Probabilistic generative models and unsupervised learning I

Zoubin Ghahramani

Department of Engineering
University of Cambridge, UK

Machine Learning Department
Carnegie Mellon University, USA

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

IPAM Probabilistic Models of Cognition
Lectures July 2007

Some Canonical Problems

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

Coin Toss

Data: $\mathcal{D} = (H\,T\,H\,H\,H\,T\,T\,\ldots)$

Parameters: $\theta \stackrel{\text{def}}{=}$ Probability of heads

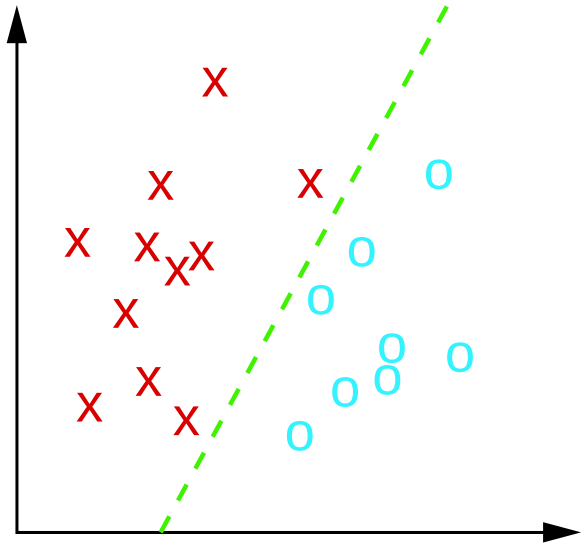
$$P(H|\theta) = \theta$$
$$P(T|\theta) = 1 - \theta$$

Goal: To infer θ from the data and predict future outcomes $P(H|\mathcal{D})$.

Linear Classification

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

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



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

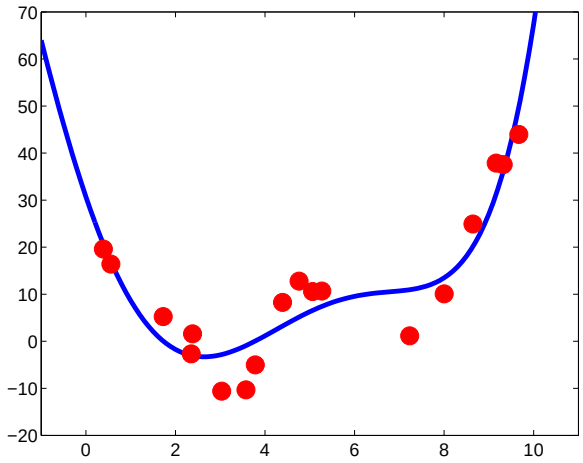
$$P(y^{(n)} = +1 | \boldsymbol{\theta}, \mathbf{x}^{(n)}) = \begin{cases} 1 & \text{if } \sum_{d=1}^{D} \theta_d \, x_d^{(n)} + \theta_0 \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

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 $n = 1, \ldots, N$

$$\begin{aligned} x^{(n)} &\in \mathbb{R} \\ y^{(n)} &\in \mathbb{R} \end{aligned}$$



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

Model:

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

where

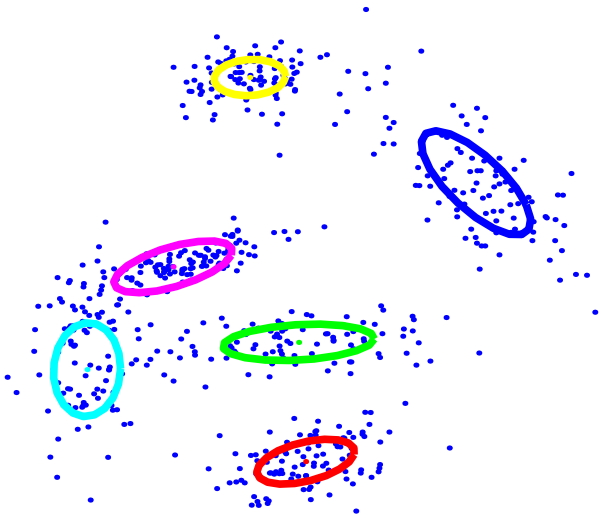
$$\epsilon \sim \mathcal{N}(0, \sigma^2)$$

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 $n = 1, \ldots, N$

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



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

Model:

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

where

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

Goal: To infer $\boldsymbol{\theta}$ from the data and predict the density $p(\mathbf{x}|\mathcal{D}, m)$

Traditionally: 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.

More "modern" view – the boundaries are blurred. Semi-supervised learning. SL and UL not really that different. One can often reduce SL and UL problems to RL. Multiple agents and game theory? Etc...

Key Ingredients

Data

We will represent data by vectors in some vector space[1]

Let $\mathbf{x}$ denote a data point with elements $\mathbf{x} = (x_1, x_2, \ldots, x_D)$

The elements of $\mathbf{x}$, e.g. x_d, represent measured (observed) features of the data point; D denotes the number of measured features of each point.

The data set $\mathcal{D}$ consists of N data points:

$$\mathcal{D} = \{\mathbf{x}^{(1)}, \mathbf{x}^{(2)} \ldots, \mathbf{x}^{(N)}\}$$

[1] This assumption can be relaxed.

