Probabilistic machine learning

Zoubin Ghahramani

Department of Engineering
University of Cambridge, UK

zoubin@eng.cam.ac.uk
http://learning.eng.cam.ac.uk/zoubin/

Nijmegen
2015

What is Machine Learning?

Many related terms:

- Pattern Recognition
- Neural Networks and Deep Learning
- Data Mining
- Adaptive Control
- Statistical Modelling
- Data analytics / data science
- Artificial Intelligence
- Machine Learning

Learning:
The view from different fields

- Engineering: signal processing, system identification, adaptive and optimal control, information theory, robotics,...
- Computer Science: Artificial Intelligence, computer vision, information retrieval, natural language processing, data mining,...
- Statistics: estimation, learning theory, data science, inference from data,...
- Cognitive Science and Psychology: perception, movement control, reinforcement learning, mathematical psychology, computational linguistics,...
- Computational Neuroscience: neuronal networks, neural information processing, ...
- Economics: decision theory, game theory, operational research, e-commerce, choice modelling,...

Different fields, Convergent ideas

- The same set of ideas and mathematical tools have emerged in many of these fields, albeit with different emphases.

- *Machine learning is an interdisciplinary field focusing on both the mathematical foundations and practical applications of systems that learn, reason and act.*

Machine Learning has *many* applications

Bioinformatics

Computer Vision

Robotics

Scientific Data Analysis

Natural Language Processing

Machine Learning

Information Retrieval

Speech Recognition

Recommender Systems

Signal Processing

Machine Translation

Medical Informatics

Targeted Advertising

Finance

Data Compression

Modeling vs toolbox views of Machine Learning

- **Machine Learning is a toolbox of methods for processing data**: feed the data into one of many possible methods; choose methods that have good theoretical or empirical performance; make predictions and decisions

- **Machine Learning is the science of learning models from data**: define a space of possible models; learn the parameters and structure of the models from data; make predictions and decisions

Probabilistic Modelling

- A model describes data that one could observe from a system
- If we use the mathematics of probability theory to express all forms of uncertainty and noise associated with our model...
- ...then *inverse probability* (i.e. Bayes rule) allows us to infer unknown quantities, adapt our models, make predictions and learn from data.

Bayes Rule

$$P(\text{hypothesis}|\text{data}) = \frac{P(\text{data}|\text{hypothesis})P(\text{hypothesis})}{P(\text{data})}$$

Rev'd Thomas Bayes (1702–1761)

- Bayes rule tells us how to do inference about hypotheses from data.
- Learning and prediction can be seen as forms of inference.

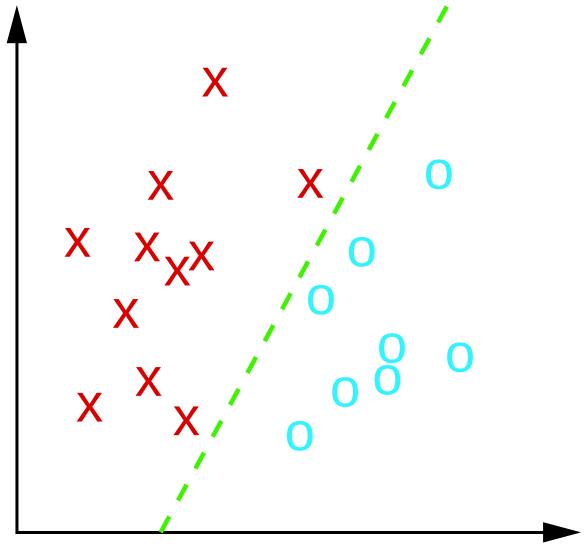
Some Canonical Machine Learning Problems

- Linear Classification
- Polynomial Regression
- Clustering with Gaussian Mixtures (Density Estimation)

Linear Classification

Data: $\mathcal{D} = \{(\mathbf{x}^{(n)}, y^{(n)})\}$ for $n = 1, \ldots, N$ data points

$$\begin{aligned} \mathbf{x}^{(n)} &\in \mathbb{R}^D \\ y^{(n)} &\in \{+1, -1\} \end{aligned}$$



Model:

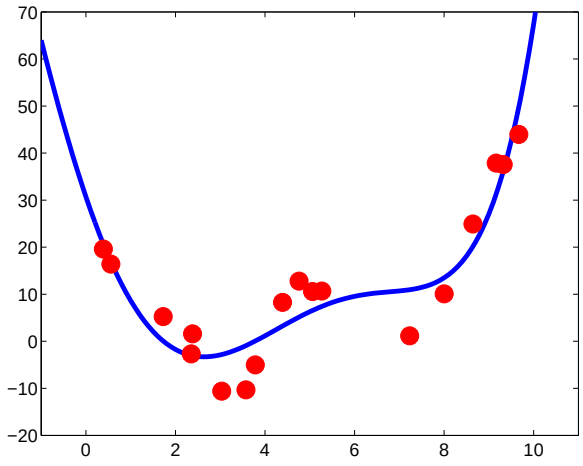
$$P(y^{(n)} = +1 | \boldsymbol{\theta}, \mathbf{x}^{(n)}) = \begin{cases} 1 & \text{if } \sum_{d=1}^{D} \theta_d \, x_d^{(n)} + \theta_0 \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

Parameters: $\boldsymbol{\theta} \in \mathbb{R}^{D+1}$

Goal: To infer $\boldsymbol{\theta}$ from the data and to predict future labels $P(y|\mathcal{D}, \mathbf{x})$

