Unsupervised Learning

Week 1: Introduction, Statistical Basics, and a bit of Information Theory

Zoubin Ghahramani
zoubin@gatsby.ucl.ac.uk

Gatsby Computational Neuroscience Unit, and
MSc in Intelligent Systems, Dept Computer Science
University College London

Term 1, Autumn 2004

Three Types of Learning

Imagine an organism or machine which experiences a series of sensory inputs:

$$x_1, x_2, x_3, x_4, \ldots$$

Supervised learning: The machine is also given desired outputs $y_1, y_2, \ldots$, and its goal is to learn to produce the correct output given a new input.

Unsupervised learning: The goal of the machine is to build a model of x that can be used for reasoning, decision making, predicting things, communicating etc.

Reinforcement learning: The machine can also produce actions $a_1, a_2, \ldots$ which affect the state of the world, and receives rewards (or punishments) $r_1, r_2, \ldots$. Its goal is to learn to act in a way that maximises rewards in the long term.

Goals of Supervised Learning

Classification: The desired outputs y_i are discrete class labels.
The goal is to classify new inputs correctly (i.e. to generalize).

Regression: The desired outputs y_i are continuous valued.
The goal is to predict the output accurately for new inputs.

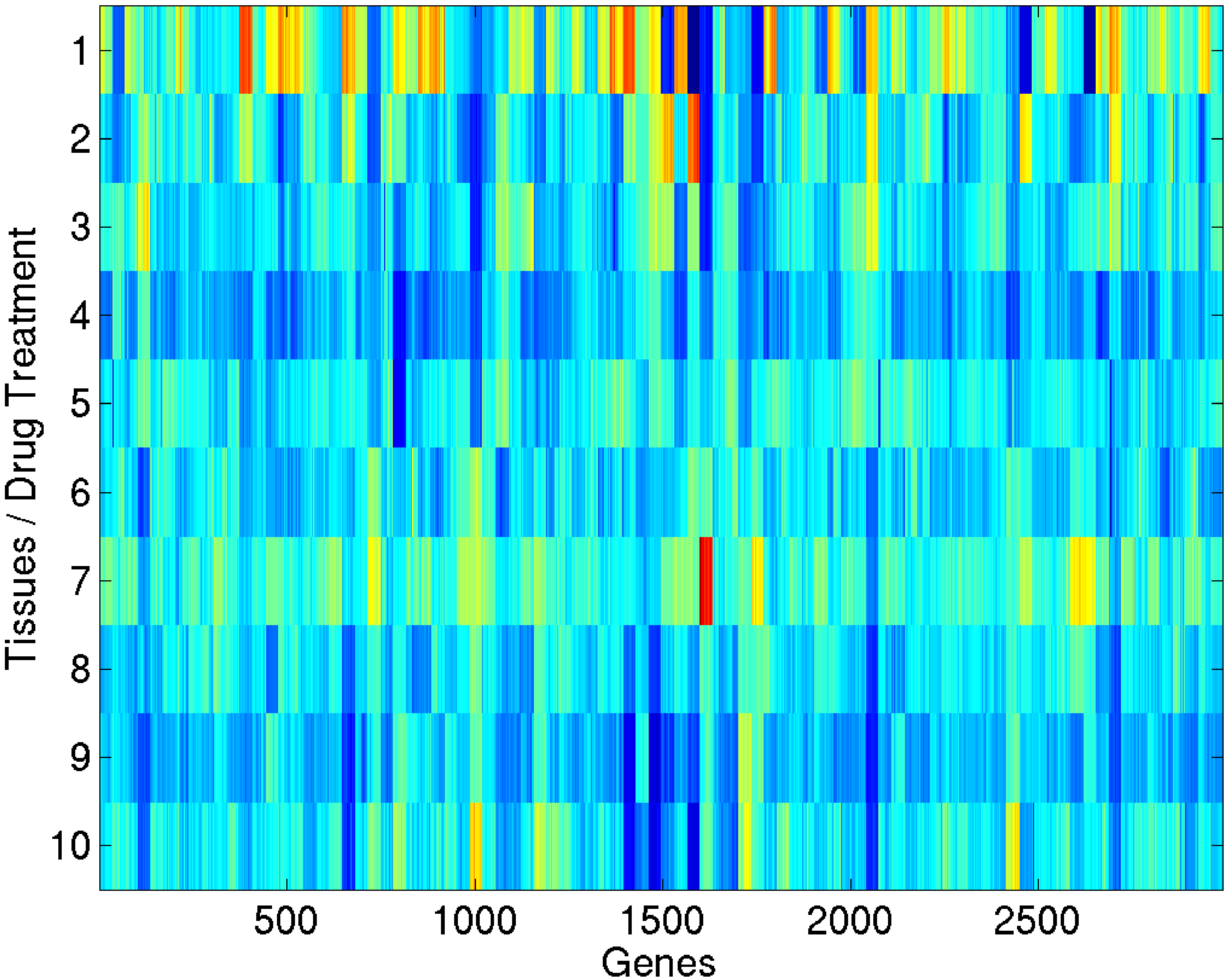
Goals of Unsupervised Learning

To build a model or find useful representations of the data, for example:

