Supervised statistical machine learning: principles and applications

Devika Subramanian

Rice University

devika@rice.edu

Aug 15 and 16, 2019

Instructional staff

- Instructor: Devika Subramanian, devika@rice.edu
- Teaching Assistants: Eleni Litsa (el24@rice.edu) and Jayvee Abella (jayvee.r.abella@rice.edu)



Class schedule

Date	Time	Topic
Aug 15	8:30 - 10:30	Intro to supervised machine learning
Aug 15	10:30 - 10:45	Coffee break
Aug 15	10:45 - 12:15	Linear models
Aug 15	12:15 - 1:30	Lunch
Aug 15	1:30 - 3:30	Non-linear models: part I
Aug 15	3:30 - 3:45	Coffee break
Aug 15	3:45 - 5:00	Non-linear models: part II
Aug 16	8:30 - 10:30	Deep learning: part I
Aug 16	10:30 - 10:45	Coffee break
Aug 16	10:45 - 12:15	Deep learning: part II

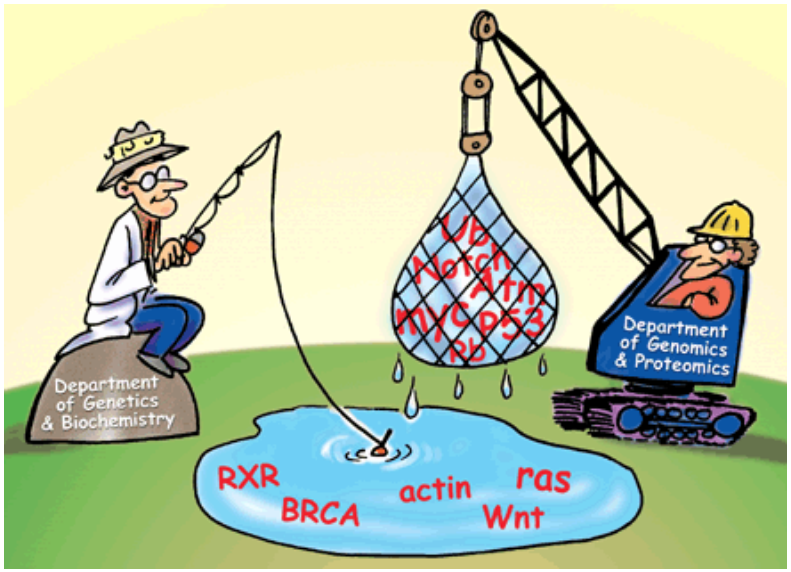
Goals of the course

- introduce several state-of-the-art algorithms in supervised statistical machine learning.
- show how each of these algorithms applies to real-world problems in science and engineering.
- provide practice in applying algorithms by solving real problems.

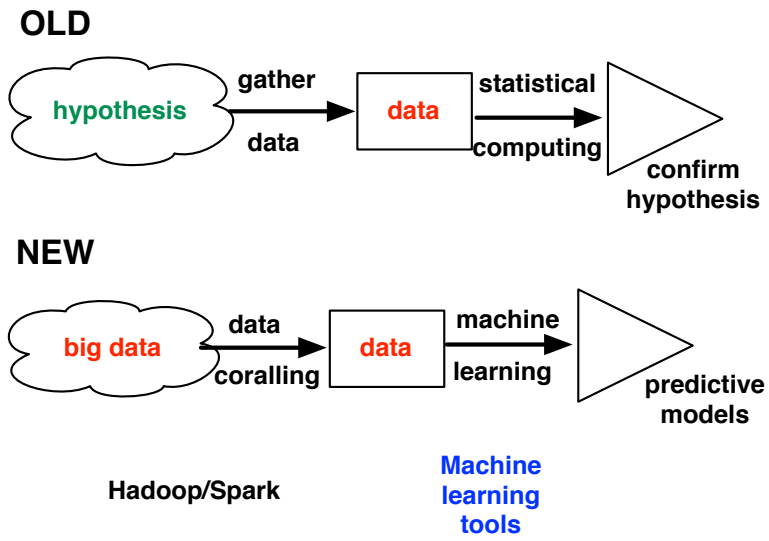
Perspective: the big data era

- How do we exploit the massive amounts of data we gather to help answer questions that matter to us?
  - I trillion web pages, 1 hour of video/second uploaded on YouTube, etc.
  - 1M transactions per hour on Walmart which has databases of size greater than 2.5 petabytes
  - . . . but beware of the long tail problem!
- Machine learning is the primary technological enabler for data-driven science and data-driven decision-making!