Polynomial Regression

Data: $\mathcal{D} = \{(x^{(n)}, y^{(n)})\}$ for $n = 1, \ldots, N$



$$x^{(n)} \in \mathbb{R}$$
$$y^{(n)} \in \mathbb{R}$$

Model:

$$y^{(n)} = a_0 + a_1 x^{(n)} + a_2 {x^{(n)}}^2 \ldots + a_m {x^{(n)}}^m + \epsilon$$

where

$$\epsilon \sim \mathcal{N}(0, \sigma^2)$$

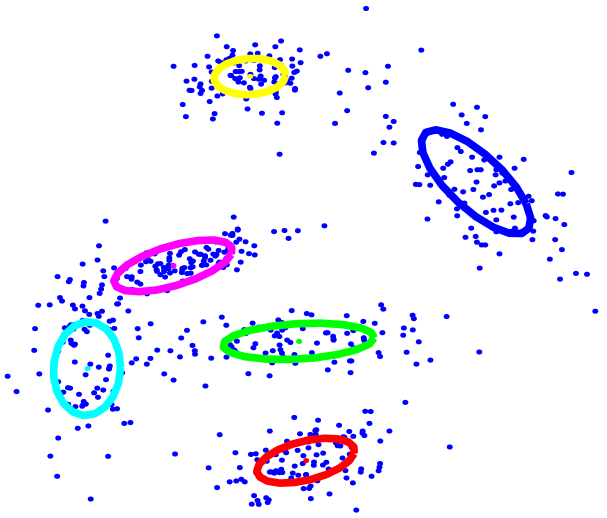
Parameters: $\boldsymbol{\theta} = (a_0, \ldots, a_m, \sigma)$

Goal: To infer $\boldsymbol{\theta}$ from the data and to predict future outputs $P(y|\mathcal{D}, x, m)$

Clustering with Gaussian Mixtures (Density Estimation)

Data: $\mathcal{D} = \{\mathbf{x}^{(n)}\}$ for $n = 1, \ldots, N$

$$\mathbf{x}^{(n)} \in \mathbb{R}^D$$



Model:

$$\mathbf{x}^{(n)} \sim \sum_{i=1}^{m} \pi_i \, p_i(\mathbf{x}^{(n)})$$

where

$$p_i(\mathbf{x}^{(n)}) = \mathcal{N}(\mu^{(i)}, \Sigma^{(i)})$$

Parameters: $\boldsymbol{\theta} = \big((\mu^{(1)}, \Sigma^{(1)}) \ldots, (\mu^{(m)}, \Sigma^{(m)}), \boldsymbol{\pi}\big)$

Goal: To infer $\boldsymbol{\theta}$ from the data, predict the density $p(\mathbf{x}|\mathcal{D}, m)$, and infer which points belong to the same cluster.

Probabilisic Machine Learning

Everything follows from two simple rules:

Sum rule: $P(x) = \sum_y P(x, y)$

Product rule: $P(x, y) = P(x)P(y|x)$

$$P(\theta|\mathcal{D}, m) = \frac{P(\mathcal{D}|\theta, m)P(\theta|m)}{P(\mathcal{D}|m)}$$

$P(\mathcal{D}\|\theta, m)$	likelihood of parameters θ in model m
$P(\theta\|m)$	prior probability of θ
$P(\theta\|\mathcal{D}, m)$	posterior of θ given data $\mathcal{D}$

Prediction:

$$P(x|\mathcal{D}, m) = \int P(x|\theta, \mathcal{D}, m)P(\theta|\mathcal{D}, m)d\theta$$

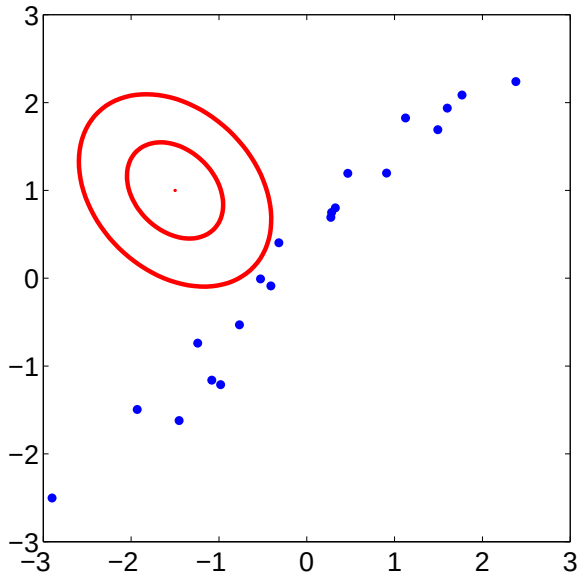
Model Comparison:

$$P(m|\mathcal{D}) = \frac{P(\mathcal{D}|m)P(m)}{P(\mathcal{D})}$$

$$P(\mathcal{D}|m) = \int P(\mathcal{D}|\theta, m)P(\theta|m)\, d\theta$$

A Simple Example: Learning a Gaussian

$$P(\theta|\mathcal{D}, m) = \frac{P(\mathcal{D}|\theta, m)P(\theta|m)}{P(\mathcal{D}|m)}$$



- The model m is a multivariate Gaussian.
- Data, $\mathcal{D}$ are the blue dots.
- Parameters θ are the mean vector and covariance matrix of the Gaussian.