- finding clusters
- dimensionality reduction
- finding the hidden causes or sources of the data
- modeling the data density

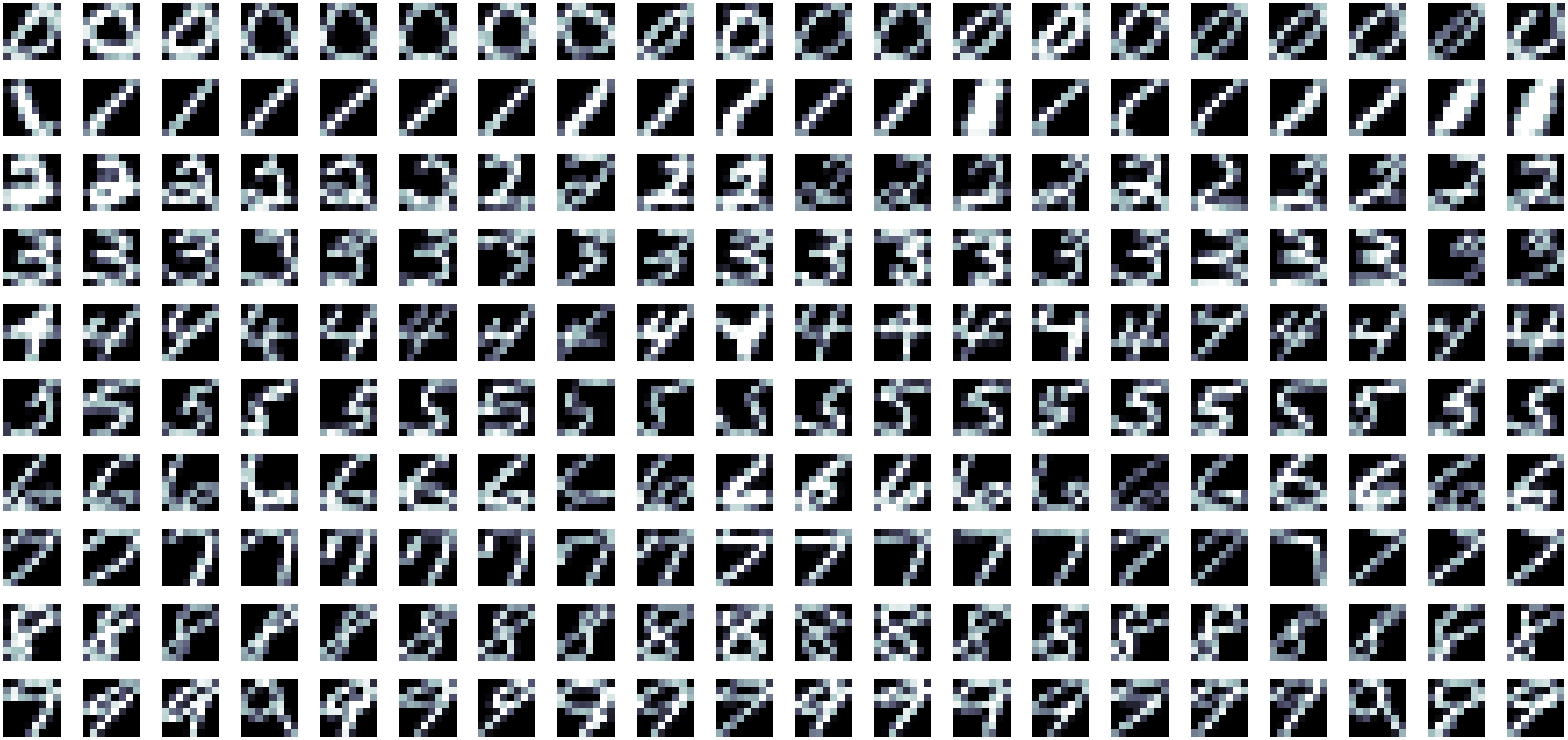
Uses of Unsupervised Learning

- data compression
- outlier detection
- classification
- make other learning tasks easier
- a theory of human learning and perception



Tissues / Drug Treatment
1
2
3
4
5
6
7
8
9
10
500
1000
1500
2000
2500
Genes

Handwritten Digits



Web Pages

Categorisation
Clustering
Relations between pages

Web Images Groups News Froogle **more »**

Unsupervised Learning | Search | Advanced Search Preferences

Web Results **1** - **10** of about **150,000** for **Unsupervised Learning**. (**0.27** seconds)

Mixture modelling, Clustering, Intrinsic classification ...
Mixture Modelling page. Welcome to David Dowe's clustering, mixture modelling and **unsupervised learning** page. Mixture modelling (or **...**
www.csse.monash.edu.au/~dld/mixture.modelling.page.html - 26k - 4 Oct 2004 - Cached - Similar pages

