MAT 3373
Methods of Machine Learning

Chapter 2
Regression and Value Estimation

P. Boily (uOttawa)

Fall – 2023

Outline

Outline

Outline

Main Reference:

- *Data Understanding, Data Analysis, and Data Science*, chapter 20.

2 – Regression and Value Estimation

In Chapter 1 (*Machine Learning 101*), we provided a (basically) math-free general overview of machine learning.

In this chapter, we begin our mathematical treatment of **supervised learning**, with a focus on **regression** and **value estimation**, as well as **parametric methods**.

This is a prelude to Chapters 3 (*Classification and Supervised Learning*) and 4 (*Clustering*), although we could also go straight to Chapter 5 (*Additional Topics*) from here.

2.1 – Statistical Learning

Statistical learning (**SL**, **UL**, **SSL**, **RL**) is a series of **procedures** and **approaches** that allows analysts to tackle problems such as:

- identifying **risk factors** associated with **breast/prostate cancer**;
- predicting if a patient is likely to have a 2nd **heart attack** within 30 days of the 1st using **demographics**, **diet**, **clinical measurements**, etc.;
- establishing a link between **salary** and **demographic information** in population survey data;
- predicting the yearly **inflation rate** using various **economical indicators**, etc.

2.1.1 – Supervised Learning Framework

In the **SL** environment, the **outcome** is denoted by Y, and the vector of p **predictors** by $\vec{X} = (X_1, \ldots, X_p)$.

- if Y is **quantitative** (price, height), predicting Y in terms of $\vec{X}$ is a **regression** task;
- if Y takes on values in a **finite unordered set** (survived/died, colours, vegetation types, etc.), it is a **classification** task.

Either of these is achieved with the use of **training data** (historical observations or cases), denoted by $[\mathbf{X} \mid \mathbf{Y}]$.

We say of such data that it is **labeled** by $\mathbf{Y}$.

The observation IDs are not part of the data. With n observations, we have:

obs.	**predictor** 1		**predictor** p	**resp.**
1	$x_{1,1}$	$\cdots$	$x_{1,p-1}$	y_1
$\vdots$	$\vdots$		$\vdots$	$\vdots$
n	$x_{n,1}$	$\cdots$	$x_{n,p-1}$	y_n

SL objectives:

- "accurately" predict **unseen test** cases;
- understand which inputs **affect** outcomes (if any), and how;
- assess quality of **predictions/inferences** made from training data.

Statistical Learning vs. Machine Learning

The term "statistical learning" is not used frequently in practice; we speak instead of **machine learning**. If a distinction must be made, we could argue that:

- **statistical learning** arises from statistical-like models, and the emphasis is usually placed on **interpretability**, **precision**, and **uncertainty**;
- machine learning arise from artificial intelligence studies, with emphasis on **large scale applications** and **prediction accuracy**.

The dividing line is **blurry** – the vocabulary used by practitioners mostly belies their **educational backgrounds**.

Motivating Example

Throughout this chapter (and the next 3), we illustrate ML concepts and notions *via* the `Gapminder` dataset, which records socio-demographic information relating to the planet's nations, from 1960 to 2011.

More information can be found at the **Gapminder Foundation**'s website: `https://www.gapminder.org`

(Code snippets can be found in DUDADS, in the appropriate sections).

For now, we are interested in finding links between **life expectancy**, at various times, and the other variables (which become **predictors**).

The dataset contains 7139 observations of 9 variables.

- identifier (i, X_1): **country $\times$ year**
- 2 categorical predictors:
  - X_2: **region**
  - X_3: **continent**
- 4 numerical predictors:
  - X_4: **population**
  - X_5: **infant mortality**
  - X_6: **fertility**
  - X_7: **gross domestic product in 1999 dollars**
- numerical response Y: **life expectancy**

```
'data.frame':   7139 obs. of  9 variables:
 $ country          : Factor w/ 185 levels "Albania","Algeria",..: 2 5 8 9 11 ...
 $ year             : int  1960 1960 1960 1960 1960 1960 1960 1960 1960 1960 ...
 $ region           : Factor w/ 22 levels "Australia and New Zealand",..: 11 ...
 $ continent        : Factor w/ 5 levels "Africa","Americas",..: 1 2 5 4 2 3 ...
 $ population       : int  11124892 20619075 10292328 ...
 $ infant_mortality: num  148.2 59.9 20.3 37.3 51 ...
 $ fertility        : num  7.65 3.11 3.45 2.7 4.5 6.73 4.33 2.6 6.28 6.7 ...
 $ gdp              : num  1.38e+10 1.08e+11 9.67e+10 5.24e+10 1.31e+09 ...
 $ life_expectancy : num  47.5 65.4 70.9 68.8 62 ...
```

We are looking for **models** of the form

$$Y = \underbrace{f(X_1, \ldots, X_7)}_{\substack{\text{systematic component} \\ \text{of } Y \text{ explained by } X}} + \varepsilon \equiv f(\vec{X}) + \underbrace{\varepsilon,}_{\mathrm{E}(\varepsilon) = 0} \qquad \varepsilon : \textbf{random error term}$$

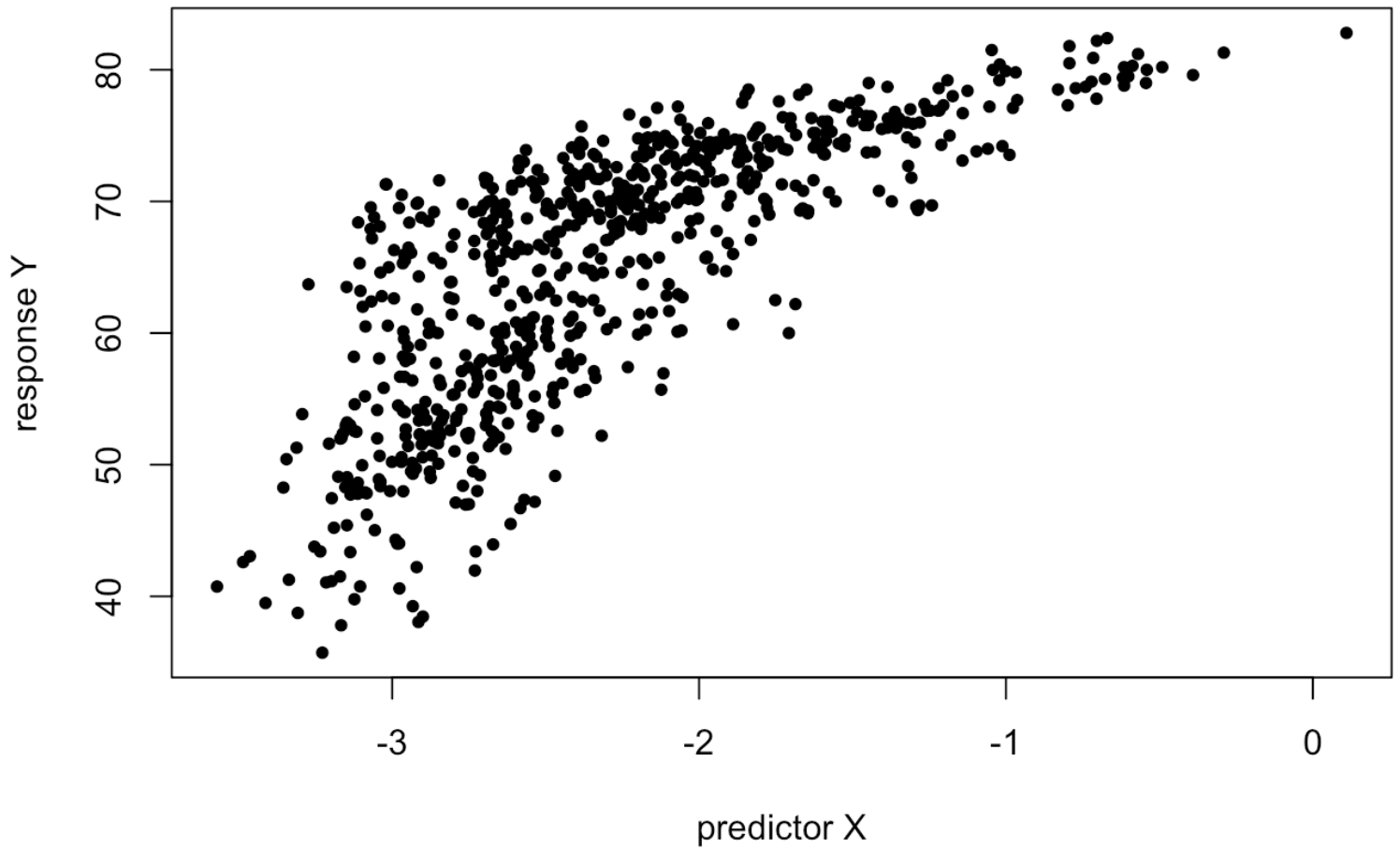
2.1.2 – Systematic Component and Regression

The **systematic component** is the engine for **predictions** and **inferences**. As long as f is "**good**", we can:

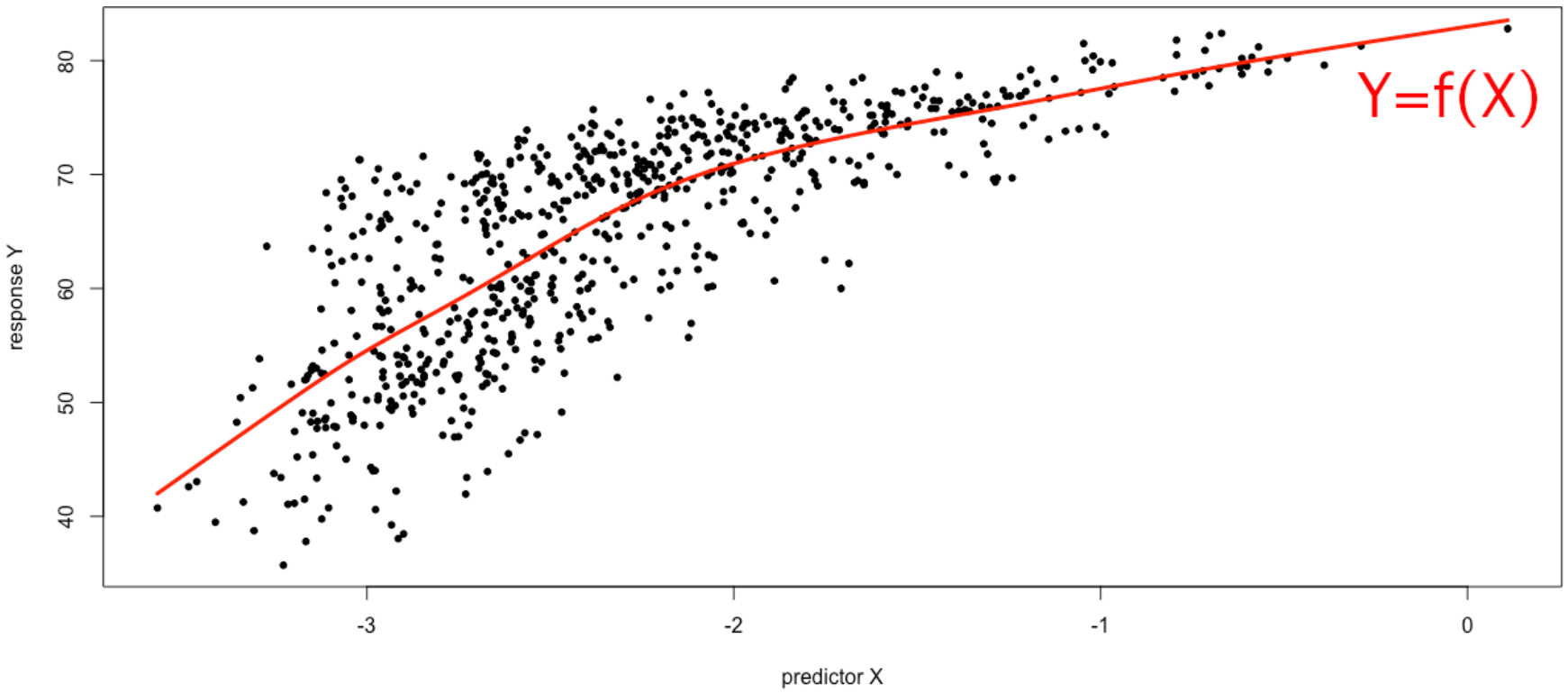
- **predict** the response Y at a **new point** $\vec{X} = \mathbf{x}$;
- **understand** which features of $\vec{X}$ **explain the variation** in Y;
- attempt to **understand** the **effect of each feature** X_j on Y.

We start with a **one-predictor** model with systematic component f:

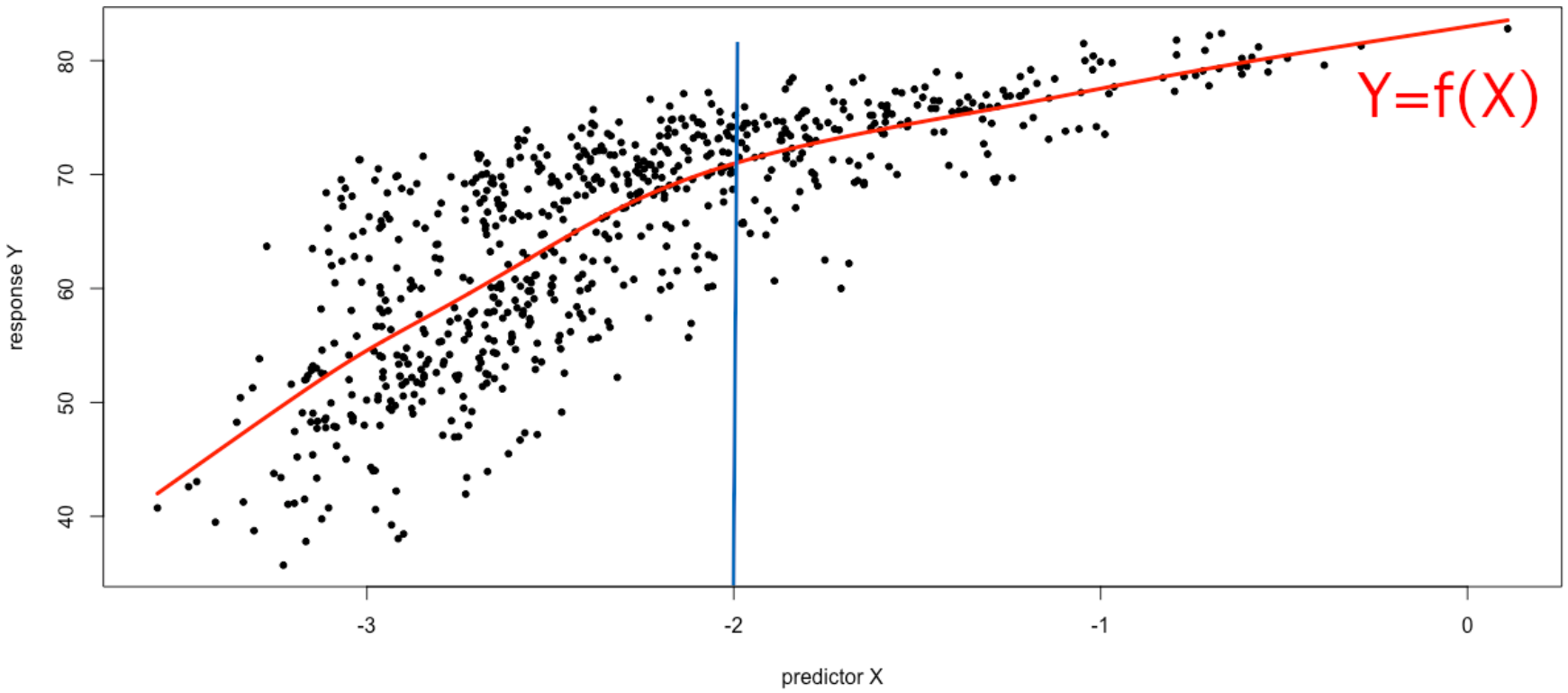
$$Y = f(X) + \varepsilon.$$



What is the ideal f in this case? How can we find it?



A regression model for a subset of the Gapminder data. What would be a good value of $f(-2)$, say?



A regression model for a subset of the Gapminder data, with vertical line at $X = -2$.

Ideally, we would like to have $f(-2) = \mathrm{E}[Y \mid X = -2]$.

For any x in the range of X,

$$f(x) = \mathrm{E}[Y \mid X = x]$$

is the **regression function of Y on X**. In the general setting with p predictors, the regression function is

$$f(\mathbf{x}) = f(x_1, \ldots, x_p) = \mathrm{E}[Y \mid X_1 = x_1, \ldots, X_p = x_p] = \mathrm{E}[Y \mid \vec{X} = \mathbf{x}].$$

It is **optimal** in the sense that it **minimizes** the average square deviation from the response variable, that is to say,

$$f = \arg\min_g \left\{ \mathrm{E}\left[(Y - g(\vec{X}))^2 \mid \vec{X} = \mathbf{x} \right] \right\}.$$

The term $\varepsilon = \varepsilon_{\vec{X}} = Y - f(\vec{X})$ is the **irreducible error** of the regression.

Typically, $\varepsilon_{\vec{X}} \neq 0$ for all $\vec{X}$; even when f is known **exactly**, there is **uncertainty** due to **noise-generating mechanisms** in the "real world".

Let $\hat{f}$ be an **approximation** of f: then $\hat{f}(\vec{X}) = \hat{Y} \approx Y = f(\vec{X}) + \varepsilon$, and

$$\begin{aligned} \mathrm{E}\left[(Y - \hat{Y})^2 \mid \vec{X} = \mathbf{x}\right] &= \mathrm{E}\left[(f(\vec{X}) + \varepsilon - \hat{f}(\vec{X}))^2 \mid \vec{X} = \mathbf{x}\right] \\ &= \underbrace{\left[f(\mathbf{x}) - \hat{f}(\mathbf{x})\right]^2}_{\text{reducible}} + \underbrace{\mathrm{Var}(\varepsilon)}_{\text{irreducible}}. \end{aligned}$$

But $\mathrm{Var}(\varepsilon)$ is not a property of $\hat{f}$; **minimizing** $\mathrm{E}\left[(Y - \hat{Y})^2\right]$ is achieved by working through the **reducible component**.

Learning a model means using the **training data** to find an estimate $\hat{f}$ of f that **minimizes this reducible component**, in some way.