That's it!

Questions

- What motivates the Bayesian framework?
- Where does the prior come from?
- How do we do these integrals?

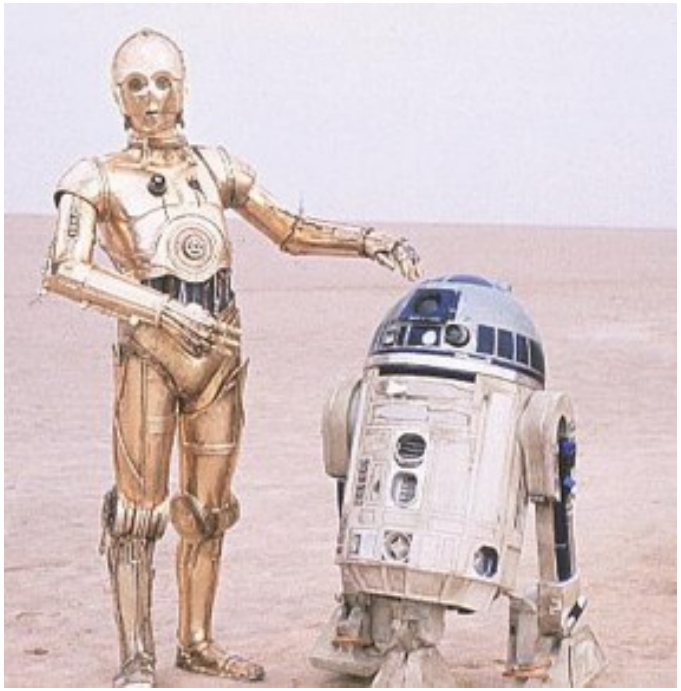
Representing Beliefs (Artificial Intelligence)

Consider a robot. In order to behave intelligently the robot should be able to represent beliefs about propositions in the world:

"my charging station is at location (x,y,z)"

"my rangefinder is malfunctioning"

"that stormtrooper is hostile"



We want to represent the **strength** of these beliefs numerically in the brain of the robot, and we want to know what mathematical rules we should use to manipulate those beliefs.

Representing Beliefs II

Let's use $b(x)$ to represent the strength of belief in (plausibility of) proposition x.

$0 \leq b(x) \leq 1$		
$b(x) = 0$	x	is definitely **not true**
$b(x) = 1$	x	is definitely **true**
$b(x\|y)$		strength of belief that x is true given that we know y is true

Cox Axioms (Desiderata):

- Strengths of belief (degrees of plausibility) are represented by real numbers
- Qualitative correspondence with common sense
- Consistency
  - If a conclusion can be reasoned in several ways, then each way should lead to the same answer.
  - The robot must always take into account all relevant evidence.
  - Equivalent states of knowledge are represented by equivalent plausibility assignments.

Consequence: Belief functions (e.g. $b(x)$, $b(x|y)$, $b(x, y)$) must satisfy the rules of probability theory, including sum rule, product rule and therefore Bayes rule.

(Cox 1946; Jaynes, 1996; van Horn, 2003)

The Dutch Book Theorem

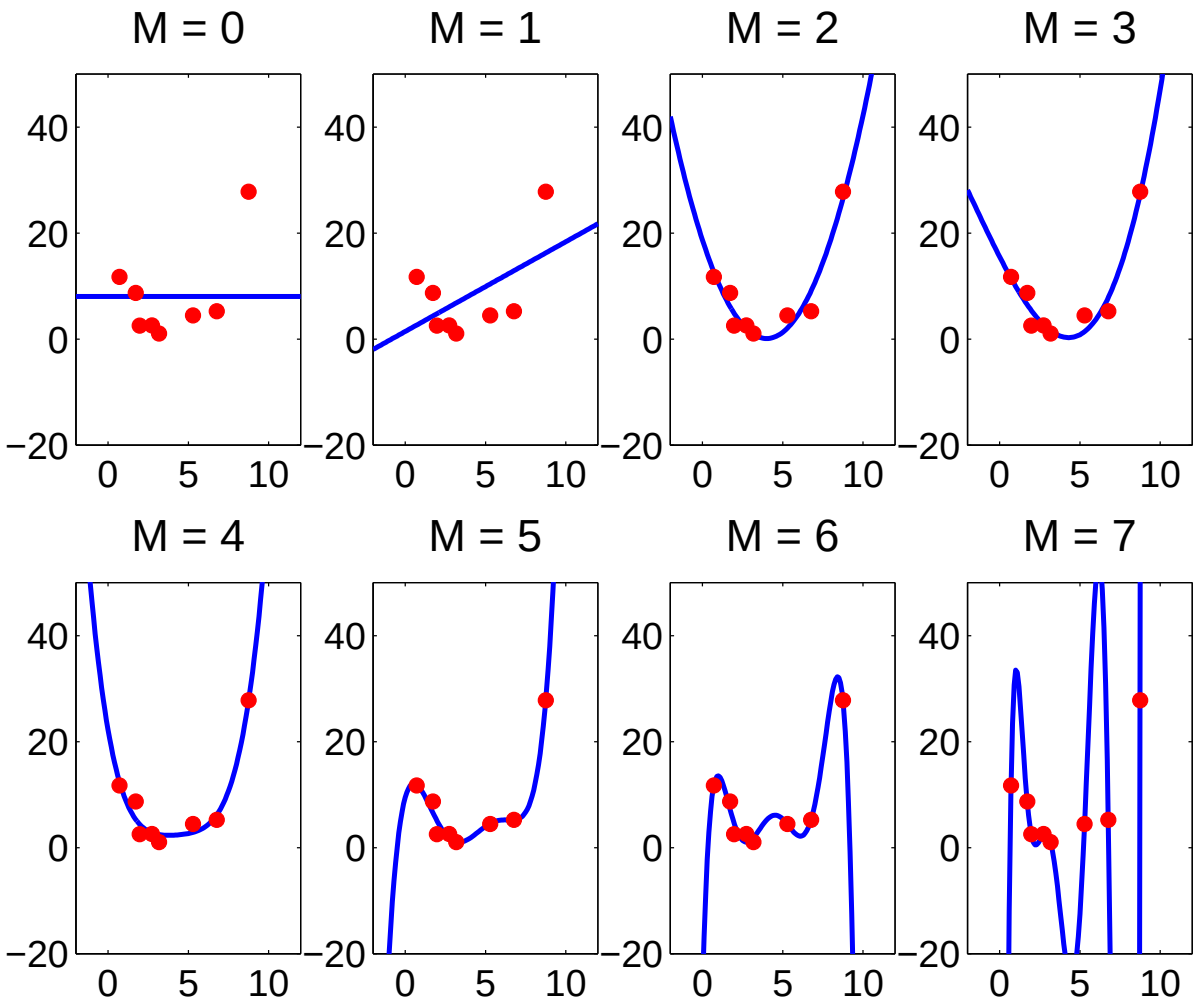
Assume you are willing to accept bets with odds proportional to the strength of your beliefs. That is, $b(x) = 0.9$ implies that you will accept a bet:

$$\begin{cases} x & \text{is true} & \text{win} & \geq \$1 \\ x & \text{is false} & \text{lose} & \$9 \end{cases}$$

Then, unless your beliefs satisfy the rules of probability theory, including Bayes rule, there exists a set of simultaneous bets (called a "Dutch Book") which you are willing to accept, and for which **you are guaranteed to lose money, no matter what the outcome**.

The only way to guard against Dutch Books to to ensure that your beliefs are coherent: i.e. satisfy the rules of probability.

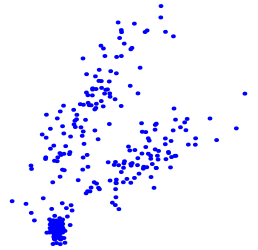
Model Selection



Learning Model Structure

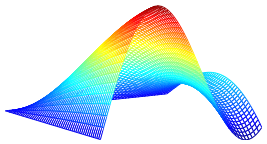
How many clusters in the data?

k-means, mixture models



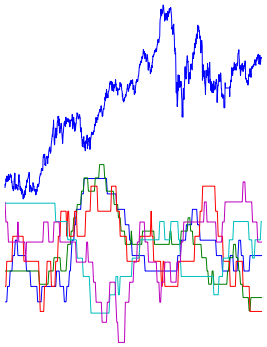
What is the intrinsic dimensionality of the data?

PCA, LLE, Isomap, GPLVM



Is this input relevant to predicting that output?

feature / variable selection



What is the order of a dynamical system?

state-space models, ARMA, GARCH

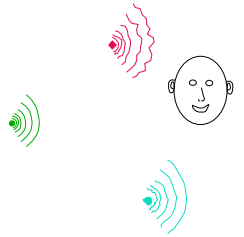
How many states in a hidden Markov model?

HMM

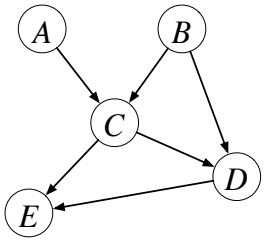
SVYDAAAQLTADVKKDLRDSWKVIGSDKKGNGVA

How many independent sources in the input?

ICA



What is the structure of a graphical model?



Bayesian Occam's Razor and Model Selection

Compare model classes, e.g. m and m', using posterior probabilities given $\mathcal{D}$:

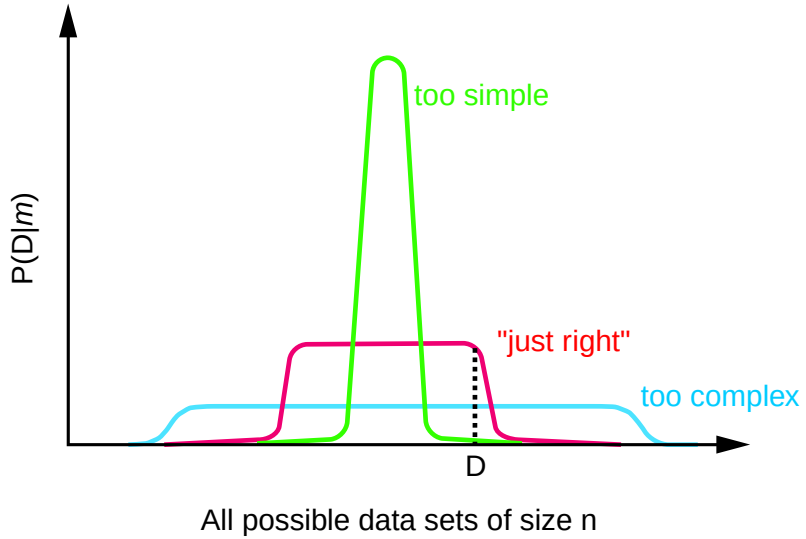
$$p(m|\mathcal{D}) = \frac{p(\mathcal{D}|m)\,p(m)}{p(\mathcal{D})}, \qquad p(\mathcal{D}|m) = \int p(\mathcal{D}|\boldsymbol{\theta}, m)\; p(\boldsymbol{\theta}|m)\; d\boldsymbol{\theta}$$