ACL'99 Workshop -- **Unsupervised Learning** in Natural Language ...
PROGRAM. ACL'99 Workshop **Unsupervised Learning** in Natural Language Processing. University of Maryland June 21, 1999. Endorsed by SIGNLL **...**
www.ai.sri.com/~kehler/unsup-acl-99.html - 5k - Cached - Similar pages

Unsupervised learning and Clustering
cgm.cs.mcgill.ca/~soss/cs644/projects/wijhe/ - 1k - Cached - Similar pages

NIPS*98 Workshop - Integrating Supervised and **Unsupervised** ...
NIPS*98 Workshop "Integrating Supervised and **Unsupervised Learning**" Friday, December 4, 1998. **...** 4:45-5:30, Theories of **Unsupervised Learning** and Missing Values. **...**
www-2.cs.cmu.edu/~mccallum/supunsup/ - 7k - Cached - Similar pages

NIPS Tutorial 1999
Probabilistic Models for **Unsupervised Learning** Tutorial presented at the 1999 NIPS Conference by Zoubin Ghahramani and Sam Roweis. **...**
www.gatsby.ucl.ac.uk/~zoubin/NIPStutorial.html - 4k - Cached - Similar pages

Gatsby Course: **Unsupervised Learning** : Homepage
Unsupervised Learning (Fall 2000). **...** Syllabus (resources page): 10/10 1 - Introduction to **Unsupervised Learning** Geoff project: (ps, pdf). **...**
www.gatsby.ucl.ac.uk/~quaid/course/ - 15k - Cached - Similar pages
[More results from www.gatsby.ucl.ac.uk]

[PDF] **Unsupervised Learning** of the Morphology of a Natural Language
File Format: PDF/Adobe Acrobat - View as HTML
Page 1. Page 2. Page 3. Page 4. Page 5. Page 6. Page 7. Page 8. Page 9. Page 10. Page 11. Page 12. Page 13. Page 14. Page 15. Page 16. Page 17. Page 18. Page 19 **...**
acl.ldc.upenn.edu/J/J01/J01-2001.pdf - Similar pages

Unsupervised Learning - The MIT Press
... From Bradford Books: **Unsupervised Learning** Foundations of Neural Computation Edited by Geoffrey Hinton and Terrence J. Sejnowski Since its founding in 1989 by **...**
mitpress.mit.edu/book-home.tcl?isbn=026258168X - 13k - Cached - Similar pages

[PS] **Unsupervised Learning** of Disambiguation Rules for Part of
File Format: Adobe PostScript - View as Text
Unsupervised Learning of Disambiguation Rules for Part of. Speech Tagging. Eric Brill. 1. **...** It is possible to use **unsupervised learning** to train stochastic. **...**
www.cs.jhu.edu/~brill/acl-wkshp.ps - Similar pages

The **Unsupervised Learning** Group (ULG) at UT Austin
The **Unsupervised Learning** Group (ULG). What ? The **Unsupervised Learning** Group (ULG) is a group of graduate students from the Computer **...**
www.lans.ece.utexas.edu/ulg/ - 14k - Cached - Similar pages

Result Page: 1 2 3 4 5 6 7 8 9 10 **Next**

Why a statistical approach?

- A probabilistic model of the data can be used to
  - make inferences about missing inputs
  - generate predictions/fantasies/imagery
  - make decisions which minimise expected loss
  - communicate the data in an efficient way

- Statistical modelling is equivalent to other views of learning:
  - information theoretic: finding compact representations of the data
  - physical analogies: minimising free energy of a corresponding statistical mechanical system

Information, Probability and Entropy

Information is the reduction of uncertainty. How do we measure uncertainty?

Some axioms (informal):

- if something is certain its uncertainty $= 0$

- uncertainty should be maximum if all choices are equally probable

- uncertainty (information) should add for independent sources

This leads to a discrete random variable X having uncertainty equal to the entropy function:

$$H(X) = -\sum_{X=x} P(X=x) \log P(X=x)$$

measured in *bits* (**bi**nary digi**ts**) if the base 2 logarithm is used or *nats* (**na**tural digi**ts**) if the natural (base e) logarithm is used.