Key Ingredients

Data

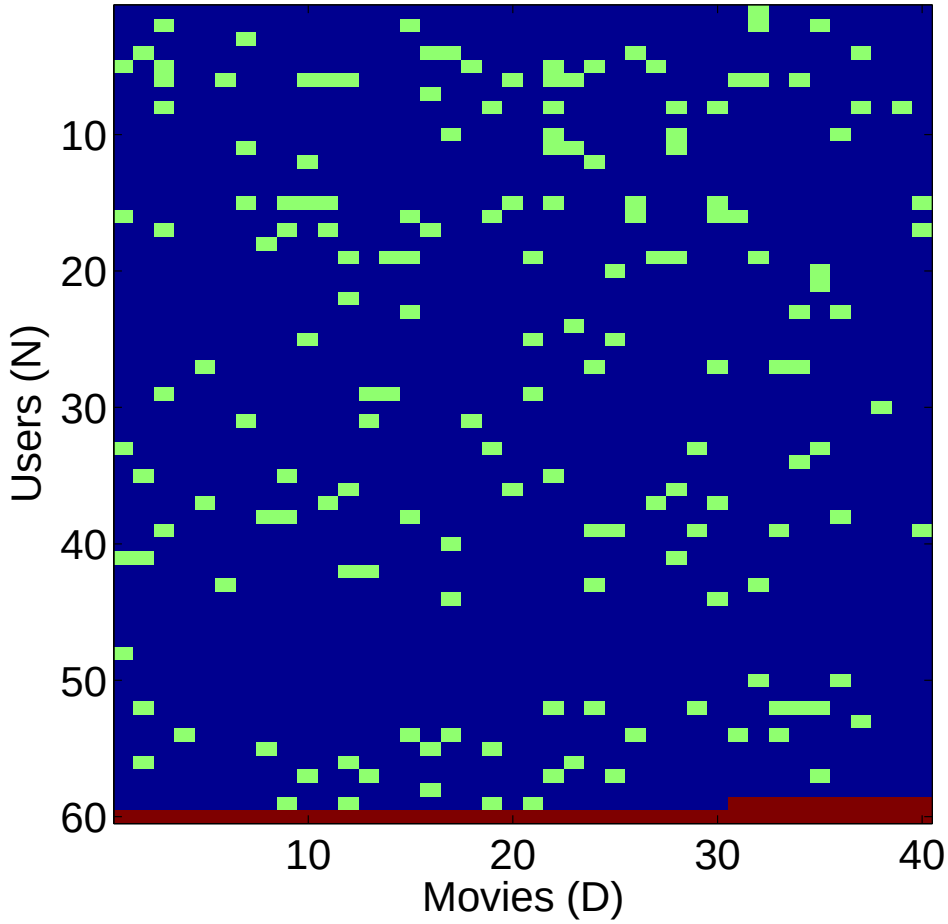
Let $\mathbf{x} = (x_1, x_2, \ldots, x_D)$ denote a data point, and $\mathcal{D} = \{\mathbf{x}^{(1)}, \mathbf{x}^{(2)} \ldots, \mathbf{x}^{(N)}\}$, a data set

Predictions

We are generally interested in predicting something based on the observed data set.

Given $\mathcal{D}$ what can we say about $\mathbf{x}^{(N+1)}$?

Given $\mathcal{D}$ and $x_1^{(N+1)}, x_2^{(N+1)}, \ldots, x_{D-1}^{(N+1)}$, what can we say about $x_D^{(N+1)}$?



Key Ingredients

Data

Let $\mathbf{x} = (x_1, x_2, \ldots, x_D)$ denote a data point, and $\mathcal{D} = \{\mathbf{x}^{(1)}, \mathbf{x}^{(2)} \ldots, \mathbf{x}^{(N)}\}$, a data set

Predictions

We are generally interested in predicting something based on the observed data set.

Given $\mathcal{D}$ what can we say about $\mathbf{x}^{(N+1)}$?

Given $\mathcal{D}$ and $x_1^{(N+1)}, x_2^{(N+1)}, \ldots, x_{D-1}^{(N+1)}$, what can we say about $x_D^{(N+1)}$?

Model

To make predictions, we need to make some *assumptions*. We can often express these assumptions in the form of a model, with some parameters, θ

Given data $\mathcal{D}$, we learn the model parameters θ, from which we can predict new data points.

The model can often be expressed as a *probability distribution over data points*

Basic Rules of Probability

Let X be a random variable taking values x in some set $\mathcal{X}$.

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

Probabilities normalise: $\sum_{x \in \mathcal{X}} P(X = x) = 1$ 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 $X = x$ and $Y = y$ is: $P(X = x, Y = y)$.

The marginal probability of $X = x$ is: $P(X = x) = \sum_y P(X = x, y)$, assuming y is discrete. I will generally write $P(x)$ to mean $P(X = x)$.

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.

Information, Probability and Entropy

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

Some axioms (informally):

- 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:

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

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 Relating to Information Theory

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

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

- Mutual information

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

- Independent random variables: $P(x, y) = P(x)P(y) \, \forall x \, \forall y$

How do we relate information theory and probabilistic modelling?

The source coding problem

Imagine we have a set of symbols $\mathcal{X} = \{\mathtt{a}, \mathtt{b}, \mathtt{c}, \mathtt{d}, \mathtt{e}, \mathtt{f}, \mathtt{g}, \mathtt{h}\}$.

We want to transmit these symbols over some binary communication channel, i.e. using a sequence of bits to represent the symbols.

Since we have 8 symbols, we could use 3 bits per symbol ($2^3 = 8$). For example: $\mathtt{a} = 000$, $\mathtt{b} = 001$, $\mathtt{c} = 010$, $\ldots$, $\mathtt{h} = 111$

Is this optimal?

What if some symbols, e.g. a, are much more probable than other symbols, e.g. f? Shouldn't we use fewer bits to transmit the more probable symbols?

Think of a discrete variable X taking on values in $\mathcal{X}$, having probability distribution $P(X)$.

How does the probability distribution $P(X)$ relate to the number of bits we need for each symbol to optimally and losslessly transmit symbols from $\mathcal{X}$?

Shannon's Source Coding Theorem

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

$$H(X) = -\sum_{x \in \mathcal{X}} P(x) \log_2 P(x)$$

Shannon's source coding theorem: Consider a random variable X, with entropy $H(X)$. A sequence of n independent draws from X can be losslessly compressed into a minimum expected code of length $n\mathcal{L}$ bits, 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

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

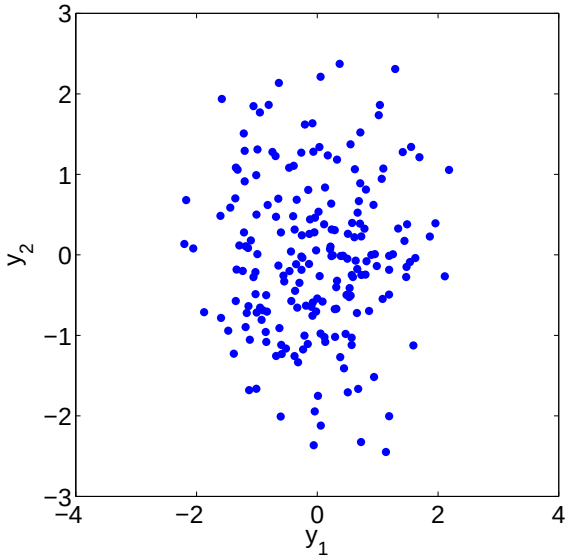
where the relative-entropy or Kullback-Leibler divergence is

