# Inference in Statistical Modelling and Machine Learning

A short course

May 29, 2026

# Table of contents

# Table of contents

# Orientation: Statistical Modelling, Machine Learning and Inference

# Machine Learning vs Statistics

## Machine Learning

A machine is said to learn from experience E with respect to some class of tasks T, and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.

*Tom Mitchell (1997)*

## Statistics

If statistics incorporated computing methodology from its inception as a fundamental tool, as opposed to simply a convenient way to apply existing tools, [Machine Learning] would not have needed to exist—it would have been part of statistics.

*Jerry Friedman (1997)*

# Inference

## Inference in everyday life

- The process of reaching a conclusion based on evidence and reasoning.
- Some inferences are certain.
- Some inferences are *risky* (we are interested in these).

## Statistical Inference (a.k.a. statistical / machine learning)

- Learn (infer) a probabilistic model of your observations.
- Make predictions, spot patterns or take decisions using your model.
- We often classify learning problems as **supervised** or **unsupervised**

# Supervised Learning

**Supervised learning** tasks require us to predict some *response* $y$ based on some predictor $x$. Our prediction is

$$\underbrace{\hat{y}}_{\text{prediction}} = \underbrace{\hat{g}(x)}_{\text{prediction function}}$$

**Examples:**

- given a series of clinical measurements, including blood sugar levels, weight, height, blood pressure and so on, predict whether that person has diabetes;
- given a photograph of a plant, predict the plant's species;
- given an audio recording of a person speaking, predict the person's age.

## Regression vs Classification

- $y$ continuous $\Longrightarrow$ Regression
- $y$ categorical $\Longrightarrow$ Classification

## Exercise

For each of the supervised learning examples above

(i) Identify the task (T), the experience (E), and suggest a performance measure (P).

(ii) Identify each example as a classification or a regression problem.

# Overfitting vs underfitting

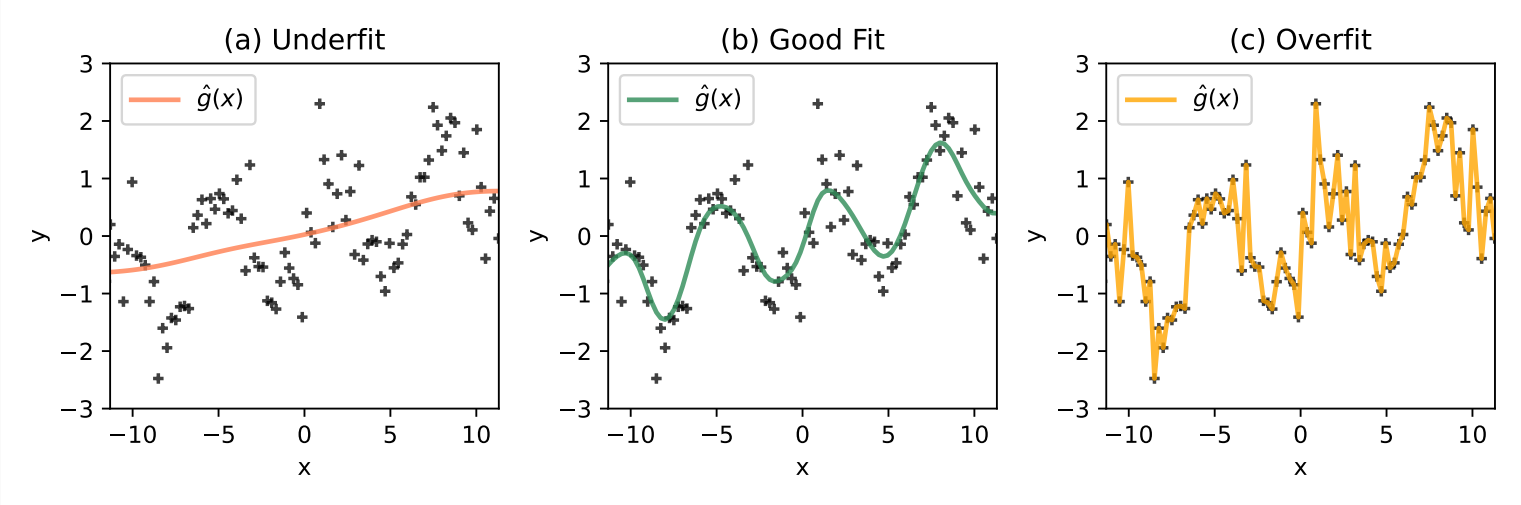



**Figure 1:** Prediction functions fitted to the same predictor-response data.

## Exercise

There are **infinitely** many choices for $\hat{g}$. Which $\hat{g}$ above would you use to predict the response based on a **new and unseen** predictor and why?

# Unsupervised learning

**Unsupervised learning** tasks require us to understand *patterns* in data or to build a model of the data *as a whole*. There is no distinction between predictor and response. Often we seek an approximate **density** $\hat{f}(x)$ which describes how *typical* $x$ is.

**Examples:**

- Given a list of financial transaction sizes for a single customer, estimate how *typical* a new transaction is for that customer.
- Given a set of speech sounds from an unknown language, determine how many *phonemes* are in the language.

**Exercise**

For each of the unsupervised learning examples above identify the task (T), the experience (E), and suggest a performance measure (P).

# Overfitting vs underfitting

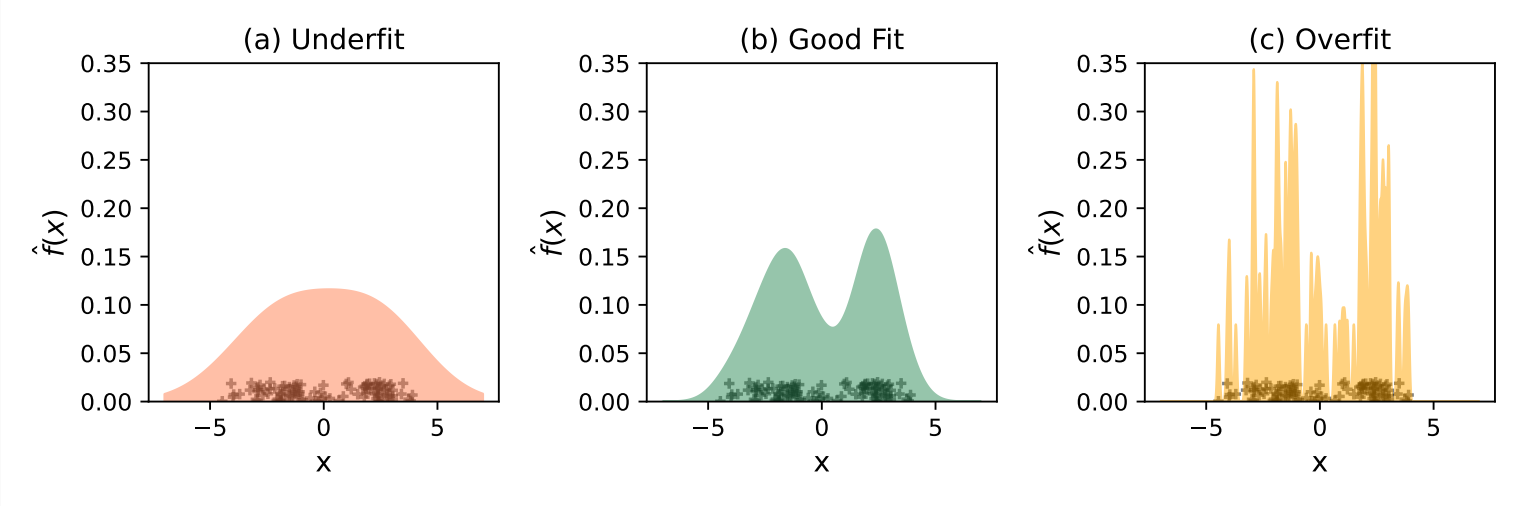



**Figure 2:** Three different candidates for a density function $\hat{f}$, fitted to the same set of data $x_1, x_2, \ldots, x_{50}$ from some source (e.g. standardised transaction sizes).

## Exercise

Which $\hat{f}$ above would you use to measure how typical a **new and unseen** observation from the same source was, and why?

# Supervised Learning Warm-up

# A medieval prediction problem

You are the court mathematician in a medieval kingdom. The threat of war is ever present...

- The Queen's inventors have devised a new weapon whose mechanism of action remains a closely kept secret.
- The measured power, $y$, appears to depend on a quantity $x$ which the inventors control.
- Each day, the inventors perform a test using a different value of $x$, with the aim of finding how it affects $y$.
- After each test they hand you an envelope which contains the results $(x_k, y_k)$, where $k$ denotes the test number.
- Can you come up with a way of predicting $y$ from $x$ in future tests?

# A simple model

- You have dataset $D = \{(x_i, y_i)\}_{i=1}^{40}$ (forty envelopes)
- To predict $y$ from $x$ we can fit a line or curve.

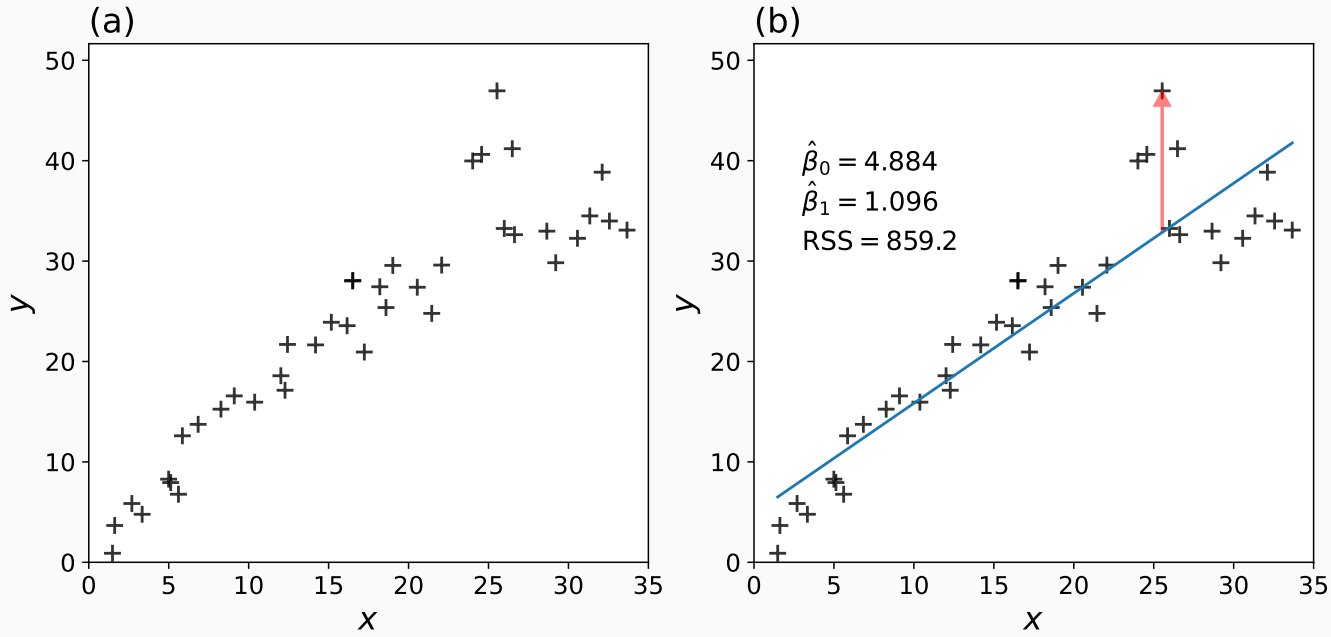



**Figure 3:** (a) Raw data $D$ provided by the Queen's inventors. (b) A prediction line fitted to $D$. Red arrow shows a prediction error.

# Prediction functions

- Given $x$ we want a prediction $\hat{y}$ of $y$.
- The blue line represents a **prediction function**, $\hat{g}$, that maps $x$ values to $\hat{y}$ values.
- The general form of functions in the $\mathrm{LIN}$ (linear) family is $g(x) = \beta_0 + \beta_1 x$
- Given **predictor** $x$ we predict **response**

$$\hat{y} = \hat{g}(x) = \hat{\beta}_0 + \hat{\beta}_1 x$$

- $\hat{\beta}_0$ and $\hat{\beta}_1$ are **coefficient estimates**.

## Exercise

Suppose we have a prediction function $\hat{g}(x) = 5 + x$. Consider the data point $(x_i, y_i) = (20, 30)$.

(i) What is our predicted value, $\hat{y}_i$, of $y_i$ given $x_i$?

(ii) The error or **residual** in this prediction is

$$\hat{\epsilon}_i = y_i - \hat{y}_i.$$

Calculate this residual.

# Fitting prediction functions

- Given a prediction function $\hat{g}$, we can measure its performance on $D$ with the **residual sum of squares**

$$\mathrm{RSS}(\hat{g}; D) = \sum_{(x,y) \in D} (y - \hat{g}(x))^2 .$$

- The best performing linear prediction function is

$$\hat{g} = \operatorname*{arg\,min}_{g \in \mathbf{LIN}} \mathrm{RSS}(g; D).$$

  $\hat{g}$ is the function in the $\mathbf{LIN}$ family with the smallest $\mathrm{RSS}$.

- The $\mathrm{RSS}$ is an example of a **loss function** which we minimise to learn the **parameters** of a prediction function.

# arg min and arg max

The expression

$$\arg\min_{a \in A} \phi(a)$$

denotes the element of set $A$ at which the function $\phi$ defined on $A$ achieves its minimum in $A$. So, for example,

$$\arg\min_{x \in \mathbb{R}} (x^2 - 2) = 0.$$

The expression

$$\arg\max_{a \in A} \phi(a)$$

denotes the element of set $A$ at which $\phi$ achieves its *maximum* in $A$.

**Exercise**

Find

$$\arg\min_{x \in \mathbb{R}} (x^2 - 4x + 10).$$

## Exercise

Suppose you want to fit the constant prediction function $g(x) = c$ to a dataset $D = \{(x_i, y_i)\}_{i=1}^n$. The residual sum of squares depends only on $c$ and the data, and can be written

$$\text{RSS}(c; D) = \sum_{i=1}^{n} (y_i - c)^2.$$

Show that the residual sum of squares is minimized when $c$ is the mean of the $y$ values in $D$. That is

$$\operatorname*{arg\,min}_{c \in \mathbb{R}} \text{RSS}(c; D) = \frac{1}{n} \sum_{i=1}^{n} y_i.$$

## Exercise

Let LINO be the family of linear functions which pass through the origin, having the form

$$g(x) = \beta x.$$

Given a dataset $D = \{(x_i, y_i)\}_{i=1}^{n}$, the RSS using a LINO prediction function is

$$\text{RSS}(\beta; D) = \sum_{k=i}^{n} (y_i - \beta x_i)^2.$$

Find an expression for the parameter value $\hat{\beta}$ which minimises the RSS.

# More prediction functions

- Maybe the linear prediction function is too simple.
- Suppose we try functions of the form

$$g(x) = \beta_0 + \beta_1 x + \beta_2 x^2.$$

- We choose our prediction function by minimising the $\mathrm{RSS}$

$$\hat{g} = \operatorname*{arg\,min}_{g \in \mathrm{QUAD}} \mathrm{RSS}(g; D),$$

  where QUAD is the family of all quadratic functions of $x$.
- We could go even further and try the 'P15' family of functions, which have the form $g(x) = \sum_{i=0}^{15} \beta_i x^i$.

# Results

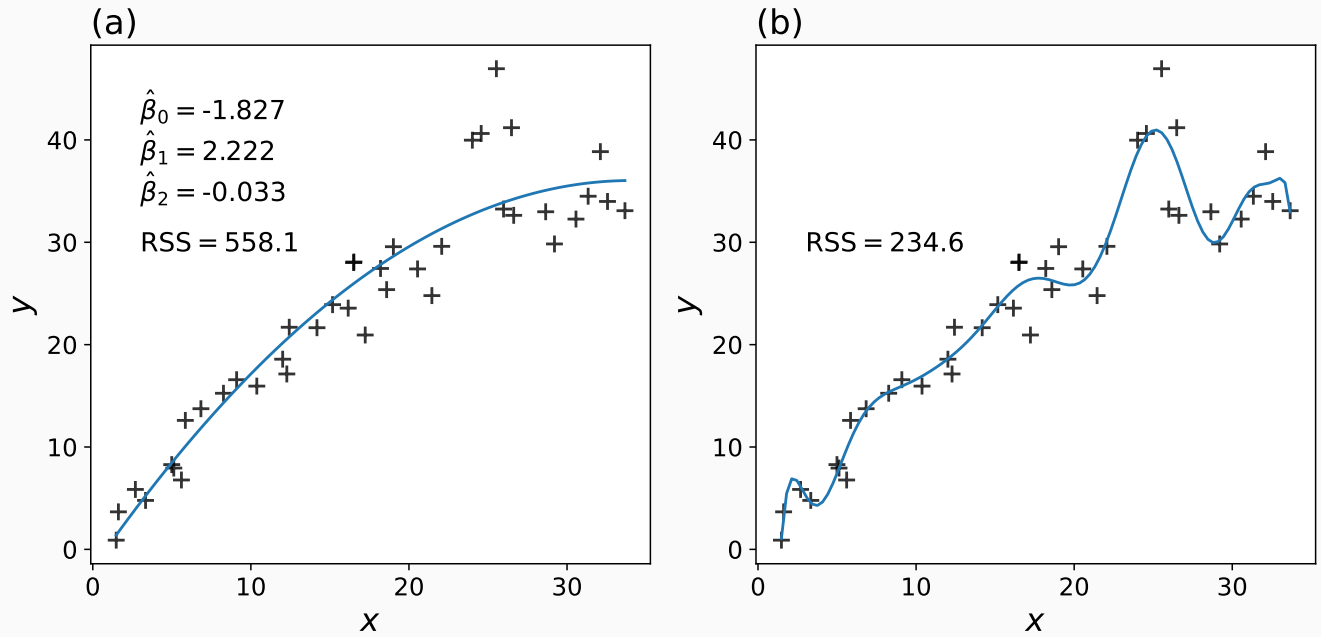



**Figure 4:** (a) QUAD and (b) P15 family prediction curves fitted to $D$ by minimizing RSS.

**Exercise**

Which curve would you use to predict future results and why?

# Cross validation

- We want to predict **future** $y$ values from **future** $x$ values.
- RSS measures performance on **current** data.
- Does low RSS mean a better prediction function? No!
- We want to know how well $\hat{g}$ performs on data **not** in $D$.
- We use 'cross validation' to estimate this performance.

**Leave one out cross validation (LOOCV)**

For each $i \in \{1, 2, \ldots, n\}$

1. Remove data point $(x_i, y_i)$ from $D$ to give $D_{-i}$.
2. Find $\hat{g}_{-i}$ using $D_{-i}$.
3. Calculate residual $r_i = y_i - \hat{g}_{-i}(x_i)$.

Calculate cross validated mean squared error $\mathrm{MSE}^{\mathrm{CV}} = \frac{1}{n}\sum_{i=1}^{n} r_i^2$

# Results for polynomials

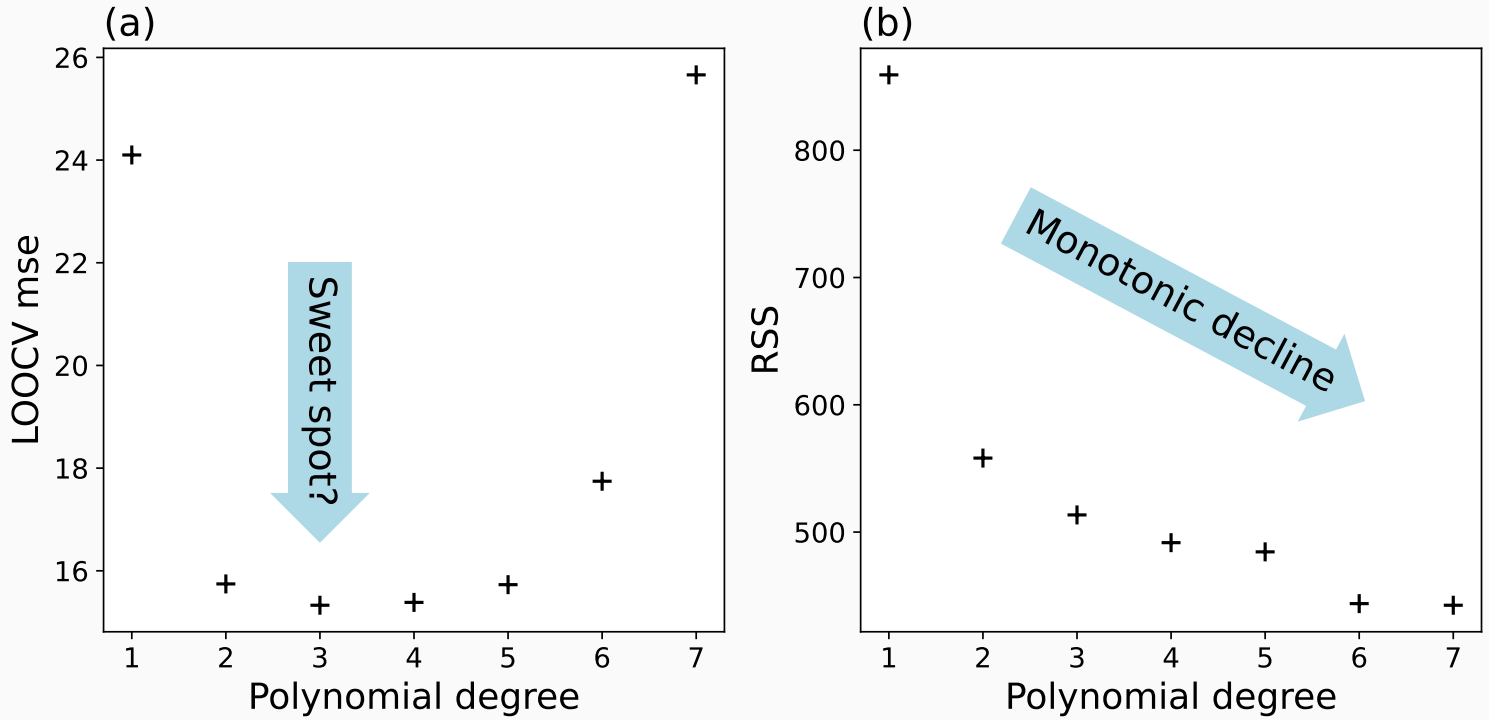



**Figure 5:** (a) LOOCV mean squared error and (b) RSS for polynomials of degrees 1 through 7 fitted to *D*.

# Signal and noise

We can caricature the relationship between $y$ and $x$ as

$$y = \underbrace{\mathfrak{g}(x)}_{\text{systematic term}} + \underbrace{\text{noise}}_{\text{unpredictable term}}$$

- Cross validation estimates the **out of sample** performance of our prediction function.
- Excessively flexible prediction function families perform poorly because they **fit to noise in the data**.
- Excessively simple prediction function families perform poorly because they **fail to capture systematic patterns**.
- Increasing prediction functions flexibility will **always** reduce the RSS.
- The RSS cannot be used to detect overfitting.

# Some new information arrives

- It is **rumoured** that the weapon exploits a chemical reaction between two powders (A and B), with $x$ the percentage of A.
- When $x = x_{\text{opt}}$ the reaction is **complete**.
- When $x < x_{\text{opt}}$, A is the limiting factor so $y \propto x$. When $x > x_{\text{opt}}$, the situation flips. This is an AB-MIX function.

$$g(x) = \begin{cases} \frac{y_{\max}}{x_{\text{opt}}} x & \text{when } x \in [0, x_{\text{opt}}] \\ \frac{y_{\max}}{100 - x_{\text{opt}}} (100 - x) & \text{when } x \in [x_{\text{opt}}, 100] \end{cases}$$

# Cross validation

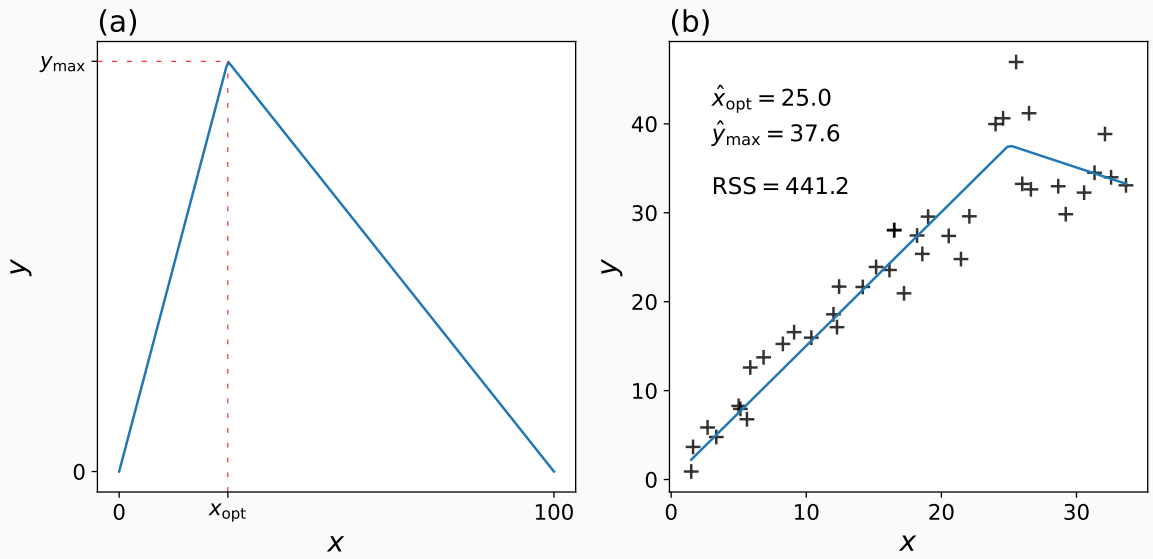



**Figure 6:** (a) A member of AB-MIX. (b) Optimal AB-MIX function.

Cross validation results: $\mathrm{MSE}^{\mathrm{CV}}_{\text{AB-MIX}} = 12.9$, $\mathrm{MSE}^{\mathrm{CV}}_{\mathrm{P3}} = 15.3$.

**Exercise**

Which model should we use to predict the result of next day's test?

# A polynomial surprise

If we explore higher order polynomials, we get a surprise...

$\mathrm{MSE}_{\mathrm{P13}}^{\mathrm{CV}} = 11.5$, outperforming P3 and AB-MIX.

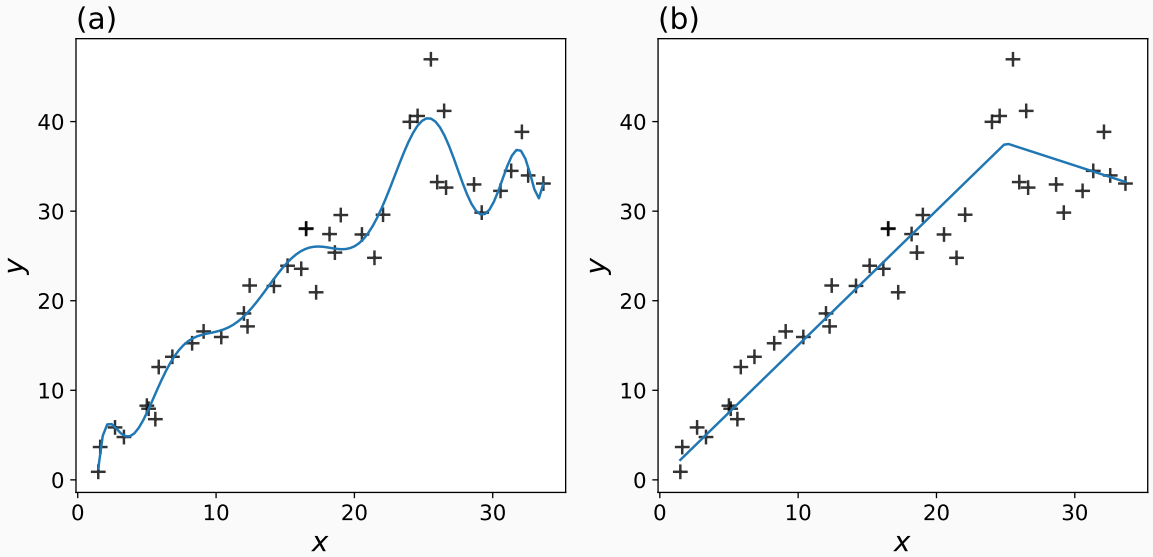



**Figure 7:** (a) Optimal P13 prediction function. (b) Optimal AB-MIX prediction function.

# Background knowledge is essential

**Exercise**

Would you choose P13 or AB-MIX?

- If we look hard enough, it will **always** be possible to find a **highly flexible** function family that gets a tiny cross validation error on $D$.
- We'll call this family CHEAT
- Its low cross validation error will almost certainly be a **fluke**.
- It will perform poorly on genuinely **new data**.
- Cross validation alone is not sufficient to select models.
- We **must** use common sense to make sensible **assumptions**.

# Some new data arrives ($D_{\text{test}}$)

The MSE on this **test data** is

$$\text{MSE}(\hat{g}, D_{\text{test}}) = \frac{1}{|D_{\text{test}}|} \sum_{(x,y) \in D_{\text{test}}} (y - \hat{g}(x))^2$$

We find $\text{MSE}^{\text{TEST}}_{\text{AB-MIX}} = 29.7$ and $\text{MSE}^{\text{TEST}}_{\text{P13}} = 168$

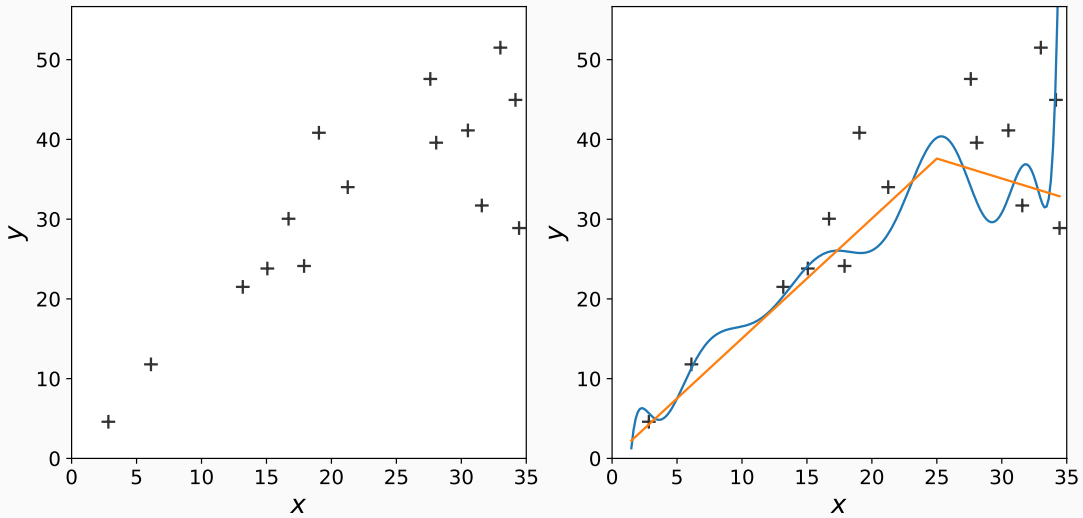



**Figure 8:** (a) $D_{\text{test}}$. (b) P13 and AB-MIX fitted to $D$, but plotted alongside $D_{\text{test}}$.

# Conclusions

1. Given a set of predictor-response pairs, we can learn the parameters of a **prediction function** by minimising a **loss**.
2. More **flexible** prediction function families usually reduce loss. Excessive flexibility leads to **overfitting** (fitting to noise).
3. Overfitting leads to poor performance on **unseen** data.
4. **Cross-validation** estimates the performance of a prediction method (e.g. 'fit a P3 function by minimising RSS') on unseen data.
5. When cross-validation is used to compare a large number of prediction methods, the winner's score is not a reliable guide to its performance on unseen data.
6. When deciding which prediction methods to compare, we must rely on **prior knowledge** to pick plausible candidates.

# Unsupervised Learning Warm-up

# Another medieval prediction problem

At times of danger to the realm, the Queen's predecessors instructed their officials to bury hundreds of hoards of treasure at secret locations.

- Locations were recorded on paper slips, locked in the archives.
- All but 90 slips have been eaten by bookworms.

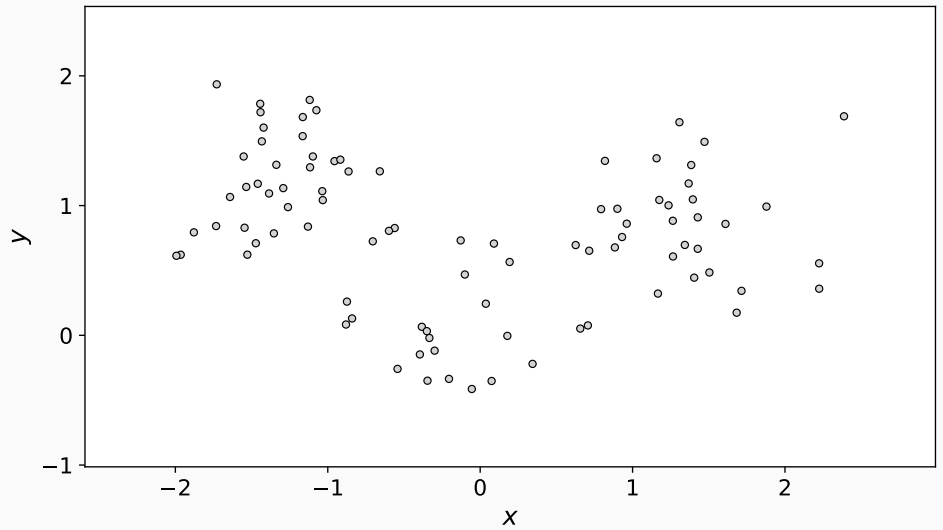



**Figure 9:** Map of known hoard locations. The set of $(x, y)$ pairs plotted here is our dataset, $D$. The coordinate system is a state secret.

# Hoard score function

The Queen's ministers would like you to analyse $D$ and provide them with a 'hoard score' function, $f$.

- The hoard score function will be used to decide where to search.
- The hoard score $f(x, y)$ should be a non-negative number proportional to the estimated chance of finding a new hoard when excavating at $(x, y)$.

If

$$\int_{\mathbb{R}^2} f(x, y) dx dy = 1$$

then we can interpret $f$ as a **probability density**.

**Exercise**

Probability densities describe the distributions of random variables. What random variables does $f$ describe?

# Hoards along a line

- Let's work through a simpler one-dimensional problem.
- Figure 10 shows a cluster of hoards located at $x_1, x_2, \ldots, x_n$.

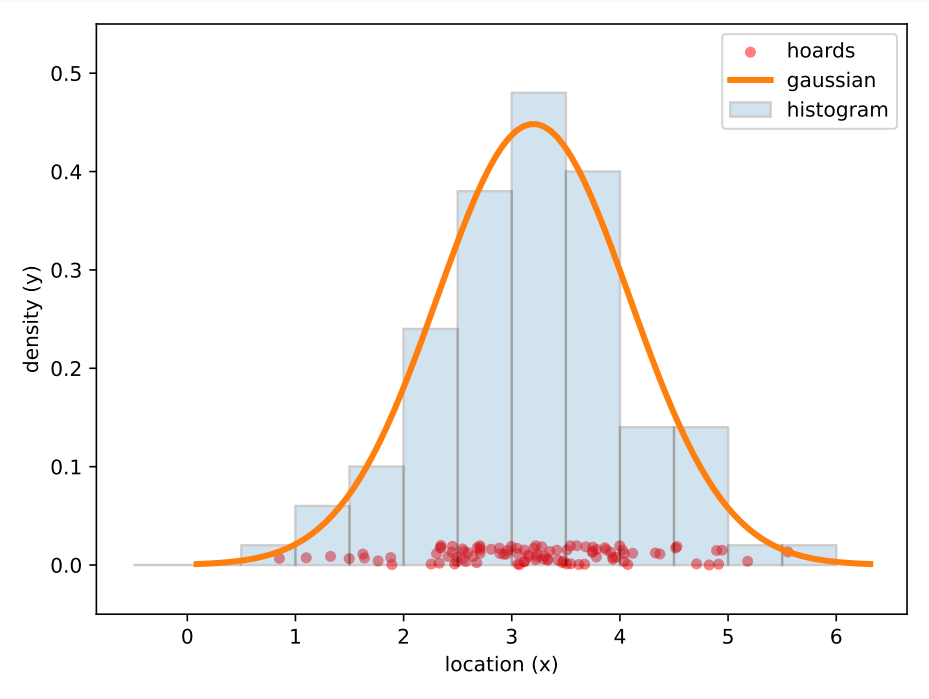



**Figure 10:** A one-dimensional distribution of hoards, plotted as red dots with some vertical jitter to aid visualization.

# Approximating distributions

We want to describe the distribution of points using a density which smooths out some of the detail.

- One option is a **histogram**.
- Another option is a **parametric density**.

For a single clump of points like ours, a sensible choice of **parametric density** is the Gaussian 'blob'

$$f(x; \mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right),$$

where $\mu$ is the centre of the blob and $\sigma$ measures its width.

# Learning parameters

How should we choose $\mu$ and $\sigma$ so $f(x; \mu, \sigma)$ is a good fit to the data?

- We want the density to be large where the datapoints (hoards) are, and small where they aren't.
- Really what we need is a mathematical measure of fit.

A natural first thought is to use the sum of the values of the density function at each of the datapoints:

$$L_{\text{sum}}(\boldsymbol{\theta}) = \sum_{i=1}^{n} f(x_i; \mu, \sigma).$$

Here $\boldsymbol{\theta} = (\mu, \sigma)$ is the **parameter vector**. The bigger that sum, the more concentrated the density is near the hoards. So maybe we should pick a value for $\boldsymbol{\theta}$ that makes $L_{\text{sum}}(\boldsymbol{\theta})$ as big as possible? This turns out not to be a good idea...

# Problems with the sum measure

**Exercise**

Suppose we centre our blob on the first datapoint, so $\boldsymbol{\theta} = (x_1, \sigma)$.

(i) Determine $\lim_{\sigma \to 0} f(x_1; \mu, \sigma)$ when $\mu = x_1$.

(ii) Determine $\lim_{\sigma \to 0} L_{\text{sum}}(\boldsymbol{\theta})$ when $\boldsymbol{\theta} = (x_1, \sigma)$.

It is possible to make $L_{\text{sum}}$ as big as we like by placing the centre of our Gaussian blob on one of the hoards, and then letting $\sigma \to 0$. So if we try to maximise $L_{\text{sum}}$, we will end up with a density that is infinitely concentrated on one datapoint.

# Product Measure

Let's try the product

$$L_{\mathrm{prod}}(\boldsymbol{\theta}) = \prod_{i=1}^{n} f(x_i; \mu, \sigma).$$

**Exercise**

Suppose we centre our blob on the first datapoint, so $\boldsymbol{\theta} = (x_1, \sigma)$.

(i) Determine $\lim_{\sigma \to 0} f(x_i; \mu, \sigma)$ when $\mu = x_1$ and $i \neq 1$.

(ii) Determine $\lim_{\sigma \to 0} L_{\mathrm{prod}}(\boldsymbol{\theta})$ when $\boldsymbol{\theta} = (x_1, \sigma)$.

The product measure is large when the hoard score is concentrated near the hoards, but pays a price for excessive concentration.

# Finding $\hat{\theta}$

We want to find a $\boldsymbol{\theta}$ that **maximises** $L_{\text{prod}}(\boldsymbol{\theta})$.

- Products of many densities tend to be either unmanageably large or unmanageably small.
- We can fix that by working with the natural logarithm of $L_{\text{prod}}(\boldsymbol{\theta})$, written $\ell(\boldsymbol{\theta}) = \log L_{\text{prod}}(\boldsymbol{\theta})$.

Since log is an increasing function, choosing a $\boldsymbol{\theta}$ that maximises $\ell(\boldsymbol{\theta})$ is equivalent to choosing a $\boldsymbol{\theta}$ that maximises $L_{\text{prod}}(\boldsymbol{\theta})$.

**Exercise**

Show that

$$\ell(\boldsymbol{\theta}) = -\frac{1}{2\sigma^2}\sum_{i=1}^{n}(x_i - \mu)^2 - n\log\sigma - \frac{n}{2}\log(2\pi).$$

It is actually possible to obtain exact expressions for the values of $\mu$ and $\sigma$ which maximise $\ell(\boldsymbol{\theta})$. They are simply the sample mean and sample standard deviation of the data,

$$\hat{\mu} = \frac{1}{n}\sum_{i=1}^{n} x_i, \quad \hat{\sigma}^2 = \frac{1}{n}\sum_{i=1}^{n}(x_i - \hat{\mu})^2.$$

**Exercise**

The maximum of $\ell(\boldsymbol{\theta})$ can be found by looking for its stationary point, which solves

$$\frac{\partial \ell}{\partial \mu} = 0, \quad \frac{\partial \ell}{\partial \sigma} = 0.$$

By computing these derivatives, show that $\ell(\boldsymbol{\theta})$ is maximized when $\mu$ and $\sigma$ are the sample mean and sample standard deviation of the data.

It is time to return to the Queen's treasure hoards. We'll start by generalising our **Gaussian blob** to two dimensions.

- In two dimensions a circularly-symmetric Gaussian blob centred at the coordinate vector $\boldsymbol{\mu} = (\mu_x, \mu_y)$ is

$$f(x, y) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{(x-\mu_x)^2 + (y-\mu_y)^2}{2\sigma^2}\right).$$

- If we use vector notation, we can write this as

$$f(\mathbf{x}) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{|\mathbf{x}-\boldsymbol{\mu}|^2}{2\sigma^2}\right).$$

  The factor outside the exponential ensures that the density integrates to one.

A single blob is not going to be sufficient to describe the distribution of hoard finds. Let's try a **mixture** of three blobs

$$f(\boldsymbol{x};\boldsymbol{\theta}) = \sum_{i=1}^{3} \frac{w_i}{2\pi\sigma_i^2} \exp\left(-\frac{|\boldsymbol{x}-\boldsymbol{\mu}_i|^2}{2\sigma_i^2}\right).$$

Vectors $\boldsymbol{\mu}_1$, $\boldsymbol{\mu}_2$ and $\boldsymbol{\mu}_3$ are the blob centres, scalars $\sigma_1, \sigma_2, \sigma_3 > 0$ are their spatial sizes, and scalars $w_1, w_2, w_3 \geq 0$ are the weights in the weighted sum which satisfy

$$w_1 + w_2 + w_3 = 1,$$

making $f$ a density called a **mixture model**.

