

Review

- Task: the behavior or task that's being improved; e.g., classification, object recognition, acting in an environment
- Data: the experiences that are being used to improve performance in the task
- Measure of performance: How can the improvement be measured? Examples:
  - Provide more accurate solutions (e.g., increasing the accuracy in prediction)
  - Cover a wider range of problems
  - Obtain answers more economically (e.g., improved speed)
  - Simplify codified knowledge
  - New skills that were not presented initially

Machine learning ingredients

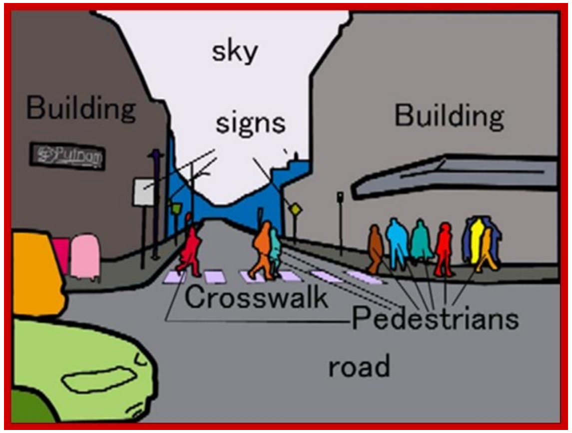
- Prior assumptions
  - What do we know a prior about the problem?
- Data
  - What kind of data do we have?
- Representation
  - How do we represent the data?
- Model / hypothesis space
  - What hypotheses are we willing to entertain to explain the data?
- Feedback / learning signal
  - What kind of learning signal do we have (labels, delayed)?
- Learning algorithm
  - How do we update the model (or set of hypotheses) from feedback?
- Evaluation
  - How well did we do? Should we change the model?

Key types of machine learning

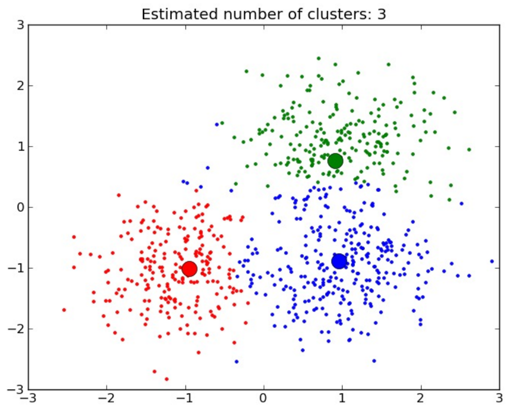
- Supervised learning
  - Provide *labeled* training data
  - Give the correct answers – input/output pairs
- Semi-supervised learning
  - Provide *some* labeled training data, other data unlabeled
  - Give *some* correct answers, others unknown
- Reinforcement learning
  - Provide occasional, usually delayed, feedback or reward
  - E.g., win or lose game (but no feedback on individual moves)
- Unsupervised learning
  - No direct learning signal or labels
  - The task is typically to find structure in the data (e.g., clustering, dimensionality reduction, density estimation, anomaly detection)

Key machine learning problems – examples

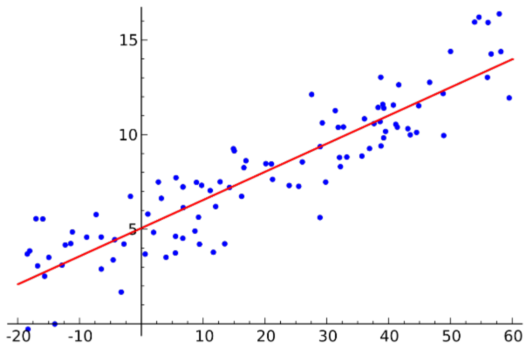
Classification / labeling



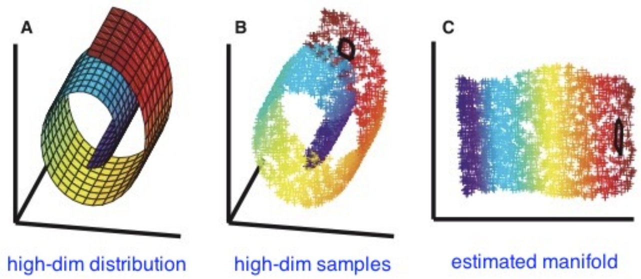
Clustering



Regression



Dimensionality reduction



Key machine learning problems – examples

Outputs		Supervised learning	Unsupervised learning
	Discrete / categorical	Classification / labeling Binary / multi-class	Clustering
	Continuous	Regression Real-valued prediction	Dimensionality reduction