Estimating the Regression Function

In **theory**, we know that the regression function is

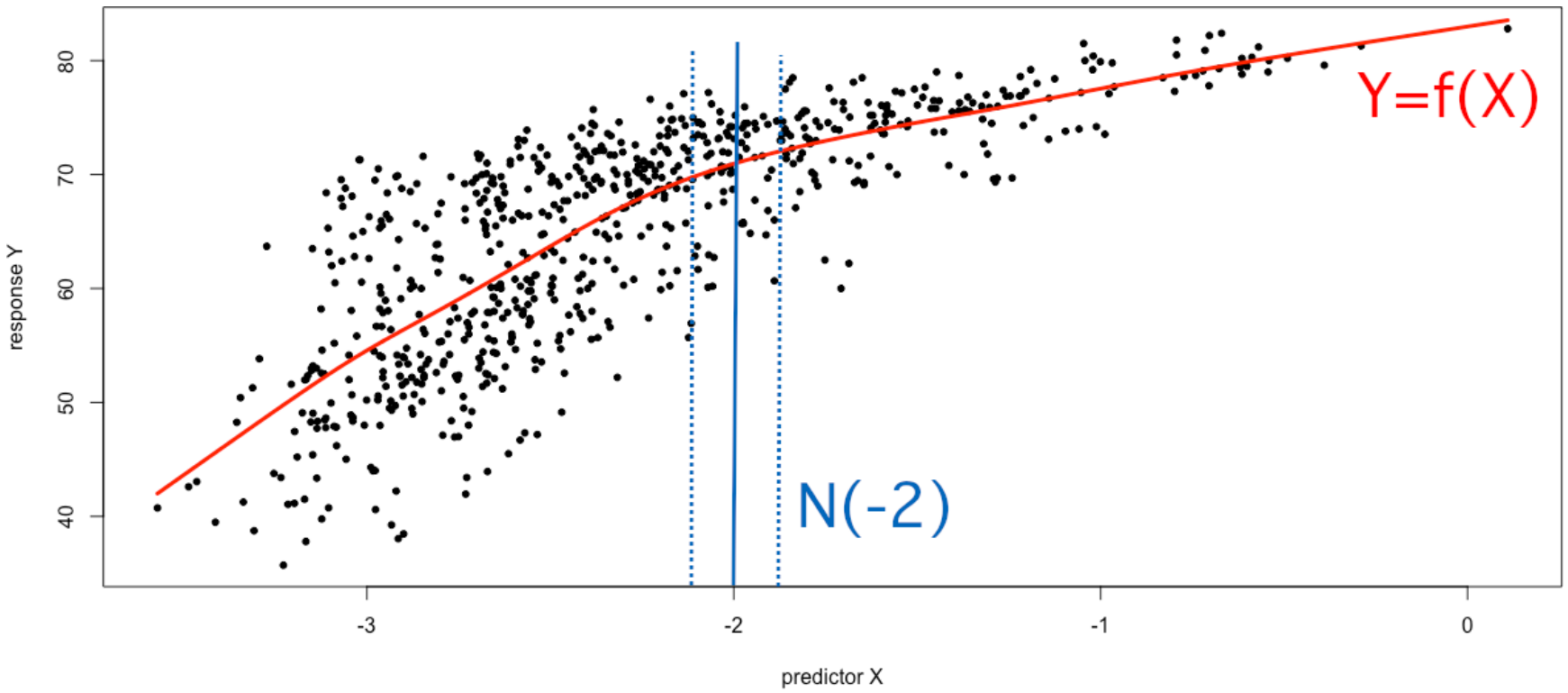
$$f(\mathbf{x}) = \mathrm{E}[Y \mid \vec{X} = \mathbf{x}];$$

in **practice**, however, there might be too few observations at $\vec{X} = \mathbf{x}$ to trust the estimate provided by the **sample mean**.

One solution: approximate $f(\mathbf{x})$ with a **nearest neighbour** (NN) **average**

$$\hat{f}(\mathbf{x}) = \mathrm{Avg}\{Y \mid \vec{X} \in N(\mathbf{x})\},$$

where $N(\mathbf{x})$ is a **neighbourhood** of $\mathbf{x}$.



A regression model for a subset of the Gapminder data, with vertical line at $X = -2$ and neighbourhood $N(-2)$.

This works reasonably well when p is "**small**" ($p \leq 4$?) and N is "**large**".

It fails when p is too **large** $\implies$ **curse of dimensionality**: in **high-p** spaces, NNs are **far apart** and we leave the **"local" neighbourhood** of $\mathbf{x}$ to build $N(\mathbf{x})$, which compromises the **quality** of $\hat{f}(\mathbf{x}) \approx f(\mathbf{x})$ (Chapter 5).

Statistical learning methods estimate $\hat{f}$ of f by minimizing the **reducible component** through either **parametric** or **non-parametric** approaches:

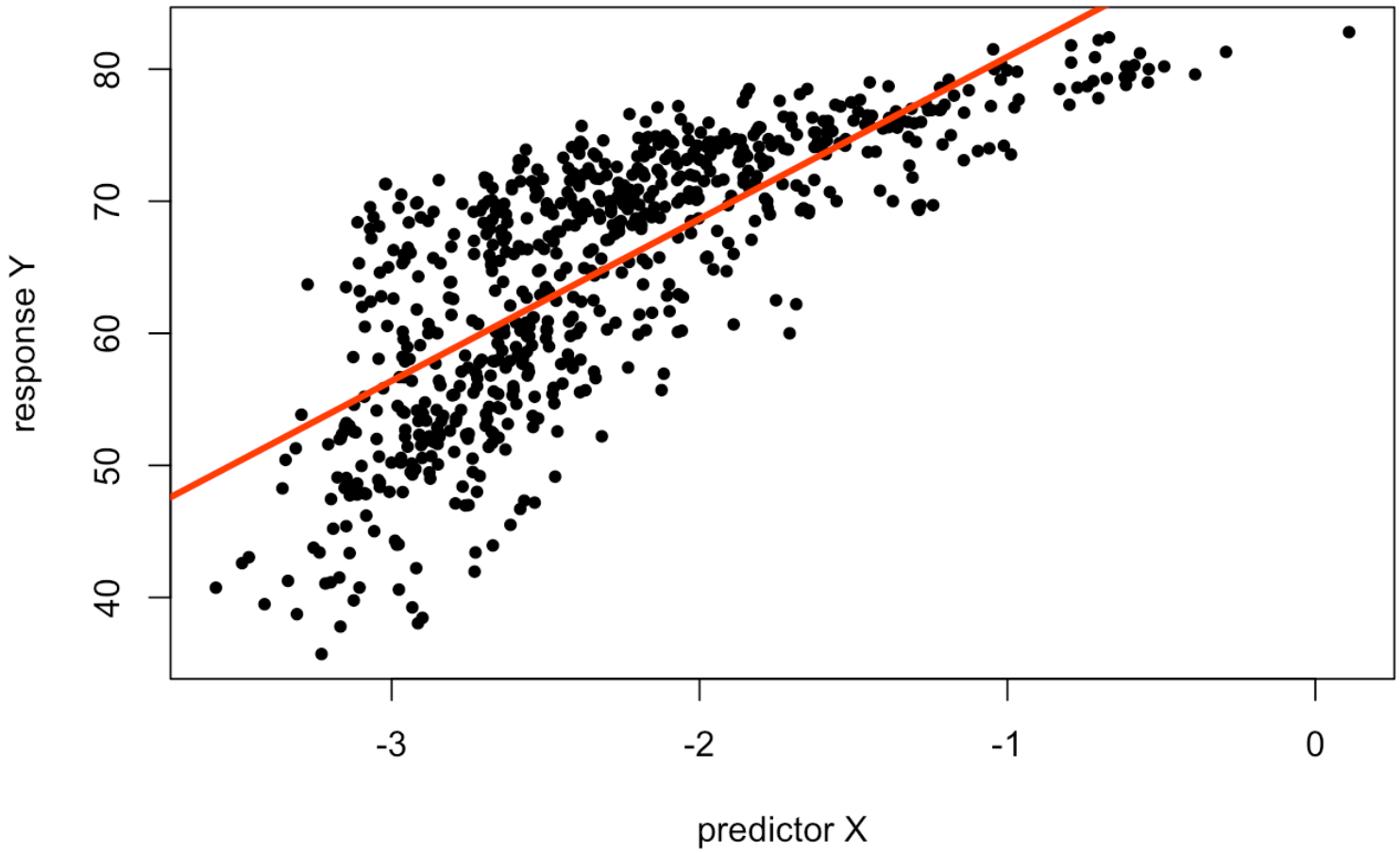
- **parametric** – assumptions are made about the form of f;
- **non-parametric** – no such assumptions are made.

Linear regression is **parametric**; it assumes that f is **linear** and $\approx$

$$\hat{f}(\mathbf{x}) = f_L(\mathbf{x}) = \beta_0 + \beta_1 x_1 + \cdots + \beta_{p-1} x_{p-1}.$$

Objective: learn the $p+1$ parameters $\beta_0, \beta_1, \ldots, \beta_{p-1}$ with **training data**.

This **almost never holds**, but it gives an **interpretable** approximation of f.



Example: if the **true fit** of the motivating example was

$$\text{life expectancy} = f(\text{fertility}, \text{infant mortality}, \text{gdp}) + \varepsilon,$$

then the linear regression model is

$$\begin{aligned} f(\text{fertility},&\text{infant mortality}, \text{gdp}) \approx f_L(\text{fertility}, \text{infant mortality}, \text{gdp}) \\ &= \beta_0 + \beta_1 \cdot \textbf{fertility} + \beta_2 \cdot \textbf{infant mortality} + \beta_3 \cdot \textbf{gdp}. \end{aligned}$$

Advantages: f_L is **interpretable** and it is easier to learn $p+1$ **parameters** than a **whole function** f.

Limitations: the linear model **does not usually match** the true f; when $f_L \not\approx f$, predictions **suffer**.

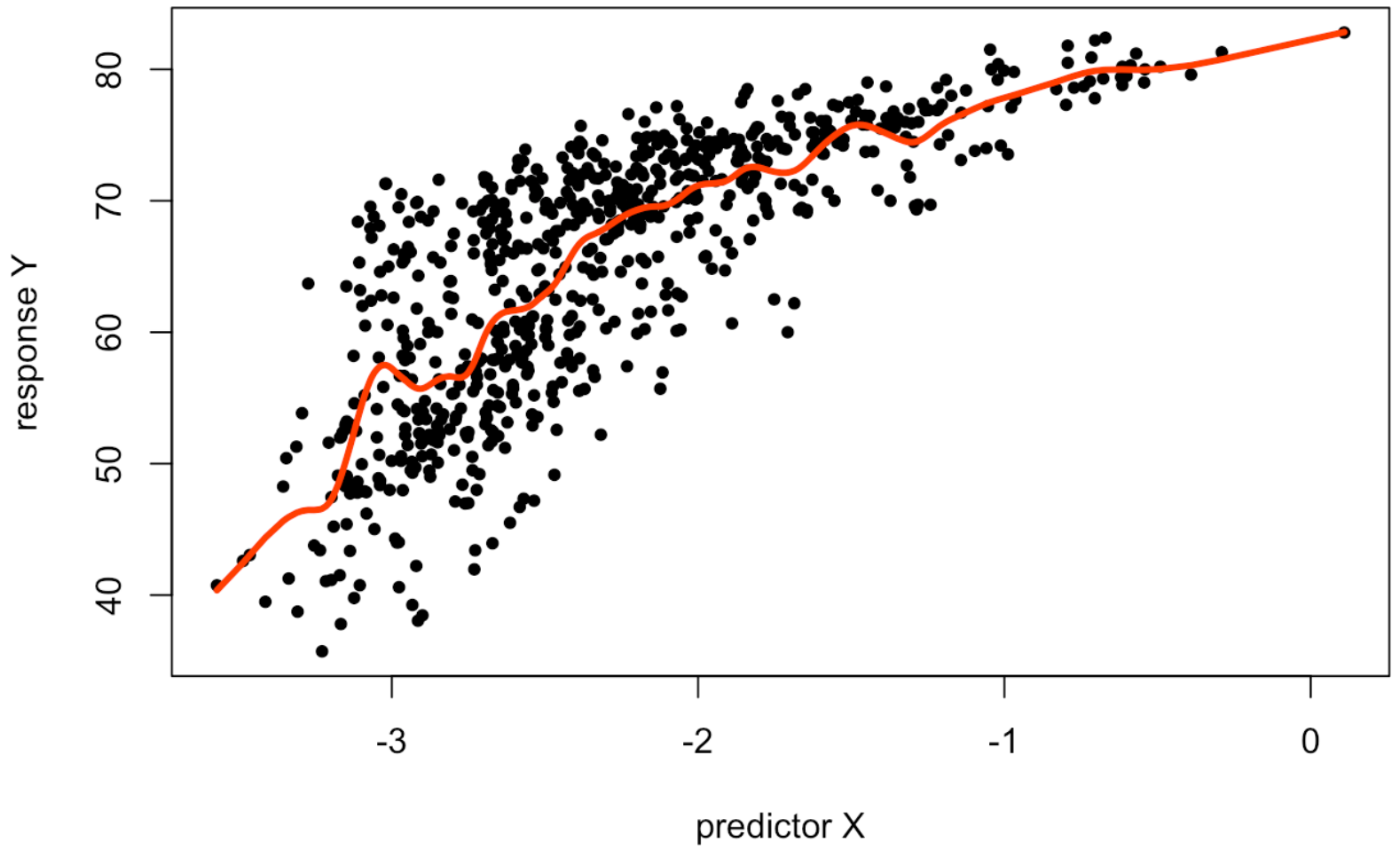
Alternative: why not consider **more complex** functions in order to get **better estimates**?

Short answer: this comes at a **cost** – the resulting functions are usually **more difficult to learn** and they tend to **overfit the data** $\Longrightarrow$ they mis-interpret **data noise** for **data signal**.

Splines are examples of **non-parametric** models: no assumption is made about the **form** of f.

They simply **approximate** f by getting close to the data points without being too **rough** or too **wiggly**.

The example on the next slide shows a typical **spline**.


response Y
40
50
60
70
80
-3
-2
-1
0
predictor X

Advantage: **non-parametric** approaches have the potential to fit a **wider range** of regression shapes.

Limitation: in such contexts, estimating f is not just learning a **small number** of parameters $\Longrightarrow$ more data is required for **accurate estimates**; these models are susceptible to **overfitting.**

Non-parametric methods are usually more **flexible** $\Longrightarrow$ they can produce a **large range** of shapes when estimating the true f $\Longrightarrow$ **more** datasets can be analyzed meaningfully.

Parametric methods are usually more **interpretable** $\Longrightarrow$ **small set** of parameters to learn; easier to make sense of them $\Longrightarrow$ better understanding of how the predictors **interact** to produce outputs.

- **high flexibility**, **low interpretability**: **ensemble learning**, **support vector machines**, **neural networks**, **splines**;
- **low flexibility**, **high interpretability**: **LASSO**, **ridge regression**, **OLS**
- **medium flexibility**, **medium interpretability**: GAMs, regression trees.
- **high-flexibility**, **high-interpretability**: $\varnothing$!!!

The **trade-off** between two **competing desirable** properties is the calling card of ML $\Longrightarrow$ look for them **throughout the course**.

2.1.3 – Model Evaluation

Ideally: we might want to identify the modeling approach that performs "**best**", and use it **for all problems**.

But: the discussion on **trade-offs** shows that the concept of "**best performance**" cannot be defined **in a way that meets all desired requirements**, so ... ?

No Free-Lunch Theorem: **no single method is optimal over all possible datasets**.

Given a specific task and dataset, how do we select the approach that will yield the "**best**" results?

This is the **main ML challenge**.

To evaluate a model's performance, we must be able to measure how well **predictions** match the **observed data**. In a **regression setting**, we use:

- **correlation** $\rho_{\hat{y},y}$;
- **mean squared error** (MSE): $\text{Avg}\{(y_i - \hat{f}(\mathbf{x}_i))^2\}$;
- **mean absolute error** (MAE): $\text{Avg}\{|y_i - \hat{f}(\mathbf{x}_i)|\}$;
- **normalized mean squared error** (NMSE): $\text{MSE} \,/\, \text{Avg}\{(y_i - \overline{y})^2\}$;
- **normalized mean absolute error** (NMAE): $\text{MAE} \,/\, \text{Avg}\{|y_i - \overline{y}|\}$;
- **mean average percentage error** (MAPE): $\text{Avg}\left\{\dfrac{|y_i - \hat{f}(\mathbf{x}_i)|}{y_i}\right\}$, $y_i \neq 0$,
- etc.

MSE has **convenient** mathematical properties, so we use it as our **go-to metric**, but the conceptual notions we will discuss next would be **qualitatively similar** for all performance evaluation tools.

IMPORTANT: we determine the suitability of a model for **predictive** purposes by evaluating the selected metric(s) on **testing** (or **unseen**) **data** (or unseen data), NOT on the **training** data.

Without **test data**, the model can **at best** describe the **training data**.

Example: if we are trying to determine whether any clinical measurement in patients are likely to predict the onset of Alzheimer's disease, we do not particularly care if the algorithm does a good job of telling us that the patients we have already tested for the disease have it or not – **it is new patients that are of interest**.

It would be surprising if the performance on the **test data** is any good when the performance on the **training** data is middling (see **overfitting**).

Let $\mathrm{Tr} = \{(\mathbf{x}_i, y_i) \mid i = 1, \ldots, N\}$ be the **training set** and suppose that **we fit** $\hat{f}$ **over** Tr; i.e., we use some ML method to estimate

$$Y = f(\vec{X}) + \varepsilon \quad \text{by} \quad \hat{Y} = \hat{f}(\vec{X}).$$

Hopefully, we have $\hat{f}(\mathbf{x}_i) \approx y_i$ for all $i = 1, \ldots, N$, and

$$\mathrm{MSE}_{\mathrm{Tr}} = \frac{1}{N} \sum_{i=1}^{N} (y_i - \hat{f}(\mathbf{x}_i))^2 \quad \text{is } \textbf{small}.$$

If $\mathrm{MSE}_{\mathrm{Tr}}$ is small, then the model does a good job of **describing** Tr. But this is largely **irrelevant** to our ability to make **good predictions**: does

$$\hat{f}(\mathbf{x}^*) \approx f(\mathbf{x}^*) = y^*, \quad \text{for observations } (\mathbf{x}^*, y^*) \notin \mathrm{Tr}?$$

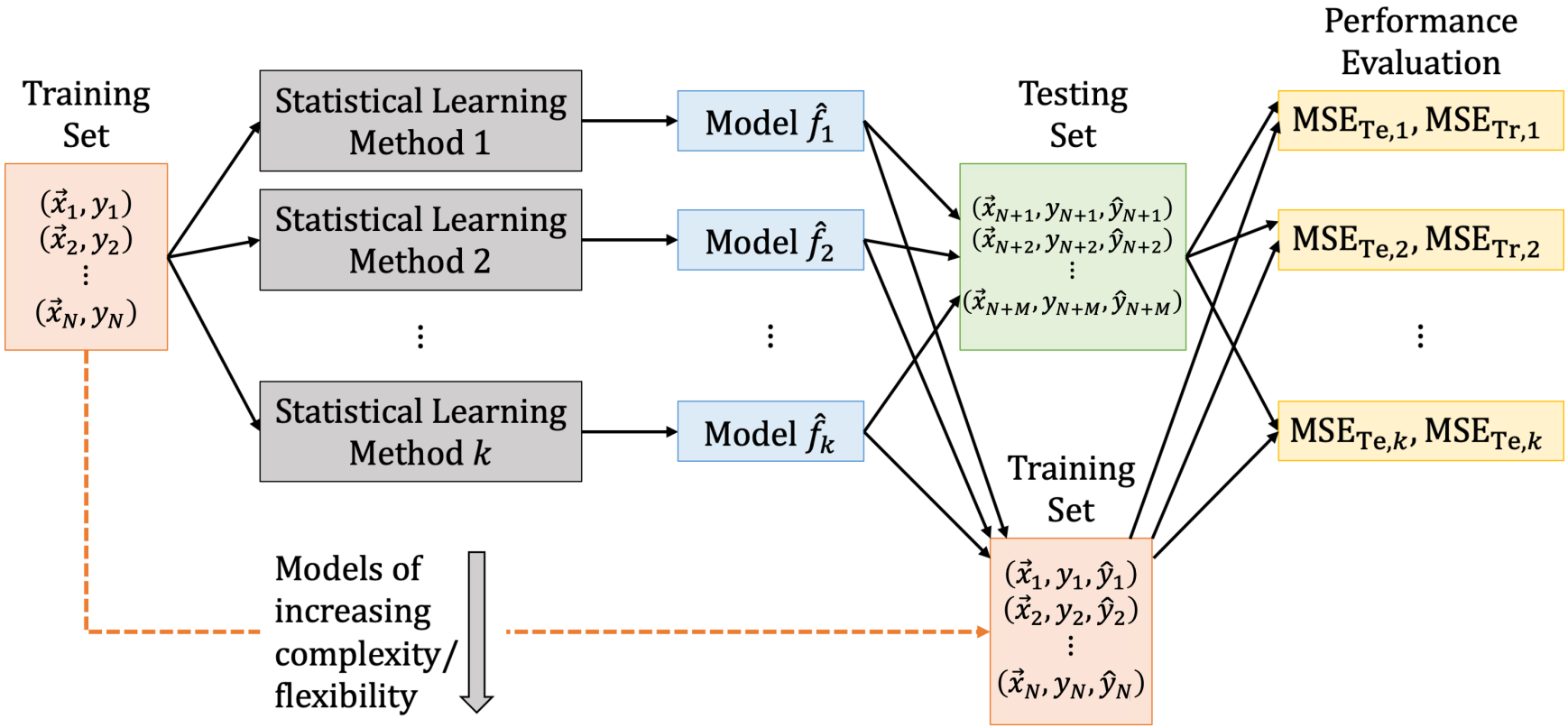
An **optimal** ML method for a **combination** of **task** and **dataset** **minimizes**

$$\mathrm{MSE}_{\mathrm{Te}} = \frac{1}{M} \sum_{j=N+1}^{N+M=n} (y_j - \hat{f}(\mathbf{x}_j))^2$$

over the **test set** $\mathrm{Te} = \{(\mathbf{x}_j, y_j) \mid j = N+1, \ldots, N+M = n\}$, where, *a priori*, none of the test observations were in Tr (at least, their **indices** were not).