Some Definitions and Intuitions

- Surprise (for event $X = x$): $-\log P(X = x)$
- Entropy = average surpise: $H(X) = -\sum_{X=x} P(X = x)\ \log_2 P(X = x)$
- Conditional entropy

$$H(X|Y) = -\sum_x \sum_y P(x, y) \log_2 P(x|y)$$

- Mutual information

$$I(X;Y) = H(X) - H(X|Y) = H(Y) - H(Y|X) = H(X) + H(Y) - H(X,Y)$$

- Kullback-Leibler divergence (relative entropy)

$$KL(P(X)\|Q(X)) = \sum_x P(x) \log \frac{P(x)}{Q(x)}$$

- Relation between Mutual information and KL: $I(X;Y) = KL(P(X,Y)\|P(X)P(Y))$
- Independent random variables: $P(X,Y) = P(X)P(Y)$
- Conditional independence $X \perp\!\!\!\perp Y|Z$ (X conditionally independent of Y given Z) means $P(X,Y|Z) = P(X|Z)P(Y|Z)$ and $P(X|Y,Z) = P(X|Z)$

Shannon's Source Coding Theorem

A discrete random variable X, distributed according to $P(X)$ has entropy equal to:

$$H(X) = -\sum_{x} P(x) \log P(x)$$

Shannon's source coding theorem: n independent samples of the random variable X, with entropy $H(X)$, can be compressed into minimum expected code of length $n\mathcal{L}$, where

$$H(X) \leq \mathcal{L} < H(X) + \frac{1}{n}$$

If each symbol is given a code length $l(x) = -\log_2 Q(x)$ then the expected per-symbol length $\mathcal{L}_Q$ of the code is

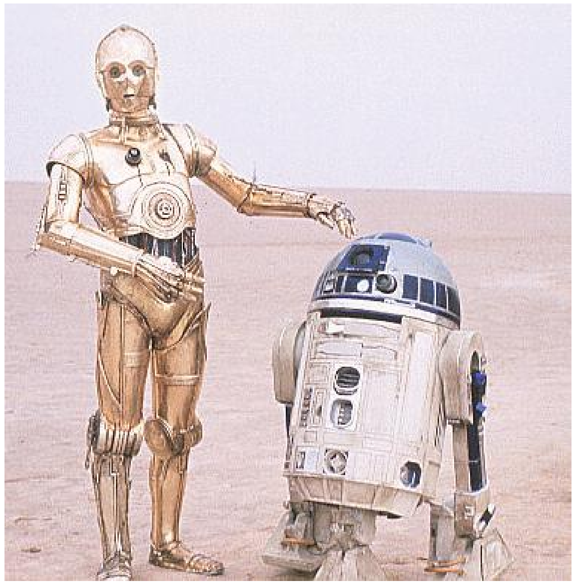
$$H(X) + KL(P\|Q) \leq \mathcal{L}_Q < H(X) + KL(P\|Q) + \frac{1}{n},$$

where the relative-entropy or Kullback-Leibler divergence is

$$KL(P\|Q) = \sum_{x} P(x) \log \frac{P(x)}{Q(x)} \geq 0$$

Learning: A Statistical Approach II

- Goal: to represent the beliefs of learning agents.
- Cox Axioms lead to the following:
  If plausibilities/beliefs are represented by real numbers, then the only reasonable and consistent way to manipulate them is Bayes rule.
- Frequency vs belief interpretation of probabilities
- The Dutch Book Theorem:
  If you are willing to bet on your beliefs, then unless they satisfy Bayes rule there will always be a set of bets ("Dutch book") that you would accept which is guaranteed to lose you money, no matter what outcome!



Basic Rules of Probability

Probabilities are non-negative $P(x) \geq 0 \; \forall x$.

Probabilities normalise: $\sum_x P(x) = 1$ for discrete distributions and $\int p(x)dx = 1$ for probability densities.

The joint probability of x and y is: $P(x, y)$.

The marginal probability of x is: $P(x) = \sum_y P(x, y)$.

The conditional probability of x given y is: $P(x|y) = P(x, y)/P(y)$
Bayes Rule:

$$P(x, y) = P(x)P(y|x) = P(y)P(x|y) \quad \Rightarrow \quad \boxed{P(y|x) = \frac{P(x|y)P(y)}{P(x)}}$$

Warning: I will not be obsessively careful in my use of p and P for probability density and probability distribution. Should be obvious from context.

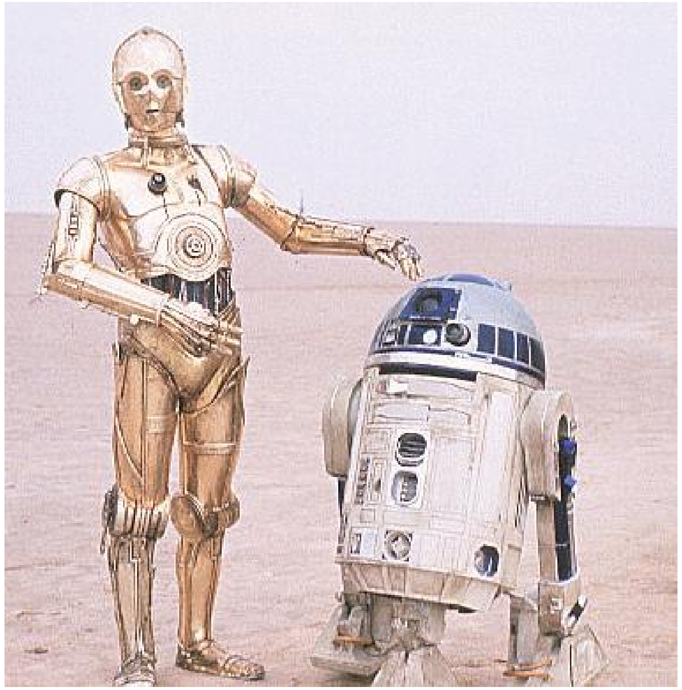
Representing Beliefs (Artificial Intelligence)

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 rules (calculus) 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 more than one way, then every way should lead to the same answer.
  - The robot always takes 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 Bayes rule. (see Jaynes, *Probability Theory: The Logic of Science*)

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:

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

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.