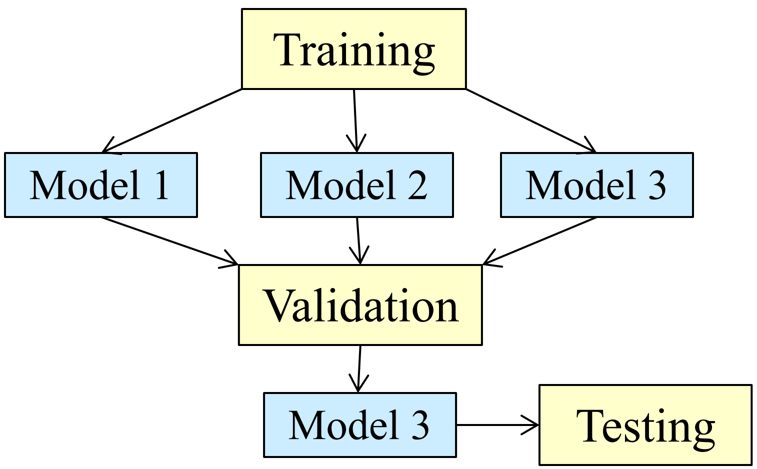
Regression = function estimation w/ scalar output

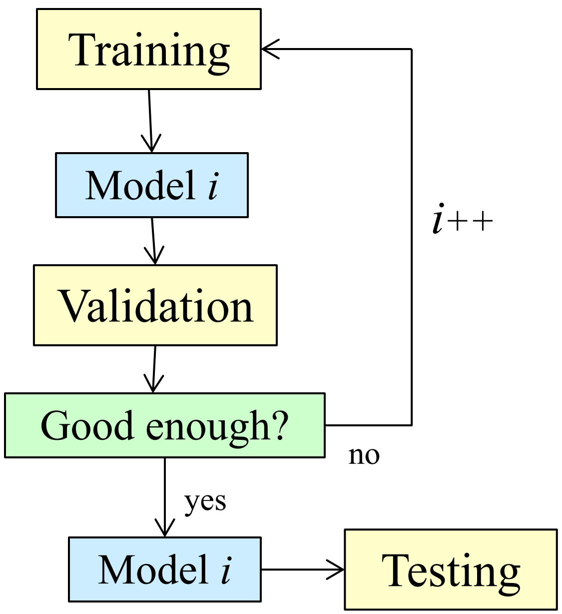
Logistic regression – the dependent variable is categorical

- Usually refers to binary classification

Training and testing a ML model

- We typically divide the dataset into three subsets:
  - Training data is used for learning the parameters of the models
  - Validation data is used to decide which model to employ
  - Test data is used to get a final, unbiased estimate of how well the model works