To condense our notation we bundle up all the parameters — here $\boldsymbol{\mu}_1$, $\boldsymbol{\mu}_2$, $\boldsymbol{\mu}_3$, $\sigma_1$, $\sigma_2$, $\sigma_3$, $w_1$ and $w_2$ — into a single vector, $\boldsymbol{\theta}$.

# Finding $\hat{\theta}$

We need to adjust $\boldsymbol{\theta}$ to find a hoard score function that fits $D$ well.

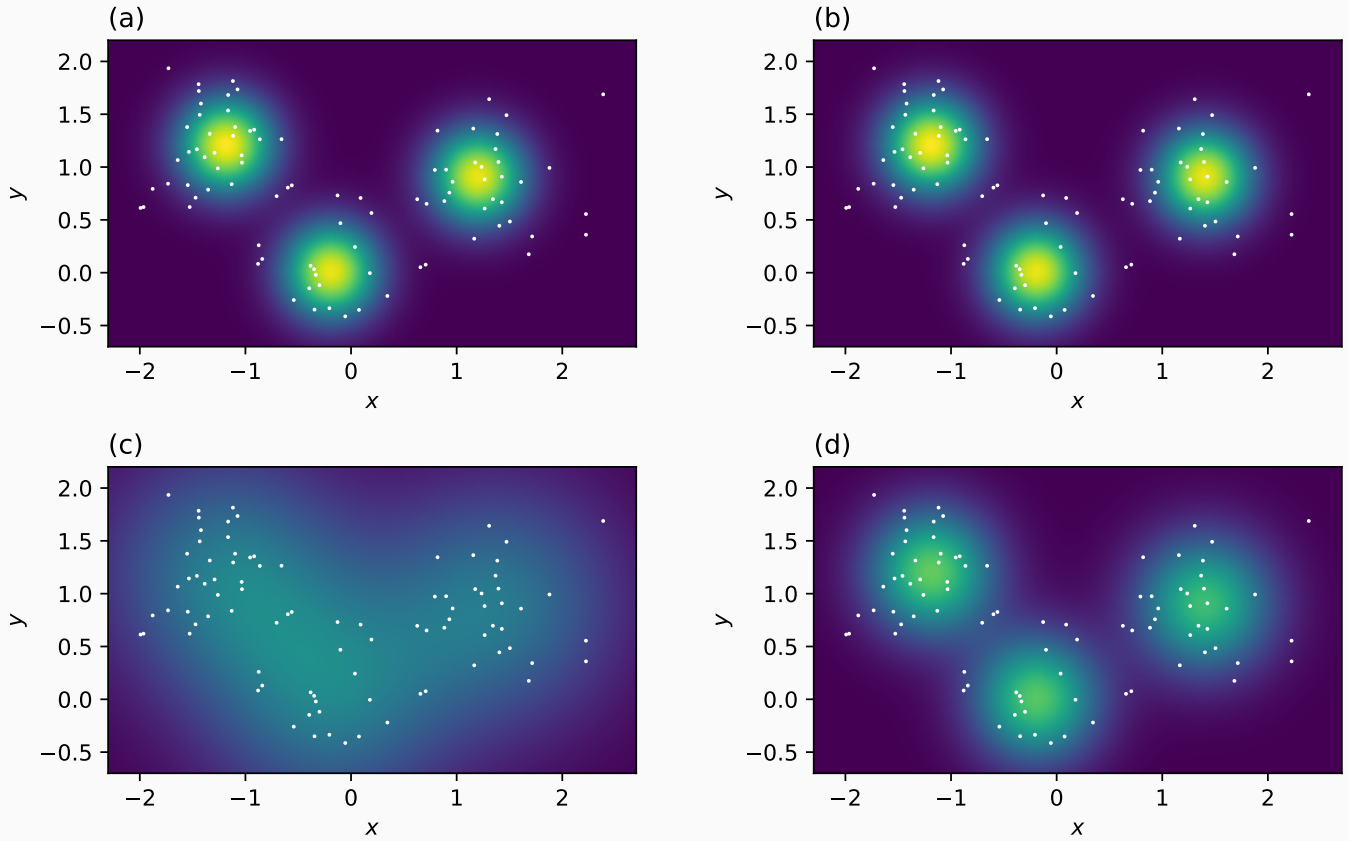



**Figure 11:** Dataset $D$, with overlaid heat maps of some trial-and-error 'fits' of the three-component mixture model. Is it the best we can do?

- Inspired by our attempts to fit a single blob to a cluster of points in one dimension, let us try using the same log-product measure of fit,

$$\ell(\boldsymbol{\theta}) = \sum_{\mathsf{x} \in D} \log f(\mathsf{x}; \boldsymbol{\theta}).$$

- Our task is to choose $\boldsymbol{\theta}$ so as to make $\ell(\boldsymbol{\theta})$ as large as possible.
- This is considerably more difficult than the single-blob version we solved above.
- There is no *analytical* solution. We must use computational techniques (we return to the details later in the course).

# Three component results

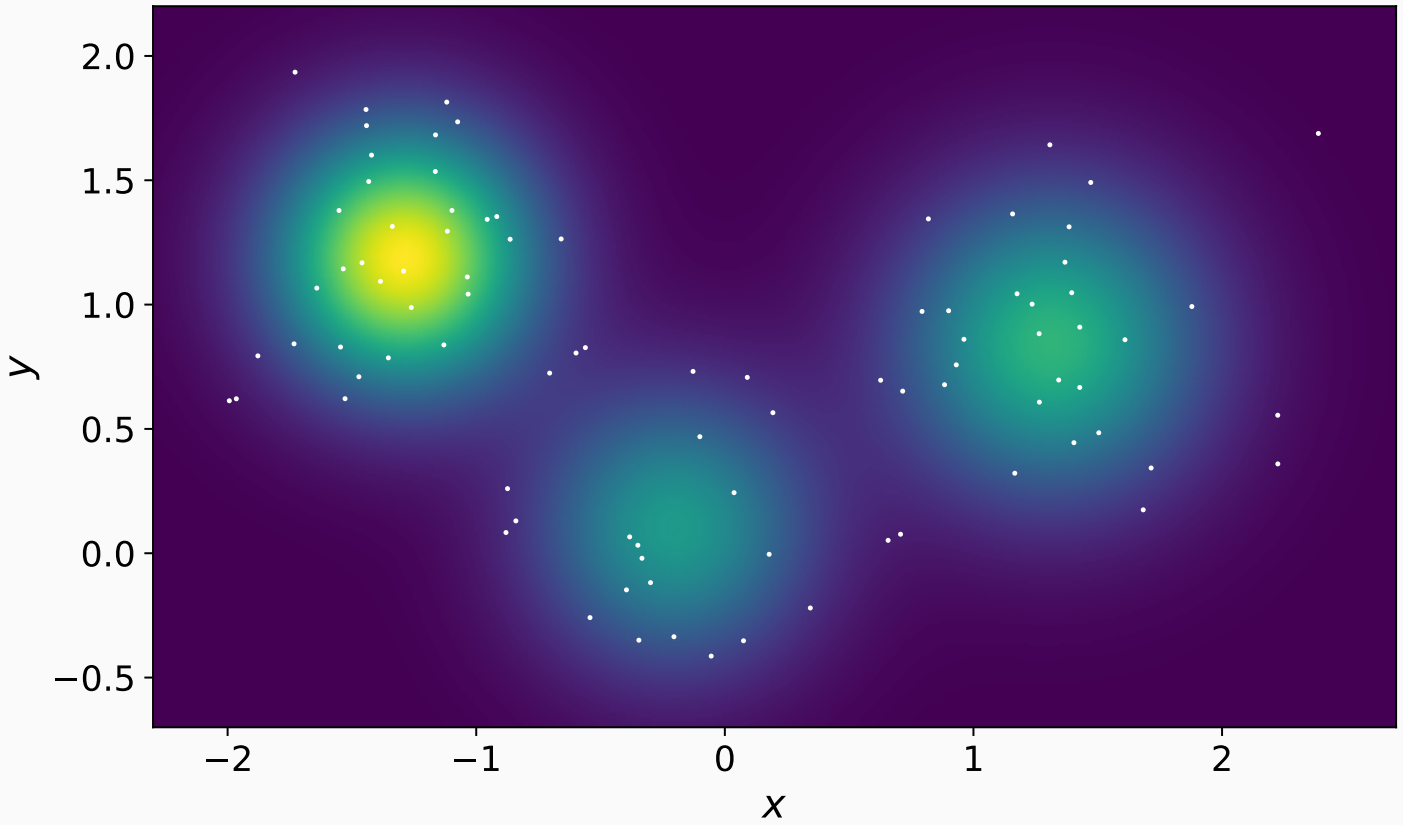



**Figure 12:** Heat map of optimised three component hoard score function

# More components?

- The algorithmic fitting procedure seems preferable to the trial-and-error method.
- We are still relying on our intuitive visual judgement that there are three clusters in the data.
- Should we be fitting a hoard score function with two, or four, or even ten Gaussian blobs?

The $k$-component mixture density is

$$f_k(\boldsymbol{x}; \boldsymbol{\theta}_k) = \sum_{i=1}^{k} \frac{w_i}{2\pi\sigma_i^2} \exp\left(-\frac{|\boldsymbol{x} - \boldsymbol{\mu}_i|^2}{2\sigma_i^2}\right).$$

Larger $k$ implies a **more flexible** model.

# Estimating $\theta_k$.

We define the log-product measure of fit for the $k$-component model:

$$\ell_k(\boldsymbol{\theta}_k) = \sum_{i=1}^{n} \log f_k(\boldsymbol{x}_i; \boldsymbol{\theta}_k).$$

Our parameter vector estimate $\hat{\boldsymbol{\theta}}_k$ is a maximiser of this function.

By increasing $k$ we will increase $\ell_k(\hat{\boldsymbol{\theta}}_k)$.

We also define $\hat{\boldsymbol{\theta}}_{k,-i}$ to be the parameter vector estimate we obtain when we **leave out** point $\boldsymbol{x}_i$ from $D$. The cross validated product measure is

$$\hat{\ell}_k^{\mathrm{CV}} = \sum_{i=1}^{n} \log f_k(\boldsymbol{x}_i; \boldsymbol{\theta}_{k,-i}).$$

# Examples

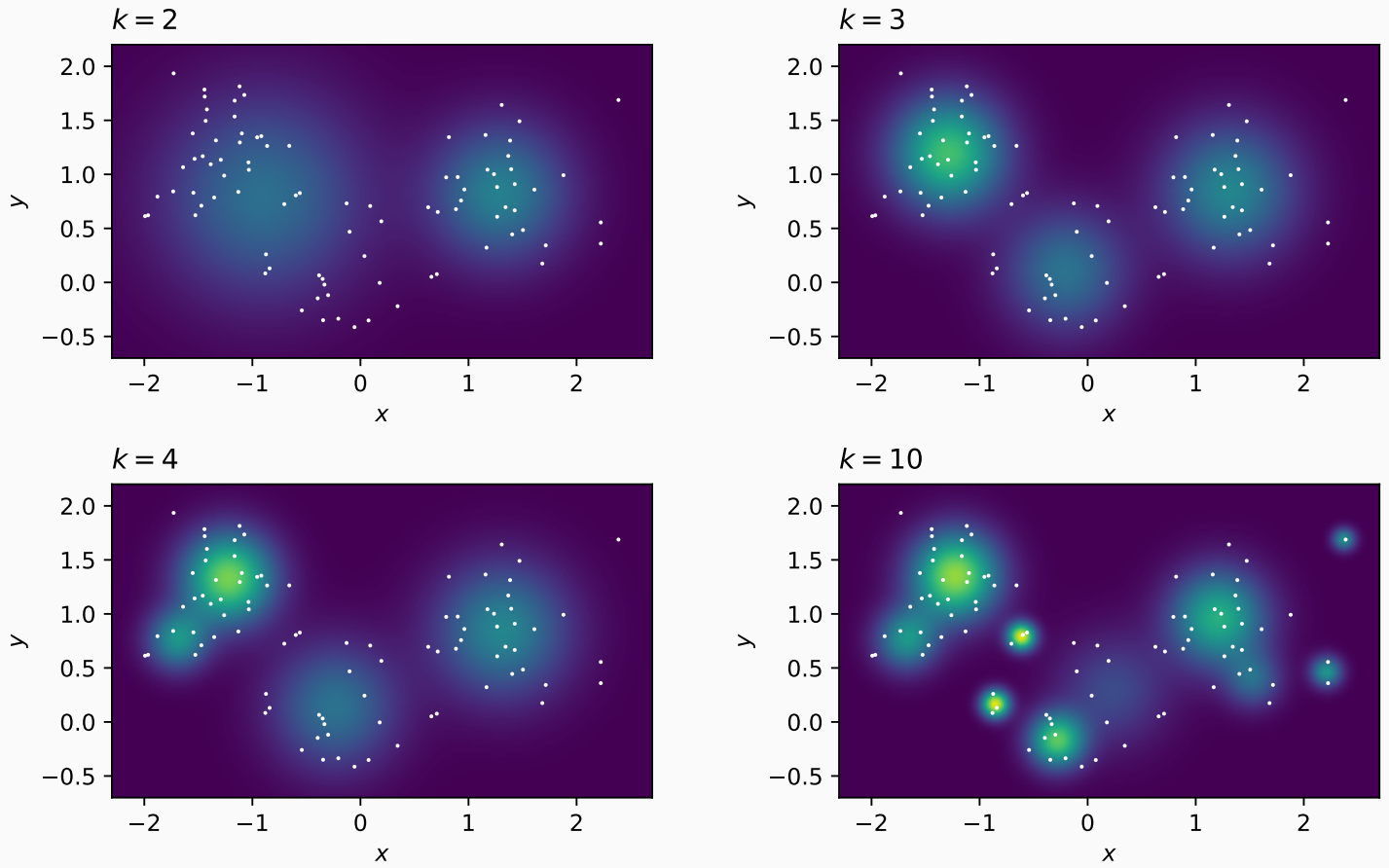



**Figure 13:** Dataset $D$ plotted four times, with overlaid heat maps of optimised $k$-component hoard score functions. The $k = 3$ panel looks slightly different from Figure 12 because the color scheme here is adapted to fit the range of hoard score values across the four panels.

# Choosing $k$

- $k$ measures the **flexibility** of the model
- Large $k$ leads to **overfitting**
- The cross validated log-product approximates performance on **unseen** data. $k = 3$ appears to be a good choice.

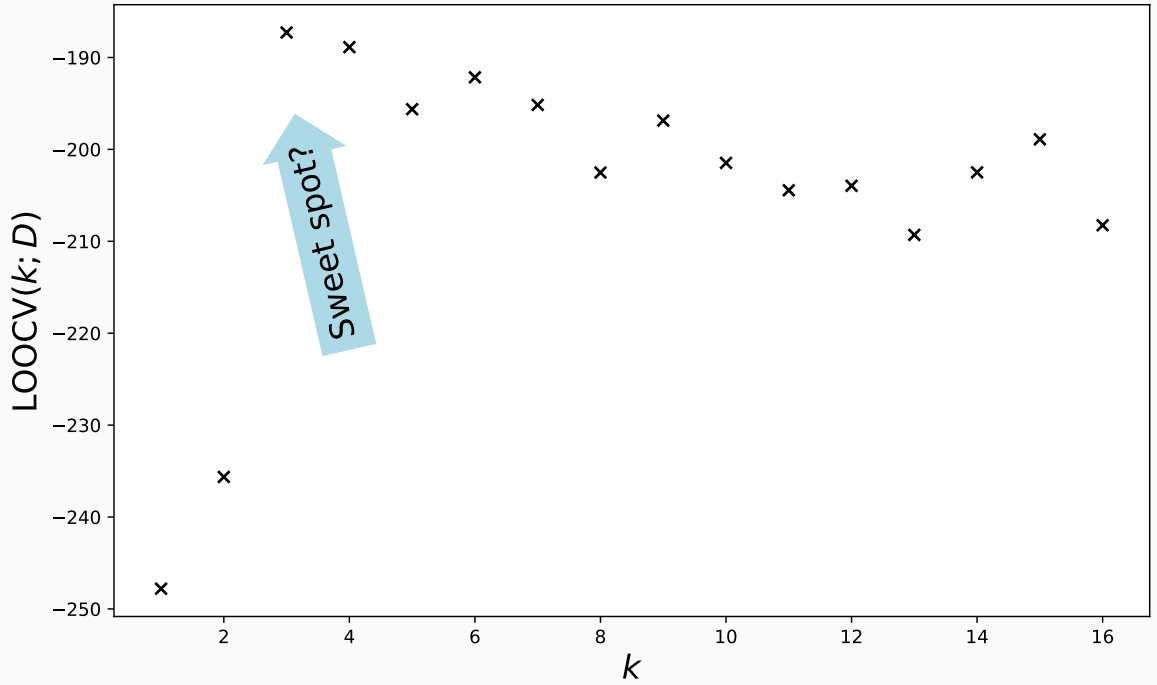



**Figure 14:** Leave-one-out cross validation score plotted against number of blobs, $k$.

# Conclusions

1. Within a function family, we found the best-fitting distribution by optimising a quantitative measure of fit: the logarithm of the product of the distribution function values at each data point.
2. This procedure led to **overfitting** when the function family was too **flexible**, and **underfitting** when it was not flexible enough.
3. We used **cross-validation** to find a sweet spot — a function family that was just flexible enough.

# Probabilistic Modelling

# Prediction functions and scientific hypotheses

Given a dataset $D = \{(x_i, y_i)\}_{i=1}^n$, imagine that you have picked what you judge is the best prediction function, $\hat{g}$, so that given $x$ you predict $y$ to be

$$\hat{y} = \hat{g}(x).$$

- Have you just endorsed a **scientific hypothesis**?
- Certainly you don't believe every future observation will be of the form $(x, \hat{g}(x))$, lying precisely on your prediction curve.
- Perhaps you think future observations will lie near that curve; but 'near' is **vague**.
- Scientific hypotheses should be sufficiently **clear-cut** that we can understand what it would mean for them to be **true** or **false**.

# Probabilistic hypotheses

A scientific hypothesis should have the following features.

1. It should be **genuinely predictive**, so it can't just be a claim about the currently available data.
2. It should **allow for uncertainty**: you think $x$ fixes $y$ roughly, but not exactly.
3. It shouldn't be vague: it should be a claim with **well-defined truth conditions**.

The second and third requirements look like they might be in tension, but they can both be satisfied if our hypothesis is a *probabilistic* claim about the *process that generated the data*.

# Data generating process

- When we fit a model, we use a single dataset, $D$.
- The mechanism that produced $D$ could, in principle, be re-run to obtain a sequence of datasets $D_1, D_2, D_3, \ldots$
- A **probabilistic model** capable of generating such sequences represents a **hypothesis** about the nature of the mechanism.

**Example**

Consider the experiment of repeatedly tossing a coin. Suppose we hypothesise that toss produces H or T with equal probability. This hypothesis has a well defined truth condition (the fraction of heads should converge to 0.5) but still allows for uncertainty at the level of individual observations.

# Defining a probabilistic model

Consider the supervised learning problem with dataset

$$D = \{(x_i, y_i)\}_{i=1}^{n}.$$

Each datapoint $(x_i, y_i)$ is the realisation of a **random vector** $(X, Y)$, whose components $X$ and $Y$ are **random variables**.

Suppose that the 'physics' of the situation is such that $Y$ is equal to a function of $X$ plus some noise,

$$Y = \mathfrak{g}(X) + \mathcal{E}^{\text{noise}}.$$

Here $\mathcal{E}^{\text{noise}}$ is a random variable whose **conditional expectation** $\mathbb{E}(\mathcal{E}^{\text{noise}}|X = x)$ is zero for every $x$. The expected value of $Y$ given any particular value $x$ of $X$ is therefore just $\mathfrak{g}(x)$.

The function $\mathfrak{g}$ represents the true systematic dependence of $Y$ on $X$; $\mathcal{E}^{\text{noise}}$ represents the true noise.

To learn an approximate model of $\mathfrak{g}$ and $\mathcal{E}^{\text{noise}}$ we assume a **conditional probabilistic model**

$$Y = g(X; \boldsymbol{\theta}) + \mathcal{E}_{\boldsymbol{\theta}}.$$

- $\boldsymbol{\theta}$ is a parameter vector, learned from data.
- $\mathbb{E}(\mathcal{E}_{\boldsymbol{\theta}}) = 0$ and $\mathcal{E}_{\boldsymbol{\theta}} \perp\!\!\!\perp X$.
- The model specifies a distribution of $Y$ **conditional** on $X$.
- Given $X$ we can predict $Y$ to be its **expected value**:

$$\hat{y} = \mathbb{E}(Y|X) = g(x; \hat{\boldsymbol{\theta}}) = \underbrace{\hat{g}(x)}_{\substack{\text{prediction} \\ \text{function}}}$$

# Example

An example dataset $D = \{(x_k, y_k)\}_{k=1}^{10}$ is shown in Figure 15.

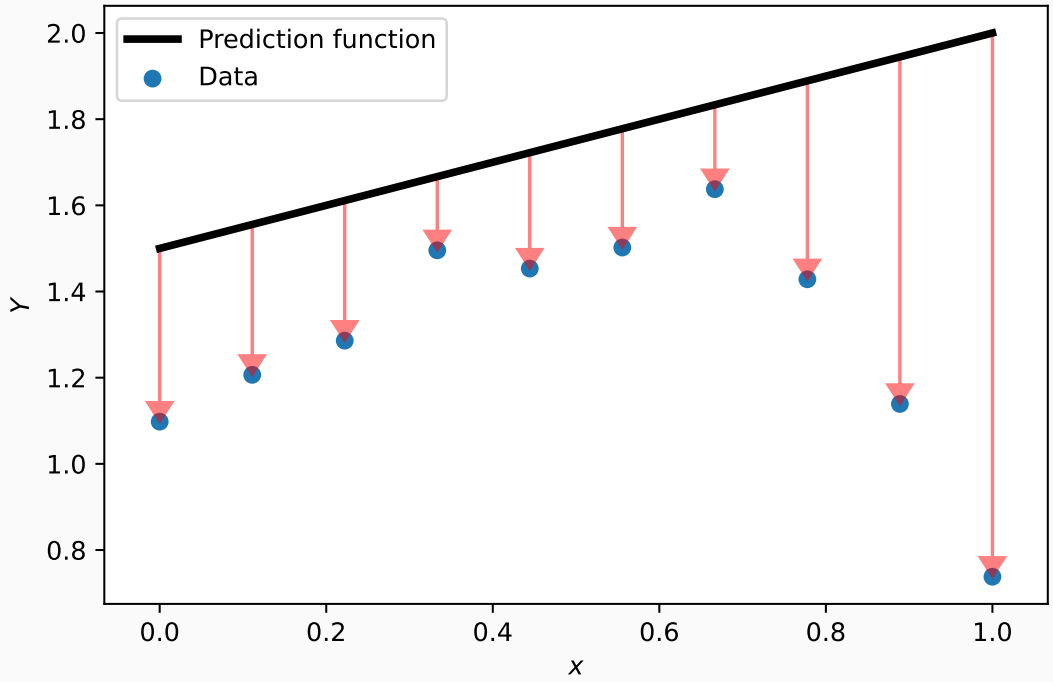



**Figure 15:** Blue dots show some observational data. Black line shows a (bad) linear predictive model. Red arrows show residuals.

# Defining a model

The $y_k$ in $D$ were artificially generated using a model of the form

$$Y = \mathfrak{g}(X) + \mathcal{E}^{\text{noise}}, \tag{1}$$

where $\mathfrak{g}$ is an unknown-to-us function and $\mathcal{E}$ is a zero-mean normal random variable. The $x_k$ are equally spaced in $[0, 1]$. Since our task is predicting $y$ given $x$, we don't care about the process that generated the $x$ values.

Let's begin by trying to fit a *very* simple conditional probabilistic model,

$$Y = \beta_0 + \beta_1 X + \mathcal{E},$$
$$\mathcal{E} \sim \mathcal{N}(0, \sigma^2).$$

Here $\mathcal{E} \sim \mathcal{N}(0, \sigma^2)$ means '$\mathcal{E}$ is **normally distributed** with mean zero and variance $\sigma^2$'. The model has parameter vector $\boldsymbol{\theta} = (\beta_0, \beta_1, \sigma)$.

# Fitting a model

- We want to assess the quality of fit achieved by a particular $\boldsymbol{\theta}$.
- Consider the hypothesis that the observed $y$ values were generated from the observed $x$ values by the model.
- Let us calculate the probability of this hypothesis.

**Exercise**

Show that our model implies that the conditional density of $Y$ given $X = x$ is

$$f_{Y|X}(y|x;\boldsymbol{\theta}) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(y-\beta_0-\beta_1 x)^2}{2\sigma^2}\right).$$

The probability density of a single response $y_k$ given a predictor $x_k$ is then $f_{Y|X}(y_k|x_k;\boldsymbol{\theta})$.

# Conditional probability density of $D$

Now consider a sequence of $n$ predictor-response observations

$$(x_1, y_1), (x_2, y_2), \ldots (x_n, y_n).$$

We write the collected predictor and response values as vectors

$$\underline{x} = (x_1, x_2, \ldots, x_n)$$
$$\underline{y} = (y_1, y_2, \ldots, y_n)$$

which are viewed as realisations of random vectors $\underline{X} = (X_1, \ldots, X_n)$ and $\underline{Y} = (Y_1, \ldots, Y_n)$.

The dataset may be viewed as a single realisation of $\underline{X}$, followed by a realisation of $\underline{Y}$ where $Y_k = \beta_0 + \beta_1 X_k + \mathcal{E}_k$. Crucially, the realisations of the noise are assumed to be **independent**.

Due to the independence of the noise variables, the $Y_k$ are independent given $\underline{X}$, written $Y_i \perp\!\!\!\perp Y_j | \underline{X}$, $\forall i \neq j$.

Therefore

$$\begin{aligned} f_{\underline{Y}|\underline{X}}(\underline{y}|\underline{x};\boldsymbol{\theta}) &= f_{Y|X}(y_1|x_1;\boldsymbol{\theta}) \times f_{Y|X}(y_2|x_2;\boldsymbol{\theta}) \times \ldots \times f_{Y|X}(y_n|x_n;\boldsymbol{\theta}) \\ &= \prod_{k=1}^{n} f_{Y|X}(y_k|x_k;\boldsymbol{\theta}). \end{aligned}$$

Viewed as a function of $\boldsymbol{\theta}$, this is the *likelihood function*. It is often written as $\mathcal{L}(\boldsymbol{\theta})$. The likelihood tends to be a very small or very large number, so we work with its logarithm

$$\ell(\boldsymbol{\theta}) = \log \mathcal{L}(\boldsymbol{\theta}) = \log f_{\underline{Y}|\underline{X}}(\underline{y}|\underline{x};\boldsymbol{\theta}) = \sum_{k=1}^{n} \log f_{Y|X}(y_k|x_k;\boldsymbol{\theta}).$$

# Calculating $\ell(\boldsymbol{\theta})$

## Exercise

Let $a_1, a_2, \ldots, a_n$ be a sequence of positive numbers.

(a) Use summation notation to write the 'log product' $\log\left(\prod_{i=1}^{n} a_i\right)$ in terms of $\log(a_1), \log(a_2), \ldots, \log(a_n)$.

(b) Write the log product

$$\log\left(\prod_{i=1}^{n} \frac{e^{\frac{a_i}{b}}}{c}\right)$$

in terms of $b, c, n$ and the sum $S = \sum_{i=1}^{n} a_i$.

(c) Show that

$$\ell(\boldsymbol{\theta}) = -\frac{1}{2\sigma^2}\sum_{k=1}^{n}(y_k - \beta_0 - \beta_1 x_k)^2 - n\log(\sqrt{2\pi}\sigma),$$

# Maximum likelihood

The likelihood is the probability density of the data, given the model with parameter vector $\boldsymbol{\theta}$. Choosing $\boldsymbol{\theta}$ to maximise $\mathcal{L}(\boldsymbol{\theta})$ is the...

**Principle of maximum likelihood**

Given a probabilistic model of your data with parameter vector $\boldsymbol{\theta} = (\theta_0, \theta_1, \ldots, \theta_m)$, the **likelihood function** is

$$\mathcal{L}(\boldsymbol{\theta}) = \prod_{k=1}^{n} f_{Y|X}(y_k|x_k; \boldsymbol{\theta}).$$

The *log-likelihood* is $\ell(\boldsymbol{\theta}) = \log \mathcal{L}(\boldsymbol{\theta})$. The principle of maximum likelihood tells us to choose the model parameters so as to maximize the likelihood, i.e. to pick parameter vector

$$\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}) = \arg\max_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta}).$$

# Applying the principle

Maximum likelihood for our linear model yields

$$\hat{g}(x) = \hat{\beta}_0 + \hat{\beta}_1 x = 1.36 - 0.14x,$$

shown in Figure 16. We also find $\hat{\sigma} = 0.25$.

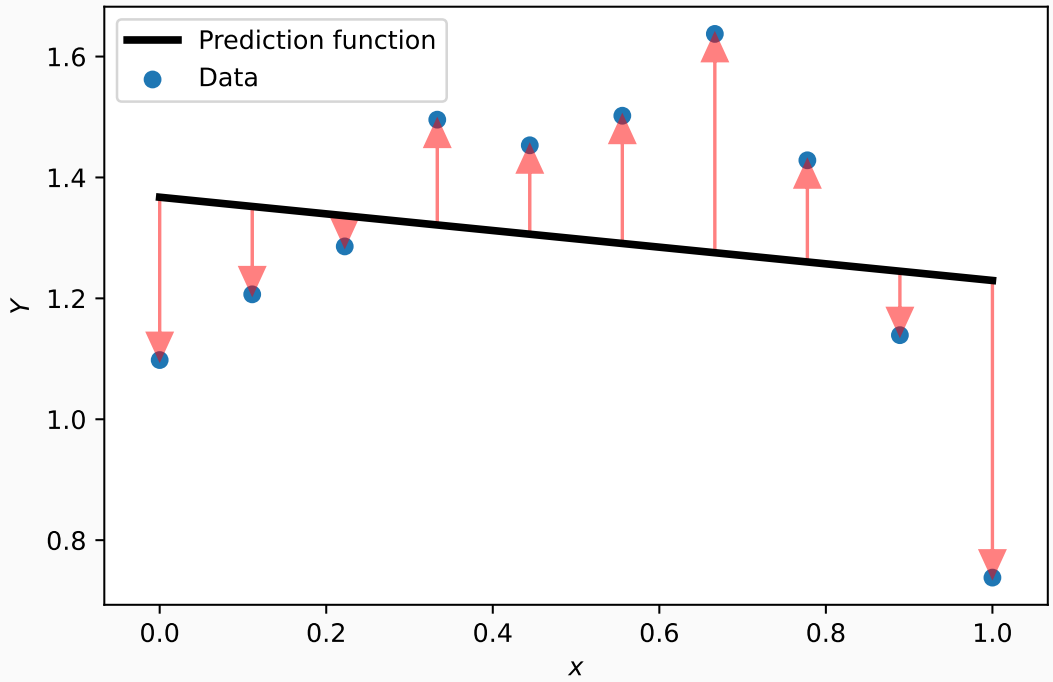



**Figure 16:** Maximum likelihood linear model

# Quadratic model $Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \mathcal{E}$

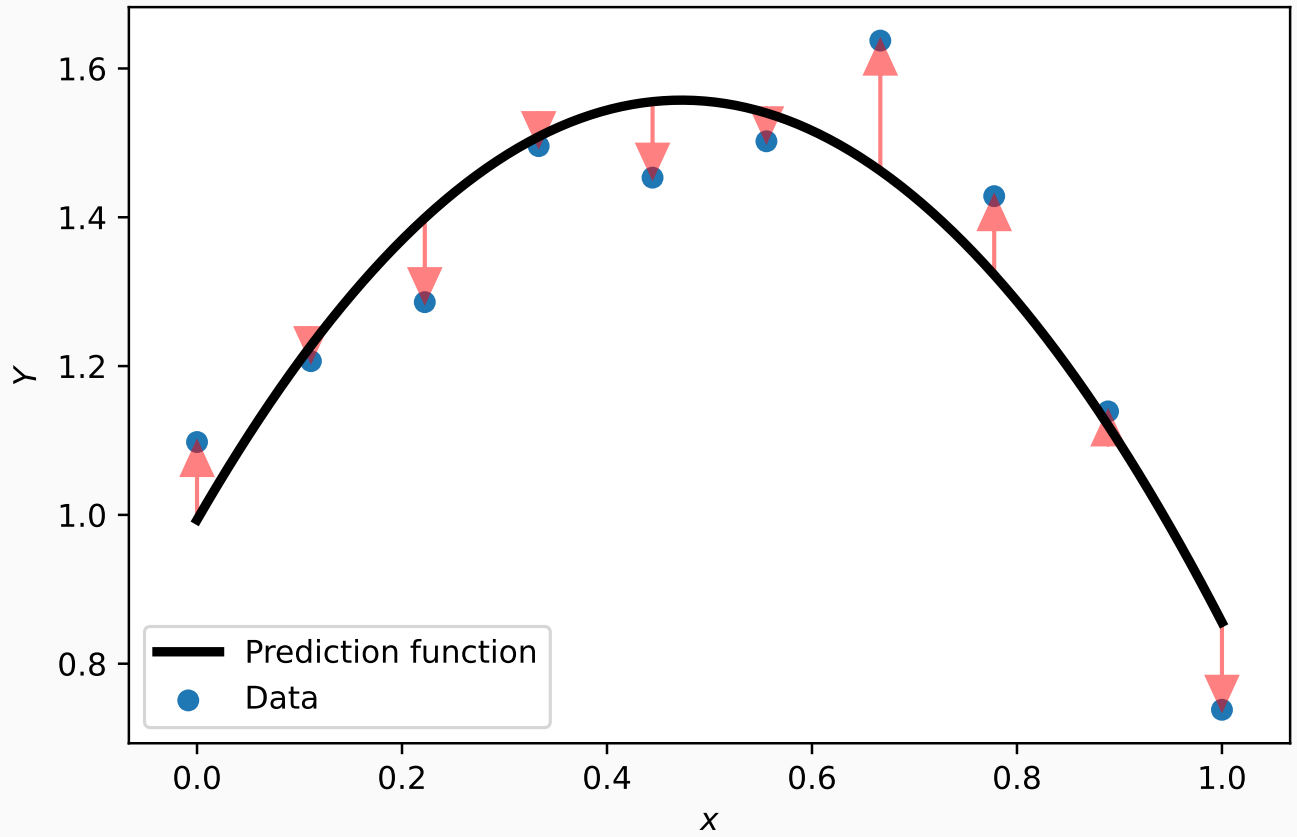



**Figure 17:** Maximum likelihood quadratic model with $\hat{\beta}_0 = 0.99$, $\hat{\beta}_1 = 2.38$, $\beta_2 = -2.52$. We have dramatically reduced the residuals, and consequently reduced our estimate of the noise magnitude to $\hat{\sigma} = 0.1$.

# Maximum likelihood and least squares

Consider a model where $g$ is allowed to depend on $m$ parameters, which we write as a vector $\boldsymbol{\beta} = (\beta_0, \ldots, \beta_m)$, so

$$Y = g(X; \boldsymbol{\beta}) + \mathcal{E}$$

where $\mathcal{E} \sim \mathcal{N}(0, \sigma^2)$.

**Exercise**

Given a data set $D = \{(x, y_i)\}_{i=1}^n$, show that the the log-likelihood is

$$\begin{aligned} \ell(\boldsymbol{\theta}) &= \sum_{k=1}^{n} \log f_{Y|X}(y_k | x_k; \boldsymbol{\theta}) \\ &= -\frac{1}{2\sigma^2} \sum_{k=1}^{n} (y_k - g(x_k; \boldsymbol{\beta}))^2 - n \log(\sqrt{2\pi}\sigma). \end{aligned}$$

For given $\sigma$, maximising the likelihood with respect to $\boldsymbol{\beta}$ requires us to maximize

$$-\sum_{k=1}^{n}(y_k - g(x_k;\boldsymbol{\beta}))^2 = -\mathrm{RSS}(\boldsymbol{\beta}).$$

Applying the principle of maximum likelihood we have

$$\hat{\boldsymbol{\beta}} = \arg\max_{\boldsymbol{\beta}} \left( -\sum_{k=1}^{n}(y_k - g(x_k;\boldsymbol{\beta}))^2 \right) = \arg\min_{\boldsymbol{\beta}} \mathrm{RSS}(\boldsymbol{\beta}).$$

Therefore, the maximum likelihood estimates coincide with the minimum RSS estimates used in least squares.

# Example

## Exercise

Consider the the following dataset collected by observing the pair of random variables $(X, Y)$ four times.

| $X$ | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| $Y$ | 0.5 | 4.8 | 8.3 | 17.5 |

Consider a conditional probabilistic model $Y = \beta X^2 + \mathcal{E}$ where $\mathcal{E} \sim \mathcal{N}(0, \sigma^2)$, and $\mathcal{E}$ and $X$ are independent.

(a) What is the parameter vector $\boldsymbol{\theta}$ of this model?

(b) Find the log likelihood $\ell(\boldsymbol{\theta})$ given arbitrary dataset $D = \{(x_i, y_i)\}_{i=1}^n$, according to this model.

(c) Obtain the maximum likelihood estimate $\hat{\beta}$ of $\beta$ in terms of $D$.

(d) Calculate $\hat{\beta}$ using the dataset given in the table above. Plot both the data and the curve $y = \hat{\beta}x^2$ on the same pair of axes.

# Maximum likelihood and overfitting

- Consider a sequence of increasingly flexible models, with the $p$th model in the sequence taking the form

$$Y = \sum_{i=0}^{p} \beta_i X^i + \mathcal{E},$$

  where $\mathcal{E} \sim \mathcal{N}(0, \sigma^2)$ and $p$ is the polynomial order of the model.
- Having estimated the model parameters $\boldsymbol{\theta} = (\beta_0, \ldots, \beta_p, \sigma)$ by maximum likelihood, we can evaluate the log-likelihood $\ell(\hat{\boldsymbol{\theta}})$.
- This gives the log-probability density of the data according to the fitted model.

# Observations

- As $p$ increases, so does the probability density of the data, $\ell(\hat{\boldsymbol{\theta}})$, according to our fitted model.
- Our estimate of $\sigma$ gets smaller, meaning that we are allowing the systematic component to take over as the explanatory mechanism for patterns in the data. We are **overfitting**.
- It is always possible to find $\boldsymbol{\theta}$ such that the conditional probability density function $f_{Y|X}(y|x;\boldsymbol{\theta})$ achieves very large values when evaluated on individual datapoints in $D$.
- The problem is that when a new point $(x_0, y_0)$ is observed, the model will likely assign it a very low probability density.
- If we were to attempt to predict $y_0$ from $x_0$, such a model would do a poor job because the conditional density function $f_{Y|X}(-|x_0;\boldsymbol{\theta})$ would be sharply peaked in the wrong place.

# Cross validation

Let $\hat{g}_{-i}(x)$ be the prediction function fitted to $D_{-i} = D \setminus (x_i, y_i)$, and define

$$\mathrm{MSE}^{\mathrm{CV}} = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{g}_{-i}(x_i))^2.$$

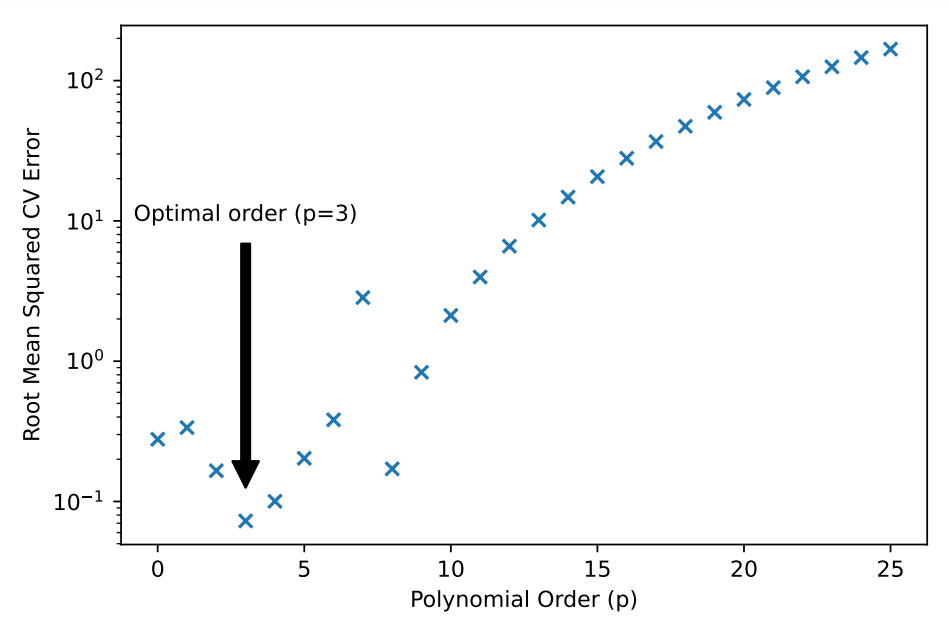



**Figure 18:** CV errors in maximum likelihood polynomial prediction functions.

# Unsupervised case

- A **conditional** probabilistic model approximates the density of $Y$ given $X$, written $f_{Y|X}(y|x)$.
- This can be used to **predict** $Y$ from $X$ — **supervised learning**.
- But suppose we want to model the mechanism which generated a dataset $D = \{\boldsymbol{x}_i\}_{i=1}^n$ of observations of a random vector $\boldsymbol{X}$ (e.g. the burial location of the next hoard).
- A probabilistic model of the mechanism approximates the density $f_X(\boldsymbol{x})$.
- Given a model $f_X(\boldsymbol{x}; \boldsymbol{\theta})$ of the density, the log-likelihood is

$$\ell(\boldsymbol{\theta}) = \log \mathcal{L}(\boldsymbol{\theta}) = \log \prod_{i=1}^{n} f_X(\boldsymbol{x}_i; \boldsymbol{\theta}) = \sum_{i=1}^{n} \log f_X(\boldsymbol{x}_i; \boldsymbol{\theta})$$

- The maximum likelihood principle is $\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta})$.