Bayesian Learning

Apply the basic rules of probability to learning from data.
Data set: $\mathcal{D} = \{x_1, \ldots, x_n\}$ Models: m, m' etc. Model parameters: θ

Prior probabilities on models: $P(m)$, $P(m')$ etc.

Prior probabilities on model parameters: e.g. $P(\theta|m)$

Model of data given parameters: $P(x|\theta, m)$

If the data are independently and identically distributed then:

$$P(\mathcal{D}|\theta, m) = \prod_{i=1}^{n} P(x_i|\theta, m)$$

Posterior probability of model parameters:

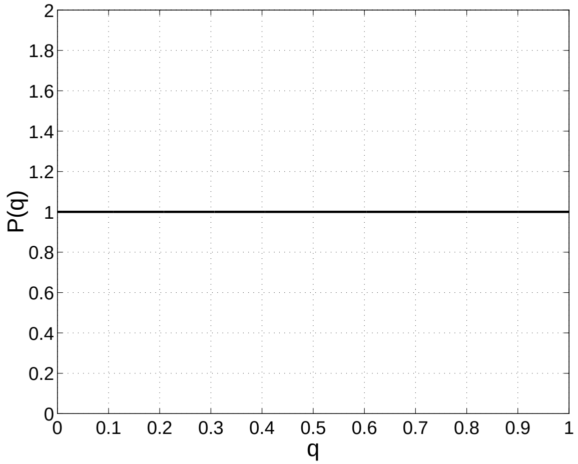
$$P(\theta|\mathcal{D}, m) = \frac{P(\mathcal{D}|\theta, m)P(\theta|m)}{P(\mathcal{D}|m)}$$

Posterior probability of models:

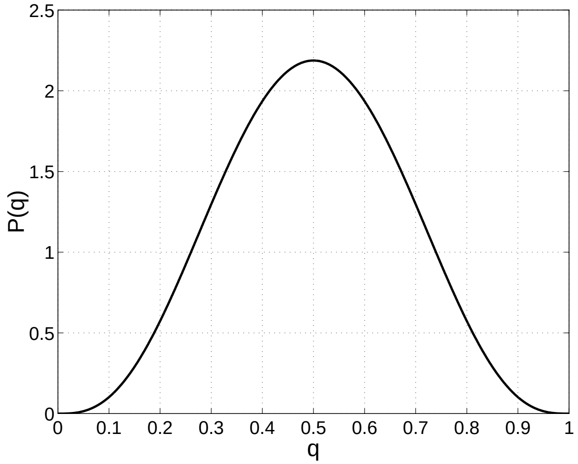
$$P(m|\mathcal{D}) = \frac{P(m)P(\mathcal{D}|m)}{P(\mathcal{D})}$$

Bayesian Learning: A coin toss example

Coin toss: One parameter q — the odds of obtaining *heads*
So our space of models is the set $q \in [0, 1]$.
Learner A believes all values of q are equally plausible;
Learner B believes that it is more plausible that the coin is "fair" $(q \approx 0.5)$ than "biased".



A



B

These priors beliefs can be described by the Beta distribution:

$$p(q|\alpha_1, \alpha_2) = \frac{q^{(\alpha_1 - 1)}(1-q)^{(\alpha_2 - 1)}}{B(\alpha_1, \alpha_2)} = \mathrm{Beta}(q|\alpha_1, \alpha_2)$$

for A: $\alpha_1 = \alpha_2 = 1.0$ and B: $\alpha_1 = \alpha_2 = 4.0$.

Bayesian Learning: The coin toss (cont)

Two possible outcomes:

$$p(\text{heads}|q) = q \qquad p(\text{tails}|q) = 1 - q \tag{1}$$

Imagine we observe a single coin toss and it comes out *heads*
The probability of the observed data (likelihood) is:

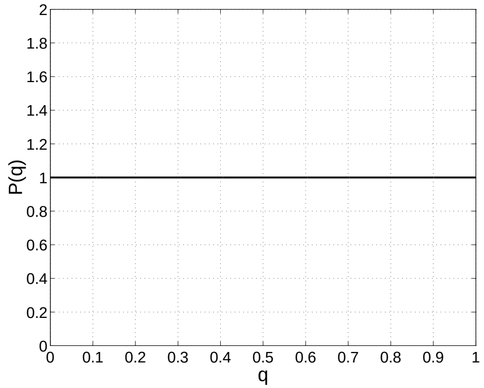
$$p(\text{heads}|q) = q \tag{2}$$

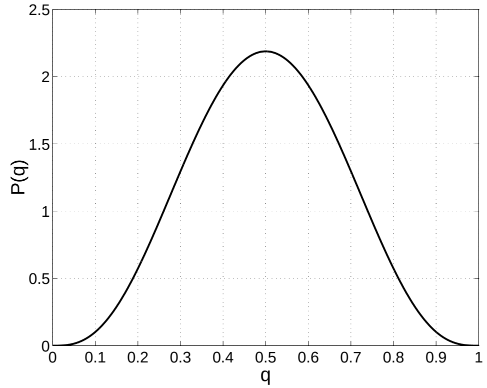
Using Bayes Rule, we multiply the prior, $p(q)$ by the likelihood and renormalise to get the posterior probability:

$$\begin{aligned} p(q|\text{heads}) &= \frac{p(q)p(\text{heads}|q)}{p(\text{heads})} \propto q \, \mathrm{Beta}(q|\alpha_1, \alpha_2) \\ &\propto q \; q^{(\alpha_1 - 1)}(1-q)^{(\alpha_2 - 1)} = \mathrm{Beta}(q|\alpha_1 + 1, \alpha_2) \end{aligned}$$

Bayesian Learning: The coin toss (cont)

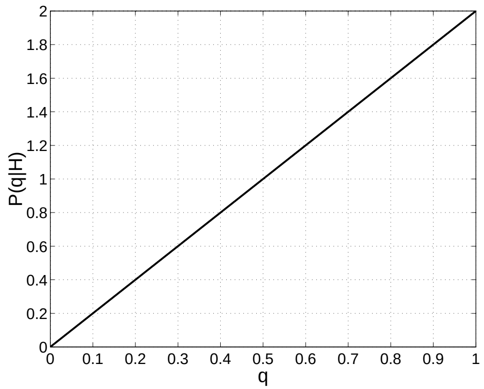
Prior

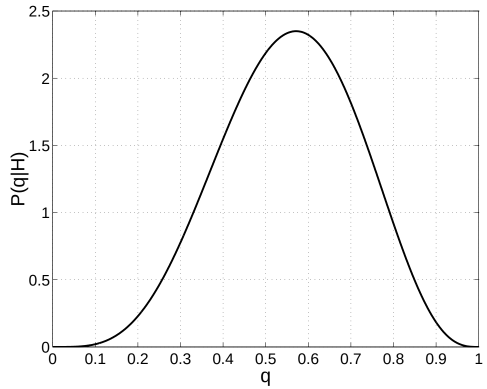




A

B





Posterior

Some Terminology

Maximum Likelihood (ML) Learning: Does not assume a prior over the model parameters. Finds a parameter setting that maximises the likelihood of the data: $P(\mathcal{D}|\theta)$.

Maximum a Posteriori (MAP) Learning: Assumes a prior over the model parameters $P(\theta)$. Finds a parameter setting that maximises the posterior: $P(\theta|\mathcal{D}) \propto P(\theta)P(\mathcal{D}|\theta)$.

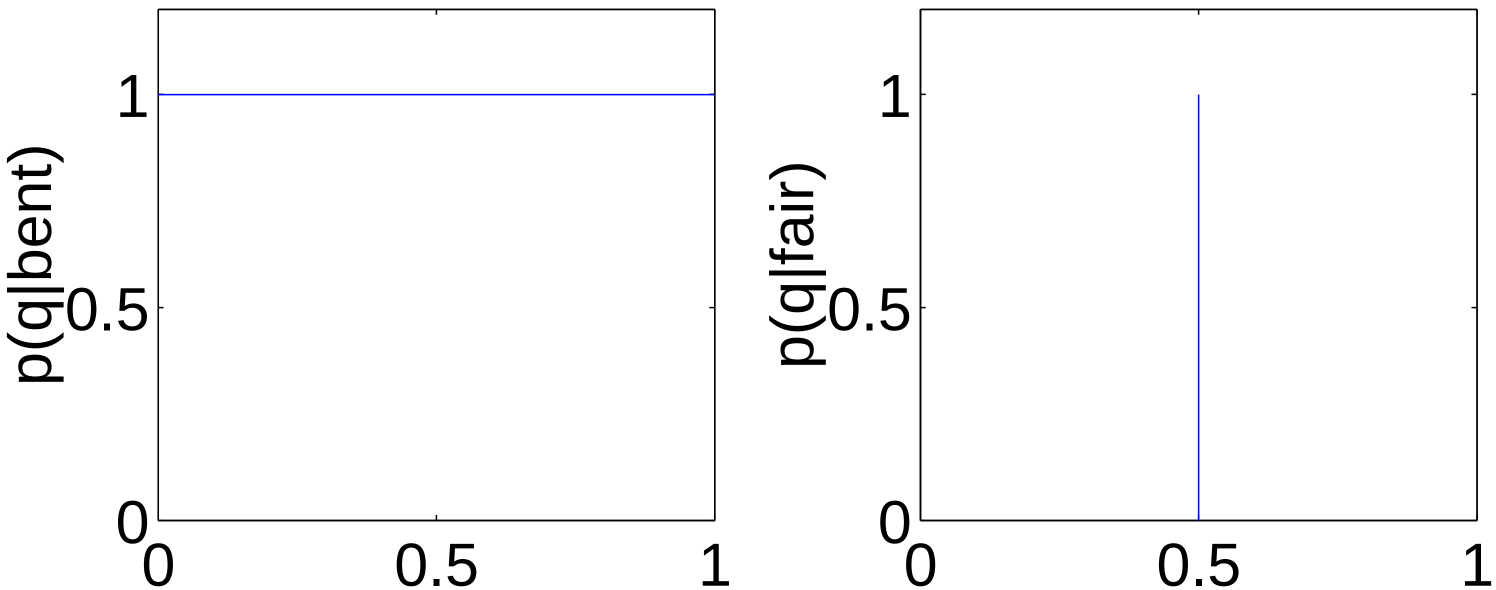
Bayesian Learning: Assumes a prior over the model parameters. Computes the posterior distribution of the parameters: $P(\theta|\mathcal{D})$.