Science in the big data era

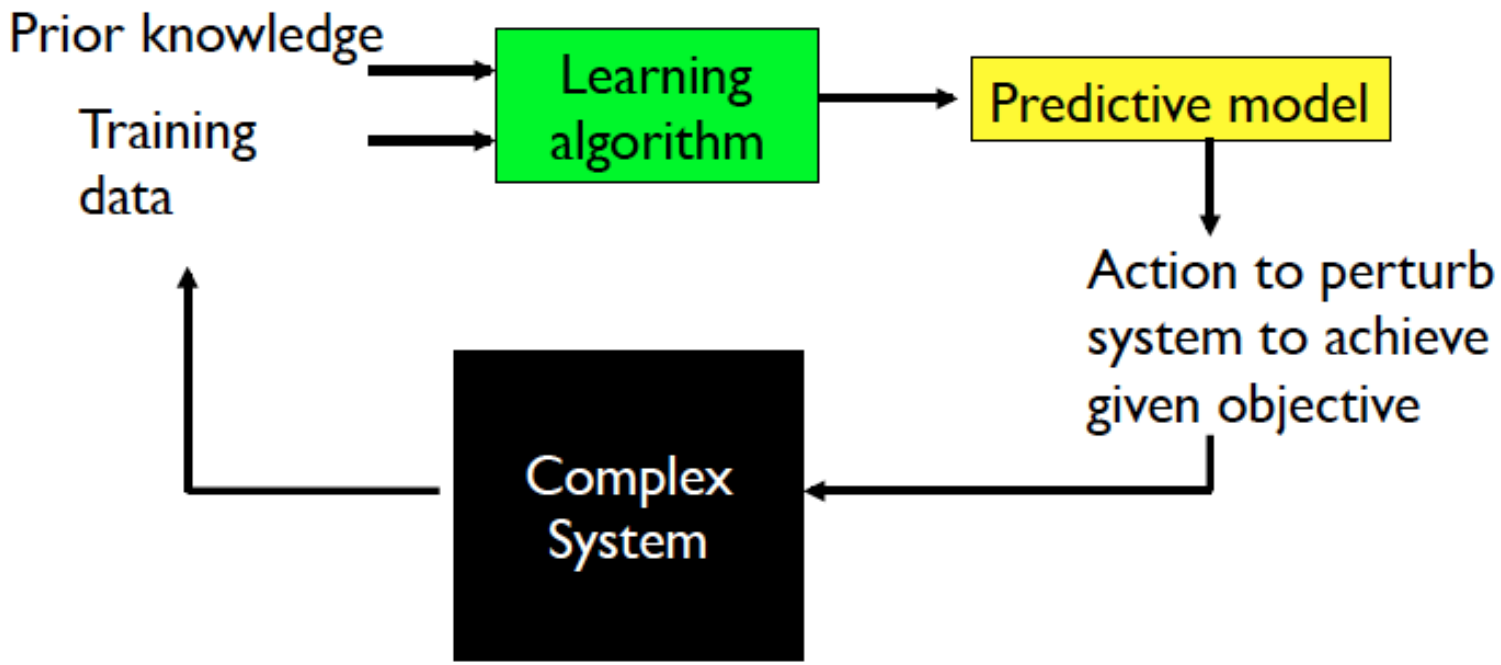


Stanley Fields, Science 2005



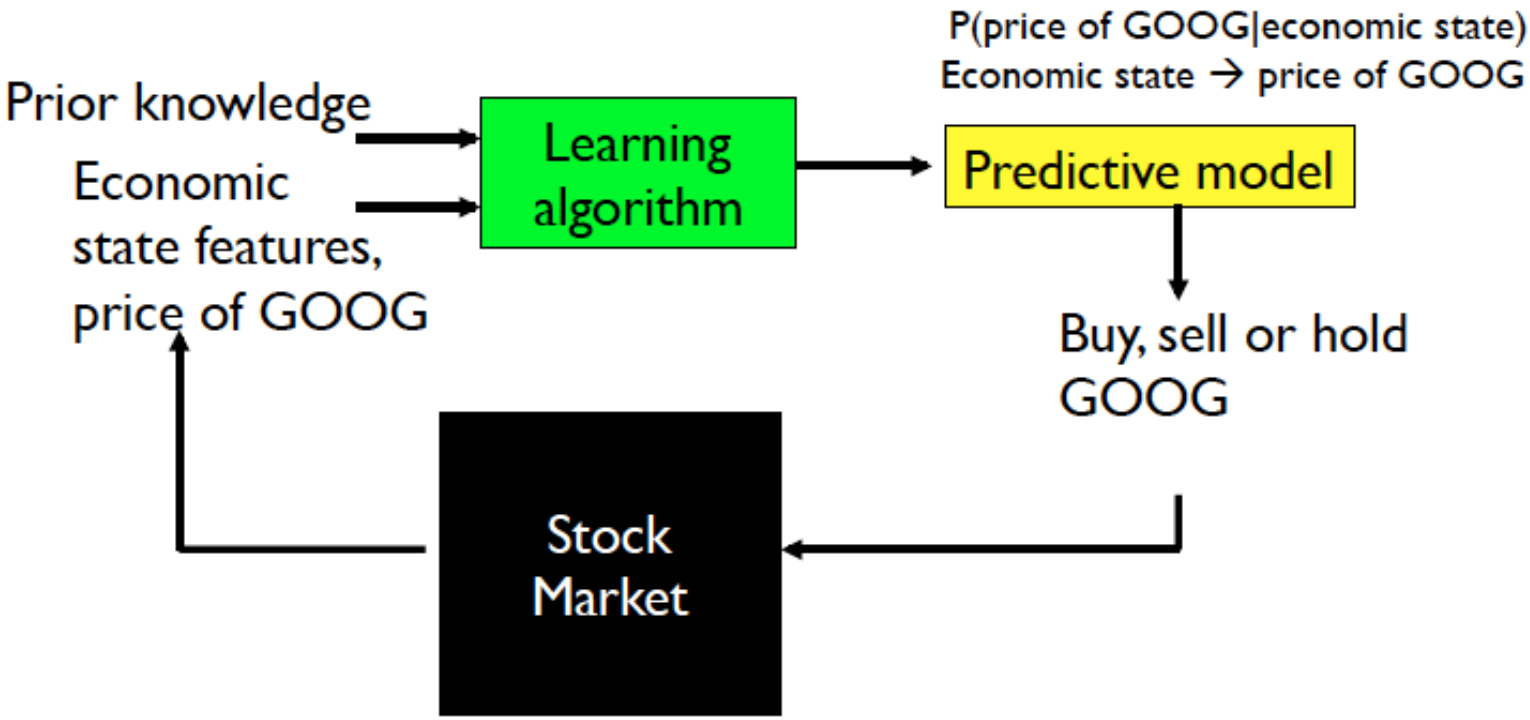
Concept: Jeff Phillips, Utah

What is statistical machine learning?



Observe a complex system and build models to predict its response.

An example from finance



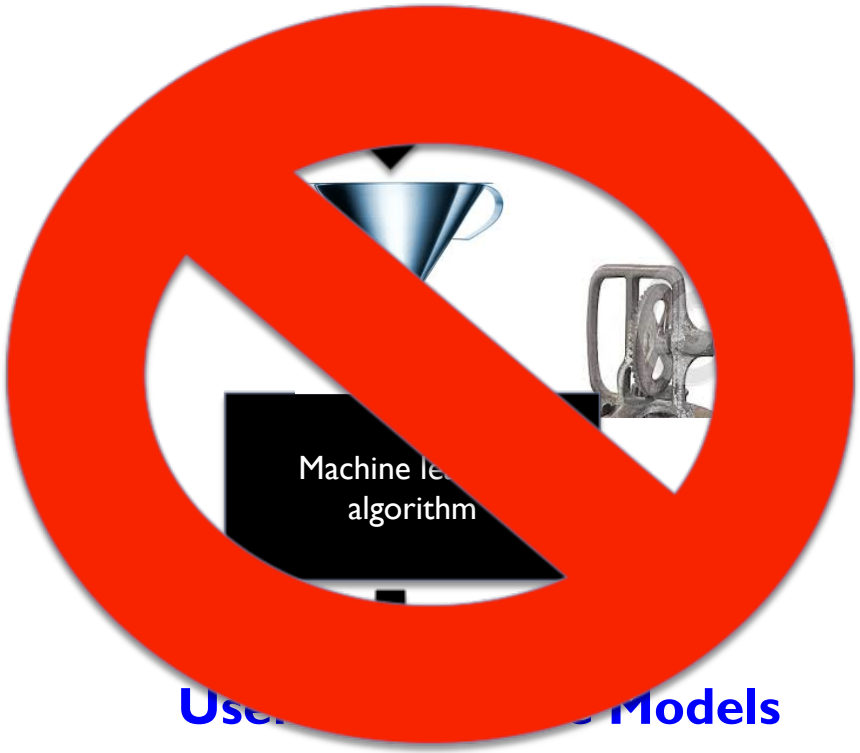
Observe a complex system and build models to predict its response.
To evaluate your predictive model: check if you are making money!

Machine learning successes

- Amazon/Netflix recommender systems.
- Fraud detection/mortgage lending
- Handwriting recognition in smartphones/face detection in cameras
- Market prediction
- Self-driving cars, route planning
- Speech recognition: Alexa, Siri, Google
- Alpha Go, Poker playing, Atari gamebots
- Google diabetic retinopathy detector

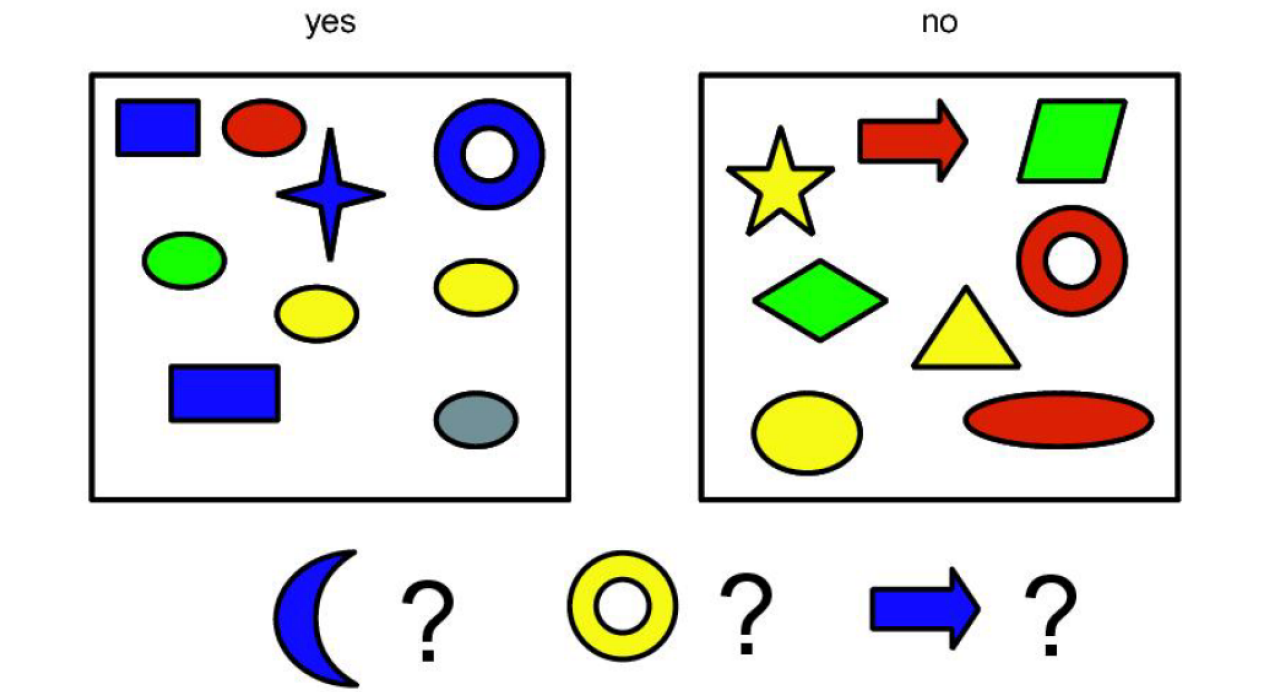
What machine learning is not (yet)!

What machine learning is not (yet)!



Use Models

Supervised machine learning: classification



Training data: (x, y) where x is an object, and $y \in \{yes, no\}$
Learning objective: classify new objects x to $\{yes, no\}$

Source: Kevin Murphy: Machine Learning: a probabilistic perspective, MIT Press

Representation of examples: hand-made features

D features (attributes)

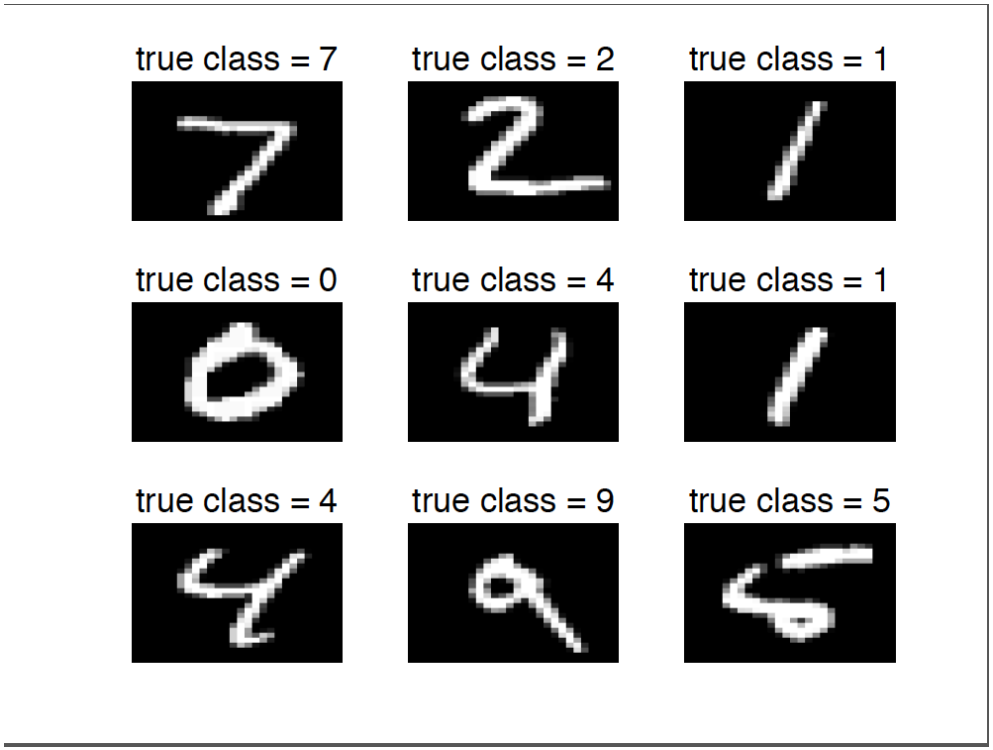
N cases

Color	Shape	Size (cm)	Label
Blue	Square	10	1
Red	Ellipse	2.4	1
Red	Ellipse	20.7	0