# Conclusions

- A conditional probabilistic model describes how the conditional probability density of a response variable given a predictor variable depends on the value of the predictor.
- The form of the dependence is controlled by a parameter vector $\boldsymbol{\theta}$. The space of possible values of $\boldsymbol{\theta}$ is the model's hypothesis space.
- To fit a model to data, one must make a data-driven choice of a specific hypothesis in the model's hypothesis space. In other words, one must pick a specific value of $\boldsymbol{\theta}$.
- The principle of maximum likelihood is one way of doing this.
- For a wide class of conditional probabilistic models, maximising likelihood is equivalent to minimising RSS.

# Frequentist and Bayesian Uncertainty

# Frequentist probability

Frequentist probability is defined using the idea of a *repeatable experiment*. The set of possible outcomes is the **sample space** $\Omega$. Subsets of $\Omega$ are called **events**.

**Example**

Suppose we perform the experiment of tossing a coin three times. $\Omega = \{HHH, HHT, HTH, HTT, THH, THT, TTH, TTT\}$. The event, $A$, of getting exactly two tails is $A = \{HTT, THT, TTH\}$.

**Exercise**

Suppose we throw a four-sided (tetrahedral) dice twice.

(a) Write down $\Omega$.

(b) Write down the event $A$ of getting a total score of more than 5.

A **probability measure Pr** is a function which maps events to numbers in $[0, 1]$, called their *probabilities*.

In the frequentist world, this function is defined as follows.

**Frequentist definition of probability**

Suppose we repeat an experiment $n$ times. Let $n(A)$ be the number of times that event $A$ occurs. According to a frequentist,

$$\Pr(A) = \lim_{n \to \infty} \frac{n(A)}{n}.$$

If $B$ is also an event then $A \cap B$ is the event that both $A$ and $B$ occur. The frequentist definition of conditional probability is

$$\Pr(A|B) = \lim_{n \to \infty} \frac{n(A \cap B)}{n(B)} = \frac{\Pr(A \cap B)}{\Pr(B)}.$$

If $A \perp\!\!\!\perp B$ then $\Pr(A|B) = \Pr(A)$ so $\Pr(A \cap B) = \Pr(A)\Pr(B)$.

# Lizard Sex Ratios

- The ratio of males to females in most animal species is approximately 1:1
- **Fisher's principle** provides an explanation: deviations from an equal ratio confer a **reproductive advantage** on the minority sex.
- Genes which cause individuals to produce more offspring of the minority sex will therefore be reproduced more effectively, equalising the ratio.
- Deviation from an equal sex ratio indicates the presence of evolutionary forces beyond Fisher's principle.
- We will address the problem of estimating the birth sex ratio in a population of lizards living in Tasmania.

# Observational data

Over a sixteen year period, researchers observed the sexes of hundreds of newborn lizards.

- Define the random variable

$$X_k = \begin{cases} 1 & \text{if the } k\text{th lizard is male,} \\ 0 & \text{if it is female.} \end{cases}$$

- The researchers observed the sequence $X_1, X_2, X_3, \ldots$ obtaining $D = \{x_1, x_2, x_3, \ldots\}$.

Our aim is to *infer* the sex ratio based on these observations.

# Probability model

We will assume the $X_k$s are *independent and identically distributed* (i.i.d.) with mass function

$$\Pr(X_k = x; p) = \begin{cases} p & \text{if } x = 1, \\ 1 - p & \text{if } x = 0, \end{cases}$$

where $p \in [0, 1]$. We imagine that there is a true value $p^*$ of the parameter – the true probability that any particular lizard is male.

Each $X_k$ is a *Bernoulli random variable* with parameter $p$. We write this $X_k \sim \text{Bernoulli}(p)$.

**Exercise**

Show that we can rewrite the Bernoulli probability mass function as

$$f_X(x; p) = p^x (1 - p)^{1-x}, \quad x \in \{0, 1\}.$$

# Likelihood function

Because the $X_i$s are i.i.d. we can write the joint probability mass function of the whole sequence $\underline{X} = (X_1, X_2, \ldots, X_n)$ as

$$f_{\underline{X}}(\underline{x}; p) = \prod_{i=1}^{n} f_X(x_i; p).$$

Our data $D$ is a single realisation $\underline{x}$ of $\underline{X}$.

The probability of the data given the model is the *likelihood function*

$$\mathcal{L}(p) = f_{\underline{X}}(\underline{x}; p) = \prod_{i=1}^{n} f_X(x_i; p).$$

The maximum likelihood estimator of $p$ is

$$\hat{p} = \arg\max_{p} \mathcal{L}(p).$$

**Exercise**

(a) Let $s_n = x_1 + x_2 + \ldots + x_n$. Show that

$$\mathcal{L}(p) = p^{s_n}(1-p)^{n-s_n}.$$

(b) Let $\ell(p) = \log \mathcal{L}(p)$. Show that

$$\ell(p) = \log \mathcal{L}(p) = s_n \log p + (n - s_n) \log(1-p).$$

(c) The maximum likelihood occurs where $\ell'(p) = 0$. Using this condition, show that

$$\hat{p}_n^{\mathrm{ML}} = \frac{s_n}{n}.$$

Our estimate for the probability of producing a male lizard is just the fraction of the lizards observed so far that are male.

# Sequence of estimates

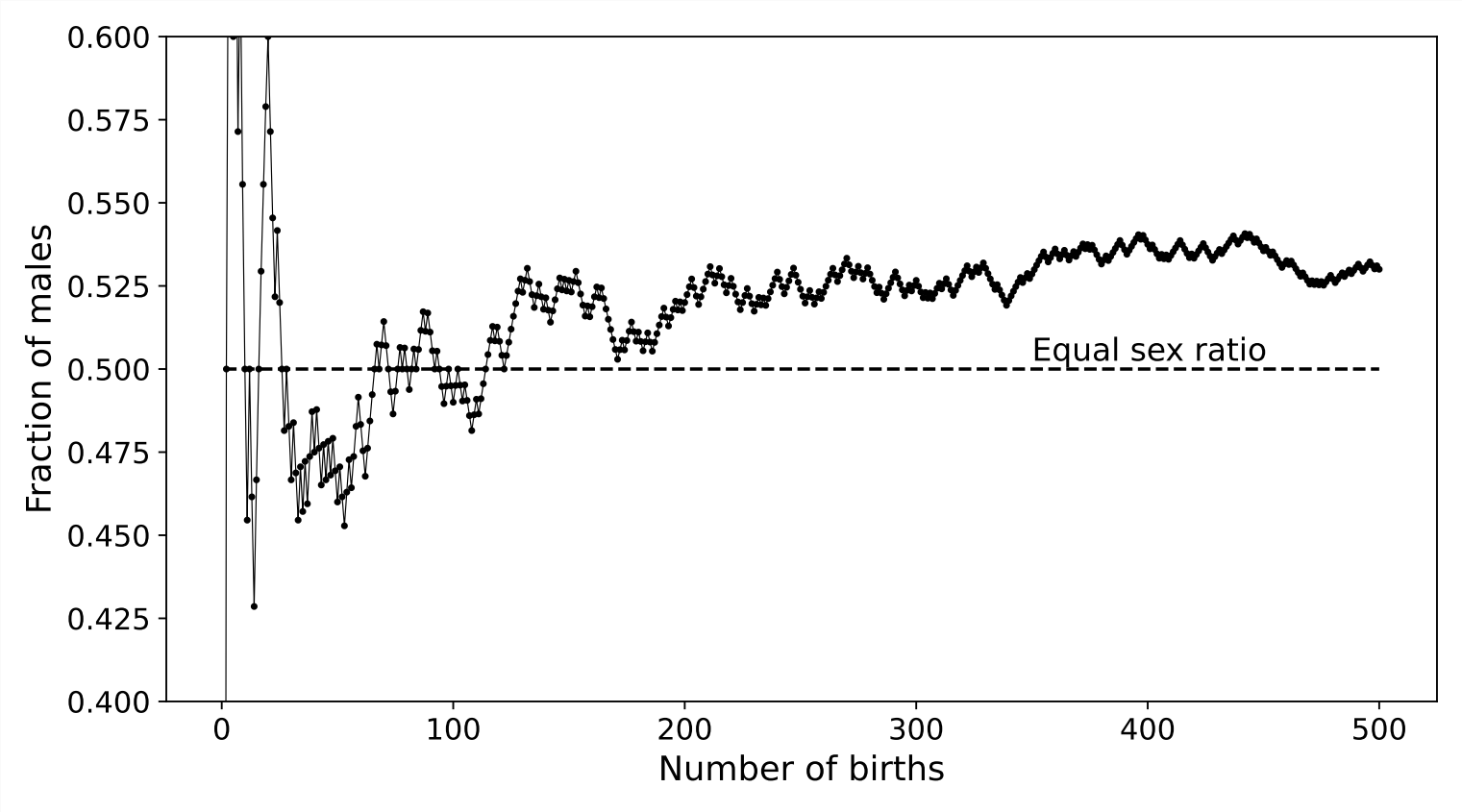



**Figure 19:** The cumulative fraction of males, $\hat{p} = n^{-1} \sum_{i=1}^{n} x_i$, calculated from a sequence of observations of baby viviparous lizards.

# Frequentist uncertainty

Our confidence in $\hat{p}_n^{\mathrm{ML}}$ as an estimate of $p$ grows as we observe more lizards. But how should we quantify this confidence?

In general it is useful to think of a parameter estimation method as a **function**, which we'll call $t$, of our observation vector $\underline{x} = (x_1, \ldots, x_n)$.

In our lizards example this function was defined by

$$t(\underline{x}) = \frac{1}{n} \sum_{i=1}^{n} x_i.$$

$t$ is called an *estimator*. Given dataset $\underline{x}$ the estimate it produces is

$$\hat{p} = t(\underline{x}).$$

To quantify uncertainty in an estimator, the frequentist considers the following thought experiment:

- Imagine our data consists of i.i.d. samples from an **unknown** probability distribution $F$.
- Let $\underline{x}_1, \underline{x}_2, \ldots$ be a sequence of datasets generated using $F$.
- If we plug these samples into our estimator we get a sequence of estimates for $\hat{p}$,

$$t(\underline{x}_1), t(\underline{x}_2), \ldots$$

- Uncertainty in our estimator is characterised by the distribution of these estimates, known as the **sampling distribution**.

# General case

Consider a system from which we can extract datasets, thought of as realisations of a random vector $\underline{X} = (X_1, \ldots, X_n)$.

## Frequentist uncertainty

Let $\underline{x}$ be the realisation of a vector $\underline{X}$ of i.i.d. random variables with common distribution $F$. Suppose we have constructed an estimator function $t$ for a parameter $\theta$. The distribution of the random variable

$$\hat{\Theta} = t(\underline{X})$$

is known as the **sampling distribution** of the estimator. The word 'estimator' is correctly applied both to the function $t$ and to the random variable $\hat{\Theta}$. Realisations of $\hat{\Theta}$ are *estimates*.

Frequentist uncertainty in the estimate $\hat{\theta} = t(\underline{x})$ is represented by the sampling distribution of estimator $t(\underline{X})$.

# Bias and consistency

Suppose our model says the distribution of $\underline{X}$ is $F_{\underline{X},\theta}$, where $\theta$ is a parameter to be estimated. The **bias** of estimator $t$ is defined as

$$b(\theta) = \mathbb{E}_{\underline{X} \sim F_{\underline{X},\theta}}(t(\underline{X})) - \theta.$$

Notice that the bias is a function of $\theta$. If $b(\theta) = 0$ for all possible values $\theta$, then the estimator is said to be *unbiased*; otherwise it is *biased*.

Many useful estimators are biased. In practice it is often more important to ask whether an estimator would approach the true value of the parameter as the sample size is increased, if the model were correct. Estimators with this property are said to be *consistent*.

# Standard Error

We can measure the typical spread of estimates that we are likely to get using the standard deviation of $\hat{\Theta}$. We call this the 'standard error' of our estimator.

**Standard error**

Let $\underline{X}$ be a vector of i.i.d. random variables with common distribution $F$. The **standard error** of an estimator, $t(\underline{X})$, of $\theta$, is

$$\mathrm{se} = \mathrm{se}(t) = \sqrt{\mathrm{Var}(\hat{\Theta})} = \sqrt{\mathrm{Var}(t(\underline{X}))}$$

where $\mathrm{Var}(-)$ denotes the variance taken with respect to the distribution of $\underline{X}$, or equivalently with respect to the sampling distribution of the estimator.

# Standard error in $\hat{p}$.

In our lizard sex ratio example our estimator of $p$ is

$$\hat{p} = t(\underline{x}) = \frac{1}{n}\sum_{i=1}^{n} x_i.$$

We know that

$$S_n = \sum_{i=1}^{n} X_i \sim \text{Binomial}(n, p^*),$$

so $\text{Var}(S_n) = np^*(1-p^*)$.

**Exercise**

Making use of the fact that for any random variable $Z$ and scalar $a$, $\text{Var}(aZ) = a^2\text{Var}(Z)$, calculate

$$\text{Var}\left(\frac{S_n}{n}\right).$$

Since

$$\mathrm{Var}(t(\underline{X})) = \mathrm{Var}\left(\frac{S_n}{n}\right) = \frac{p^*(1-p)^*}{n}$$

then

$$\mathrm{se} = \sqrt{\frac{p^*(1-p^*)}{n}} \qquad (\bigstar)$$

Since $p^*$ is unknown, we estimate the standard error by 'plugging in' our estimate of $p$ into ($\bigstar$). Hence

$$\hat{\mathrm{se}} = \sqrt{\frac{\hat{p}(1-\hat{p})}{n}}.$$

For example, using the first $n = 500$ births we obtain $\hat{p} = 0.531$ giving an approximate standard error of $\hat{\mathrm{se}} = 0.022$.

**Exercise**

A coin has probability $p$ of turning up heads. You obtain 14 heads from 20 flips. Estimate $p$ and the standard error in your estimate.

# Confidence intervals

**Definition of confidence interval**

Let $\underline{X}$ be a vector of i.i.d. random variables with common distribution $F$. Let $\theta$ be a parameter which we want to estimate. Suppose we define a procedure for constructing intervals of the form

$$C(\underline{X}) = [a(\underline{X}), b(\underline{X})]$$

such that for some number $\alpha \in [0, 1]$

$$\mathbf{Pr}(\theta \in C(\underline{X})) = 1 - \alpha.$$

If $\underline{x}$ is a single realisation of $\underline{X}$ then the interval $C(\underline{x}) = [a(\underline{x}), b(\underline{x})]$ is called a $1 - \alpha$ **confidence interval**. The number $1 - \alpha$ is called the *coverage* of the interval. Often we set $\alpha = 0.05$ giving a 95% confidence interval.

# Constructing confidence intervals

To simplify notation we'll use the same symbol, $\hat{\theta}$, for both the estimator function and the estimate, writing

$$\hat{\theta} = \hat{\theta}(\underline{x}),$$
$$\hat{\Theta} = \hat{\theta}(\underline{X}).$$

It can be shown that the sampling distributions of maximum likelihood estimators are approximately normal, i.e. that $\hat{\theta}(\underline{X}) \sim \mathcal{N}(\theta^*, \mathrm{se})$ or, equivalently that

$$\hat{\theta}(\underline{X}) \overset{d}{\approx} \theta^* + \mathrm{se} \times Z$$

where $Z \sim \mathcal{N}(0, 1)$. Suppose that we set the upper limit of our interval to be

$$b(\underline{X}) = \hat{\theta}(\underline{X}) + c \times \hat{\mathrm{se}}(\underline{X})$$

where $c$ is a constant to be determined and $\hat{\mathrm{se}}(\underline{X})$ is our estimate of the standard error based on data $\underline{X}$.

We have

$$\mathsf{Pr}(\theta^* > b(\underline{X})) \approx \mathsf{Pr}\left(Z < -c \times \frac{\hat{\mathrm{se}}(\underline{X})}{\mathrm{se}}\right) \approx \mathsf{Pr}(Z < -c). \tag{2}$$

We determine $c$ by imposing the condition

$$\mathsf{Pr}(Z < -c) = \frac{\alpha}{2}.$$

For a 95% interval, this yields $c = 1.96$. A similar argument can be used to derive the lower limit $a(\underline{X}) = \hat{\theta}(\underline{X}) - c \times \hat{\mathrm{se}}(\underline{X})$.

Given a dataset $\underline{x}$, the interval constructed according to the rule

$$C(\underline{x}) = [\hat{\theta}(\underline{x}) - 1.96 \times \hat{\mathrm{se}}(\underline{x}), \hat{\theta}(\underline{x}) + 1.96 \times \hat{\mathrm{se}}(\underline{x})]$$

is an approximate 95% confidence interval.

# Confidence interval for lizard sex ratio

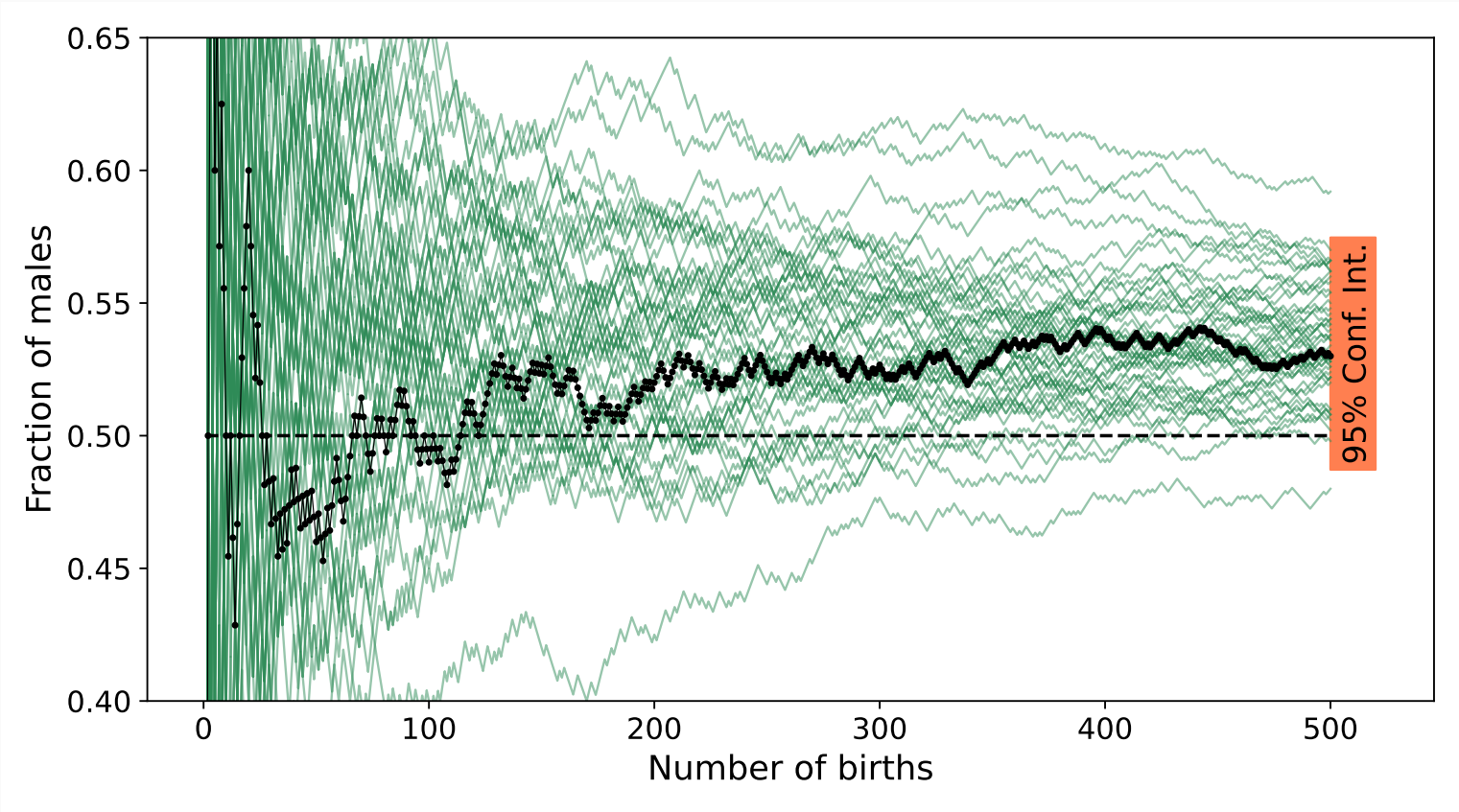



**Figure 20:** Green lines show cumulative fractions of males calculated from 50 synthetic datasets generated using the i.i.d. Bernoulli model with $\hat{p} = 0.531$. Black curve shows real data and orange bar shows the 95% confidence interval calculated from the real data.

# Summary and exercise

For our lizard data

$$\hat{p}(\underline{x}) = \frac{1}{n}\sum_{i=1}^{n} x_i \qquad \hat{\mathrm{se}} = \sqrt{\frac{\hat{p}(1-\hat{p})}{n}}.$$

The interval constructed using the rule

$$C(\underline{x}) = [\hat{p} - 1.96\hat{\mathrm{se}}, \hat{p} + 1.96\hat{\mathrm{se}}]$$

is an approximate 95% confidence interval.

**Exercise**

Of the first $n = 1000$ baby lizards observed, 549 were male. Compute a 95% confidence interval for $p$ using this new data.

# Interpretation of confidence intervals

The probability statement

$$\Pr(\theta \in C(\underline{X})) = 1 - \alpha$$

is open to misinterpretation. It does *not* mean that given a specific dataset, $\underline{x}$, the probability that $\theta \in C(\underline{x})$ is $1 - \alpha$. After all, the statement $\theta \in C(\underline{x})$ does not involve any random quantities (according to a **frequentist**).

The correct interpretation is that if we repeated the data gathering process many times, computing a series of intervals

$$C(\underline{x}_1), C(\underline{x}_2), \ldots$$

then about a fraction $1 - \alpha$ of them would contain the true parameter.

In the frequentist world the *interval* which we computed from the data is treated as the random quantity, not the parameter.

# The Bayesian way

The **frequentist** definition of probability is based on the concept of an **infinitely repeatable trial**. There are important kinds of uncertainty it cannot describe.

We **cannot attach frequentist probabilities** to the following hypotheses

- $\theta \in [a, b]$ where $\theta$ is an unknown parameter.
- The ship sank at coordinates $(x, y)$.
- Suspect X stole the jewels.
- Unicorns exist.

We can attach probabilities to hypotheses like these if we interpret probability as our *degree of belief* in their truth. This is the **Bayesian** interpretation of probability.

To make probability statements about parameters, we view $p$ as a single realisation of a random variable $P$.

Although we cannot observe $P$—it is a **latent variable**—we imagine that it will still take a specific value, determined by the evolutionary process which has created the current population of lizards.

We declare our initial state of knowledge about the value of $P$ using a **prior** probability density function $f_P(p)$.

The quantity

$$\Pr(P \in [a, b]) = \int_a^b f_P(p)dp$$

is our degree of belief *before we have observed any baby lizards* that the realised value of $P$ lies in the interval $[a, b]$.

# Bayes' Rule

Given two random variables with joint density (or mass) function $f_{X,Y}(x, y)$, the conditional density (or mass) function of $X$ given $Y$ is

$$f_{X|Y}(x|y) = \frac{f_{X,Y}(x, y)}{f_Y(y)} \qquad (\bigstar)$$

where $f_Y(y)$ is called the **marginal** density of $Y$, given by

$$f_Y(x) = \int f_{X,Y}(x, y)dx.$$

**Exercise**

Using the definition of conditional density ($\bigstar$), establish **Bayes' Rule**

$$f_{X|Y}(x|y) = \frac{f_{Y|X}(y|x)f_X(x)}{f_Y(y)}.$$

# Updating the prior after observing one lizard

Since $P$ is a random variable, it has a joint mass-density function with $X_1$. Using the definition of conditional probability we can write

$$f_{X,P}(x,p) = f_{X|P}(x|p)f_P(p),$$

where $f_{X|P}(x|p)$ is the mass function of $X_1$ given that $P = p$. This is just our original (frequentist) probability model for $X_1$,

$$\underbrace{f_{X|P}(x|p)}_{\text{Bayesian}} = p^x(1-p)^{1-x} = \underbrace{f_X(x;p)}_{\text{Frequentist}}.$$

The probability density of $P$ given the **observation** $X_1 = x$ may be obtained using **Bayes' rule**,

$$\underbrace{f_{P|X}(p|x)}_{\text{Posterior}} = \frac{f_{X|P}(x|p)f_P(p)}{f_X(x)}.$$

The density $f_{P|X}(p|x)$ is called the **posterior**, meaning 'coming after'.

# Posterior $\propto$ Likelihood $\times$ Prior

In applying Bayes' rule, we introduced a new mass function, $f_X(x)$, the marginal distribution of $X_1$, given by

$$f_X(x) = \int_0^1 f_{X,P}(x,p)dp = \int_0^1 f_{X|P}(x|p)f_P(p)dp.$$

$f_X(x)$ has no counterpart in the frequentist world. It acts as the normalising constant for the posterior distribution.

$f_{X|P}(x|p)$ is the **likelihood** of $p$ based on one observation. Thus

$$\text{Posterior} \propto \text{Likelihood} \times \text{Prior}.$$

**Exercise**

Suppose that $f_{X,P}(x,p) = 2p^{1+x}(1-p)^{1-x}$. Find $f_X(0)$ and $f_X(1)$.

# Calculating the posterior

Let us take an open minded approach and assume that all values of $P$ are equally plausible, corresponding to a *uniform* prior

$$f_P(p) = \begin{cases} 1 & \text{if } p \in [0,1], \\ 0 & \text{otherwise.} \end{cases}$$

The first baby lizard is male ($x_1 = 1$), giving posterior

$$\begin{aligned} f_{P|X}(p|x_1) &= \frac{f_{X|P}(1|p) f_P(p)}{f_X(1)} \\ &= \frac{p \times 1}{f_X(1)} \qquad (\bigstar) \\ &= 2p. \end{aligned}$$

**Exercise**

Show that the marginal $f_X(1)$ in the denominator of $(\bigstar)$ is $\frac{1}{2}$.

# Prior and posterior

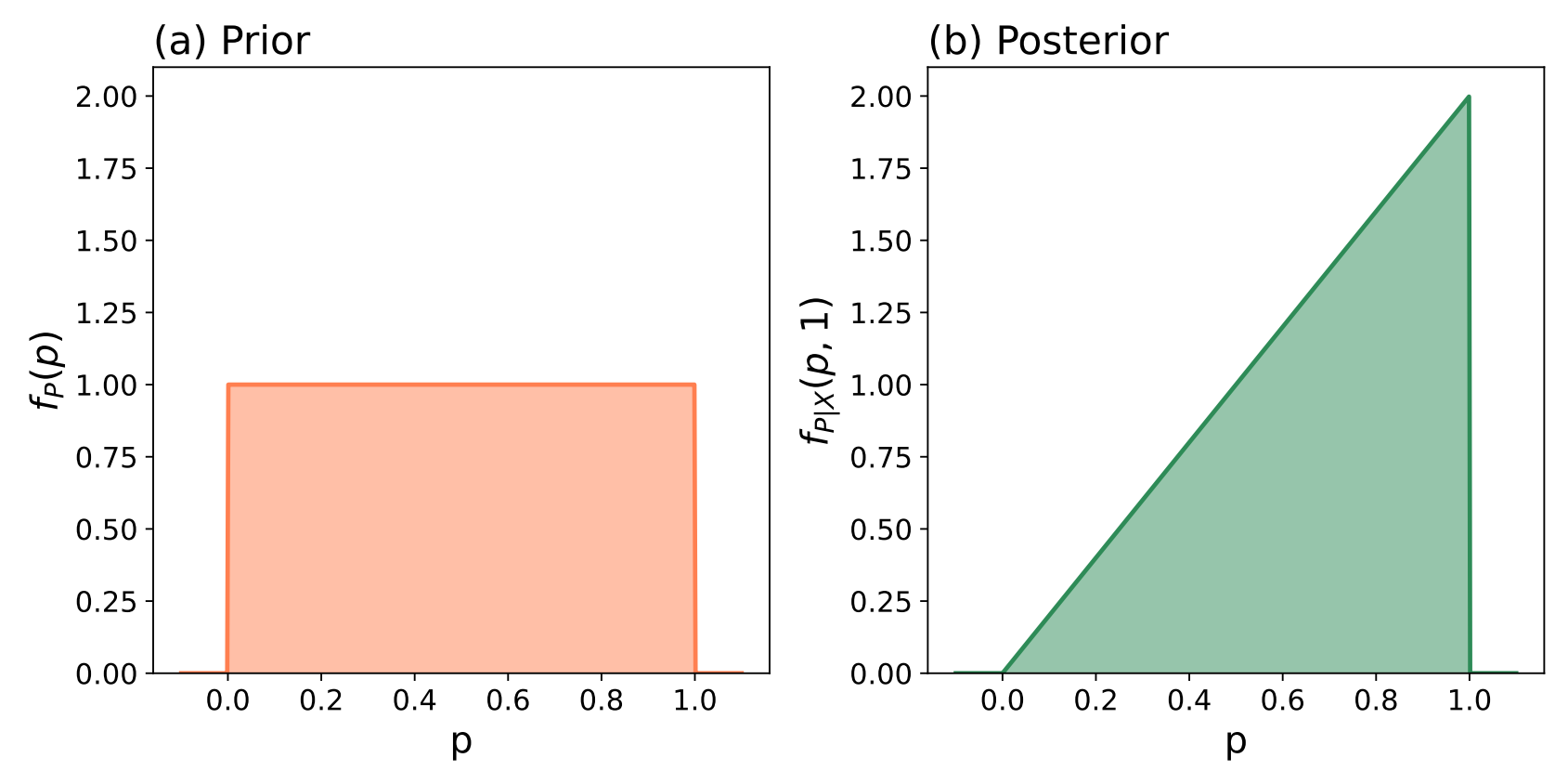



**Figure 21:** (a) Uniform prior distribution of the probability that a lizard will be born male. (b) Posterior distribution after observing a single male baby lizard.

# Many observations

Since our lizard observations are independent and identically distributed, their joint probability mass function is

$$f_{\underline{X}|P}(\underline{x}|p) = \prod_{i=1}^{n} f_{X|P}(x_i|p)$$

which is just the likelihood function $\mathcal{L}(p)$. A single application of Bayes' rule gives us the posterior

$$f_{P|\underline{X}}(p|\underline{x}) = \frac{f_{\underline{X}|P}(\underline{x}|p)f_P(p)}{f_{\underline{X}}(\underline{x})} \propto \mathcal{L}(p)f_P(p).$$

If we think of the marginal in the denominator purely as a normalising constant, we can write

$$f_{P|\underline{X}}(p|\underline{x}) = \frac{\mathcal{L}(p)f_P(p)}{c},$$

where $c$ is chosen so that $\int_0^1 f_{P|\underline{X}}(p|\underline{x})dp = 1$. This is often the simplest way to write the relationship between prior, posterior and likelihood.

# Application to lizard data

We found previously that $\mathcal{L}(p) = p^{s_n}(1-p)^{n-s_n}$, where $s_n = \sum_{i=1}^{n} x_i$. With a uniform prior, the posterior is

$$f_{P|\underline{X}}(p|\underline{x}) = \frac{p^{s_n}(1-p)^{n-s_n}}{c}.$$

In this example we can recognise that the posterior is a member of a well known family: the beta distributions.

**Beta distribution**

If $P \in [0,1]$ is beta distributed we write $P \sim \mathrm{Beta}(\alpha, \beta)$, where $\alpha > 0$ and $\beta > 0$ are parameters. The probability density function of $P$ is

$$f_P(p) = \frac{p^{\alpha-1}(1-p)^{\beta-1}}{B(\alpha,\beta)},$$

where $B(\alpha,\beta)$ is the beta function. We have $\mathbb{E}(P) = \alpha/(\alpha+\beta)$ and $\mathrm{mode}(P) = (\alpha-1)/(\alpha+\beta-2)$.

# Posteriors

We see that our posterior is a beta distribution with $\alpha = s_n + 1$ and $\beta = n - s_n + 1$, or equivalently

$$P \sim \text{Beta}(s_n + 1, n - s_n + 1). \tag{3}$$

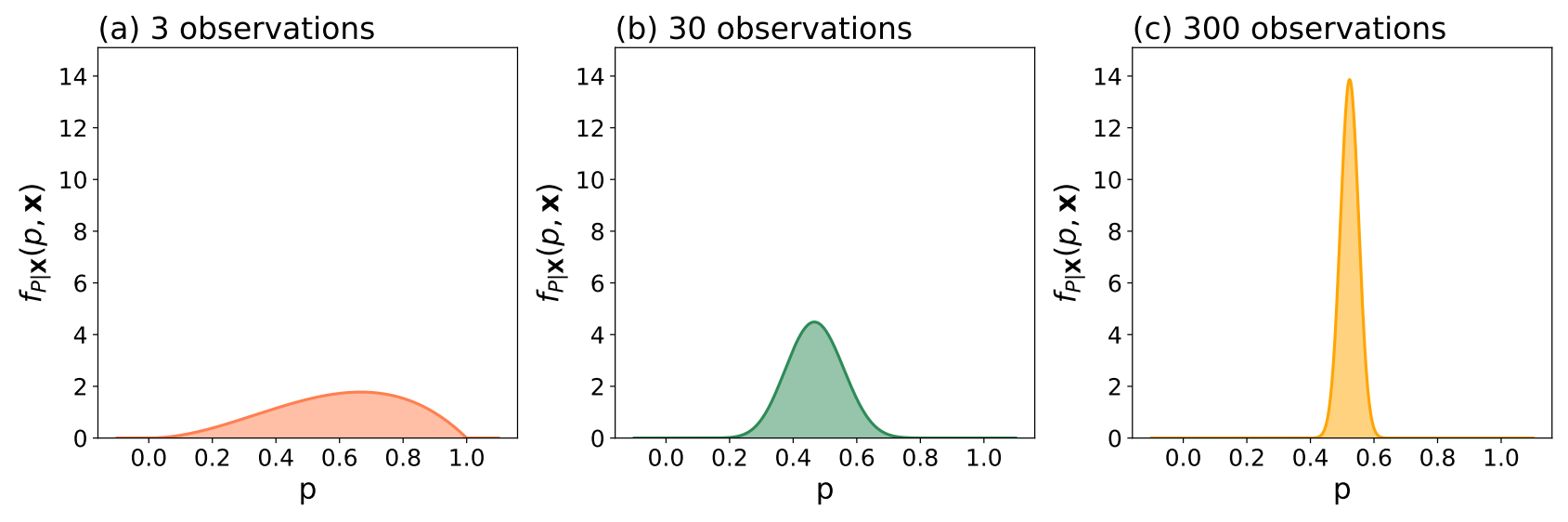



**Figure 22:** Posterior distributions of the male probability $P$ after making increasing numbers of of observations.

# Uncertainty and posterior mean

The posterior distribution captures our uncertainty in the realised value of $P$. As we make more observations it becomes increasingly sharply peaked, indicating increased certainty.

Once we have the posterior, we can obtain parameter estimates. One option is the **posterior mean** (PM) estimate

$$\hat{p}^{\mathrm{PM}} = \mathbb{E}(P) = \frac{\sum_{i=1}^{n} x_i + 1}{n + 2}.$$

**Exercise**

Use the properties of the Beta distribution to derive the posterior mean result given above.

# MAP estimate

Another option for estimating $p$ is the **maximum a posteriori** (MAP) estimate

$$\hat{p}^{\mathrm{MAP}} = \mathrm{mode}(P) = \frac{1}{n}\sum_{i=1}^{n} x_i = \hat{p}^{\mathrm{ML}}.$$

The fact that the MAP and maximum likelihood estimates coincide in this case should come as no surprise: we selected a uniform prior, so the posterior density is proportional to the likelihood.

**Exercise**

Use the properties of the Beta distribution to derive the MAP estimate given above.

# Credible intervals

In the Bayesian world we can define intervals which have a specified probability of containing the parameter, according to the posterior distribution. To do this, we first select the desired probability $1 - \alpha$. We then find $a$ and $b$ such that

$$\int_{-\infty}^{a} f_{\Theta|\underline{X}}(\theta, \underline{x}) d\theta = \int_{b}^{\infty} f_{\Theta|\underline{X}}(\theta, \underline{x}) d\theta = \frac{\alpha}{2}.$$

Letting $C = [a, b]$, we then have

$$\mathbf{Pr}(\Theta \in C|\underline{x}) = \int_{a}^{b} f_{\Theta|\underline{X}}(\theta, \underline{x}) d\theta = 1 - \alpha.$$

In other words, the probability (our degree of belief) that $\Theta$ lies in $C$ is $1 - \alpha$. The interval $C$ is called the $1 - \alpha$ credible interval.

# Conclusions

- **Frequentist** probabilities represent long-run event frequencies in **repeatable trials**.
- Therefore, frequentist probability can be used to express uncertainty only about events that are outcomes of repeatable trials.
- Frequentist uncertainty in parameter estimates is represented by the **sampling distribution** and **confidence intervals**.
- **Bayesian** probabilities represent **degrees of belief**.
- Therefore, Bayesian probability can be used to express uncertainty about *any* hypothesis.
- Bayesian uncertainty is represented by the **posterior distribution** and **credible intervals**.

# Frequentist Linear Regression

# Motivation

- Previously we considered conditional probabilistic models of a response variable $Y$ based on **one** predictor $X$.
- We now generalise to the case of **multiple** predictors $X_1, X_2, \ldots, X_p$.
- We will consider models which assume the response variable depends **linearly** on the predictors. These are known as **linear regression** models.
- We will explore model **performance** and **selection**, **overfitting**, model **interpretation**, **explanatory power**, and the connections between variable **correlations** and model **parameters**.
- We will also lay the ground work for introducing **regularisation**—a systematic way to control model complexity.

# Predicting body fat percentage

- Accurately measuring body fat percentage (BFP) requires specialist equipment.
- Can we estimate it using simple body measurements?
- We will explore that question using measurements of BFP, height, body mass index (BMI), age, and various body part circumferences for a group of 248 men.

We write our data set $D = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{n}$ where $y_i$ is the BFP of the $i$th individual. We write the $i$th predictor **vector**

$$\boldsymbol{x}_i = (x_{i1}, x_{i2}, \ldots, x_{ip})$$

where $x_{ij}$ is the value of the $j$th measurement of the $i$th individual.

# Conditional probabilistic model

Let $Y, X_1, X_2, \ldots, X_p$, be the measurements of a man selected at random from the population. We write $\boldsymbol{X} = (X_1, X_2, \ldots, X_p)$.

We hypothesise the following **multiple linear** model:

$$\begin{aligned} Y &= \beta_0 + \beta_1 X_1 + \ldots + \beta_p X_p + \mathcal{E} \\ \mathcal{E} &\sim \mathcal{N}(0, \sigma^2), \end{aligned}$$

so

$$Y|\{\boldsymbol{X} = \boldsymbol{x}\} \sim \mathcal{N}\left(\beta_0 + \sum_{j=1}^{p} \beta_j x_j, \sigma^2\right).$$

The probability density of $Y$ given $\boldsymbol{X}$ is therefore

$$f_{Y|X}(y|\boldsymbol{x}; \boldsymbol{\theta}) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{1}{2\sigma^2}\left(y - \beta_0 - \sum_{j=1}^{p} \beta_j x_j\right)^2\right),$$

where $\boldsymbol{x} = (x_1, \ldots, x_p)$ and $\boldsymbol{\theta} = (\beta_0, \beta_1, \ldots, \beta_p, \sigma)$.

# Maximum likelihood and regression function

Assuming that observations of $Y$ are conditionally independent given $\boldsymbol{X}$, then the likelihood function is

$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{\sigma^n (2\pi)^{\frac{n}{2}}} \exp\left( -\frac{1}{2\sigma^2} \sum_{i=1}^{n} \left( y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 \right).$$

Maximum likelihood parameter estimates are given by

$$\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}).$$

We predict the value of $Y$ given that $\boldsymbol{X} = \boldsymbol{x}$ using the *regression function* $r(\boldsymbol{x}) = \mathbb{E}(Y|\boldsymbol{X} = \boldsymbol{x})$.

If $\boldsymbol{x}$ is a new observation of the predictor vector, then, since $\mathbb{E}(\mathcal{E}) = 0$, our prediction of the response variable is

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x_1 + \ldots + \hat{\beta}_p x_p.$$

Figure 23 shows the relationship between three variables which we might expect to be strongly associated with body fat. The relationships between predictor and response are approximately *linear*. That is, we can write them in the form

$$Y = a + bX + \text{`noise'}.$$

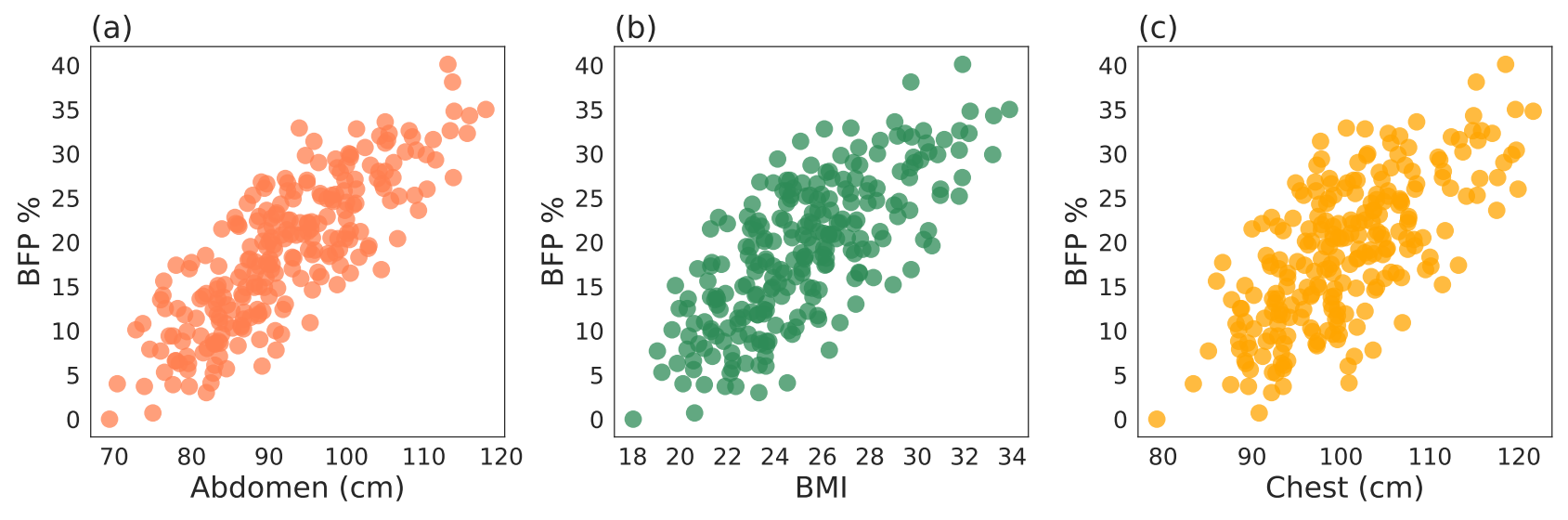



**Figure 23:** Scatter plots of BFP against Abdomen, BMI, and Chest.

# Correlations

The strength of the **linear** relationship between variables $X_i$ and $X_j$ can be measured by their **correlation**

$$\rho(X_i, X_j) = \frac{\mathrm{Cov}(X_i, X_j)}{\sqrt{\mathrm{Var}(X_i)\mathrm{Var}(X_j)}}.$$

The Pearson correlation coefficient between measurement $i$ and measurement $j$ is a sample approximation to the correlation, given by

$$\hat{\rho}(X_i, X_j) = \frac{\sum_{k=1}^{n}(x_{ki} - \bar{x}_i)(x_{kj} - \bar{x}_j)}{\sqrt{\sum_{k=1}^{n}(x_{ki} - \bar{x}_i)^2}\sqrt{\sum_{k=1}^{n}(x_{kj} - \bar{x}_j)^2}} \in [-1, 1]$$

where

$$\bar{x}_j = \frac{1}{n}\sum_{k=1}^{n} x_{kj}.$$

# Pearson correlations

A high correlation between $Y$ and $X_i$ suggests that $X_i$ may be a good predictor of $Y$. However, relationships **between** predictors matter.