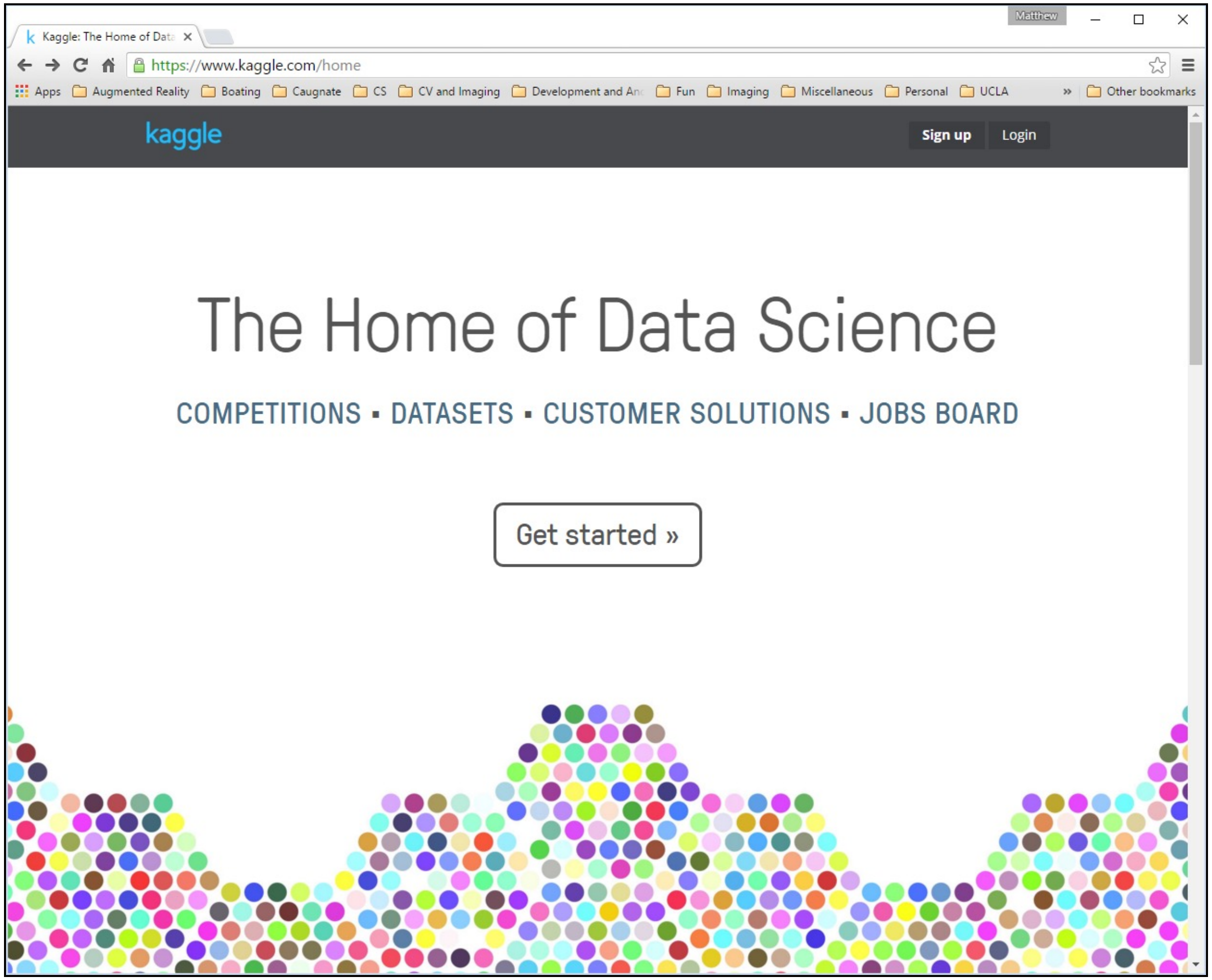


Sometimes reduced to training and testing
a single mode (no validation step)

Miscellaneous ML items

Kaggle

Data science competitions



The Netflix Prize

- The Netflix Prize sought to substantially improve the accuracy of predictions about how much someone is going to enjoy a movie based on their movie preferences
- $1M grand prize awarded in 2009
- Provided teams with:
  - Anonymous rating data
  - A prediction accuracy bar that was 10% better than what Netflix could do on the same training data set

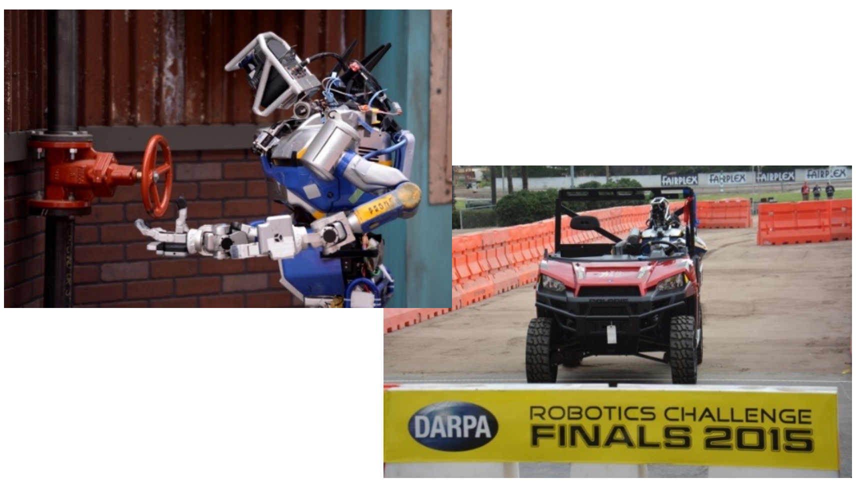
Winner: BellKor's Pragmatic Chaos

DARPA Grand Challenges

2004, 2005 Grand Challenges

2007 Urban Challenge

2012-2015 Robotics Challenge



2013 FANG Challenge

Fast Adaptable Next-Generation Ground Vehicle

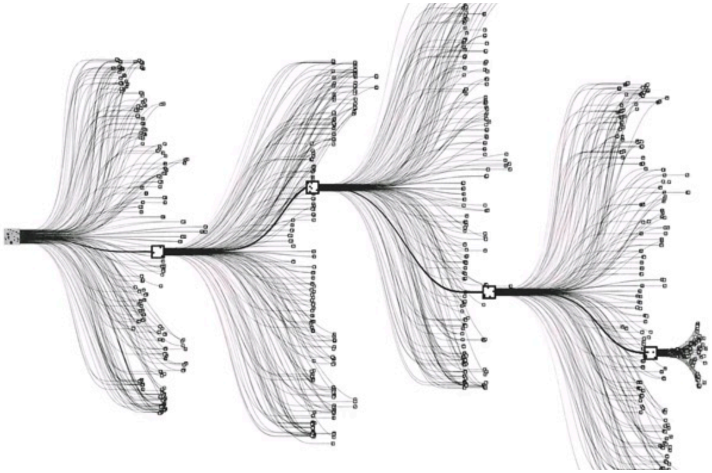
IBM Watson

- QA system in 2021
- Jeopardy! was selected as the ultimate test of the machine's capabilities because it relied on many human cognitive abilities traditionally seen beyond the capability of computers, such as:
  - The ability to discern double meanings of words, puns, rhymes, and inferred hints.
  - Extremely rapid responses
  - The ability to process vast amounts of information to make complex and subtle logical connections
- To meet this grand challenge, the Watson team focused on three key capabilities:
  - Natural language processing
  - Hypothesis generation
  - Evidence-based learning