Data matrix **X** Label vector **y**

Source: Kevin Murphy: Machine Learning: a probabilistic perspective, MIT Press

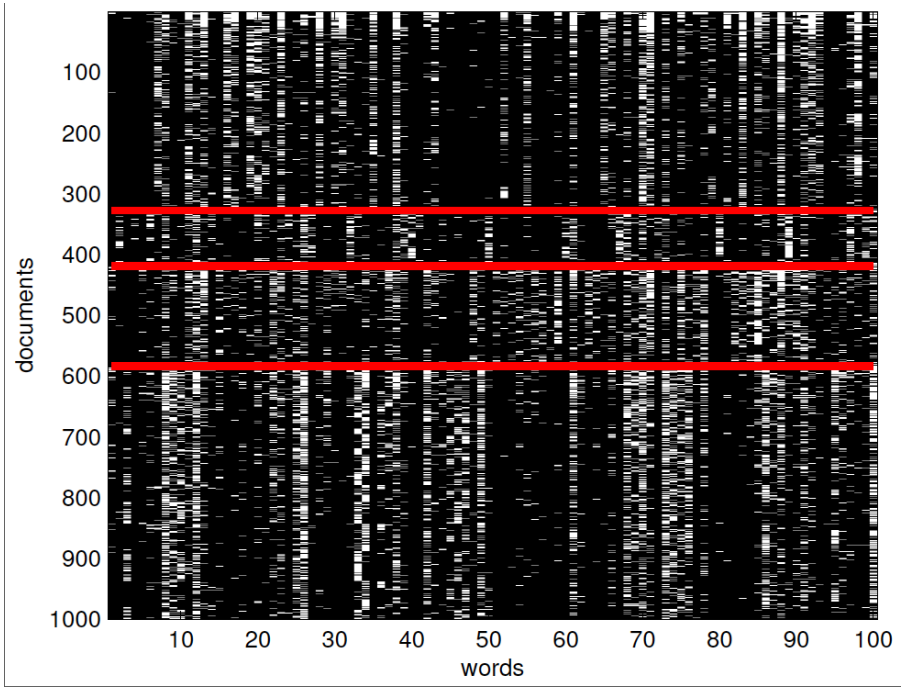
Representation of examples: raw pixels



$\mathbf{X}$: $m \times d$ array, $\mathbf{y}$: $m \times 1$ vector, where $m = 9$, $d = 784$, and each $y \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$

Source: Kevin Murphy: Machine Learning: a probabilistic perspective, MIT Press

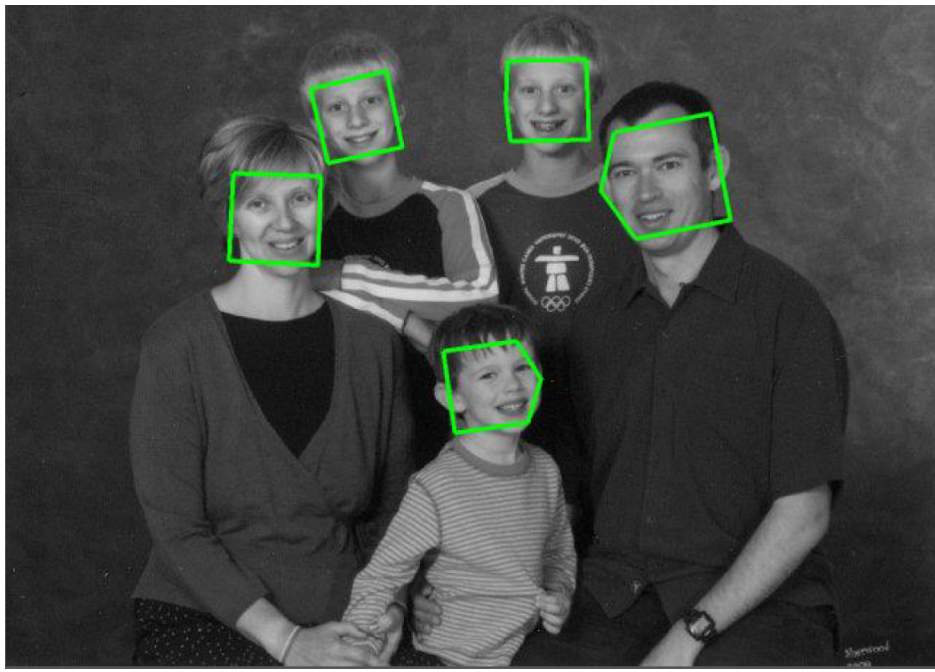
Representation of examples: bags of words



Bag of words representation: each document is represented by a boolean vector denoting presence or absence of a fixed set of words in it. $\mathbf{X}$: $m \times d$ array, $\mathbf{y}$: $m \times 1$ vector, where $m = 1000$, $d = 100$, and each $y \in \{1, 2, 3, 4\}$

Source: Kevin Murphy: Machine Learning: a probabilistic perspective, MIT Press

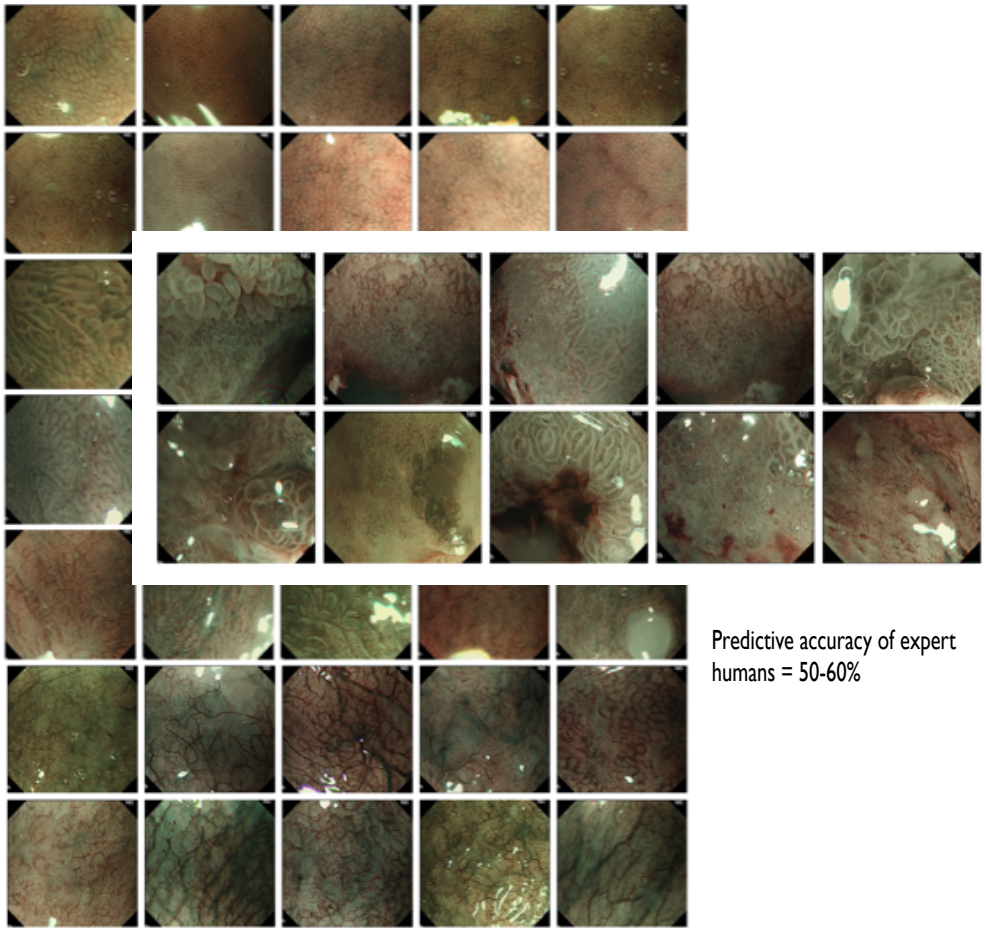
Representation of examples: raw pixels



Each image is represented as an vector of intensities of the pixels in it. Each image pixel is classified as belonging to a face or not.