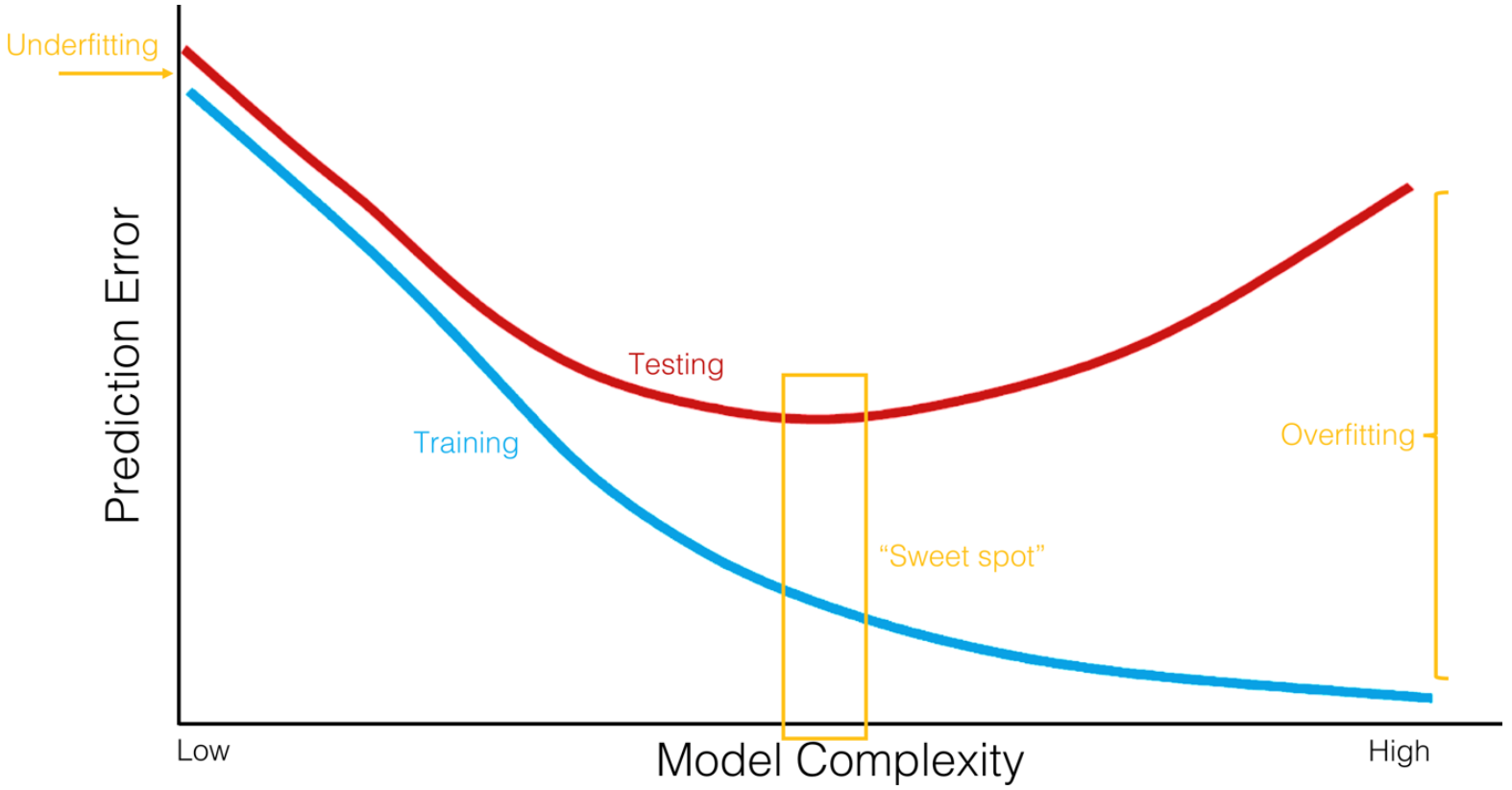
However, observations in Te may end up being identical to some of the training observations, due to the reality of the scenario under consideration.

The general situation is illustrated on the next slide: the **training/testing paradigm**. Tr is fed into a variety of ML methods, possibly arranged in **increasing order of complexity**, yielding a sequence of models.



These models are then used to make predictions $\hat{y}$ on Te (using only the **predictors**); the $\hat{y}$ are then compared with the y to evaluate the model's performance on Te (this can also be done for Tr).

2.1.4 – Bias-Variance Trade-Off



Generic illustration of the bias-variance trade-off

When **model complexity** $\nearrow$, $\mathrm{MSE}_{\mathrm{Tr}}$ $\searrow$; but $\mathrm{MSE}_{\mathrm{Te}}$ first $\searrow$, then $\nearrow$.

Models that are **too simple** have "**large**" $\mathrm{MSE}_{\mathrm{Tr}}$ and MSE_{Te} (**underfitting**); models that are **too complex** have "**small**" $\mathrm{MSE}_{\mathrm{Tr}}$ and "**large**" $\mathrm{MSE}_{\mathrm{Te}}$ (**overfitting**).

The "**U**" shape of $\mathrm{MSE}_{\mathrm{Te}}$ is **generic** – something of this nature occurs for nearly **all** datasets and choice of SL family of methods (for regression and for classification): **underfitting** and **overfitting** are facts of ML life.

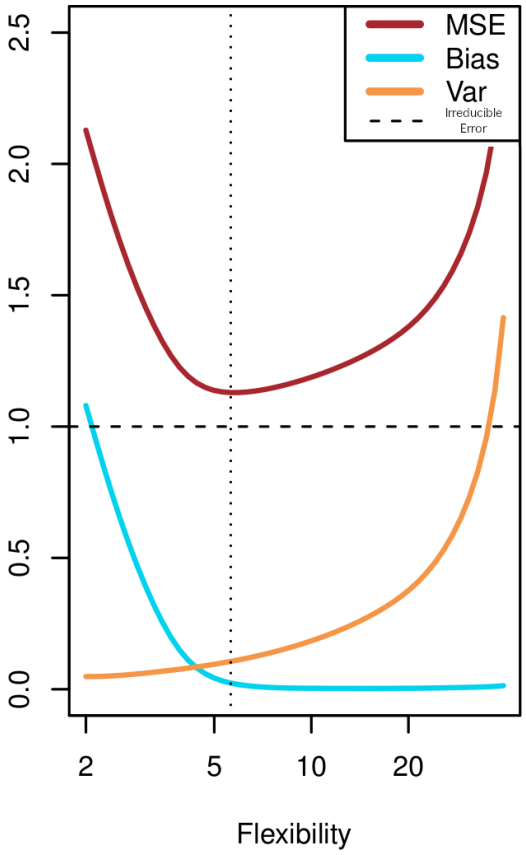
Consider a **test** observation $(\mathbf{x}^*, y^*)$ and a **fitted model** $\hat{f}$ trained on Tr approximating the true model

$$Y = f(\vec{X}) + \varepsilon, \quad \text{where } f(\mathbf{x}) = \mathrm{E}[Y \mid \vec{X} = \mathbf{x}].$$

The **expected test** MSE **at** $\mathbf{x}^*$ can be decomposed into 3 fundamental quantities

$$\begin{aligned}\mathrm{E}\left[\mathrm{MSE}_{\mathrm{Te}}(\mathbf{x}^*)\right] &= \mathrm{E}\left[(y^* - \hat{f}(\mathbf{x}^*))^2\right] \\ &= \underbrace{\mathrm{Var}(\hat{f}(\mathbf{x}^*))}_{\text{variance}} + \underbrace{\left\{\mathrm{E}\left[\hat{f}(\mathbf{x}^*) - f(\mathbf{x}^*)\right]\right\}^2}_{\text{squared bias}} + \mathrm{Var}(\varepsilon).\end{aligned}$$

- $\mathrm{Var}(\varepsilon)$ is the **irreducible error** (due to the **inherent noise** in the data);
- $\mathrm{Var}(\hat{f}(\mathbf{x}^*))$ is the **variance component error**, which arises since **different** Tr would yield **different** fitted models $\hat{f}$, and
- the other term is the **(squared) bias component error**, which arises due to the "**difficult**" problem being approximated by a "**simple**" model.



Expected test error decomposition

The **overall expected test** MSE is the **average** of $\mathrm{E}[\mathrm{MSE}_{\mathrm{Te}}(\mathbf{x}^*)]$ over all allowable $\mathbf{x}^*$ in the testing space.

Note that, by construction:

$$\mathrm{E}[\mathrm{MSE}_{\mathrm{Te}}] \geq \mathrm{Var}(\varepsilon).$$

Flexible/complex methods have $\uparrow$ variance and $\downarrow$ bias; **rigid/simpler** methods have $\uparrow$ bias and $\downarrow$ variance.

This interplay causes models to **underfit** (**high bias**) or **overfit** (**high variance**) the data (**bias-variance trade-off**).

Take-Aways:

- the optimal regression function $Y = f(\vec{X}) + \varepsilon$ for **numerical responses** is

$$f(x) = \mathrm{E}[Y \mid \vec{X} = \mathbf{x}];$$

- models are **learned** on training data Tr;
- in practice, we learn the best model from a **restricted** group of model families;
- the best model $\hat{f}(\mathbf{x})$ **minimizes** the reducible part of the prediction error $\mathrm{MSE}_{\mathrm{Te}}$, **evaluated** on testing data Te;
- the **bias-variance trade-off** tells us that models that are too simple/rigid **underfit** the data, and that models that are too complex/flexible **overfit** the data;
- the total prediction error on Te is **bounded below** by the irreducible error $\mathrm{Var}(\varepsilon)$;
- a **predictive** model's performance can only be evaluated on **unseen data** (i.e., not drawn from Tr); if this requirement is not met, the model is at best **descriptive**.

2.2 – Regression Modeling

In the regression setting, the "**best**" estimate of $f(\mathbf{x}) = \mathrm{E}[Y \mid \vec{X} = \mathbf{x}]$ is the model $\hat{f}$ that **minimizes**

$$\mathrm{MSE}_{\mathrm{Te}}(\hat{f}) = \underset{\mathbf{x}^* \in \mathrm{Te}}{\mathrm{Avg}} \left\{ \mathrm{E}\left[(y^* - \hat{f}(\mathbf{x}^*))^2 \right] \right\}.$$

In practice: we try to **learn** the best $\hat{f}$ from **restricted families of models**. But no matter what $\hat{f}$ is,

$$\mathrm{MSE}_{\mathrm{Te}}(\hat{f}) \geq \mathrm{Var}(\varepsilon).$$

In the **ordinary least square** framework (OLS), we assume that f is **nearly globally linear**:

$$f(\mathbf{x}) \approx \hat{f}_{\mathsf{OLS}}(\mathbf{x}) = \mathbf{x}^\top \boldsymbol{\beta}.$$

In reality, the true f is **almost never** linear, but the linear assumption yields models $\hat{f}$ that are **conceptually** and **practically** useful:

- $\hat{f}$ is easily **interpretable**, and
- the associated prediction error $\mathrm{MSE}_{\mathrm{Te}}(\hat{f})$ is often acceptably "**small**".

By some estimation, up to **90%** of real-world data applications end up using OLS (or LR) as their final model, (at least, not that long ago), as:

- OLS/LR models are straightforward to **train** (and **interpret**);
- $\mathrm{MSE}_{\mathrm{Te}}$ has a **closed-form linear expression** (OLS), and
- OLS solution can be computed using **sing simple matrix manipulations**.

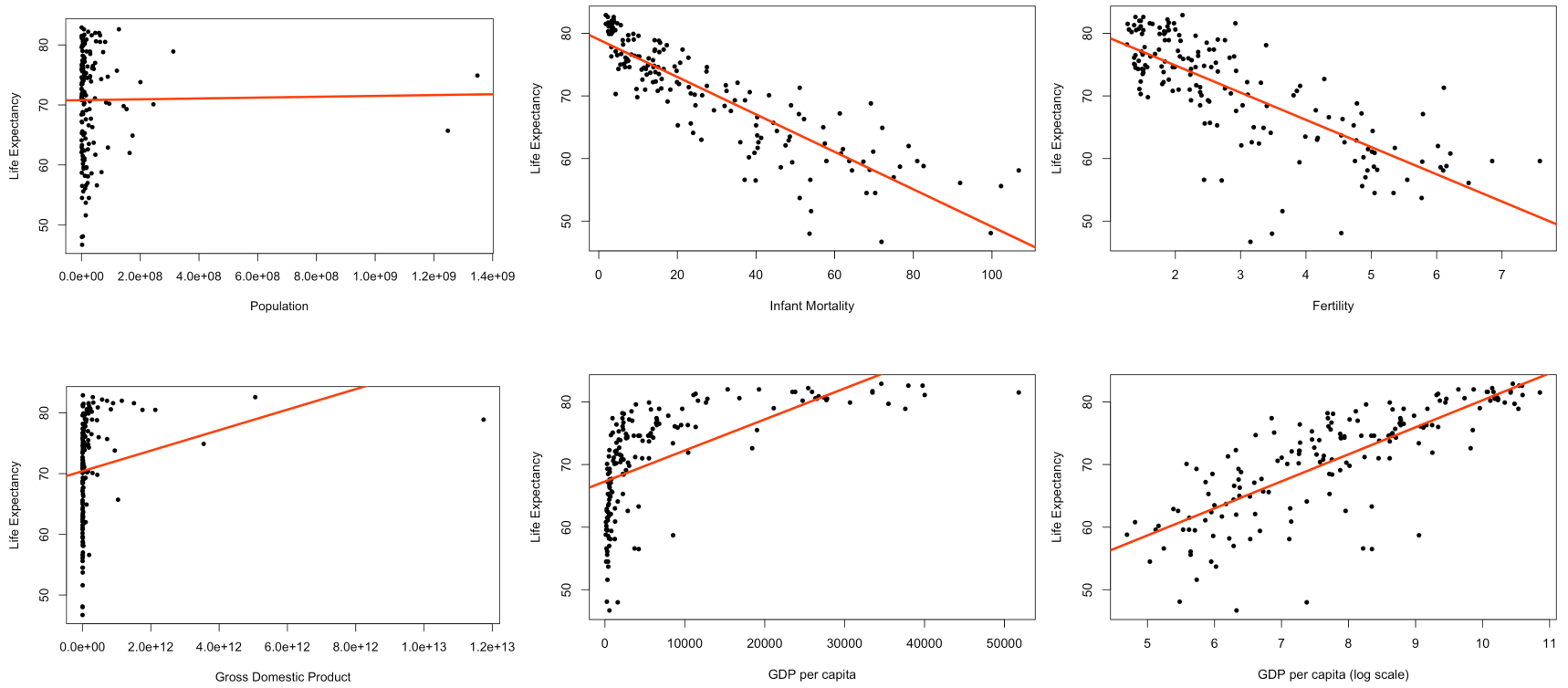
Requirements: the dataset must be very carefully prepared (**cleaning**, **encoding**, **imputation of missing values**, **creation of new variables**, **transformation of variables**, etc.), perhaps more so than with other models.

Gapminder Example

Let us revisit the `Gapminder` dataset, focusing on observations from 2011.

- Is there a relationship between **gross domestic product** and **life expectancy**?
- How **strong** is the relationship?
- Is the relationship **linear**?
- Which factors contribute to the **life expectancy**?
- How accurately could we predict **life expectancy** given a set of new observations?
- Are there combinations of factors that are linked with **life expectancy**?

Can the scatterplots of various predictors against **life expectancy** for the 2011 Gapminder data be used to answer these questions?



(DUDADS, 20.2, *Regression Modeling*)

2.2.1 – Formalism

Consider a dataset $\mathrm{Tr} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$ with N **observations** and $p-1$ **features**.

The corresponding **design matrix**, **response vector**, and **coefficient vector** are, respectively,

$$\mathbf{X} = \begin{pmatrix} 1 & x_{1,1} & \cdots & x_{1,p-1} \\ \vdots & \vdots & & \vdots \\ 1 & x_{N,1} & \cdots & x_{N,p-1} \end{pmatrix}, \quad \mathbf{Y} = \begin{pmatrix} y_1 \\ \vdots \\ y_N \end{pmatrix}, \quad \boldsymbol{\beta} = \begin{pmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_{p-1} \end{pmatrix}.$$

Objective: find f s.t. $\mathbf{Y} = f(\mathbf{X}) + \boldsymbol{\varepsilon}$. The OLS solution assumes that $\hat{\mathbf{Y}} = f(\mathbf{X}) = \mathbf{X}\boldsymbol{\beta}$; we must thus **learn** $\boldsymbol{\beta}$ using the **training data** Tr.

If $\hat{\boldsymbol{\beta}}$ is an estimate of the true $\boldsymbol{\beta}$, the **linear** model associated with Tr is

$$\hat{f}(\mathbf{x}) = \hat{\beta}_0 + \hat{\beta}_1 x_1 + \cdots + \hat{\beta}_{p-1} x_{p-1}.$$

How do we find $\hat{\boldsymbol{\beta}}$? The OLS estimate $\hat{\boldsymbol{\beta}}_{\mathbf{OLS}}$ minimizes the **loss function**

$$\mathcal{L}(\boldsymbol{\beta}) = \|\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}\|_2^2 = (\mathbf{Y} - \mathbf{X}\boldsymbol{\beta})^\top (\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}).$$

Assuming data **regularity**, $\mathcal{L}$ is **non-degenerate, non-negative, quadratic** in $\boldsymbol{\beta}$; with no constraints on $\boldsymbol{\beta}$, minimizers must also be **critical points**.

We seek a minimizer satisfying the **canonical** (normal) **equations**:

$$\nabla \mathcal{L}(\boldsymbol{\beta}) = -2(\mathbf{X}^\top \mathbf{Y} - \mathbf{X}^\top \mathbf{X}\boldsymbol{\beta}) = \mathbf{0} \implies \mathbf{X}^\top \mathbf{X}\hat{\boldsymbol{\beta}} = \mathbf{X}^\top \mathbf{Y}.$$

If $\mathbf{X}^\top\mathbf{X}$ is **invertible**, we have learned the **unique** $\hat{\boldsymbol{\beta}}$ on Tr using OLS:

$$\hat{\boldsymbol{\beta}}_{\mathbf{OLS}} = (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{Y}, \quad \text{with} \quad \mathrm{Var}(\hat{\boldsymbol{\beta}}_{\mathbf{OLS}}) = \hat{\sigma}^2(\mathbf{X}^\top\mathbf{X})^{-1},$$

where $\hat{\sigma}^2$ is the variance of the **residuals** $\mathbf{e} = \mathbf{Y} - \hat{\mathbf{Y}}$.

Note that $\mathbf{X}^\top\mathbf{X}$ is $p \times p$; $p \ll N \implies$ easy to invert, even when N is **large**.

If we write $\mathbf{x} = (1, x_1, \ldots, x_{p-1})^\top$, the **fitted value** of $\hat{f}$ at **input** $\mathbf{x}_i \in \mathrm{Tr}$ and the **predicted value** at **an arbitrary** $\mathbf{x}^* \in \mathbb{R}^p$ are thus

$$\hat{y}_i = \hat{f}_{\mathbf{OLS}}(\mathbf{x}_i) = \mathbf{x}_i^\top\hat{\boldsymbol{\beta}}_{\mathbf{OLS}} \quad \text{and} \quad \hat{y}^* = \hat{f}_{\mathbf{OLS}}(\mathbf{x}^*) = \mathbf{x}^{*\top}\hat{\boldsymbol{\beta}}_{\mathbf{OLS}}.$$

Effectively, the **fitted surface** is completely described by the p parameters $\hat{\boldsymbol{\beta}}_{\mathbf{OLS}}$; this number is a measure of the **learner's complexity**.

