Probabilistic & Unsupervised Learning

Introduction and Foundations

Maneesh Sahani
maneesh@gatsby.ucl.ac.uk

Gatsby Computational Neuroscience Unit, and
MSc ML/CSML, Dept Computer Science
University College London

Term 1, Autumn 2016

What do we mean by learning?

Jan Steen

Not just remembering:

- Systematising (noisy) observations: discovering structure.
- Predicting new outcomes: generalising.
- Choosing actions wisely.

Three learning problems

- Systematising (noisy) observations: discovering structure.
  - **Unsupervised learning.** Observe (sensory) input alone:

    $$x_1, x_2, x_3, x_4, \dots$$

    Describe pattern of data [$p(x)$], identify and extract underlying structural variables [$x_i \to y_i$].
- Predicting new outcomes: generalising.
  - **Supervised learning.** Observe input/output pairs ("teaching"):

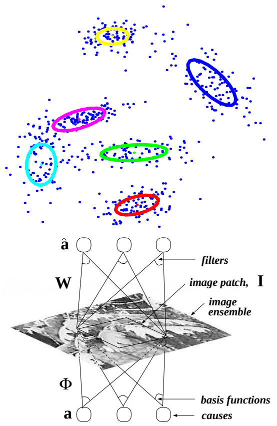
    $$(x_1, y_1), (x_2, y_2), (x_3, y_3), (x_4, y_4), \dots$$

    Predict the correct y^* for new test input x^*.
- Choosing actions wisely.
  - **Reinforcement learning.** Rewards or payoffs (and possibly also inputs) depend on actions:

    $$x_1 : a_1 \to r_1, x_2 : a_2 \to r_2, x_3 : a_3 \to r_3 \dots$$

    Find a policy for action choice that maximises payoff.

Unsupervised Learning



Find underlying structure:

- separate generating processes (clusters)
- reduced dimensionality representations
- good explanations (causes) of the data
- modelling the data density

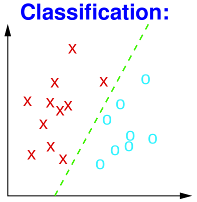
Uses of Unsupervised Learning:

- structure discovery, science
- data compression
- outlier detection
- input to supervised/reinforcement algorithms (causes may be more simply related to outputs or rewards)
- a theory of biological learning and perception

Supervised learning

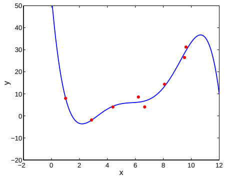
Two main examples:

Classification:



Discrete (class label) outputs.

Regression:



Continuous-values outputs.

But also: ranks, relationships, trees etc.

Variants may relate to unsupervised learning:

- semi-supervised learning (most x unlabelled; assumes structure of $\{x\}$ and relationship $x \to y$ are linked).
- multitask (transfer) learning (predict different y in different contexts; assumes links between structure of relationships).

A probabilistic approach

Data are generated by random and/or unknown processes.

Our approach to learning starts with a probabilistic model of data production:

$$P(\text{data}|\text{parameters}) \qquad P(x|\theta) \text{ or } P(y|x,\theta)$$

This is the generative model or likelihood.

- The probabilistic model can be used to
  - make inferences about missing inputs
  - generate predictions/fantasies/imagery
  - make predictions or decisions which minimise expected loss
  - communicate the data in an efficient way
- Probabilistic modelling is often 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
  - structural risk: compensate for overconfidence in powerful models

The calculus of probabilities naturally handles randomness. It is also the right way to reason about unknown values.

Representing beliefs

Let $b(x)$ represent our strength of belief in (plausibility of) proposition x:

$0 \le b(x) \le 1$		
$b(x) = 0$	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):

- Let $b(x)$ be real. As $b(x)$ increases, $b(\neg x)$ decreases, and so the function mapping $b(x) \leftrightarrow b(\neg x)$ is monotonically decreasing and self-inverse.
- $b(x \wedge y)$ depends only on $b(y)$ and $b(x|y)$.
- Consistency
  - If a conclusion can be reasoned in more than one way, then every way should lead to the same answer.
  - Beliefs 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 be isomorphic to probabilities, satisfying all the usual laws, including Bayes rule. (See Jaynes, *Probability Theory: The Logic of Science*)

Basic rules of probability

- Probabilities are non-negative $P(x) \ge 0 \; \forall x$.
- Probabilities normalise: $\sum_{x \in \mathcal{X}} P(x) = 1$ for distributions if x is a discrete variable and $\int_{-\infty}^{+\infty} p(x)dx = 1$ for probability densities over continuous variables
- The joint probability of x and y is: $P(x,y)$.
- The marginal probability of x is: $P(x) = \sum_y P(x,y)$, assuming y is discrete.
- The conditional probability of x given y is: $P(x|y) = P(x,y)/P(y)$
- Bayes Rule:

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

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.