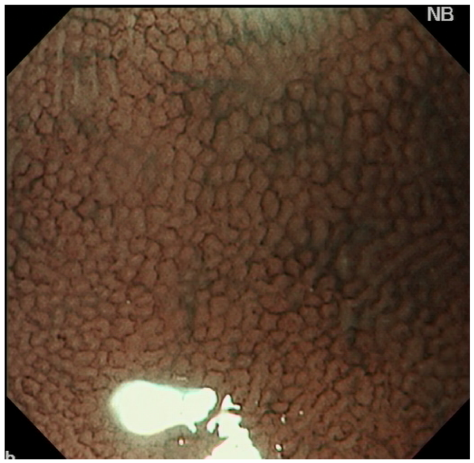
Source: Kevin Murphy: Machine Learning: a probabilistic perspective, MIT Press

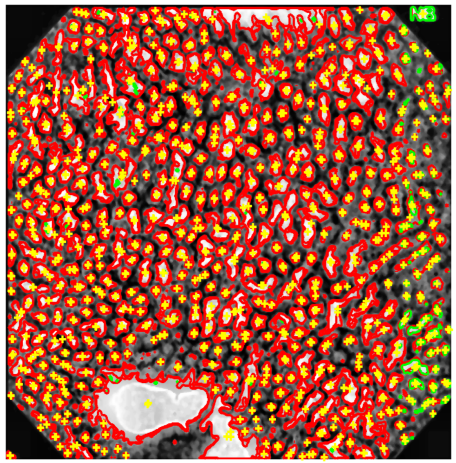
Representation of examples: raw pixels



Source: Subramanian et. al. under review

Representation of examples: hand-made features



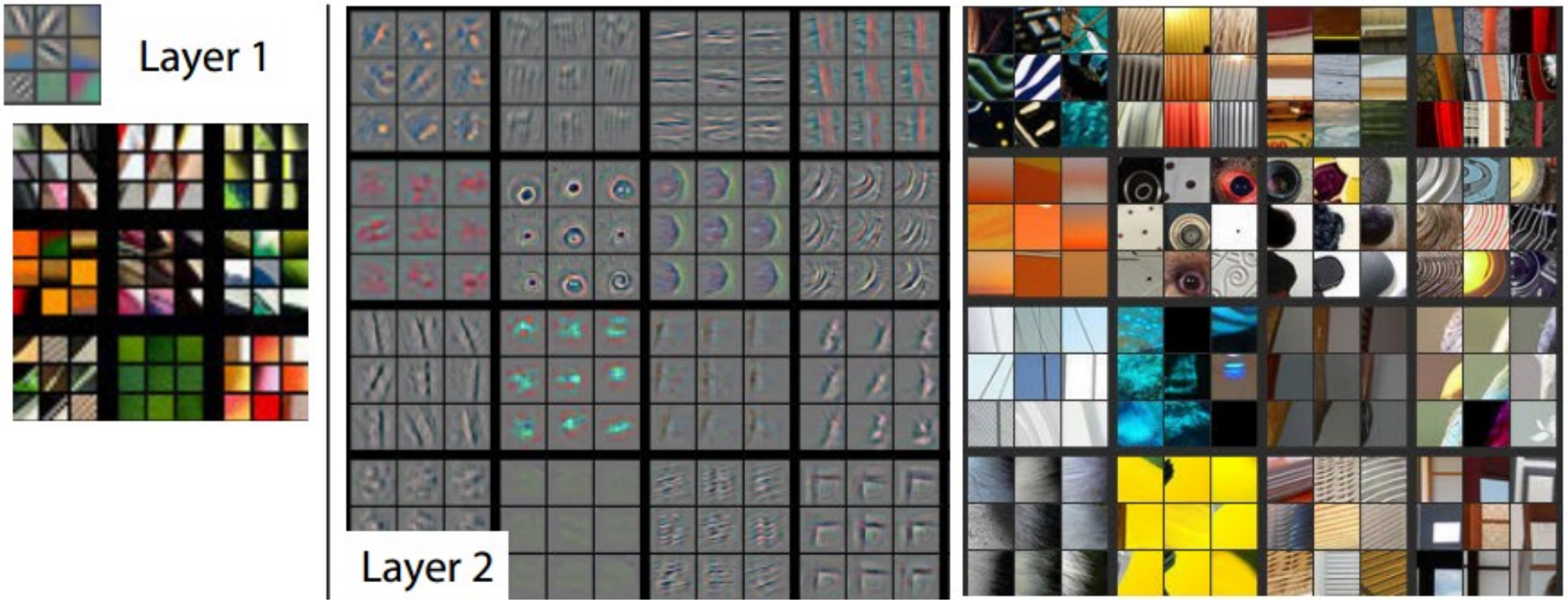


numpits_near_avg = 20.61
numpits = 1207
numobjects = 1272
avg_intensity = 0.7510
avg_energy = 99.935
sumareas = 249389
medarea = 115.5
mean_original_intensity = 67.355
mean_processed_intensity = 0.4819
mean_irregularity = 1.9679
mean_pit_hue = 0.0688

Each image is represented as an vector of extracted features. Each image is classified as being benign or cancerous.

Source: Subramanian et. al. under review

Representation of examples: learned features



The features learned at the first two levels on the Imagenet task (1000 classes). Zeiler and Fergus, 2013

Supervised machine learning

Given:

- A finite set of training data: pairs (x, y) where $x \in \mathcal{X}$ and $y \in \mathcal{Y}$,

$$\mathcal{D} = \{(x^{(i)}, y^{(i)}) | 1 \leq i \leq m, x^{(i)} \in \mathcal{X}, y^{(i)} \in \mathcal{Y}\}$$

Find:

- a function $h : \mathcal{X} \rightarrow \mathcal{Y}$ that predicts correctly on unseen examples.

When $\mathcal{Y}$ is a discrete set, we have a *classification problem*, and when $\mathcal{Y}$ is continuous, we have a *regression* problem.

Fundamental questions in supervised machine learning

- What data should be gathered to make predictions, and how should it be represented? (Feature extraction and selection)
- What class of models to build from observed data and prior knowledge? (Model selection)
- How can we be sure that we have a good predictive model? (Model validation)
- What algorithms should we use to learn these models from data? How can we scale them?

Given:

- A finite set of training data: pairs (x, y) where $x \in \mathcal{X}$ and $y \in \mathcal{Y}$,

$$\mathcal{D} = \{(x^{(i)}, y^{(i)}) | 1 \leq i \leq m,$$

$$x^{(i)} \in \mathcal{X}, y^{(i)} \in \mathcal{Y}\}$$

Find:

- a function $h : \mathcal{X} \rightarrow \mathcal{Y}$ that predicts correctly on unseen examples.

When to use supervised learning

- When we do not have analytical models of a complex system to compute outputs for specific inputs, but we have LOTS of input, output example pairs.
  - deciding whether a ligand will bind to a protein, whether someone will click on a link, detecting a face, riding a bicycle, driving a car.
- When requirements/environments change rapidly, so no fixed decision rule works for all time.
  - deciding to buy, sell or hold a stock based on prior stock prices, detecting spam in email.
- When there is tremendous individual variability and need for customization
  - detecting spam for a specific user, other document classification tasks, mortality prediction, hurricane damage prediction.

Class description

- Solving data analytics problems using supervised statistical machine learning methods
- Algorithms for regression and classification
  - Linear models: k-NN, linear regression, logistic regression, regularization
  - Non-linear models: Kernel and sparse kernel methods (SVMs), adaptive basis function models (decision trees)
  - Ensemble models: bagging and boosting (random forests and gradient boosting)
  - Deep learning: multilayer feedforward networks, convolutional networks, recurrent networks.
- Emphasis
  - model selection and model evaluation
  - hands-on experience in building predictive models in Python
  - developing an understanding of how and why a model works

Overview

- What is machine learning?
- Supervised machine learning
- Concepts in machine learning
  - Supervised learning is function optimization
  - Parametric and non-parametric models
  - A simple parametric model: linear regression
  - A simple non-parametric model: K nearest neighbors
  - The bias/variance tradeoff

Supervised machine learning

Given:

- A finite set of training data: pairs (x, y) where $x \in \mathcal{X}$ and $y \in \mathcal{Y}$,

$$\mathcal{D} = \{(x^{(i)}, y^{(i)}) | 1 \leq i \leq m, x^{(i)} \in \mathcal{X}, y^{(i)} \in \mathcal{Y}\}$$