Google AlphaGo

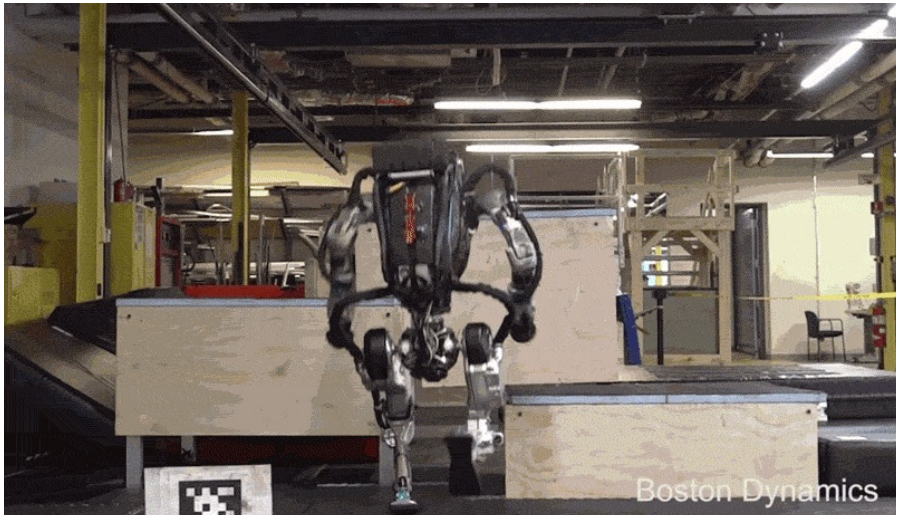
AlphaGo

- The first computer program to beat a professional player at the game of Go
- Defeated top Go player, Lee Sedol, in March 2016, 4-1
  - AlphaGo Zero 3-0 Ke Jie, No.1 world-ranking Go player, in May 2017



- Uses deep neural networks that are trained by a novel combination of supervised learning from human expert games, and reinforcement learning from games of self-play.

Boston Dynamics



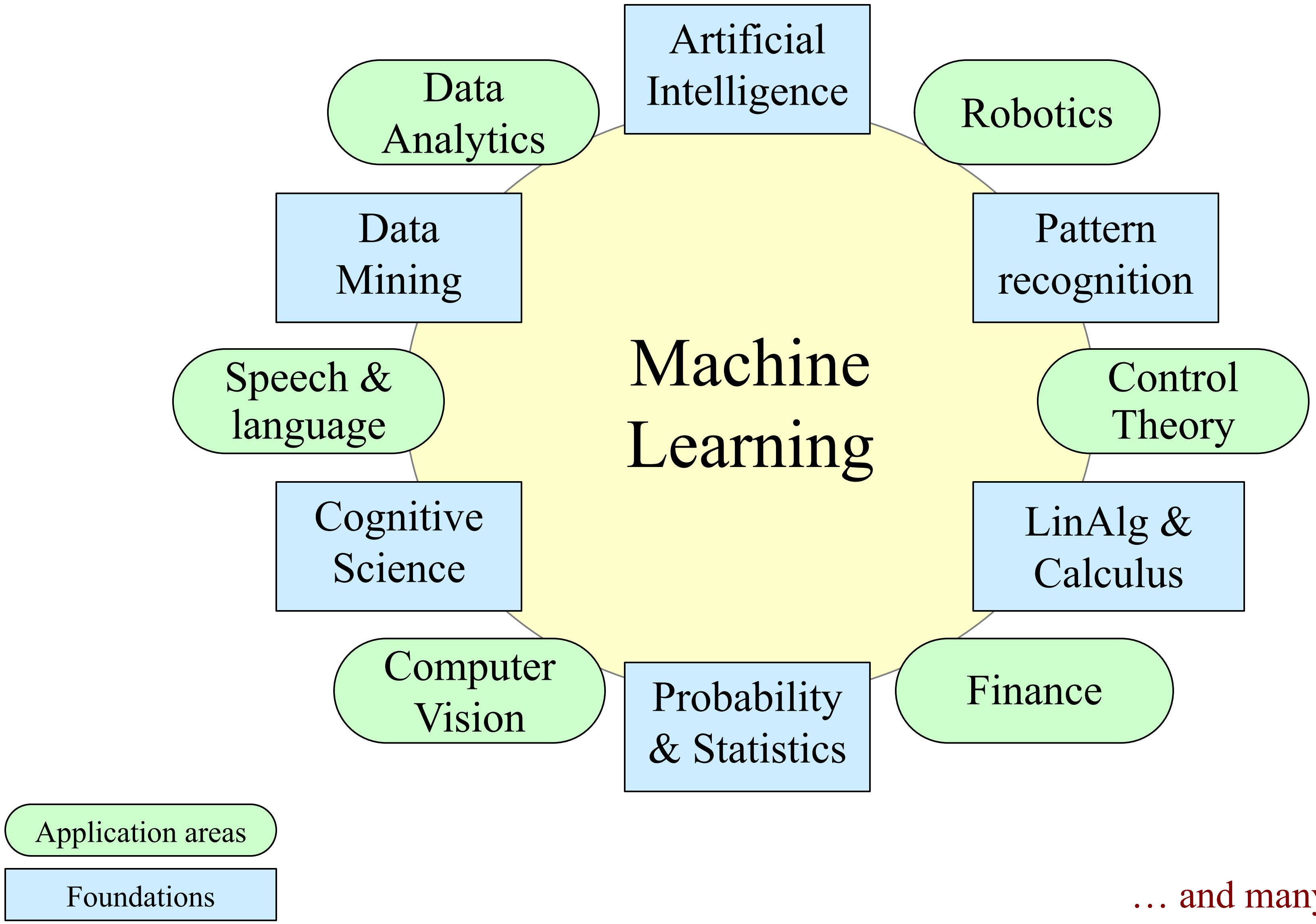
Machine learning is the design and analysis of algorithms that improve their performance at some task with experience.