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:

$$x \text{ at } 1:9 \Rightarrow \left\{ \begin{array}{llll} x & \text{is true} & \text{win} & \geq £1 \\ x & \text{is false} & \text{lose} & £9 \end{array} \right.$$

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**.
E.g. suppose $A \cap B = \varnothing$, then

$$\left\{ \begin{array}{rl} b(A) & = 0.3 \\ b(B) & = 0.2 \\ b(A \cup B) & = 0.6 \end{array} \right\} \Rightarrow \text{accept the bets} \left\{ \begin{array}{rl} \neg A & \text{at } 3:7 \\ \neg B & \text{at } 2:8 \\ A \cup B & \text{at } 4:6 \end{array} \right\}$$

But then:

$$\begin{array}{rll} \neg A \cap B & \Rightarrow & \text{win } +3-8+4 = -1 \\ A \cap \neg B & \Rightarrow & \text{win } -7+2+4 = -1 \\ \neg A \cap \neg B & \Rightarrow & \text{win } +3+2-6 = -1 \end{array}$$

The only way to guard against Dutch Books is 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.

- Problem specification:

  Data: $\mathcal{D} = \{x_1, \dots, x_n\}$ Models: $\mathcal{M}_1$, $\mathcal{M}_2$, etc. Parameters: θ_i (per model)

  Prior probability of models: $P(\mathcal{M}_i)$.

  Prior probabilities of model parameters: $P(\theta_i|\mathcal{M}_i)$

  Model of data given parameters (likelihood model): $P(x|\theta_i, \mathcal{M}_i)$

- Data probability (likelihood)

  $$P(\mathcal{D}|\theta_i, \mathcal{M}_i) = \prod_{j=1}^{n} P(x_j|\theta_i, \mathcal{M}_i) \equiv \mathcal{L}(\theta_i)$$

  (provided the data are independently and identically distributed (iid).

- Parameter learning (posterior):

  $$P(\theta_i|\mathcal{D}, \mathcal{M}_i) = \frac{P(\mathcal{D}|\theta_i, \mathcal{M}_i)P(\theta_i|\mathcal{M}_i)}{P(\mathcal{D}|\mathcal{M}_i)}; \quad P(\mathcal{D}|\mathcal{M}_i) = \int d\theta_i\, P(\mathcal{D}|\theta_i, \mathcal{M}_i)P(\theta_i|\mathcal{M}_i)$$

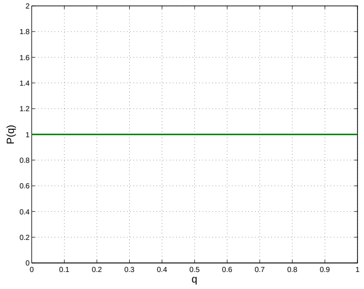
  $P(\mathcal{D}|\mathcal{M}_i)$ is called the marginal likelihood or evidence for $\mathcal{M}_i$. It is proportional to the posterior probability model $\mathcal{M}_i$ being the one that generated the data.

- Model selection:

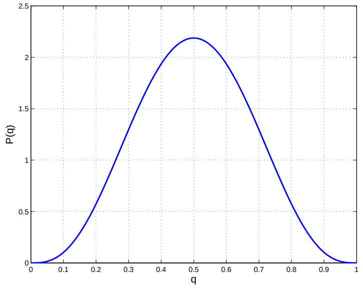
  $$P(\mathcal{M}_i|\mathcal{D}) = \frac{P(\mathcal{D}|\mathcal{M}_i)P(\mathcal{M}_i)}{P(\mathcal{D})}$$

Bayesian learning: A coin toss example

Coin toss: One parameter q — the probability of obtaining *heads*
So our space of models is the set of distributions over $q \in [0, 1]$.
Learner A believes model $\mathcal{M}_\text{A}$: all values of q are equally plausible;
Learner B believes model $\mathcal{M}_\text{B}$: more plausible that the coin is “fair” ($q \approx 0.5$) than “biased”.



A: $\alpha_1 = \alpha_2 = 1.0$



B: $\alpha_1 = \alpha_2 = 4.0$

Both prior beliefs can be described by the Beta distribution:

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

Bayesian learning: The coin toss (cont)

Now we observe a toss. Two possible outcomes:

$$p(\text{H}|q) = q \qquad p(\text{T}|q) = 1 - q$$

Suppose our single coin toss comes out *heads*

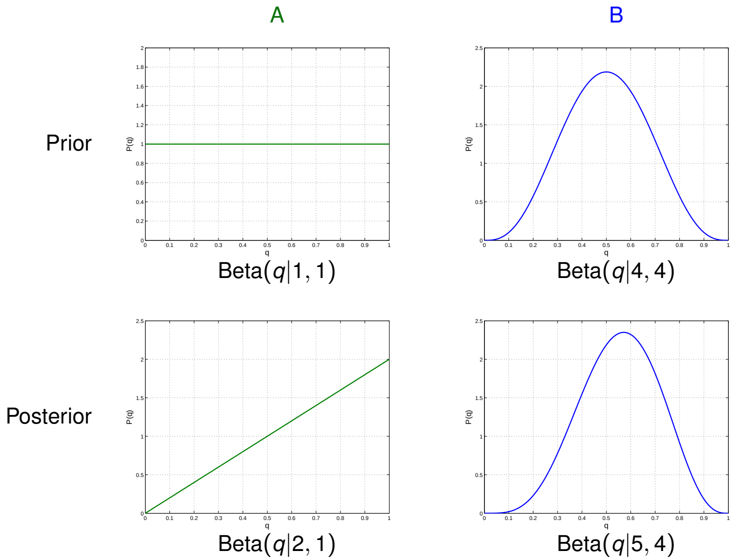
The probability of the observed data (likelihood) is:

$$p(\text{H}|q) = q$$

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

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

Bayesian learning: The coin toss (cont)