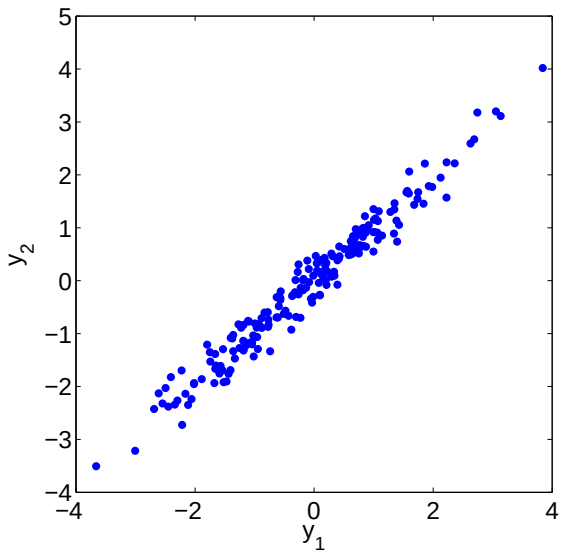
$$KL(P\|Q) = \sum_x P(x) \log_2 \frac{P(x)}{Q(x)} \geq 0$$

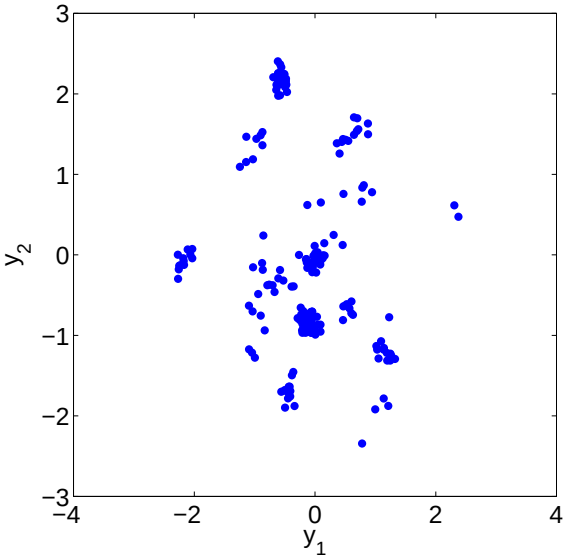
Take home message: better probabilistic models $\equiv$ more efficient codes.

Modelling Data and Parameter Estimation

A few simple data sets

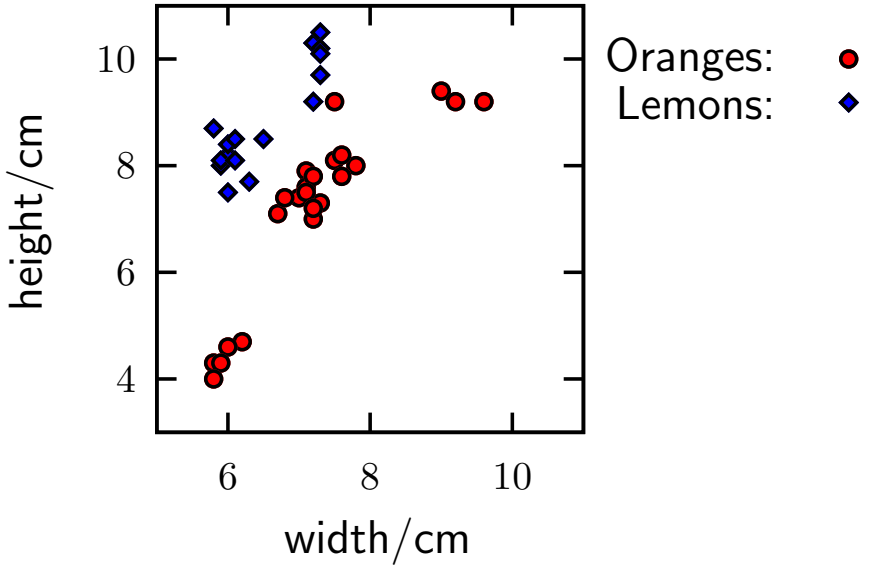






A more interesting data set:

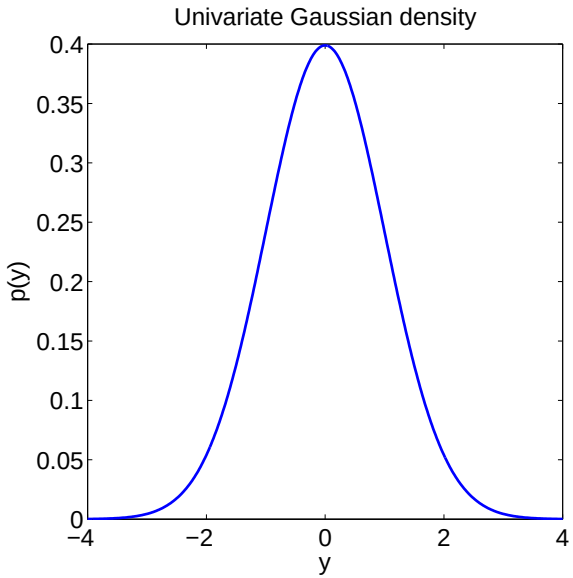
Here $D = 2$, $\mathbf{y} \in \mathbb{R}^2$.



A very simple model

Univariate Gaussian density ($y \in \mathbb{R}$):

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



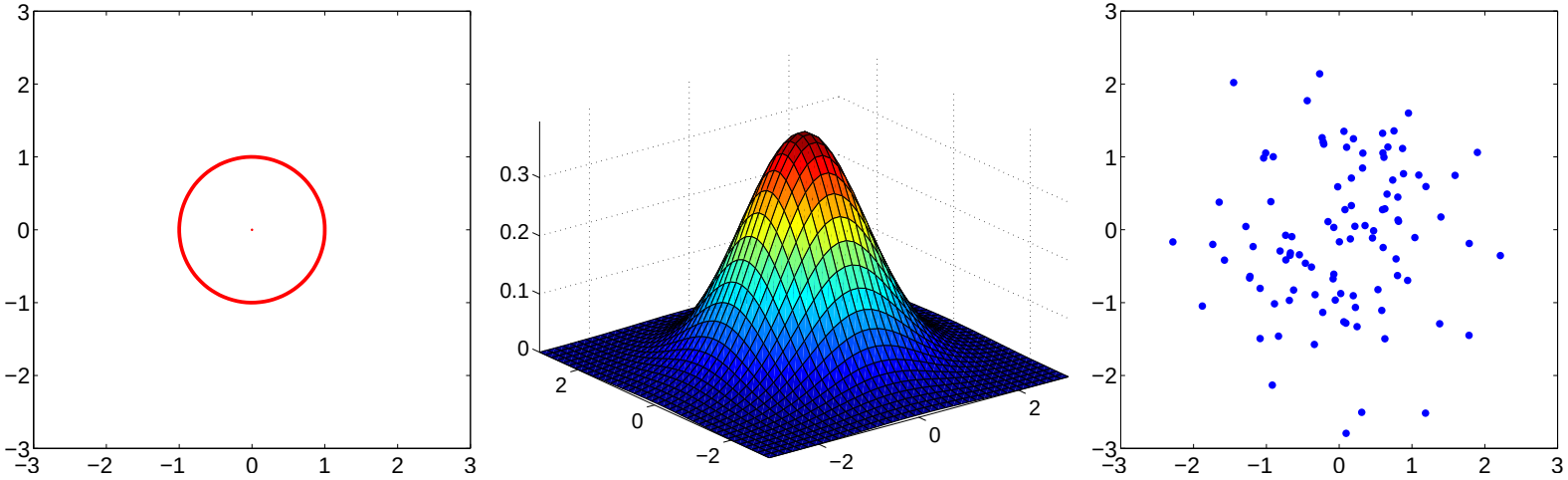
This model has parameters $\boldsymbol{\theta} = \{\mu, \sigma\}$ which model the mean and standard deviation of the data, respectively.