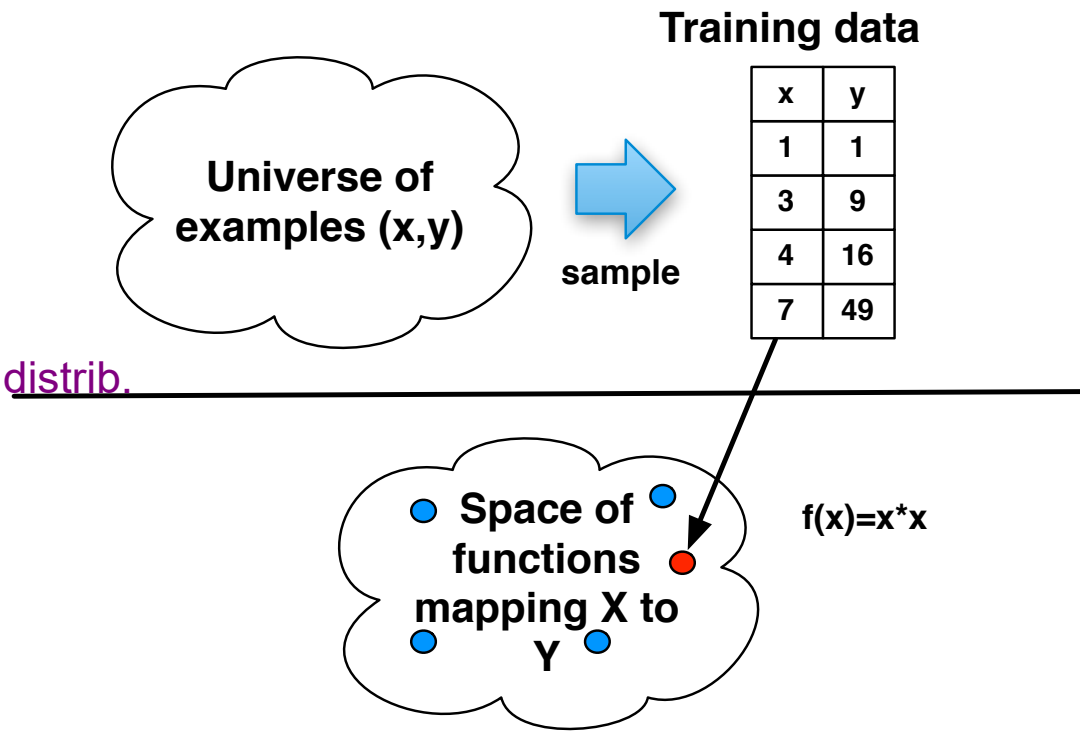
Find:

- a function $h : \mathcal{X} \rightarrow \mathcal{Y}$ that predicts correctly on unseen examples.

This specification of the supervised learning problem is incomplete. What is the space of functions h to consider? How can we tell if the learned function h will correctly predict on unseen examples?

Assumptions underlying supervised learning

- The training data set $\mathcal{D}$ is a representative sample of the space of all possible labeled examples. Each pair (x, y) in $\mathcal{D}$ is drawn independently from the same distribution P of examples (i.i.d. assumption). identical independent distrib.
- We also make smoothness assumptions about the family of functions h mapping input features to output.
- We need a measure of how close h is to the true function f.



Loss functions

We need a way to measure how close predictions made by a function h are to the true values.

$$ExpectedLoss(h) = E_{(x,y)\sim P}\ L(h(x), y)$$

But, we do not have access to the distribution over all examples. Therefore, we use empirical loss as an approximation to the expected loss, assuming that the training data is a representative sample. $(x^{(i)}, y^{(i)})$ denotes the i^{th} example in training data $\mathcal{D}$.

$$EmpiricalLoss(h) = \frac{1}{m}\sum_{i=1}^{m} L(h(x^{(i)}), y^{(i)})$$

Mathematical formulation of supervised learning

Given:

- A finite set of training data

$$\mathcal{D} = \{(x^{(i)}, y^{(i)}) | 1 \leq i \leq m, x^{(i)} \in \mathcal{X}, y^{(i)} \in \mathcal{Y}\}$$

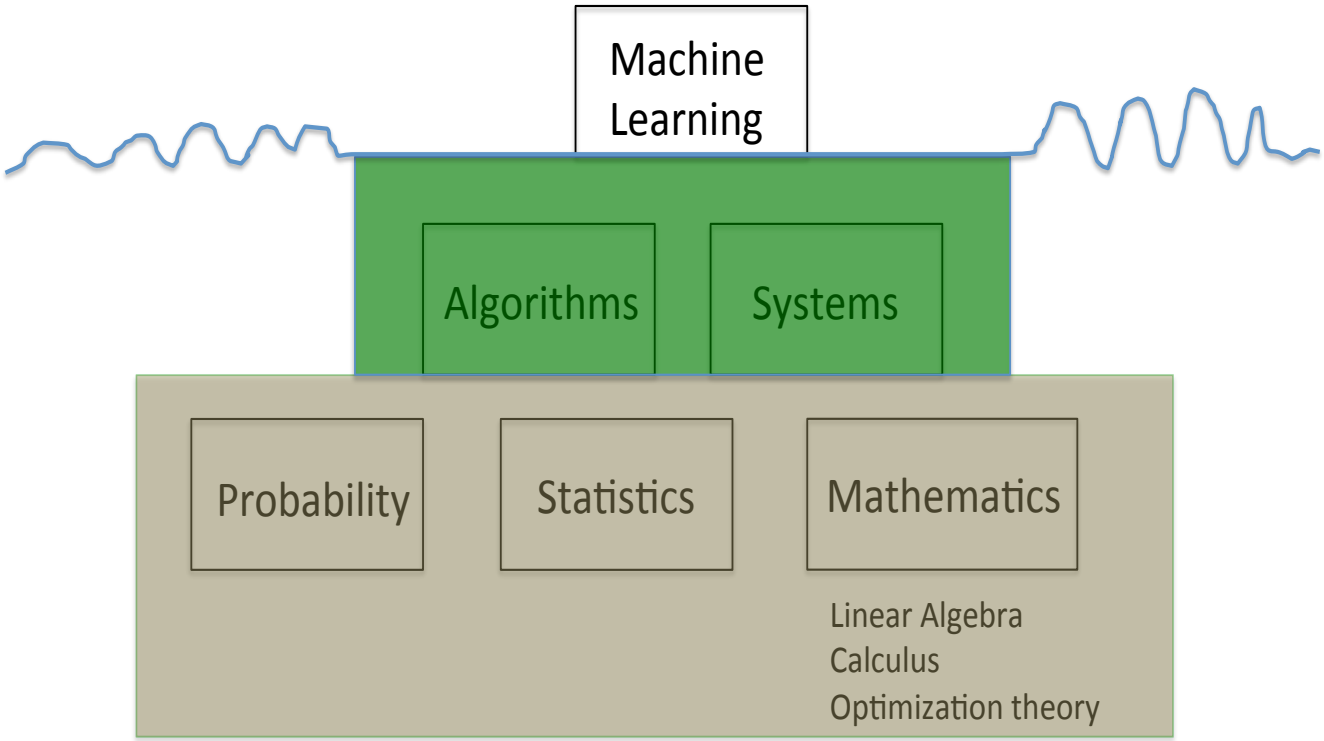
- A class of functions $\mathcal{H} : \mathcal{X} \rightarrow \mathcal{Y}$ model selection
- A loss function $L : \mathcal{Y} \times \mathcal{Y} \rightarrow \Re$ quality measure

Find:

- a function $h^* \in \mathcal{H}$ which minimizes *empirical loss*.

$$\begin{aligned} h^* &= argmin_{h \in \mathcal{H}} EmpiricalLoss(h) \\ &= argmin_{h \in \mathcal{H}} \left[\frac{1}{m} \sum_{i=1}^{m} L(h(x^{(i)}), y^{(i)}) \right] \end{aligned}$$

The foundations of machine learning



Parametric and non-parametric models h

- Parametric models
  - Have a fixed number of parameters.
  - Are generally computationally faster.
  - Make strong assumptions about the true function f.
  - Example: linear regression. $x \in \Re^d, \theta \in \Re^{d+1}$.

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \ldots + \theta_d x_d = \theta^T x$$

- Non-parametric models
  - Have a flexible number of parameters, and the number of parameters grows with the size of the training data.
  - Are generally computationally slower.
  - Make fewer assumptions about the true function f.
  - Example: K-nearest neighbors.

A simple parametric model: linear regression

Given:

- Training data $\mathcal{D} = \{(x^{(i)}, y^{(i)}), x \in \Re, y \in \Re\}$
- Function class: Linear function parameterized by vector θ of length $d+1$ (order of model is d)

$$h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \ldots + \theta_d x^d$$

- Loss function $L(h(x), y) = (y - h(x))^2$ – squared error

$$J(\theta) = \frac{1}{m} \sum_{i=1}^{m} (y^{(i)} - h_\theta(x^{(i)}))^2$$

To find the best model, we solve the following minimization problem:

$$\theta^* = argmin_\theta J(\theta) = argmin_\theta \frac{1}{m} \sum_{i=1}^{m} (y^{(i)} - h_\theta(x^{(i)}))^2$$

