Bayesian machine learning

Zoubin Ghahramani

Department of Engineering
University of Cambridge, UK

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

Imperial College Lectures
2014

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.*

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.

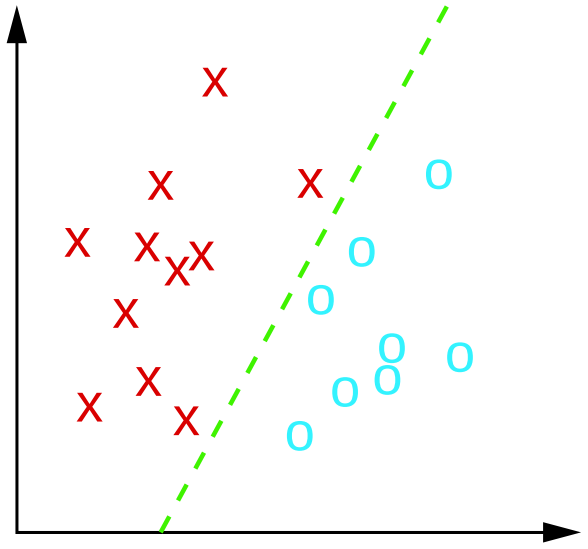
Some Canonical Machine Learning Problems

- 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:

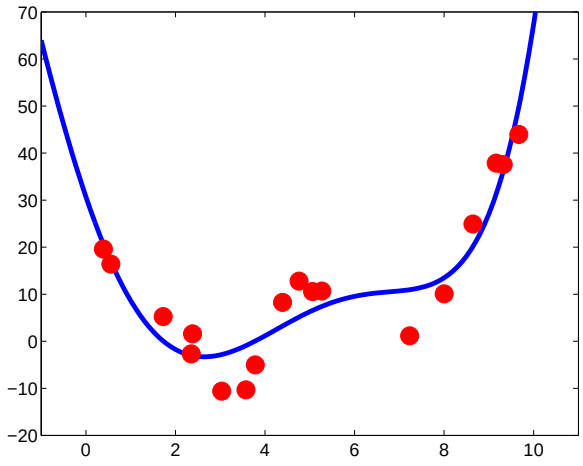
$$P(y^{(n)} = +1 | \boldsymbol{\theta}, \mathbf{x}^{(n)}) = \begin{cases} 1 & \text{if } \sum_{d=1}^{D} \theta_d \, x_d^{(n)} + \theta_0 \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

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

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

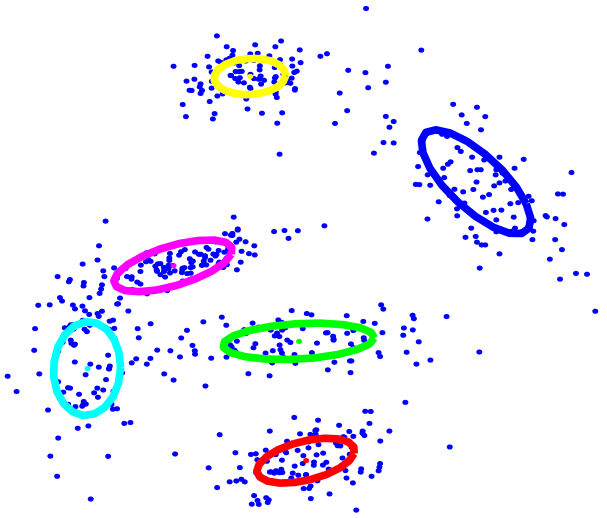
Parameters: $\boldsymbol{\theta} = (a_0, \ldots, a_m, \sigma)$

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.

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:

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

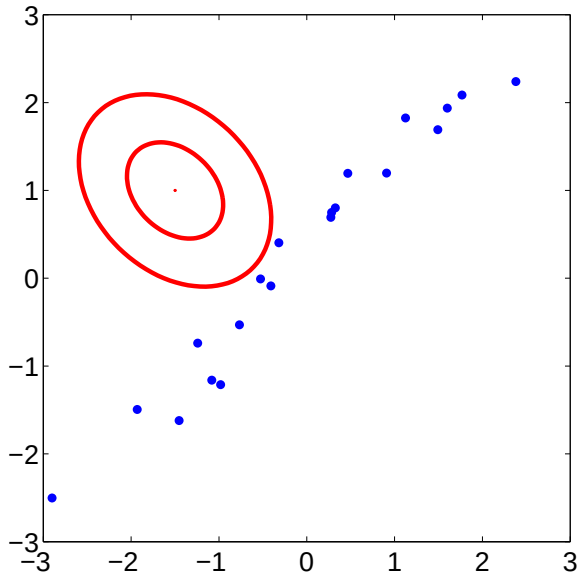
Model Comparison:

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

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

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?

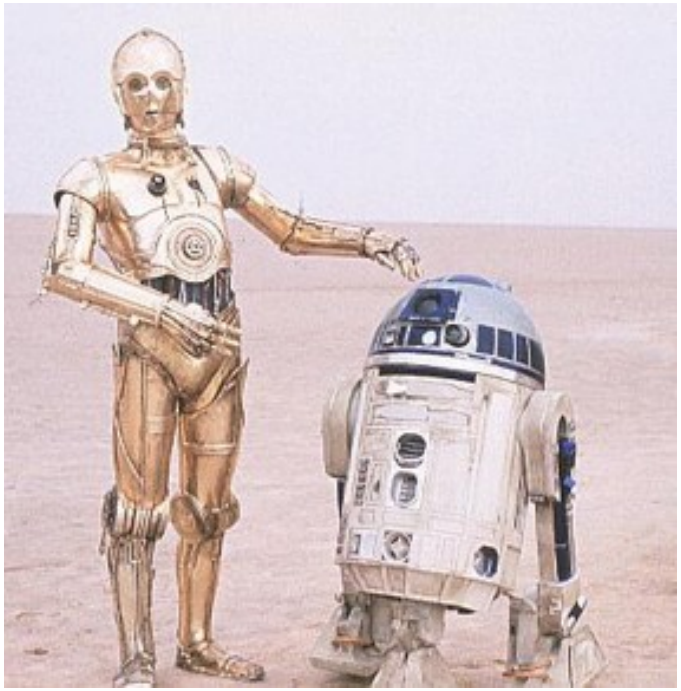
Representing Beliefs (Artificial Intelligence)

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.