Interpretations of the Marginal Likelihood ("model evidence"):

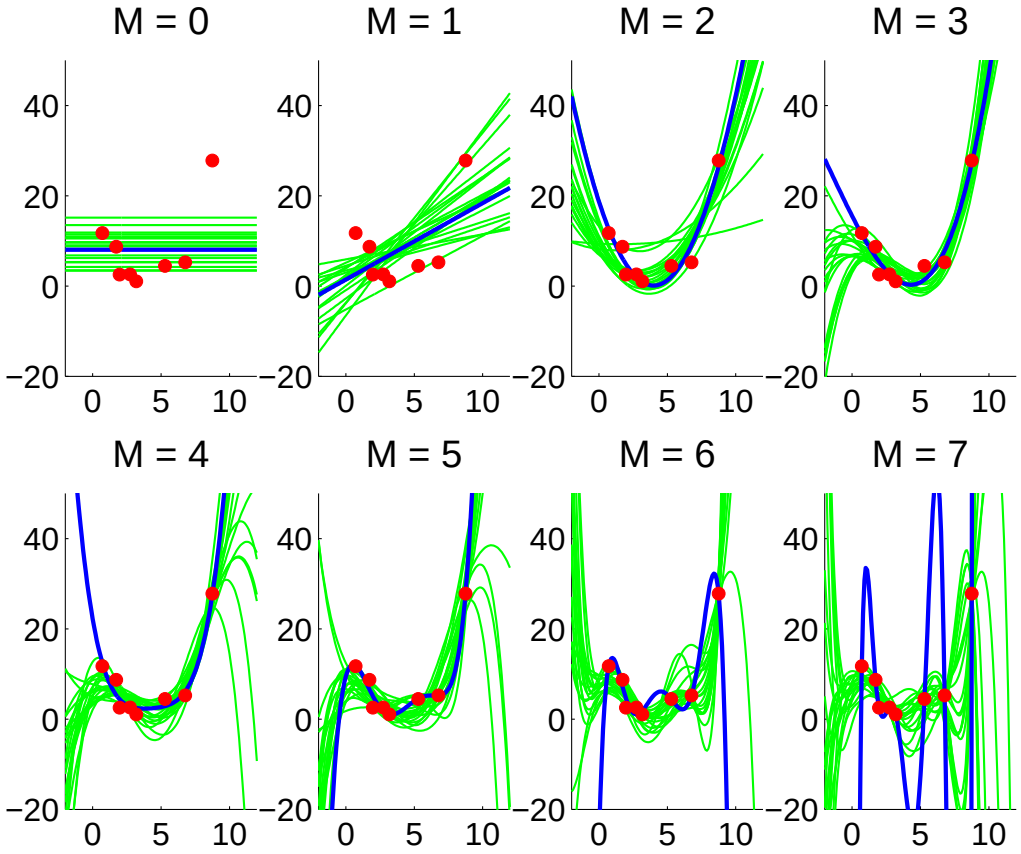
- The probability that *randomly selected* parameters from the prior would generate $\mathcal{D}$.
- Probability of the data under the model, *averaging* over all possible parameter values.
- $\log_2\left(\frac{1}{p(\mathcal{D}|m)}\right)$ is the number of *bits of surprise* at observing data $\mathcal{D}$ under model m.

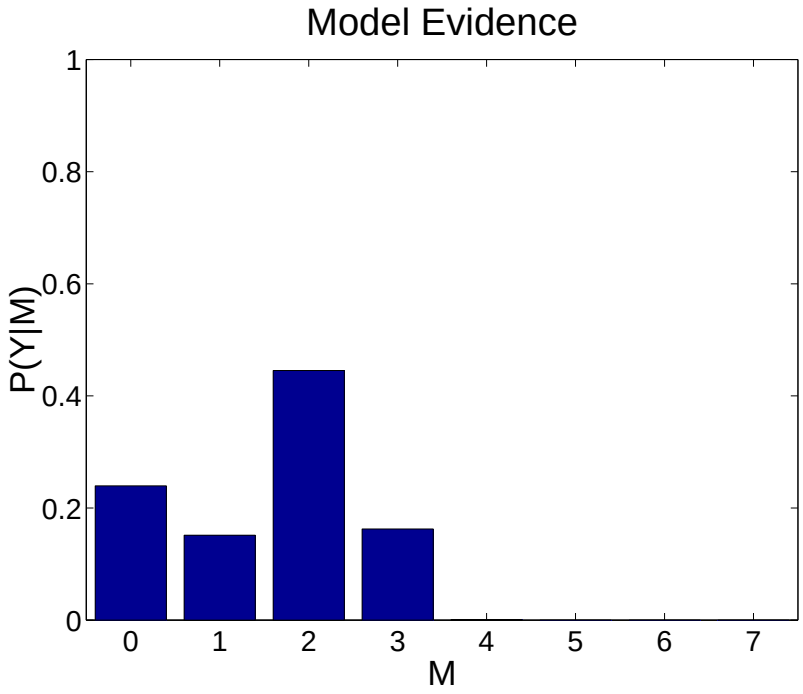
Model classes that are too simple are unlikely to generate the data set.