What is machine learning?

“machine” + “learning”

Algorithms Programs Systems	…that…	Make sense of data Learn from data Improve with experience Adapt to the user/situation

Related topics

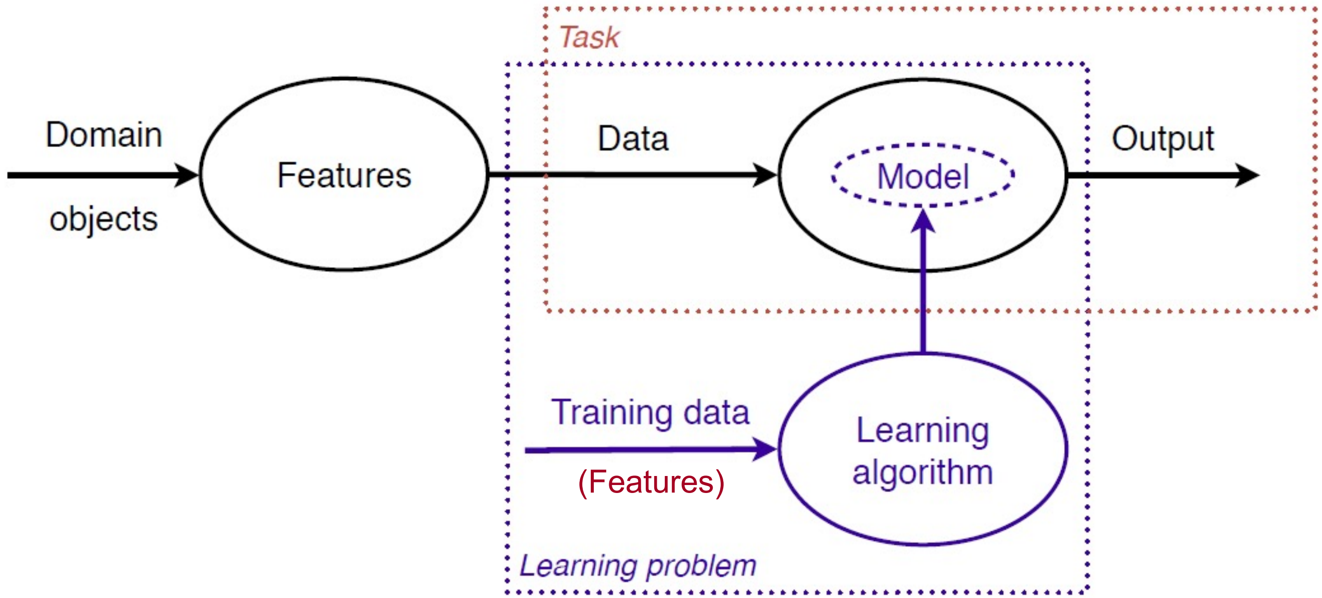


… and many more!

Some key machine learning concepts

Textbook prologue

A machine learning system



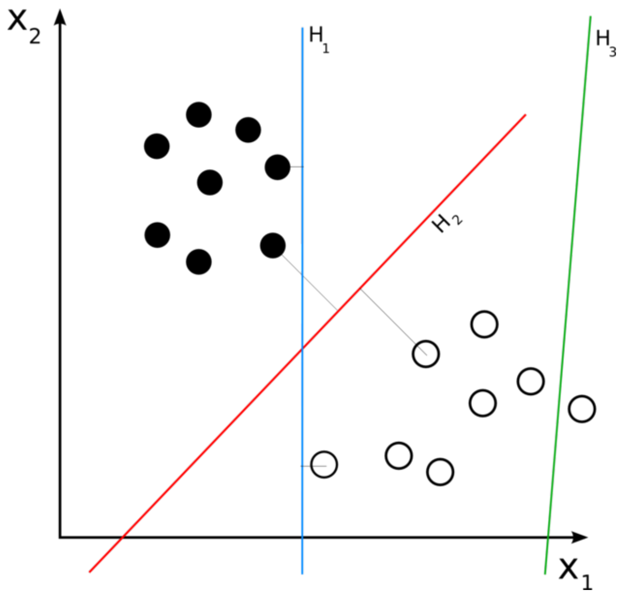
A **task** requires an appropriate mapping – a **model** – from data described by **features** to outputs. Tasks are addressed by **models**.

Obtaining such a model from training data is what constitutes a **learning problem**.