Learning about a coin II

Consider two alternative models of a coin, “fair” and “bent”. A priori, we may think that “fair” is more probable, eg:

$$p(\text{fair}) = 0.8, \qquad p(\text{bent}) = 0.2$$

For the bent coin, (a little unrealistically) all parameter values could be equally likely, where the fair coin has a fixed probability:



We make 10 tosses, and get: T H T H T T T T T T

Learning about a coin. . .

The evidence for the fair model is: $p(\mathcal{D}|\text{fair}) = (1/2)^{10} \simeq 0.001$
and for the bent model:

$$p(\mathcal{D}|\text{bent}) = \int dq\; p(\mathcal{D}|q, \text{bent})p(q|\text{bent}) = \int dq\; q^2(1-q)^8 = \text{B}(3, 9) \simeq 0.002$$

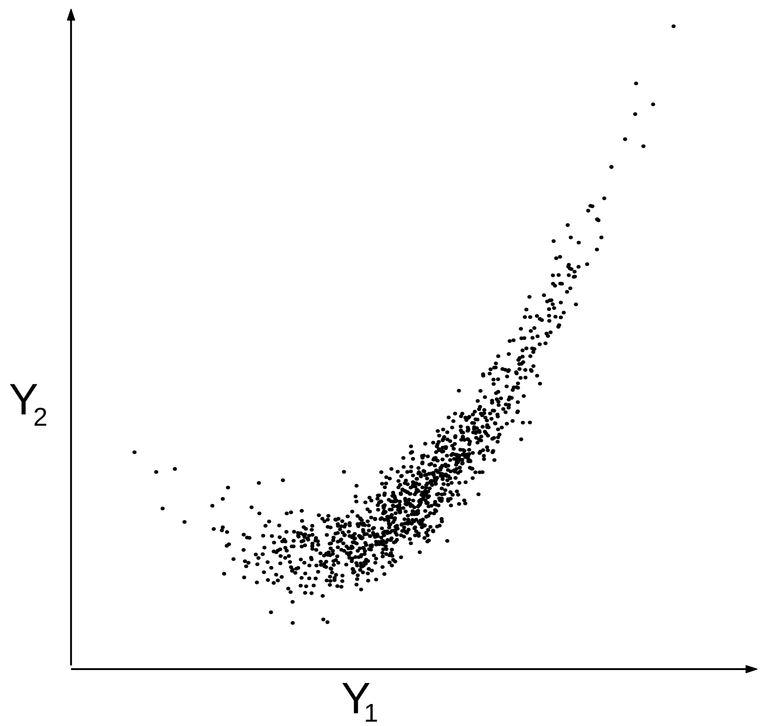
The posterior for the models, by Bayes rule:

$$p(\text{fair}|\mathcal{D}) \propto 0.0008, \qquad p(\text{bent}|\mathcal{D}) \propto 0.0004,$$

ie, two thirds probability that the coin is fair.
How do we make predictions? By weighting the predictions from each model by their probability. Probability of Head at next toss is:

$$\frac{2}{3} \times \frac{1}{2} + \frac{1}{3} \times \frac{3}{12} = \frac{5}{12}.$$

Simple Statistical Modelling: modelling correlations



Assume:

- we have a data set $Y = \{\mathbf{y}_1, \ldots, \mathbf{y}_N\}$
- each data point is a vector of D features: $\mathbf{y}_i = [y_{i1} \ldots y_{iD}]$
- the data points are i.i.d. (independent and identically distributed).