Asymptotic Certainty

Assume that data set $\mathcal{D}_n$, consisting of n data points, was generated from some true θ^*, then under some regularity conditions, as long as $p(\theta^*) > 0$

$$\lim_{n\to\infty} p(\theta|\mathcal{D}_n) = \delta(\theta - \theta^*)$$

In the **unrealizable case**, where data was generated from some $p^*(x)$ which cannot be modelled by any θ, then the posterior will converge to

$$\lim_{n\to\infty} p(\theta|\mathcal{D}_n) = \delta(\theta - \hat{\theta})$$

where $\hat{\theta}$ minimizes $\mathrm{KL}(p^*(x), p(x|\theta))$:

$$\hat{\theta} = \operatorname*{argmin}_\theta \int p^*(x) \log \frac{p^*(x)}{p(x|\theta)}\, dx = \operatorname*{argmax}_\theta \int p^*(x) \log p(x|\theta)\, dx$$

Warning: careful with the regularity conditions, these are just sketches of the theoretical results

Asymptotic Consensus

Consider two Bayesians with *different priors*, $p_1(\theta)$ and $p_2(\theta)$,
who observe the *same data* $\mathcal{D}$.

Assume both Bayesians agree on the set of possible and impossible values of θ:

$$\{\theta : p_1(\theta) > 0\} = \{\theta : p_2(\theta) > 0\}$$

Then, in the limit of $n \to \infty$, the posteriors, $p_1(\theta|\mathcal{D}_n)$ and $p_2(\theta|\mathcal{D}_n)$ will converge
(in uniform distance between distributions $\rho(P_1, P_2) = \sup_E |P_1(E) - P_2(E)|$)

coin toss demo: `bayescoin`...

A simple probabilistic calculation

Consider a binary variable ("rain" or "no rain" for weather, "heads" or "tails" for a coin), $x \in \{1, 0\}$. Let's model observations $\mathcal{D} = \{x_n : n = 1 \ldots N\}$ using a Bernoulli distribution:

$$P(x_n|q) = q^{x_n}(1-q)^{(1-x_n)}$$

for $0 \leq q \leq 1$.

Q: Is this a sensible model?
Q: Do we have any other choices?

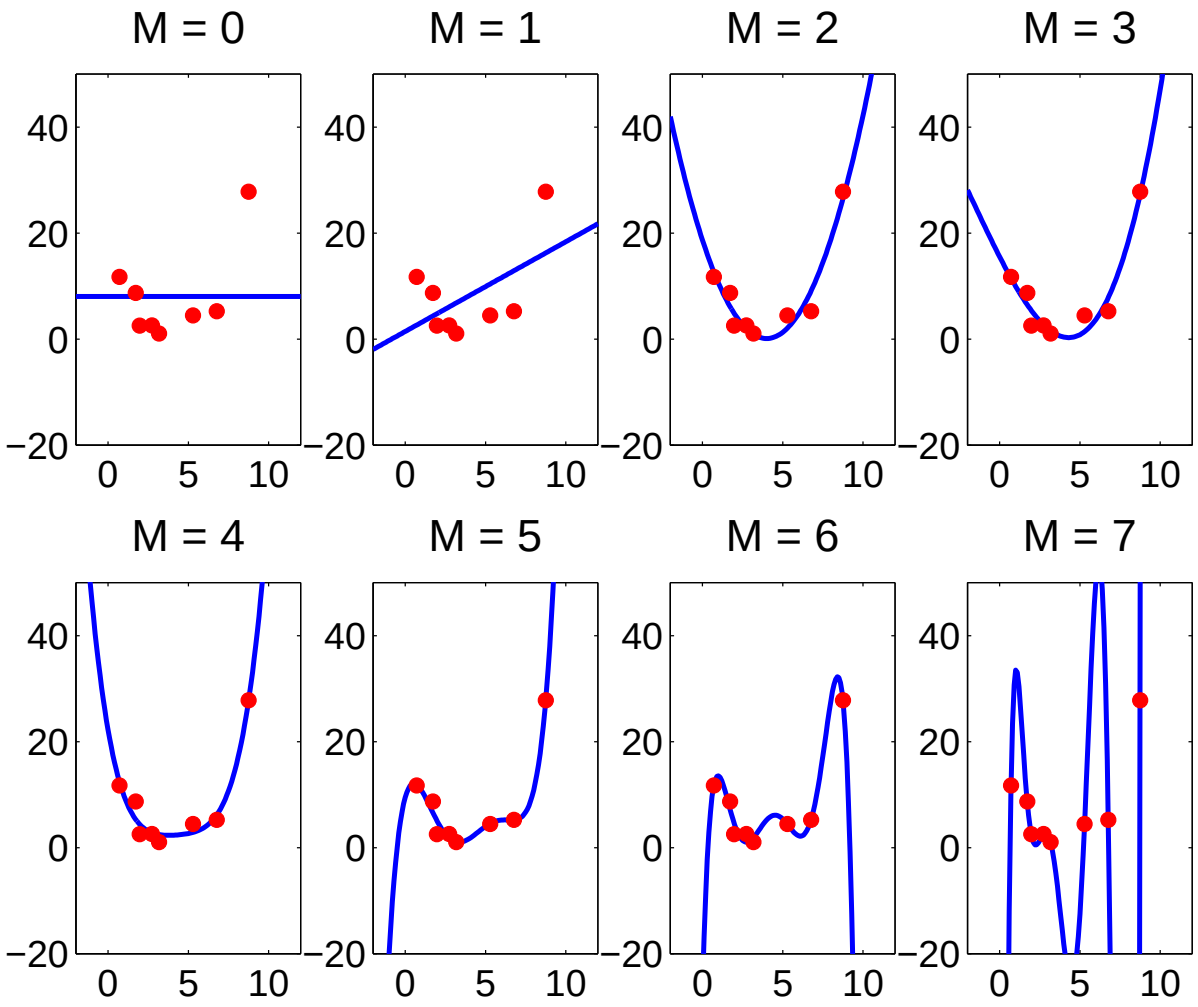
To learn the model parameter q, we need to start with a prior, and condition on the observed data $\mathcal{D}$. For example, a uniform prior would look like this:

$$P(q) = 1 \text{ for } 0 \leq q \leq 1$$

Q: What is the posterior after $x_1 = 1$?

coin toss demo: `bayescoin`

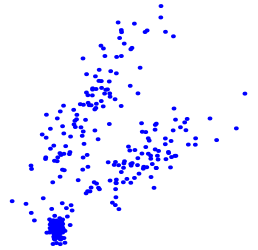
Model Selection



Learning Model Structure

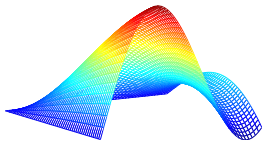
How many clusters in the data?

k-means, mixture models



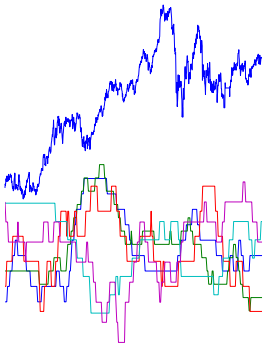
What is the intrinsic dimensionality of the data?

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

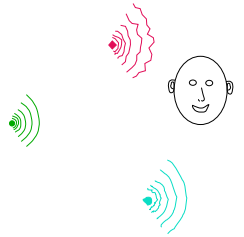
How many states in a hidden Markov model?

HMM

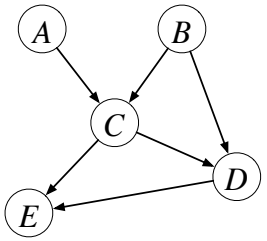
SVYDAAAQLTADVKKDLRDSWKVIGSDKKGNGVA

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 $\mathcal{D}$:

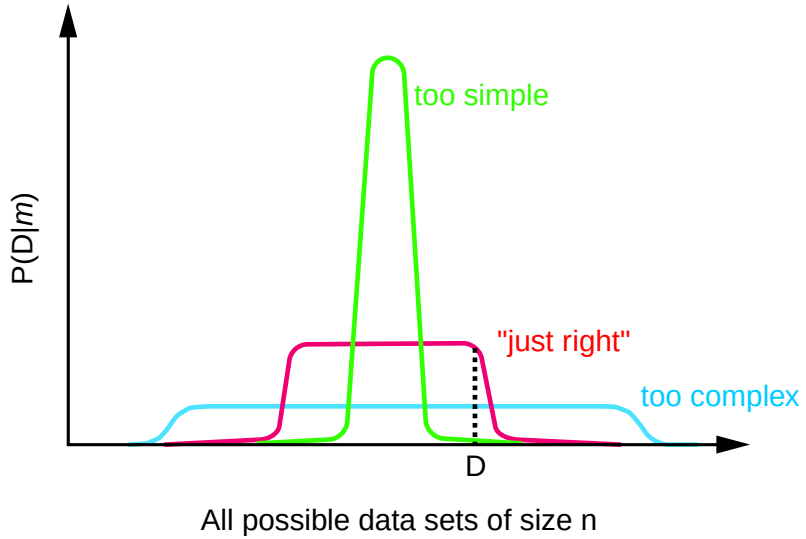
$$p(m|\mathcal{D}) = \frac{p(\mathcal{D}|m)\,p(m)}{p(\mathcal{D})}, \qquad p(\mathcal{D}|m) = \int p(\mathcal{D}|\boldsymbol{\theta}, m)\; p(\boldsymbol{\theta}|m)\; d\boldsymbol{\theta}$$