Motivating Example

We study a subset of the Gapminder dataset: the observations for 2011, the predictor variables **infant mortality** X_1 and **fertility** X_2, and the response **life expectancy** Y (DUDADS, 20.2, *Regression Modeling*, Formalism).

Tr contains $N = 166$ observations and $p = 2$ predictor features.

```
'data.frame':   166 obs. of  3 variables:
 $ const           : num  1 1 1 1 1 1 1 1 1 1 1 1 ...
 $ infant_mortality: num  14.3 22.8 106.8 7.2 12.7 ...
 $ fertility       : num  1.75 2.83 6.1 2.12 2.2 ...
```

The design matrix $\mathbf{X}$ is 166×3; the response is 166×1.

The constituents of the **normal equations** are:

$$\mathbf{X}^\top\mathbf{X} = \begin{pmatrix} 166.0 & 4537.3 & 486.54 \\ 4537.3 & 225043.25 & 18445.28 \\ 486.54 & 18445.28 & 1790.238 \end{pmatrix}, \quad \mathbf{X}^\top\mathbf{Y} = \begin{pmatrix} 11756.7 \\ 291153.33 \\ 32874.95 \end{pmatrix}.$$

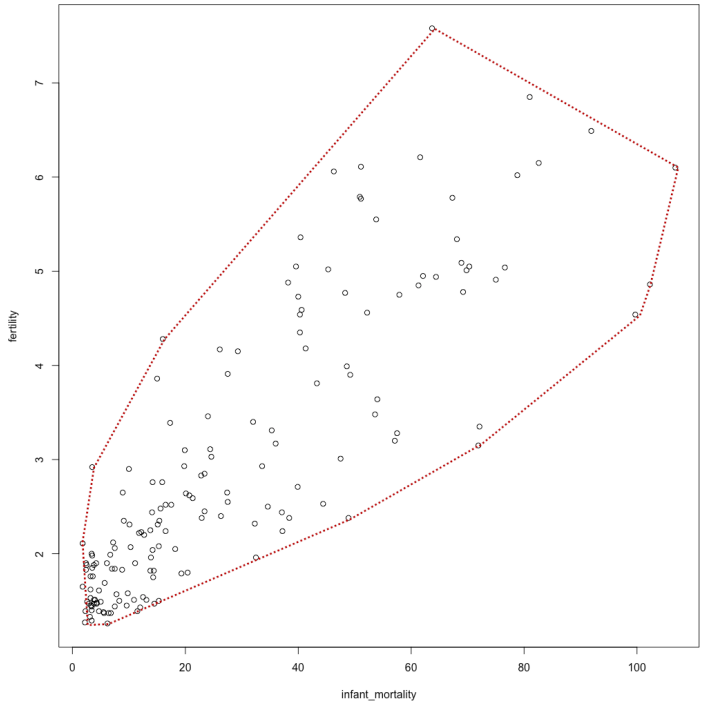
Thus,

$$\hat{\boldsymbol{\beta}}_{\text{OLS}} = (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{Y} = \begin{pmatrix} 79.677 \\ -0.276 \\ -0.443 \end{pmatrix},$$

and the OLS **fitted surface** is:

$$y^* = \hat{f}_{\text{OLS}}(\mathbf{x}^*) = 79.677 - 0.276x_1^* - 0.443x_2^*$$

for an observation $\mathbf{x}^* = (x_1^*, x_2^*)$ – **the $x_0^* = 1$ is sometimes dropped**.



Predictor envelope for the Gapminder subset

Warning: predictions should not be made for observations outside the **range** (or the **envelope**) of the training predictors.

In this example, the **predictor envelope** is shown in red on the left – one should resist the temptation to try to predict y^* at $\mathbf{x}^* = (100, 2)$, say.

Least Squares Assumptions

Since the family of OLS learners is a **subset** of all possible learners, the best we can say about $\hat{f}_{\text{OLS}}$ is that

$$\text{MSE}_{\text{Te}}(\hat{f}_{\text{OLS}}) \geq \min_{\hat{f}} \left\{ \text{MSE}_{\text{Te}}(\hat{f}) \right\} \geq \text{Var}(\varepsilon).$$

In practice, we can approximate f with **any** learner $\hat{f}$. If we want $\hat{f}$ to be **useful**, we need $\hat{f}$ to be (at least) a "**decent**" approximation of f.

Trade-off: when we restrict learners to **specific** families (i.e., we impose **structure** on the learners), we introduce assumptions on the **data**.

It is crucial to understand what these may be to **verify/validate** them.

OLS assumptions

- **linearity:** the response variable is a **linear combination** of the predictors
- **homoscedasticity:** the error variance is **constant for all** predictor levels
- **uncorrelated errors:** the error is **uncorrelated** between observations
- **full rank of design matrix $\mathbf{X}$:** the predictors are not **multicollinear**
- **weak exogeneity:** predictor values are free of **measurement error**

Mathematically, this translates to Tr being **exact** and

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon},\ \mathrm{E}[\boldsymbol{\varepsilon} \mid \mathbf{X}] = \mathbf{0},\ \mathrm{E}[\boldsymbol{\varepsilon}\boldsymbol{\varepsilon}^\top \mid \mathbf{X}] = \sigma^2\mathbf{I}_n \overset{?}{\Longrightarrow} \mathbf{Y} \mid \mathbf{X} \sim \mathcal{N}(\mathbf{X}\boldsymbol{\beta}, \sigma^2\mathbf{I}_n).$$

How can we determine if the choice of model is **valid**? In the ML context, there is only one true test:

does the model make "good" predictions?

2.2.2 – Least Squares Properties

Assume OLS holds. What can we say about the **linear regression** results? (DUDADS, 20.2, *Regression Modeling*, Least Squares Properties)

```
Residuals: Min       1Q   Median       3Q      Max
      -15.3233  -2.0057   0.2003   2.9570  10.6370

Coefficients:
                Estimate Std. Error t value Pr(>|t|)
(Intercept)      79.6759     0.7985  99.786   <2e-16 ***
infant_mortality -0.2763     0.0248 -11.138   <2e-16 ***
fertility        -0.4440     0.4131  -1.075    0.284

Residual standard error: 4.172 on 163 degrees of freedom
Multiple R-squared:  0.7612,    Adjusted R-squared:  0.7583
F-statistic: 259.8 on 2 and 163 DF,  p-value: < 2.2e-16
```

Coefficient of Determination

Write $\text{SSE} = \mathbf{Y}^\top[\mathbf{I}_n - \mathbf{X}(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top]\mathbf{Y}$ and $\text{SST} = \mathbf{Y}^\top\mathbf{Y} - n\bar{y}^2$.

The **coefficient of determination** of the OLS regression is

$$R^2 = 1 - \frac{\text{SSE}}{\text{SST}} = \frac{\text{Cov}^2(\mathbf{Y}, \mathbf{X}\hat{\boldsymbol{\beta}})}{\sigma_y^2\sigma_{\hat{y}}^2} \quad \text{(if } \beta_0 \neq 0\text{)}.$$

The **coefficient of determination** identifies the **proportion of the variation in the response** is **explained** by $\hat{f}_{\text{OLS}}$, so $0 \leq R^2 \leq 1$.

If $R^2 \approx 0$, then the predictor $X_1, \ldots, X_p$ have **little explanatory power** on the response Y; if $R^2 \approx 1$, then the linear fit is "**good**", as **much** of the variation in the response is **explained** by the predictors.

The value of p also affects the goodness-of-fit (cf. **curse of dimensionality**):

$$R_a^2 = 1 - \frac{N-1}{N-p}(1-R^2) = 1 - \frac{\mathrm{SSE}/(N-p)}{\mathrm{SST}/(N-1)}$$

is the **adjusted coefficient of determination** of the linear regression. While R_a^2 can be negative, it is always smaller than R^2. It also plays a role in the **feature selection** process.

In the Gapminder example, we have: $\mathrm{SSE} = 2837.69$, $\mathrm{SST} = 11882.18$, $R^2 = 0.7612$, and $R_a^2 = 0.7583$.

This suggests that a **fair** proportion of the variability in the life expectancy (about 75.8%) is explained by infant mortality and fertility.

Significance of Regression

We determine if at **least one** of the predictors $X_1, \ldots, X_{p-1}$ is useful in predicting the response Y by pitting

$$H_0 : (\beta_1, \ldots, \beta_{p-1}) = \mathbf{0} \quad \text{against} \quad H_1 : (\beta_1, \ldots, \beta_{p-1}) \neq \mathbf{0}.$$

Under the null hypothesis H_0, the $F-$statistic

$$F^* = \frac{(\mathrm{SST} - \mathrm{SSE})/p}{\mathrm{SSE}/(N-p)} \sim F_{p,N-p}.$$

If $F^* \geq F_{p,N-p;\alpha}$, then we **reject H_0 in favour of H_1 at significance α**.

In the Gapminder model, we have $F^* = 259.8$ and $F_{2,164;0.05} = 3.051$. Since $F^* \geq F_{2,164;0.05}$, at least one of $\beta_1, \beta_2 \neq 0$, with probability 95%.

Interpretation of the Coefficients

For $j = 1, \ldots, p$, the coefficient β_j is the **average** effect on Y of a $1-$**unit increase** in X_j, **holding all other predictors fixed**.

If the predictors are **uncorrelated**, each coefficient can be **tested** (and **estimated**) separately, and the interpretation is reasonable **(in theory)**.

In practice, we can not always **control** the predictor variables, and it might be impossible to "**hold all other predictors fixed**."

When the predictors are **correlated**, there are potential **variance inflation** issues for the estimated regression coefficients, and the interpretion is **risky**, since when X_j changes, so do **the other predictors**.

The interpretation can also be read as a claim of **causality**, which **should be avoided** when dealing with observational data.

> "The only way to find out what will happen when a complex system is disturbed is to disturb the system, not merely to observe it passively." [G.E.P. Box, "Use and abuse of regression," Journal of Technometrics, vol. 8, no. 4, pp. 625–629, Nov. 1966.]

In the Gapminder example, the correlation between X_1 and X_2 is

$$\rho(X_1, X_2) = -0.8714863.$$

The predictors are thus **strongly correlated**, and the standard interpretation is not available to us.

Hypothesis Testing

We can determine if a specific predictor X_j is useful in **predicting the response** Y, by testing for

$$H_0 : \beta_j = 0 \quad \text{against} \quad H_1 : \beta_j \neq 0.$$

Under the null hypothesis H_0, the test statistic is

$$t^* = \frac{\hat{\beta}_j}{\text{se}(\hat{\beta}_j)} \sim T_{N-2},$$

where $\text{se}(\hat{\beta}_j) = \sqrt{\hat{\sigma}^2 (\mathbf{X}^\top \mathbf{X})^{-1}_{j+1,j+1}}$, and $\hat{\sigma}^2 = \frac{\text{SSE}}{N-p}$, and T_{n-2} is the Student T distribution with $n-2$ degrees of freedom.

At significance α, if $|t^*| \geq |t_{n-2;\alpha/2}|$, then we **reject H_0 in favour of H_1**.

In the Gapminder model, we have $N = 166$, $p = 2$, $\hat{\beta}_1 = -0.276$ so

$$\hat{\sigma}^2 = 17.51661, \quad \text{se}(\hat{\beta}_1) = 0.02488045.$$

Since $t^* = -11.08275$ and $T_{164;0.025} = -1.974535$, then $|t^*| \geq |t_{164;0.025}|$, and $\beta_1 \neq 0$ with 95% probability (in the **frequentist** interpretation).

If we conduct multiple tests **simultaneously**, we must remember that continuously going to the well "exhausts" the dataset's **predictive power**:

- Bonferroni procedure
- Working-Hotelling procedure
- Scheffé procedure

Confidence Intervals

The **standard error** of $\hat{\beta}_j$ reflects how the estimate would vary under **various** Tr; it can be used to compute a $(1-\alpha)\%$ **confidence interval** for β_j:

$$\mathrm{C.I.}(\beta_j; 1-\alpha) \equiv \hat{\beta}_j \pm z_{\alpha/2} \cdot \mathrm{se}(\hat{\beta}_j);$$

at $\alpha = 0.05$, $z_{\alpha/2} = 1.96 \approx 2$, so: $\mathrm{C.I.}(\beta_j; 0.95) \equiv \hat{\beta}_j \pm 2\,\mathrm{se}(\hat{\beta}_j)$.

coeff.	est.	s.e.	t*	95% CI
β_0	79.677	0.7985	99.786	$[78.1, 81.3,]$
β_1	-0.276	0.0248	-11.138	$[-0.33, -0.23]$
β_2	0.443	0.4131	-1.075	$[-1.27, 0.38]$

In **frequentist** statistics, the 95% CI has a **special** interpretation – **not** that there is a 95% chance that β_j falls in **that** CI; rather, that the approach used to build the 95% CI with different Tr yields an interval in which β_j falls approximately 95% of the time.

The resulting CI also depend on the **underlying model**. The 95% CI for β_1 in the **full** Gapminder model is $[-0.33, -0.23]$, but the corresponding $\mathrm{C.I.}$ in the **reduced** Gapminder model

$$\hat{Y} = \gamma_0 + \gamma_1 X_1$$

is $[-0.33, -0.27]$.

The estimates may be **distinct** as well: here, $\hat{\beta}_1 = -0.276 \neq -0.299 = \hat{\gamma}_1$.

```
Residuals:
     Min       1Q   Median       3Q      Max
-14.9729  -1.9716   0.1726   2.9727  11.0275

Coefficients:
                Estimate Std. Error t value Pr(>|t|)
(Intercept)     78.99279    0.48357  163.35   <2e-16 ***
infant_mortality -0.29888    0.01313  -22.76   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 4.174 on 164 degrees of freedom
Multiple R-squared:  0.7595,    Adjusted R-squared:  0.758
F-statistic: 517.9 on 1 and 164 DF,  p-value: < 2.2e-16
```

Feature Selection (First Pass)

How do we determine if **all** the predictors help explain the response Y, or if only **a (proper) subset** of the predictors is needed?

The most direct approach to solve this problem (in the OLS context) is to run **best subsets** regression.

The procedure is simple: fit an OLS model for **all possible subsets of predictors** and select the **optimal model** based on a criterion that balances **training error** with **model size**.

There are 2^{p+1} such models (so quickly becomes **unmanageable**). In practice, we need to **automate** and **speed-up the search** through a collection of **predictor subsets**. OLS approaches include **forward selection** and **backward selection** (see Chapter 5).

Forward step-wise selection is a **bottom-up** approach:

1. start with the **null model** $\mathcal{M}_0 : Y = \beta_0 + \varepsilon$;
2. fit p simple linear regressions $Y = \beta_0 + \beta_j X_j + \varepsilon$ and add to $\mathcal{M}_0$ the predictor X_{j_1} resulting in the **lowest** SSE: $\mathcal{M}_1 : Y = \beta_0 + \beta_{j_1} X_{j_1} + \varepsilon$;
3. add to $\mathcal{M}_1$ the predictor X_{j_2} that results in the **lowest** SSE among all the $2-$**variable models**: $\mathcal{M}_2 : Y = \beta_0 + \beta_{j_1} X_{j_1} + \beta_{j_2} X_{j_2} + \varepsilon$;
4. the process continues until some **stopping criterion** is met.

Backward step-wise selection is a **top-down** approach, and it works in reverse, **removing** predictors from the **full model** one-by-one.

In both approaches, we have to run numerous regressions; at most

$$p + (p-1) + \cdots + 2 + 1 = \frac{p(p+1)}{2} \ll 2^{p+1} \quad \text{(when } p \text{ is large)}$$

Other Questions

How do we handle:

- **qualitative variables**? (dummy binary variables);
- **interaction terms**? (add features);
- **outliers**? (median regression, Theil-Sen estimate);
- **heteroscedasticity**? (data transformations, weighted least square regression, Bayesian regression);
- **high-leverage** observations? (robust regression);
- **collinearity**? (centered variables, principal component analysis, generalized linear models, partial least square regression);
- **multiple tests**? (Bonferroni correction, Working-Hotelling procedure, Scheffé procedure).

2.2.3 – Generalizations

Dilemma: OLS assumptions are **convenient** but not always **realistic**:

- use **remedial measures** to transform the data into a compliant set;
- **extend assumptions** and work out corresponding formalism.

Extensions:

- **generalized linear models** (GLM) – responses with non-normal distributions (next);
- **classifiers** – extend regression to categorical responses (Chapter 3);
- **non-linear methods** – responses not linear combinations of the predictors (Section 2.6);
- **tree-based methods** and **ensemble learning methods** – used for predictor interactions (Sections 3.6, 3.7);
- **regularization methods** – model selection and feature selection (Section 2.3).

Generalized Linear Models

Apart from the **error structure**, a GLM is essentially a linear model:

$$Y_i \sim \mathcal{D}(\mu_i; \boldsymbol{\theta}_i), \quad \text{where} \quad g(\mu_i) = \eta_i = \mathbf{x}_i^\top \boldsymbol{\beta}, \quad i = 1, \ldots, N.$$

A GLM consists of:

- a **systematic component** $\mathbf{x}_i^\top \boldsymbol{\beta}$;
- a **random component** specified by the distribution $\mathcal{D}$ for Y_i, and
- a **link function** g.

The **systematic component** is specified in terms of the **linear predictor** for the i^{th} observation $\eta_i = \mathbf{x}_i^\top \boldsymbol{\beta}$; the general ideas and concepts of OLS carry over to GLM, with the added presence of the **link function** and the **random component** of the response y_i.

In practice, the **link function** g is often taken to be **smooth** and **monotonic** (so at least differentiable and invertible).

The **random component** $\mathcal{D}$ is usually selected from the **exponential family** of distributions (normal, binomial, Poisson, Gamma, etc.)

Example: OLS

- **systematic component** $\eta_i = \mathbf{x}_i^\top \boldsymbol{\beta}$;
- **random component** $Y_i \sim \mathcal{N}(\mu_i, \sigma^2)$;
- **link function** $g(\mu) = \mu$.

More substantial example: in the early stages of a rumour spreading, the rate at which new individual learn the information **increases exponentially** over time.

If μ_i is the **expected number of people** who have heard the rumour on day t_i, a model of the form $\mu_i = \gamma \exp(\delta t_i)$ might be appropriate:

$$\underbrace{\ln(\mu_i)}_{\text{link}} = \ln \gamma + \delta t_i = \beta_0 + \beta_1 t_i = \underbrace{(1, t_i)^\top (\beta_0, \beta_1)}_{\text{systematic component}} .$$

Furthermore, since we measure a count of individuals, it could be reasonable to use the **Poisson** distribution:

$$Y_i \sim \underbrace{\text{Poisson}(\mu_i)}_{\text{random component}}, \quad \ln(\mu_i) = (1, t_i)^\top (\beta_0, \beta_1).$$

GLM advantages:

- no need to transform the response Y if it is **not normal**;
- if the link produces **additive effects**, homoscedasticity is not **necessary**;
- link and random component are separate $\Longrightarrow$ **model flexibility**;
- models are still fitted *via* a **maximum likelihood** procedure;
- **inference tools** and **model checks** still apply;
- **easily implemented** (`glm()`, etc.), and
- framework unites various regression modeling approaches (OLS, logistic, Poisson, etc.) under a **single umbrella**.

2.3 – Regularization

We discuss the curse of dimensionality (CoD), subset selection, and dimension reduction in Chapter 5.

Another approach to dealing with high-dimensionality: **least absolute shrinkage and selection operator** (LASSO) and its variants.

Assumptions: Tr consists of N **centered** and **scaled** observations $\mathbf{x}_i = (x_{i,1}, \cdots, x_{i,p-1})$, with responses y_i (and intercept $x_{i,0} = 1$).