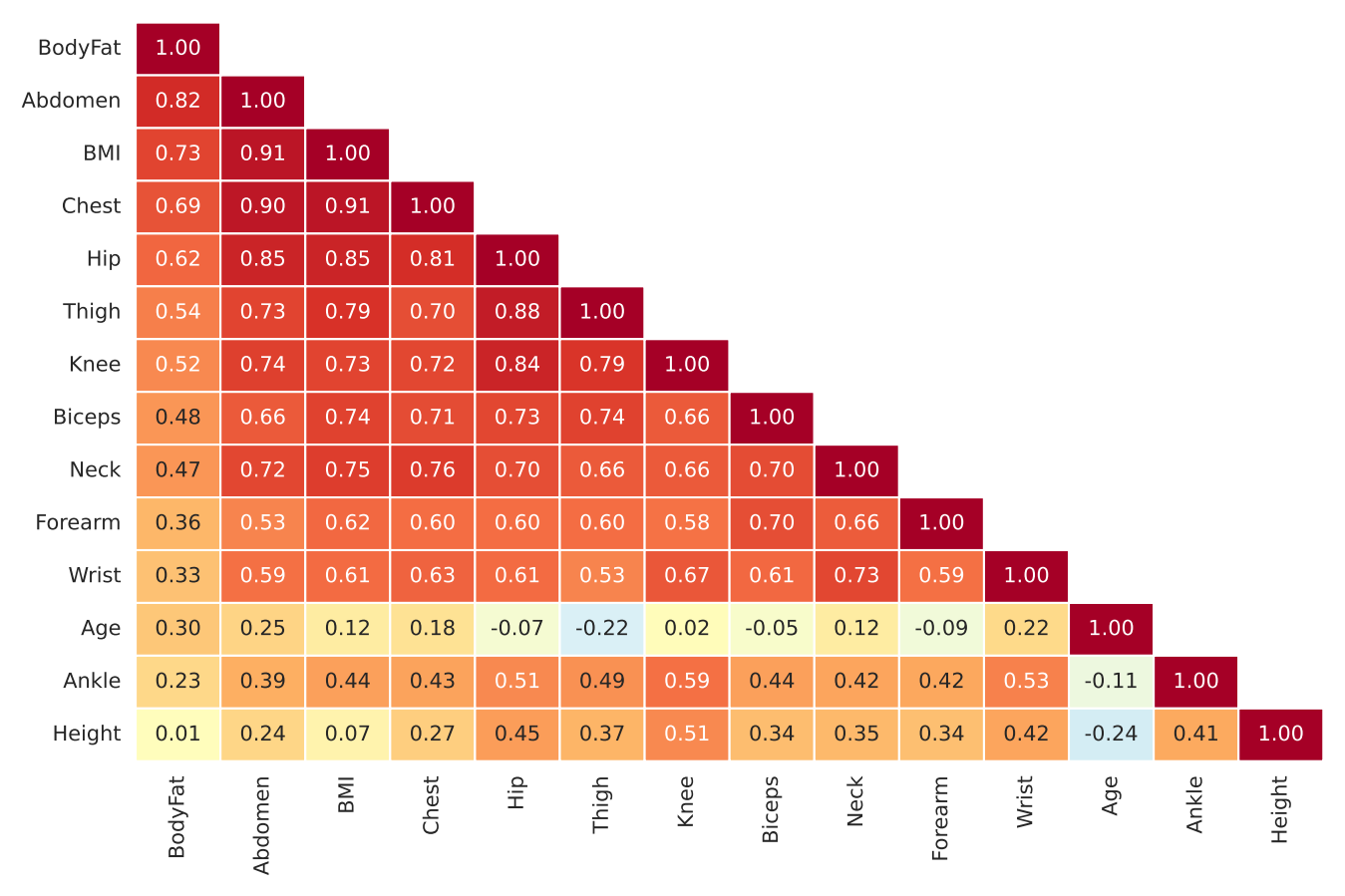



**Figure 24:** Sample correlations between measurements in our dataset.

# k-fold cross validation

- We previously introduced **leave one out cross validation**.
- A faster alternative is **k-fold cross validation**.
- We randomly split the dataset into $k$ equal chunks or 'folds'.
- We **hold out** one fold and fit the model to the remaining data.
- We then calculate model prediction errors on the **held-out** fold.
- We repeat this for each fold and average the errors.
- Typically we set $k = 5$ or $k = 10$.
- k-fold cross validation errors measure **performance on unseen data**, and can be used for **model selection** (choosing the best model).
- We also hold out a small(ish) **test set** from the model selection process, which is used to obtain an unbiased estimate of the **test error** for the final model we select.

# Cross validation pipeline

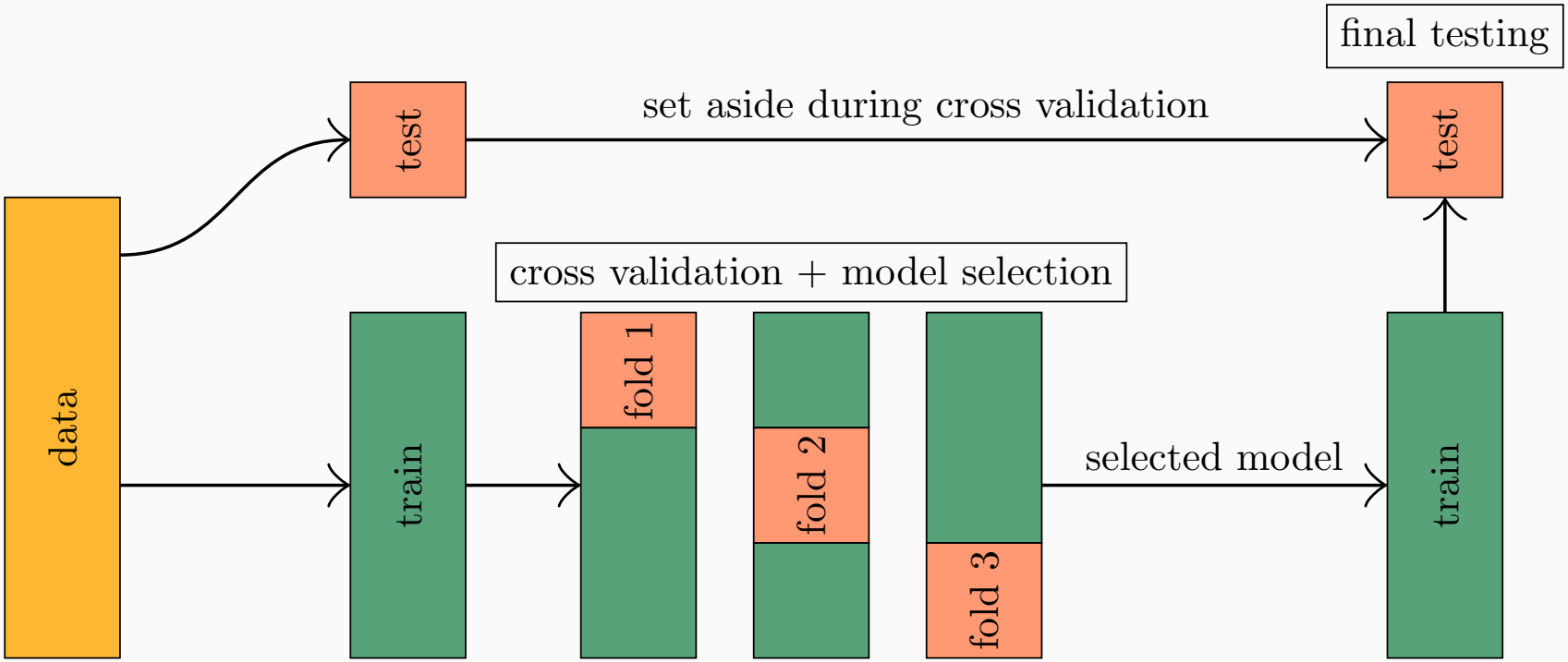



**Figure 25:** The cross validation, model selection, and final testing pipeline. The pipeline consists of (i) performing a train-test split, (ii) selecting a model using cross-validation (iii) training the selected model on the full training set and (vi) testing the model on the held out test set.

# Single-predictor models

Suppose we select a single predictor and fit a model of the form

$$Y = \beta_0 + \beta_1 X + \mathcal{E}$$
$$\mathcal{E} \sim \mathcal{N}(0, \sigma).$$

Here $X$ represents one of the available predictors $X_1, X_2, \ldots, X_p$.

Maximising the likelihood computed using $D_{\text{train}}$ we obtain the following closed form coefficient estimates:

$$\hat{\beta}_1 = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2}$$
$$\hat{\beta}_0 = \bar{y} - \hat{\beta}_1 \bar{x}.$$

Using these estimates, if we are given a new predictor value $x$, then we predict the corresponding BFP ($y$) to be $\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x$.

## Exercise

The likelihood function corresponding to the linear model $Y = \beta_0 + \beta_1 X + \mathcal{E}$ is

$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{\sigma^n (2\pi)^{\frac{n}{2}}} \exp\left( -\frac{1}{2\sigma^2} \sum_{i=1}^{n} (y_i - \beta_0 - \beta_1 x_i)^2 \right).$$

(a) Show that $\ell(\boldsymbol{\theta}) = -\frac{1}{2\sigma^2}\mathrm{RSS} - n\log(\sigma\sqrt{2\pi})$, where

$$\mathrm{RSS}(\beta_0, \beta_1) = \sum_{i=1}^{n} (\beta_0 + \beta_1 x_i - y_i)^2.$$

(b) Find expressions for $\hat{\beta}_0$ and $\hat{\beta}_1$ by solving the equations

$$\frac{\partial \mathrm{RSS}}{\partial \beta_0} = 0, \qquad \frac{\partial \mathrm{RSS}}{\partial \beta_1} = 0$$

# The sampling distribution

Having fitted a model, we want to characterise our level of certainty in the values of the model parameters.

Suppose **hypothetically** that we had access to many independently sampled datasets from the same population,

$$D_1, D_2, \ldots$$

The distribution of parameter estimates we would obtain by fitting the model to each of these datasets in turn is the **sampling distribution**.

The sampling distribution is how we characterise frequentist uncertainty in the fitted model.

**In reality we don't have multiple datasets**.

We can **synthesize** new datasets by sampling from the **empirical distribution**, which places probability weight $1/n$ on each data point we have observed already,

$$f_{X,Y}^{\text{emp}}(x, y) = \begin{cases} \frac{1}{n} & \text{if } (x, y) \in D \\ 0 & \text{otherwise .} \end{cases}$$

Samples from the empirical distribution match the statistical properties of $D$, and can be used to **approximate** truly new samples. We write the synthetic **'bootstrap' samples** $\hat{D}_1, \hat{D}_2, \ldots, \hat{D}_B$.

Fitting the model to each of these in turn produces a set of parameter estimates $\hat{\beta}_1^{(1)}, \hat{\beta}_1^{(2)}, \ldots, \hat{\beta}_1^{(B)}$ from which the **bootstrap standard error** can be calculated,

$$\hat{\text{se}}(\hat{\beta}_1) = \sqrt{\frac{1}{B}\sum_{b=1}^{B}\left(\hat{\beta}_1^{(b)} - \frac{1}{B}\sum_{i=1}^{B}\hat{\beta}_1^{(i)}\right)^2}.$$

# Standardising predictors

To compare the sensitivity of our models to different different predictors it useful to *standardise* them so that their variations within the training data are all of a similar magnitude.

The magnitude of variations in predictor $X_k$ can be measured using the standard deviation estimate $\hat{\sigma}_k$, defined by

$$\hat{\sigma}_k^2 = \frac{1}{n}\sum_{i=1}^{n}(x_{ik} - \bar{x}_k)^2.$$

We then define *standardised* predictors

$$z_{ik} = \frac{x_{ik} - \bar{x}_k}{\hat{\sigma}_k} \tag{4}$$

where $\bar{x}_k$ is the (training) sample mean of $X_k$.

Coefficient estimates based on standardised predictors are denoted $\hat{\beta}_1^*$ rather than $\hat{\beta}_1$.

# Results for single-predictor models

Table 1 summarises five alternative single-predictor models. Predictors which are more highly correlated to the response produce predictions with smaller errors.

**Table 1:** Coefficients, bootstrap standard errors, cross validated RMSE and $R^2$ results for single-predictor linear models. Starred parameters, $\beta_1^*$, are based on standardised predictors.

| Predictor | $\hat{\beta}_1$ | $\hat{\text{se}}(\hat{\beta}_1)$ | $\hat{\beta}_1^*$ | $\hat{\text{se}}(\hat{\beta}_1^*)$ | RMSE | $R^2$ |
|---|---|---|---|---|---|---|
| Abdomen | 0.636 | 0.034 | 6.160 | 0.341 | 4.792 | 0.634 |
| BMI | 1.847 | 0.118 | 5.599 | 0.357 | 5.417 | 0.523 |
| Chest | 0.644 | 0.050 | 5.050 | 0.378 | 5.997 | 0.426 |
| Hip | 0.830 | 0.071 | 4.820 | 0.424 | 6.182 | 0.388 |
| Wrist | 2.861 | 0.607 | 2.575 | 0.572 | 7.458 | 0.111 |

# Checking assumptions

After estimating the coefficients, our approximation to the true relationship between predictor and response is

$$Y = \hat{\beta}_0 + \hat{\beta}_1 X + \mathcal{E},$$

where $\mathcal{E} \sim \mathcal{N}(0, \hat{\sigma})$ and $\mathcal{E} \perp\!\!\!\perp X$. Here $\hat{\sigma}$ is the maximum likelihood estimate of $\sigma$. Letting our prediction of the response based on $X$ define a random variable

$$\hat{Y} = \hat{\beta}_0 + \hat{\beta}_1 X,$$

our fitted model can be written

$$Y - \hat{Y} = \mathcal{E}.$$

According to our assumptions about $\mathcal{E}$, the random variable $Y - \hat{Y}$ should be normal and independent of $X$. Realisations of $Y - \hat{Y}$ are the *residuals*, $e_i = y_i - \hat{y}_i$.

# Residual Plots

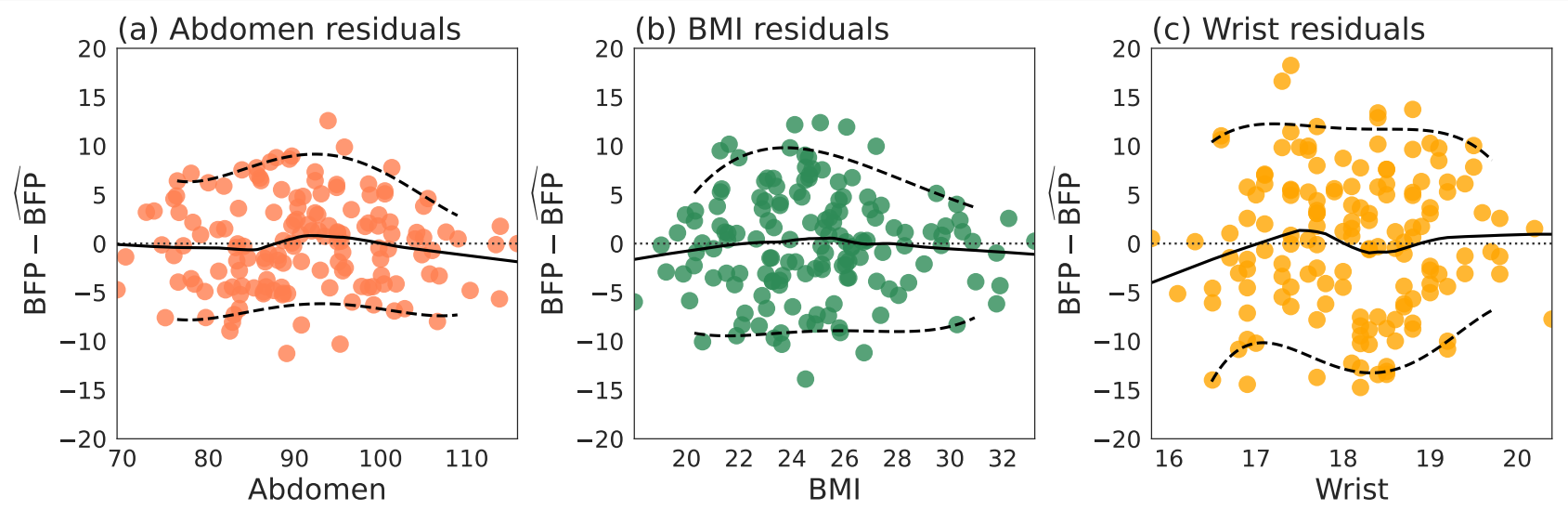



**Figure 26:** Scatter plots of the residuals $e_i = y_i - \hat{y}_i$ (or $\text{BFP} - \widehat{\text{BFP}}$) against the predictors $x_i$ for single-predictor models. Solid curves give the approximate relationship between the mean residual and the predictor. Dashed curves show approximate 5% and 95% quantiles of the residual distribution.

# Key points

- The mean residual functions (solid curves in Fig. 26) all lie close to the zero function; this suggests that a linear model was appropriate.
- In each case, the width of the scatter depends only weakly on the predictor, implying that the noise is approximately **homoskedastic**.
- The distribution of noise is approximately normal (Fig. 27).

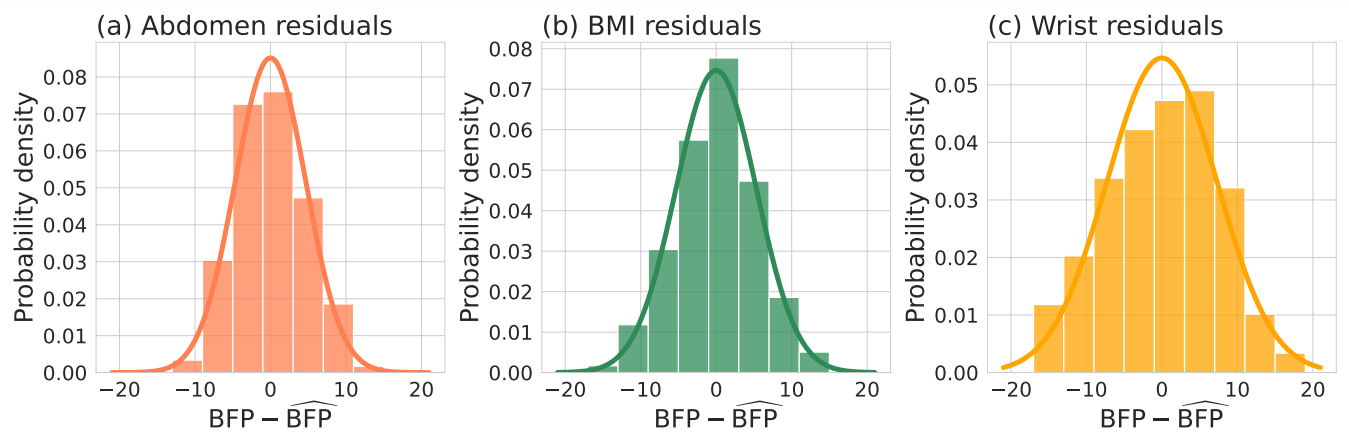



**Figure 27:** Histograms approximating the distributions of residuals.

# Coefficient of determination

We can quantify how much of the variation in $Y$ is explained by $\hat{Y}$ using

$$\mathrm{RSS} = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2,$$

where the sum may be over the training data, or over a dataset which was not used for training (e.g. a fold or a test set).

The **root mean square error** is defined

$$\mathrm{RMSE} = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2} = \sqrt{\mathrm{MSE}},$$

where $\mathrm{MSE}$ is the **mean squared error**. The $\mathrm{MSE}$, $\mathrm{RMSE}$ and $\mathrm{RSS}$ are all measures of *absolute* model error.

It is useful to introduce a baseline measure of the magnitudes of the variations in $Y$. For this purpose we define the *total sum of squares*

$$\mathrm{TSS} = \sum_{i=1}^{n} (y_i - \bar{y})^2.$$

We now define a coefficient which measures the **explanatory power** of linear models.

**Coefficient of determination**

The *coefficient of determination*

$$R^2 = 1 - \frac{\mathrm{RSS}}{\mathrm{TSS}}$$

measures the fraction of the variation in $Y$ which is explained by the model. We assume, unless stated otherwise, that $R^2$ is calculated using $D_{\mathrm{train}}$.

# Connection to correlation

Suppose that the **true** relationship between $X$ and $Y$ is of the form

$$Y = \beta_0 + \beta_1 X.$$

Here, **all** of the variation in $Y$ is explained by variations in $X$, so $Y$ and $X$ are **perfectly correlated**.

A model fitted to data produced by this mechanism will make perfect predictions, so $\mathrm{RSS} = 0$ and $R^2 = 1$.

Now suppose that the **true** relationship between $X$ and $Y$ is of the form

$$Y = \beta_0 + \mathcal{E}.$$

Here, **none** of the variation in $Y$ is explained by variations in $X$, so $Y$ and $X$ are **perfectly uncorrelated**.

A model fitted to data produced by this mechanism will yield a prediction function $\hat{y} = \bar{y}$. In this case we would find $\mathrm{RSS} = \mathrm{TSS}$ so $R^2 = 0$.

The fact that $R^2 = 1$ when $Y$ and $X$ are perfectly correlated and $R^2 = 0$ when they are perfectly uncorrelated is no coincidence. **We can show that $R^2$ is equal to the square of Pearson correlation coefficient between $Y$ and $X$.**

Given the the model $Y = \beta_0 + \beta_1 X + \mathcal{E}$, the maximum likelihood coefficient estimates are

$$\hat{\beta}_1 = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2} \qquad (\bigstar)$$
$$\hat{\beta}_0 = \bar{y} - \hat{\beta}_1 \bar{x}$$

and the maximum likelihood variance estimates are

$$\hat{\sigma_y}^2 = \frac{1}{n}\sum_{i=1}^{n}(y_i - \bar{y})^2$$
$$\hat{\sigma_x}^2 = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2.$$

## Exercise

Let $\hat{\rho}$ be the Pearson correlation between $Y$ and $X$. We can show that $R^2 = \hat{\rho}^2$.

(a) Making use of equation ($\bigstar$) show that $\hat{\beta}_1 = \frac{\hat{\rho}\hat{\sigma}_y}{\hat{\sigma}_x}$.

(b) Show that the $\mathrm{RSS}$ computed from $D_{\mathtt{train}}$ may be written

$$\mathrm{RSS} = \sum_{i=1}^{n} \Big(\hat{\beta}_1(x_i - \bar{x}) - (y_i - \bar{y})\Big)^2$$

(c) Use your answers to (a) and (b) to show that $\mathrm{RSS} = n\hat{\sigma}_y{}^2(1 - \hat{\rho}^2)$.

(d) Using that fact that $\mathrm{TSS} = n\hat{\sigma}_y{}^2$, show that

$$R^2 = 1 - \frac{\mathrm{RSS}}{\mathrm{TSS}} = \hat{\rho}^2.$$

# Single-predictor models: summary

- For **linear** predictor-response relationships, predictors having greater correlation (magnitude) with the response will produce more accurate predictive models.
- In practice we can only **estimate** correlations.
- $R^2$ (as normally used) measures the ability of a linear model to describe the **data it was trained on**, while cross validation scores measure performance on **unseen data**.
- Due to the simplicity of single-predictor linear models, this distinction is typically not consequential—we are unlikely to overfit.

# Multiple predictor models

Will adding more predictors necessarily produce more accurate predictions? If the answer is no, then which predictors should we use?

Let's suppose we have decided to use a given subset of the predictors. Let $P \subseteq \{1, 2, \ldots, p\}$ be their indices. Our model is then

$$Y = \beta_0 + \sum_{k \in P} \beta_k X_k + \mathcal{E}.$$

After we have estimated our model coefficients from the training data, we can view the predictions of our model as realisations of a random variable

$$\hat{Y} = \hat{\beta}_0 + \sum_{k \in P} \hat{\beta}_k X_k.$$

# Coefficient of determination

$R^2$ is defined identically to the single-predictor case:

$$R^2 = 1 - \frac{\mathrm{RSS}}{\mathrm{TSS}} = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}.$$

In the single-predictor case we found $R^2 = \hat{\rho}(X, Y)^2$. With multiple predictors, we have

$$R^2 = \hat{\rho}(Y, \hat{Y})^2.$$

Overfitting can be detected using a cross validated value of $R^2$. We compute the $\mathrm{RSS}$ and $\mathrm{TSS}$ and $R^2$ for each fold based on the model obtained from the remaining training data. Averaging the $R^2$ values over the folds we obtain $R^2_{\mathrm{test}}$.

Finding $R^2_{\mathrm{test}} \ll R^2$ is a sign of **overfitting**.

# Two predictors

Consider two-predictor models using abdomen + one extra measurement.

Fitting the BMI-Abdomen model we obtain

$$\hat{y} = 18.54 + 5.86 z_{\mathrm{Abd}} + 0.33 z_{\mathrm{BMI}}.$$

$z_{\mathrm{Abd}}$ and $z_{\mathrm{BMI}}$ are **standardised** abdomen and BMI, representing **deviations** from the mean values of these measurements.

The 'intercept' of our prediction function is the mean BFP in our training data.

Predictor coefficients provide a measure—not yet precisely defined—of how sensitive BFP is to predictor deviations.

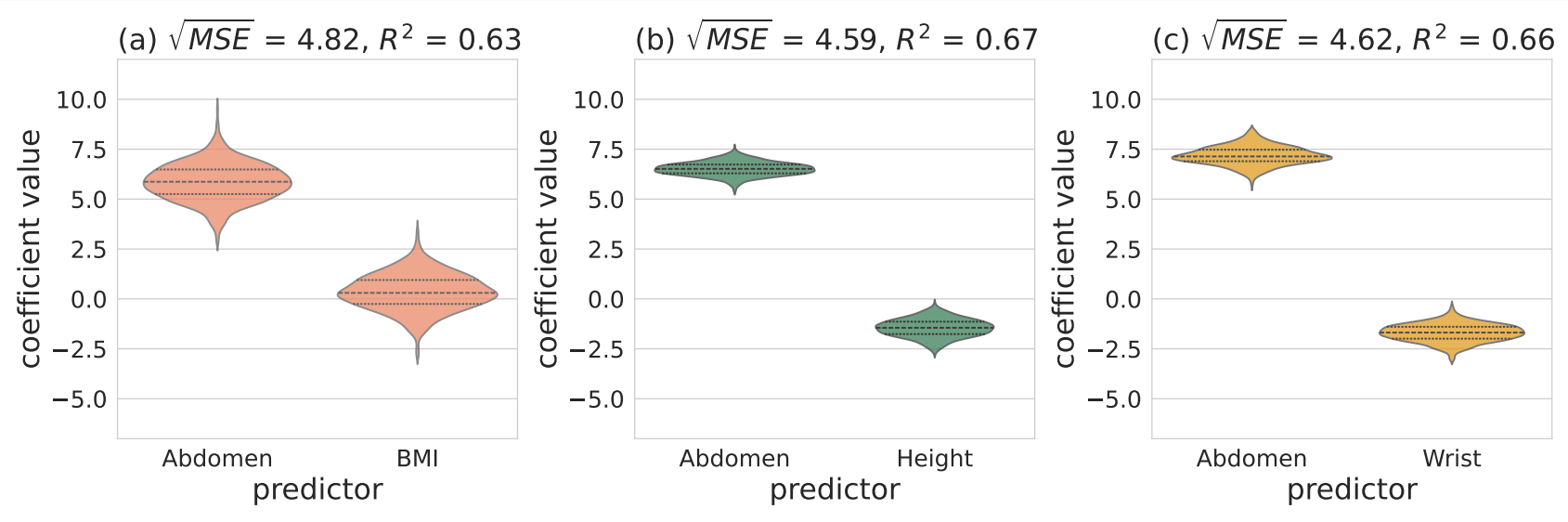



**Figure 28:** Bootstrapped sampling distributions of standardised predictor coefficients for two-predictor models. Plot titles give cross validated $\mathrm{RMSE}$ estimates and $R^2$ values. For reference, the cross validated $\mathrm{RMSE}$ for a linear model using only abdomen is 4.79.

# Observations

- Adding BMI to the model has **reduced** its predictive abilities!
- On its own, BMI was the second best predictor of BFP. Why does combining it with abdomen reduce performance?
- In our two-predictor model, abdomen circumference is doing the **heavy lifting** and very little use is made of BMI, even though they are both highly correlated with BFP. ($\beta^*_{\text{Abd}} \approx 18 \times \beta^*_{\text{BMI}}$ )
- The sampling distribution of $\beta^*_{\text{BMI}}$ is very broad. The **sign** of our estimate is a matter of roughly equal chance depending on noise in the dataset-generation process.
- In the single-predictor setting, predictor coefficients were proportional to the correlation of the predictor with the response. **This is no longer true in two-predictor models.**

# Explanation

We assumed the following conditional probabilistic model:

$$Y = \beta_0^* + \beta_{\text{Abd}}^* Z_{\text{Abd}} + \beta_{\text{BMI}}^* Z_{\text{BMI}} + \mathcal{E},$$

where $\mathcal{E} \sim \mathcal{N}(0, \sigma^2)$. Having observed our predictors we have

$$Y | \{Z_{\text{Abd}} = z_{\text{Abd}}, Z_{\text{BMI}} = z_{\text{BMI}}\} \sim \mathcal{N}(\beta_0^* + \beta_{\text{Abd}}^* z_{\text{Abd}} + \beta_{\text{BMI}}^* z_{\text{BMI}}, \sigma^2).$$

Now suppose we **only observe the abdomen predictor**, then

$$Y = \underbrace{\beta_0^* + \beta_{\text{Abd}}^* z_{\text{Abd}}}_{\text{one-predictor 'intercept'}} + \beta_{\text{BMI}}^* Z_{\text{BMI}} + \mathcal{E}.$$

Viewed as a *one-predictor* model, this describes the relationship between BFP and standardised BMI **within the subset of the population who have a standardised abdomen given by** $z_{\text{Abd}}$.

The coefficient $\beta_{\text{BMI}}^*$ will be proportional to the correlation between BFP and BMI **within this subset of the population**.

# Interpretation

- $\beta^*_{\mathrm{BMI}}$ is the sensitivity of $Y$ to $Z_{\mathrm{BMI}}$ when $Z_{\mathrm{Abd}}$ is held fixed.
- Its small magnitude makes sense—if we already know the abdomen measurement, which we know to be the best single predictor of BFP, then knowing BMI is unlikely to tell us a great deal more.
- We can reverse this interpretation and consider a subset of the population who all have the same BMI.
- Within this group there will be individuals who carry a lot of their weight in the form of muscles, and individuals who carry a lot of their weight as fat, often concentrated around the abdomen.
- Abdomen circumference therefore remains a strong predictor of BFP even if we already know BMI.

## Exercise

Consider the two-predictor model $Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \mathcal{E}$. The **conditional covariance** between $Y$ and $X_1$ given $X_2$ is

$$\mathrm{Cov}(Y, X_1|X_2) = \mathbb{E}(YX_1|X_2) - \mathbb{E}(Y|X_2)\mathbb{E}(X_1|X_2),$$

where $\mathbb{E}(-|X_2)$ denotes expectation conditional on $X_2$. If we observe $X_2 = x_2$ then the conditional expectation is written $\mathbb{E}(-|X_2 = x_2)$. Establish the following:

(a) $\mathbb{E}(YX_1|X_2) = (\beta_0 + \beta_2 X_2)\mathbb{E}(X_1|X_2) + \beta_1 \mathbb{E}(X_1^2|X_2)$.

(b) $\mathbb{E}(Y|X_2) = \beta_0 + \beta_1 \mathbb{E}(X_1|X_2) + \beta_2 X_2$.

(c) $\mathrm{Cov}(Y, X_1|X_2) = \beta_1 \mathrm{Var}(X_1|X_2)$

(d)

$$\beta_1 = \rho(Y, X_1|X_2)\sqrt{\frac{\mathrm{Var}(Y|X_2)}{\mathrm{Var}(X_1|X_2)}}$$

The previous exercise showed that for a two-predictor model, $\beta_1$ is proportional to the conditional correlation between $Y$ and $X_1$ given $X_2$:

$$\beta_1 = \rho(Y, X_1 | X_2 = x_2) \sqrt{\frac{\mathrm{Var}(Y | X_2 = x_2)}{\mathrm{Var}(X_1 | X_2 = x_2)}}.$$

This relationship may be generalised to the case of a linear model with $p$ predictors,

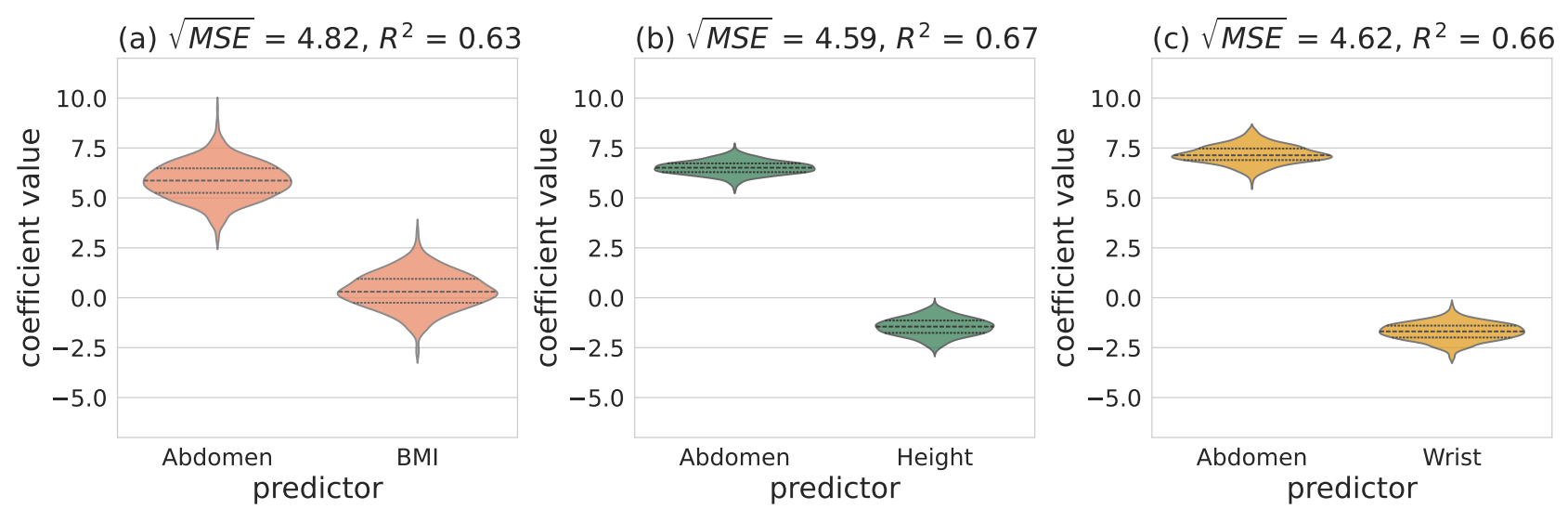
$$Y = \beta_0 + \sum_{k=1}^{p} \beta_k X_k + \mathcal{E}.$$

Let $\boldsymbol{X}_{-k}$ denote the vector of predictors excluding $X_k$. We then have

$$\beta_k = \frac{\mathrm{Cov}(Y, X_k | \boldsymbol{X}_{-k} = \boldsymbol{x}_{-k})}{\mathrm{Var}(X_k | \boldsymbol{X}_{-k} = \boldsymbol{x}_{-k})}.$$

# Colinearity

Consider the sampling distribution of our Abdomen-BMI model.




Compared to the two alternative two-predictor models—(b) and (c) above—standard errors in the abdomen-BMI model are large.

While abdomen and BMI are individually good predictors of BFP, they also exhibit a comparatively high level of *collinearity*. That is, they have a strong linear relationship (and a high correlation).

The extent of colinearity between predictors is shown in Figure 29.

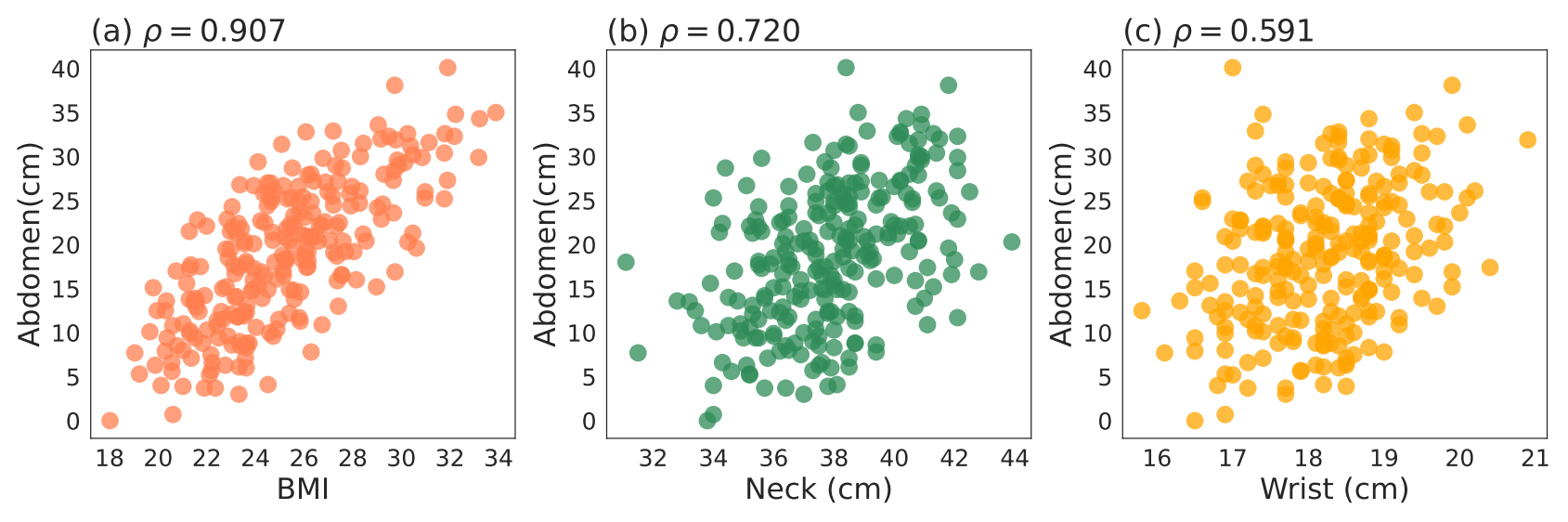



**Figure 29:** Scatter plots of the predictor pairs used for the two-predictor models.

Strongly collinear predictors tend to increase or decrease together, so using them in combination typically yields minimal improvements in predictive performance. It is also difficult to determine their respective contribution to variations in the response.

# Selecting predictors

- Combining the most effective **individual** predictors is not necessarily the best way to improve predictive accuracy.
- We need predictor combinations which **complement** each other.
- We want strong **individual** associations with the response variable but weak associations **between** predictors.

Height and wrist have substantially lower individual correlations with BFP than does BMI.

However, in combination with abdomen they achieve lower prediction errors ($\mathrm{RMSE} = 4.59$ for Abdomen-Neck vs $\mathrm{RMSE} = 4.82$ for Abdomen-BMI), and smaller standard errors in their coefficients.