A slighly more complicated model

Multivariate Gaussian density ($\mathbf{y} \in \mathbb{R}^D$):

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

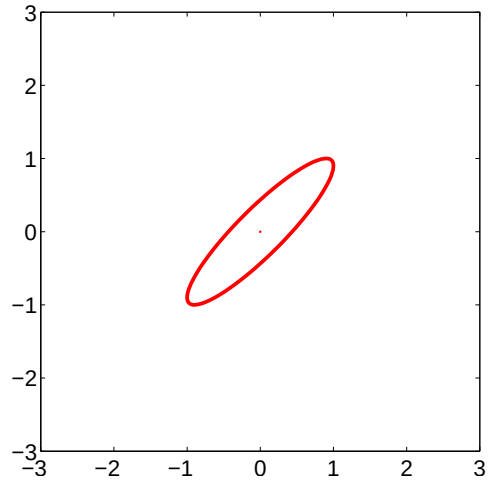
$$\mu = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \quad \Sigma = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

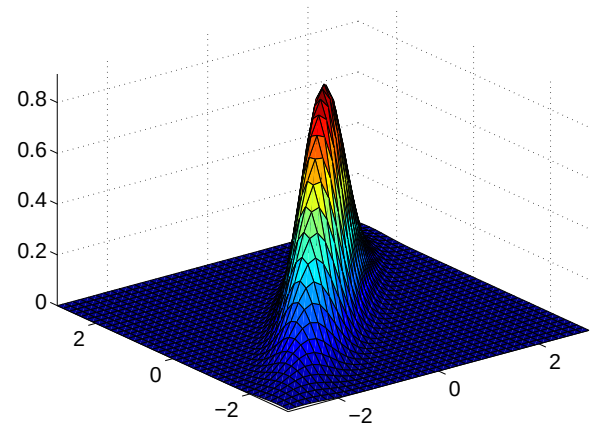


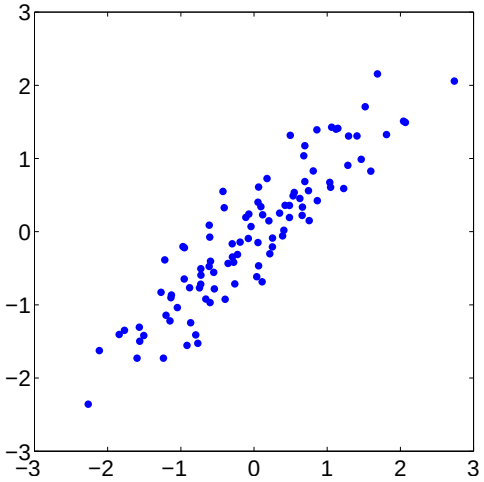
This model has parameters $\boldsymbol{\theta} = \{\mu, \Sigma\}$ which model the mean and covariance matrix of the data.

The multivariate Gaussian density

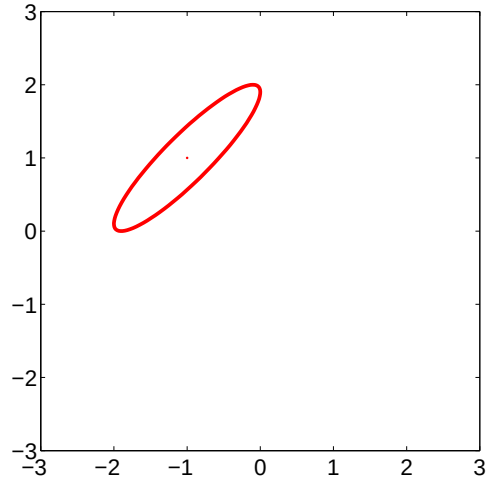
$$\mu = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \quad \Sigma = \begin{bmatrix} 1 & 0.9 \\ 0.9 & 1 \end{bmatrix}$$