We recast OLS in the following form:

$$\hat{\boldsymbol{\beta}}_{\mathrm{OLS}} = \arg\min_{\boldsymbol{\beta}}\{\|\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}\|_2^2\} = \arg\min_{\boldsymbol{\beta}}\{\mathrm{SSE}\}.$$

In general, there are **no restrictions** on the values of the coefficients $\hat{\beta}_{\mathrm{OLS},j}$, $j = 1, \ldots, p-1$ – large magnitudes imply that corresponding features **play an important role** in predicting the target. Remember: the predictors are **centered**.

This observation forms the basis of a series of useful OLS variants.

Throughout, we let

- $\hat{\beta}_{\mathrm{OLS},j}$ be the jth OLS coefficient, and
- $\lambda > 0$ be a **threshold** whose value depends on the specific training dataset Tr.

Ridge Regression

RR is a method to **regularize** the OLS regression coefficients.

Effectively, it shrinks the OLS coefficients by penalizing solutions with **large magnitudes** – if the magnitude of a specific coefficient is large, then it must have great relevance in predicting the target variable.

If the solution still comes out with a **large magnitude** for a given coefficient in spite of the penalty, then that predictor must **really be important**.

This leads to a modified OLS problem:

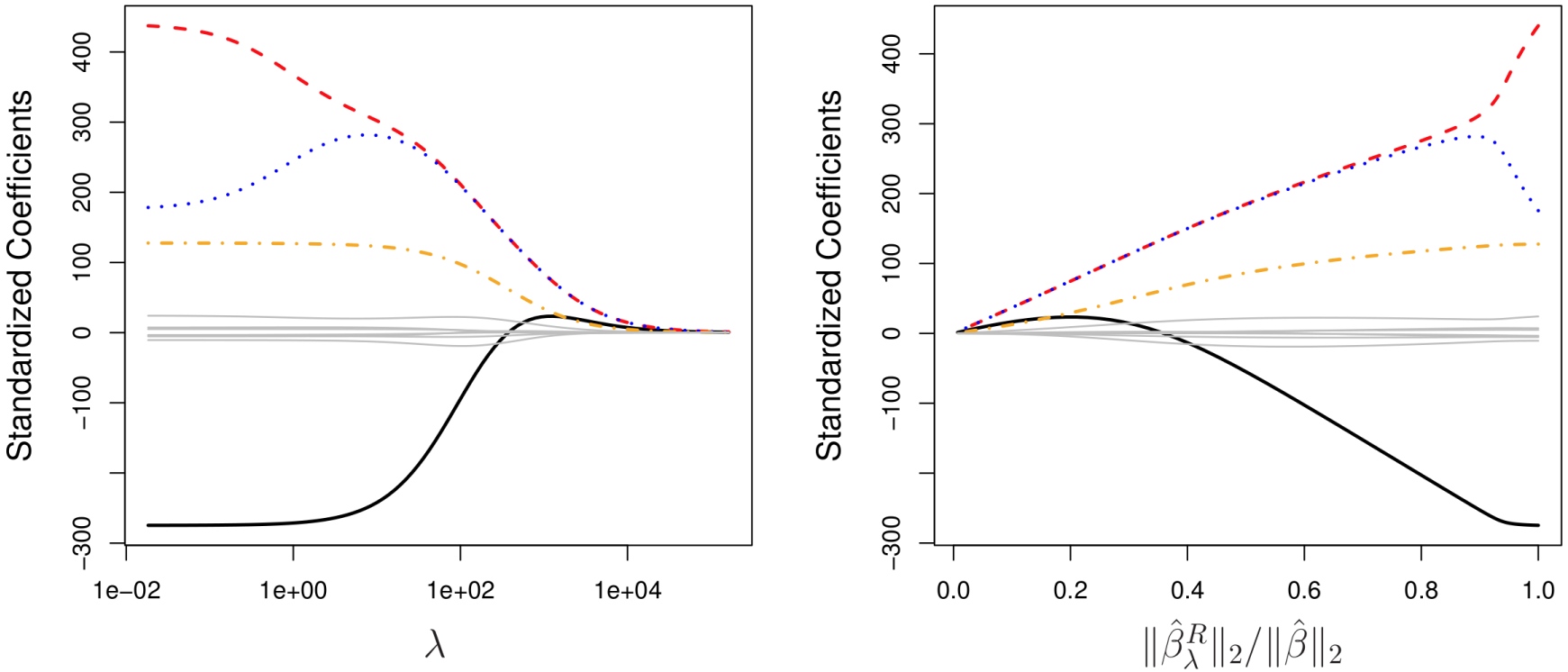
$$\hat{\boldsymbol{\beta}}_{\mathrm{RR}} = \arg\min_{\boldsymbol{\beta}} \{ \underbrace{\|\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}\|_2^2}_{\mathrm{SSE}} + \underbrace{N\lambda\|\boldsymbol{\beta}\|_2^2}_{\text{shrinkage penalty}} \}, \quad \|\boldsymbol{\beta}\|_2^2 = \sum_j \beta_j^2.$$

The objective function value is **small** when SSE is **small** (i.e., the model is a **good fit** to the data on Tr) and when the **shrinkage penalty** is **small** (i.e., when each β_j is **small**).

RR solutions are typically obtained *via* numerical methods.

The **hyperparameter** $\lambda > 0$ controls the **relative impact** of both components:

- if λ is **small**, then the shrinkage penalty is **small** even if the individual coefficients β_j are **large**;
- if λ is **large**, then the shrinkage penalty is only small when all coefficients β_j are **small**.



Ridge regression coefficients in a generic problem; note how the coefficients converge to 0 when the threshold lambda increases (left); the ratio between the magnitude of the ridge regression parameter and the corresponding OLS parameter is shown on the right.

Setting the proper value for λ is crucial; it can be done via **cross-validation** (Section 2.4.1).

Note that he OLS estimates are **equivariant**: if $\hat{\beta}_j$ is the estimate for the coefficient β_j of X_j, then $\hat{\beta}_j/c$ is the estimate for the coefficient of the scaled variable cX_j.

RR coefficients do not have this property, however, which is why the dataset must be centered and scaled to start with.

Why use RR? RR estimates can help mitigate the **bias-variance trade-off** and reduce issues related to **overfitting**, even if they do not actually **reduce the dimensions** of Tr.

The next variant does so, however.

Regression With Best Subset Selection

BSS uses a different **penalty term**, which effectively sets some of the coefficients to $0 \implies$ only select the features with **non-zero coefficients**?

The problem consists in solving a modified version of the OLS scenario:

$$\hat{\boldsymbol{\beta}}_{\text{BS}} = \arg\min_{\boldsymbol{\beta}} \{ \underbrace{\|\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}\|_2^2}_{\text{SSE}} + \underbrace{N\lambda\|\boldsymbol{\beta}\|_0}_{\text{shrinkage penalty}} \}, \quad \|\boldsymbol{\beta}\|_0 = \sum_j \text{sgn}(|\beta_j|).$$

Solving the BS problem typically (also) requires **numerical methods** and **cross-validation**.

The slight modification to the shrinkage penalty is enough to overcome the lack of equivariance.

LASSO

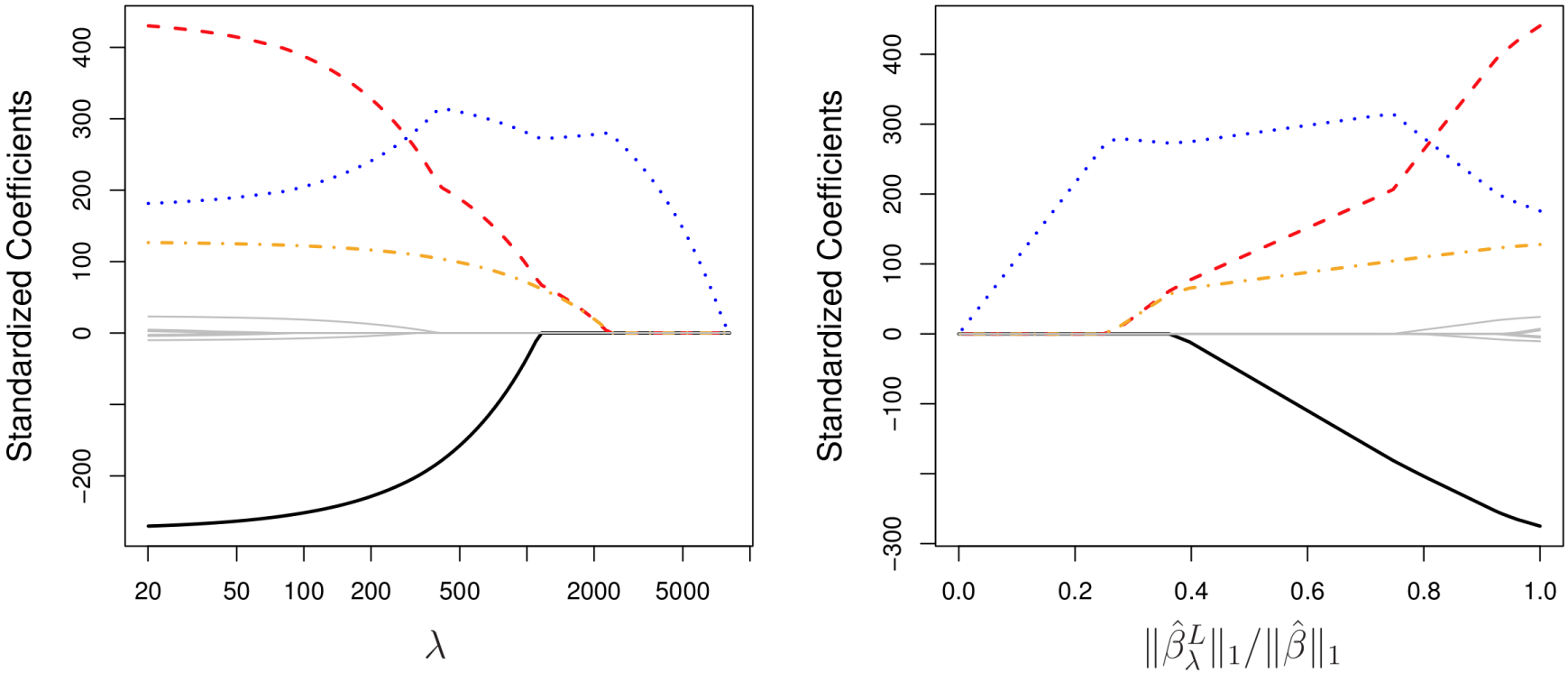
LASSO is an alternative to RR and BS obtained by solving

$$\hat{\boldsymbol{\beta}}_{\mathrm{L}} = \arg\min_{\boldsymbol{\beta}} \{ \underbrace{\|\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}\|_2^2}_{\mathrm{SSE}} + \underbrace{N\lambda\|\boldsymbol{\beta}\|_1}_{\text{shrinkage penalty}} \}, \quad \|\boldsymbol{\beta}\|_1 = \sum_j |\beta_j|.$$

The penalty effectively forces coefficients which **combine the properties** of RR and BS, selecting **at most** $\max\{p, N\}$ features.

At most **one predictor** is selected per group of **highly correlated variables** (the other coefficients are forced down to 0 when λ is **large enough**).

The LASSO typically produces **simpler models**, but **predictive accuracy** matters too (i.e., $\mathrm{MSE}_{\mathrm{Te}}$).



LASSO coefficients in a generic problem; note how the coefficients goes directly to 0 after a certain threshold lambda (left); the ratio between the magnitude of the LASSO parameter and the corresponding OLS parameter is shown on the right.

Generalizations

Other penalty functions (or **combinations**) provides various extensions: **elastic nets**; **group, fused and adaptive lassos**; **bridge regression**, etc.

The regularization formalism assumed an **underlying OLS model**, but it generalizes to **arbitrary regression/classification** models as well.

For a **loss** function $\mathcal{L}(\mathbf{Y}, \hat{\mathbf{Y}}(\mathbf{W}))$ between the actual target $\mathbf{Y}$ and the fitted values $\hat{\mathbf{Y}}(\mathbf{W})$ (with model parameters $\mathbf{W}$), and a **penalty** vector $\mathbf{R}(\mathbf{W}) = (R_1(\mathbf{W}), \cdots, R_k(\mathbf{W}))^\top$, the **regularized parametrization** $\mathbf{W}^*$ solves the **general regularization** problem

$$\mathbf{W}^* = \arg\min_{\mathbf{W}} \{ \underbrace{\mathcal{L}(\mathbf{Y}, \hat{\mathbf{Y}}(\mathbf{W}))}_{\text{loss}} + \underbrace{N\boldsymbol{\lambda}^\top \mathbf{R}(\mathbf{W})}_{\text{shrinkage penalty}} \}, \quad \boldsymbol{\lambda} \in \mathbb{R}^k_+ \text{ (cross-validation).}$$

Gapminder Example

In `R`, regularization is implemented in the package `glmnet` (among others). In `glmnet()` the parameter `alpha` controls the elastic net mixture: LASSO (`alpha = 1`), RR (`alpha = 0`).

We are interested in modeling **life expectancy** Y in the 2011 Gapminder dataset as a function of **population**, **infant mortality**, **fertility**, **gdp**, and continental membership (we use the entire set as a training set Tr).

A priori, an OLS model on Tr would take the form

$$\begin{aligned} Y = \alpha_0 &+ \alpha_1 \cdot \text{population} + \alpha_2 \cdot \text{infant mortality} + \alpha_3 \cdot \text{fertility} + \alpha_4 \cdot \text{gpd} \\ &+ \alpha_5 \cdot \text{Africa} + \alpha_6 \cdot \text{Americas} + \alpha_7 \cdot \text{Asia} + \alpha_8 \cdot \text{Europe} + \alpha_9 \cdot \text{Oceania}. \end{aligned}$$

Data prep: (DUDADS, 20.2, *Regression Modeling*, Shrinkage Methods)

1. create **dummy variables** for the continents;
2. select the **response variable**;
3. **scale** and **center** Tr.

The LASSO model ($\alpha = 1$) for hyperparameter $\lambda = 1$ is:

$$Y = 70.82 - 5.58(\text{infant mortality}) - 1.13(\text{Africa}) - 0.03(\text{Oceania})$$

The RR model ($\alpha = 0$) for hyperparameter $\lambda = 1$ is:

$$\begin{aligned} Y = 70.82 &- 0.34(\text{population}) - 4.4(\text{infant mortality}) - 0.63(\text{fertility}) \\ &+ 0.58(\text{gdp}) - 1.62(\text{Africa}) + 0.55(\text{Americas}) + 0.61(\text{Asia}) \\ &+ 1.01(\text{Europe}) - 0.68(\text{Oceania}) \end{aligned}$$

The coefficient values themselves are not as important as their signs and the fact that they are roughly similar in both models.

Comments:

- the choice of $\lambda = 1$ was **arbitrary**;
- we have not been evaluating the result on **test data** ;
- a regular OLS on the centered and scaled Tr yields

$$\begin{aligned} Y = 70.82 &- 0.29(\text{population}) - 6.1(\text{infant mortality}) + 0.74(\text{fertility}) \\ &+ 0.56(\text{gdp}) + 0.19(\text{Africa}) + 2.01(\text{Americas}) + 2.36(\text{Asia}) \\ &+ 2.63(\text{Europe}) + 0.00(\text{Oceania}) \end{aligned}$$

2.4 – Resampling Methods

Problem: how to determine the **variability** of a regression fit?

Solution: draw **different samples** from available data, **fit a model** to each **sample**, **examine the extent** to which the **fits differ** from one another.

Resampling methods provide additional information about a fitted model, by applying the same fitting approach to various sub-samples of Tr:

- **cross-validation** estimates the **test error** associated with a modeling approach in order to evaluate **model performance**;
- the **bootstrap** provides a measure of **accuracy**, **standard deviation**, **bias**, etc. of various model parameter estimates;
- the **jackknife** is a simpler approach with the same aims as the **bootstrap**.

The **test error** $\mathcal{E}_{\mathrm{Te}}$ associated with a ML model is the **average error** from predicting the response for observations **not in** Tr.

The **training error** $\mathcal{E}_{\mathrm{Tr}}$ is computed directly by comparing the **model's predictions** to the **true responses in** Tr.

In general, the latter underestimates the former, **dramatically so** when the model complexity increases (see **variance-bias trade-off**).

Classical solutions:

- make **direct adjustments** to $\mathcal{E}_{\mathrm{Tr}}$ to estimate $\mathcal{E}_{\mathrm{Te}}$ (e.g., Mallow's C_p statistic, R_a^2, AIC, BIC, etc. in OLS);
- ignore the problem and hope it goes away.

Drawbacks: **adjustment statistics** are not always readily available; and the other one... well.

The **validation approach** is used to estimate the **test error** associated with a particular statistical model on a set of observations.

1. set aside a **random hold-out** subset $\mathrm{Va} \subseteq \mathrm{Tr}$;
2. **fit the model** on $\mathrm{Tr} \setminus \mathrm{Va}$;
3. **evaluate** the model's performance on Va;
4. the resulting **validation set error** $\mathcal{E}_{\mathrm{Va}}$ provides an estimate for $\mathcal{E}_{\mathrm{Te}}$.

This approach is easy to **implement** and **interpret**, but:

- $\mathcal{E}_{\mathrm{Va}}$ is **volatile**, and **highly dependent** on the choice of Va;
- the model is fitted on a **proper subset** of the available observations, possibly leading to $\mathcal{E}_{\mathrm{Va}} \gg \mathcal{E}_{\mathrm{Te}}$, in general;
- a number of classical statistical models can provide $\mathcal{E}_{\mathrm{Te}}$ estimates without having to resort to such an approach.

2.4.1 – Cross-Validation

K-fold cross-validation is used to estimate the **test error** without losing observations to a **hold-out** set. It also provides a basis for **model selection**.

Procedure:

1. Divide Tr **randomly** into K "equal"-sized **folds** (typically, $K = 4, 5, 10$).

2. Each fold plays, in succession, the role of Va. If there are N observations in the dataset, partition

$$\{1, \ldots, N\} = \underbrace{\mathcal{C}_1}_{\text{fold } 1} \sqcup \cdots \sqcup \underbrace{\mathcal{C}_K}_{\text{fold } K}.$$

   If $|\mathcal{C}_k| = n_k$, we expect $n_k \approx \frac{N}{K}$ for all $k = 1, \ldots, K$.