# The all-predictor model

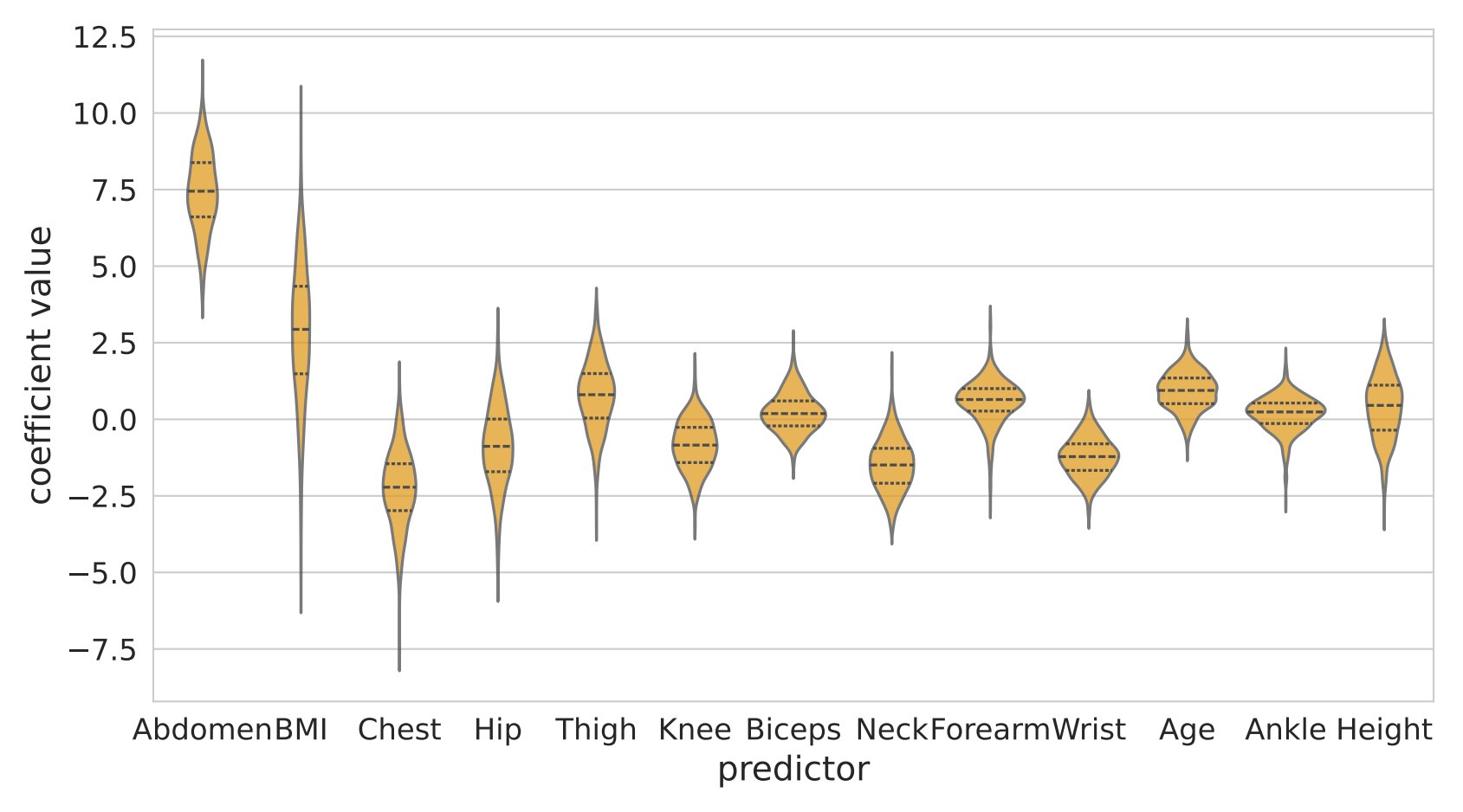



**Figure 30:** Sampling distributions for the all-predictor model. The cross validated error estimate for this model is $\mathrm{RMSE} = 4.822$.

# Overfitting

- The all-predictor model the **highest RMSE so far**.
- If we want to optimise predictive performance we should **not use all the predictors**.
- Standard errors are also the largest we have seen.
- When more predictors are available, there are more ways to explain the pattern of responses in terms of them.
- We have **less certainty** about each possible explanation, and greater scope for **overfitting**.

The $R^2$ values for this all-predictor model are

$$R^2 = 0.702$$
$$R^2_{\text{test}} = 0.525.$$

$R^2_{\text{test}} \ll R^2$ is a sign of overfitting.

# Best subset selection

- The **optimal model** will use a **strict subset** of the predictors.
- One way to search for this **best subset** is to test all of the $2^{13} = 8192$ possible predictor combinations.
- This is feasible using the current dataset (but still time consuming) because fitting linear models can be done very efficiently.
- However, the **exponential relationship** between the number of possible models and the number of predictors quickly makes **exhaustive search impossible**.

# Forward stepwise selection

Suppose we label predictors using the set of integers

$$S = \{1, 2, \ldots, p\}.$$

The aim is to construct a sequence of predictor sets $P_0, P_1, P_2, \ldots, P_p$ where $P_k$ is our guess at the best set of $k$ predictors, and $P_0 = \{\}$.

Let $\mathrm{MSE}(P)$ be the cross validated $\mathrm{MSE}$ of the model with predictor set $P$. Given $P_k$ we first find the extra predictor which when added to $P_k$ produces the smallest error:

$$\hat{i} = \underset{i \in S \setminus P_k}{\textbf{arg min}}\ \mathrm{MSE}\left(P_k \cup \{i\}\right).$$

We then add $\hat{i}$ to $P_k$, to form $P_{k+1}$

$$P_{k+1} = P_k \cup \{\hat{i}\}.$$

Having constructed the sequence of 'best' predictor sets of each size, $P_0, P_1, P_2, \ldots, P_p$, we find the 'best' overall predictor set

$$\hat{P} = \operatorname*{arg\,min}_{P_k} \mathrm{MSE}(P_k).$$

Without exhaustive search we cannot *guarantee* to find the best subset.

The idea of stepwise selection is to *efficiently* find a model which, even if it is not strictly optimal, still comes close.

**Exercise**

If we have $p$ predictors, show that the number of models we need to test using stepwise selection is $1 + p(p+1)/2$ versus $2^p$ for exhaustive search. For example, when $p = 13$ this means performing 92 tests versus 8192.

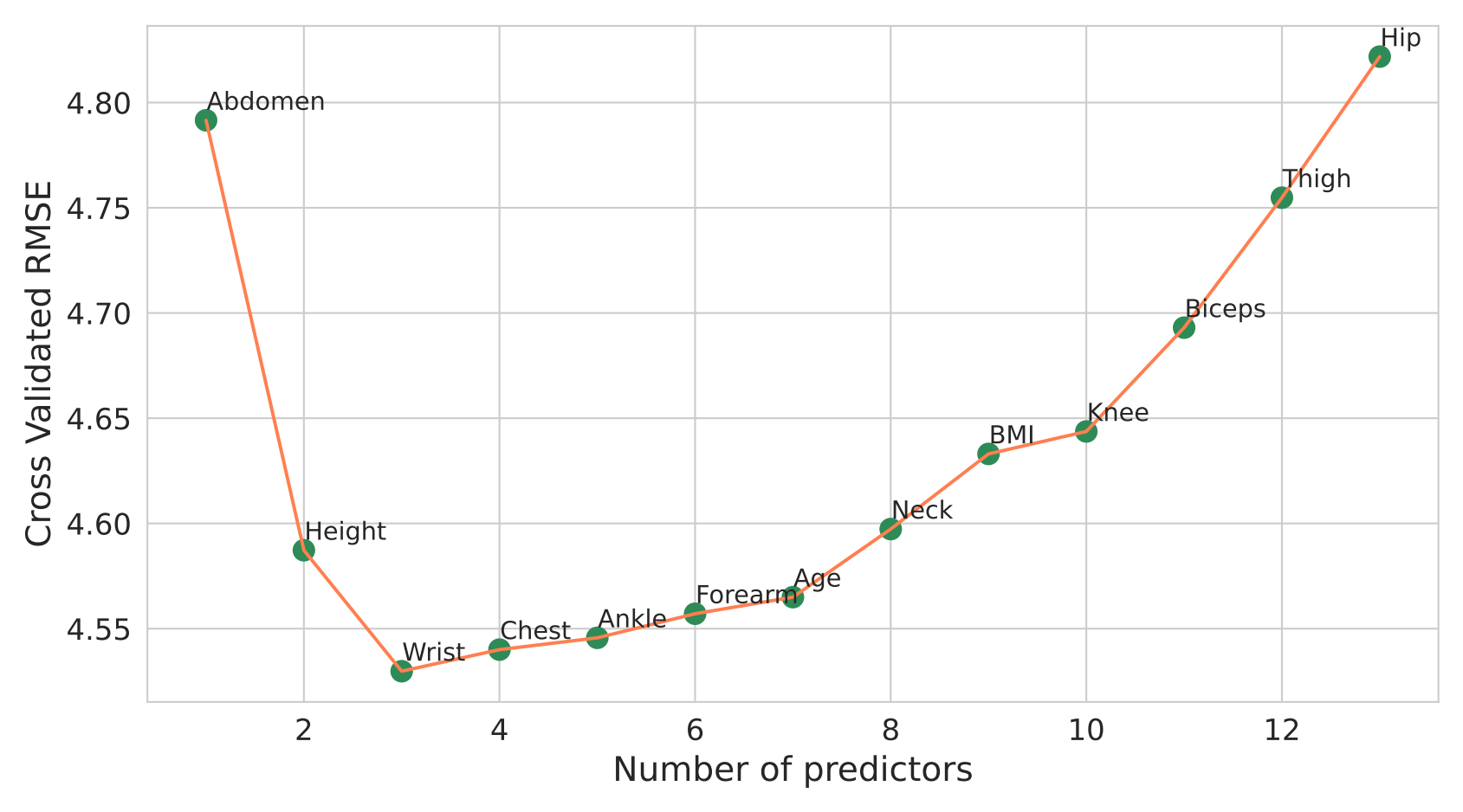



**Figure 31:** Cross validated root mean squared errors for models using predictor sets determined by forward stepwise selection. Points are labeled with the additional predictor added to the previous subset.

# Bias-variance trade off

Model **complexity** can be manipulated in many different ways.

Plots of predictive performance against model complexity typically produce **U-shaped curves**.

We see this U-shape because too little complexity leads to **under-fitting** while too much complexity leads to **over-fitting**, both of which lead to poor performance.

Finding the sweet spot between under- and over-fitting may be understood in a precise mathematical way as balancing two distinct sources of error: **bias** and **variance**.

# Bias-variance trade off for prediction functions

Given a dataset $D = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$, suppose we want to build a prediction function of the form $g(\boldsymbol{x}; \boldsymbol{\theta})$.

We estimate $\boldsymbol{\theta}$ by minimising the total squared error loss,

$$\mathcal{L}(\boldsymbol{\theta}; D) = \sum_{i=1}^{n} (g(\boldsymbol{x}_i, \boldsymbol{\theta}) - y_i)^2.$$

Minimising this loss defines an estimator function, $\boldsymbol{t}$, which maps datasets to parameter estimates:

$$\hat{\boldsymbol{\theta}} = \boldsymbol{t}(D) = \arg\min_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}; D).$$

We view $D$ as a single realisation of a **random** dataset $\mathcal{D} = \{(\boldsymbol{X}_i, Y_i)\}_{i=1}^n$, where $Y = \mathfrak{g}(\boldsymbol{X}) + \mathcal{E}$.

Repeating the data generation process would produce a sequence of realisations of the random vector $\hat{\boldsymbol{\Theta}} = \boldsymbol{t}(\mathcal{D})$.

To measure prediction performance we average over both the **test data** and over variations in models due to the **training data**.

We first define a predictor random variable which depends on the training dataset:

$$\hat{Y} = g(x, t(\mathcal{D})).$$

The squared test error is

$$(Y - \hat{Y})^2 = (\mathfrak{g}(x) + \mathcal{E} - g(x; t(\mathcal{D})))^2.$$

Notice that there are two independent sources of randomness here: the **training** dataset $\mathcal{D}$ and the **test** noise variable $\mathcal{E}$.

Overall performance is the average squared test error over $\mathcal{D}$ **and** $\mathcal{E}$. This is known as the **risk**,

$$\mathrm{RISK}(x) = \mathbb{E}\left((Y - \hat{Y})^2\right) = \mathbb{E}\left((g(x; t(\mathcal{D})) - \mathfrak{g}(x) - \mathcal{E})^2\right).$$

We first calculate the expected squared error conditional on $\mathcal{D}$

**Exercise**

Making use of the fact that $\mathbb{E}(\mathcal{E}) = 0$ and $\mathbb{E}(\mathcal{E}^2) = \sigma^2$, and that $\mathcal{E}$ is independent of the training data, establish that

$$\mathbb{E}\left((g(x; t(\mathcal{D})) - \mathfrak{g}(x) - \mathcal{E})^2 | \mathcal{D}\right) = (g(x; t(\mathcal{D})) - \mathfrak{g}(x))^2 + \sigma^2.$$

We can now compute the risk by averaging this test $\mathrm{MSE}$ over $\mathcal{D}$:

$$\mathrm{RISK}(x) = \underbrace{\mathbb{E}\left((g(x; t(\mathcal{D})) - \mathfrak{g}(x))^2\right)}_{\text{Reducible}} + \underbrace{\sigma^2}_{\text{Irreducible}}.$$

**Reducible** errors are affected by the mathematical form of $g(x, \theta)$, and therefore it is possible to reduce them using good modelling choices.

**Irreducible** error derives from the noise $\mathcal{E}$, which we cannot control.

# Decomposing reducible error

The reducible error can be decomposed into two further terms, each of which captures a distinctive source of reducible error.

We first define the average value of the prediction function over all possible training sets:

$$\bar{g}(x) = \mathbb{E}\left(g(x; t(\mathcal{D})\right) = \mathbb{E}\left(g(x; \hat{\boldsymbol{\Theta}})\right).$$

Here we have made use of the fact that averaging over the training data is equivalent to averaging over the sampling distribution of the parameter vector. We then have

$$\text{RISK}(x) = \underbrace{\mathbb{E}\left((g(x; \hat{\boldsymbol{\Theta}}) - \bar{g}(x))^2\right)}_{\text{Variance}} + \underbrace{(\bar{g}(x) - \mathfrak{g}(x))^2}_{\text{Bias}^2} + \underbrace{\sigma^2}_{\text{Noise}}.$$

# Visualising the bias-variance trade off

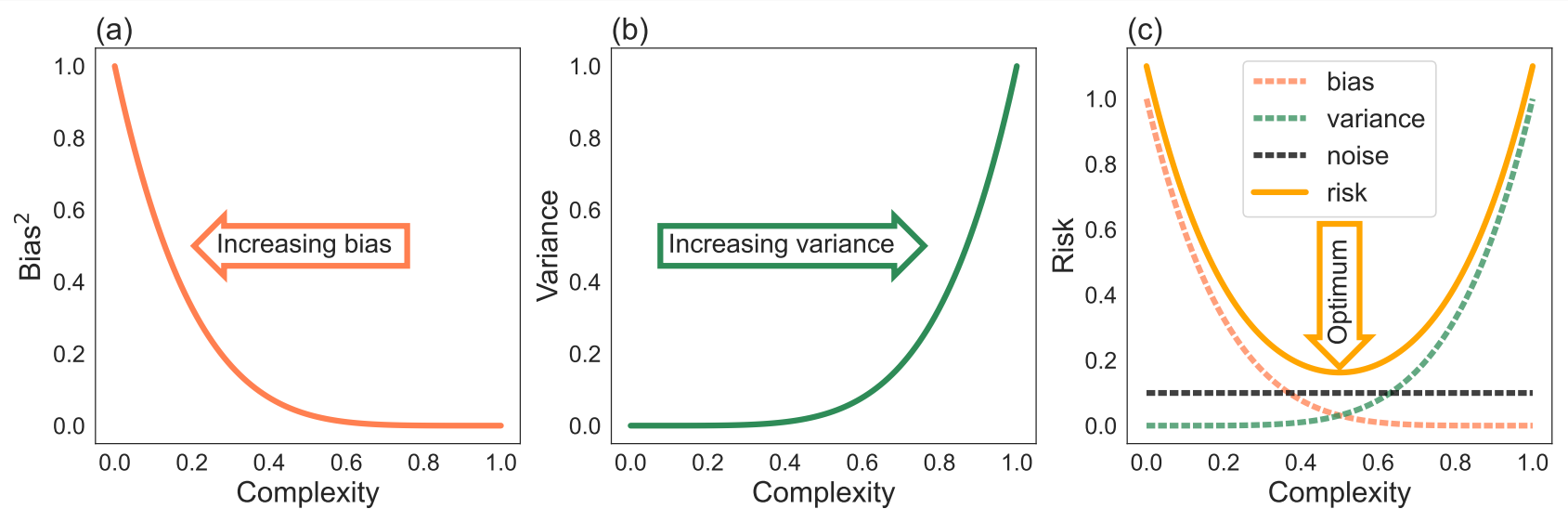



**Figure 32:** The bias-variance trade off.

# Calculation details

## Exercise

The risk of a prediction model may be written

$$\mathrm{RISK}(x) = \mathbb{E}\left((g(x;\hat{\boldsymbol{\Theta}}) - \mathfrak{g}(x))^2\right) + \sigma^2.$$

We can decompose the expectation on the right-hand side into a variance term and a bias term. To simplify intermediate steps in our calculation we will use the abbreviations

$$\hat{g} = g(x;\hat{\boldsymbol{\Theta}}), \quad \bar{g} = \bar{g}(x), \quad \mathfrak{g} = \mathfrak{g}(x)$$

and note that $\hat{g}$ is a random variable satisfying $\mathbb{E}(\hat{g}) = \bar{g}$, whereas $\bar{g}$ and $\mathfrak{g}$ are non-random. Show that

$$\begin{aligned}\mathbb{E}\left((g(x;\hat{\boldsymbol{\Theta}}) - \mathfrak{g}(x))^2\right) &= \mathbb{E}\left((\hat{g} - \bar{g}) + (\bar{g} - \mathfrak{g}))^2\right) \\ &= \mathbb{E}\left((\hat{g} - \bar{g})^2\right) + (\bar{g} - \mathfrak{g})^2\end{aligned}$$

# Directed Acyclic Graphs

# Why directed acyclic graphs?

- *Directed acyclic graphs* (DAGs) are a useful tool for representing conditional probabilistic relationships between random variables.
- DAGs are sometimes referred to as *Bayesian networks*, but they are not essentially Bayesian.
- DAGs can describe statistical models (whether Bayesian or frequentist), clarify dependencies, and support causal reasoning.

# Graph basics

- A *graph* is a set of *nodes* and a set of *edges* connecting pairs of nodes.
- Connected nodes are said to be *adjacent*, and a *path* through a graph is a sequence of adjacent nodes.
- If edges have a direction they are called arrows, and the graph is said to be *directed*.
- A path which follows arrow directions is a *directed path*.
- A directed path which starts and ends in the same place is a *directed cycle*.
- A graph with **directed edges** but **no directed cycles** is a **directed acyclic graph**.

# Conditional distributions, marginals, and independence

- For continuous random variables, the *conditional probability density* function of $Y$ given the occurrence of the value of $X$ can be written as

$$f_{Y|X}(y|x) = \frac{f_{X,Y}(x,y)}{f_X(x)},$$

since $f_{X,Y}(x,y) = f_X(x)\, f_{Y|X}(y|x)$.

- If $f_Y(y) = f_{Y|X}(y|x)$, then $X$ and $Y$ are called *marginally independent*, written $X \perp\!\!\!\perp Y$. This means that

$$f_{X,Y}(x,y) = f_X(x) f_Y(y).$$

- A *marginal probability density* is obtained from a joint density by integrating the latter over all variables but one. From the joint density $f_{X,Y}(x,y)$ we can define two marginal densities:

$$f_X(x) = \int f_{X,Y}(x,y)\, dy \quad \text{and} \quad f_Y(y) = \int f_{X,Y}(x,y)\, dx.$$

# A DAG with three nodes

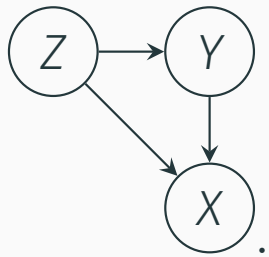
Consider three continuous random variables $X, Y, Z$ with joint density $f_{X,Y,Z}(x, y, z)$. We have

$$f_{X|Y,Z}(x|y, z) = \frac{f_{X,Y,Z}(x, y, z)}{f_{Y,Z}(y, z)},$$
$$f_{Y|Z}(y|z) = \frac{f_{Y,Z}(y, z)}{f_Z(z)},$$

and so the joint *factorises* into a **product of univariate densities**:

$$f_{X,Y,Z}(x, y, z) = f_{X|Y,Z}(x|y, z) f_{Y|Z}(y|z) f_Z(z).$$

Each factor is a density of *one* variable, conditional on zero or more others (called 'parents'). The factorisation can be represented by a DAG with *nodes* for random variables and *arrows* from parents to children:



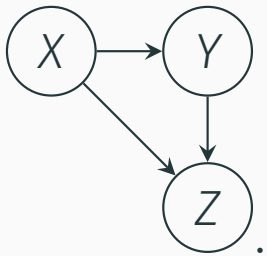

.

# A DAG with three nodes – other factorisations

The factorisation of $f_{X,Y,Z}(x, y, z)$ into $f_{X|Y,Z}(x|y, z)f_{Y|Z}(y|z)f_Z(z)$ is **not unique**. Another factorization,

$$f_{X,Y,Z}(x, y, z) = f_{Z|Y,X}(z|y, x)f_{Y|X}(y|x)f_X(x),$$

leads to the DAG




.

**Exercise**

Find one further factorisation of $f_{X,Y,Z}(x, y, z)$ into a product of univariate conditional densities and draw the corresponding DAG.

# No restriction on dependency

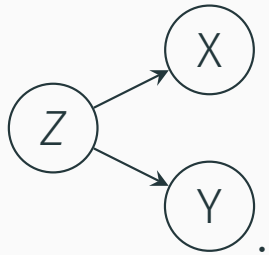
Suppose that the density $f_{X|Y,Z}(x|y,z)$ does not in fact depend on $y$, so that

$$f_{X|Y,Z}(x|y,z) = f_{X|Z}(x|z).$$

Then we can factorise the joint as follows:

$$f_{X,Y,Z}(x,y,z) = f_{X|Z}(x|z)f_{Y|Z}(y|z)f_Z(z),$$

giving the DAG




.

The previous DAG encodes the assumption that $X$ and $Y$ are *conditionally independent* given $Z$, written $X \perp\!\!\!\perp Y | Z$. That is, the joint density of $X$, $Y$ and $Z$ factorises as though all statistical dependence between $X$ and $Y$ is attributable to their individual relationships with $Z$.

**Exercise**

Let $W, X, Y, Z$ be random variables. Suppose that their joint density admits the factorisation

$$f_{W,X,Y,Z}(w, x, y, z) = f_{W|Y}(w|y) f_{X|Y}(x|y) f_{Y|Z}(y|z) f_Z(z).$$

Draw the corresponding DAG.

**Exercise**

Suppose that $X$, $Y$ and $Z$ are mutually independent so $f_{X,Y,Z}(x, y, z) = f_X(x) f_Y(y) f_Z(z)$. Draw the corresponding DAG.

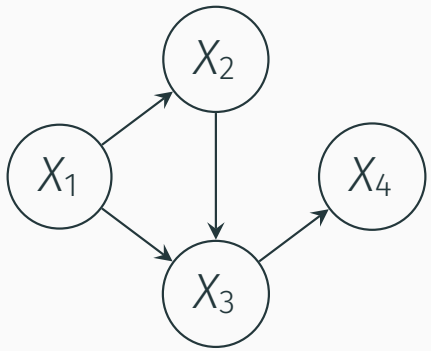
# One-to-one correspondence between factorisations and DAGs

Given a factorisation of a multivariate density into univariate ones, there is a **unique** DAG representation; the **reverse** is also true.

Let a DAG have nodes $X_1, \ldots, X_n$ and let $\boldsymbol{PA}_k$ be the set of parent nodes of $X_k$. The corresponding factorisation is

$$f_{X_1,\ldots,X_n}(x_1, \ldots, x_n) = \prod_{k=1}^{n} f_{X_k|\boldsymbol{PA}_k}(x_k|\boldsymbol{pa}_k).$$

For example, the DAG

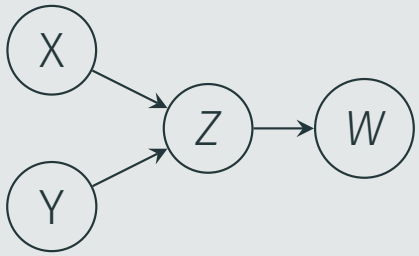



corresponds to the factorisation

$$f_{X_1,X_2,X_3,X_4}(x_1, x_2, x_3, x_4) = f_{X_1}(x_1) f_{X_2|X_1}(x_2|x_1) f_{X_3|X_1,X_2}(x_3|x_1, x_2) f_{X_4|X_3}(x_4|x_3).$$

**Exercise**

Let $W, X, Y, Z$ be random variables. Write down the factorisation of their multivariate density corresponding to the DAG




## Remarks

- DAGs are used to describe multivariate probability distributions in terms of conditional distributions.
- The results for DAG also hold for discrete random variables if we replace densities with mass functions.

# Conditional independence in DAGs

- A DAG in which **every pair of nodes is connected** is called *fully connected*.
- A fully connected DAG represents a **completely arbitrary** multivariate distribution; its structure implies no constraints on the dependencies between variables.
- Removing edges from such a DAG introduces constraints on the dependencies between variables.

**Question**: Given a DAG how do we determine all the conditional independence relationships implied by its structure?

# Conditional independence

- Two continuous random variables $X$ and $Y$ are *marginally independent*, written as $X \perp\!\!\!\perp Y$, if
$$f_{X,Y}(x, y) = f_X(x) f_Y(y).$$
- We say that $X$ and $Y$ are *conditionally independent* given $Z$, written as $X \perp\!\!\!\perp Y|Z$, if
$$f_{X,Y|Z}(x, y|z) = f_{X|Z}(x|z) f_{Y|Z}(y|z).$$
- If $\boldsymbol{A}, \boldsymbol{B}$ and $\boldsymbol{C}$ are disjoint sets of random variables (represented as random vectors), then we write $\boldsymbol{A} \perp\!\!\!\perp \boldsymbol{B}|\boldsymbol{C}$ if
$$f_{\boldsymbol{A},\boldsymbol{B}|\boldsymbol{C}}(\boldsymbol{a}, \boldsymbol{b}|\boldsymbol{c}) = f_{\boldsymbol{A}|\boldsymbol{C}}(\boldsymbol{a}|\boldsymbol{c}) f_{\boldsymbol{B}|\boldsymbol{C}}(\boldsymbol{b}|\boldsymbol{c}).$$
- Independence is a special case of conditional independence
$$X \perp\!\!\!\perp Y|\emptyset \equiv X \perp\!\!\!\perp Y,$$
where $\emptyset$ is the empty set.

The following exercise aims to establish that $f_{X|Y,Z}(x|y,z) = f_{X|Z}(x|z)$ implies that $X \perp\!\!\!\perp Y|Z$, and vice versa.

**Exercise**

(a) Show that

$$f_{X|Y,Z}(x|y,z) = \frac{f_{X,Y|Z}(x,y|z)}{f_{Y|Z}(y|z)}.$$

(b) Use the previous result to show that $f_{X|Y,Z}(x|y,z) = f_{X|Z}(x|z)$ implies that

$$f_{X,Y|Z}(x,y|z) = f_{X|Z}(x|z) f_{Y|Z}(y|z).$$

(c) Show that the converse implication also holds.

Given a DAG, we can systematically discover conditional independence relationships using a concept called *d-separation*. To motivate this concept we'll first consider three simple graphs, each with three nodes: the *pipe*, the *fork* and the *collider*.

# Pipe

Here is a pipe:


It corresponds to the factorisation

$$f_{X,Y,Z}(x,y,z) = f_X(x) f_{Y|X}(y|x) f_{Z|Y}(z|y),$$

which implies that $X \perp\!\!\!\perp Z|Y$. To see the implication, write down the joint density of $X$ and $Z$ conditional on $Y$:

$$f_{X,Z|Y}(x,z|y) = \frac{f_{X,Y,Z}(x,y,z)}{f_Y(y)} = \frac{f_{X,Y}(x,y) f_{Z|Y}(z|y)}{f_Y(y)} = f_{X|Y}(x|y) f_{Z|Y}(z|y).$$

As practical example of the pipe, suppose that Hrothgar sings a single musical note to Hilda, out of earshot of Godiva. Some time later, Hilda attempts to sing the same note to Godiva, who attempts to copy it. If $X$, $Y$ and $Z$ are the pitches of the notes sung by Hrothgar, Hilda and Godiva, then we might reasonably expect that $X \perp\!\!\!\perp Z|Y$.

# Fork

Here is a fork:


.

It corresponds to the factorisation

$$f_{X,Y,Z}(x,y,z) = f_Y(y) f_{X|Y}(x|y) f_{Z|Y}(z|y).$$

Since $f_{X,Y,Z}(x,y,z) = f_Y(y) f_{X,Z|Y}(x,z|y)$, the above factorisation implies that

$$f_{X,Z|Y}(x,z|y) = f_{X|Y}(x|y) f_{Z|Y}(z|y),$$

and hence that $X \perp\!\!\!\perp Z|Y$.

In both the pipe and the fork, $Y$ is said to *block* the path between $X$ and $Z$. Informally, conditioning on $Y$ breaks any statistical dependence that might otherwise be present between $X$ and $Z$ due to their mutual connection with $Y$.

# Collider

Here is a collider:


.

It corresponds to the factorisation

$$f_{X,Y,Z}(x,y,z) = f_X(x)f_Z(z)f_{Y|X,Z}(y|x,z).$$

This factorisation implies that $X$ and $Z$ are *marginally independent*, i.e., $X \perp\!\!\!\perp Z$, since

$$f_{X,Z}(x,z) = \int f_{X,Y,Z}(x,y,z)dy = f_X(x)f_Z(z)\int f_{Y|X,Z}(y|x,z)dy = f_X(x)f_Z(z).$$

By contrast, there is no guarantee that the density $f_{X,Z|Y}(x,z|y)$ factorises into the product $f_{X|Y}(x|y)f_{Z|Y}(z|y)$, so we cannot deduce from the collider DAG that $X \perp\!\!\!\perp Z|Y$. In fact, the generic collider case is that $X$ and $Z$ are *conditionally dependent* given $Y$.

# More on a collider

The middle node of a collider behaves in the **opposite** way to the middle node of a pipe or fork. Node $Y$

- *blocks* the path from $X$ to $Z$ when it is *not* conditioned on, but
- *unblocks* the path from $X$ to $Z$ when it *is* conditioned on.

As a practical example, suppose that the sales revenue, $Y$, of a musical artist is the sum talent $X$, beauty $Z$, and luck $\mathcal{E}$:

$$Y = X + Z + \mathcal{E}.$$

Suppose $X$, $Z$ and $\mathcal{E}$ are bestowed upon musicians independently. We see that for a given size of revenue, a more beautiful artist will tend to be less talented and vice versa. Beauty and talent are therefore conditionally dependent given revenue.

- Suppose that the middle node $Y$ of a collider is the parent of one or more nodes which may be the parents of further nodes and so on, forming a set of *descendants* of $Y$, written $\boldsymbol{DESC}(Y)$.
- A collider becomes unblocked if *$Y$ or any of its descendants* are conditioned on.

## Summary

- Conditioning on the middle node of a pipe or a fork blocks the path, making the end nodes conditionally independent.
- Conditioning on the middle node of a collider (or any of its descendants) unblocks the path, making the end nodes conditionally dependent.

# d-separation and conditional independence

Generalising from our pipe, fork and collider examples, the concept of d-separation allows us to identify the conditional independence relationships implied by any DAG.

## d-separation and conditional independence

Suppose that $\boldsymbol{A}$, $\boldsymbol{B}$ and $\boldsymbol{C}$ are three disjoint sets of nodes in a DAG. An undirected path, $P$, from a node in $\boldsymbol{A}$ to a node in $\boldsymbol{B}$ is said to be blocked by $\boldsymbol{C}$ if *at least one* of the following conditions are satisfied

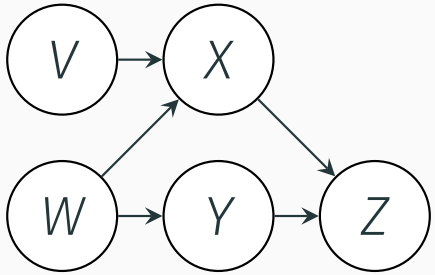
(a) $P$ contains a pipe $X \rightarrow Y \rightarrow Z$ or $X \leftarrow Y \leftarrow Z$ where $Y \in \boldsymbol{C}$.

(b) $P$ contains a fork $X \leftarrow Y \rightarrow Z$ where $Y \in \boldsymbol{C}$.

(c) $P$ contains a collider $X \rightarrow Y \leftarrow Z$ where $Y \notin \boldsymbol{C}$ and $\boldsymbol{DESC}(Y) \cap \boldsymbol{C} = \emptyset$.

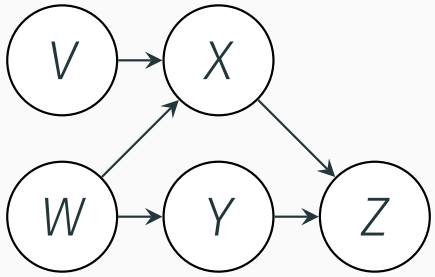
If all paths from $\boldsymbol{A}$ to $\boldsymbol{B}$ are blocked by $\boldsymbol{C}$, then $\boldsymbol{A}$ and $\boldsymbol{B}$ are said to be d-separated by $\boldsymbol{C}$; in a distribution that factorises according to the DAG, this implies $\boldsymbol{A} \perp\!\!\!\perp \boldsymbol{B} | \boldsymbol{C}$.

# An example




We show that $Y \perp\!\!\!\perp V | \{W, X\}$ by considering **all paths** between $Y$ and $V$.

1. The path $Y \to \underbrace{Z \leftarrow X \leftarrow V}_{\text{pipe}}$ contains a pipe whose central node is in the conditioning set, so it is blocked. The path also contains collider $Y \to Z \leftarrow X$; since $Z$ is not in $\{W, X\}$ and has no descendants, this also suffices to block the path.
2. The path $\underbrace{Y \leftarrow W \to X}_{\text{fork}} \leftarrow V$ contains a fork whose central node is in the conditioning set, so it too is blocked. There is also a collider $W \to X \leftarrow V$ whose central node *is* in the conditioning set, but this does not 'unblock' the path. (A path is blocked as soon as *any segment* is blocked.)




Another example: $V \perp\!\!\!\perp W|\emptyset$, since

- $V \to X \leftarrow W$ is a collider and $X \notin \emptyset$.
- $X \to Z \leftarrow Y$ is a collider in path $V \to X \to Z \leftarrow Y \leftarrow W$ and $Z \notin \emptyset$.

**Exercise**

For the above DAG, use d-separation to establish the following conditional independence relations, explaining your reasoning in each case.

(a) $X \perp\!\!\!\perp Y|W$.

(b) $W \perp\!\!\!\perp Z|\{X, Y\}$.

(c) $V \perp\!\!\!\perp Y|\emptyset$.

# Bayesian Regression and Regularisation

- Regularisation reduces overfitting by penalising model complexity.
- We will look at *ridge* and *lasso* penalties. These can be understood as implicit Bayesian priors on model parameters.
  - Consider a simple linear regression model,

    $$Y = \beta_0 + \beta_1 X + \mathcal{E} \text{ with } \mathcal{E} \sim \mathcal{N}(0, \sigma^2),$$

    where the parameter vector $\boldsymbol{\theta} = (\beta_0, \beta_1, \sigma)$ is fixed but *unknown*.
  - In Bayesian world, the **parameters form a random vector**

    $$\boldsymbol{\Theta} = (B_0, B_1, \Sigma).$$

  - So the Bayesian linear regression model contains random variables $X, Y, B_0, B_1, \mathcal{E}$ and $\Sigma$, where

    $$Y = B_0 + B_1 X + \mathcal{E} \text{ with } \mathcal{E} \sim \mathcal{N}(0, \Sigma^2).$$

# Simple Bayesian linear regression

$$Y = B_0 + B_1 X + \mathcal{E}$$

- Bayesian regression specifies the relationship between $Y, X$ and $\mathcal{E}$ conditional on the event $\boldsymbol{\Theta} = (\beta_0, \beta_1, \sigma)$.
- We separate the coefficients $B_0, B_1$ from others to form a vector $\boldsymbol{B} = (B_0, B_1)$.
- The density of $Y$ conditional on $\boldsymbol{B}$, $\boldsymbol{\Sigma}$ and $X$ is

$$f_{Y|\boldsymbol{B},\boldsymbol{\Sigma},X}(y|\boldsymbol{\beta}, \sigma, x) = \mathcal{N}(y|\beta_0 + \beta_1 x, \sigma^2),$$

  where $\mathcal{N}(y|\mu, \sigma^2)$ is the normal probability density function with mean $\mu$ and variance $\sigma^2$.
- **The goal**: combine *priors* on $\boldsymbol{B}$ and $\boldsymbol{\Sigma}$ with a *dataset* of predictor-response observations $\{(x_i, y_i)\}_{i=1}^n$ to *infer posterior parameter distributions*.

# Factorisation and the DAG

By the standard factorisation,

$$f_{Y,B,\Sigma,X}(y,\beta,\sigma,x) = f_{Y|B,\Sigma,X}(y|\beta,\sigma,x) f_{B,\Sigma|X}(\beta,\sigma|x) f_X(x),$$

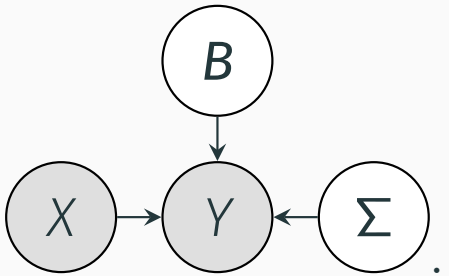
where $f_{B,\Sigma|X}(\beta,\sigma|x)$ is the prior parameter distribution given the predictor $X$.

We assume that $B$ and $\Sigma$ are independent of $X$ and of each other:

$$f_{B,\Sigma|X}(\beta,\sigma|x) = f_B(\beta) f_\Sigma(\sigma).$$

Therefore,

$$f_{Y,B,\Sigma,X}(y,\beta,\sigma,x) = f_{Y|B,\Sigma,X}(y|\beta,\sigma,x) f_B(\beta) f_\Sigma(\sigma) f_X(x).$$



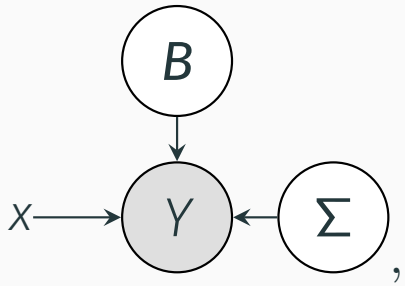

We shade variables that can be observed to distinguish them from *latent* variables—the model parameters—which cannot be observed.

# A constant node

The predictor variable $X$ is observed (or fixed) both when collecting training data and when making predictions. The conditional density of the remaining variables given $X$ is

$$f_{Y,B,\Sigma|X}(y, \beta, \sigma|x) = f_{Y|B,\Sigma,X}(y|\beta, \sigma, x) f_B(\beta) f_\Sigma(\sigma).$$

To represent this factorisation graphically we introduce a *constant node*, $x$, which is not circled, representing *known* quantities. The new DAG is



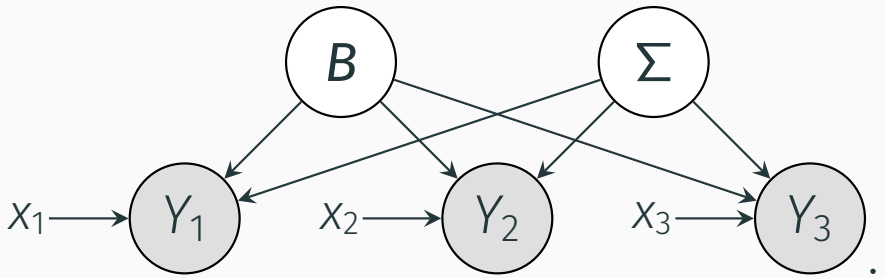

representing a conditional probabilistic model of the parameters and response variable given the predictor.

# Adding the dataset

We now add the dataset, $\underline{X} = (X_1, \ldots, X_n)$ and $\underline{Y} = (Y_1, \ldots, Y_n)$. Assuming that the noise variables are conditionally independent given the predictors, the response variables are too. Therefore

$$f_{\underline{Y},B,\Sigma|\underline{X}}(\underline{y}, \beta, \sigma|\underline{X}) = f_B(\beta) f_\Sigma(\sigma) \prod_{i=1}^{n} f_{Y|B,\Sigma,X}(y_i|\beta, \sigma, x_i)$$
$$= \underbrace{f_B(\beta) f_\Sigma(\sigma)}_{\text{prior}} \underbrace{\mathcal{L}(\beta, \sigma)}_{\text{likelihood}}.$$

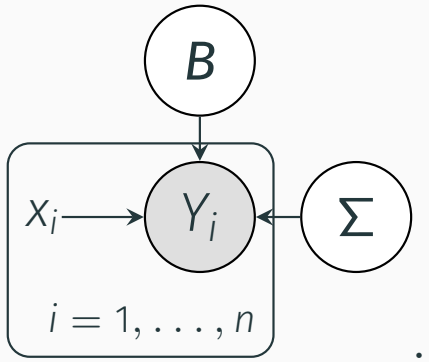
We assume that the model parameters are sampled only once, and so the prior densities only appear once in the factorisation. As an illustration, the DAG representation when $n = 3$ is




.

# The posterior parameter distribution

For arbitrary $n$, the DAG can represented as




.

The *posterior parameter distribution* is

$$f_{B,\Sigma|\underline{Y},\underline{X}}(\boldsymbol{\beta}, \sigma|\underline{y}, \underline{X}) = \frac{f_{\underline{Y},B,\Sigma|\underline{X}}(\underline{y}, \boldsymbol{\beta}, \sigma|\underline{X})}{f_{\underline{Y}|\underline{X}}(\underline{y}|\underline{X})} \propto \mathcal{L}(\boldsymbol{\beta}, \sigma) f_B(\boldsymbol{\beta}) f_\Sigma(\sigma).$$

- $f_{\underline{Y}|\underline{X}}(\underline{y}|\underline{X})$ is the marginal distribution of the responses given the predictors.
- Overall, we have another instance of the familiar pattern

$$\text{Posterior} \propto \text{Likelihood} \times \text{Prior}.$$

# Generalising to multiple predictors

- For $p > 1$ predictor variables, the model is

$$Y = B_0 + \sum_{k=1}^{p} B_k X_k + \mathcal{E} \text{ with } \mathcal{E} \sim \mathcal{N}(0, \sigma^2).$$

- Using the vector notation, it is

$$Y = \boldsymbol{X}^T \boldsymbol{B} + \mathcal{E} \text{ with } \mathcal{E} \sim \mathcal{N}(0, \sigma^2),$$

  where $\boldsymbol{X} = (1, X_1, \ldots, X_p)^T$ and $\boldsymbol{B} = (B_0, B_1, \ldots, B_p)^T$ are two (column) vectors of dimension $p + 1$.