One of the simplest forms of unsupervised learning: model the **mean** of the data and the **correlations** between the D features in the data

We can use a multi-variate Gaussian model:

$$p(\mathbf{y}|\mu, \Sigma) = |2\pi\Sigma|^{-\frac{1}{2}} \exp\left\{-\frac{1}{2}(\mathbf{y}-\mu)^\top \Sigma^{-1} (\mathbf{y}-\mu)\right\}$$

ML Estimation of a Gaussian

Data set $Y = \{\mathbf{y}_1, \ldots, \mathbf{y}_N\}$, likelihood: $p(Y|\mu, \Sigma) = \prod_{n=1}^{N} p(\mathbf{y}_n|\mu, \Sigma)$

Maximize likelihood ⇔ maximize log likelihood
Goal: find μ and Σ that maximise log likelihood:

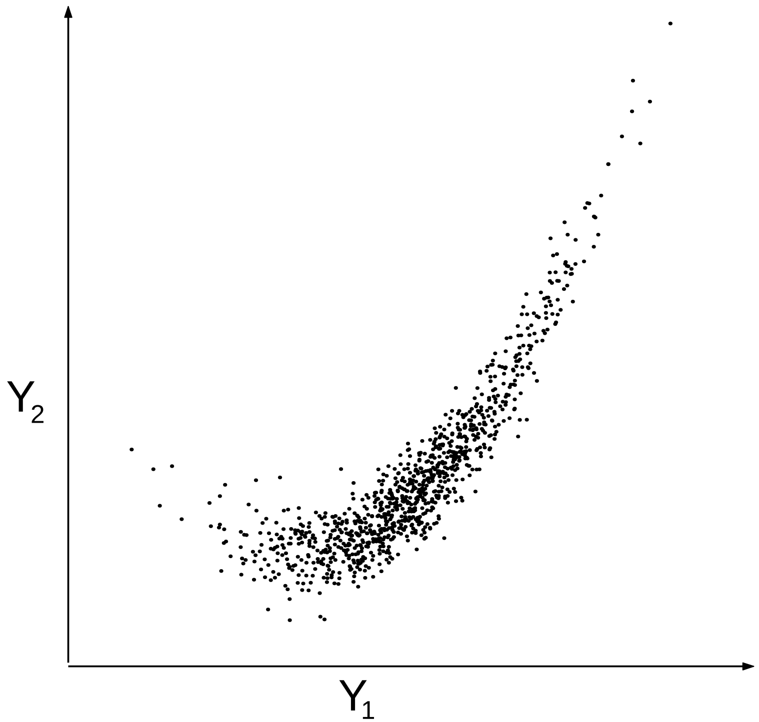
$$\begin{aligned} \mathcal{L} &= \log \prod_{n=1}^{N} p(\mathbf{y}_n|\mu, \Sigma) = \sum_n \log p(\mathbf{y}_n|\mu, \Sigma) \\ &= -\frac{N}{2} \log |2\pi\Sigma| - \frac{1}{2} \sum_n (\mathbf{y}_n - \mu)^\top \Sigma^{-1} (\mathbf{y}_n - \mu) \end{aligned} \tag{3}$$

Note: equivalently, minimise $-\mathcal{L}$, which is *quadratic* in μ
Procedure: take derivatives and set to zero:

$$\frac{\partial \mathcal{L}}{\partial \mu} = 0 \quad \Rightarrow \quad \hat{\mu} = \frac{1}{N} \sum_n \mathbf{y}_n \quad \text{(sample mean)}$$

$$\frac{\partial \mathcal{L}}{\partial \Sigma} = 0 \quad \Rightarrow \quad \hat{\Sigma} = \frac{1}{N} \sum_n (\mathbf{y}_n - \hat{\mu})(\mathbf{y}_n - \hat{\mu})^\top \quad \text{(sample covariance)}$$

Note



modelling correlations

$\Updownarrow$

maximising likelihood of a Gaussian model

$\Updownarrow$

minimising a squared error cost function

$\Updownarrow$

minimizing data coding cost in bits (assuming Gaussian distributed)

Error functions, noise models, and likelihoods

- **Squared error:** $(y-\mu)^2$
  Gaussian noise assumption, y is real-valued

- **Absolute error:** $|y-\mu|$
  Exponential noise assumption, y real or positive

- **Binary cross entropy error:**
  $-y\log p-(1-y)\log(1-p)$
  Binomial noise assumption, y binary

- **Cross entropy error:** $\sum_i y_i \log p_i$
  Multinomial noise assumption, y is discrete (binary unit vector)

Three Limitations

- What about higher order statistical structure in the data? $\Rightarrow$ nonlinear and hierarchical models

- What happens if there are outliers? $\Rightarrow$ other noise models

- There are $D(D+1)/2$ parameters in the multi-variate Gaussian model. What if D is very large?

  $\Rightarrow$ dimensionality reduction

End Notes

For some matrix identities and matrix derivatives see:
`www.cs.toronto.edu/∼roweis/notes/matrixid.pdf`

Also, see Tom Minka's notes on matrix algebra at CMU.
`http://www.stat.cmu.edu/∼minka/papers/matrix.html`