Learning problems are solved by **learning algorithms** that produce **models**.

One ML model: Linear classification

- Outputs a classification (one of N possible classes) based on the value of a linear combination of the characteristics (features)



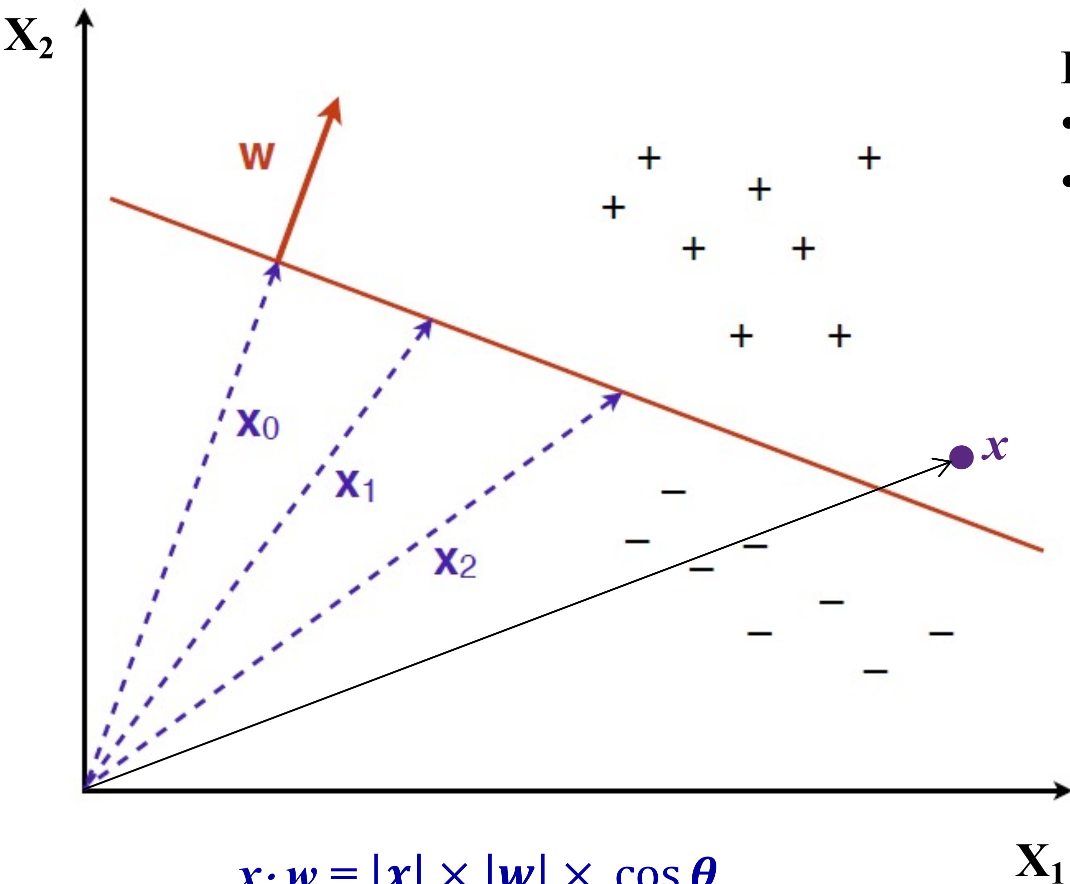
Example with 2 features – 2D feature vector (x_1, x_2):

- Linear classifiers H_1 and H_2 successfully partition the two classes of dots
- H3 does not

QUIZ: Which is better, H_1 or H_2? Why?

Linear classification

Goal: Find a linear decision boundary



$$\boldsymbol{x} \cdot \boldsymbol{w} = |\boldsymbol{x}| \times |\boldsymbol{w}| \times \cos \boldsymbol{\theta}$$

$$\boldsymbol{x} \cdot \boldsymbol{w} = \boldsymbol{x}^T \boldsymbol{w} = \boldsymbol{w}^T \boldsymbol{x}$$

$$\boldsymbol{x_0} \cdot \boldsymbol{w} = \boldsymbol{x_1} \cdot \boldsymbol{w} = \boldsymbol{x_2} \cdot \boldsymbol{w} = t$$

How to determine if a feature vector $\boldsymbol{x}$ is on the + or – side of the line?

Evaluate the dot product of $\boldsymbol{x}$ and $\boldsymbol{w}$:

- If $\boldsymbol{x} \cdot \boldsymbol{w} > t$, then +
- Otherwise –

2 features means 2D classification and a 1D classification boundary

N features means N dimensional classification and an N-1 dimensional classification boundary

Dimensions	Linear boundary
1	Point
2	Line
3	Plane
>3	Hyperplane

Homogeneous coordinates

- Instead of writing $\boldsymbol{x} \cdot \boldsymbol{w} > t$, let's use homogeneous coordinates to simplify the decision rule to $\boldsymbol{x}^\circ \cdot \boldsymbol{w}^\circ > 0$

$$\boldsymbol{x}^\circ = \begin{bmatrix} \boldsymbol{x} \\ 1 \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \\ 1 \end{bmatrix}$$

$$\boldsymbol{w}^\circ = \begin{bmatrix} \boldsymbol{w} \\ -t \end{bmatrix} = \begin{bmatrix} w_1 \\ w_2 \\ -t \end{bmatrix}$$