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

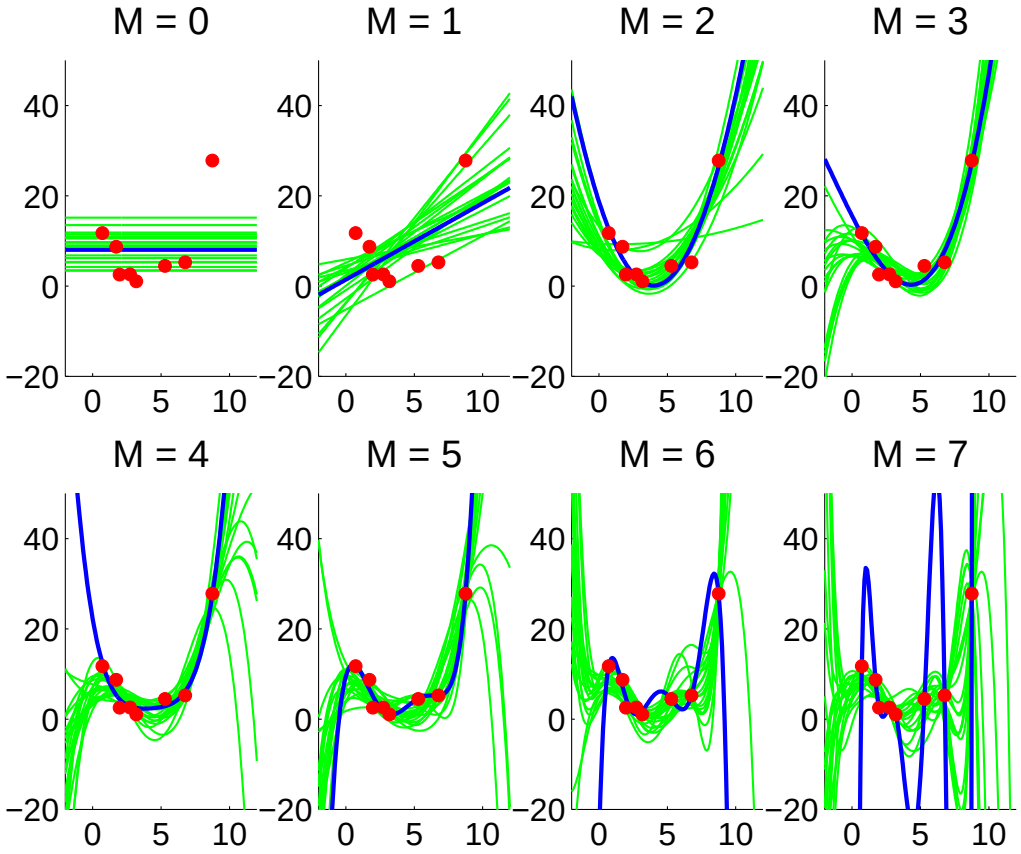
- The probability that *randomly selected* parameters from the prior would generate $\mathcal{D}$.
- Probability of the data under the model, *averaging* over all possible parameter values.
- $\log_2\left(\frac{1}{p(\mathcal{D}|m)}\right)$ is the number of *bits of surprise* at observing data $\mathcal{D}$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

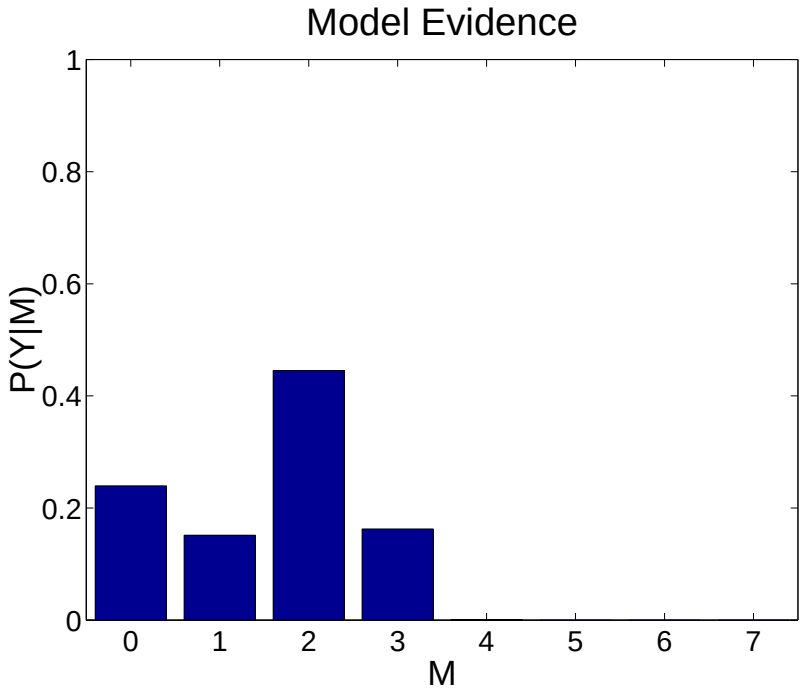
Model classes that are too simple are unlikely to generate the data set.

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.

A prior exercise: logistic regression

Consider logistic regression:[1]

$$P(y = 1|\mathbf{w}, b, \mathbf{x}) = p = \frac{1}{1 + \exp\{-\mathbf{w}^\top \mathbf{x} + b\}}$$

Assume $\mathbf{x} \in \{0, 1\}^D$. How should we choose the prior $p(\mathbf{w}, b)$?
For simplicity, let's focus on the case:

$$p(\mathbf{w}) = N(0, \sigma^2 I) \qquad p(b) = N(0, \sigma^2)$$

Q: For $D = 1$, what happens if we choose $\sigma \gg 0$ as a "non-informative" prior?

Q: How does p behave for very small σ?

Q: Repeat for $D = 10$.

Q: Repeat for $D = 1000$ where we expect $\mathbf{x}$ to be sparse (with density 1%).

Q: Can you think of a hierarchical prior which captures a range of desired behaviors?

`demo: blr`

[1] or a typical neuron in a neural network

How to choose stupid priors: a tutorial

- Choose an improper or uninformative prior so that your marginal likelihoods are meaningless.
- Alternatively, choose very dogmatic narrow priors that can't adapt to the data.
- Choose a prior that is very hard to compute with.
- After choosing your prior, don't sample from your model to see whether simulated data make sense.
- Never question the prior. Don't describe your prior in your paper, so that your work is not reproducible.

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.