- The conditional distribution of $Y$ is normal:

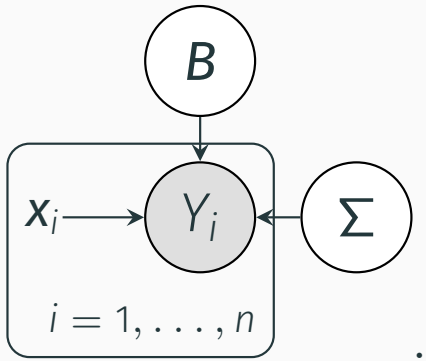
$$f_{Y|\boldsymbol{B},\Sigma,\boldsymbol{X}}(y|\boldsymbol{\beta}, \sigma, \boldsymbol{x}) = \mathcal{N}(y|\boldsymbol{x}^T\boldsymbol{\beta}, \sigma^2).$$

# The model and DAG

The factorisation of the conditional density of the response and the parameters given the predictor vector is

$$f_{Y,B,\Sigma|X}(y, \beta, \sigma|x) = f_{Y|B,\Sigma,X}(y|\beta, \sigma, x) f_B(\beta) f_\Sigma(\sigma).$$

The dataset is an observation of a sequence of predictor vectors $X_1, \ldots, X_n$ and responses $Y_1, \ldots, Y_n$. Under the assumption of i.i.d. realisations of the noise, the DAG is




.

The set of all predictors forms the *design matrix*

$$\underline{X} = (X_1, \ldots, X_n)^T,$$

an $n \times (p + 1)$ matrix with each row being a single predictor.

# The posterior of the multiple model

The dataset is a single observation of the design matrix,

$$\underline{\boldsymbol{X}} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)^T,$$

along with a single observation of the response vector,

$$\underline{y} = (y_1, \ldots, y_n)^T.$$

(We will sometimes write $x_{ij}$ to denote the $i$th observation of the $j$th predictor, i.e. entry $[\underline{\boldsymbol{X}}]_{ij}$ of the design matrix.) The joint density of the observations and the parameters given the predictor matrix is then

$$f_{\underline{Y},B,\Sigma|\underline{X}}(\underline{y}, \boldsymbol{\beta}, \sigma|\underline{\boldsymbol{X}}) = f_B(\boldsymbol{\beta}) f_\Sigma(\sigma) \prod_{i=1}^{n} f_{Y|B,\Sigma,X}(y_i|\boldsymbol{\beta}, \sigma, \boldsymbol{x}_i) = \underbrace{f_B(\boldsymbol{\beta}) f_\Sigma(\sigma)}_{\text{prior}} \underbrace{\mathcal{L}(\boldsymbol{\beta}, \sigma)}_{\text{likelihood}}.$$

Finally, the posterior is

$$f_{B,\Sigma|\underline{Y},\underline{X}}(\boldsymbol{\beta}, \sigma|\underline{y}, \underline{\boldsymbol{X}}) = \frac{f_{\underline{Y},B,\Sigma|\underline{X}}(\underline{y}, \boldsymbol{\beta}, \sigma|\underline{\boldsymbol{X}})}{f_{\underline{Y}|\underline{X}}(\underline{y}|\underline{\boldsymbol{X}})} \propto \mathcal{L}(\boldsymbol{\beta}, \sigma) f_B(\boldsymbol{\beta}) f_\Sigma(\sigma).$$

# Computing the posterior

To obtain the posterior, we need to

- compute the likelihood;
- select the priors.

For the likelihood,

$$\begin{aligned}\mathcal{L}(\boldsymbol{\beta}, \sigma) &= \prod_{i=1}^{n} \mathcal{N}(y_i | \mathbf{x}_i^T \boldsymbol{\beta}, \sigma^2) \\ &= \frac{1}{(2\pi)^{\frac{n}{2}} \sigma^n} \exp\left( -\frac{1}{2\sigma^2} (\underline{y} - \underline{\mathbf{x}}\boldsymbol{\beta})^T (\underline{y} - \underline{\mathbf{x}}\boldsymbol{\beta}) \right).\end{aligned}$$

The following identity will come in handy:

$$(\underline{y} - \underline{\mathbf{x}}\boldsymbol{\beta})^T (\underline{y} - \underline{\mathbf{x}}\boldsymbol{\beta}) = (\boldsymbol{\beta} - \hat{\boldsymbol{\beta}})^T \underline{\mathbf{x}}^T \underline{\mathbf{x}} (\boldsymbol{\beta} - \hat{\boldsymbol{\beta}}) + (\underline{y} - \underline{\mathbf{x}}\hat{\boldsymbol{\beta}})^T (\underline{y} - \underline{\mathbf{x}}\hat{\boldsymbol{\beta}}),$$

where

$$\hat{\boldsymbol{\beta}} = (\underline{\mathbf{x}}^T \underline{\mathbf{x}})^{-1} \underline{\mathbf{x}}^T \underline{y}$$

is the maximum likelihood estimate of $\boldsymbol{\beta}$. The next slide walks you through a proof of the identity.

# Proving the handy identity

## Exercise

(i) Show that $\underline{x}^T\underline{x}\hat{\boldsymbol{\beta}} = \underline{x}^T\underline{y}$, where $\hat{\boldsymbol{\beta}}$ is the vector defined on the previous slide.

(ii) By expanding the product $(\underline{y} - \underline{x}\boldsymbol{\beta})^T(\underline{y} - \underline{x}\boldsymbol{\beta})$, show that

$$(\underline{y} - \underline{x}\boldsymbol{\beta})^T(\underline{y} - \underline{x}\boldsymbol{\beta}) = (\boldsymbol{\beta} - \hat{\boldsymbol{\beta}})^T\underline{x}^T\underline{x}(\boldsymbol{\beta} - \hat{\boldsymbol{\beta}}) + \underline{y}^T\underline{y} - \hat{\boldsymbol{\beta}}\underline{x}^T\underline{x}\hat{\boldsymbol{\beta}}.$$

(iii) Show that $\underline{y}^T\underline{y} - \hat{\boldsymbol{\beta}}^T\underline{x}^T\underline{x}\hat{\boldsymbol{\beta}} = (\underline{y} - \underline{x}\hat{\boldsymbol{\beta}})^T(\underline{y} - \underline{x}\hat{\boldsymbol{\beta}})$, and hence complete the proof of the identity on the previous slide.

(iv) Using that identity, explain why $\hat{\boldsymbol{\beta}}$ is indeed the maximum likelihood estimate of $\boldsymbol{\beta}$.

# Likelihood in the standard form

- The second term on the right-hand side of the handy identity is

$$(\underline{y} - \underline{\mathbf{x}}\hat{\boldsymbol{\beta}})^T(\underline{y} - \underline{\mathbf{x}}\hat{\boldsymbol{\beta}}) = \sum_{i=1}^{n}\left(y_i - \hat{\beta}_0 - \sum_{j=1}^{p}\hat{\beta}_j x_{ij}\right)^2,$$

  which is the RSS obtained when $\hat{\boldsymbol{\beta}}$ is used as coefficient vector.

- An unbiased frequentist estimator of $\sigma^2$ is

$$s^2 = \frac{(\underline{y} - \underline{\mathbf{x}}\hat{\boldsymbol{\beta}})^T(\underline{y} - \underline{\mathbf{x}}\hat{\boldsymbol{\beta}})}{v} \text{ with } v = n - p - 1.$$

- We can therefore write the likelihood function as

$$\begin{aligned}\mathcal{L}(\boldsymbol{\beta}, \sigma) &= \frac{1}{(2\pi)^{\frac{n}{2}}\sigma^n}\exp\left(-\frac{1}{2\sigma^2}(\boldsymbol{\beta} - \hat{\boldsymbol{\beta}})^T\underline{\mathbf{x}}^T\underline{\mathbf{x}}(\boldsymbol{\beta} - \hat{\boldsymbol{\beta}})\right)\exp\left(-\frac{vs^2}{2\sigma^2}\right)\\ &= \frac{1}{(2\pi)^{\frac{v}{2}}\sigma^v\sqrt{\det(\underline{\mathbf{x}}^T\underline{\mathbf{x}})}}\exp\left(-\frac{vs^2}{2\sigma^2}\right)\mathcal{N}(\boldsymbol{\beta}|\hat{\boldsymbol{\beta}}, \sigma^2(\underline{\mathbf{x}}^T\underline{\mathbf{x}})^{-1}),\end{aligned}$$

  where in the final equality we have used standard shorthand for the *multivariate normal* density, reviewed in the next slide.

Let $\boldsymbol{Z} = (Z_1, \ldots, Z_n)^T$ be a random vector with probability density function

$$f_{\boldsymbol{Z}}(\boldsymbol{z}; \boldsymbol{\mu}, \underline{\boldsymbol{\Sigma}}) = \frac{1}{(2\pi)^{\frac{n}{2}} \det(\underline{\boldsymbol{\Sigma}})^{\frac{1}{2}}} \exp\left(-\frac{1}{2}(\boldsymbol{z} - \boldsymbol{\mu})^T \underline{\boldsymbol{\Sigma}}^{-1} (\boldsymbol{z} - \boldsymbol{\mu})\right),$$

where $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_n)^T$ is an $n$-dimensional vector and $\underline{\boldsymbol{\Sigma}}$ is a symmetric positive definite $n \times n$ matrix with inverse $\underline{\boldsymbol{\Sigma}}^{-1}$. The random variables $Z_1, \ldots, Z_n$ are said to follow the *multivariate normal* distribution with means $\mu_1, \ldots, \mu_n$ and covariances

$$\mathrm{Cov}(Z_i, Z_j) = \mathbb{E}((Z_i - \mu_i)(Z_j - \mu_j)) = [\underline{\boldsymbol{\Sigma}}]_{ij}.$$

We write $\boldsymbol{Z} \sim \mathcal{N}(\boldsymbol{\mu}, \underline{\boldsymbol{\Sigma}})$. A shorthand for the multivariate normal density is $\mathcal{N}(\boldsymbol{z}|\boldsymbol{\mu}, \underline{\boldsymbol{\Sigma}})$.

# A note on likelihoods and densities

- Now that we have written the likelihood as a product of a multivariate normal density in $\boldsymbol{\beta}$ and a simple function of $\sigma$, it will be easier to identify the *posterior* as a known density in the parameters (so long as we make a sensible choice of prior). This justifies the effort we put into rearranging the likelihood function.
- However, it is important to remember that likelihood functions are not themselves probability densities. Using symbols for densities when writing likelihoods is just a notational convenience.

# Choosing our priors

In the absence of prior knowledge we can opt for a *vague* or *non-informative* prior. The posterior will then look much like the likelihood function (but normalised).

- A common choice of $\boldsymbol{B}$ is the uniform (flat) prior,

$$f_B(\boldsymbol{\beta}) = c_B,$$

  where $c_B$ is a constant.
- A convenient choice of non-informative prior for $\boldsymbol{\Sigma}$ is that $\log \boldsymbol{\Sigma}$ is uniform (or $\boldsymbol{\Sigma} = e^U$ where $U$ is uniform), because $\boldsymbol{\Sigma} > 0$. So the prior density of $\boldsymbol{\Sigma}$ is

$$f_\Sigma(\sigma) = \frac{c_\Sigma}{\sigma} \text{ with } \sigma > 0.$$

The posterior, according to the above assumptions, is then

$$f_{B,\Sigma|\underline{Y},\underline{X}}(\boldsymbol{\beta}, \sigma|\underline{y}, \underline{x}) = c\sigma^{-v-1} \exp\left(-\frac{vs^2}{2\sigma^2}\right) \mathcal{N}(\boldsymbol{\beta}|\hat{\boldsymbol{\beta}}, \sigma^2(\underline{x}^T\underline{x})^{-1}),$$

where $c$ is a normalising constant.

# Factorising the posterior

We could find $c$ numerically, by computing the multidimensional integral of $f_{B,\Sigma|\underline{Y},\underline{X}}(\boldsymbol{\beta}, \sigma|\underline{y}, \underline{x})$ over $\boldsymbol{\beta}$ and $\sigma$. But this will be impractical in high-dimensional spaces. A better strategy is to write the posterior in terms of known distributions which are normalised by definition.

- The posterior factorises as

$$f_{B,\Sigma|\underline{Y},\underline{X}}(\boldsymbol{\beta}, \sigma|\underline{y}, \underline{x}) = f_{B|\Sigma,\underline{Y},\underline{X}}(\boldsymbol{\beta}|\sigma, \underline{y}, \underline{x}) f_{\Sigma|\underline{Y},\underline{X}}(\sigma|\underline{y}, \underline{x}).$$

- Comparing with

$$f_{B,\Sigma|\underline{Y},\underline{X}}(\boldsymbol{\beta}, \sigma|\underline{y}, \underline{x}) = c\sigma^{-\nu-1} \exp\left(-\frac{\nu s^2}{2\sigma^2}\right) \mathcal{N}(\boldsymbol{\beta}|\hat{\boldsymbol{\beta}}, \sigma^2(\underline{x}^T\underline{x})^{-1}),$$

  we obtain that

$$f_{B|\Sigma,\underline{Y},\underline{X}}(\boldsymbol{\beta}|\sigma, \underline{y}, \underline{x}) = \mathcal{N}(\boldsymbol{\beta}|\hat{\boldsymbol{\beta}}, \sigma^2(\underline{x}^T\underline{x})^{-1}),$$

$$f_{\Sigma|\underline{Y},\underline{X}}(\sigma|\underline{y}, \underline{x}) = c\sigma^{-(\nu+1)} \exp\left(-\frac{\nu s^2}{2\sigma^2}\right).$$

The former is a multivariate normal density. The latter is closely related to the *inverse gamma* distribution...

# The gamma and inverse gamma distributions

A random variable $X$ is called *gamma* distributed with shape $a$ and rate $b$, written $X \sim \mathrm{Gamma}(a, b)$, if its probability density is

$$f_X(x; a, b) = \frac{b^a}{\Gamma(a)} x^{a-1} e^{-xb} := \mathrm{Ga}(x|a, b),$$

where $\Gamma(a) = \int_0^\infty t^{a-1} e^t dt$ is the gamma function. We have $\mathbb{E}(X) = \frac{a}{b}, \mathrm{Var}(X) = \frac{a}{b^2}, \text{ and, for } a \geq 1, \mathrm{mode}(X) = \frac{a-1}{b}$. (For $0 < a < 1, \mathrm{mode}(X) = 0$.) $Z = \frac{1}{X}$ with $X \sim \mathrm{Gamma}(a, b)$ is *inverse gamma* distributed with shape $a$ and scale $b$, written $Z \sim \text{Inv-Gamma}(a, b)$. Its density is

$$f_Z(z; a, b) = \frac{b^a}{\Gamma(a)} z^{-(a+1)} e^{-\frac{b}{z}} := \mathrm{IG}(x|a, b).$$

We have $\mathbb{E}(Z) = \frac{b}{a-1}, \mathrm{mode}(Z) = \frac{b}{a+1}, \mathrm{Var}(Z) = \frac{b^2}{(a-1)^2(a-2)}$, where $\mathbb{E}(Z)$ and $\mathrm{Var}(Z)$ exist provided $a > 1$ and $a > 2$ respectively.

# Sampling the posterior

By changing the variable from $\Sigma$ to $Z = \Sigma^2$ in the posterior density $f_{\Sigma|\underline{Y},\underline{X}}(\sigma|\underline{y},\underline{x}) = c\sigma^{-(v+1)} \exp\left(-\frac{vs^2}{2\sigma^2}\right)$, we have

$$f_{Z|\underline{Y},\underline{X}}(z|\underline{y},\underline{x}) = \frac{c}{2} z^{-\left(\frac{v}{2}+1\right)} \exp\left(-\frac{vs^2}{2z}\right) = \mathrm{IG}\left(z \middle| \frac{v}{2}, \frac{vs^2}{2}\right).$$

That is, conditional on the data, $\Sigma^2$ is inverse gamma distributed:

$$\Sigma^2 \Big| \{\underline{Y} = \underline{y}, \underline{X} = \underline{x}\} \sim \text{Inv-Gamma}\left(\frac{v}{2}, \frac{vs^2}{2}\right).$$

To draw samples $(B, \Sigma)$ from the posterior we do the following:

(i) generate $Z \sim \text{Inv-Gamma}\left(\frac{v}{2}, \frac{vs^2}{2}\right)$ and let $\Sigma = \sqrt{Z}$;

(ii) generate $B \sim \mathcal{N}\left(\hat{\beta}, \Sigma^2(\underline{x}^T\underline{x})^{-1}\right)$.

These steps can be carried out using standard sampling algorithms.

# Application to body fat data: Single predictor

Consider first the case of a single predictor, the standardised abdomen circumference $Z_{\mathrm{Abd}}$. Our linear model is then

$$Y = \beta_0 + \beta_1 Z_{\mathrm{Abd}} + \mathcal{E},$$

where $\mathcal{E} \sim \mathcal{N}(0, \sigma^2)$.

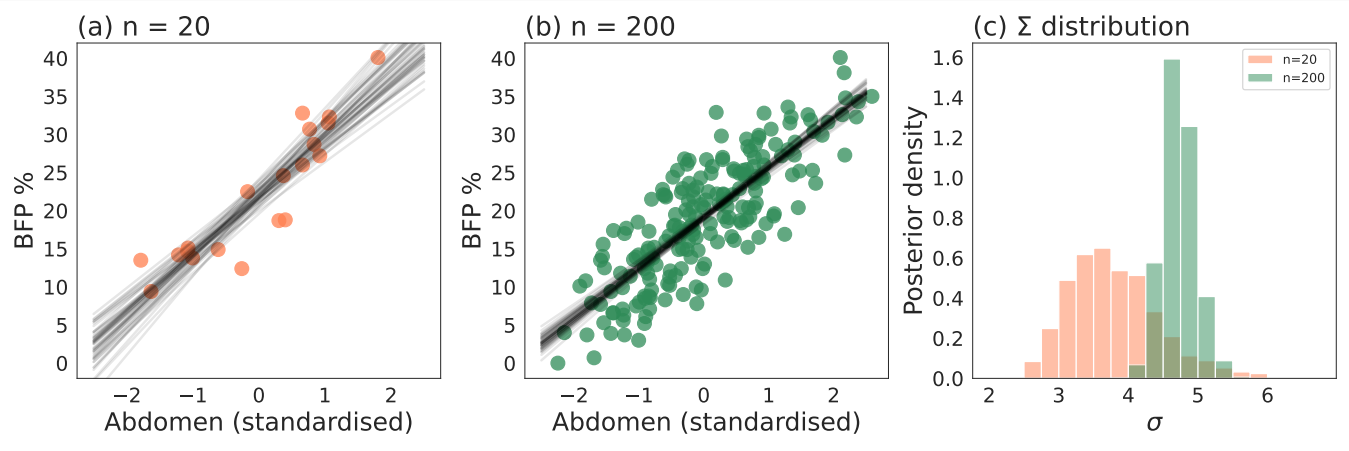



**Figure 33:** (a) and (b) each show fifty posterior samples of the regression function $y = B_0 + B_1 x$ based on datasets of size (a) $n = 20$ and (b) $n = 200$. (c) Histograms of $10^3$ samples of $\boldsymbol{\Sigma}$ from the same posteriors used in (a) and (b).

# Application to body fat data: All predictors

Now consider the model that uses all the standardised predictors:

$$Y = \boldsymbol{\beta}^T \boldsymbol{Z} + \mathcal{E},$$

where $\boldsymbol{Z} = (1, Z_1, \ldots, Z_p)$ and $\mathcal{E} \sim \mathcal{N}(0, \sigma^2)$. For most coefficients, the posterior is uncertain even about the sign!

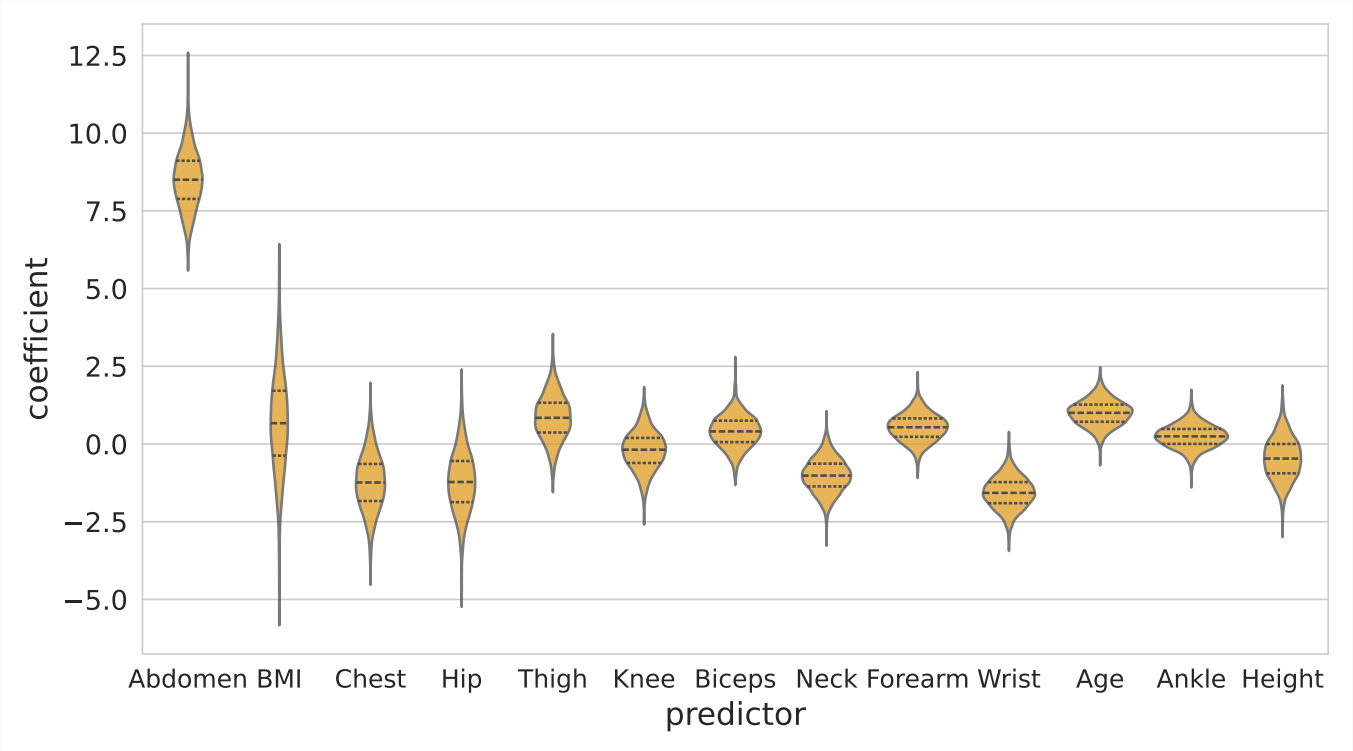


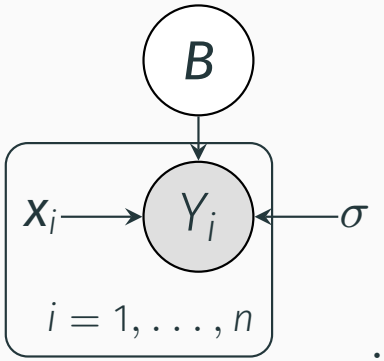

**Figure 34:** Marginal posterior distributions of the model coefficients

# Shrinkage prior

- When using all body fat predictors, many coefficients of the marginal posterior are peaked close to zero with comparatively high variance.
- The sensitivity of the response to these predictors is both *small* and *uncertain*, and therefore the coefficient estimates will be sensitive to noise in the data, leading to overfitting.
- This was also observed when we analysed the multiple linear model using a frequentist approach. Excluding predictors improved cross validation performance.
- The Bayesian approach to tackle overfitting is to impose a *shrinkage prior* on the coefficients. The aim is to shrink coefficient estimates associated with weaker (i.e. less predictively useful) predictors.

# Gaussian priors

A model in which the noise variance is known corresponds to the DAG




.

To use a *Gaussian* shrinkage prior is to assume that the coefficients $B_1, \ldots, B_p$ are i.i.d. normal with zero mean and variance $\frac{\sigma^2}{\lambda}$, and that $B_0$ is independently normal with zero mean and variance $\frac{\sigma^2}{\epsilon\lambda}$, i.e.

$$f_B(\boldsymbol{\beta}) = \mathcal{N}\left(\boldsymbol{\beta} \middle| \mathbf{0}, \frac{\sigma^2}{\lambda} \underline{I}_\epsilon^{-1}\right),$$

where $\underline{I}_\epsilon = \mathrm{diag}(\epsilon, 1, \ldots, 1)$.

- $\lambda$ is the *shrinkage hyperparameter*. Increasing $\lambda$ concentrates the prior closer to zero.
- Taking $\epsilon \to 0$ in the posterior corresponds to a flat prior on $B_0$.

# Gaussian priors

A Gaussian prior leads to the posterior

$$f_{B|\underline{Y},\underline{X}}(\boldsymbol{\beta}|\underline{y},\underline{\boldsymbol{X}}) = c \exp\left(-\frac{(\underline{y}-\underline{\boldsymbol{X}}\boldsymbol{\beta})^T(\underline{y}-\underline{\boldsymbol{X}}\boldsymbol{\beta})+\lambda\boldsymbol{\beta}^T\underline{I}_\epsilon\boldsymbol{\beta}}{2\sigma^2}\right),$$

where $c$ is a normalising constant.

It is possible to express the posterior as a normal density from which we can sample $\boldsymbol{B}$.

**Exercise**

Show that the above posterior is multivariate normal with covariance matrix $\underline{\boldsymbol{\Sigma}} = \sigma^2(\underline{\boldsymbol{X}}^T\underline{\boldsymbol{X}} + \lambda\underline{I}_\epsilon)^{-1}$ and mean vector $\boldsymbol{\mu} = \sigma^{-2}\underline{\boldsymbol{\Sigma}}\underline{\boldsymbol{X}}^T\underline{y}$.

# Marginal posteriors of model coefficients

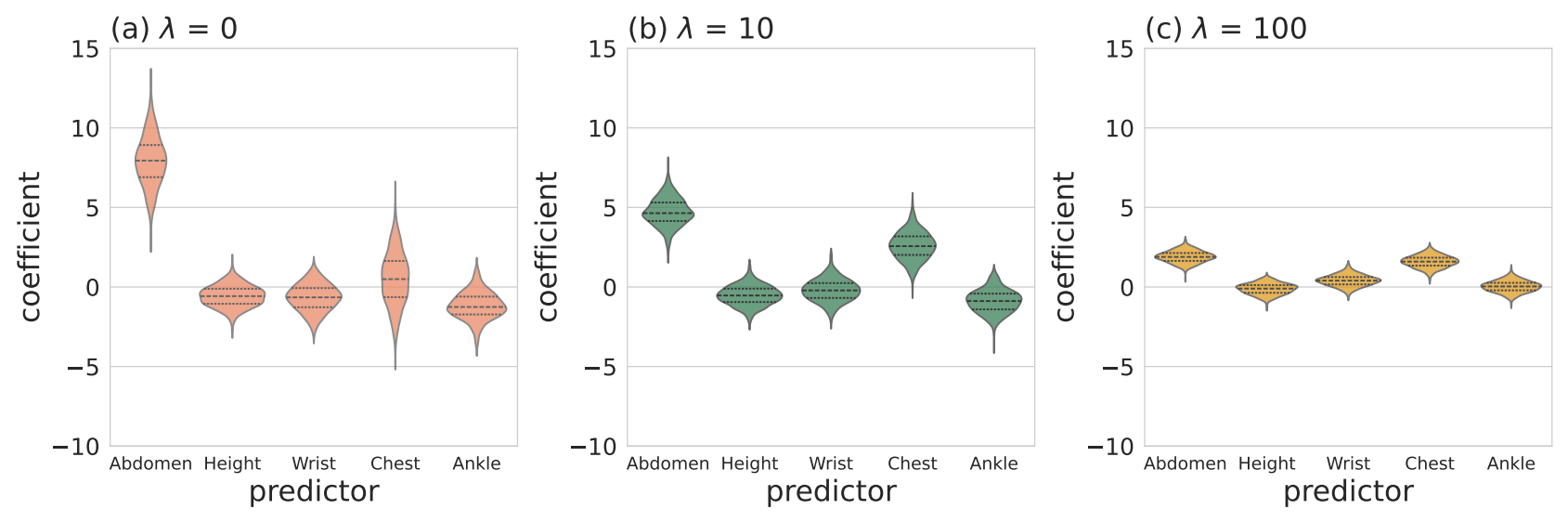



**Figure 35:** Marginal posteriors of model coefficients in a four-predictor model with three different values of the shrinkage hyperparameter $\lambda \in \{0, 10, 100\}$.

- The prior shrinks both coefficient magnitudes and variances.
- The prior *constrains* the model so that it is less flexible and therefore less susceptible to noise.
- Just the right amount of shrinkage will optimise model performance.

# Ridge regression

The maximum a posteriori (MAP) estimate of the coefficient vector is

$$\hat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta}}{\arg\min} \left( \sum_{i=1}^{n} \left( y_i - \beta_0 - \sum_{k=1}^{p} \beta_k x_{ik} \right)^2 + \lambda \sum_{k=1}^{p} \beta_k^2 \right).$$

- The objective to be minimised is proportional to the negative exponent of the posterior to be maximised. It is equal to the residual sum of squares plus a *shrinkage penalty* originating from the prior.
- MAP coefficient estimation with Gaussian shrinkage prior is therefore a penalised form of ordinary least squares regression, known as *ridge regression*.
- Computing $\hat{\boldsymbol{\beta}}$ is an optimisation problem that can be solved efficiently.

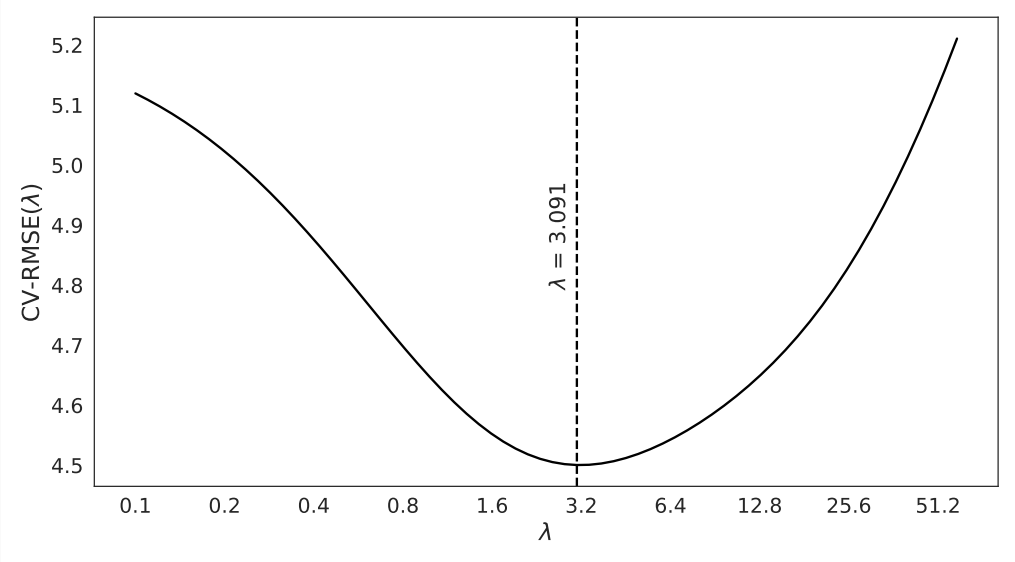



**Figure 36:** Cross validated $\mathrm{RMSE}$ in the all-predictor model of BFP as a function of ridge shrinkage hyperparameter $\lambda$ for a dataset of size $n = 50$ randomly selected from the full dataset.

- U-shaped error curve: trade-off between bias and variance.
- Small $\lambda$ means low bias but high variance. The model is more flexible, but also more sensitive to noise.
- Large $\lambda$ means high bias but low variance. The model is less flexible, but also less sensitive to noise.
- There is a sweet spot ($\lambda \approx 3$) where predictive power is optimised.

# Laplace priors

The other commonly used shrinkage prior is the Laplace prior.

**The Laplace distribution**

A Laplace random variable with a location parameter $\mu$ and a scale parameter $b$ has the probability density

$$f_X(x; \mu, b) = \frac{1}{2b} \exp\left(-\frac{|x - \mu|}{b}\right) := \mathrm{Lap}(x|\mu, b).$$

The Laplace prior assumes that the coefficients $B_1, \ldots, B_p$ are i.i.d. Laplace random variables with location parameter 0 and scale parameter $b$.

Smaller values of $b$ correspond to prior distributions more tightly concentrated around zero.

# Lasso regression

The maximum a posteriori estimate under the Laplace prior is

$$\hat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta}}{\arg\min} \left( \sum_{i=1}^{n} \left( y_i - \beta_0 - \sum_{k=1}^{p} \beta_k x_{ik} \right)^2 + \lambda \sum_{k=1}^{p} |\beta_k| \right),$$

where $\lambda = \frac{1}{b}$.

- Once again, the objective to be minimised is the residual sum of squares plus a *shrinkage penalty* originating from the prior.
- MAP coefficient estimation with Laplace shrinkage prior is a penalised form of ordinary least squares regression, known as *lasso regression*.
- Computing $\hat{\boldsymbol{\beta}}$ is an optimisation problem that can be solved efficiently.

# Classification

# Concept of classification

Here are the first few rows of a biomedical dataset. Rows correspond to patients, and columns to variables.

| Fasting blood sugar (mg/dL) | Hemoglobin A1C (%) | Cholesterol triglycerides (mg/dL) | Hypertension | Socioeconomic status (0=low, 1=mid, 2=high) | Diagnosis |
|---|---|---|---|---|---|
| 120.2 | 7.93 | 235.4 | no | 2 | no |
| 75.1 | 4.50 | 232.9 | no | 1 | no |
| 187.3 | 8.62 | 180.4 | no | 0 | yes |
| 134.6 | 6.75 | 299.6 | no | 1 | yes |
| 179.5 | 5.56 | 337.8 | no | 1 | no |
| 85.4 | 9.76 | 271.5 | no | 1 | no |
| 151.2 | 7.18 | 295.0 | yes | 1 | yes |
| 87.4 | 9.71 | 248.6 | no | 0 | no |

The patients in the dataset are a random sample from a large population of similar patients.

Could we predict whether a new patient sampled from this population has diabetes by observing the values of the other variables?

# Concept of classification

A given patient either has diabetes or does not. So our prediction target is *binary*.

Binary variables are a special case of categorical variables. A variable is categorical if it takes values in a finite set that lacks natural arithmetical structure. The possible values are known as ‘categories’, ‘levels’ or ‘classes’.

The set of possible values may still have a natural *order*, e.g. {low, mid, high}. If so, the categorical variable is *ordinal*. Otherwise it is *nominal*.

**Exercise**

In the diabetes dataset, does socioeconomic status count as a categorical variable?

# Concept of classification

- Prediction problems in which the prediction target is categorical are known as *classification problems*.
- We'll begin by considering binary classification, and then generalise to multi-class nominal targets.
- We won't consider ordinal targets.

# Probabilistic binary classification

We study tools whose output is a *probability* — in the diabetes case, a probability that a patient is diabetic.

- Probability is often of interest to a classification tool's users.
- Probabilistic output can be useful when fitting a classifier to training data.

The diabetes dataset takes the form $D = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$, where

- $\boldsymbol{x}_i$ is the vector of predictor variables for the $i$th patient,
- $y_i$ is the diabetes status of the $i$th patient, and
- all binary variables are coded as 0s and 1s. In particular, $y_i$ takes the value 1 if the $i$th patient has diabetes, and 0 otherwise.

When we feed a predictor vector $\boldsymbol{x}$ to a probabilistic classifier $g$, it returns a probability $g(\boldsymbol{x})$. This is the predicted probability that a patient with predictor vector $\boldsymbol{x}$ is diabetic.

# Probabilistic binary classification

A probabilistic binary classifier $g$ corresponds to the following hypothesis about the conditional distribution of $Y$ given $\boldsymbol{X} = \boldsymbol{x}$:

$$Y \mid \{\boldsymbol{X} = \boldsymbol{x}\} \sim \text{Bernoulli}(g(\boldsymbol{x})).$$

According to this hypothesis, the conditional probability that the class label of a patient randomly drawn from the relevant population takes value $y$ given that the patient's predictor vector is $\boldsymbol{x}$ is

$$f_{Y|\boldsymbol{X}}(y|\boldsymbol{x}) = g(\boldsymbol{x})^y \left(1 - g(\boldsymbol{x})\right)^{1-y}.$$

**Question**

Verify the above equation by checking the cases $y = 1$ (patient is diabetic) and $y = 0$ (patient is not diabetic) separately.

# Cross-entropy loss

As each example $(x, y)$ in $D$ is drawn independently from the dataset, the likelihood function is

$$\mathcal{L}(g; D) = \prod_{(x,y)\in D} g(x)^y \left(1 - g(x)\right)^{1-y}.$$

Picking a classifier (from some family of candidates) by maximising likelihood is equivalent to picking a classifier that minimises the *cross-entropy loss*, defined by

$$\begin{aligned} l(g; D) &= -\frac{1}{n} \log \mathcal{L}(g; D) \\ &= -\frac{1}{n} \sum_{(x,y)\in D} \left(y \log g\left(x\right) + (1 - y) \log\left(1 - g(x)\right)\right). \end{aligned}$$

Notice that

$$y \log g(x) + (1 - y) \log(1 - g(x)) = \begin{cases} \log g(x) & \text{if patient is diabetic} \\ \log(1 - g(x)) & \text{if patient is not diabetic} \end{cases}$$

# Cross-entropy loss

Good classifiers consistently assign high probabilities to the correct class labels and thus achieve low cross-entropy loss.

**Question**

Show that $l(g; D)$ can never be negative. In what situations is it zero?

**Question**

Show that $\hat{g} = \underset{g \in G}{\arg\max}\, \mathcal{L}(g; D)$ if and only if $\hat{g} = \underset{g \in G}{\arg\min}\, l(g; D)$.

Cross-entropy loss is more convenient to work with in practice than likelihood, as its expected value does not scale with the dataset size.

**Question**

Let random dataset $\mathcal{D}$ be a sequence of $n$ i.i.d. copies of random vector $(X, Y)$. Show that $\mathbb{E}_{\mathcal{D}}(l(g; \mathcal{D}))$ does not depend on $n$.

# The logit function and logistic regression

Probabilities must lie between 0 and 1, so linear probabilistic classifiers ('$g(\boldsymbol{x}; \beta_0, \boldsymbol{\beta}) = \beta_0 + \boldsymbol{\beta}^T \boldsymbol{x}$') don't make sense. But given some fixed invertible *link function*, $\psi : (0, 1) \to \mathbb{R}$, we can define a family of probabilistic classifiers by

$$\psi\left(g(\boldsymbol{x}; \beta_0, \boldsymbol{\beta})\right) = \beta_0 + \boldsymbol{\beta}^T \boldsymbol{x}.$$

One common choice for $\psi$ is the *logit function*, $\operatorname{logit}(p) := \log \frac{p}{1-p}$.

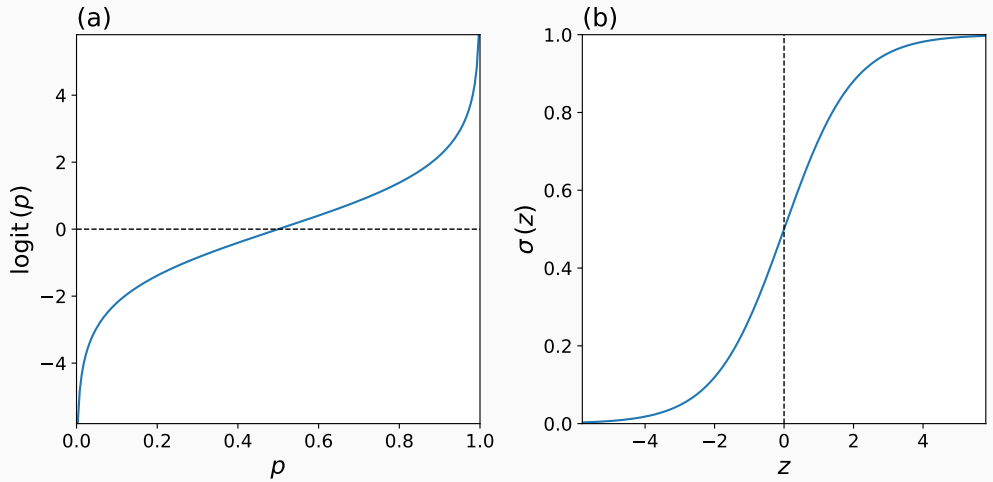



**Figure 37:** (a) The logit function, $\operatorname{logit}(p) = \log \frac{p}{1-p}$, plotted against $p$. (b) The logistic function (inverse of logit), $\sigma(z) = \frac{1}{1+e^{-z}}$, plotted against $z$.

# The logit function and logistic regression

The quantity $\operatorname{logit}(p) = \log \frac{p}{1-p}$ is also called the *log odds*.

- If $p$ is a probability, then $\frac{p}{1-p}$ is the corresponding *odds ratio*.
- $\lim_{p \to 0+} \frac{p}{1-p} = 0$ and $\lim_{p \to 1-} \frac{p}{1-p} = +\infty$.
- $\log \frac{p}{1-p}$ can take any real value.

The standard *logistic function* or *standard sigmoid* is the inverse of the logit function, defined by $\sigma(z) = \frac{1}{1+e^{-z}}$.

## Question

Compute the log odds corresponding to each of the following probabilities: $10^{-12}$; 1%; 52%; 53%. Which of the following probability changes corresponds to the greater change in log odds:

- a change in probability from $10^{-12}$ to 1%;
- a change in probability from 52% to 53%?

# Logistic regression

Using logit as our link function, we can postulate that the log odds depends linearly on the predictor vector. Equivalently,

$$g(\boldsymbol{x}; \beta_0, \boldsymbol{\beta}) = \sigma(\beta_0 + \boldsymbol{\beta}^T \boldsymbol{x}) = \frac{1}{1 + e^{-(\beta_0 + \boldsymbol{\beta}^T \boldsymbol{x})}}.$$

If we fit that model to data by minimising the cross-entropy loss (or a regularised version of this loss: see later slides), we are doing *binary logistic regression*.

**Question**

Show that the cross-entropy loss $l(\beta_0, \boldsymbol{\beta}; D)$ is

$$\frac{1}{n} \sum_{(\boldsymbol{x}, y) \in D} \left( y \log(1 + e^{-(\beta_0 + \boldsymbol{\beta}^T \boldsymbol{x})}) + (1 - y) \log(1 + e^{\beta_0 + \boldsymbol{\beta}^T \boldsymbol{x}}) \right).$$

There is no simple formula for the values of $\beta_0$ and $\boldsymbol{\beta}$ that minimise this loss, but it is usually easy to find them computationally.