$\mathrm{Beta}(q|1,1)$

$\mathrm{Beta}(q|4,4)$

$\mathrm{Beta}(q|2,1)$

$\mathrm{Beta}(q|5,4)$

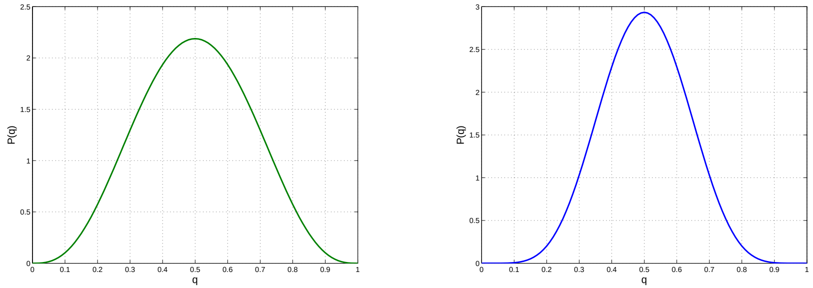
Bayesian learning: The coin toss (cont)

What about multiple tosses? Suppose we observe $\mathcal{D} = \{$ H H T H T T $\}$:

$$p(\{\text{ H H T H T T }\}|q) = qq(1-q)q(1-q)(1-q) = q^3(1-q)^3$$

This is still straightforward:

$$\begin{aligned} p(q|\mathcal{D}) &= \frac{p(q)p(\mathcal{D}|q)}{p(\mathcal{D})} \propto q^3(1-q)^3\ \mathrm{Beta}(q|\alpha_1,\alpha_2) \\ &\propto \mathrm{Beta}(q|\alpha_1+3,\alpha_2+3) \end{aligned}$$



Conjugate priors

Updating the prior to form the posterior was particularly easy in these examples. This is because we used a conjugate prior for an exponential family likelihood.
Exponential family distributions take the form:

$$P(x|\theta) = g(\theta)f(x)e^{\phi(\theta)^{\mathsf{T}}\mathbf{T}(x)}$$

with $g(\theta)$ the normalising constant. Given n iid observations,

$$P(\{x_i\}|\theta) = \prod_i P(x_i|\theta) = g(\theta)^n e^{\phi(\theta)^{\mathsf{T}}\left(\sum_i \mathbf{T}(x_i)\right)} \prod_i f(x_i)$$

Thus, if the prior takes the conjugate form

$$P(\theta) = F(\boldsymbol{\tau},\nu)g(\theta)^{\nu}e^{\phi(\theta)^{\mathsf{T}}\boldsymbol{\tau}}$$

with $F(\boldsymbol{\tau},\nu)$ the normaliser, then the posterior is

$$P(\theta|\{x_i\}) \propto P(\{x_i\}|\theta)P(\theta) \propto g(\theta)^{\nu+n}e^{\phi(\theta)^{\mathsf{T}}\left(\boldsymbol{\tau}+\sum_i \mathbf{T}(x_i)\right)}$$

with the normaliser given by $F\left(\boldsymbol{\tau}+\sum_i \mathbf{T}(x_i),\nu+n\right)$.

Conjugate priors

The posterior given an exponential family likelihood and conjugate prior is:

$$P(\theta|\{x_i\}) = F\left(\boldsymbol{\tau}+\textstyle\sum_i \mathbf{T}(x_i),\nu+n\right)g(\theta)^{\nu+n}\exp\left[\phi(\theta)^{\mathsf{T}}\left(\boldsymbol{\tau}+\textstyle\sum_i \mathbf{T}(x_i)\right)\right]$$

Here,

$\phi(\theta)$ is the vector of natural parameters

$\sum_i \mathbf{T}(x_i)$ is the vector of sufficient statistics

$\boldsymbol{\tau}$ are pseudo-observations which define the prior

ν is the scale of the prior (need not be an integer)

As new data come in, each one increments the sufficient statistics vector and the scale to define the posterior.

The prior appears to be based on “pseudo-observations”, but:

1. This is different to applying Bayes’ rule. No prior! Sometimes we can take a uniform prior (say on $[0,1]$ for q), but for unbounded θ, there may be no equivalent.
2. A valid conjugate prior might have non-integral ν or impossible $\boldsymbol{\tau}$, with no likelihood equivalent.

Conjugacy in the coin flip

Distributions are not always written in their natural exponential form.