Laplace Approximation

data set $\mathbf{y}$, models $m, m', \ldots,$ parameter $\boldsymbol{\theta}, \boldsymbol{\theta}' \ldots$

Model Comparison: $p(m|\mathbf{y}) \propto p(m)p(\mathbf{y}|m)$

For large amounts of data (relative to number of parameters, d) under some regularity conditions, the parameter posterior is approximately Gaussian around the MAP estimate $\hat{\boldsymbol{\theta}}$:

$$p(\boldsymbol{\theta}|\mathbf{y}, m) \approx (2\pi)^{-\frac{d}{2}}|A|^{\frac{1}{2}} \exp\left\{-\frac{1}{2}(\boldsymbol{\theta} - \hat{\boldsymbol{\theta}})^\top A\,(\boldsymbol{\theta} - \hat{\boldsymbol{\theta}})\right\}$$

where $-A$ is the $d \times d$ Hessian of the log posterior $A_{ij} = -\frac{d^2}{d\theta_i d\theta_j} \ln p(\boldsymbol{\theta}|\mathbf{y}, m)\Big|_{\boldsymbol{\theta}=\hat{\boldsymbol{\theta}}}$

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

Evaluating the above expression for $\ln p(\mathbf{y}|m)$ at $\hat{\boldsymbol{\theta}}$:

$$\ln p(\mathbf{y}|m) \approx \ln p(\hat{\boldsymbol{\theta}}|m) + \ln p(\mathbf{y}|\hat{\boldsymbol{\theta}}, m) + \frac{d}{2}\ln 2\pi - \frac{1}{2}\ln|A|$$

This can be used for model comparison/selection.

Bayesian Information Criterion (BIC)

BIC can be obtained from the Laplace approximation:

$$\ln p(\mathbf{y}|m) \approx \ln p(\hat{\boldsymbol{\theta}}|m) + \ln p(\mathbf{y}|\hat{\boldsymbol{\theta}}, m) + \frac{d}{2}\ln 2\pi - \frac{1}{2}\ln |A|$$

by taking the large sample limit $(n \to \infty)$ where n is the number of data points:

$$\ln p(\mathbf{y}|m) \approx \ln p(\mathbf{y}|\hat{\boldsymbol{\theta}}, m) - \frac{d}{2}\ln n$$

Properties:

- Quick and easy to compute
- It does not depend on the prior
- We can use the ML estimate of θ instead of the MAP estimate
- It is equivalent to the MDL criterion
- Assumes that as $n \to \infty$, all the parameters are well-determined (i.e. the model is identifiable; otherwise, d should be the number of well-determined parameters)
- **Danger:** counting parameters can be deceiving! (c.f. sinusoid, infinite models)

Lower Bounding the Marginal Likelihood

Variational Bayesian Learning

Let the latent variables be $\mathbf{x}$, observed data $\mathbf{y}$ and the parameters $\boldsymbol{\theta}$.
We can lower bound the marginal likelihood (Jensen's inequality):

$$\begin{aligned}
\ln p(\mathbf{y}|m) &= \ln \int p(\mathbf{y}, \mathbf{x}, \boldsymbol{\theta}|m)\, d\mathbf{x}\, d\boldsymbol{\theta} \\
&= \ln \int q(\mathbf{x}, \boldsymbol{\theta}) \frac{p(\mathbf{y}, \mathbf{x}, \boldsymbol{\theta}|m)}{q(\mathbf{x}, \boldsymbol{\theta})}\, d\mathbf{x}\, d\boldsymbol{\theta} \\
&\geq \int q(\mathbf{x}, \boldsymbol{\theta}) \ln \frac{p(\mathbf{y}, \mathbf{x}, \boldsymbol{\theta}|m)}{q(\mathbf{x}, \boldsymbol{\theta})}\, d\mathbf{x}\, d\boldsymbol{\theta}.
\end{aligned}$$

Use a simpler, factorised approximation for $q(\mathbf{x}, \boldsymbol{\theta}) \approx q_{\mathbf{x}}(\mathbf{x}) q_{\boldsymbol{\theta}}(\boldsymbol{\theta})$:

$$\begin{aligned}
\ln p(\mathbf{y}|m) &\geq \int q_{\mathbf{x}}(\mathbf{x}) q_{\boldsymbol{\theta}}(\boldsymbol{\theta}) \ln \frac{p(\mathbf{y}, \mathbf{x}, \boldsymbol{\theta}|m)}{q_{\mathbf{x}}(\mathbf{x}) q_{\boldsymbol{\theta}}(\boldsymbol{\theta})}\, d\mathbf{x}\, d\boldsymbol{\theta} \\
&\stackrel{\text{def}}{=} \mathcal{F}_m(q_{\mathbf{x}}(\mathbf{x}), q_{\boldsymbol{\theta}}(\boldsymbol{\theta}), \mathbf{y}).
\end{aligned}$$

Variational Bayesian Learning . . .

Maximizing this lower bound, $\mathcal{F}_m$, leads to **EM-like** iterative updates:

$$q_{\mathbf{x}}^{(t+1)}(\mathbf{x}) \quad \propto \quad \exp\left[\int \ln p(\mathbf{x},\mathbf{y}|\boldsymbol{\theta},m)\, q_{\boldsymbol{\theta}}^{(t)}(\boldsymbol{\theta})\, d\boldsymbol{\theta}\right] \qquad \text{E-like step}$$

$$q_{\boldsymbol{\theta}}^{(t+1)}(\boldsymbol{\theta}) \quad \propto \quad p(\boldsymbol{\theta}|m)\ \exp\left[\int \ln p(\mathbf{x},\mathbf{y}|\boldsymbol{\theta},m)\, q_{\mathbf{x}}^{(t+1)}(\mathbf{x})\, d\mathbf{x}\right] \qquad \text{M-like step}$$