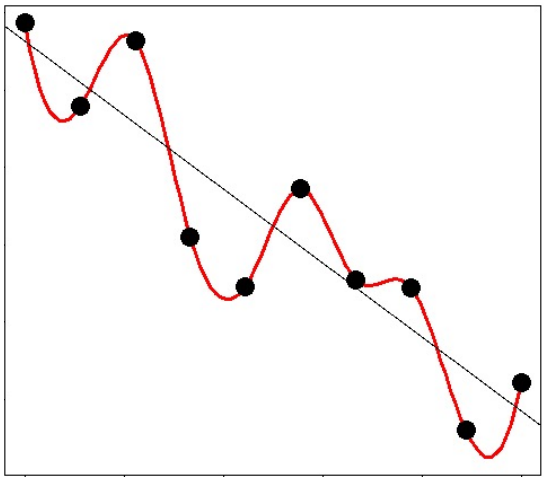
Note: I represent homogeneous coordinates a little differently from the textbook!

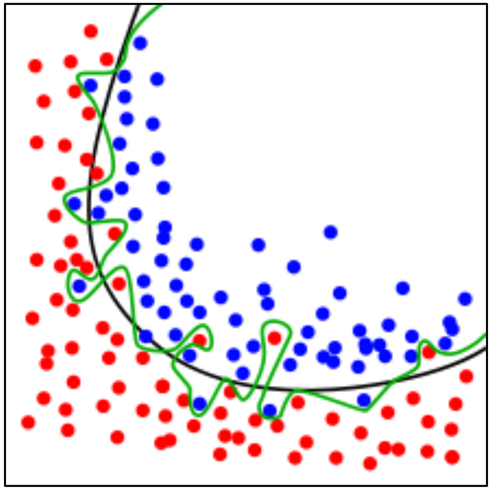
- Homogeneous coordinates embeds an N-dimensional representation in an N+1-dimensional space
- Advantage: The decision boundary passes through the origin of the extended coordinate system
  - Simplifies the math (or at least the notation)

Overfitting

- Overfitting and generalization are important concepts in machine learning

- Overfitting: Learning that results in good performance on the training data but poor performance on the real task
  - Example: Memorization or lookup table
  - Example: Fitting a model to the data that has more parameters than needed





Generalization

- We want machine learning solutions that ***generalize*** to the range of inputs/data that will be seen – not just solutions that work well on the training data
- The real aim of machine learning is to do well on test data that is not known during learning
- Choosing values for the parameters that minimize the error <u>on the training data</u> is not necessarily the best policy.

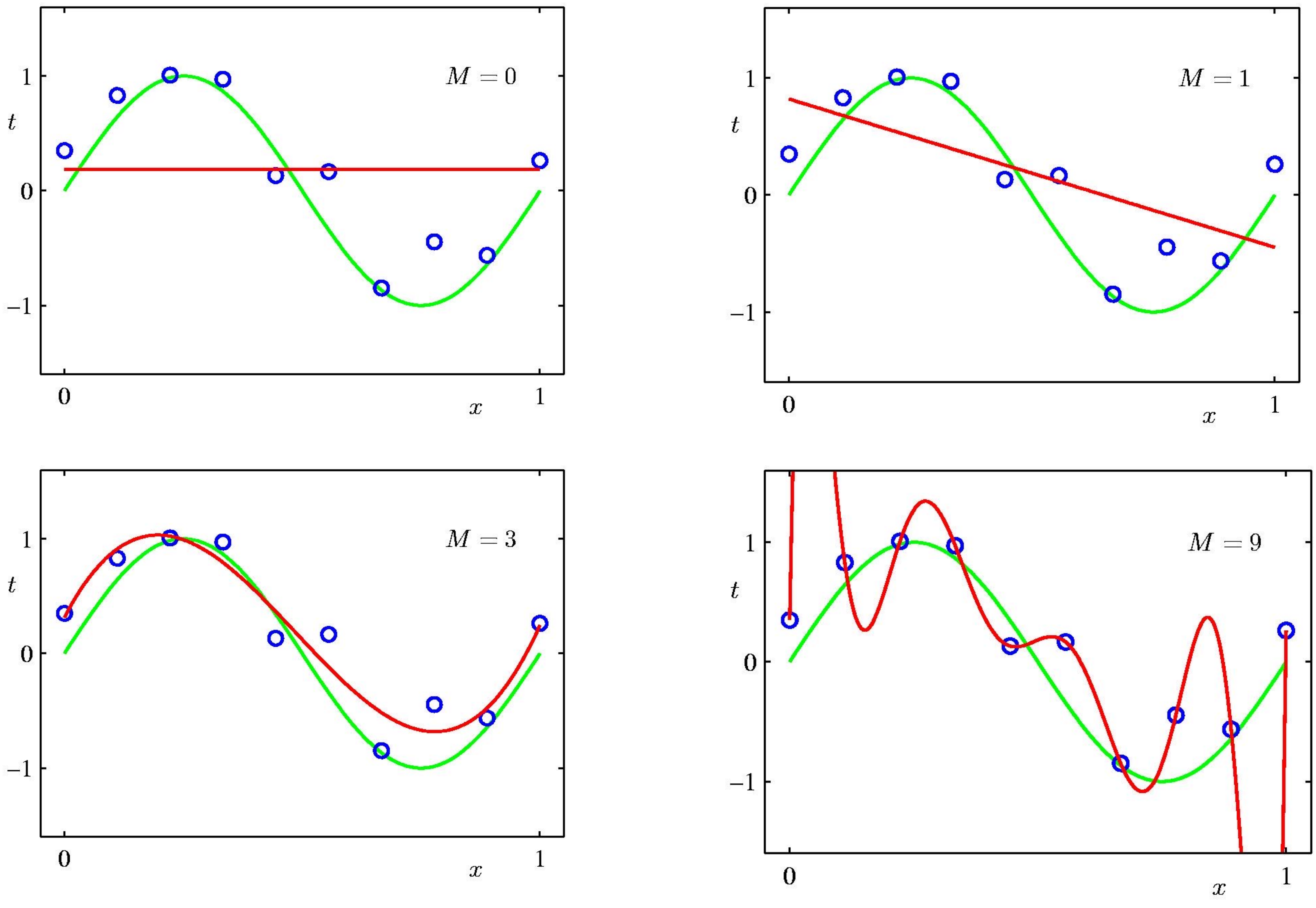
- We want the learning machine to model the true regularities in the data and to ignore the noise in the data
  - But the learning machine does not know which regularities are real and which are accidental quirks of the particular set of training examples we happen to have! So we have to help….