The Bernoulli distribution (a single coin flip) with parameter q and observation $x \in \{0, 1\}$, can be written:

$$\begin{aligned} P(x|q) &= q^x (1-q)^{(1-x)} \\ &= e^{x \log q + (1-x) \log(1-q)} \\ &= e^{\log(1-q) + x \log(q/(1-q))} \\ &= (1-q) e^{\log(q/(1-q)) x} \end{aligned}$$

So the natural parameter is the log odds $\log(q/(1-q))$, and the sufficient stats (for multiple tosses) is the number of heads.

The conjugate prior is

$$\begin{aligned} P(q) &= F(\tau, \nu)\ (1-q)^{\nu} e^{\log(q/(1-q))\tau} \\ &= F(\tau, \nu)\ (1-q)^{\nu} e^{\tau \log q - \tau \log(1-q)} \\ &= F(\tau, \nu)\ (1-q)^{\nu - \tau} q^{\tau} \end{aligned}$$

which has the form of the Beta distribution $\Rightarrow F(\tau, \nu) = 1/B(\tau + 1, \nu - \tau + 1)$.

In general, then, the posterior will be $P(q|\{x_i\}) = \text{Beta}(\alpha_1, \alpha_2)$, with

$$\alpha_1 = 1 + \tau + \textstyle\sum_i x_i \qquad\qquad \alpha_2 = 1 + (\nu + n) - \left(\tau + \textstyle\sum_i x_i\right)$$

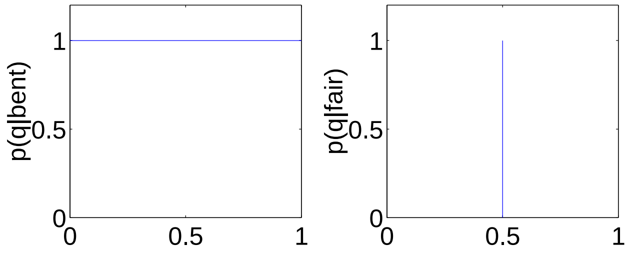
If we observe a head, we add 1 to the sufficient statistic $\sum x_i$, and also 1 to the count n. This increments α_1. If we observe a tail we add 1 to n, but not to $\sum x_i$, incrementing α_2.

Bayesian coins – comparing models

We have seen how to update posteriors within each model. To study the choice of model, consider two more extreme models: "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, we assume all parameter values are equally likely, whilst the fair coin has a fixed probability:



We make 10 tosses, and get: $\mathcal{D} = (\text{T H T H T T T T T T})$.

Bayesian coins – comparing models

Which model should we prefer *a posteriori* (i.e. after seeing the data)?

The evidence for the fair model is:

$$P(\mathcal{D}|\text{fair}) \ = (1/2)^{10} \approx 0.001$$

and for the bent model is:

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

Thus, 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, a two-thirds probability that the coin is fair.

How do we make predictions? Could choose the fair model (model selection).

Or could weight the predictions from each model by their probability (model averaging).

Probability of H at next toss is:

$$P(\text{H}|\mathcal{D}) = P(\text{H}|\mathcal{D}, \text{fair}) P(\text{fair}|\mathcal{D}) + P(\text{H}|\mathcal{D}, \text{bent}) P(\text{bent}|\mathcal{D}) = \frac{2}{3} \times \frac{1}{2} + \frac{1}{3} \times \frac{3}{12} = \frac{5}{12}.$$

Learning parameters

The Bayesian probabilistic prescription tells us how to reason about models and their parameters. But it is often impractical for realistic models (outside the exponential family).

- **Point estimates** of parameters or other predictions
  - Compute posterior and find single parameter that minimises expected loss.

    $$\theta^{\text{BP}} = \underset{\hat{\theta}}{\text{argmin}} \left\langle L(\hat{\theta}, \theta) \right\rangle_{P(\theta|\mathcal{D})}$$

    - $\langle \theta \rangle_{P(\theta|\mathcal{D})}$ minimises squared loss.
  - **Maximum a Posteriori (MAP) estimate**: Assume a prior over the model parameters $P(\theta)$, and compute parameters that are most probable under the posterior:

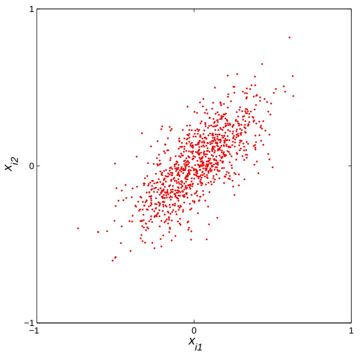
    $$\theta^{\text{MAP}} = \text{argmax}\ P(\theta|\mathcal{D}) = \text{argmax}\ P(\theta) P(\mathcal{D}|\theta)\ .$$

    - Equivalent to minimising the 0/1 loss.
  - **Maximum Likelihood (ML) Learning**: No prior over the parameters. Compute parameter value that maximises the likelihood function alone:

    $$\theta^{\text{ML}} = \text{argmax}\ P(\mathcal{D}|\theta)\ .$$

    - Parameterisation-independent.
- **Approximations** may allow us to recover samples from posterior, or to find a distribution which is close in some sense.
- Choosing between these and other alternatives may be a matter of definition, of goals (loss function), or of practicality.
- For the next few weeks we will look at ML and MAP learning in more complex models. We will then return to the fully Bayesian formulation for the few intersting cases where it is tractable. Approximations will be addressed in the second half of the course.