Maximizing $\mathcal{F}_m$ is equivalent to minimizing KL-divergence between the *approximate posterior*, $q_{\boldsymbol{\theta}}(\boldsymbol{\theta})\ q_{\mathbf{x}}(\mathbf{x})$ and the *exact posterior*, $p(\boldsymbol{\theta},\mathbf{x}|\mathbf{y},m)$:

$$\ln p(\mathbf{y}|m) - \mathcal{F}_m(q_{\mathbf{x}}(\mathbf{x}), q_{\boldsymbol{\theta}}(\boldsymbol{\theta}), \mathbf{y}) = \int q_{\mathbf{x}}(\mathbf{x})\ q_{\boldsymbol{\theta}}(\boldsymbol{\theta}) \ln \frac{q_{\mathbf{x}}(\mathbf{x})\ q_{\boldsymbol{\theta}}(\boldsymbol{\theta})}{p(\boldsymbol{\theta},\mathbf{x}|\mathbf{y},m)}\, d\mathbf{x}\, d\boldsymbol{\theta} = \mathbf{KL}(q\|p)$$

In the limit as $n \to \infty$, for identifiable models, the variational lower bound approaches the BIC criterion.

The Variational Bayesian EM algorithm

EM for MAP estimation

Goal: maximize $p(\boldsymbol{\theta}|\mathbf{y}, m)$ w.r.t. $\boldsymbol{\theta}$

E Step: compute

$$q_{\mathbf{x}}^{(t+1)}(\mathbf{x}) = p(\mathbf{x}|\mathbf{y}, \boldsymbol{\theta}^{(t)})$$

M Step:

$$\boldsymbol{\theta}^{(t+1)} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \int q_{\mathbf{x}}^{(t+1)}(\mathbf{x}) \ln p(\mathbf{x}, \mathbf{y}, \boldsymbol{\theta}) \, d\mathbf{x}$$

Variational Bayesian EM

Goal: lower bound $p(\mathbf{y}|m)$

VB-E Step: compute

$$q_{\mathbf{x}}^{(t+1)}(\mathbf{x}) = p(\mathbf{x}|\mathbf{y}, \bar{\boldsymbol{\phi}}^{(t)})$$

VB-M Step:

$$q_{\boldsymbol{\theta}}^{(t+1)}(\boldsymbol{\theta}) \propto \exp\left[\int q_{\mathbf{x}}^{(t+1)}(\mathbf{x}) \ln p(\mathbf{x}, \mathbf{y}, \boldsymbol{\theta}) \, d\mathbf{x}\right]$$

Properties:

- Reduces to the EM algorithm if $q_{\boldsymbol{\theta}}(\boldsymbol{\theta}) = \delta(\boldsymbol{\theta} - \boldsymbol{\theta}^*)$.
- $\mathcal{F}_m$ increases monotonically, and incorporates the model complexity penalty.
- Analytical parameter distributions (but not constrained to be Gaussian).
- VB-E step has same complexity as corresponding E step.
- We can use the junction tree, belief propagation, Kalman filter, etc, algorithms in the VB-E step of VB-EM, but **using expected natural parameters**, $\bar{\boldsymbol{\phi}}$.

Variational Bayesian EM

The Variational Bayesian EM algorithm has been used to approximate Bayesian learning in a wide range of models such as:

- probabilistic PCA and factor analysis
- mixtures of Gaussians and mixtures of factor analysers
- hidden Markov models
- state-space models (linear dynamical systems)
- independent components analysis (ICA)
- discrete graphical models...

The main advantage is that it can be used to **automatically do model selection** and does not suffer from overfitting to the same extent as ML methods do.

Also it is about as computationally demanding as the usual EM algorithm.

See: `www.variational-bayes.org`

mixture of Gaussians demo: `run_simple`

An Overview of Sampling Methods

Monte Carlo Methods:

- Simple Monte Carlo
- Rejection Sampling
- Importance Sampling
- etc.

Markov Chain Monte Carlo Methods:

- Gibbs Sampling
- Metropolis Algorithm
- Hybrid Monte Carlo
- etc.

Exact Sampling Methods

Markov chain Monte Carlo (MCMC) methods

Assume we are interested in drawing samples from some desired distribution $p^*(\theta)$, e.g. $p^*(\theta) = p(\theta|\mathcal{D}, m)$.

We define a Markov chain:

$$\theta_0 \to \theta_1 \to \theta_2 \to \theta_3 \to \theta_4 \to \theta_5 \ldots$$

where $\theta_0 \sim p_0(\theta)$, $\theta_1 \sim p_1(\theta)$, etc, with the property that:

$$p_t(\theta') = \int p_{t-1}(\theta)\ T(\theta \to \theta')\, d\theta$$

where $T(\theta \to \theta')$ is the **Markov chain transition probability** from θ to θ'.

We say that $p^*(\theta)$ is an **invariant (or stationary) distribution** of the Markov chain defined by T iff:

$$p^*(\theta') = \int p^*(\theta)\, T(\theta \to \theta')\, d\theta$$

Markov chain Monte Carlo (MCMC) methods

We have a Markov chain $\theta_0 \to \theta_1 \to \theta_2 \to \theta_3 \to \ldots$ where $\theta_0 \sim p_0(\theta)$, $\theta_1 \sim p_1(\theta)$, etc, with the property that:

$$p_t(\theta') = \int p_{t-1}(\theta)\ T(\theta \to \theta')\, d\theta$$

where $T(\theta \to \theta')$ is the Markov chain transition probability from θ to θ'.
A useful condition that implies invariance of $p^*(\theta)$ is **detailed balance**:

$$p^*(\theta')\ T(\theta' \to \theta) = p^*(\theta)\ T(\theta \to \theta')$$

MCMC methods define **ergodic** Markov chains, which converge to a unique stationary distribution (also called an *equilibrium distribution*) regardless of the initial conditions $p_0(\theta)$:

$$\lim_{t \to \infty} p_t(\theta) = p^*(\theta)$$

Procedure: define an MCMC method with equilibrium distribution $p(\theta|\mathcal{D}, m)$, run method and collect samples. There are also sampling methods for $p(\mathcal{D}|m)$.