3. For all $k = 1, \ldots, K$, **fit a model** on observations $\{1, \ldots, N\} \setminus \mathcal{C}_k$ and denote the **validation error** on $\mathcal{C}_k$ by E_k. There are multiple options but regression models typically use

$$E_k = \sum_{i \in \mathcal{C}_k} \frac{(y_i - \hat{y}_i)^2}{n_k}.$$

4. Write $\overline{E}$ for the **average** of the E_k.

5. The **cross-validation estimate** of the $\mathcal{E}_{\text{Te}}$ and its **standard error** are

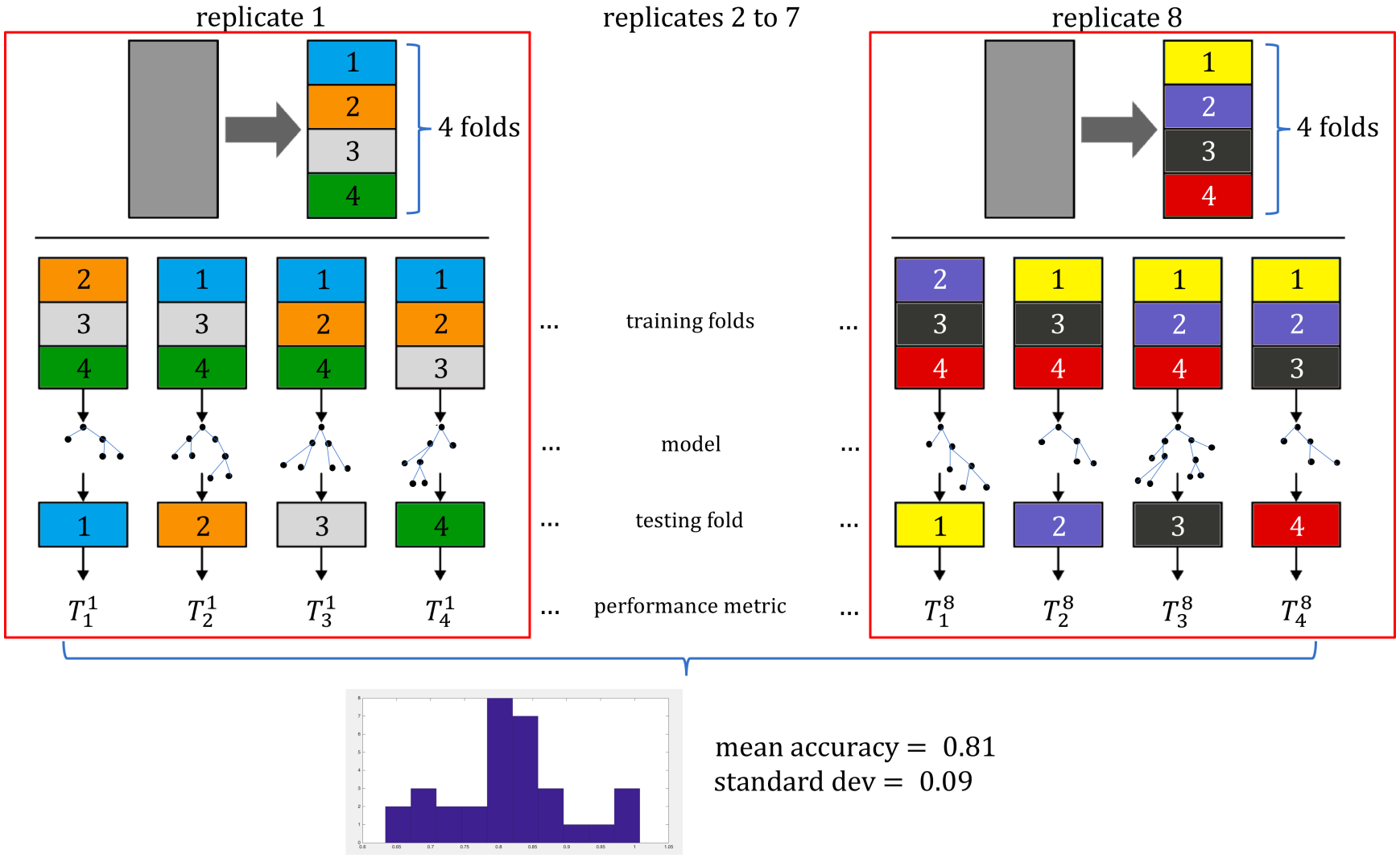
$$\text{CV}_{(K)} = \sum_{k=1}^{K} \frac{n_k}{N} E_k, \quad \widehat{\text{se}}\left(\text{CV}_{(K)}\right) = \sqrt{\frac{1}{K-1} \sum_{k=1}^{K} (E_k - \overline{E})^2}.$$

$\mathrm{CV}_{(K)}$ can help estimate how well a ML procedure performs on unseen data. But if we are interested in **selecting a method** from a **list of candidates** we do not care about the **specific value of** $\mathrm{CV}_{(K)}$ so much as for **which method** it is **minimal**.

From the perspective of **bias reduction** (in the estimate for $\mathcal{E}_{\mathrm{Te}}$), the "best" choice is $K = N$, yielding N models and N estimates for $\mathcal{E}_{\mathrm{Te}}$. These estimates are **highly correlated** and the mean of highly correlated estimates has **high variance** (see bias-variance trade-off and Section 2.4.3).

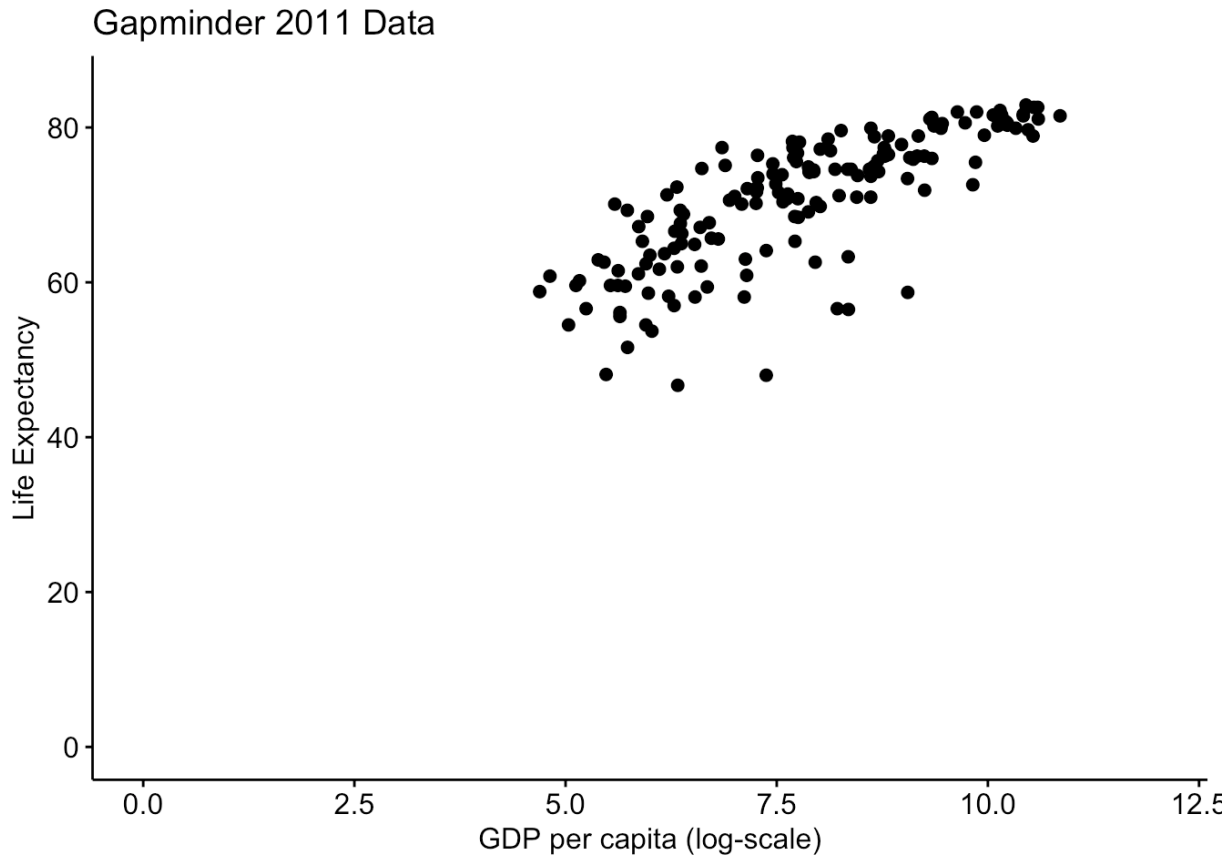
This algorithm could be **replicated** n times to generate the distribution of some **evaluation metric**, not necessarily $\mathcal{E}_{\mathrm{Te}}$.

A schematic illustration of $n = 8$ replicates of $K = 4-$fold cross-validation is shown on the next slide.



mean accuracy = 0.81
standard dev = 0.09

Gapminder Example



We use cross-validation in the `Gapminder` dataset to estimate the test error $\mathrm{MSE}_{\mathrm{Te}}$ when predicting life expectancy against the logarithm of the GDP per capita for the 2011 data (code in DUDADS, 20.3, $\mathit{Cross\text{-}Validation}$).

We split the dataset into $K = 10$ **random** folds $\mathcal{C}_k$ ($n_k = 16$ or 17), and fit an OLS model

$$\Lambda_k : y_{i,k} = \beta_{0;k} + \beta_{1;k} x_{i,k} + \varepsilon_{i,k}, \quad i = 1, \ldots, n_k, k = 1, \ldots, K,$$

with $(x_{i,k}, y_{i,k}) \in \mathrm{Tr} \setminus \mathcal{C}_k$ (containing 149 or 150 observations).

The estimates for β_0, β_1, and $\mathrm{MSE}_{\mathrm{Te}}$ are likely to be **correlated** accross $\mathcal{C}_k$, since the $\mathrm{Tr} \setminus \mathcal{C}_k$ share a **fair number** of observations.

Each $\mathcal{C}_k$ is used, in turn, to evaluate Λ_k's MSE performance.

n_k	$\hat{\beta}_{0;k}$	$\hat{\beta}_{1;k}$	$\mathrm{MSE}_{\mathrm{CV};k}$
17	37.69812	4.254105	33.859051
17	36.22257	4.442061	21.415376
17	37.59386	4.247255	45.620933
17	36.66761	4.345484	29.584469
17	37.49685	4.268917	24.127300
17	36.49849	4.386124	19.398769
16	36.78991	4.380887	48.157391
16	36.91113	4.331365	23.142625
16	37.41767	4.274771	7.837172
16	37.68955	4.254600	19.743046

The $10-$fold CV estimate of $\mathrm{MSE}_{\mathrm{Te}}$ is thus

$$\overline{\mathrm{MSE}_{\mathrm{Te}}} = \frac{1}{10}\sum_{k=1}^{10}\mathrm{MSE}_{\mathrm{Te}_k} = 27.29;$$

$$\mathrm{CV}_{(K)} = \sum_{k=1}^{10}\frac{n_k}{166}\mathrm{MSE}_{\mathrm{Te}_k} = 27.35;$$

$$\widehat{\mathrm{se}}\left(\mathrm{CV}_{(K)}\right) = \sqrt{\frac{1}{10-1}\sum_{k=1}^{10}(\mathrm{MSE}_{\mathrm{Te}_k} - \overline{\mathrm{MSE}_{\mathrm{Te}}})^2} = 12.38.$$

Thus, $27.35 \pm 2(12.38) \equiv (2.59, 52.11)$ is a 95% C.I. for the $\mathrm{MSE}_{\mathrm{Te}}$.

We can also get $10-$fold CV estimates of β_0, β_1: we have

$$\beta_{0_{(K)}} = \sum_{k=1}^{10} \frac{n_k}{166} \beta_{0;k} = 37.10, \quad \widehat{\text{se}}\left(\beta_{0_{(K)}}\right) = \sqrt{\frac{1}{10-1} \sum_{k=1}^{10} (\beta_{0;k} - \overline{\beta_0})^2} = 0.54;$$

$$\beta_{1_{(K)}} = \sum_{k=1}^{10} \frac{n_k}{166} \beta_{1;k} = 4.32, \quad \widehat{\text{se}}\left(\beta_{1_{(K)}}\right) = \sqrt{\frac{1}{10-1} \sum_{k=1}^{10} (\beta_{1;k} - \overline{\beta_1})^2} = 0.07;$$

so

$$\text{C.I.}(\beta_0; 0.95) \equiv 37.10 \pm 2(0.54) \equiv (36.00, 38.18);$$
$$\text{C.I.}(\beta_1; 0.95) \equiv 4.32 \pm 2(0.07) \equiv (4.18, 4.56).$$

LASSO and Regression Ridge Revisited

How would we pick the optimal **hyperparameter** λ in **regularization** methods? Let us revisit the example from Section 2.3.

As before, we are interested in modeling **life expectancy** Y in the 2011 `Gapminder` dataset as a function of **population**, **infant mortality**, **fertility**, **gdp**, and **continental membership**.

We run a $5-$**fold cross-validation** LASSO regression for a variety of hyperparameter values λ, and evaluate $\mathrm{CV}_{(5)}(\lambda)$, the **CV test error** for each λ (using MSE).

The **optimal** λ^* is the one that **minimizes** $\mathrm{CV}_{(5)}(\lambda)$.

The **LASSO model** for the optimal hyperparameter $\lambda^* = 0.31$ is:

$$\begin{aligned} Y = 70.82 &- 5.74(\text{infant mortality}) + 0.16(\text{gdp}) - 1.76(\text{Africa}) \\ &+ 0.12(\text{Europe}) - 0.80(\text{Oceania}) \end{aligned}$$

The **RR model** for the optimal hyperparameter $\lambda = 0.74$ is:

$$\begin{aligned} Y = 70.82 &- 0.34(\text{population}) - 4.7(\text{infant mortality}) - 0.42(\text{fertility}) \\ &+ 0.58(\text{gdp}) - 1.62(\text{Africa}) + 0.55(\text{Americas}) + 0.63(\text{Asia}) \\ &+ 1.01(\text{Europe}) - 0.72(\text{Oceania}) \end{aligned}$$

Compare with the OLS results, and the LASSO and RR results when $\lambda = 1$.

2.4.2 – Bootstrapping

The **bootstrap procedure** uses **re-sampling** of Tr to **mimic the process of obtaining new replicates** and estimate the variability of a model parameter **without the need to generate new observations**.

A **bootstrap sample** Tr^* of Tr, with $|\mathrm{Tr}| = N$, is a sample of N observations drawn from Tr **with replacement**.

The process is **repeated** m times to obtain **bootstrap samples** Tr_i^* and parameter estimates $\hat{\alpha}_i^*$, for $i = 1, \ldots, M$, from which we derive a **bootstrap estimate** (with standard error):

$$\hat{\alpha}^* = \frac{1}{M}\sum_{i=1}^{M}\hat{\alpha}_i^*, \quad \widehat{\mathrm{se}}\,(\hat{\alpha}^*) = \sqrt{\frac{1}{M-1}\sum_{i=1}^{M}(\hat{\alpha}_i^* - \hat{\alpha}^*)^2}.$$

The bootstrap can also be used to build **approximate frequentist** C.I. for the parameter α; this is not as straightforward as one might think, so caution is advised.

We can even construct a **covariance structure** for the parameters, given **enough** replicates. We will see an example focusing on regression coefficients in the next example.

In more **complex/realistic** scenarios, the appropriate bootstrap procedure might need to be more **sophisticated** than what we have described.

For instance, sampling **with replacement** at the observation level would not preserve the covariance structure of **time series** data, so we might need to sample over **time periods**.

It is also not typically as accurate for **correlated data**.

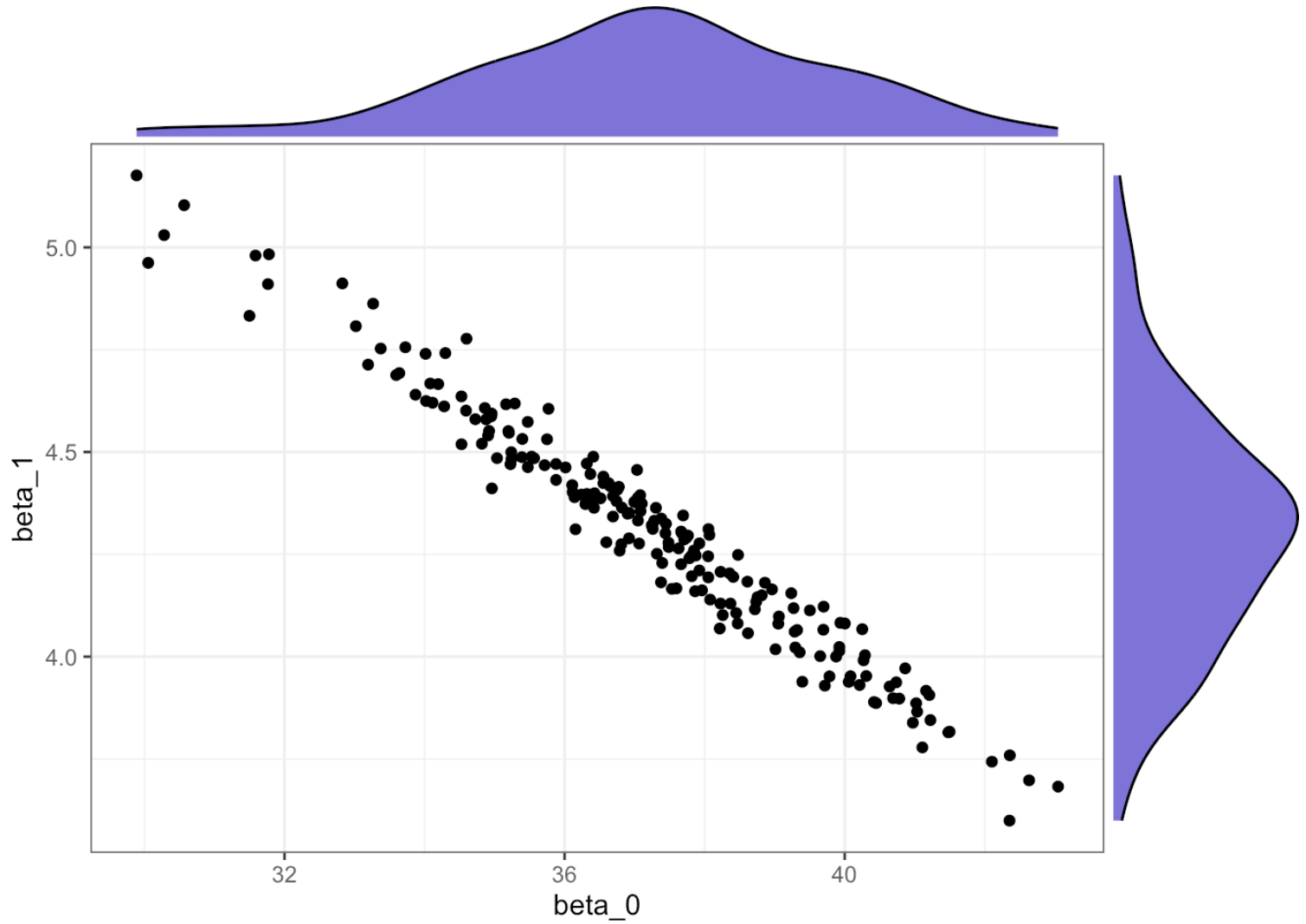
Gapminder Example

We use the bootstrap procedure for the OLS problem with **life expectancy** Y and the **log of the GDP per capita** X in the 2011 `Gapminder` data (code found in DUDADS, 20.3, $Bootstrap$).

We draw $M = 200$ bootstrap samples of size $N = 166$ from the original dataset (**with replacement**). For each sample $1 \leq i \leq M$, we find the OLS fit and retain the **intercept** $\beta_{0;i}$ and **slope** $\beta_{1;i}$.

The **joint distribution** of $\boldsymbol{\beta} = (\beta_0, \beta_1)^\top$ is shown on the next slide, together with the **marginal distributions** for each parameter.

(It would be useful to learn to make such graphs)


5.0
4.5
4.0
beta_1
32
36
40
beta_0

We see that $\boldsymbol{\beta}$ roughly follows a multivariate normal $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, with

$$\boldsymbol{\mu} \approx \hat{\boldsymbol{\mu}}^* = \begin{pmatrix} 37.22 \\ 4.31 \end{pmatrix}, \quad \boldsymbol{\Sigma} \approx \hat{\boldsymbol{\Sigma}}^* = \begin{pmatrix} 6.32 & -0.72 \\ -0.72 & 0.08 \end{pmatrix}.$$

The vector $\hat{\boldsymbol{\mu}}^*$ provides the **bootstrap estimates**.

The corresponding **standard error** estimates are $\widehat{\mathrm{se}}(\hat{\boldsymbol{\mu}}^*) = (2.51, 0.29)^\top$:

$$\mathrm{C.I.}(\beta_0; 0.95) = 37.22 \pm 2(2.51) \equiv (32.19, 42.25);$$
$$\mathrm{C.I.}(\beta_1; 0.95) = 4.31 \pm 2(0.29) \equiv (3.73, 4.89).$$

How do these compare to the cross-validation estimates?

2.4.3 – Jackknife

The **jackknife estimator** arises from cross-validation when $K = N$; the sole difference is in the **standard error estimate**:

$$\widehat{\mathrm{se}}(\hat{\alpha}^*) = \sqrt{\frac{N-1}{N}\sum_{i=1}^{N}(\hat{\alpha}_i^* - \overline{\hat{\alpha}^*})^2}.$$

It is also known as **leave-one-out-validation** (and typically provides looser estimates than cross-validation).