Modelling associations between variables



- Data set $\mathcal{D} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$
- with each data point a vector of D features: $\mathbf{x}_i = [x_{i1} \ldots x_{iD}]$
- Assume data are i.i.d. (independent and identically distributed).

A simple forms of unsupervised (structure) learning: model the **mean** of the data and the **correlations** between the D features in the data.

We can use a multivariate Gaussian model:

$$p(\mathbf{x}|\boldsymbol{\mu}, \Sigma) = \mathcal{N}(\boldsymbol{\mu}, \Sigma) = |2\pi\Sigma|^{-\frac{1}{2}} \exp\left\{-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu})^{\mathsf{T}}\Sigma^{-1}(\mathbf{x} - \boldsymbol{\mu})\right\}$$

ML Learning for a Gaussian

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

Goal: find $\boldsymbol{\mu}$ and Σ that maximise likelihood ⇔ maximise log likelihood:

$$\mathcal{L}\ell = \prod_{n=1}^{N} p(\mathbf{x}_n|\boldsymbol{\mu}, \Sigma) = \log \prod_{n=1}^{N} p(\mathbf{x}_n|\boldsymbol{\mu}, \Sigma) \ = \sum_n \log p(\mathbf{x}_n|\boldsymbol{\mu}, \Sigma)$$

$$= -\frac{N}{2} \log |2\pi\Sigma| - \frac{1}{2} \sum_n (\mathbf{x}_n - \boldsymbol{\mu})^{\mathsf{T}}\Sigma^{-1}(\mathbf{x}_n - \boldsymbol{\mu})$$

Note: equivalently, minimise $-\ell$, which is *quadratic* in $\boldsymbol{\mu}$

Procedure: take derivatives and set to zero:

$$\frac{\partial \ell}{\partial \boldsymbol{\mu}} = 0 \quad \Rightarrow \quad \hat{\boldsymbol{\mu}} = \frac{1}{N}\sum_n \mathbf{x}_n \quad \text{(sample mean)}$$

$$\frac{\partial \ell}{\partial \Sigma} = 0 \quad \Rightarrow \quad \hat{\Sigma} = \frac{1}{N}\sum_n (\mathbf{x}_n - \hat{\boldsymbol{\mu}})(\mathbf{x}_n - \hat{\boldsymbol{\mu}})^{\mathsf{T}} \quad \text{(sample covariance)}$$

Refresher – matrix derivatives of scalar forms

We will use the following facts:

$$\mathbf{x}^{\mathsf{T}}A\mathbf{y} = \mathbf{y}^{\mathsf{T}}A^{\mathsf{T}}\mathbf{x} = \mathrm{Tr}\left[\mathbf{x}^{\mathsf{T}}A\mathbf{y}\right] \quad \text{(scalars equal their own transpose and trace)}$$

$$\mathrm{Tr}\,[A] = \mathrm{Tr}\left[A^{\mathsf{T}}\right] \qquad \mathrm{Tr}\,[ABC] = \mathrm{Tr}\,[CAB] = \mathrm{Tr}\,[BCA]$$

$$\frac{\partial}{\partial A_{ij}}\mathrm{Tr}\left[A^{\mathsf{T}}B\right] = \frac{\partial}{\partial A_{ij}}\sum_n [A^{\mathsf{T}}B]_{nn} = \frac{\partial}{\partial A_{ij}}\sum_n\sum_m A^{\mathsf{T}}_{nm}B_{mn} = \frac{\partial}{\partial A_{ij}}\sum_{mn} A_{mn}B_{mn} = B_{ij}$$

$$\Rightarrow \frac{\partial}{\partial A}\mathrm{Tr}\left[A^{\mathsf{T}}B\right] = B$$

$$\frac{\partial}{\partial A}\mathrm{Tr}\left[A^{\mathsf{T}}BAC\right] = \frac{\partial}{\partial A}\mathrm{Tr}\left[F_1(A)^{\mathsf{T}}BF_2(A)C\right] \text{ with } F_1 \text{ and } F_2 \text{ both identity maps}$$

$$= \frac{\partial}{\partial F_1}\mathrm{Tr}\left[F_1^{\mathsf{T}}BF_2C\right]\frac{\partial F_1}{\partial A} + \frac{\partial}{\partial F_2}\mathrm{Tr}\left[F_1^{\mathsf{T}}BF_2C\right]\frac{\partial F_2}{\partial A} = \frac{\partial}{\partial F_1}\mathrm{Tr}\left[F_1^{\mathsf{T}}BF_2C\right]\frac{\partial F_1}{\partial A} + \frac{\partial}{\partial F_2}\mathrm{Tr}\left[CF\right.$$