`demos?`

Discussion

Myths and misconceptions about Bayesian methods

- **Bayesian methods make assumptions where other methods don't**
  All methods make assumptions! Otherwise it's impossible to predict. Bayesian methods are transparent in their assumptions whereas other methods are often opaque.

- **If you don't have the right prior you won't do well**
  Certainly a poor model will predict poorly but there is no such thing as the right prior! Your model (both prior and likelihood) should capture a reasonable range of possibilities. When in doubt you can choose vague priors (cf nonparametrics).

- **Maximum A Posteriori (MAP) is a Bayesian method**
  MAP is similar to regularization and offers no particular Bayesian advantages. The key ingredient in Bayesian methods is to average over your uncertain variables and parameters, rather than to optimize.

Myths and misconceptions about Bayesian methods

- **Bayesian methods don't have theoretical guarantees**
  One can often apply frequentist style generalization error bounds to Bayesian methods (e.g. PAC-Bayes). Moreover, it is often possible to prove frequentist convergence, consistency and rates for Bayesian methods.

- **Bayesian methods are generative**
  You can use Bayesian approaches for both generative and discriminative learning (e.g. Gaussian process classification).

- **Bayesian methods don't scale well**
  With the right inference methods (variational, MCMC) it is possible to scale to very large datasets (e.g. excellent results for Bayesian Probabilistic Matrix Factorization on the Netflix dataset using MCMC), but it's true that averaging/integration is often more expensive than optimization.

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.

TrueSkill: a planetary scale Bayesian model

Microsoft Research

Search Micr

Our research Connections Careers About us

All Downloads Events Groups News People Projects Publications

TrueSkill™ Ranking System

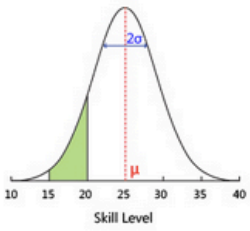
The TrueSkill™ ranking system is a skill based ranking system for Xbox Live developed at Microsoft Research.

The TrueSkill ranking system is a skill based ranking system for Xbox Live developed at Microsoft Research. The purpose of a ranking system is to both identify and track the skills of gamers in a game (mode) in order to be able to match them into competitive matches. The TrueSkill ranking system only uses the final standings of all teams in a game in order to update the skill estimates (ranks) of all gamers playing in this game. Ranking systems have been proposed for many sports but possibly the most prominent ranking system in use today is ELO.

Ranking Players

So, what is so special about the TrueSkill ranking system? In short, the biggest difference to other ranking systems is that in the TrueSkill ranking system skill is characterised by **two** numbers:

- The average skill of the gamer (μ in the picture).
- The degree of uncertainty in the gamer's skill (σ in the picture).



The ranking system maintains a **belief** in every gamer's skill using these two numbers. If the uncertainty is still high, the ranking system does not yet know exactly the skill of the gamer. In contrast, if the uncertainty is small, the ranking system has a strong belief that the skill of the gamer is close to the average skill.

On the right hand side, a belief curve of the TrueSkill ranking system is drawn. For example, the green area is the belief of the TrueSkill ranking system that the gamer has a skill between level 15 and 20.

Maintaining an uncertainty allows the system to make big changes to the skill estimates early on but small changes after a series of consistent games has been played. As a result, the TrueSkill ranking

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 further reading:

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

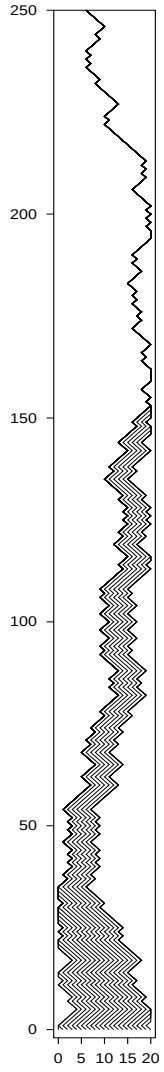
- Ghahramani, Z. (2004) Unsupervised Learning. In Bousquet, O., von Luxburg, U. and Rätsch, G. *Advanced Lectures in Machine Learning.* Lecture Notes in Computer Science 3176, pages 72-112. Berlin: Springer-Verlag.

Thanks for your patience!

Appendix

Exact Sampling

a.k.a. perfect simulation, coupling from the past



(from MacKay 2003)

- Coupling: running multiple Markov chains (MCs) using the same random seeds. E.g. imagine starting a Markov chain at each possible value of the state (θ).
- Coalescence: if two coupled MCs end up at the same state at time t, then they will forever follow the same path.
- Monotonicity: Rather than running an MC starting from every state, find a partial ordering of the states preserved by the coupled transitions, and track the highest and lowest elements of the partial ordering. When these coalesce, MCs started from all initial states would have coalesced.
- Running from the past: Start at $t = -K$ in the past, if highest and lowest elements of the MC have coalesced by time $t = 0$ then all MCs started at $t = -\infty$ would have coalesced, therefore the chain must be at equilibrium, therefore $\theta_0 \sim p^*(\theta)$.

Bottom Line This procedure, *when* it produces a sample, will produce one from the *exact* distribution $p^*(\theta)$.

Expectation Propagation (EP)

Data (iid) $\mathcal{D} = \{\mathbf{x}^{(1)} \dots, \mathbf{x}^{(N)}\}$, model $p(\mathbf{x}|\boldsymbol{\theta})$, with parameter prior $p(\boldsymbol{\theta})$.