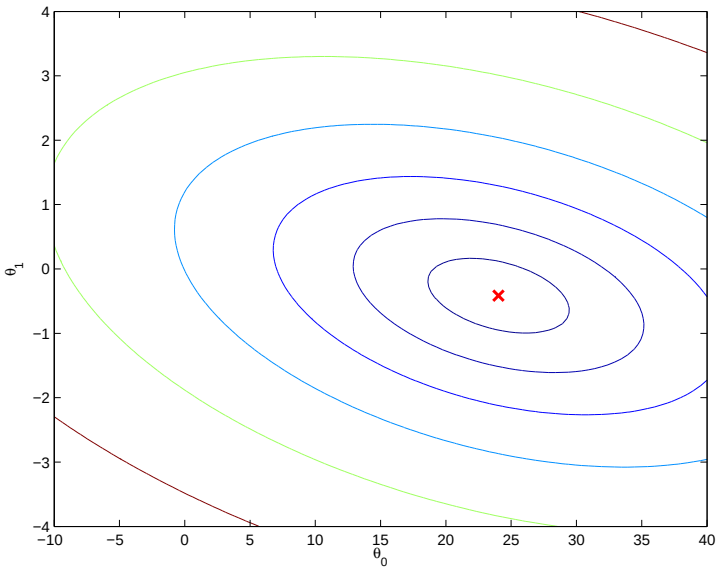
Estimating linear regression parameters

We have reformulated the problem of finding the best linear model of order d to be

$$argmin_\theta J(\theta)$$

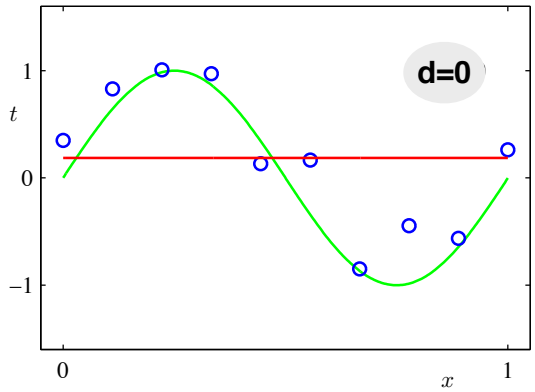
Since J is a convex function with a unique global minimum, this minimization problem can be solved in one of three ways.

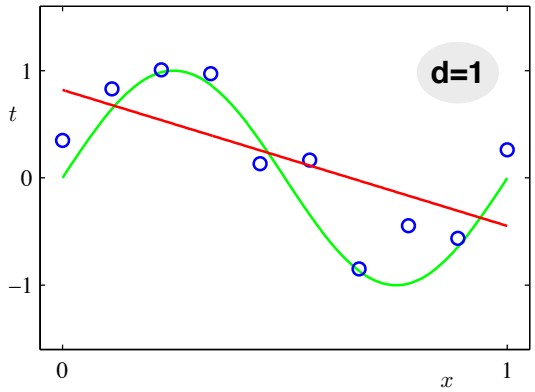
- Batch gradient descent
- Stochastic gradient descent
- Direct closed-form solution

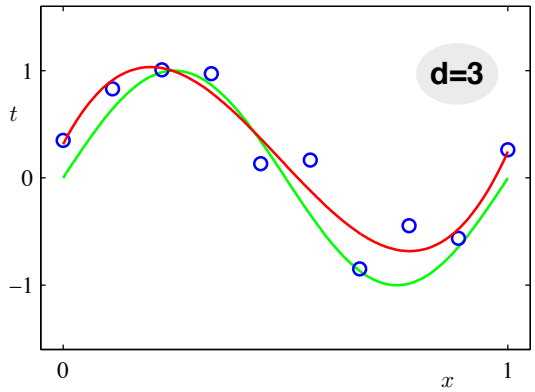


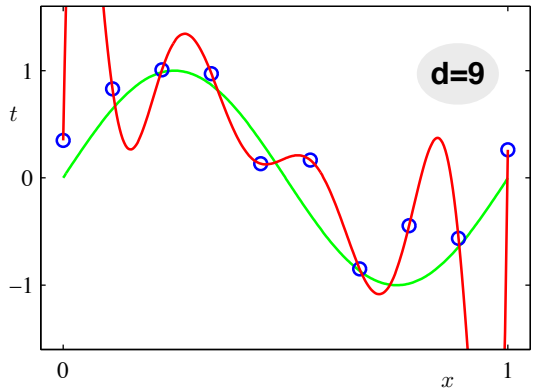
Contour plot of $J(\theta)$ for model of order 1: $h(x) = \theta_0 + \theta_1 x$

Linear regression models of various orders





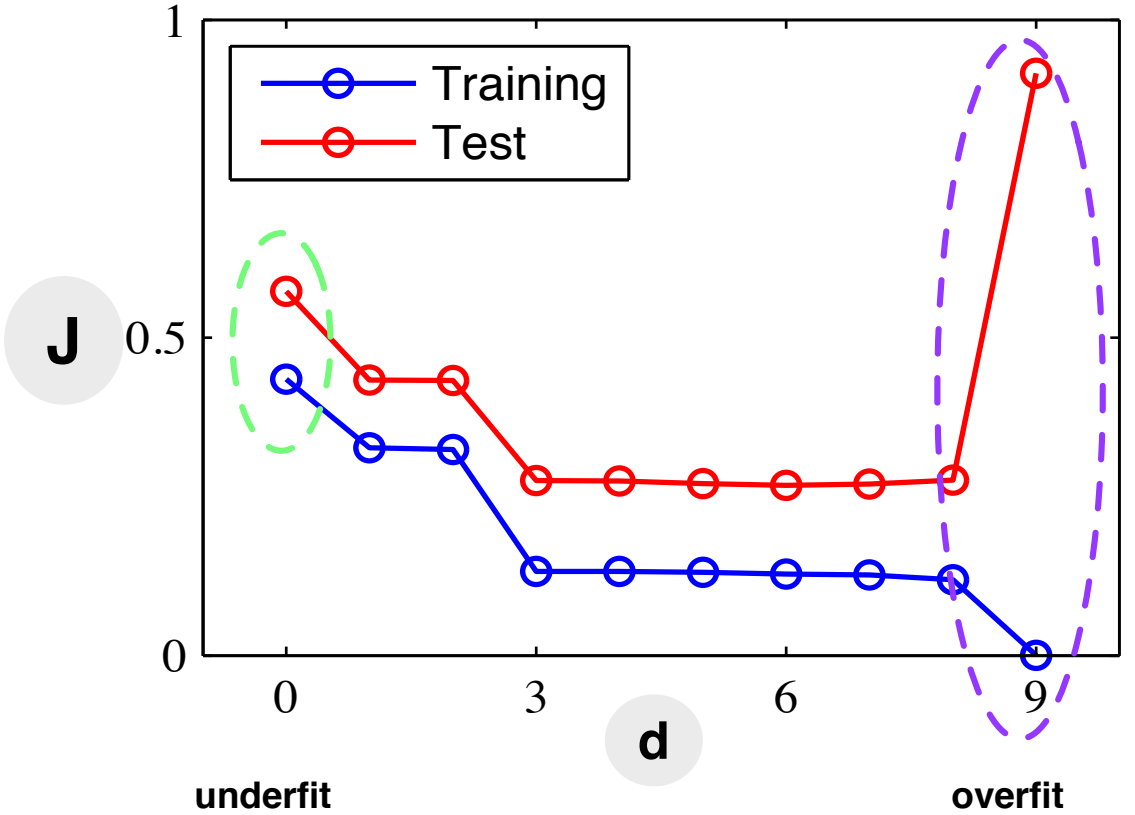




drive error to zero: overfit?

Green line: true function, blue dots: training data, red line: best linear model Source: The elements of statistical learning, Hastie et. al.

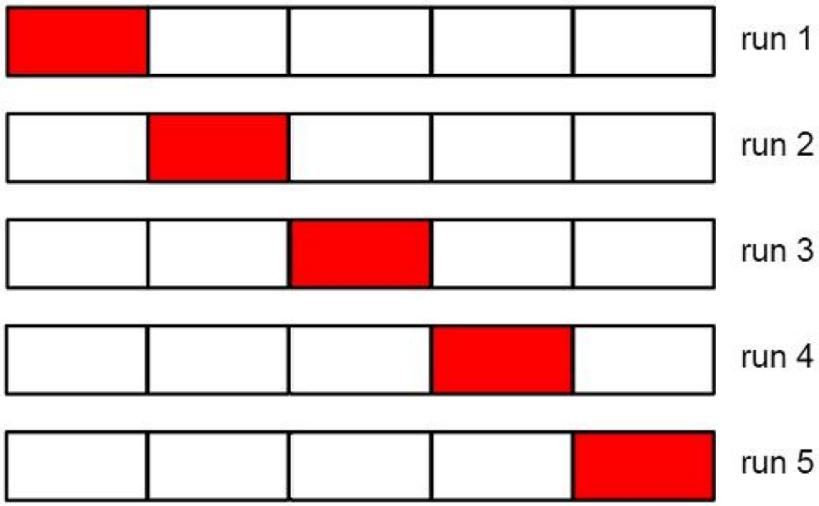
Selecting model order using training and test data



Occam's razor would choose d=3

Training data is used to estimate model parameter θ^*, set-aside test data is used to evaluate model performance. Source: The elements of statistical learning, Hastie et. al