$$= BF_2C + B^{\mathsf{T}}F_1C^{\mathsf{T}} = BAC + B^{\mathsf{T}}AC^{\mathsf{T}}$$

$$\frac{\partial}{\partial A_{ij}}\log|A| = \frac{1}{|A|}\frac{\partial}{\partial A_{ij}}|A| = \frac{1}{|A|}\frac{\partial}{\partial A_{ij}}\sum_k(-1)^{i+k}A_{ik}\,|[A]_{ik}| = \frac{1}{|A|}(-1)^{i+j}\,|[A]_{ij}|$$

$$\Rightarrow \frac{\partial}{\partial A}\log|A| = (A^{-1})^{\mathsf{T}}$$

Gaussian Derivatives

$$\frac{\partial(-\ell)}{\partial \boldsymbol{\mu}} = \frac{\partial}{\partial \boldsymbol{\mu}}\left[\frac{N}{2}\log|2\pi\Sigma| + \frac{1}{2}\sum_n(\mathbf{x}_n - \boldsymbol{\mu})^{\mathsf{T}}\Sigma^{-1}(\mathbf{x}_n - \boldsymbol{\mu})\right]$$

$$= \frac{1}{2}\sum_n \frac{\partial}{\partial \boldsymbol{\mu}}\left[(\mathbf{x}_n - \boldsymbol{\mu})^{\mathsf{T}}\Sigma^{-1}(\mathbf{x}_n - \boldsymbol{\mu})\right]$$

$$= \frac{1}{2}\sum_n \frac{\partial}{\partial \boldsymbol{\mu}}\left[\mathbf{x}_n^{\mathsf{T}}\Sigma^{-1}\mathbf{x}_n + \boldsymbol{\mu}^{\mathsf{T}}\Sigma^{-1}\boldsymbol{\mu} - 2\boldsymbol{\mu}^{\mathsf{T}}\Sigma^{-1}\mathbf{x}_n\right]$$

$$= \frac{1}{2}\sum_n \frac{\partial}{\partial \boldsymbol{\mu}}\left[\boldsymbol{\mu}^{\mathsf{T}}\Sigma^{-1}\boldsymbol{\mu}\right] - 2\frac{\partial}{\partial \boldsymbol{\mu}}\left[\boldsymbol{\mu}^{\mathsf{T}}\Sigma^{-1}\mathbf{x}_n\right]$$

$$= \frac{1}{2}\sum_n\left[2\Sigma^{-1}\boldsymbol{\mu} - 2\Sigma^{-1}\mathbf{x}_n\right]$$

$$= N\Sigma^{-1}\boldsymbol{\mu} - \Sigma^{-1}\sum_n \mathbf{x}_n$$

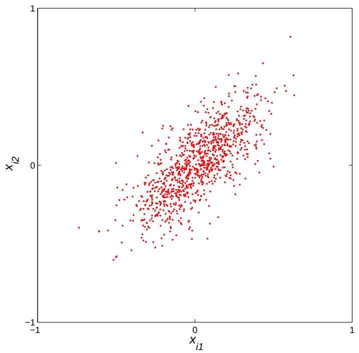
$$= 0 \Rightarrow \hat{\boldsymbol{\mu}} = \frac{1}{N}\sum_n \mathbf{x}_n$$

Gaussian Derivatives

$$\begin{aligned}\frac{\partial(-\ell)}{\partial\Sigma^{-1}} &= \frac{\partial}{\partial\Sigma^{-1}}\left[\frac{N}{2}\log|2\pi\Sigma| + \frac{1}{2}\sum_n(\mathbf{x}_n-\boldsymbol{\mu})^\mathsf{T}\Sigma^{-1}(\mathbf{x}_n-\boldsymbol{\mu})\right]\\ &= \frac{\partial}{\partial\Sigma^{-1}}\left[\frac{N}{2}\log|2\pi I|\right] - \frac{\partial}{\partial\Sigma^{-1}}\left[\frac{N}{2}\log|\Sigma^{-1}|\right]\\ &\quad + \frac{1}{2}\sum_n\frac{\partial}{\partial\Sigma^{-1}}\left[(\mathbf{x}_n-\boldsymbol{\mu})^\mathsf{T}\Sigma^{-1}(\mathbf{x}_n-\boldsymbol{\mu})\right]\\ &= -\frac{N}{2}\Sigma^\mathsf{T} + \frac{1}{2}\sum_n(\mathbf{x}_n-\boldsymbol{\mu})(\mathbf{x}_n-\boldsymbol{\mu})^\mathsf{T}\end{aligned}$$

$$= 0 \Rightarrow \hat{\Sigma} = \frac{1}{N}\sum_n(\mathbf{x}_n-\boldsymbol{\mu})(\mathbf{x}_n-\boldsymbol{\mu})^\mathsf{T}$$

Equivalences



modelling correlations

⇕

maximising likelihood of a Gaussian model

⇕

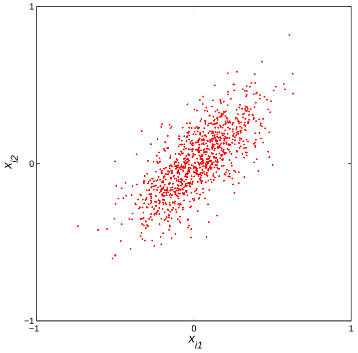
minimising a squared error cost function

⇕

minimizing data coding cost in bits (assuming Gaussian distributed)

Multivariate Linear Regression

The relationship between variables can also be modelled as a conditional distribution.