The parameter posterior is: $$p(\boldsymbol{\theta}|\mathcal{D}) = \frac{1}{p(\mathcal{D})} p(\boldsymbol{\theta}) \prod_{i=1}^{N} p(\mathbf{x}^{(i)}|\boldsymbol{\theta})$$

We can write this as product of factors over $\boldsymbol{\theta}$: $$p(\boldsymbol{\theta}) \prod_{i=1}^{N} p(\mathbf{x}^{(i)}|\boldsymbol{\theta}) = \prod_{i=0}^{N} f_i(\boldsymbol{\theta})$$

where $f_0(\boldsymbol{\theta}) \stackrel{\text{def}}{=} p(\boldsymbol{\theta})$ and $f_i(\boldsymbol{\theta}) \stackrel{\text{def}}{=} p(\mathbf{x}^{(i)}|\boldsymbol{\theta})$ and we will ignore the constants.

We wish to approximate this by a product of *simpler* terms: $$q(\boldsymbol{\theta}) \stackrel{\text{def}}{=} \prod_{i=0}^{N} \tilde{f}_i(\boldsymbol{\theta})$$

$$\min_{q(\boldsymbol{\theta})} \mathrm{KL}\left(\prod_{i=0}^{N} f_i(\boldsymbol{\theta}) \middle\| \prod_{i=0}^{N} \tilde{f}_i(\boldsymbol{\theta}) \right) \quad \text{(intractable)}$$

$$\min_{\tilde{f}_i(\boldsymbol{\theta})} \mathrm{KL}\left(f_i(\boldsymbol{\theta}) \| \tilde{f}_i(\boldsymbol{\theta}) \right) \quad \text{(simple, non-iterative, inaccurate)}$$

$$\min_{\tilde{f}_i(\boldsymbol{\theta})} \mathrm{KL}\left(f_i(\boldsymbol{\theta}) \prod_{j \neq i} \tilde{f}_j(\boldsymbol{\theta}) \middle\| \tilde{f}_i(\boldsymbol{\theta}) \prod_{j \neq i} \tilde{f}_j(\boldsymbol{\theta}) \right) \quad \text{(simple, iterative, accurate)} \leftarrow \text{EP}$$

Expectation Propagation

Input $f_0(\boldsymbol{\theta}) \dots f_N(\boldsymbol{\theta})$
Initialize $\tilde{f}_0(\boldsymbol{\theta}) = f_0(\boldsymbol{\theta})$, $\tilde{f}_i(\boldsymbol{\theta}) = 1$ for $i > 0$, $q(\boldsymbol{\theta}) = \prod_i \tilde{f}_i(\boldsymbol{\theta})$
repeat
  for $i = 0 \dots N$ **do**
    Deletion: $q_{\backslash i}(\boldsymbol{\theta}) \leftarrow \dfrac{q(\boldsymbol{\theta})}{\tilde{f}_i(\boldsymbol{\theta})} = \prod_{j \neq i} \tilde{f}_j(\boldsymbol{\theta})$
    Projection: $\tilde{f}_i^{\text{new}}(\boldsymbol{\theta}) \leftarrow \arg\min_{f(\boldsymbol{\theta})} \mathrm{KL}(f_i(\boldsymbol{\theta}) q_{\backslash i}(\boldsymbol{\theta}) \| f(\boldsymbol{\theta}) q_{\backslash i}(\boldsymbol{\theta}))$
    Inclusion: $q(\boldsymbol{\theta}) \leftarrow \tilde{f}_i^{\text{new}}(\boldsymbol{\theta})\, q_{\backslash i}(\boldsymbol{\theta})$
  end for
until convergence

The EP algorithm. Some variations are possible: here we assumed that f_0 is in the exponential family, and we updated sequentially over i. The names for the steps (deletion, projection, inclusion) are not the same as in (Minka, 2001)

- Minimizes the opposite KL to variational methods
- $\tilde{f}_i(\boldsymbol{\theta})$ in exponential family $\rightarrow$ projection step is moment matching
- Loopy belief propagation and assumed density filtering are special cases
- No convergence guarantee (although convergent forms can be developed)

Reconciling Bayesian and Frequentist Views

Frequentist theory tends to focus on **sampling properties** of estimators, i.e. what would have happened had we observed other data sets from our model. Also look at **minimax performance** of methods – i.e. what is the worst case performance if the environment is adversarial. Frequentist methods often optimize some penalized cost function.

Bayesian methods focus on **expected loss** under the posterior. Bayesian methods generally do not make use of optimization, except at the point at which decisions are to be made.

There are some reasons why frequentist procedures are useful to Bayesians:

- **Communication:** If Bayesian A wants to convince Bayesians B, C, and D of the validity of some inference (or even non-Bayesians) then he or she must determine that not only does this inference follows from prior p_A but also would have followed from p_B, p_C and p_D, etc. For this reason it's useful sometimes to find a prior which has good frequentist (sampling / worst-case) properties, even though acting on the prior would not be coherent with our beliefs.
- **Robustness:** Priors with good frequentist properties can be more robust to mis-specifications of the prior. Two ways of dealing with robustness issues are to make sure that the prior is vague enough, and to make use of a loss function to penalize costly errors.

also, see PAC-Bayesian frequentist bounds on Bayesian procedures.

Two views of machine learning

- The goal of machine learning is to produce general purpose black-box algorithms for learning. I should be able to put my algorithm online, so lots of people can download it. If people want to apply it to problems A, B, C, D... then it should work regardless of the problem, and the user should not have to think too much.

- If I want to solve problem A it seems silly to use some general purpose method that was never designed for A. I should really try to understand what problem A is, learn about the properties of the data, and use as much expert knowledge as I can. Only then should I think of designing a method to solve A.

"I know nothing about the subject, but I'm happy to give you my expert opinion."