There is a **leave-d-out-validation** variant:

$$\text{“}N-1\text{”} \mapsto \text{“}N-d\text{”}, \quad \text{“}N\text{”} \mapsto \text{“}d\binom{N}{d}\text{”}.$$

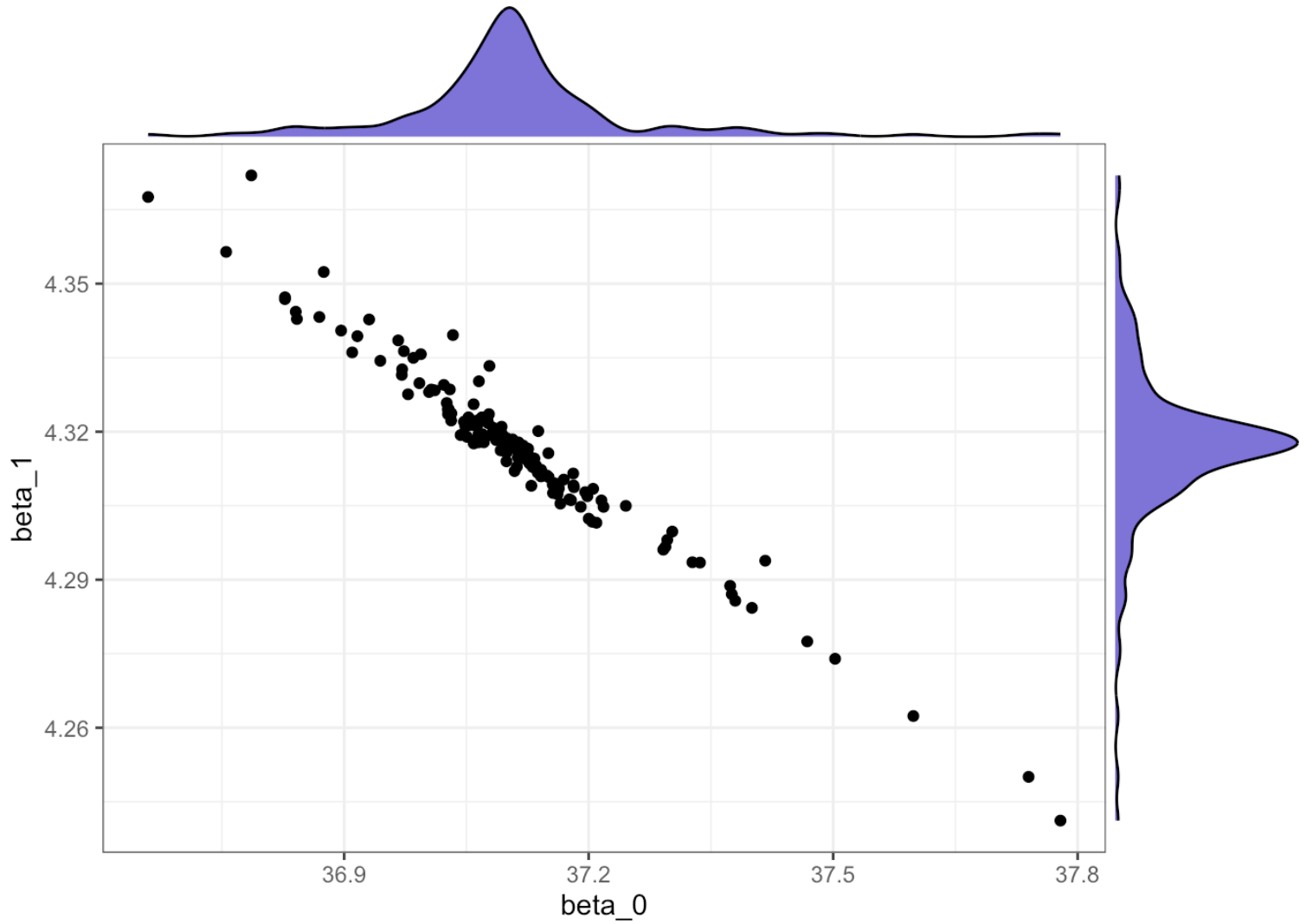
Gapminder Example

We use the jackknife procedure for the OLS problem with **life expectancy** Y and the **log of the GDP per capita** X in the 2011 `Gapminder` data (code found in DUDADS, 20.3, $Jackknife$).

For each fold $1 \leq k \leq N$, we find the OLS fit and retain the **intercept** $\beta_{0;k}$ and **slope** $\beta_{1;k}$.

The **joint distribution** of $\boldsymbol{\beta} = (\beta_0, \beta_1)^\top$ is shown on the next slide, together with the **marginal distributions** for each parameter.

Visually, we distinctly see that the number of jackknife points is smaller, but the covariance structure should at least "point" in the same direction as was the case for the bootstrap.


4.35
4.32
4.29
4.26
beta_1
36.9
37.2
37.5
37.8
beta_0

We see that $\boldsymbol{\beta}$ roughly follows a multivariate normal $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, with

$$\boldsymbol{\mu} \approx \hat{\boldsymbol{\mu}}^* = \begin{pmatrix} 37.11 \\ 4.32 \end{pmatrix}, \quad \boldsymbol{\Sigma} \approx \hat{\boldsymbol{\Sigma}}^* = \begin{pmatrix} 0.021 & -0.002 \\ -0.002 & 0.0003 \end{pmatrix}.$$

The vector $\hat{\boldsymbol{\mu}}^*$ provides the **jackknife estimates**.

The corresponding **standard error** estimates are $\widehat{\mathrm{se}}(\hat{\boldsymbol{\mu}}^*) = (1.86, 0.21)^\top$:

$$\mathrm{C.I.}(\beta_0; 0.95) = 37.22 \pm 2(1.86) \equiv (33.38, 40.83);$$
$$\mathrm{C.I.}(\beta_1; 0.95) = 4.31 \pm 2(0.21) \equiv (3.89, 4.79).$$

How do these compare to the cross-validation and the bootstrap estimates?

2.5 – Model Selection

A linear model $\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}$, $\boldsymbol{\varepsilon} \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I}_n)$ should be seen as an attempt to approximate the (**not necessarily linear**) regression function

$$y = f(\mathbf{x}) = \mathrm{E}\{Y \mid (X_1, \ldots, X_p) = \mathbf{x}\} \quad \text{(notice \# predictors)}.$$

For now, we assume a **linear relationship** between the response Y and the predictors $(X_1, \ldots, X_p)$, which we fit using the **OLS** framework:

$$\mathbf{b} = \arg\min_{\boldsymbol{\beta}} \{\|\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}\|_2^2\}.$$

But prediction accuracy suffers when $p > n$; model interpretability can be improved by removing **irrelevant features** (i.e., by **reducing** p).

There are 3 classes of methods to do so:

- **shrinkage/regularization methods** (see Section 2.3);
- **dimension reduction**, in which we project the p predictors onto an $M-$**dimensional manifold** $\mathcal{H}$, with $M \ll p$ (see Chapter 5), and
- **subset selection**, we identify a subset of the p predictors for which there is evidence of a **(strong) association** with the response, and we fit a model to this reduced set using the OLS framework – given p predictors (some of which may be interaction terms, binary variables, polynomial powers, etc.), there are 2^p OLS models that can be fit on Tr.

But which of those models should be selected as the **best model**?

2.5.1 – Best Subset Selection

In the **best subset selection** (BSS) approach, the search for the best model is usually broken down into 3 stages:

1. let $\mathcal{M}_0$ denote the **null model** (no predictor) which simply predicts the sample mean for all observations;

2. for $k = 1, \ldots, p$ (and as long as the model can be fit):

   (a) fit **every** model that contains k predictors (there are $\binom{p}{k}$ of them);
   (b) pick the model with **smallest** SSE (**largest** R^2) and denote it by $\mathcal{M}_k$;

3. select a **unique** model from $\{\mathcal{M}_0, \ldots, \mathcal{M}_p\}$ using C_p (AIC), BIC, R_a^2, or any other appropriate metric.

We cannot use SSE or R^2 as metrics in the last step, as we would always select $\mathcal{M}_p$ since SSE **decreases monotonically** with k and R^2 **increases monotonically** with k.

BSS is conceptually simple, but with 2^p models to try out, it quickly becomes **computationally infeasible** for large p ($p > 40$, say).

When p is large, the chances of finding a model that performs **well** according to step 3 but **poorly** for new data **increase**, which can lead to **overfitting** and **high-variance** estimates.

We are assuming that all models are **OLS** models, but subset selection algorithms can be used for other families of methods; all that is required are appropriate **training** error estimates for step 2b and **test** error estimates for step 3.

2.5.2 – Stepwise Selection

Stepwise selection (SS) methods attempt to overcome this challenge by only looking at a **restricted** set of models. **Forward stepwise selection** (FSS) starts with the **null model** $\mathcal{M}_0$ and adding predictors one-by-one until it reaches the **full model** $\mathcal{M}_p$:

1. Let $\mathcal{M}_0$ denote the **null model**;

2. For $k = 0, \ldots, p-1$ (and as long as the model can be fit):
   (a) consider the $p-k$ models that add a **single predictor** to $\mathcal{M}_k$;
   (b) pick the model with **smallest** SSE (**largest** R^2), denote it by $\mathcal{M}_{k+1}$;

3. select a **unique** model from $\{\mathcal{M}_0, \ldots, \mathcal{M}_p\}$ using C_p (AIC), BIC, R_a^2, or any other appropriate metric.

Backward stepwise selection (also BSS, unfortunately) works the other way, starting with the **full model** $\mathcal{M}_p$ and removing predictors one-by-one until it reaches the **null model** $\mathcal{M}_0$:

1. Let $\mathcal{M}_p$ denote the **full model**;

2. For $k = p, \ldots, 1$ (and as long as the model can be fit):

   (a) consider the k models that remove a **single predictor** from $\mathcal{M}_k$;
   (b) pick the model with **smallest** SSE (**largest** R^2), denote it by $\mathcal{M}_{k-1}$;

3. select a **unique** model from $\{\mathcal{M}_0, \ldots, \mathcal{M}_p\}$ using C_p (AIC), BIC, R_a^2, or any other appropriate metric.

The computational advantage of SS over B(est)SS is evident: instead of having to fit 2^p models, SS only requires to fit

$$1 + p + (p-1) + \cdots + 2 + 1 = \frac{p^2 + p + 2}{2}.$$

While there is no guarantee that the "**best**" model (among the 2^p B(est)SS models) is found in the SS models, SS can be used in settings where p is **too large** for BSS to be computationally feasible.

For OLS models, **backward SS** only works if $p \leq n$ (otherwise OLS might not have a unique parameter solution); if $p > n$, only **FSS** is viable.

Hybrid selection (HS) methods attempt to mimic BSS while keeping model computation in a manageable range, not unlike in SS.

2.5.3 – Selecting the Optimal Model

The **full model** always has **largest** R^2/**smallest** SSE.

Indeed, it is a measure of $\mathcal{E}_{\mathrm{Tr}}$, and as such, is subject to the **overfitting** property found in the **bias-variance trade-off** diagram

In order to estimate $\mathcal{E}_{\mathrm{Te}}$ (and thus pick the **optimal** model in the list $\{\mathcal{M}_0, \ldots, \mathcal{M}_p\}$, we can either:

- **adjust** $\mathcal{E}_{\mathrm{Tr}}$ to account for the bias induced by **overfitting**, and/or
- **directly** estimate $\mathcal{E}_{\mathrm{Te}}$ using a hold-out Va or **cross-validation**.

Adjustment Statistics

Commonly, we use one of the following **adjustment statistics**:

- **Mallow's C_p**
- the **Akaike information criterion** (AIC)
- the **Bayesian information criteria** (BIC), or
- the **adjusted coefficience of determination R_a^2**.

The first three of these must be **minimized**, while the last must be **maximized**.

The **adjustment statistics** require the following quantities:

- N, p, and $d = p + 2$
- $\hat{\sigma}^2$, the estimate of $\mathrm{Var}(\varepsilon)$ (the **irreducible error**);
- SSE and SST.

Mallow's C_p statistic is

$$C_p = \frac{1}{N}(\mathrm{SSE} + 2d\hat{\sigma}^2) = \frac{1}{N}\mathrm{SSE} + \underbrace{\frac{2d\hat{\sigma}^2}{N}}_{\text{adjustment}}.$$

As d increases, so does the adjustment term. Note that if $\hat{\sigma}^2$ is an unbiased estimate of $\mathrm{Var}(\varepsilon)$, C_p is an unbiased estimate of MSE.

The **Akaike information criterion** (AIC) is

$$\text{AIC} = -2 \ln L + \underbrace{2d}_{\text{adjustment}},$$

where L is the maximized value of the likelihood function for the estimated model. If the errors are **normally distributed**, this requires maximizing

$$\prod_{i=1}^{N} \frac{1}{\sqrt{2\pi}\hat{\sigma}} \exp\left(-\frac{(Y_i - \mathbf{X}_i\boldsymbol{\beta})^2}{2\hat{\sigma}^2}\right) = \frac{1}{(2\pi)^{n/2}\hat{\sigma}^n} \exp\left(-\frac{1}{2\hat{\sigma}^2}\sum_{i=1}^{N}(Y_i - \mathbf{X}_i\boldsymbol{\beta})^2\right),$$

or, upon taking the logarithm,

$$\ln L = \text{constant} - \frac{1}{2\hat{\sigma}^2}\|\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}\|^2,$$

and so

$$\arg\max_{\boldsymbol{\beta}}\{\ln L(\boldsymbol{\beta})\} = \arg\min_{\boldsymbol{\beta}}\{\|\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}\|^2\}.$$

However,

$$\begin{aligned}
\text{AIC} &= -2\ln L + 2d = \text{constant} + \frac{1}{\hat{\sigma}^2}\|\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}\|^2 + 2d \\
&= \text{constant} + \frac{\text{SSE}}{\hat{\sigma}^2} + 2d \\
&= \text{constant} + \frac{N}{\hat{\sigma}^2}\cdot\frac{1}{N}\left(\text{SSE} + 2d\hat{\sigma}^2\right) = \text{constant} + \frac{n}{\hat{\sigma}^2}C_p.
\end{aligned}$$

Evidently, when the error structure is normal, **minimizing** AIC is equivalent to **minimizing** C_p.

The **Bayesian information criterion** uses a different adjustment term:

$$\text{BIC} = \frac{1}{N}(\text{SSE} + d\hat{\sigma}^2 \ln N) = \frac{1}{N}\text{SSE} + \underbrace{d\hat{\sigma}^2 \frac{\ln N}{N}}_{\text{adjustment}}.$$

This adjustment penalizes models with a **large** number of predictors; **minimizing** BIC results in selecting models with fewer variables than those obtained by **minimizing** C_p, in general.

The **adj. coeff. of determ.** R^2_a of a $k-$parameter model $(X_0, \ldots, X_{k-1})$ is

$$R^2_{a,k} = 1 - \frac{\text{SSE}/(N-k)}{\text{SST}/(N-1)} = 1 - (1-R^2)\frac{N-1}{N-k}.$$

Maximizing $R^2_{a,k}$ **minimizes** $\frac{\text{SSE}}{n-k-1}$, penalizing **unnecessary** variables.

TL;DR: if p is the number of parameters in the **full model** (F), we want to find a **reduced model** (R) with k parameters that also fits the data well.

1. R^2_p**−criterion:** for each $k-$subset of parameters, compute the coefficient of determination $R^2_k = 1 - \frac{\text{SSE}_k}{\text{SST}}$; we find a $k-$subset such that if we increase k, the highest R^2_k does not change significantly (to 2 decimal places, say).

2. $R^2_{a,p}$**−criterion:** for each $k-$subset of parameters, compute the adjusted coefficient of determination $R^2_{a,k} = 1 - \frac{N-1}{N-k}\frac{\text{SSE}_k}{\text{SST}}$; we find a $k-$subset that maximizes $\{R^2_{a,k}\}$.

3. **Mallow's** C_p**−criterion:** $C_p = \frac{\text{SSE}_k}{\text{MSE}(F)} - (N - 2k)$; we find a $k-$subset such that C_p is small and close to k. This criterion might produce numerous appropriate reduced models.

2.5.4 – Generalization: Validation and Cross-Validation

But what if we also have non-OLS models?

To select an **optimal** $\mathcal{M}_{k^*}$ from a list of models $\{\mathcal{M}_1, \mathcal{M}_2, \ldots, \mathcal{M}_\ell\}$, either **minimize** $\mathrm{MSE}_{\mathrm{Va}}$ on a random Va over the models (**validation**); or **minimize** $\mathrm{CV}_{(K)}$ over the models (**cross-validation**).

The main advantages of this approach are that:

- there is **no need** to estimate the **irreducible error** $\mathrm{Var}(\varepsilon) = \sigma^2$;
- the method produces an estimate for $\mathrm{MSE}_{\mathrm{Te}}$ "**for free**";
- it can be used when p is **hard to pinpoint** (in DL networks, for instance).

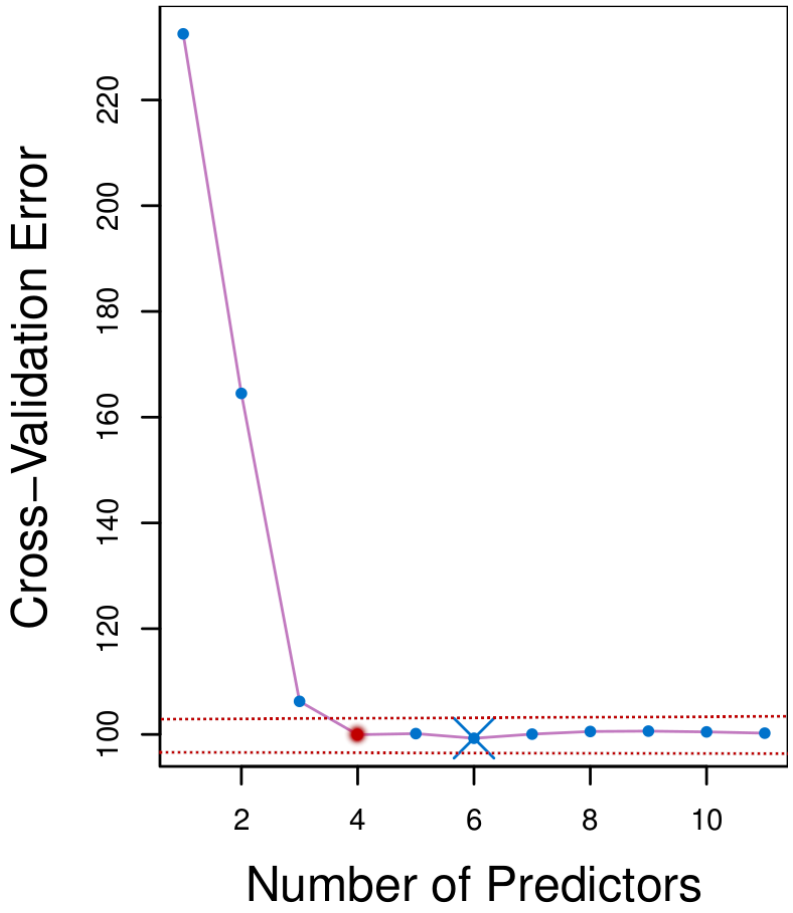
Historically: adjustment approaches were preferred because CV is **computationally demanding**, especially when p, n is **large**.

Modern Age: not as much of a problem.

Recommendation: CV is championed as the **optimal** model selection approach, using the **one standard error rule**:

1. compute $\widehat{\text{se}}\left(\widehat{\text{MSE}}_{\text{Te}}\right)$ for each $\mathcal{M}_k$
2. select the **smallest model** $\mathcal{M}_{k^*}$ for which $\widehat{\text{MSE}}_{\text{Te}}$ is within **one standard error** from the lowest point on the CV **error curve**.

Occam's Razor: "When presented with competing hypotheses about the same prediction, one should select the solution with the fewest assumptions" $\Longrightarrow$ use the **simplest model** from a group of models with **roughly similar** predictive power.



The lowest point is reached when $p = 6$ (**blue "X"**).

The dashed red lines represent the **1-standard error** limits:

$$\widehat{\mathrm{MSE}}_{\mathrm{Te}}(p = 6) \pm \widehat{\mathrm{se}}\left(\widehat{\mathrm{MSE}}_{\mathrm{Te}}(p = 6)\right)$$

According to the rule described above, we would select the model with $p = 4$ parameters (**red dot**).