# Simple logistic regression (one predictor)

- If there is just one scalar predictor $x$, we have

$$g(x; \beta_0, \beta_1) = \sigma(\beta_0 + \beta_1 x).$$

- Logistic regression with a single predictor is sometimes called *simple logistic regression*.

- The cross-entropy loss is then

$$l(\beta_0, \beta_1; D) = \frac{1}{n} \sum_{(x,y) \in D} \left( y \log(1 + e^{-(\beta_0 + \beta_1 x)}) + (1 - y) \log(1 + e^{\beta_0 + \beta_1 x}) \right).$$

# Simple logistic regressions using the diabetes dataset

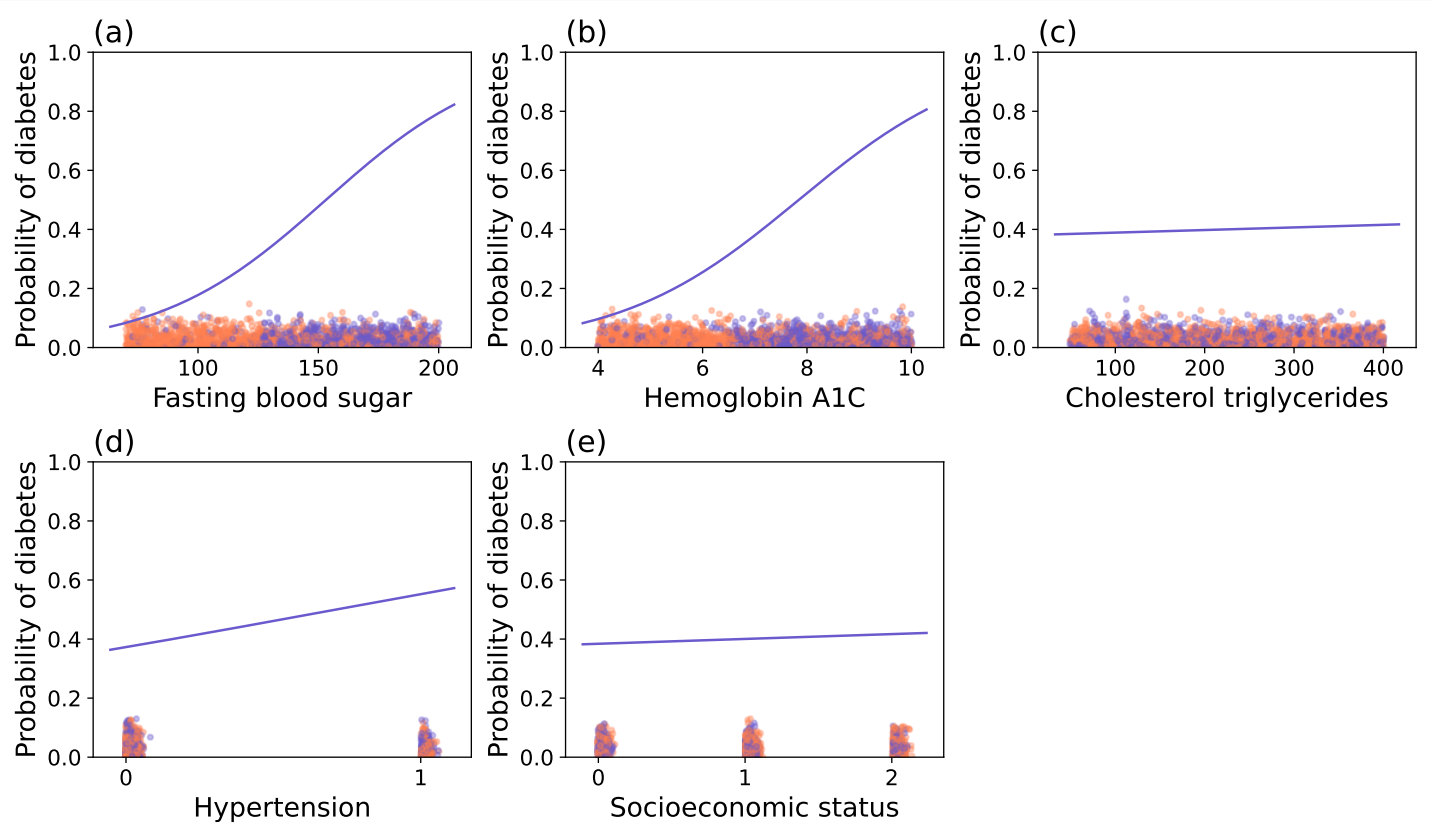



**Figure 38:** Each blue curve shows predicted probability of diabetes as a function of a single predictor. Training datapoints are plotted along the predictor axis, colour-coded by diagnosis (blue for diabetes, orange for no diabetes). Datapoints are plotted with vertical jitter. In panels (d) and (e), horizontal jitter is used too.

# Simple logistic regressions using simulated datasets

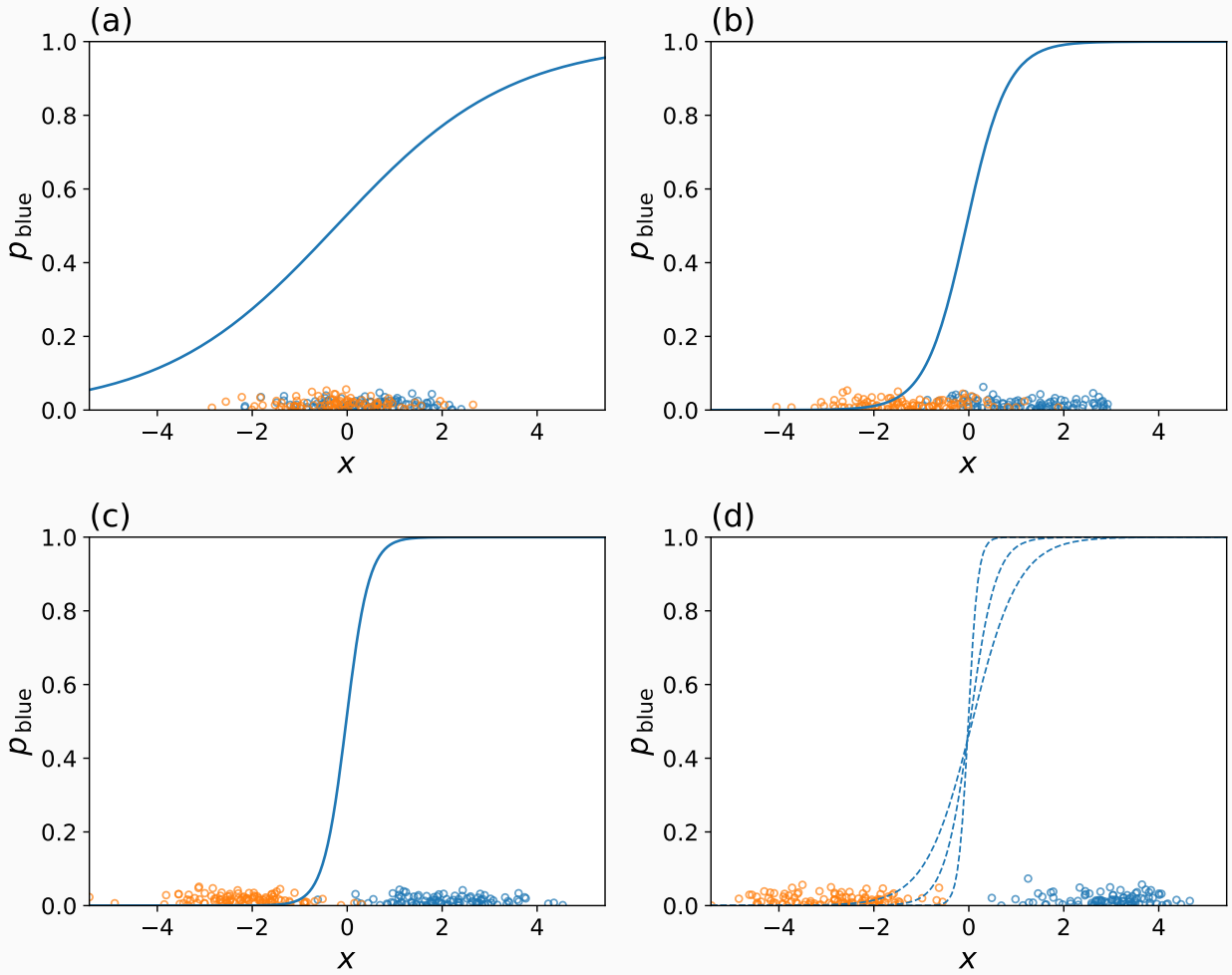



**Figure 39:** Each solid blue sigmoid represents the probability that a point is blue (rather than orange) as a function of the predictor $x$. For (d), the distributions are *linearly separable* with no best-fitting sigmoid.

# Regulisation

Adding a regularisation penalty to the cross-entropy loss ensures that a best-fitting sigmoid will exist even when the training data is linearly separable. With a ridge penalty, the loss function becomes

$$l^{\lambda}(\beta_0, \boldsymbol{\beta}; D) = \frac{1}{n}\left(\lambda\|\boldsymbol{\beta}\|^2 - \sum_{(x,y)\in D}\left(y \log g(x; \beta_0, \boldsymbol{\beta}) + (1-y)\log(1 - g(x; \beta_0, \boldsymbol{\beta}))\right)\right),$$

where $\|\boldsymbol{\beta}\|^2 = \boldsymbol{\beta}^T\boldsymbol{\beta} = \sum_{i=1}^{p}\beta_i^2$.

Regularisation penalises sharp transitions. $\beta_0$ is left out of the penalty to avoid baking hunches about the *location* of the transition into the prior.

Before fitting a regularised model, one should standardise each predictor to have the same variance over the training data. (Otherwise components of $\boldsymbol{\beta}$ corresponding to predictors with smaller ranges of variation will be subject to stronger regularisation.)

# Hyperparameter $\lambda$ and test scores

How to choose the hyperparameter $\lambda$?

- *Stratified* cross-validation: Divide the dataset into equal-sized folds with each containing roughly the same number of examples with class label $y = 1$.

How to measure performance?

- *Cross-entropy loss* evaluated on the validation data:

$$l_{\text{val}}(\beta_0, \boldsymbol{\beta}; D_{\text{val}}) = \\ -\frac{1}{n_{\text{val}}} \sum_{(\mathbf{x},y)\in D_{\text{val}}} \left(y \log g(\mathbf{x}; \beta_0, \boldsymbol{\beta}) + (1-y) \log(1 - g(\mathbf{x}; \beta_0, \boldsymbol{\beta}))\right).$$

  Lower cross-entropy loss indicates better performance.
- *Accuracy*: proportion of validation examples classified correctly. Probabilistic outputs are converted to best guesses ($\mathbf{Pr} > 0.5$ to the guess $y = 1$ and $\mathbf{Pr} \leq 0.5$ to the guess $y = 0$). Higher accuracy indicates better performance.

# Cross-validation results on the diabetes dataset

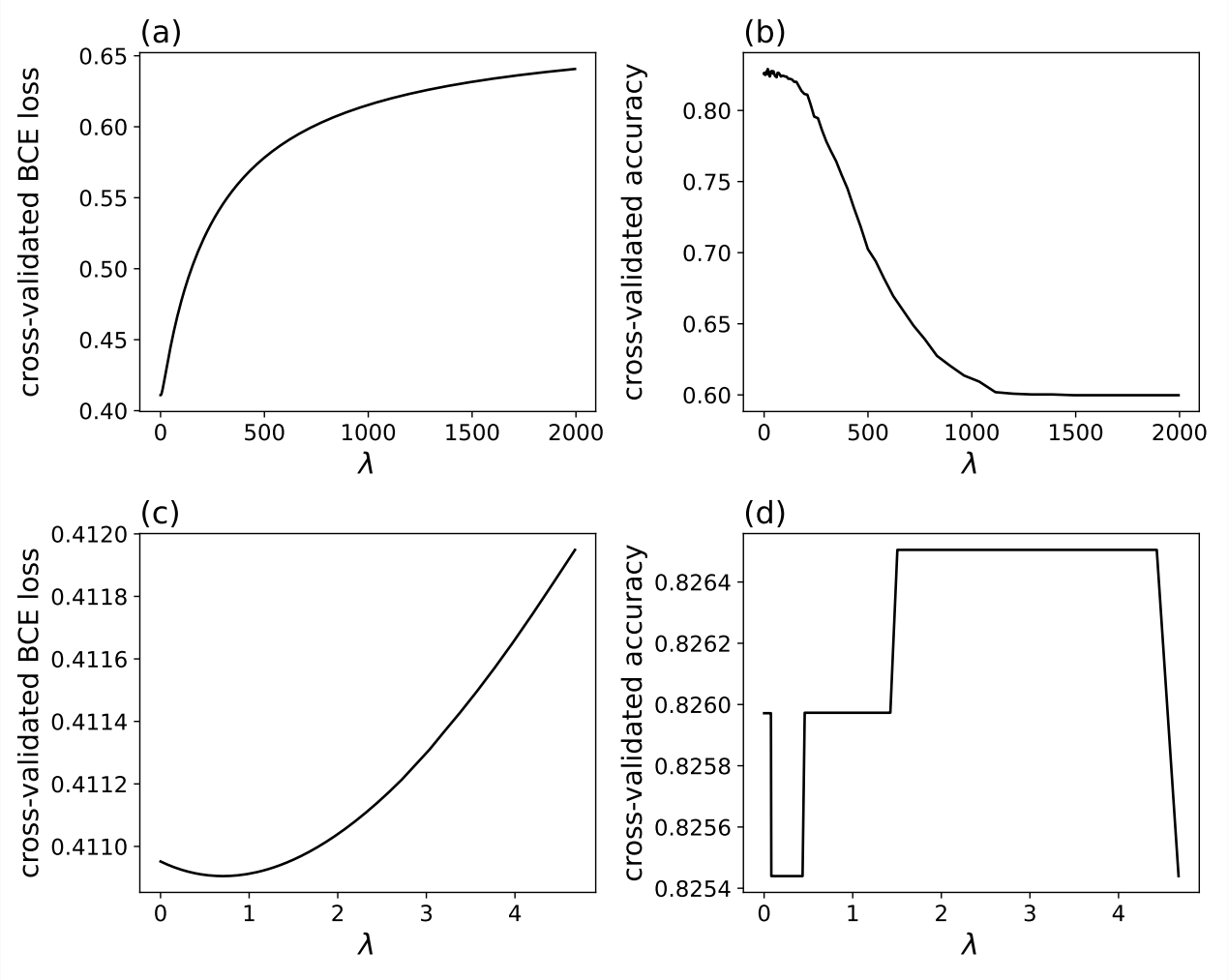



**Figure 40:** Panels (a) and (b) cover a wide range of regularisation strengths. The initial portion of this range is replotted in panels (c) and (d).

- Minimising unregularised cross-entropy gives cross-validated accuracy of roughly 83% on the diabetes dataset.
- The base rate of diabetes in the dataset, $\pi = \frac{1}{n}\sum_{i=1}^{n} y_i$, is 40%. So a boring classifier that always guessed 'no' would achieve an accuracy of 60%.
- In the strong regularisation limit where $\lambda \to \infty$, the accuracy converges to just this value: 60%. Moreover, the cross-entropy loss converges to $-(\pi \log \pi + (1-\pi)\log(1-\pi)) \approx 0.67$.

## Question

(i) Explain why pushing $\lambda$ to infinity drives every component of $\hat{\boldsymbol{\beta}}$ towards zero. That is, explain why

$$\lim_{\lambda \to \infty} \operatorname*{arg\,min}_{(\beta_0, \boldsymbol{\beta})} \left[ \frac{\lambda}{n} \|\boldsymbol{\beta}\|^2 + l(\beta_0, \boldsymbol{\beta}; D) \right] = (\hat{\beta}_0, \mathbf{0}),$$

where $\hat{\beta}_0 = \operatorname*{arg\,min}_{\beta_0} l(\beta_0, \mathbf{0}; D)$.

(ii) Show that $g(\boldsymbol{x}; \hat{\beta}_0, \mathbf{0}) = \pi$.

# Coefficients for logistic regression

**Table 2:** Coefficients for logistic regression ($\lambda = 0.7$) fitted to the diabetes dataset. Predictor components were standardised prior to fitting. Standard errors were estimated using the bootstrap method with 1000 replications.

| Predictor | Predictor index, $i$ | $\hat{\beta}_i$ | $\hat{\mathrm{se}}(\hat{\beta}_i)$ |
|---|---|---|---|
| Fasting blood sugar | 1 | 1.584 | 0.090 |
| Hemoglobin A1C | 2 | 1.494 | 0.084 |
| Cholesterol triglycerides | 3 | 0.103 | 0.063 |
| Hypertension | 4 | 0.588 | 0.073 |
| Socioeconomic status | 5 | 0.102 | 0.065 |

- This is a *tall* dataset: 1879 patients but only 5 features. Overfitting is not a significant danger, and the optimal value of $\lambda$ is close to zero.
- Fasting blood sugar and hemoglobin A1C are the most useful predictors.

# Fewer samples and more features

We would expect some *overfitting* and *regularisation to help more*.

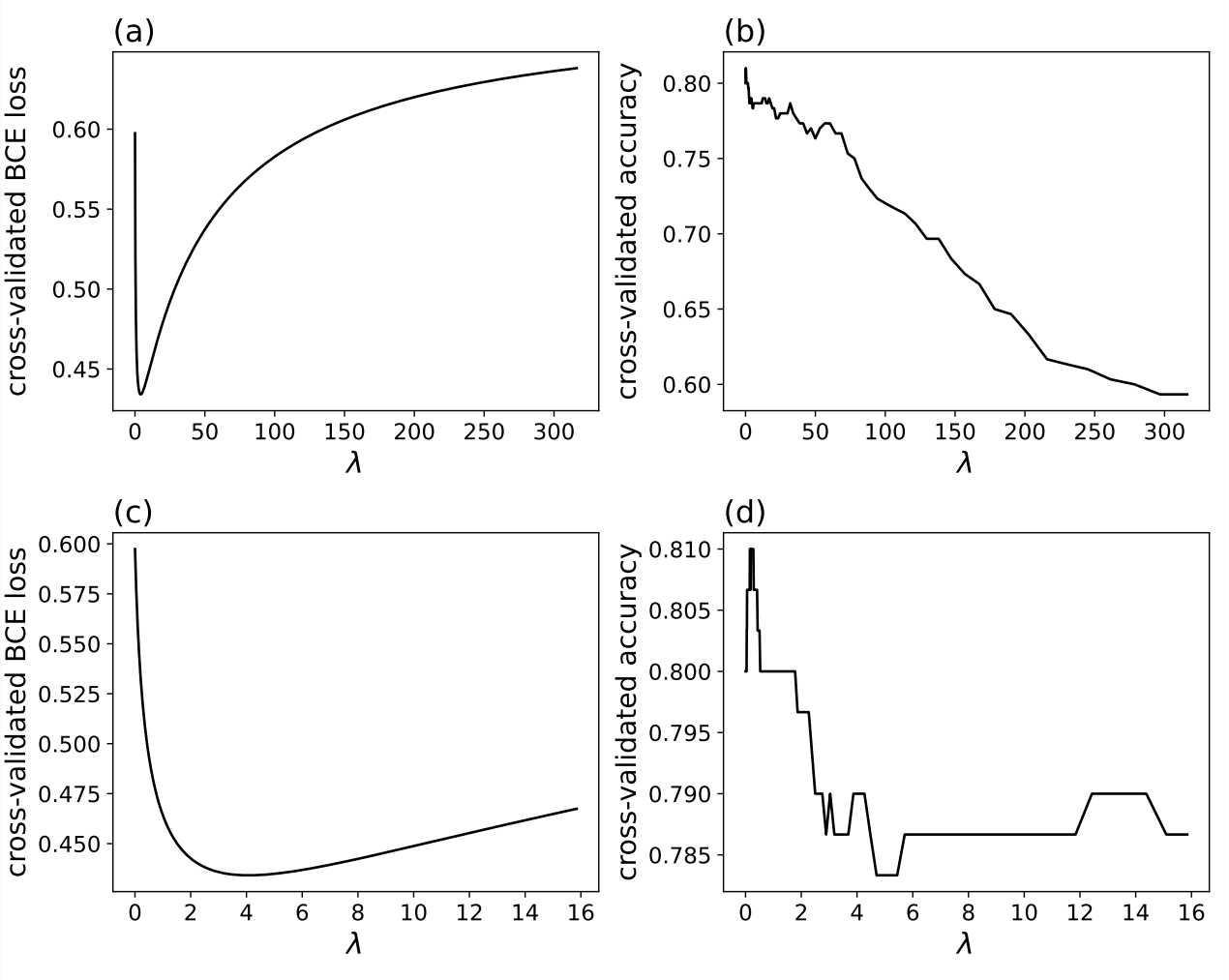



**Figure 41:** Classification to diabetes dataset of 43 features and 300 examples.

# Probabilistic multiclass classification

- In multiclass classification tasks, the prediction target is a categorical variable that can take many different values.

- Identifying the language of a text snippet is a multiclass classification task, so long as the set of possible values of the prediction target is well-defined and finite — e.g., {English, Spanish, French, German, Chinese}.

- A *probabilistic* multiclass classifier returns not just a best guess, but a probability distribution over the set of possibilities.

# Integer encoding and probability vectors

- Possible values of the prediction target can be encoded as the integers $\{0, 1, \ldots, K-1\}$, where $K$ is the total number of classes.
- A probabilistic multiclass classifier $\boldsymbol{g}$ maps predictor vectors to probability distributions over the set $\{0, 1, ..., K-1\}$.
- The map can be written as

$$\boldsymbol{x} \mapsto (g_0(\boldsymbol{x}), g_1(\boldsymbol{x}), ..., g_{K-1}(\boldsymbol{x}))\,,$$

  where $g_i(\boldsymbol{x})$ is the predicted probability of class label $i$.
- The $K$-tuple of probabilities $\boldsymbol{g}(\boldsymbol{x})$ is sometimes called a *probability vector*.

# Categorical cross-entropy loss

- $\boldsymbol{g}$ corresponds to the hypothesis that the conditional distribution of $Y$ given $\boldsymbol{X} = \boldsymbol{x}$ (for any $\boldsymbol{x}$) is

$$Y \,|\, \{\boldsymbol{X} = \boldsymbol{x}\} \sim \text{Categorical}\left(g_0(\boldsymbol{x}), g_1(\boldsymbol{x}), \ldots, g_{K-1}(\boldsymbol{x})\right).$$

- A multiclass classification dataset $D = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$ consist of $n$ independent realisations of $(\boldsymbol{X}, Y)$. The likelihood of $\boldsymbol{g}$ is the conditional probability (according to $\boldsymbol{g}$) of the observed labels $(y_1, \ldots, y_n)$ given the observed predictor vectors $(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)$,

$$\mathcal{L}(\boldsymbol{g}; D) = \prod_{(\boldsymbol{x},y)\in D} g_y(\boldsymbol{x}).$$

- Choosing $\boldsymbol{g}$ to maximise the likelihood is equivalent to choosing $\boldsymbol{g}$ to minimise the *categorical cross-entropy*

$$\begin{aligned} l(\boldsymbol{g}; D) &= -\frac{1}{n} \log \mathcal{L}(\boldsymbol{g}; D) \\ &= -\frac{1}{n} \sum_{(\boldsymbol{x},y)\in D} \log g_y(\boldsymbol{x}). \end{aligned}$$

# One-hot encoding

- Integer encoding of nominal prediction targets introduces spurious arithmetical relationships between the class labels.
- An alternative is to use probability vectors — specifically, extreme points on the probability simplex — as class labels. So, for example, when $K = 5$, the set of vector class labels would be

$$\{(1,0,0,0,0),(0,1,0,0,0),(0,0,1,0,0),(0,0,0,1,0),(0,0,0,0,1)\}.$$

- This strategy is called *one-hot encoding*. Notice that it treats the classes entirely symmetrically.
- The prediction target is now a random *vector*, $Y$.
- The expectations $\mathbb{E}(Y)$ and $\mathbb{E}(Y|X = x)$ are meaningful.

**Question**

How should we interpret the components of $\mathbb{E}(Y)$? What about the components of $\mathbb{E}(Y|X = x)$?

# One-hot encoding

One-hot encoding makes probabilistic classification look like a typical supervised task. When we feed a predictor vector $\boldsymbol{x}$ to our classifier, we want it to returns a prediction — a probability vector $\boldsymbol{g}(\boldsymbol{x})$ — that lies close to the true label $\boldsymbol{y}$. That familiar framing only makes sense if the true label is itself a probability vector.

When class labels are one-hot vectors, the categorical cross-entropy becomes

$$l(\boldsymbol{g}; D) = -\frac{1}{n} \sum_{(\boldsymbol{x},\boldsymbol{y}) \in D} \boldsymbol{y}^T \log \boldsymbol{g}(\boldsymbol{x}),$$

where the log function is applied component-wise.

**Question**

Show that the two formulae for categorical cross-entropy we have given — one assuming integer encoding, the other one-hot encoding — are equivalent.

# Multinomial logistic regression

Generalising logistic regression to the multiclass case yields *multinomial logistic regression*.

- We introduce a score for each class. The scores are linear in $\boldsymbol{x}$, so

$$s_i = \beta_0^{(i)} + \boldsymbol{\beta}^{(i)T}\boldsymbol{x} \text{ for } i = 0, 1, \ldots, K-1.$$

- The $K$ scores can be combined into a single score vector,

$$\boldsymbol{s} = \boldsymbol{\beta}_0 + \underline{\boldsymbol{\beta}}\,\boldsymbol{x} \in \mathbb{R}^K,$$

  where $\boldsymbol{\beta}_0 \in \mathbb{R}^K$ and $\underline{\boldsymbol{\beta}} \in \mathbb{R}^{K \times p}$ are parameters to be learned.

- We map the score vector $\boldsymbol{s}$ to a probability vector by applying the softmax function, defined by

$$\mathrm{softmax}(\boldsymbol{s})_i = \frac{e^{s_i}}{\sum_{j=0}^{K-1} e^{s_j}} \text{ for } i = 0, 1 \ldots, K-1.$$

- We fit the classifier to training data by minimising (regularised) categorical cross-entropy.

# Softmax function

**Question**

Verify that $\mathrm{softmax}(\boldsymbol{s})$ is a probability vector, i.e., that its entries are nonnegative numbers that sum to one.

**Question**

Show that adding a constant to each component of $\boldsymbol{s}$ (the same constant for each score) leaves $\mathrm{softmax}(\boldsymbol{s})$ unchanged.

When computing $\mathrm{softmax}(\boldsymbol{s})$, it is a good practice to subtract $\max_i s_i$ from each score. The softmax function is invariant under this transformation and it helps with numerical stability.

# Summary of multinomial logistic regression

Multinomial logistic regression is defined by a model and a fitting strategy. The (conditional probabilistic) model is

$$g(x; \beta_0, \underline{\beta}) = \mathrm{softmax}(\beta_0 + \underline{\beta}\, x),$$

where $\beta_0 \in \mathbb{R}^K$ and $\underline{\beta} \in \mathbb{R}^{K \times p}$ are the parameters. The fitting strategy is to choose parameters that minimise (regularised) categorical cross-entropy loss on training data $D = \{(x_i, y_i)\}_{i=1}^n$. With ridge regularisation, the loss is

$$l^\lambda(\beta_0, \underline{\beta}; D) = \frac{\lambda}{n} \|\underline{\beta}\|^2 + l(\beta_0, \underline{\beta}; D),$$

where $\lambda \geq 0$ is a hyperparameter, $\|\underline{\beta}\|^2 = \sum_{i=0}^{K-1} \sum_{j=1}^{p} \beta_{ij}^2$, and

$$l(\beta_0, \underline{\beta}; D) = -\frac{1}{n} \sum_{(x,y) \in D} \log g_y(x; \beta_0, \underline{\beta}).$$

As with binary logistic regression, it is good practice to standardise every predictor vector component before fitting the model.

# The special case $K = 2$ is binary logistic regression

Multinomial logistic regression with $K = 2$ is equivalent to binary logistic regression.

- The score vector in this case is $(s_0, s_1)$.
- The predicted probability of class 1 is

$$\frac{e^{s_1}}{e^{s_0} + e^{s_1}} = \frac{1}{1 + e^{s_0 - s_1}} = \sigma(s_1 - s_0),$$

  where $\sigma$ is the logistic function.
- The difference $s_1 - s_0$ is the log odds. Since both $s_1$ and $s_0$ are linear functions of $\boldsymbol{x}$ (according to the multinomial logistic regression model), the log odds is too.

# Log odds in the multinomial case

In the multinomial case, there is no single log odds in general.

## Question

We define the log odds between classes $i$ and $j$ as

$$\log\left(\frac{g_i(x;\beta_0,\underline{\beta})}{g_j(x;\beta_0,\underline{\beta})}\right).$$

Show that increasing $x_k$ by one unit, while holding all other components of the predictor vector constant, shifts the log odds between classes $i$ and $j$ by $\beta_{ik} - \beta_{jk}$.

# Application to human speech sounds

The sound of a vowel is largely determined by the position of the tongue — how *high* and how far *back* it is in the mouth. 'Backness' and 'height' together define a two-dimensional feature space.

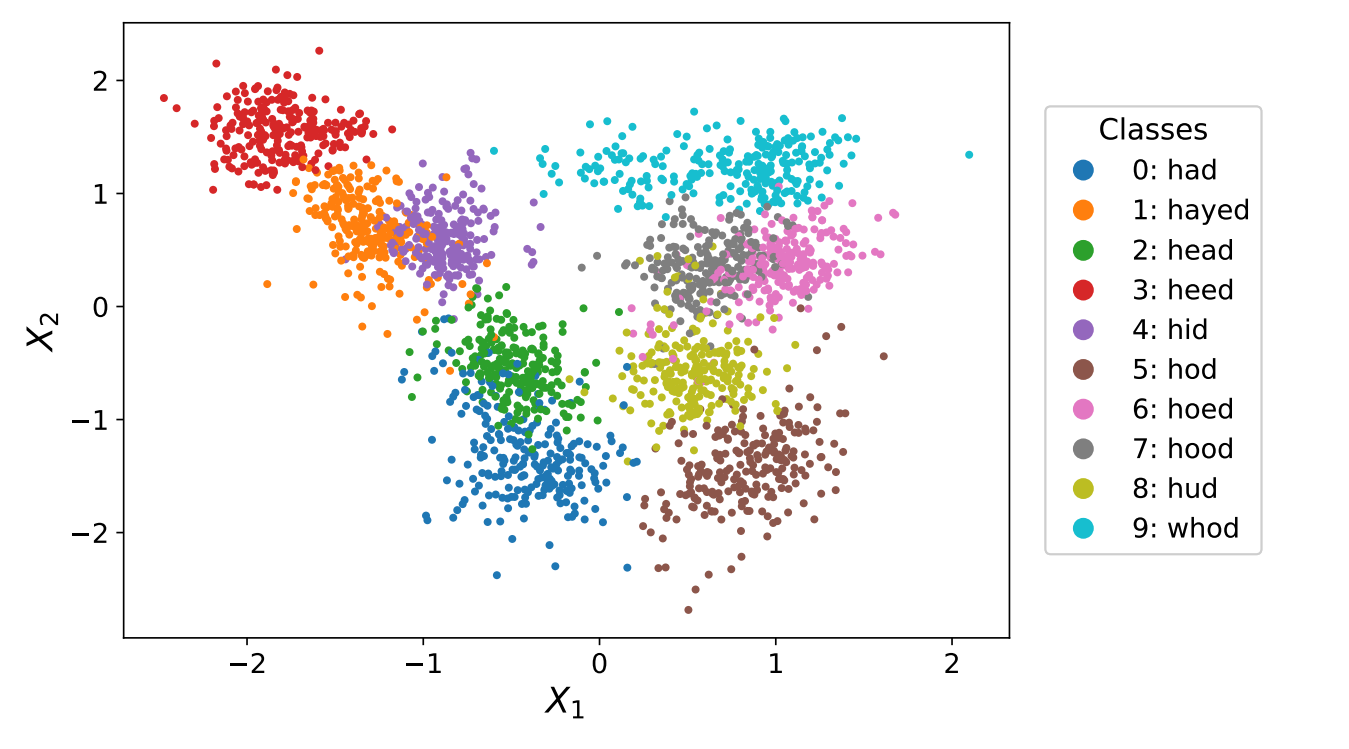



**Figure 42:** Backness ($X_1$) and height ($X_2$) measurements for ten different syllables. Each point ($n = 2371$) represents a single utterance of a syllable by a monolingual native speaker of American English.

# Categorical cross-entropy and accuracy

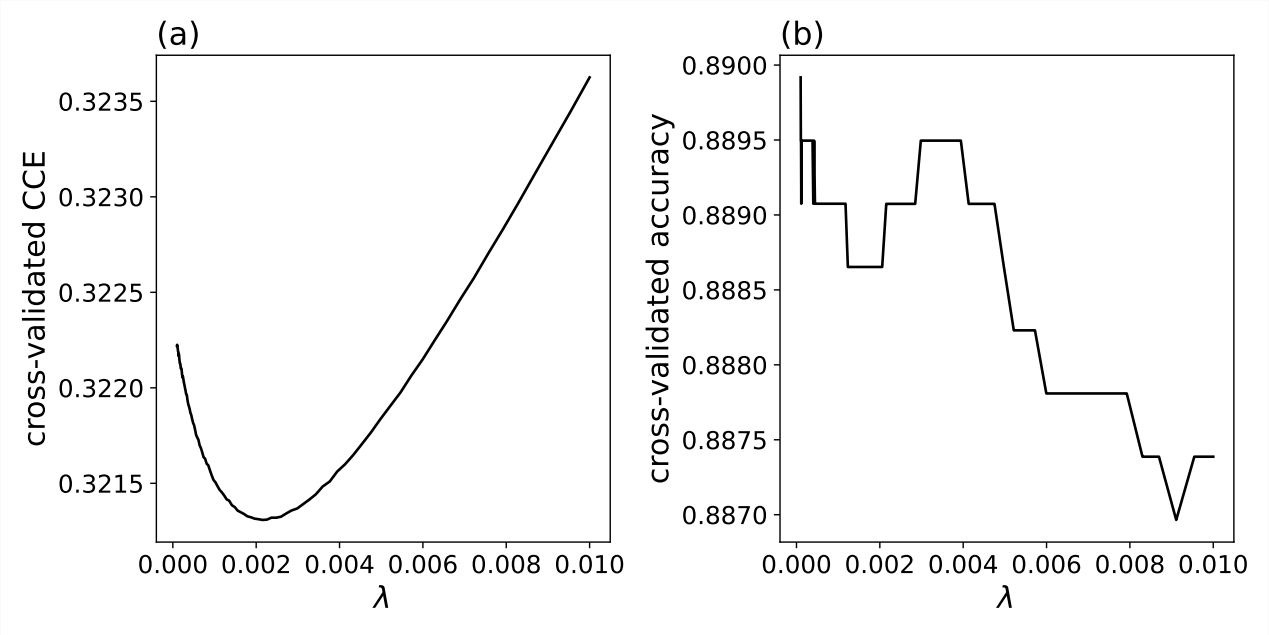



**Figure 43:** Cross-validated categorical cross-entropy loss (a) and accuracy (b) of multinomial logistic regression applied to the speech sound dataset (shown in Figure 42) as a function of the regularisation strength $\lambda$. For each value of $\lambda$, each test metric is averaged across five cross-validation folds.

Very light regularisation ($\lambda \approx 0.002$) is optimal. This is not surprising: we have a simple model (scores linear in $\boldsymbol{x}$), a small feature space (just two dimensions), and plenty of data ($n = 2371$).

# Decision regions

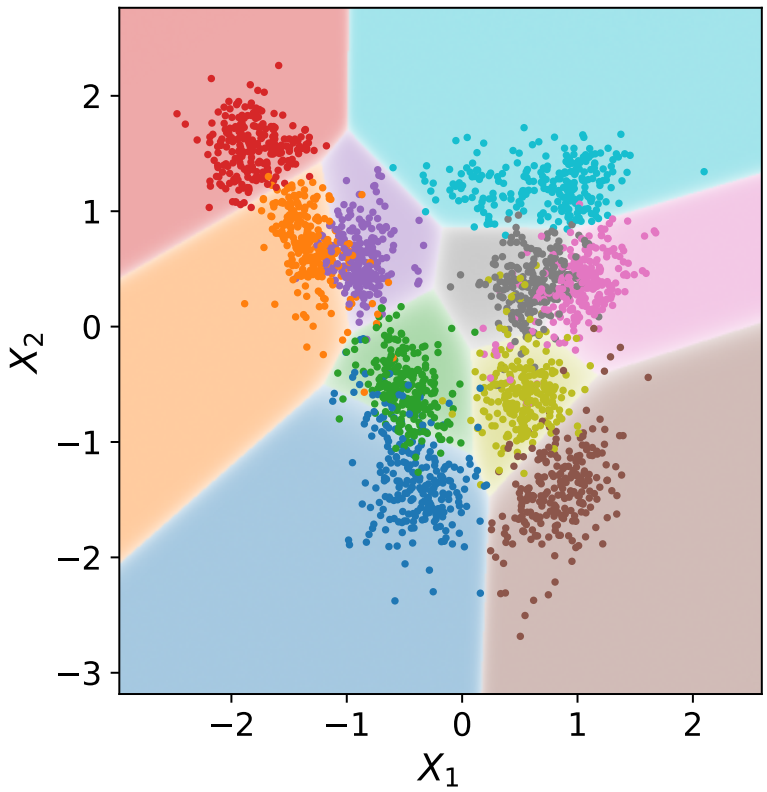



**Figure 44:** Decision regions of multinomial logistic regression model ($\lambda = 0.002$) fitted to the speech sound training data (overlaid scatter plot). Background colour represents the model's best-guess class label; background opacity represents the model's confidence in its best guess.

# The geometry of the decision region

## Question

This exercise will help you understand the geometry of the decision regions in Figure 44, and of multinomial logistic regression decision regions in general.

(a) Show that logistic regression decision regions are *convex*. That is, show that if points $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ both lie in the $i$th decision region, then every point on the line segment connecting $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ also lies in the $i$th decision region.

(b) Show that in a two-dimensional feature space, the boundary between the $i$th and $j$th decision regions (if there is one) must be a line segment, a half line or a line. (Hint: use the fact that the $i$th and $j$th components of the score vector are equal at the boundary.)

(c) What is the generalisation of the result in (b) to feature spaces of arbitrary dimensionality?

# $k$-nearest neighbours

$k$-nearest neighbours (k-NN) is a *non-parametric method*.

- No parameters to fit to the training data.
- No loss function to minimise.

It is simple and performs well in low-dimensional feature spaces.

Given a training dataset $D = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{n}$ and an unlabelled point $\boldsymbol{x}$ to classify, the procedure is as follows:

- find the $k$ training examples whose predictor vectors lie closest to $\boldsymbol{x}$ in feature space (breaking ties randomly);
- define the vector $\boldsymbol{g}(\boldsymbol{x})$ of predicted class label probabilities to be the empirical distribution of class labels within the set of $k$ nearest neighbours.

For example, if half of point $\boldsymbol{x}$'s nearest neighbours have class label 3, then $g_3(\boldsymbol{x}) = 0.5$.

# Two hyperparameters in k-NN

- The number of nearest neighbours considered, $k$, is the most obvious hyperparameter. In principle, $k$ could be any value in $\{1, 2, \ldots, n\}$, where $n$ is the size of the training set.

- The distance metric is also important. *Euclidean distance* is a common default choice, but there are other possibilities — e.g. 'Manhattan' distance, or more generally Minkowski distances of various orders. (When using any of these metrics, it is crucial to standardise features so that each predictor vector component displays a similar range of variation across the training data.)

Cross-validation is the standard way to pick a good value for $k$ and a good distance metric.

# k-NN applied to human speech sounds

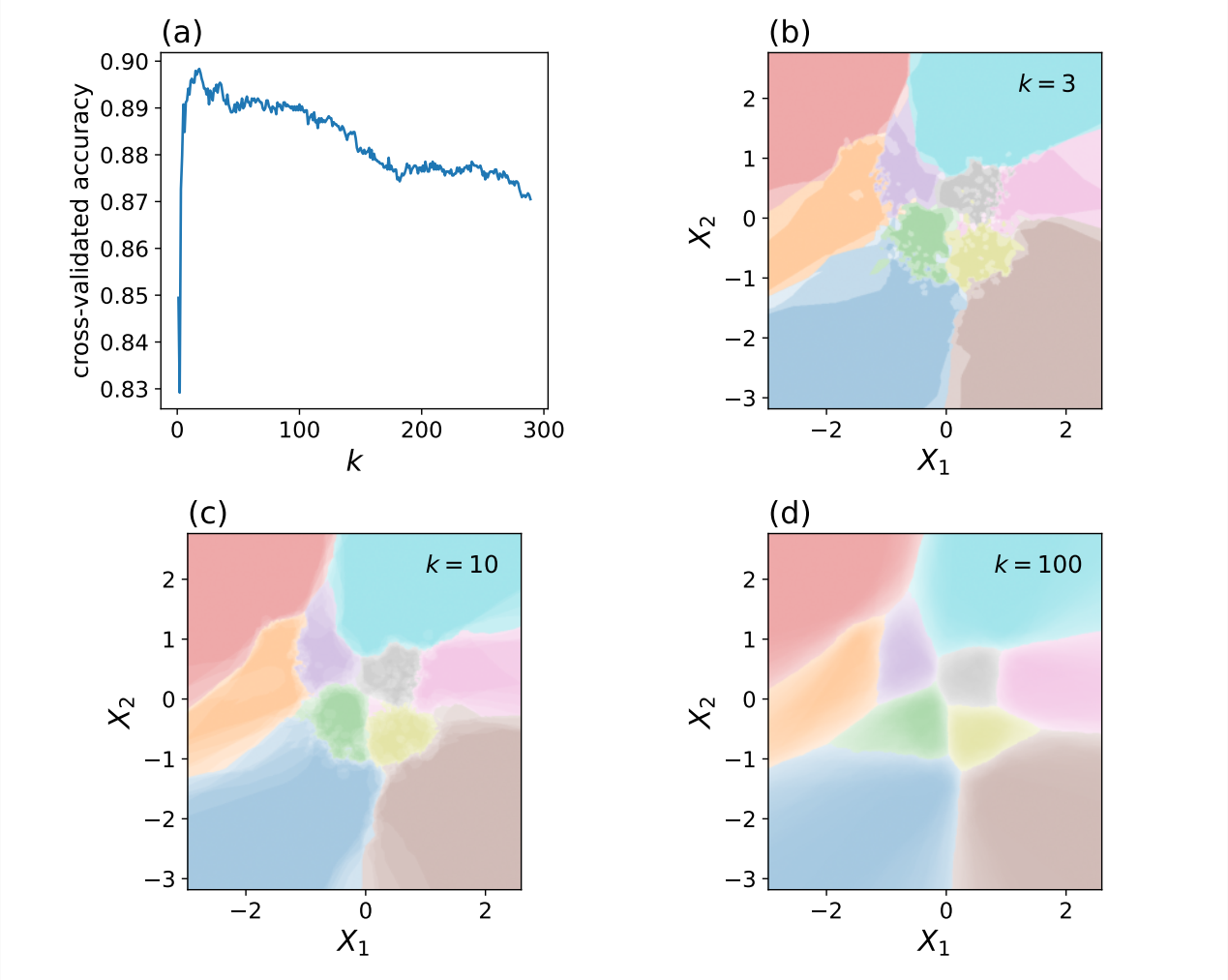



**Figure 45:** Cross-validated accuracy of k-NN applied to the human speech sounds dataset (shown in Figure 42) as a function of $k$ (sweet spot $k \approx 18$). (b)-(d) Decision regions of k-NN fitted to the dataset for $k = 3, 10$ and 100.