Model classes that are too complex can generate many possible data sets, so again, they are unlikely to generate that particular data set at random.



Bayesian Model Selection: Occam's Razor at Work





For example, for quadratic polynomials ($m = 2$): $y = a_0 + a_1 x + a_2 x^2 + \epsilon$, where $\epsilon \sim \mathcal{N}(0, \sigma^2)$ and parameters $\boldsymbol{\theta} = (a_0\ a_1\ a_2\ \sigma)$

`demo: polybayes`

On Choosing Priors

- **Objective Priors**: noninformative priors that attempt to capture ignorance and have good frequentist properties.

- **Hierarchical Priors**: multiple levels of priors:

$$\begin{aligned} p(\theta) &= \int d\boldsymbol{\alpha}\, p(\boldsymbol{\theta}|\boldsymbol{\alpha})p(\boldsymbol{\alpha}) \\ &= \int d\boldsymbol{\alpha}\, p(\boldsymbol{\theta}|\boldsymbol{\alpha}) \int d\beta\, p(\boldsymbol{\alpha}|\beta)p(\beta) \end{aligned}$$

- **Empirical Priors**: learn some of the parameters of the prior from the data ("Empirical Bayes")

- **Subjective Priors**: priors should capture our beliefs about reasonable hypotheses before observing the data as well as possible. They are subjective but not arbitrary.

Subjective Priors

Priors should capture **our beliefs and knowledge** about the range of reasonable hypotheses as well as possible.

Otherwise we (or our learning machine) will make inferences and decisions which are **not coherent** with our (its) beliefs and knowledge.

How do we know our beliefs?

- Think about the problems domain.
- Generate data from the prior. Does it match expectations?

Even very vague prior beliefs can be useful, since the data will concentrate the posterior around reasonable models.

The key ingredient of Bayesian methods is not the prior, it's the idea of averaging over different possibilities.

Bayesian Modelling

Everything follows from two simple rules:

Sum rule: $P(x) = \sum_y P(x, y)$

Product rule: $P(x, y) = P(x)P(y|x)$

$$P(\theta|\mathcal{D}, m) = \frac{P(\mathcal{D}|\theta, m)P(\theta|m)}{P(\mathcal{D}|m)}$$

$P(\mathcal{D}\|\theta, m)$	likelihood of parameters θ in model m
$P(\theta\|m)$	prior probability of θ
$P(\theta\|\mathcal{D}, m)$	posterior of θ given data $\mathcal{D}$

Prediction:

$$P(x|\mathcal{D}, m) = \int P(x|\theta, \mathcal{D}, m)P(\theta|\mathcal{D}, m)d\theta$$

Model Comparison:

$$P(m|\mathcal{D}) = \frac{P(\mathcal{D}|m)P(m)}{P(\mathcal{D})}$$

$$P(\mathcal{D}|m) = \int P(\mathcal{D}|\theta, m)P(\theta|m)\, d\theta$$

Computing Marginal Likelihoods can be Computationally Intractable

Observed data $\mathbf{y}$, hidden variables $\mathbf{x}$, parameters $\boldsymbol{\theta}$, model class m.

$$p(\mathbf{y}|m) = \int p(\mathbf{y}|\boldsymbol{\theta}, m)\, p(\boldsymbol{\theta}|m)\, d\boldsymbol{\theta}$$

- This can be a very **high dimensional integral**.
- The presence of hidden **latent variables** results in additional dimensions that need to be marginalized out.

$$p(\mathbf{y}|m) = \int \int p(\mathbf{y}, \mathbf{x}|\boldsymbol{\theta}, m)\, p(\boldsymbol{\theta}|m)\, d\mathbf{x}\, d\boldsymbol{\theta}$$

- The likelihood term can be **complicated**.

Approximation Methods for Posteriors and Marginal Likelihoods

- Laplace approximation
- Bayesian Information Criterion (BIC)
- Variational approximations
- Expectation Propagation (EP)
- Markov chain Monte Carlo methods (MCMC)
- Exact Sampling
- ...

Note: there are many other deterministic approximations; we won't review them all.

Parametric vs Nonparametric Models

- *Parametric models* assume some finite set of parameters θ. Given the parameters, future predictions, x, are independent of the observed data, $\mathcal{D}$:

$$P(x|\theta, \mathcal{D}) = P(x|\theta)$$

  therefore θ capture everything there is to know about the data.

- So the complexity of the model is bounded even if the amount of data is unbounded. This makes them not very flexible.

- *Non-parametric models* assume that the data distribution cannot be defined in terms of such a finite set of parameters. But they can often be defined by assuming an *infinite dimensional* θ. Usually we think of θ as a *function*.

- The amount of information that θ can capture about the data $\mathcal{D}$ can grow as the amount of data grows. This makes them more flexible.

Bayesian nonparametrics

A simple framework for modelling complex data.