- data $\mathcal{D} = \{(\mathbf{x}_1, \mathbf{y}_1)\ldots,(\mathbf{x}_N, \mathbf{y}_N)\}$
- each $\mathbf{x}_i$ ($\mathbf{y}_i$) is a vector of D_x (D_y) features,
- $\mathbf{y}_i$ is conditionally independent of all else, given $\mathbf{x}_i$.

A simple form of supervised (predictive) learning: model **y** as a **linear** function of **x**, with **Gaussian** noise.

$$p(\mathbf{y}|\mathbf{x}, \mathrm{W}, \Sigma_y) = |2\pi\Sigma_y|^{-\frac{1}{2}}\exp\left\{-\frac{1}{2}(\mathbf{y}-\mathrm{W}\mathbf{x})^\mathsf{T}\Sigma_y^{-1}(\mathbf{y}-\mathrm{W}\mathbf{x})\right\}$$

Multivariate Linear Regression – ML estimate

ML estimates are obtained by maximising the (conditional) likelihood, as before:

$$\begin{aligned}\ell &= \sum_i \log p(\mathbf{y}_i|\mathbf{x}_i, \mathrm{W}, \Sigma_y)\\ &= -\frac{N}{2}\log|2\pi\Sigma_y| - \frac{1}{2}\sum_i(\mathbf{y}_i-\mathrm{W}\mathbf{x}_i)^\mathsf{T}\Sigma_y^{-1}(\mathbf{y}_i-\mathrm{W}\mathbf{x}_i)\\ \frac{\partial(-\ell)}{\partial\mathrm{W}} &= \frac{\partial}{\partial\mathrm{W}}\left[\frac{N}{2}\log|2\pi\Sigma_y| + \frac{1}{2}\sum_i(\mathbf{y}_i-\mathrm{W}\mathbf{x}_i)^\mathsf{T}\Sigma_y^{-1}(\mathbf{y}_i-\mathrm{W}\mathbf{x}_i)\right]\\ &= \frac{1}{2}\sum_i\frac{\partial}{\partial\mathrm{W}}\left[(\mathbf{y}_i-\mathrm{W}\mathbf{x}_i)^\mathsf{T}\Sigma_y^{-1}(\mathbf{y}_i-\mathrm{W}\mathbf{x}_i)\right]\\ &= \frac{1}{2}\sum_i\frac{\partial}{\partial\mathrm{W}}\left[\mathbf{y}_i^\mathsf{T}\Sigma_y^{-1}\mathbf{y}_i + \mathbf{x}_i^\mathsf{T}\mathrm{W}^\mathsf{T}\Sigma_y^{-1}\mathrm{W}\mathbf{x}_i - 2\mathbf{x}_i^\mathsf{T}\mathrm{W}^\mathsf{T}\Sigma_y^{-1}\mathbf{y}_i\right]\\ &= \frac{1}{2}\sum_i\left[\frac{\partial}{\partial\mathrm{W}}\mathrm{Tr}\left[\mathrm{W}^\mathsf{T}\Sigma_y^{-1}\mathrm{W}\mathbf{x}_i\mathbf{x}_i^\mathsf{T}\right] - 2\frac{\partial}{\partial\mathrm{W}}\mathrm{Tr}\left[\mathrm{W}^\mathsf{T}\Sigma_y^{-1}\mathbf{y}_i\mathbf{x}_i^\mathsf{T}\right]\right]\\ &= \frac{1}{2}\sum_i\left[2\Sigma_y^{-1}\mathrm{W}\mathbf{x}_i\mathbf{x}_i^\mathsf{T} - 2\Sigma_y^{-1}\mathbf{y}_i\mathbf{x}_i^\mathsf{T}\right]\end{aligned}$$

$$= 0 \Rightarrow \widehat{\mathrm{W}} = \sum_i\mathbf{y}_i\mathbf{x}_i^\mathsf{T}\left(\sum_i\mathbf{x}_i\mathbf{x}_i^\mathsf{T}\right)^{-1}$$

Multivariate Linear Regression – Posterior

Let y_i be scalar (so that W is a row vector) and write $\mathbf{w}$ for the column vector of weights.