# k-NN works well on the human speech sounds dataset

**Question**

Suppose one picks a value of $k$ that is too large. Does the resulting classifier *underfit* or *overfit* the training data?

- k-NN is a flexible method in the sense that its decision regions can take almost any form.
- For k-NN to work well, the training points need to fill feature space rather densely.
- The human speech sound dataset meets that condition. Comparing Figure 45(a) with Figure 43(b) (with optimal $k$), we see that k-NN narrowly outperforms multinomial logistic regression on this classification task.

**Table 3:** A sample of five rows drawn from an expanded version of the speech sounds dataset plotted in Figure 42. Two additional features are now recorded: duration of vowel and dialect of speaker. The latter is a categorical variable with six levels and has been one-hot encoded in the penultimate six columns. There are 2371 rows in the full dataset.

| $X_1$ | $X_2$ | Duration | Mid-Atlantic | Midland | New England | North | South | West | Syllable |
|---|---|---|---|---|---|---|---|---|---|
| 0.47 | -0.98 | 0.193 | 0 | 0 | 1 | 0 | 0 | 0 | hud |
| -0.76 | 1.22 | 0.161 | 0 | 0 | 0 | 0 | 1 | 0 | hid |
| -0.24 | -1.14 | 0.174 | 1 | 0 | 0 | 0 | 0 | 0 | had |
| -0.78 | -0.95 | 0.320 | 0 | 0 | 0 | 1 | 0 | 0 | had |
| -1.55 | 0.88 | 0.211 | 1 | 0 | 0 | 0 | 0 | 0 | hayed |

# Logistic regression vs. k-NN in higher dimensional spaces

- We are now considering a multiclass classification problem in a nine-dimensional feature space.
- Multinomial logistic regression (with optimal regularisation) achieves a mean cross-validated accuracy of 91.4%.
- k-NN's mean cross-validated accuracy is highest when $k = 4$; the score achieved is 90.0%.
- k-NN did slightly better than logistic regression when the feature space was two-dimensional, but passing to nine dimensions has turned the tables in favour of the parametric method.
- Adding additional predictors (i.e., increasing the dimensionality of the feature space) while holding the number of training points constant *often* favours parametric over distance-based non-parametric prediction methods.

# Distance becomes less informative in high dimensions

Consider two i.i.d. $p$-dimensional random vectors, $\boldsymbol{X}$ and $\boldsymbol{X}'$, with i.i.d. components $X_1, \ldots, X_p$ and $X'_1, \ldots, X'_p$. The random variable $(X_i - X'_i)^2$ has the same expectation and the same variance for every $i$ — say, $\mu$ and $\sigma^2$ respectively. We have

$$\mathbb{E}[\|\boldsymbol{X} - \boldsymbol{X}'\|^2] = \mathbb{E}\left[\sum_{i=1}^{p}(X_i - X'_i)^2\right] = \mu p,$$

and

$$\mathrm{Var}[\|\boldsymbol{X} - \boldsymbol{X}'\|^2] = \mathrm{Var}\left[\sum_{i=1}^{p}(X_i - X'_i)^2\right] = \sigma^2 p.$$

The ratio of the standard deviation of $\|\boldsymbol{X} - \boldsymbol{X}'\|^2$ to its mean is

$$\frac{\sqrt{\mathrm{Var}[\|\boldsymbol{X} - \boldsymbol{X}'\|^2]}}{\mathbb{E}[\|\boldsymbol{X} - \boldsymbol{X}'\|^2]} = \frac{\sigma}{\mu\sqrt{p}}.$$

In the limit $p \to \infty$, this ratio approaches zero.

# Unsupervised Learning: A deeper dive

# Unsupervised learning vs. supervised learning

**Supervised learning**: using labelled data, i.e. a set of predictor-response pairs $D = \{\boldsymbol{x}_i, y_i\}_{i=1}^n$, predict $Y$ from $\boldsymbol{X}$ in new cases; or (more ambitiously) learn a model of the response density $f_{Y|\boldsymbol{X}}(y|\boldsymbol{x};\boldsymbol{\theta})$, where $\boldsymbol{\theta}$ is a parameter vector.

**Unsupervised learning**: using unlabelled data $D = \{\boldsymbol{x}_i\}_{i=1}^n$, learn what patterns or structures to expect in future outputs from the same data-generating process. Examples include

- *Density estimation*: learn a density function $f_{\boldsymbol{X}}(\boldsymbol{x};\boldsymbol{\theta})$.
- *Clustering*: find groups within the dataset.
- *Dimensionality reduction*: describe high dimensional data using a smaller number of composite features.

# Clustering

The **basic clustering task**: given a dataset $D = \{\boldsymbol{x}_i\}_{i=1}^n$ and an integer $K$, find a good partition $\{C_k\}_{k=1}^K$ of $D$. Good partitions group together *similar* points.

One way to operationalise 'similar' is via Euclidean distance: points that are close together are taken to be more similar than points that are far apart. We will now present

- a commonly-used measure of clustering quality that exploits Euclidean distance, and
- an algorithm (K-means) for performing the associated clustering task.

Then we will consider how to choose an appropriate value of $K$.

# Measuring clustering quality: total inertia

We encode partition $\{C_k\}_{k=1}^K$ of $D$ by assignment matrix $\underline{z}$, defined by

$$z_{ik} = \begin{cases} 1 & \text{if } \boldsymbol{x}_i \in C_k \\ 0 & \text{otherwise.} \end{cases}$$

We define the *centroid* of cluster $C_k$ to be the mean of its points,

$$\bar{\boldsymbol{x}}_k = \frac{1}{|C_k|} \sum_{i \in C_k} \boldsymbol{x}_i = \frac{\sum_{i=1}^n z_{ik} \boldsymbol{x}_i}{\sum_{i=1}^n z_{ik}},$$

and the *within-cluster variation* or *inertia* of $C_k$ to be

$$w(C_k) = \frac{1}{2|C_k|} \sum_{i,j \in C_k} \|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2 = \sum_{i \in C_k} \|\boldsymbol{x}_i - \bar{\boldsymbol{x}}_k\|^2.$$

The *total inertia* is then

$$W(\underline{z}) = \sum_{k=1}^K w(C_k) = \sum_{i=1}^n \sum_{k=1}^K z_{ik} \|\boldsymbol{x}_i - \bar{\boldsymbol{x}}_k\|^2.$$

We'll take this as our measure of clustering quality. (Lower is better.)

# K-means: the idea

The number of valid assignment matrices (a single 1 in each row, and at least one 1 in each column) is approximately $K^n$. Finding a matrix that achieves the lowest total inertia in this large space of possibilities is usually very difficult.

The K-means algorithm finds a good clustering, but in general not an optimal one.

In a good clustering, we would expect every point to lie closer to its own cluster's centroid than to the centroids of any of the other clusters.

- A typical proposed clustering will not satisfy this condition. But then we can *improve* it by reassigning points to the clusters whose centroids they lie closest to, decreasing total inertia.
- K-means continues reassigning points until no further improvements of this kind are possible.

# K-means: the algorithm

We begin by randomly initialising $K$ centroids, $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K$. (Commonly, a subset of $K$ datapoints is selected, using a random procedure that favours well-separated points.) Then we repeat the following two steps until assignments stop changing.

(i) **E-step**. For each point $i$, (re)calculate its cluster membership:

$$z_{ik} \leftarrow \begin{cases} 1 & \text{if } k = \underset{1 \leq j \leq K}{\arg\min} \|\boldsymbol{x}_i - \boldsymbol{\mu}_j\|^2 \\ 0 & \text{otherwise.} \end{cases}$$

(ii) **M-step**. For each cluster, recalculate its centroid:

$$\boldsymbol{\mu}_k \leftarrow \frac{\sum_{i=1}^n z_{ik} \boldsymbol{x}_i}{\sum_{i=1}^n z_{ik}}.$$

(The reasons for the labels 'E' and 'M' will become clear later.) In practice we run the whole algorithm several times, from different random initialisations, and pick the clustering with the lowest inertia.

# Applications to human speech sounds

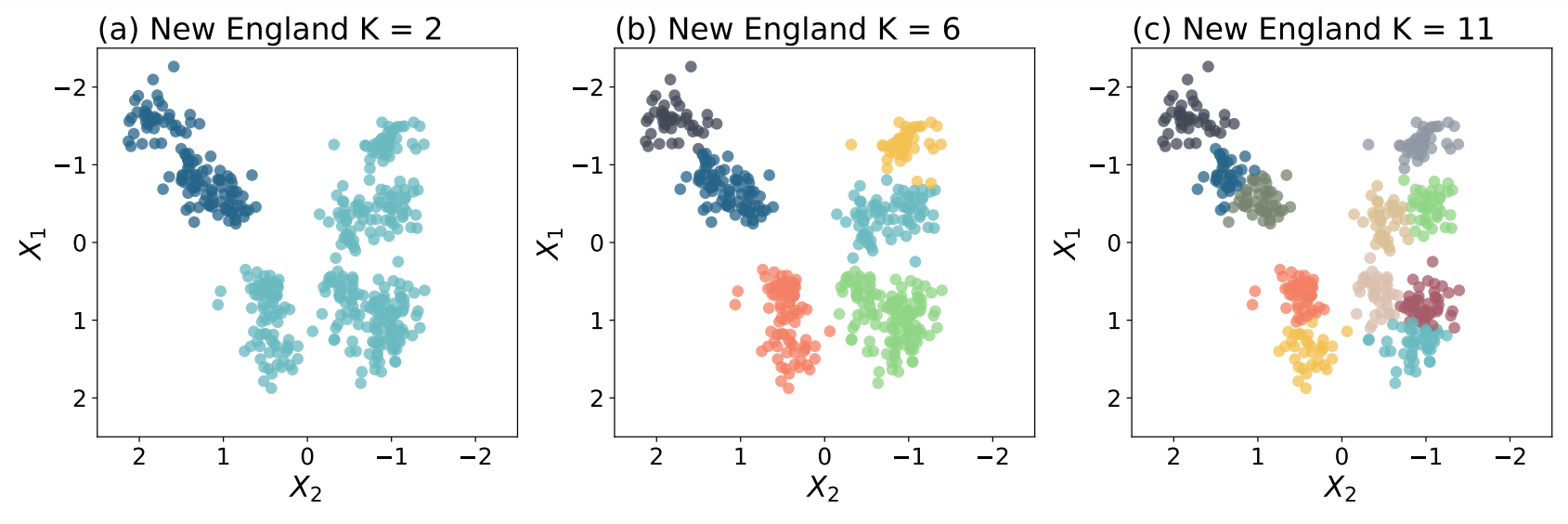



**Figure 46:** K-means clusters of New England vowel features with three different $K$ values.

# Choosing $K$

- Choosing the right number of clusters is a crucial step in cluster analysis.
- In some applications we will know $K$ in advance, but often we won't. So we need a data-driven method for picking $K$.
- For a given $K$, the total inertia $W(\hat{\underline{z}})$ measures how well a clustering explains the data. (Lower is better.)
- Unfortunately, increasing $K$ is guaranteed to reduce $W(\hat{\underline{z}})$. So we can't pick $K$ by minimising $W(\hat{\underline{z}})$.
- We'll present two common methods for picking $K$:
  - elbow method
  - null reference method

# Elbow method

Elbow method: make $K$ just large enough to explain the patterns, but no larger.

Key idea: if $K^*$ is the *true* number of clusters, then increasing $K$ by one should yield a *substantial* drop in inertia if and only if $K < K^*$.

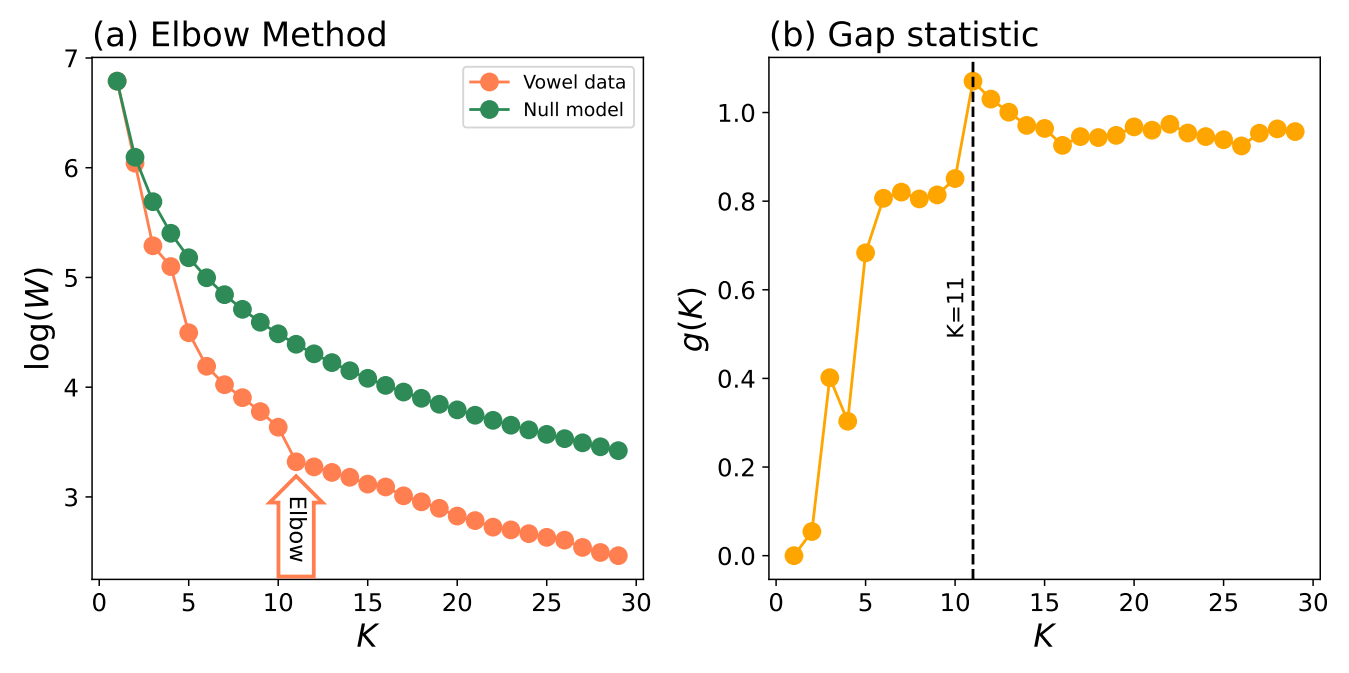



**Figure 47:** (a) Relationship between log inertia and $K$ for New England vowel data and a single cluster null dataset. (b) The corresponding gap statistic.

# Null reference method

Null reference method: compare the inertias of clusterings of (i) the real dataset and (ii) a null dataset with no 'real' clusters.

The null dataset consists of $n$ points uniformly distributed within the smallest hypercube – call its volume $V$ – that contains all the actual datapoints. What would happen if we ran K-means on this dataset?

Approximation: each resulting cluster is a set of points uniformly sampled from a hypercube of volume $V/K$, side length $s = (V/K)^{1/p}$. Relative to the centre of its cluster, a random point has coordinates $\boldsymbol{U} = (U_1, U_2, \ldots, U_p)$, where $U_i \sim \text{Uniform}\left(-\frac{s}{2}, \frac{s}{2}\right)$.

The total inertia of the null dataset on this clustering is

$$W_K^{\text{null}} = n\mathbb{E}\left(\|\boldsymbol{U}\|^2\right) = \frac{nps^2}{12} = \frac{np}{12}\left(\frac{V}{K}\right)^{\frac{2}{p}}.$$

## Exercise

Show that if $U_i \sim \text{Uniform}\left(-\frac{s}{2}, \frac{s}{2}\right)$ then $\mathbb{E}(U_i^2) = \frac{s^2}{12}$. Hence show that if $\boldsymbol{U} = (U_1, U_2, \ldots, U_p)$ then $\mathbb{E}\left(\|\boldsymbol{U}\|^2\right) = \frac{ps^2}{12}$.

# Null reference method

$W_K$ is the inertia of the *real* dataset partitioned into $K$ clusters. Define

$$r_K = \frac{W_K^{\text{null}}}{W_K} \propto \frac{K^{-\frac{2}{p}}}{W_K}.$$

For given $K$, the larger the value of $r_K$ the more we have explained relative to the null data, where clustering explains nothing.

The *gap statistic* is the logarithm of the scaled ratio $cK^{-2/p}/W_K$,

$$g(K) := \log \frac{W_1}{W_K} - \frac{2}{p} \log K,$$

where we choose $c = W_1$ so that $g(1) = 0$.

The gap statistic compares the inertia reduction factor $\log(W_1/W_k)$ from clustering the real dataset with the inertia reduction factor we would have achieved with the null dataset.

# Null reference method

In the gap statistic definition, $g(K) = \log \frac{W_1}{W_K} - \frac{2}{p} \log K$,

- $\log \frac{W_1}{W_K}$ measures how well we explained the patterns.
- $\frac{2}{p} \log K$ measures how complicated our explanation was.
- The best explanation maximises the gap.

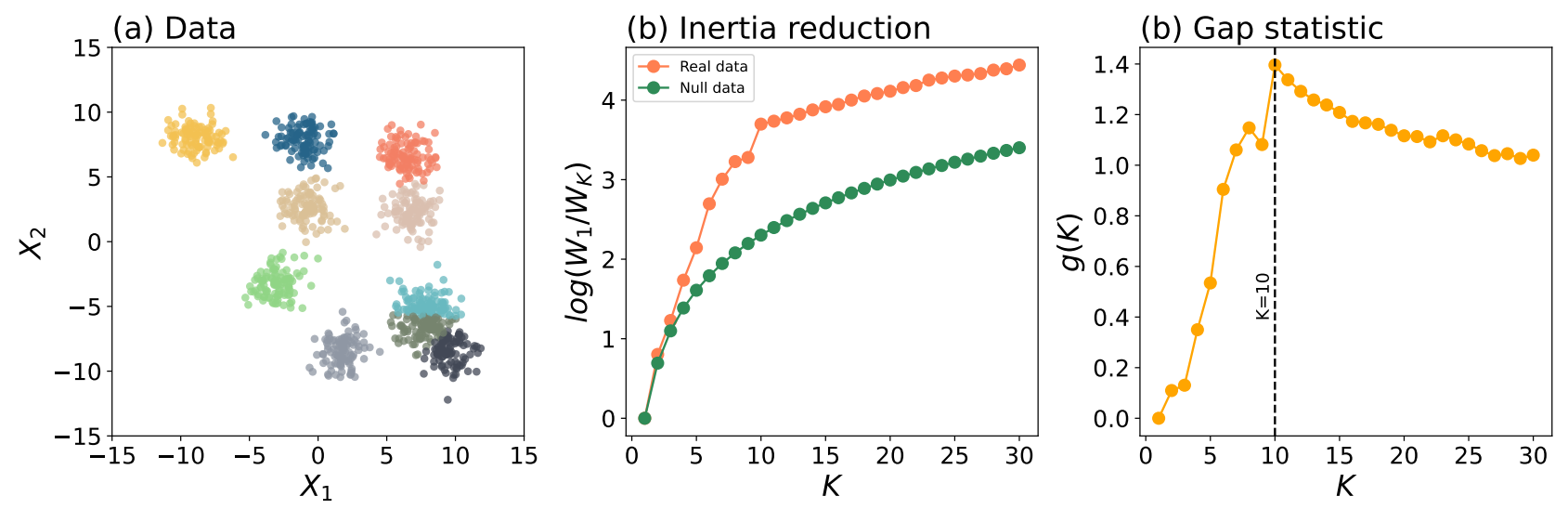



**Figure 48:** (a) A dataset consisting of samples from ten randomly located Gaussian blobs. (b) Inertia reduction measures for the data and the null data. (c) The gap statistic and estimate $\hat{K} = 10$.

# Density estimation with Gaussian mixtures

- Given a dataset of $n$ observations $\{\boldsymbol{x}_i\}_{i=1}^n$ of a random vector $\boldsymbol{X} \in \mathbb{R}^p$, we may wish to estimate $\boldsymbol{X}$'s probability density function.
- Density estimation is a central problem in statistical modelling and machine learning.
- Here, we will consider the special case in which density estimates are constrained to be *mixtures of Gaussian* distributions.

# Gaussian mixtures in one dimension

Given observations $\{x_i\}_{i=1}^n$ of a random variable $X$, a histogram is a non-parametric approximation to $X$'s density. Gaussian mixture models (GMMs) offer a simple parametric approximation strategy. They assume that $X$'s density is a weighted sum of Gaussian densities.

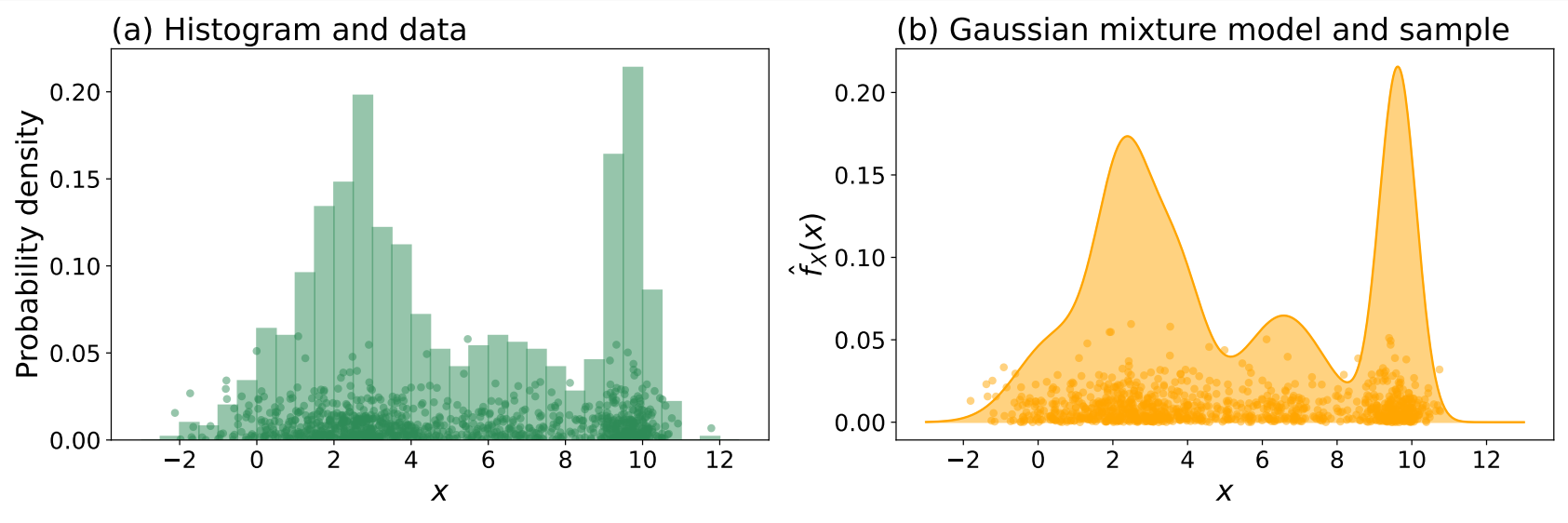



**Figure 49:** (a) Histogram computed from $n = 1000$ realisations of a random variable $X$ with unknown density. The individual datapoints are also shown. (b) A $K = 5$ component Gaussian mixture model fitted to the same dataset, along with $n = 1000$ sample.

# Maximum likelihood estimate

The density of a Gaussian mixture model with $K$ components is

$$f_X(x; \boldsymbol{\theta}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x|\mu_k, \sigma_k^2),$$

where the weights satisfy $\sum_{k=1}^{K} \pi_k = 1$ and $\pi_k \geq 0$.

The parameter vector, $\boldsymbol{\theta} = (\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\sigma})$, contains the means and variances of individual Gaussians, and their weights.

We estimate $\boldsymbol{\theta}$ by maximising the log-likelihood function

$$\ell(\boldsymbol{\theta}) = \log \mathcal{L}(\boldsymbol{\theta}) = \log \prod_{x \in D_{train}} f_X(x; \boldsymbol{\theta}) = \sum_{x \in D_{train}} \log f_X(x; \boldsymbol{\theta}).$$

# Ancestral sampling

Sampling from the density defined by a GMM is a two-step process.

1. Select a component according to the probability weights $\boldsymbol{\pi}$.
2. Sample from the Gaussian density associated with this component.

Component selection can be thought of as realising a one-hot random vector $\boldsymbol{Z} = (Z_1, Z_2, \ldots, Z_K)$, where $Z_k = 1$ if and only if the component $k$ is selected. So we have

$$\mathsf{Pr}(\boldsymbol{Z} = \boldsymbol{e}_k) = \pi_k \text{ for } k = 1, 2, \ldots, K,$$

where $\boldsymbol{e}_k$ is the $k$th standard unit vector. The probability mass function of $\boldsymbol{Z}$ is then

$$f_{\boldsymbol{Z}}(\boldsymbol{z}) = \prod_{k=1}^{K} \pi_k^{z_k},$$

where $0^0 = 1$ in calculations.

# Ancestral sampling

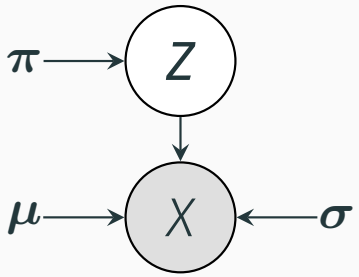
The probability density of $X$ given $\boldsymbol{z}$ is

$$f_{X|\boldsymbol{z}}(x|\boldsymbol{z}) = \prod_{k=1}^{K} \mathcal{N}(x|\mu_k, \sigma_k^2)^{z_k},$$

and the joint distribution of $X$ and $\boldsymbol{z}$ is

$$f_{X,\boldsymbol{z}}(x, \boldsymbol{z}) = f_{X|\boldsymbol{z}}(x|\boldsymbol{z}) f_{\boldsymbol{z}}(\boldsymbol{z}).$$

The datapoint-generating process has DAG representation




where $\boldsymbol{\mu} = (\mu_1, \mu_2, \ldots, \mu_K)$ and $\boldsymbol{\sigma} = (\sigma_1, \sigma_2, \ldots, \sigma_K)$ are the parameters of the conditional density $f_{X|\boldsymbol{z}}(x|\boldsymbol{z})$ and $\boldsymbol{z}$ is a *latent* variable.

Generating datapoints in this way is known as *ancestral sampling*.

# Soft and hard cluster assignment

Let $(X, \boldsymbol{Z})$ be the observed and latent variables for a single entity selected at random. The weight vector $\boldsymbol{\pi}$ gives the *prior* mass function for $\boldsymbol{Z}$, *before* observing $X$.

The *posterior* probability distribution of $\boldsymbol{Z}$, after observing $X$, is

$$f_{\boldsymbol{Z}|X}(\boldsymbol{z}|x) = \frac{f_{X|\boldsymbol{Z}}(x|\boldsymbol{z}) f_{\boldsymbol{Z}}(\boldsymbol{z})}{f_X(x)},$$

where $f_X(x) \equiv f_X(x; \boldsymbol{\theta})$. Written in terms of the model parameters,

$$f_{\boldsymbol{Z}|X}(\boldsymbol{e}_k|x) = \Pr(\boldsymbol{Z} = \boldsymbol{e}_k|X = x) = \frac{\pi_k \mathcal{N}(x|\mu_k, \sigma_k^2)}{f_X(x; \boldsymbol{\theta})}.$$

- This posterior mass function gives *soft* cluster assignments for an observed point $\boldsymbol{x}$.
- MAP estimation yields a *hard* cluster assignment for $\boldsymbol{x}$:

$$\hat{\boldsymbol{z}} = \operatorname*{arg\,max}_{1 \leq k \leq K} f_{\boldsymbol{Z}|X}(\boldsymbol{e}_k|x).$$

# Choosing $K$ by cross-validation

Partition the dataset $D$ into $K$ equal-sized folds, $D_1, D_2, \ldots, D_K$, and repeat the following steps for each fold $D_k$.

(i) Remove the fold from $D$ leaving $D_{-k} = D \setminus D_k$.

(ii) Fit the density model to $D_{-k}$, producing parameter estimate $\hat{\boldsymbol{\theta}}_{-k}$.

(iii) Compute the sample entropy for this fold,

$$\hat{\mathbb{H}}_k = -\frac{1}{|D_k|} \sum_{x \in D_k} \log f_X(x; \hat{\boldsymbol{\theta}}_{-k}).$$

Compute the overall estimate of sample entropy on one unseen fold,

$$\hat{\mathbb{H}}_{\mathrm{CV}}(K) = \frac{1}{K} \sum_{k=1}^{K} \hat{\mathbb{H}}_k.$$

Finally, select $K$ as

$$\hat{K} = \arg\min_K \hat{\mathbb{H}}_{\mathrm{CV}}(K).$$

# Comparing with K-means

The cross entropy of $K$-component GMM fitted to the New England vowel data shows that the minimum cross extropy is achieved when $K = 11$, matching the optimal number of clusters for K-means.

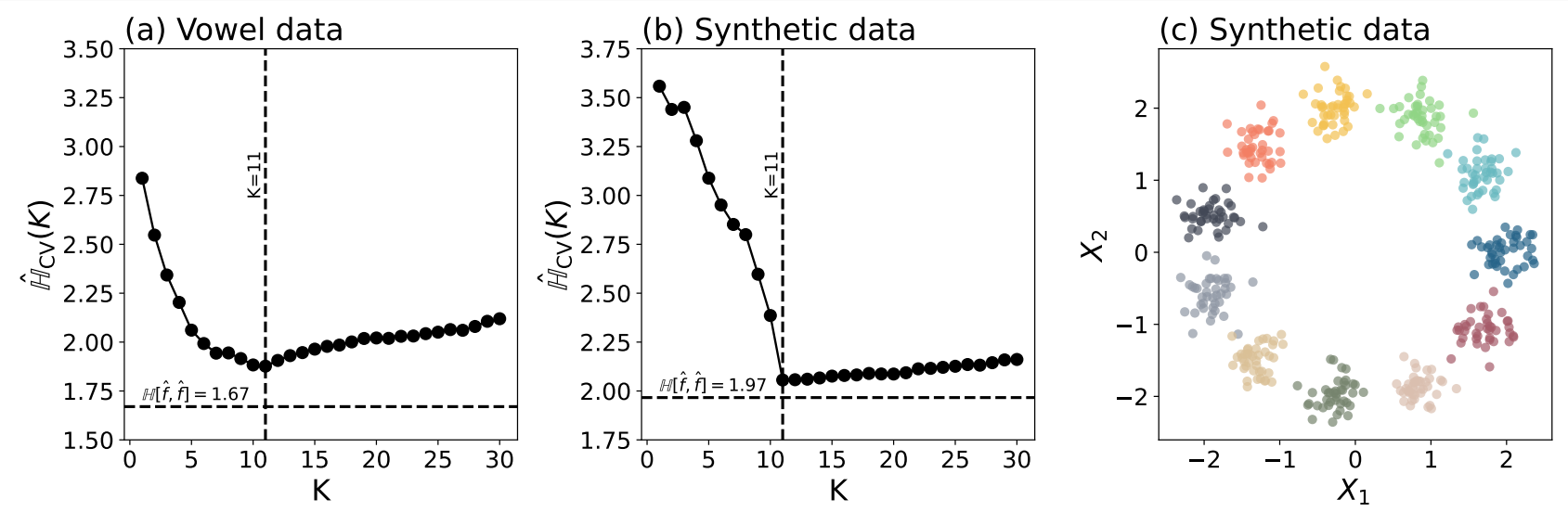



**Figure 50:** Cross-validation estimates of the cross entropy of $K$-component GMM fitted to (a) New England vowel data and (b) a synthetic dataset shown in (c).

# EM algorithm for Gaussian mixture model

*Expectation maximisation* (EM) algorithm is used to find maximum likelihood (or MAP) estimates in latent variable models in general.

For a GMM with general multivariate Gaussian distributions,

$$f_Z(z;\boldsymbol{\theta}) = \prod_{k=1}^{K} \pi_k^{z_k}$$

$$f_{X|Z}(x|z;\boldsymbol{\theta}) = \prod_{k=1}^{K} \mathcal{N}(x|\boldsymbol{\mu}_k, \underline{\boldsymbol{\Sigma}}_k)^{z_k}.$$

The joint distribution of a single observation $(X, Z)$ is

$$f_{X,Z}(x, z;\boldsymbol{\theta}) = \prod_{k=1}^{K} (\pi_k \mathcal{N}(x|\boldsymbol{\mu}_k, \underline{\boldsymbol{\Sigma}}_k))^{z_k} .$$

The joint distribution of the complete dataset $(\underline{X}, \underline{Z})$ is

$$f_{\underline{X},\underline{Z}}(\underline{x}, \underline{z};\boldsymbol{\theta}) = \prod_{i=1}^{n} \prod_{k=1}^{K} (\pi_k \mathcal{N}(x_i|\boldsymbol{\mu}_k, \underline{\boldsymbol{\Sigma}}_k))^{z_{ik}} .$$

# EM algorithm for Gaussian mixture model

The logarithm of the density of the complete dataset is

$$\log f_{\underline{X},\underline{Z}}(\underline{x},\underline{Z};\boldsymbol{\theta}) = \sum_{i=1}^{n}\sum_{k=1}^{K} z_{ik}\left(\log \pi_k + \log \mathcal{N}(\boldsymbol{x}_i|\boldsymbol{\mu}_k,\underline{\boldsymbol{\Sigma}}_k)\right).$$

This is a random function of $\boldsymbol{\theta}$, since $\underline{Z}$ is unobserved. Let $\boldsymbol{\theta}_t$ be our current proposal for a 'good' choice of parameter vector. It picks out a posterior distribution for $\underline{Z}$. We define the expectation of the random function above with respect to this distribution:

$$Q(\boldsymbol{\theta},\boldsymbol{\theta}_t) = \sum_{i=1}^{n}\sum_{k=1}^{K} \mathbb{E}(Z_{ik}|X=\boldsymbol{x}_i;\boldsymbol{\theta}_t)\left(\log \pi_k + \log \mathcal{N}(\boldsymbol{x}_i|\boldsymbol{\mu}_k,\underline{\boldsymbol{\Sigma}}_k)\right).$$

We also define the *responsibility* of component $k$ for observation $i$ to be the probability that $\boldsymbol{x}_i$ came from cluster $k$:

$$r_{ik} = \mathbb{E}(Z_{ik}|X=\boldsymbol{x}_i;\boldsymbol{\theta}) = \Pr(Z_{ik}=1|X=\boldsymbol{x}_i;\boldsymbol{\theta}) = f_{Z|X}(\boldsymbol{e}_k|\boldsymbol{x}_i;\boldsymbol{\theta}).$$

The responsibility is a function of $\boldsymbol{\theta}$. We'll write $r_{ik}(t)$ for the responsibility evaluated at $\boldsymbol{\theta}=\boldsymbol{\theta}_t$.

# EM algorithm for Gaussian mixture model

We have

$$Q(\boldsymbol{\theta}, \boldsymbol{\theta}_t) = \sum_{i=1}^{n} \sum_{k=1}^{K} r_{ik}(t) \left( \log \pi_k + \log \mathcal{N}(\boldsymbol{x}_i | \boldsymbol{\mu}_k, \underline{\boldsymbol{\Sigma}}_k) \right).$$

The core idea of the EM algorithm is that we can compute a better parameter vector proposal, $\boldsymbol{\theta}_{t+1}$, by maximising $Q$ with respect to $\boldsymbol{\theta}$:

$$\boldsymbol{\theta}_{t+1} = \arg\max_{\boldsymbol{\theta}} Q(\boldsymbol{\theta}, \boldsymbol{\theta}_t).$$

We can carry out this maximisation step (the 'M' step) cheaply – closed-form formulae exist. Then we recompute the expectations that define responsibilities (the 'E' step) and repeat. The procedure generates a sequence of parameter vector estimates $\boldsymbol{\theta}_0, \boldsymbol{\theta}_1, \ldots$ that is non-decreasing in likelihood and usually converges to a local maximum of the likelihood function.

# EM algorithm for GMM: the E-step

After each parameter vector update, we must recalculate responsibilities. Recall that responsibilities are defined by

$$r_{ik} = \mathbb{E}(Z_{ik}|X = x_i; \boldsymbol{\theta}) = f_{Z|X}(\boldsymbol{e}_k|x_i; \boldsymbol{\theta}).$$

**Exercise**

Make use of the relationship $f_{Z|X}(z|x; \boldsymbol{\theta}) = f_{X,Z}(x, z; \boldsymbol{\theta})/f_X(x; \boldsymbol{\theta})$, i.e.,

$$f_{Z|X}(\boldsymbol{e}_k|x; \boldsymbol{\theta}) = \frac{\pi_k \mathcal{N}(x|\boldsymbol{\mu}_k, \underline{\boldsymbol{\Sigma}}_k)}{f_X(x; \boldsymbol{\theta})},$$

to show that

$$r_{ik} = \frac{\pi_k \mathcal{N}(x|\boldsymbol{\mu}_k, \underline{\boldsymbol{\Sigma}}_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(x|\boldsymbol{\mu}_j, \underline{\boldsymbol{\Sigma}}_j)}.$$

# EM algorithm for GMM: the M-step

The closed-form formulae for the M-step, in which we replace our current parameter vector proposal by a better one, can be shown to be

$$\pi_k \leftarrow \frac{n_k}{n}, \quad \boldsymbol{\mu}_k \leftarrow \frac{1}{n_k} \sum_{i=1}^{n} r_{ik} \boldsymbol{x}_i,$$

$$\underline{\boldsymbol{\Sigma}}_k \leftarrow \frac{1}{n_k} \sum_{i=1}^{n} r_{ik} (\boldsymbol{x}_i - \boldsymbol{\mu}_k)(\boldsymbol{x}_i - \boldsymbol{\mu}_k)^T,$$

where

$$n_k = \sum_{i=1}^{n} r_{ik}.$$

# A summary of EM algorithm for GMM

The density $f_X(\boldsymbol{x};\boldsymbol{\theta}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_k, \underline{\boldsymbol{\Sigma}}_k)$, where $\boldsymbol{\theta}$ is a parameter vector containing $\{\pi_k, \boldsymbol{\mu}_k, \underline{\boldsymbol{\Sigma}}_k\}_{k=1}^{K}$, defines a $K$-component GMM. Given a dataset $D = \{\boldsymbol{x}_i\}_{i=1}^{n}$, the log likelihood is $\ell(\boldsymbol{\theta}) = \sum_{i=1}^{n} \log f_X(\boldsymbol{x}_i;\boldsymbol{\theta})$. The procedure below generates a sequence of parameter vector estimates $\boldsymbol{\theta}_0, \boldsymbol{\theta}_1, \ldots$ that is non-decreasing in likelihood and usually converges to a local maximum of the likelihood function.

Initialise $\boldsymbol{\theta}_0$ and generate a sequence of parameter vector estimates $\boldsymbol{\theta}_0, \boldsymbol{\theta}_1, \ldots$ by repeating the following two steps until $|\ell(\boldsymbol{\theta}_{t+1}) - \ell(\boldsymbol{\theta}_t)| < \epsilon$, where $\epsilon > 0$ is small tolerance.

(i) **E-step**: $r_{ik} \leftarrow \frac{\pi_k \mathcal{N}(\boldsymbol{x}_i|\boldsymbol{\mu}_k, \underline{\boldsymbol{\Sigma}}_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\boldsymbol{x}_i|\boldsymbol{\mu}_j, \underline{\boldsymbol{\Sigma}}_j)}, \quad n_k \leftarrow \sum_{i=1}^{n} r_{ik}.$

(ii) **M-step**:

$$\pi_k \leftarrow \frac{n_k}{n}, \quad \boldsymbol{\mu}_k \leftarrow \frac{1}{n_k} \sum_{i=1}^{n} r_{ik} \boldsymbol{x}_i,$$

$$\underline{\boldsymbol{\Sigma}}_k \leftarrow \frac{1}{n_k} \sum_{i=1}^{n} r_{ik} (\boldsymbol{x}_i - \boldsymbol{\mu}_k)(\boldsymbol{x}_i - \boldsymbol{\mu}_k)^T.$$

# EM algorithm and K-means

- The vector of responsibilities $\boldsymbol{r}_i = (r_{i1}, \ldots, r_{iK})^T$ for the datapoint $\boldsymbol{x}_i$ is the *soft version* of K-means one-hot variable $\boldsymbol{z}_i$ that assigns *all* responsibility to *one* cluster.
- Both algorithms are iterative, with each iteration involving two steps:
  - The E-step computes assignments — soft for mixtures, hard for clustering.
  - The M-step updates cluster means — maximising likelihood for mixtures and minimising inertia for clustering.
- K-means can be understood as a limiting case of GMM.
- We can use the cluster assignments output by K-means to initialise the responsibilities in GMM, from which the initial parameters $\boldsymbol{\theta}_0 = \{\pi_k, \boldsymbol{\mu}_k, \underline{\boldsymbol{\Sigma}}_k\}_{k=1}^K$ can be calculated.