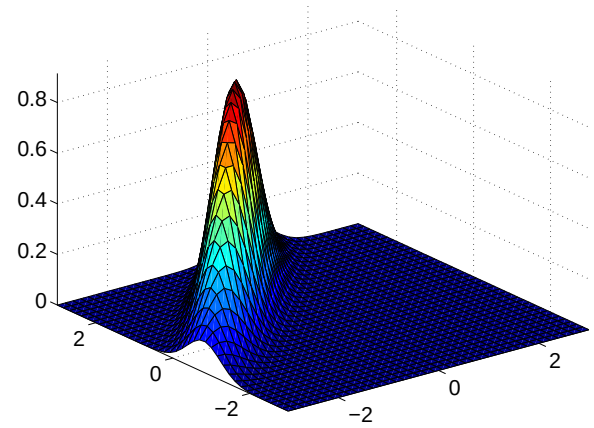


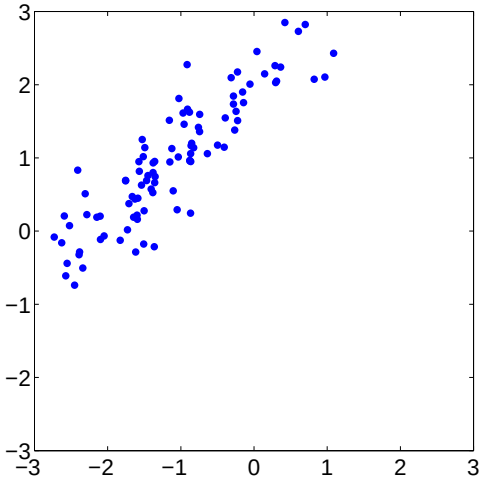




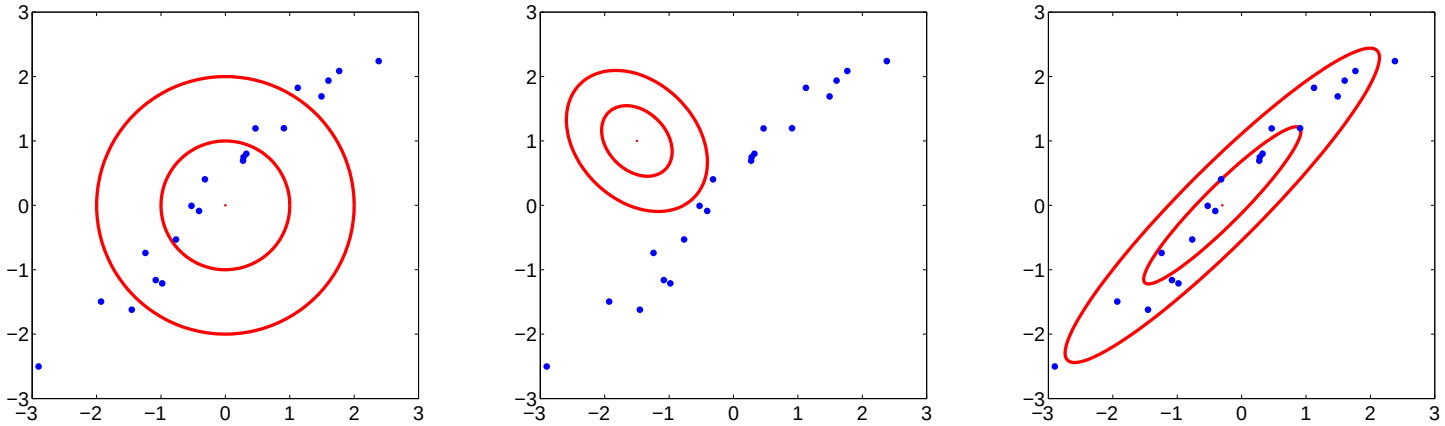
$$\mu = \begin{bmatrix} -1 \\ 1 \end{bmatrix} \quad \Sigma = \begin{bmatrix} 1 & 0.9 \\ 0.9 & 1 \end{bmatrix}$$







Fitting the model to data



Assume the data were generated independently from the model.
We can measure the likelihood of the model:

$$p(\mathcal{D}|\boldsymbol{\theta}) = \prod_{n=1}^{N} p(\mathbf{y}_n|\boldsymbol{\theta})$$

Clearly, the third model is a better fit to the data than the others:

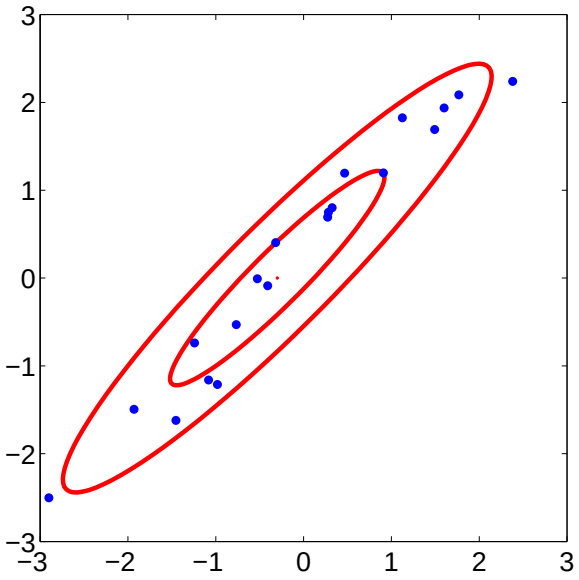
$$\begin{aligned} \log p(\mathcal{D}|\boldsymbol{\theta}_1) &= -55.38 \\ \log p(\mathcal{D}|\boldsymbol{\theta}_2) &= -238.29 \\ \log p(\mathcal{D}|\boldsymbol{\theta}_2) &= -22.14 \end{aligned}$$

The likelihood function

Data set $\mathcal{D} = \{\mathbf{y}_1, \ldots, \mathbf{y}_N\}$, the likelihood: $p(\mathcal{D}|\mu, \Sigma) = \prod_{n=1}^{N} p(\mathbf{y}_n|\mu, \Sigma)$ is a function of the model parameters

The maximum likelihood (ML) procedure finds parameters $\boldsymbol{\theta} = \{\mu, \Sigma\}$ such that:

$$\boldsymbol{\theta}_{\mathrm{ML}} = \mathrm{argmax}_{\boldsymbol{\theta}}\, p(\mathcal{D}|\boldsymbol{\theta})$$



Finding Maximum Likelihood Estimate for a Gaussian

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

Maximise likelihood $\Leftrightarrow$ maximise log likelihood

Goal: find μ and Σ that maximise log likelihood:

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

Note: equivalently, minimise $-\mathcal{L}$, which is *quadratic* in μ

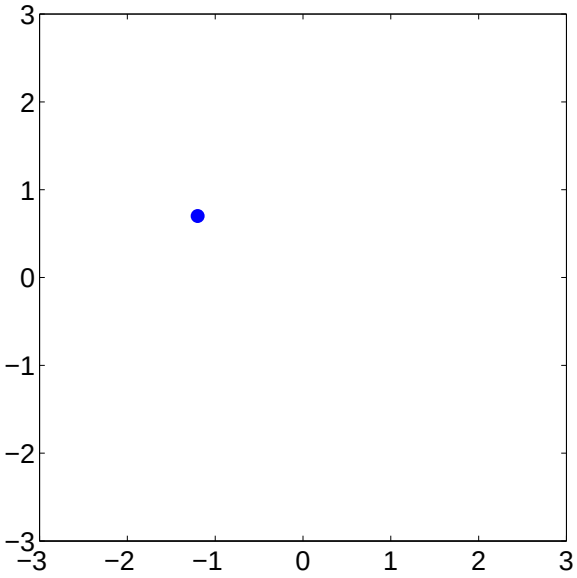
Procedure: take derivatives and set to zero:

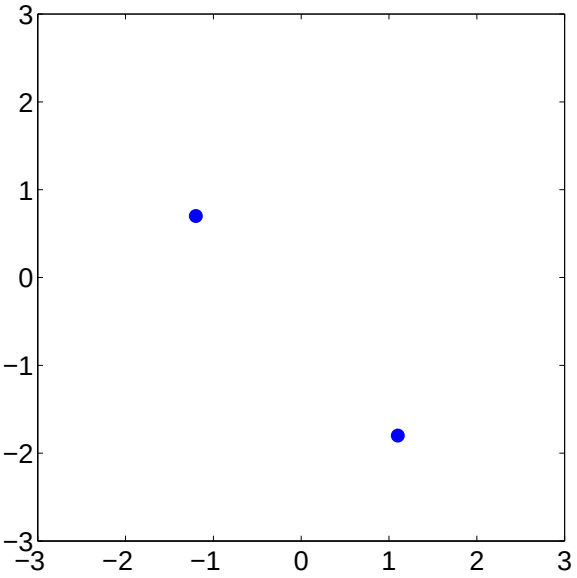
$$\frac{\partial \mathcal{L}}{\partial \mu} = 0 \quad \Rightarrow \quad \hat{\mu} = \frac{1}{N} \sum_n \mathbf{y}_n \quad \text{(sample mean)}$$

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

Two *very* simple data sets

What are the maximum likelihood estimates of θ for these data sets?





Does this make sense?

Maximum a Posteriori (MAP) Learning

- The maximum likelihood (ML) procedure finds parameters $\boldsymbol{\theta}$ such that:

$$\boldsymbol{\theta}_{\mathrm{ML}} = \mathrm{argmax}_{\boldsymbol{\theta}}\, p(\mathcal{D}|\boldsymbol{\theta})$$

  As we've seen this can give very silly results when we have small data sets.

- A common "fix": define a prior over the parameters $p(\boldsymbol{\theta})$ and try to find the maximum a posteriori (MAP) parameters:

$$\begin{aligned} \boldsymbol{\theta}_{\mathrm{MAP}} &= \mathrm{argmax}_{\boldsymbol{\theta}}\ p(\mathcal{D}|\boldsymbol{\theta})p(\boldsymbol{\theta}) \\ &= \mathrm{argmax}_{\boldsymbol{\theta}}\ \log p(\mathcal{D}|\boldsymbol{\theta}) + \log p(\boldsymbol{\theta}) \end{aligned}$$

  The log prior can be seen as a penalty terms that prefers some parameters to others. For example, we can avoid singular covariance matrices this way.

- Closely related to regularization or maximum penalized likelihood (MPL)

$$\boldsymbol{\theta}_{\mathrm{MPL}} = \mathrm{argmax}_{\boldsymbol{\theta}}\ \log p(\mathcal{D}|\boldsymbol{\theta}) - \lambda\, r(\boldsymbol{\theta})$$

  where $r(\cdot) > 0$ is a function known as the regularizer and λ is a non-negative *regularization parameter*. For example, $r(\boldsymbol{\theta}) = \sum_i \theta_i^2$ prefers small parameters.

Comments on MAP and Penalized Likelihoods

$$\begin{aligned} \boldsymbol{\theta}_{\mathrm{MAP}} &= \mathrm{argmax}_{\boldsymbol{\theta}} \log p(\mathcal{D}|\boldsymbol{\theta}) + \log p(\boldsymbol{\theta}) \\ \boldsymbol{\theta}_{\mathrm{MPL}} &= \mathrm{argmax}_{\boldsymbol{\theta}} \log p(\mathcal{D}|\boldsymbol{\theta}) - \lambda r(\boldsymbol{\theta}) \end{aligned}$$

- They are simular but not equivalent:
  - MPL is invariant to one to one reparameterization $\boldsymbol{\phi} = f(\boldsymbol{\theta})$.
  - MAP is not invariant to reparameterization $\boldsymbol{\phi} = f(\boldsymbol{\theta})$ since a nonlinear reparameterization can "squeeze" parts of the density and change the location of the maximum.

- Picking a MAP point estimate is not well justified from a Bayesian framework – at best an approximation.

- Regularization is *very* popular, but choice of λ and form of r is ad hoc. It is inadequate to think of complexity as being measured by a single scalar parameter, λ.

Bayesian Learning

Apply the basic rules of probability to learning from data.
Use probability distributions to represent uncertainty.

Data set: $\mathcal{D} = \{\mathbf{y}_1, \ldots, \mathbf{y}_N\}$
Model parameters: $\boldsymbol{\theta}$

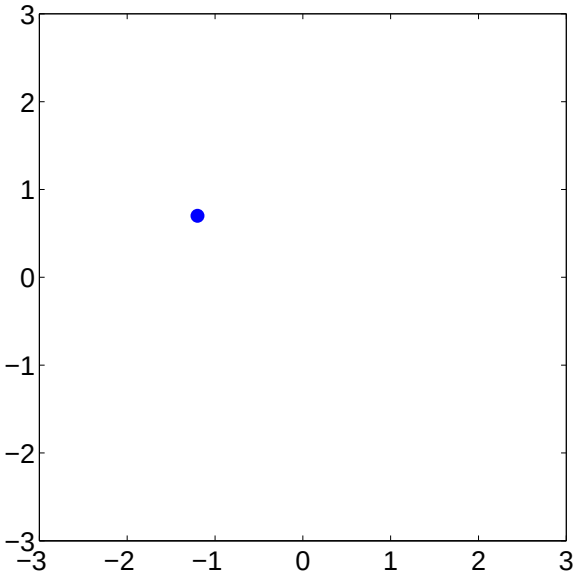
Prior probabilities of model parameters: $P(\boldsymbol{\theta})$
Model of data given parameters (likelihood model): $P(\mathbf{y}|\boldsymbol{\theta})$

If the data are independently and identically distributed then:

$$P(\mathcal{D}|\boldsymbol{\theta}) = \prod_{n=1}^{N} P(\mathbf{y}_n|\boldsymbol{\theta})$$

Posterior probability of model parameters:

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



Foundations of Bayesian Learning

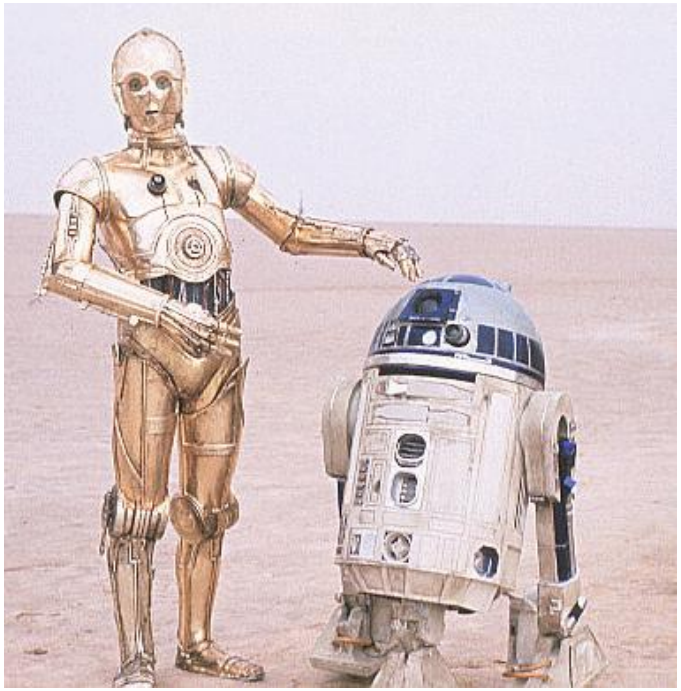
Representing Beliefs in Artificial Intelligence

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:

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

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 distibutions $\rho(P_1, P_2) = \sup_E |P_1(E) - P_2(E)|$)

coin toss demo: `bayescoin`

Bayesian Occam's Razor and Model Comparison

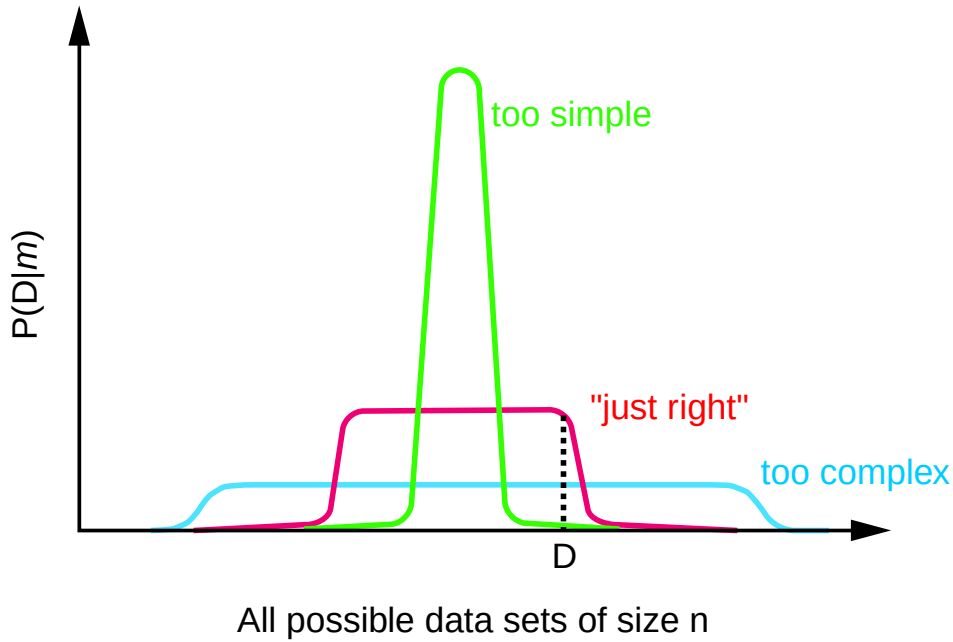
Compare model classes, e.g. m and m', using posterior probabilities given $\mathcal{D}$:

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

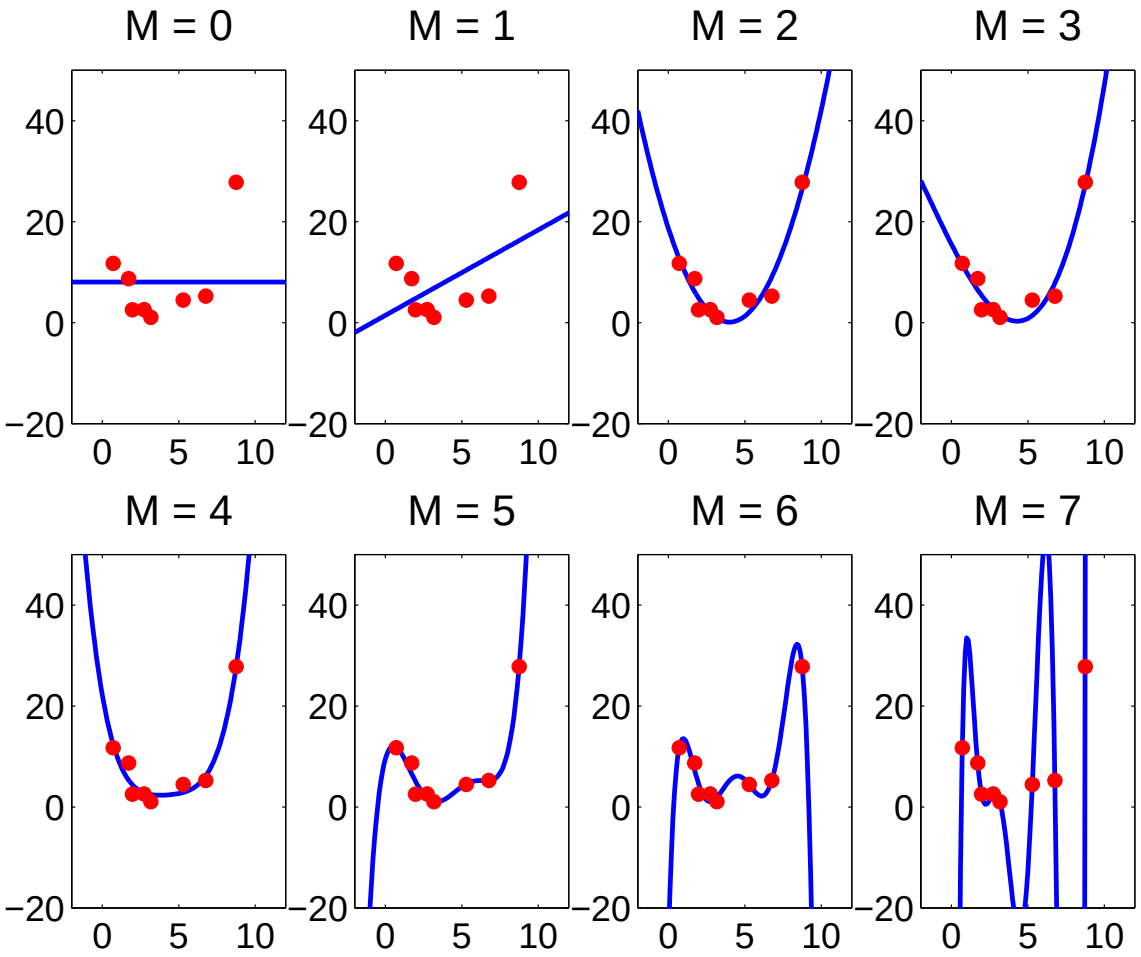
Interpretation of the Marginal Likelihood ("evidence"): The probability that *randomly selected* parameters from the prior would generate $\mathcal{D}$.