A conjugate prior for $\mathbf{w}$ is

$$P(\mathbf{w}|\mathsf{A}) = \mathcal{N}\left(\mathbf{0}, \mathsf{A}^{-1}\right)$$

Then the log posterior on $\mathbf{w}$ is

$$\begin{aligned}
\log P(\mathbf{w}|\mathcal{D}, \mathsf{A}, \sigma_y) &= \log P(\mathcal{D}|\mathbf{w}, \mathsf{A}, \sigma_y) + \log P(\mathbf{w}|\mathsf{A}, \sigma_y) - \log P(\mathcal{D}|\mathsf{A}, \sigma_y) \\
&= -\frac{1}{2}\mathbf{w}^{\mathsf{T}}\mathsf{A}\mathbf{w} - \frac{1}{2}\sum_i (y_i - \mathbf{w}^{\mathsf{T}}\mathbf{x}_i)^2 \sigma_y^{-2} + \text{const} \\
&= -\frac{1}{2}\mathbf{w}^{\mathsf{T}}\left(\mathsf{A} + \sigma_y^{-2}\sum_i \mathbf{x}_i\mathbf{x}_i^{\mathsf{T}}\right)\mathbf{w} + \mathbf{w}^{\mathsf{T}}\sum_i (y_i\mathbf{x}_i)\sigma_y^{-2} + \text{const} \\
&= -\frac{1}{2}\mathbf{w}^{\mathsf{T}}\Sigma_w^{-1}\mathbf{w} + \mathbf{w}^{\mathsf{T}}\Sigma_w^{-1}\,\Sigma_w \sum_i (y_i\mathbf{x}_i)\sigma_y^{-2} + \text{const} \\
&= \log\,\mathcal{N}\left(\Sigma_w \textstyle\sum_i (y_i\mathbf{x}_i)\sigma_y^{-2}, \Sigma_w\right)
\end{aligned}$$

MAP and ML for linear regression

As the posterior is Gaussian, the MAP and posterior mean weights are the same:

$$\mathbf{w}^{\text{MAP}} = \underbrace{\left(A + \frac{\sum_i \mathbf{x}_i\mathbf{x}_i^{\mathsf{T}}}{\sigma_y^2}\right)^{-1}}_{\Sigma_w} \frac{\sum_i y_i\mathbf{x}_i}{\sigma_y^2} = \left(A\sigma_y^2 + \sum_i \mathbf{x}_i\mathbf{x}_i^{\mathsf{T}}\right)^{-1} \sum_i y_i\mathbf{x}_i$$

Compare this to the (transposed) ML weight vector for scalar outputs:

$$\mathbf{w}^{\text{ML}} = \widehat{\mathsf{W}}^{\mathsf{T}} = \left(\sum_i \mathbf{x}_i\mathbf{x}_i^{\mathsf{T}}\right)^{-1} \sum_i y_i\mathbf{x}_i$$

- The prior acts to “inflate” the apparent covariance of inputs.
- As A is positive (semi)definite, shrinks the weights towards the prior mean (here $\mathbf{0}$).
- If $A = \alpha I$ this is known as the ridge regression estimator.
- The MAP/shrinkage/ridge weight estimate often has lower squared error (despite bias) and makes more accurate predictions on test inputs than the ML estimate.
- An example of prior-based regularisation of estimates.

Gaussians for Regression

- Models the conditional $P(\mathbf{y}|\mathbf{x})$.
- If we also model $P(\mathbf{x})$, then learning is indistinguishable from unsupervised. In particular if $P(\mathbf{x})$ is Gaussian, and $P(\mathbf{y}|\mathbf{x})$ is linear-Gaussian, then $\mathbf{x}, \mathbf{y}$ are jointly Gaussian.
- Generalised Linear Models (GLMs) generalise to non-Gaussian, exponential-family distributions and to non-linear link functions.

  $$\begin{aligned} y_i &\sim \text{ExpFam}(\mu_i, \phi) \\ g(\mu_i) &= \mathbf{w}^{\mathsf{T}}\mathbf{x}_i \end{aligned}$$

  Posterior, or even ML, estimation is not possible in closed form ⇒ iterative methods such as gradient ascent or iteratively re-weighted least squares (IRLS).
  A warning to fMRIers: SPM uses GLM for “general” (not -ised) linear model; which is just linear.
- These models: Gaussians, Linear-Gaussian Regression and GLMs are important building blocks for the more sophisticated models we will develop later.
- Gaussian models are also used for regression in Gaussian Process Models. We’ll see these later too.

Three limitations of the multivariate Gaussian model

- What about higher order statistical structure in the data?

  ⇒ nonlinear and hierarchical models

- What happens if there are outliers?

  ⇒ other noise models

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

  ⇒ dimensionality reduction

End Notes

- It is very important that you *understand* all the material in the following cribsheet:
  `http://www.gatsby.ucl.ac.uk/teaching/courses/ml1-2014/cribsheet.pdf`
- The following notes by (the late) Sam Roweis are quite useful:
  - Matrix identities and matrix derivatives:
    `http://www.cs.nyu.edu/∼roweis/notes/matrixid.pdf`
  - Gaussian identities:
    `http://www.cs.nyu.edu/∼roweis/notes/gaussid.pdf`
- Here is a useful statistics / pattern recognition glossary:
  `http://alumni.media.mit.edu/∼tpminka/statlearn/glossary/`
- Tom Minka's in-depth notes on matrix algebra:
  `http://research.microsoft.com/en-us/um/people/minka/papers/matrix/`