Which model fits the data best?

Real data = sinusoid

M = degree of polynomial fit

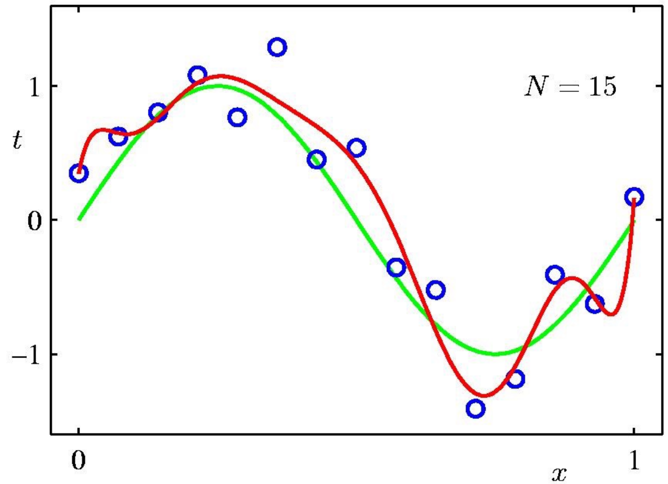
E.g., in a regression task



Reducing model complexity

If we penalize polynomials that have a high number of coefficients, we will get smoother (less wiggly) solutions

- These tend to generalize best



Ockham's Razor (aka **Occam's Razor**):
Prefer the simplest hypothesis that is consistent with the data

Note: This is a heuristic, not a logical principle or a scientific result

The curse of dimensionality

- Machine learning often involves very high-dimensional data
  - In general, the required amount of training data (and computational resources) scales exponentially with the dimensionality
- Sometimes the *intrinsic dimensionality* is lower and the problem is feasible if the relevant dimensions can be identified (and irrelevant dimensions ignored)
  - E.g., through dimensionality reduction methods
- How much training data is enough?
  - This is a difficult question in machine learning
- With a fixed number of training samples, the predictive power reduces as the dimensionality increases
  - This is known as the Hughes Effect
- Distance measures lose their usefulness in high dimensionality
  - Thus affecting clustering, classification, and other ML measures

Distance measures

- How similar are two faces? Two chess board configurations? Two countries' economies? Two DNA sequences?
  - We need ways to measure such things
- General assumption in ML: Similarity is a function of distance
  - But how to measure distance?
  - In what space? (What are the features?)
  - What's relevant and what's irrelevant in the data?
- Distance measures
  - Compute N features, resulting in a feature vector of N elements
  - The feature vector is then the only information the systems knows about the data sample
  - Define a distance measure between two feature vectors

How do we typically measure distance?

Some common distance measures

- Manhattan (L1) distance: $$d(x,y) = \sum_{i=1}^{d} |x_i - y_i|$$
  aka Cityblock distance
- Euclidian (L2) distance: $$d(x,y) = \|x - y\| = \left(\sum_{i=1}^{d} (x_i - y_i)^2\right)^{1/2}$$
- Minkowski (L_p) distance: $$d(x,y) = \left(\sum_{i=1}^{d} |x_i - y_i|^p\right)^{1/p}$$

Also:
- Mahalanobis distance
- Hamming distance
- Edit distance
- …and more…