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

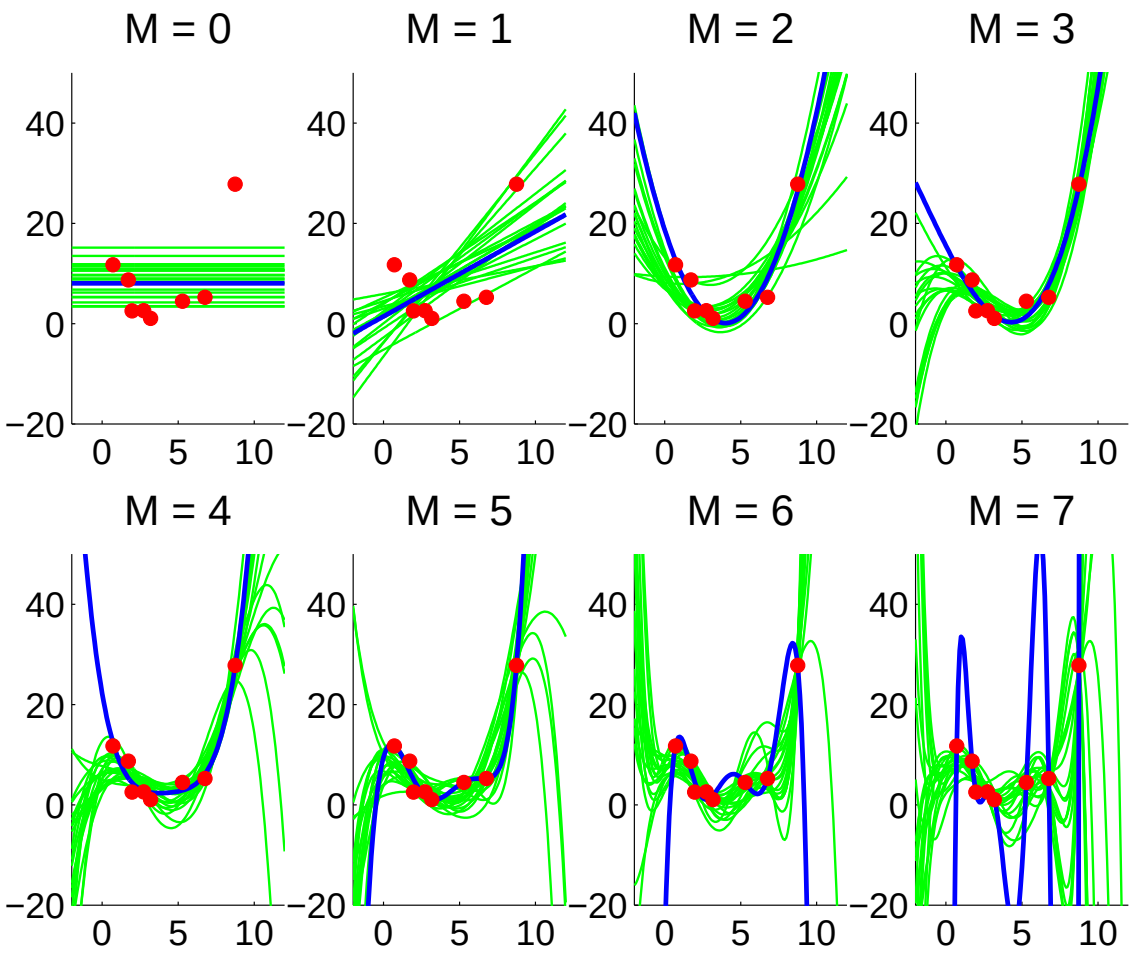
Model classes that are too complex can generate many possible data sets, so again, they are unlikely to generate that particular data set at random.

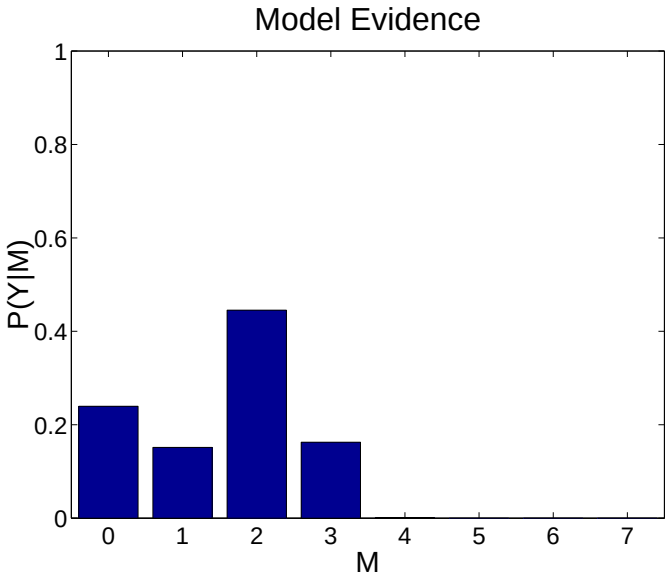


Model structure and overfitting:
A simple example: polynomial regression



Bayesian Model Comparison: Occam's Razor at Work





demo: polybayes

On Choosing Priors: Different Schools

- **Objective Priors**: noninformative priors that attempt to capture ignorance and have good frequentist properties.
- **Priors of Convenience**: some priors (e.g. conjugate priors) lend themselves to analytical solutions and computationally efficient inference. Such practical considerations are often used to pick priors.
- **Hierarchical Priors**: multiple levels of priors:[2]

$$p(\theta) = \int d\alpha\, p(\theta|\alpha)p(\alpha) = \int d\alpha\, p(\theta|\alpha) \int d\beta\, p(\alpha|\beta)p(\beta) \quad \text{(etc...)}$$

- **Empirical Priors**: learn some of the parameters of the prior from the data ("Empirical Bayes")
- **Subjective Priors**: priors should capture our beliefs as well as possible. They are subjective but not arbitrary.

The Dutch Book Theorem and Cox-Jaynes Axioms suggest the only coherent framework is the Subjective Bayesian framework – but many people in Statistics and Machine Learning don't like this.

[2] Hierarchical priors are not mutually exclusive with the other categories.