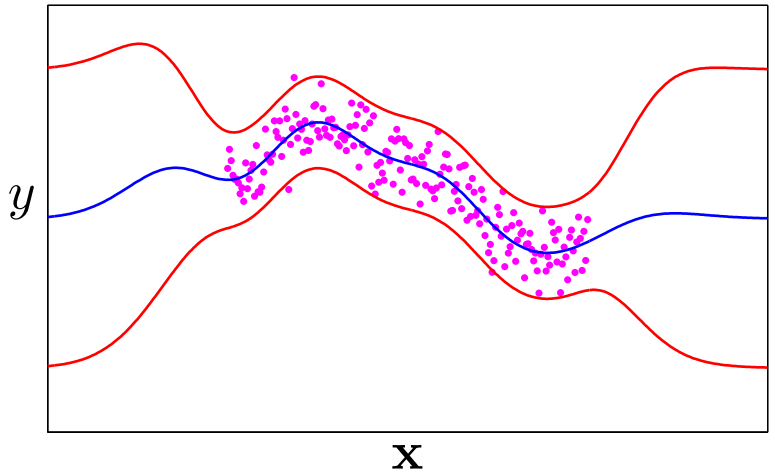
Nonparametric models can be viewed as having infinitely many parameters

Examples of non-parametric models:

Parametric	Non-parametric	Application
polynomial regression	Gaussian processes	function approx.
logistic regression	Gaussian process classifiers	classification
mixture models, k-means	Dirichlet process mixtures	clustering
hidden Markov models	infinite HMMs	time series
factor analysis / pPCA / PMF	infinite latent factor models	feature discovery
...		

Nonlinear regression and Gaussian processes

Consider the problem of nonlinear regression:
You want to learn a function f with error bars from data $\mathcal{D} = \{\mathbf{X}, \mathbf{y}\}$



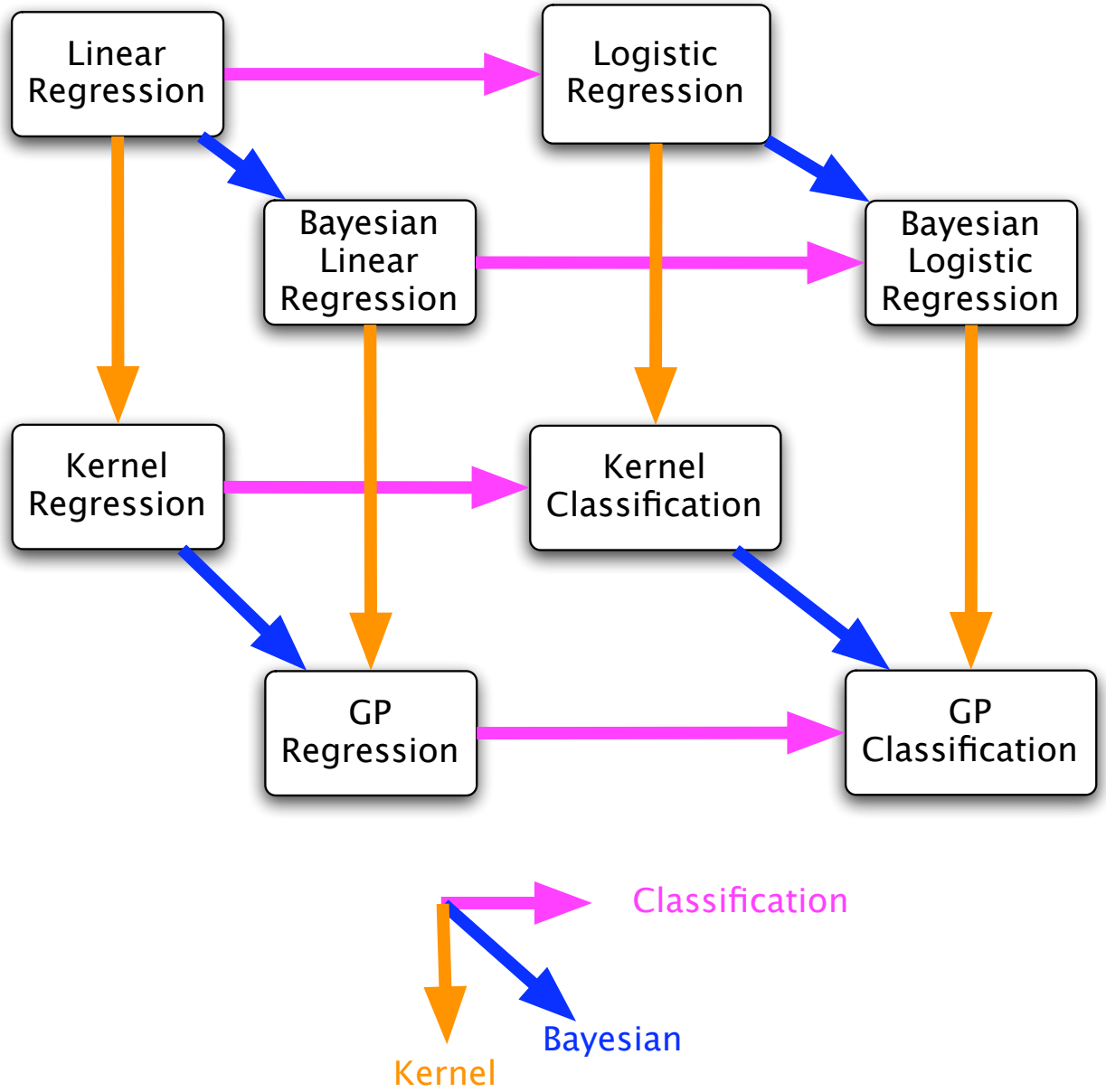
A Gaussian process defines a distribution over functions $p(f)$ which can be used for Bayesian regression:

$$p(f|\mathcal{D}) = \frac{p(f)p(\mathcal{D}|f)}{p(\mathcal{D})}$$

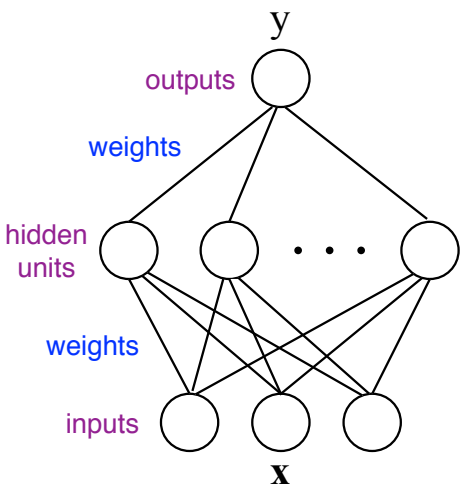
Let $\mathbf{f} = (f(x_1), f(x_2), \ldots, f(x_n))$ be an n-dimensional vector of function values evaluated at n points $x_i \in \mathcal{X}$. Note, $\mathbf{f}$ is a random variable.

Definition: $p(f)$ is a Gaussian process if for *any* finite subset $\{x_1, \ldots, x_n\} \subset \mathcal{X}$, the marginal distribution over that subset $p(\mathbf{f})$ is multivariate Gaussian.

A picture


Linear Regression
Logistic Regression
Bayesian Linear Regression
Bayesian Logistic Regression
Kernel Regression
Kernel Classification
GP Regression
GP Classification
Classification
Bayesian
Kernel

Neural networks and Gaussian processes



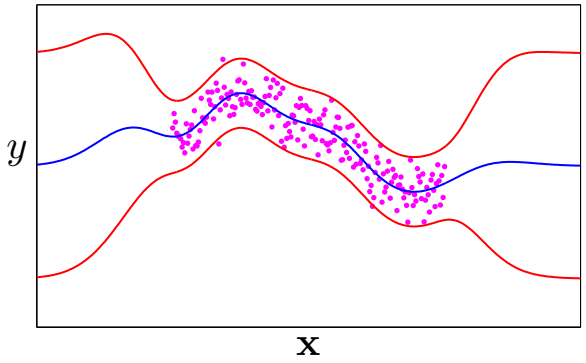
Bayesian neural network

Data: $\mathcal{D} = \{(\mathbf{x}^{(n)}, y^{(n)})\}_{n=1}^{N} = (X, \mathbf{y})$
Parameters $\boldsymbol{\theta}$ are the weights of the neural net

parameter prior	$p(\boldsymbol{\theta}\|\boldsymbol{\alpha})$
parameter posterior	$p(\boldsymbol{\theta}\|\boldsymbol{\alpha}, \mathcal{D}) \propto p(\mathbf{y}\|X, \boldsymbol{\theta})p(\boldsymbol{\theta}\|\boldsymbol{\alpha})$
prediction	$p(y'\|\mathcal{D}, \mathbf{x}', \boldsymbol{\alpha}) = \int p(y'\|\mathbf{x}', \boldsymbol{\theta})p(\boldsymbol{\theta}\|\mathcal{D}, \boldsymbol{\alpha})\, d\boldsymbol{\theta}$

A **Gaussian process** models functions $y = f(\mathbf{x})$

A multilayer perceptron (neural network) with infinitely many hidden units and Gaussian priors on the weights $\rightarrow$ a GP (Neal, 1996)



See also recent work on Deep Gaussian Processes (Damianou and Lawrence, 2013)

But surely Bayesian methods are not needed for Big Data...

- Argument: As the number of data $N \to \infty$, Bayes $\to$ maximum likelihood, prior washes out, integration becomes unneccessary!
- But this assumes we want to learn a fixed simple model from $N \to \infty$ iid data points... not really a good use of Big Data!
- More realistically, Big Data $= \{$ Large Set of little data sets $\}$, e.g. recommender systems, personalised medicine, genomes, web text, images, market baskets...
- We would really like to learn models in which the **number of parameters grows with the size of the data set** (c.f. *nonparametrics*)
- Since we still need to guard from overfitting, and represent uncertainty, a coherent way to do this is to use probabilistic models and probability theory (i.e. sum, product, Bayes rule) to learn them.

Cons and pros of Bayesian methods

Limitations and Criticisms:

- They are subjective.
- It is hard to come up with a prior, the assumptions are usually wrong.
- The closed world assumption: need to consider all possible hypotheses for the data before observing the data.
- They can be computationally demanding.
- The use of approximations weakens the coherence argument.

Advantages:

- Coherent.
- Conceptually straightforward.
- Modular.
- Often good performance.

Summary

Probabilistic (i.e. Bayesian) methods are:

- simple (just two rules)
- general (can be applied to any model)
- avoid overfitting (because you don't fit)
- are a coherent way of representing beliefs (Cox axioms)
- guard against inconsistency in decision making (Dutch books)

Some of the material in this talk is covered in these papers:

- Ghahramani, Z. (2013) Bayesian nonparametrics and the probabilistic approach to modelling. *Philosophical Transactions of the Royal Society A.* 371: 20110553.

- Ghahramani, Z. (2004) Unsupervised Learning. In Bousquet, O., von Luxburg, U. and Rätsch, G. *Advanced Lectures in Machine Learning.* Lecture Notes in Computer Science 3176, pages 72-112. Berlin: Springer-Verlag.