Evaluating models using cross validation

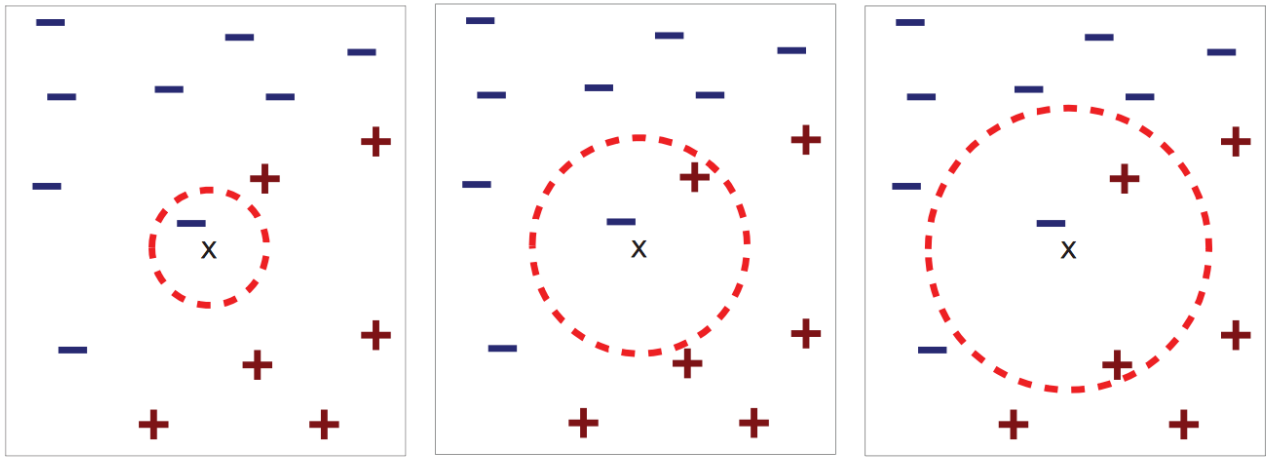


average results
calculate standard deviations;
look for large deviance between runs = outliers?
is issue in the test or training set(s)?

For each run/fold of cross validation, estimate model performance using the set aside (red) portion of the training data, and calculate optimal model parameters using the rest of the training data. Model performance, measured using the loss function, is the average over all the folds.

Source: The elements of statistical learning, Hastie et. al.

A simple non-parametric model: k-nearest neighbors

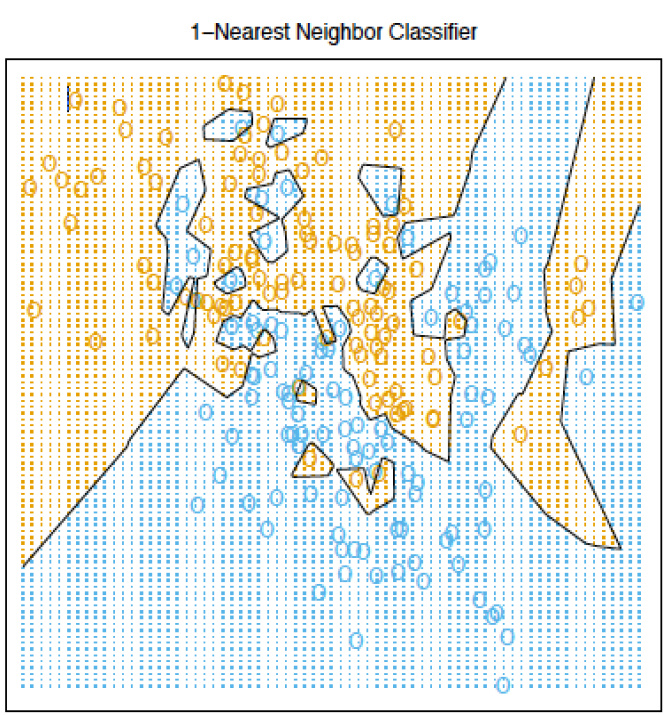


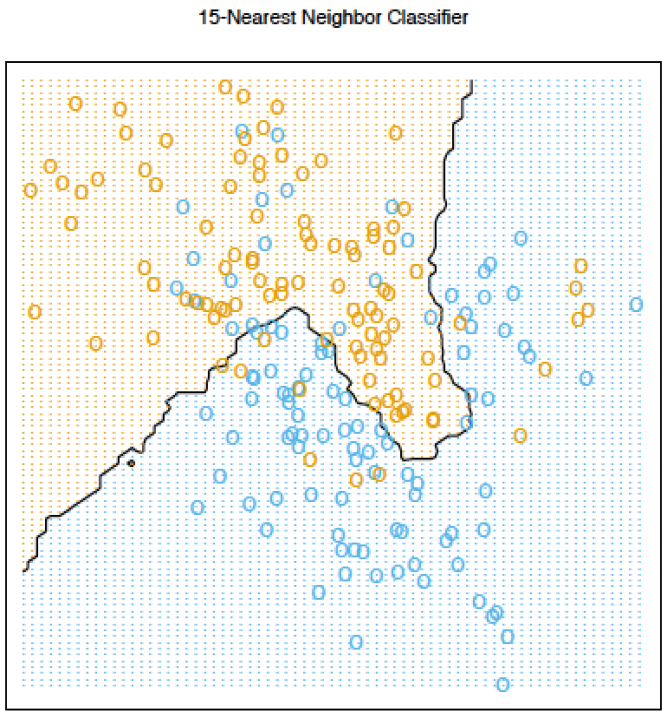
(a) 1-nearest neighbor (b) 2-nearest neighbor (c) 3-nearest neighbor

To classify a new point, find the majority vote among k nearest neighbors of the new point. Method is very sensitive to specification of distance measure and number of neighbors.

Source: MIT OCW

Model selection: k-nearest neighbors

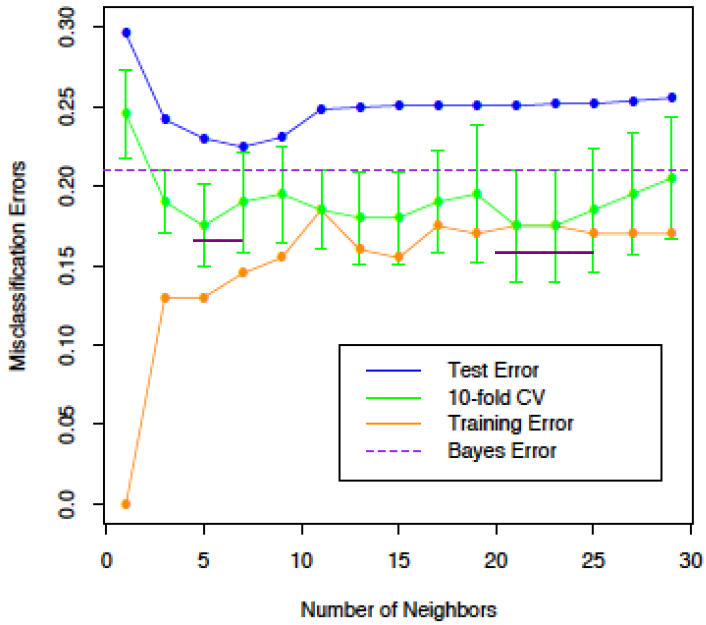




Larger values of k yield smoother predictions since we take majority vote over a bigger set of neighbors.

Source: The elements of statistical learning, Hastie et. al.

Model selection: k-nearest neighbors



for similar performing ks
choose smaller k

With cross-validation, pick the simplest model whose error is no more than one standard deviation above the best.

Source: The elements of statistical learning, Hastie et. al.

k-nearest neighbors

Obs.	X_1	X_2	X_3	Y
1	0	3	0	red
2	2	0	0	red
3	0	1	3	red
4	0	1	2	green
5	-1	0	1	green
6	1	1	1	red

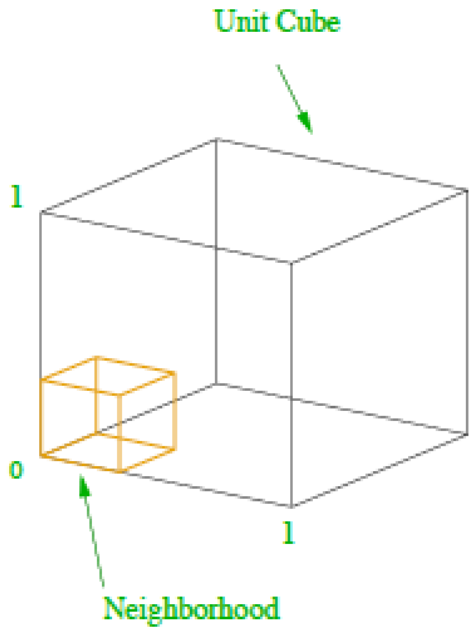
What is the prediction for Y when $X_1 = 0$, $X_2 = 0$, $X_3 = 0$ using 1-nearest neighbor?

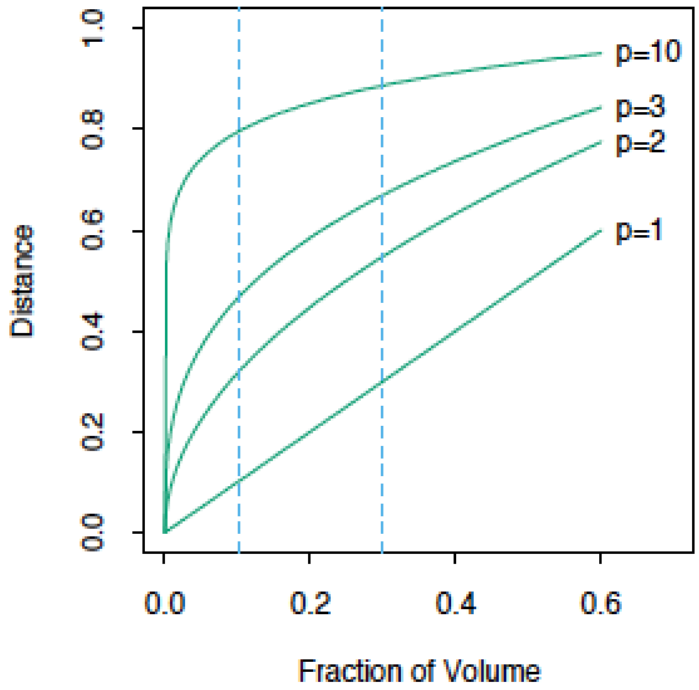
k-nearest neighbors: distance calculation

Obs.	X_1	X_2	X_3	Y	Distance from (0,0,0)
1	0	3	0	red	3
2	2	0	0	red	2
3	0	1	3	red	$\sqrt{10}$
4	0	1	2	green	$\sqrt{5}$
5	-1	0	1	green	$\sqrt{2}$
6	1	1	1	red	$\sqrt{3}$

- What is Y for $(0, 0, 0)$ when $k = 1$?
- What is Y for $(0, 0, 0)$ when $k = 3$?

The curse of dimensionality





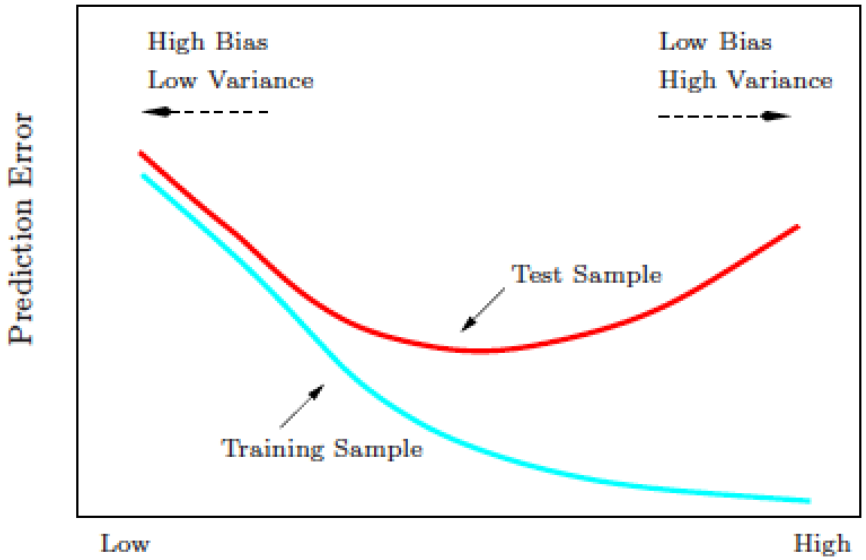
k-nearest neighbors best on low dimensional data!

suggest run unsupervised learning and perform dimension reduction. pers pref is up to k=5

Source: The elements of statistical learning, Hastie et. al.

The bias/variance tradeoff

underfitting



overfitting

for example, number of params

Variance refers to the variation in parameter estimates for a model. A highly biased model makes very strong assumptions about the nature of the true function.

Source: The elements of statistical learning, Hastie et. al.

Model errors

- Structural error (bias): error introduced by a limited function class h (infinite training data)

$$min_\theta E_{(x,y)\sim P}(y - \theta^T x)^2 = E_{(x,y)\sim P}(y - \theta^{*T} x)^2$$

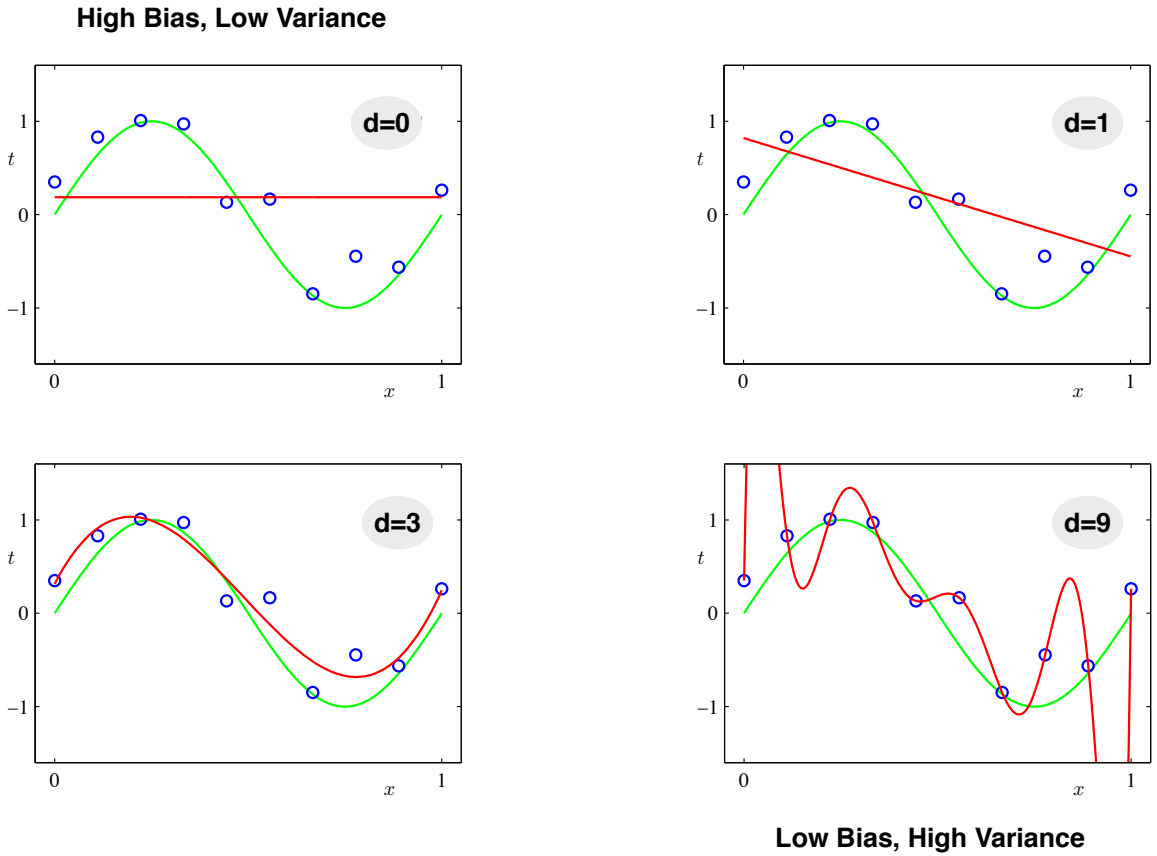
where θ^* is the parameter that minimizes expected squared loss error.

- Approximation error (variance): measures how close we get to the optimal linear prediction with finite training data.

$$E_{(x,y)\sim P} \quad (\theta^{*T} x - \hat{\theta}^{*T} x)^2$$

where $\hat{\theta}$ is the estimate computed from a finite training sample.

The bias/variance tradeoff



We will use regularization to automatically tradeoff bias and variance.

Summary (1/3)

- All models are wrong, but some models are useful – George Box.
- There is no single best model that works optimally for all kinds of problems (Wolpert 1996; the no-free-lunch theorem)
- Lots of opportunity for innovation in data representation, model specification, and algorithms for learning models from data.

No single best model, bring human judgement along the way

Summary (2/3)

- We are at an inflection point with respect to supervised machine learning: immense amount of data, compute power and new algorithms/architectures and hardware support for fast training.
- Machine learning is already at the heart of speech recognition, handwriting recognition, financial applications, home automation (Nest), e-commerce, computer vision.
- Machine learning methods have transformed search and natural language processing (understanding, translation, retrieval).
- Machine learning with big data is revolutionizing biology and other natural sciences, as well as all branches of engineering.

Summary (3/3)

- Data is a new source of power for science.
- Every scientist should learn the fundamentals of machine learning and statistical thinking.
- By combining mathematical frameworks with models learned from data, we can accelerate discoveries in science and engineer high-performance decision-making systems of the future.

Pitfalls of machine learning (1/3)

Hidden variables: A machine learning algorithm can pick up on unintentional variations in the data. Flushing out confounders is critical. Still an art.

- The DARPA enemy tank detection story — blue sky on friendly tanks
- Predicting side effects of drug combinations story — no negative data, generated on random unrealistic combinations

Pitfalls of machine learning (2/3)

Loss function specification/mis-specification of the learning objective

- The LVAD story
  orig: predict optimal insertion point/timing
  revised: predict 1 year mortality
- The Google diabetic retinopathy story

doubtfully labeled data, no consensus among experts

Pitfalls of machine learning (3/3)

Splitting data inappropiately: random split into training, validation and test set.

- the Decagon story
- the Lorenz96 story

sometimes random split is not the best, for example collection over time may overlook artifacts; choose different split gives different results

Feasible ML projects

- Problem involves learning a "simple concept" (humans can do it quickly; decision criteria not too diverse)
  - Classifying a piece of email as spam/ham vs crafting a meaningful response to a piece of email.
  - Identifying other cars in an image for a self-driving car vs identifying construction worker on road in an image.
- Lots of data available to learn a robust input-output mapping
  - learning decision rules for diabetic retinopathy from 100,000 images vs 100 images (humans)