S(tepwise) S(election) methods are used extensively in practice, but there issues:

- all intermediate tests are **biased**, as they are based on the **same data**;
- R_a^2 only takes into account the number of features in the **final model**, not the degrees of freedom that have been used up during the **entire process**;
- if the **CV error** is used, SS should be **repeated** for each sub-model.

SS is a classic example of $p-$**hacking**: obtaining results **before** declaring the hypotheses.

2.5.5 – Example: Credit Dataset

In spite of this warning, it could still be useful to know how to perform SS.

Goal: find the best FSS and BSS linear models to predict the credit card balance Y for observations contained in the training set `Credit.csv`.

```
'data.frame':   400 obs. of  12 variables:
 $ X         : int  1 2 3 4 5 6 7 8 9 10 ...
 $ Income    : num  14.9 106 104.6 148.9 55.9 ...
 $ Limit     : int  3606 6645 7075 9504 4897 8047 3388 7114 ...
 $ Rating    : int  283 483 514 681 357 569 259 512 266 491 ...
 $ Cards     : int  2 3 4 3 2 4 2 2 5 3 ...
 $ Age       : int  34 82 71 36 68 77 37 87 66 41 ...
 $ Education: int  11 15 11 11 16 10 12 9 13 19 ...
 $ Gender    : Factor w/ 2 levels "Female","Male": 2 1 2 1 2 2 ...
 $ Student   : Factor w/ 2 levels "No","Yes": 1 2 1 1 1 1 ...
 $ Married   : Factor w/ 2 levels "No","Yes": 2 2 1 1 2 1 ...
 $ Ethnicity: Factor w/ 3 levels "African American",..: 3 2 2 ...
 $ Balance   : int  333 903 580 964 331 1151 203 872 279 ...
```

We remove the id variable `X`, create **dummy variables** for the categorical levels, and **scale** the predictors (DUDADS, 20.4, *Selecting the Optimal Model*, Cross-Validation).

We start by implementing step 2 of the FSS algorithm.

```
 [1] "Rating"        "Income"              "Student.dummy"
 [4] "Limit"         "Cards"               "Age"
 [7] "Gender.dummy"  "Ethnicity.AA.dummy"  "Married.dummy"
[10] "Education"     "Ethnicity.A.dummy"
```

The best 1-par. model $\mathcal{M}_1$ uses `Rating`, the best 2-par. model $\mathcal{M}_2$ built from $\mathcal{M}_1$ uses `Rating` and `Income`, and so on.

Next, we implement step 3 by computing the **adjustment statistics** (AIC, BIC, R_a^2) and the **CV error** (with $K = 5$ folds) for each of $\mathcal{M}_0, \mathcal{M}_1, \ldots$.

The best FSS models using CV, R_a^2, AIC, and BIC are:

```
"Rating" "Income" "Student.dummy" "Limit" "Cards" "Age"

"Rating" "Income" "Student.dummy" "Limit" "Cards" "Age" "Gender.dummy"
"Ethnicity.AA.dummy" "Married.dummy"

"Rating" "Income" "Student.dummy" "Limit" "Cards" "Age" "Gender.dummy"

"Rating" "Income"
```

We can also implement BSS; its best models are:

```
"Income" "Limit" "Rating" "Cards" "Age" "Student.dummy"

"Income" "Limit" "Rating" "Cards" "Age" "Gender.dummy" "Student.dummy"
"Married.dummy" "Ethnicity.AA.dummy"

"Income" "Limit" "Rating" "Cards" "Age" "Gender.dummy" "Student.dummy"

"Income" "Rating"
```

2.6 – Nonlinear Modeling

The linearity assumption is **almost never met** in practice and the regression function $y = f(\mathbf{x}) = \mathrm{E}[Y \mid \vec{X} = \mathbf{x}]$ has to be approximated by **some other technique**.

But the assumption is often "**good enough**"; it is convenient and has multiple extensions, so it is rarely a waste of time to try that approach.

If that fails, it pays to first consider options which offer **flexibility** without sacrificing **interpretability** before jumping to **black box models**:

- **curve fitting** (polynomial regression, step functions, splines, etc.);
- **local regression methods**, or
- **generalized additive models**.

2.6.1 – Basis Functions

How do we determine whether Y is a linear combination of the predictors?

Diagnostic tests:

- visual tests, goodness-of-fit tests, Ramsey's RESET test

Remedial measures:

- transformations of the response or predictors, if appropriate
- adding new predictors that are **non-linear functions of the other predictors**, if we suspect that Y is not a linear combination of the predictors

$$y_i = \beta_0 + \sum_{j,k} \beta_{j,k} \underbrace{f_j(x_{k,i})}_{\text{basis function}} + \varepsilon_i, \quad i = 1, \ldots, N.$$

Polynomial Regression

We can extend the OLS model $y_i = \beta_0 + \beta_1 x_i + \varepsilon_i,\ i = 1, \ldots, N,$ by allowing for **polynomial basis terms** in the regression function:

$$y_i = f_d(x_i) = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \cdots + \beta_d x_i^d + \varepsilon_i, \quad i = 1, \ldots, N.$$

This f_d is **non-linear** in x_i, but it is **linear** in terms of the coefficients β_j.

Set $x_1 = X$, $x_2 = X^2$, and so on; we can estimate the regression function $y = f(\mathbf{x})$ *via* $\hat{f}(\mathbf{x}) = \mathbf{x}^\top \hat{\boldsymbol{\beta}}$, where $\hat{\boldsymbol{\beta}}$ is learned on Tr using OLS.

Typically, $\hat{\boldsymbol{\beta}}$ is of little interest – it is the **predictions** $\hat{f}(\mathbf{x})$ that are.

Since $\hat{f}(\mathbf{x})$ is **linear** in the coefficients $\hat{\beta}_i$, $i = 0, \ldots, d$, we have:

$$\begin{aligned}\mathrm{Var}(\hat{f}(\mathbf{x})) &= \mathrm{Var}(\mathbf{x}^\top \hat{\boldsymbol{\beta}}) = \sum_{i,j=0}^{d} \mathrm{Cov}(\hat{\beta}_i x_i, \hat{\beta}_j x_j) \\ &= \sum_{i,j=0}^{d} x_i x_j \,\mathrm{Cov}(\hat{\beta}_i, \hat{\beta}_j) = \mathbf{x}^\top \mathrm{Cov}(\hat{\boldsymbol{\beta}})\mathbf{x} = \sigma^2 \mathbf{x}^\top (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{x}\end{aligned}$$

$$\widehat{\mathrm{Var}}(\hat{f}(\mathbf{x})) = \frac{\mathrm{SSE}}{n-d-1} \mathbf{x}^\top (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{x} = \frac{\|\mathbf{Y} - \mathbf{X}\hat{\boldsymbol{\beta}}\|_2^2}{n-d-1} \cdot \mathbf{x}^\top (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{x};$$

assuming normality of the error terms, we have:

$$\mathrm{C.I.}(\hat{f}(\mathbf{x}); 0.95) \equiv \hat{f}(\mathbf{x}) \pm 2\sqrt{\widehat{\mathrm{Var}}(\hat{f}(\mathbf{x}))}.$$

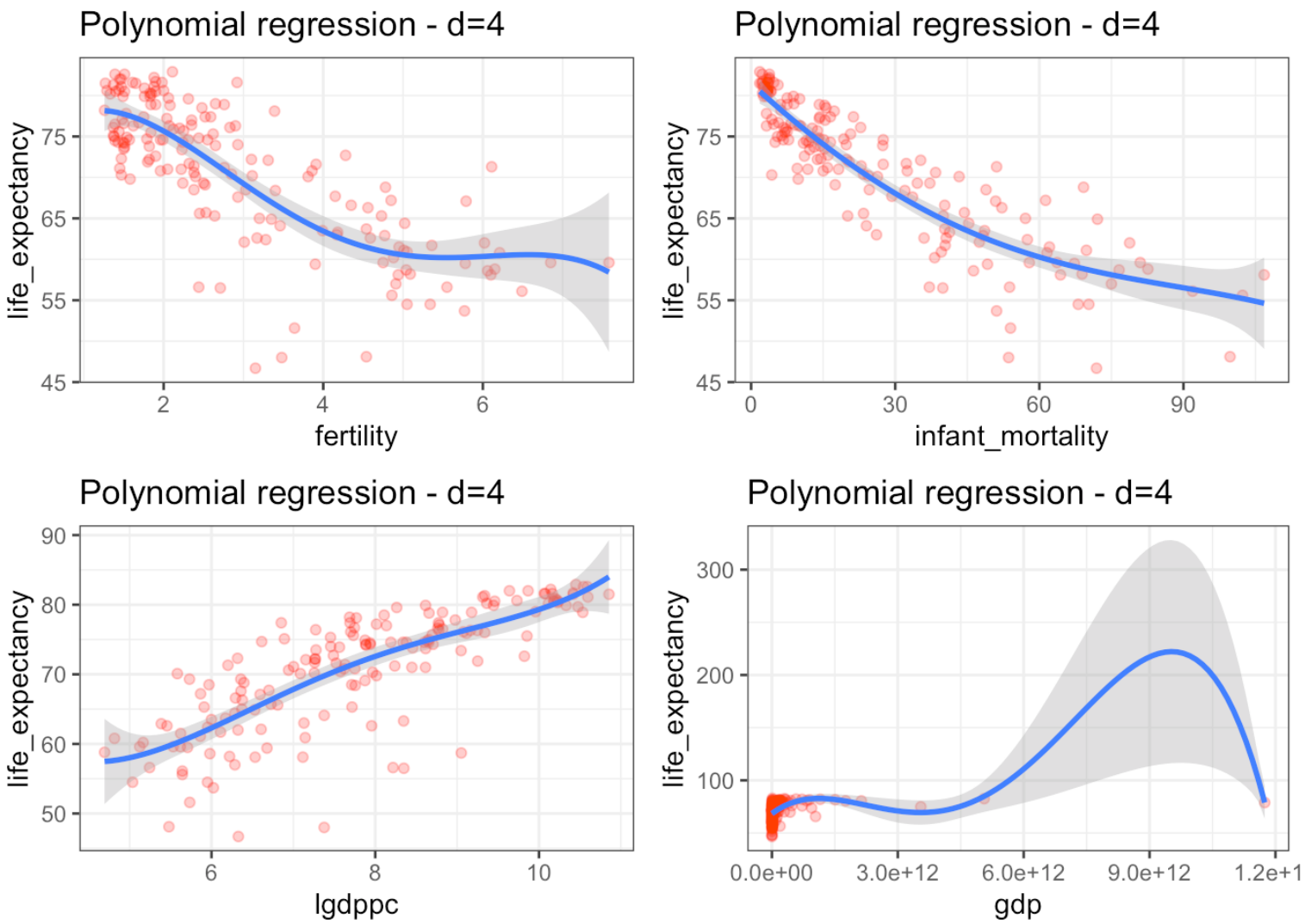
Gapminder Example

We plot **polynomial regressions** ($d = 4$) and **confidence intervals** for life expectancy against 4 different predictors in the 2011 Gapminder data **using the entire set as** Tr (DUDADS, 20.5, *Basis Function Models*).

We can either pick a **reasonable small** d (often below 4) or use cross-validation to select a d that minimizes the estimated $\mathrm{MSE}_{\mathrm{Te}}$ (it is possible to add **multiple predictors** and **interaction terms**).

For polynomials, we have ($|f(\mathbf{x})| \to \infty$ when $\|\mathbf{x}\| \to \infty$): tail behaviour is usually **horrible**.

Use polynomial regression **very carefully**, staying **within the prediction domain** and **centering the predictors** to reduce variance inflation.


Polynomial regression - d=4
life_expectancy
fertility
Polynomial regression - d=4
life_expectancy
infant_mortality
Polynomial regression - d=4
life_expectancy
lgdppc
Polynomial regression - d=4
life_expectancy
gdp

Step Functions

In polynomial regression, we are imposing a **global structure** on the non-linear function $y = f(\mathbf{x})$ that cannot always be justified.

Step functions can be used to keep things "**local**". Let c_i, $i = 1, \ldots, K$ lie in $\text{range}(X)$ and consider the following $K + 1$ new predictors:

$$C_0(X) = \mathcal{I}(X < c_1), \quad C_i(X) = \mathcal{I}(c_i \leq X < c_{i+1}), \; i = 1, \ldots, K-1$$
$$C_K(X) = \mathcal{I}(c_k \leq X),$$

where $\mathcal{I}$ is the **indicator function**

$$\mathcal{I}(\alpha) = \begin{cases} 0, & \alpha \textbf{ is false} \\ 1, & \alpha \textbf{ is true} \end{cases}$$

For any X, $C_0(X) + C_1(X) + \cdots + C_K(X) = 1$, since X lies in **exactly one** of the intervals

$$(-\infty, c_1),\ [c_1, c_2),\ \cdots,\ [c_{K-1}, C_K),\ [C_K, \infty).$$

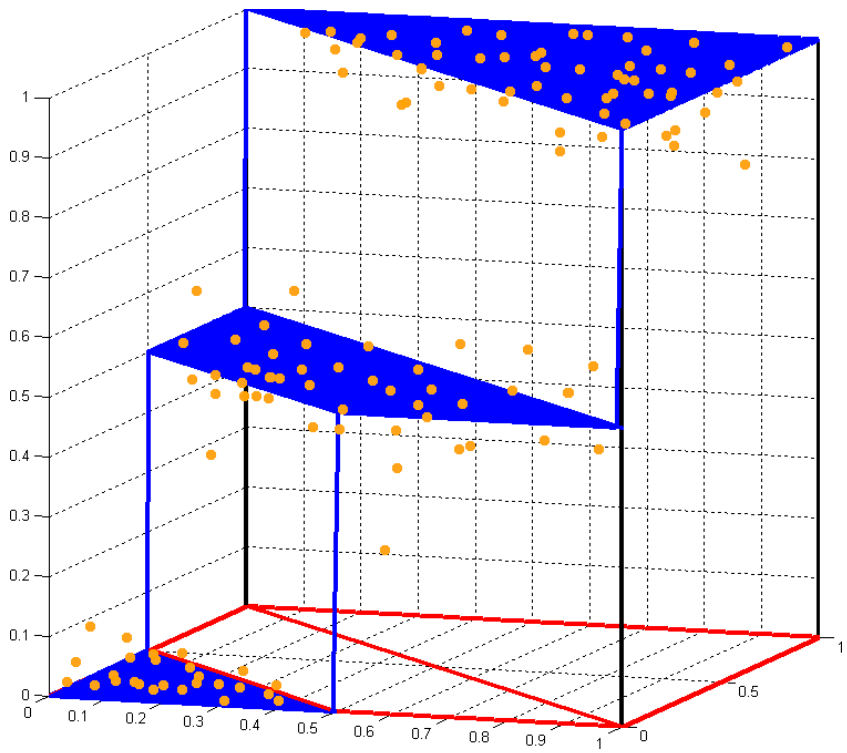
The **step function regression model** is the OLS model

$$y_i = \mathcal{S}(x_i) = \beta_0 + \beta_1 C_1(x_i) + \cdots + \beta_K C_K(x_i) + \varepsilon_i, \quad i = 1, \ldots, N.$$

For a given X, **at most one** of $C_1(X), \ldots, C_K(X)$ is $\neq 0$; thus, when $X < c_1$, we have $C_j(X) = 0$ for all $j = 1, \ldots, K$, and so

$$\beta_0 = \text{Avg}\{Y \mid X < c_1\}.$$

For $X \in [c_j, c_{j+1})$, we have $\hat{y} = \beta_0 + \beta_j$, so β_j represents the **average increase in Y for $[c_j, c_{j+1})$ relative to $(-\infty, c_1)$**.



The step functions do capture the general trends, but how easily interpretable are they?

Challenge: there is no easy way to find the **number** K and select the **position of the breakpoints** $c_1, \ldots, c_K$, unless there are **natural gaps** in the predictors (but see CART, Section 3.6).

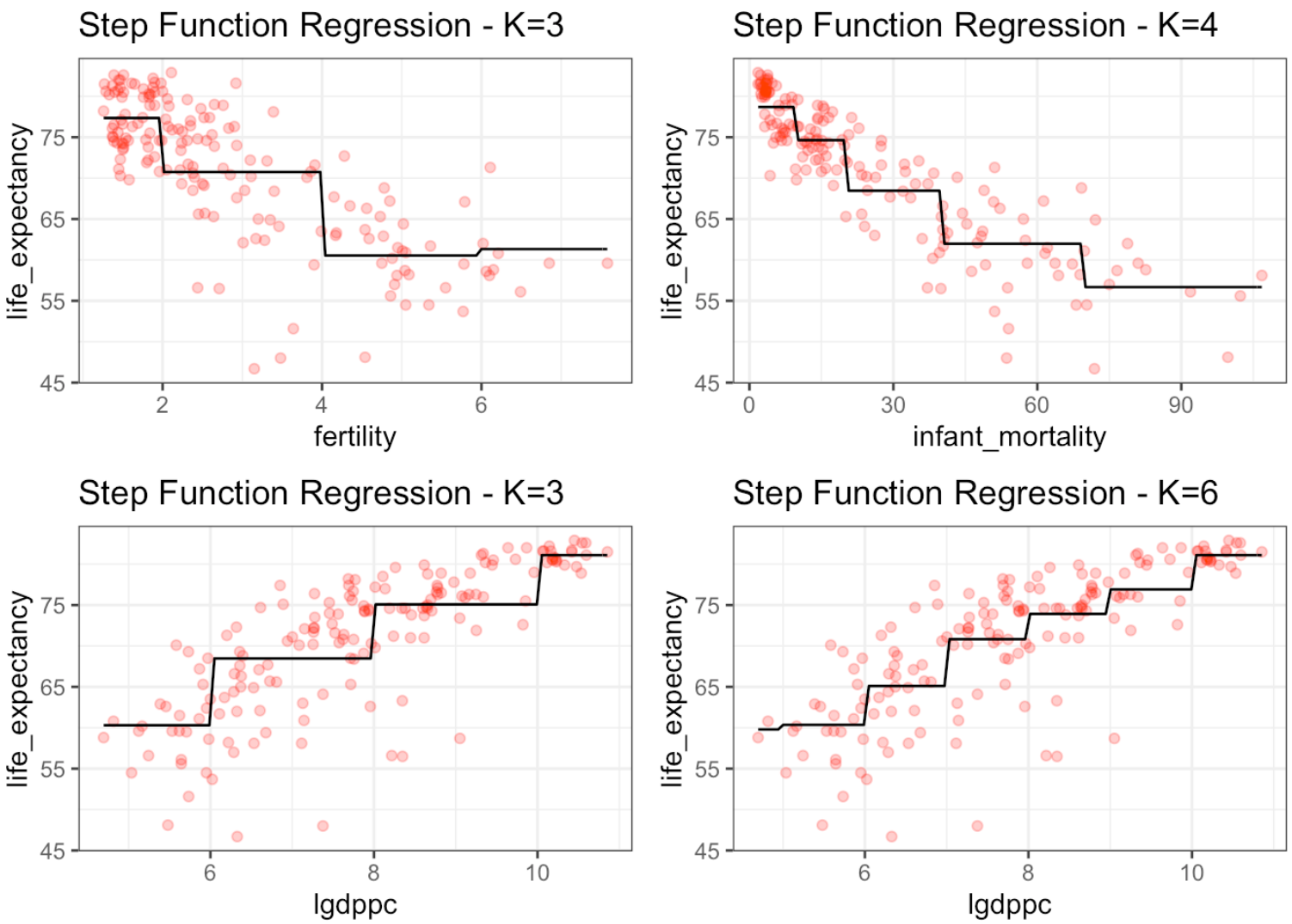
Step function regression or polynomial regression could be achieved in **higher dimensions**: the **basic principles** remain the same, except that the number of parameters **increases drastically**, which can create some **overfitting issues**.

Gapminder Example

We plot **step function regressions** for life expectancy against 3 different predictors in the 2011 Gapminder data **using the entire set as** Tr (DUDADS, 20.5, *Basis Function Models*).

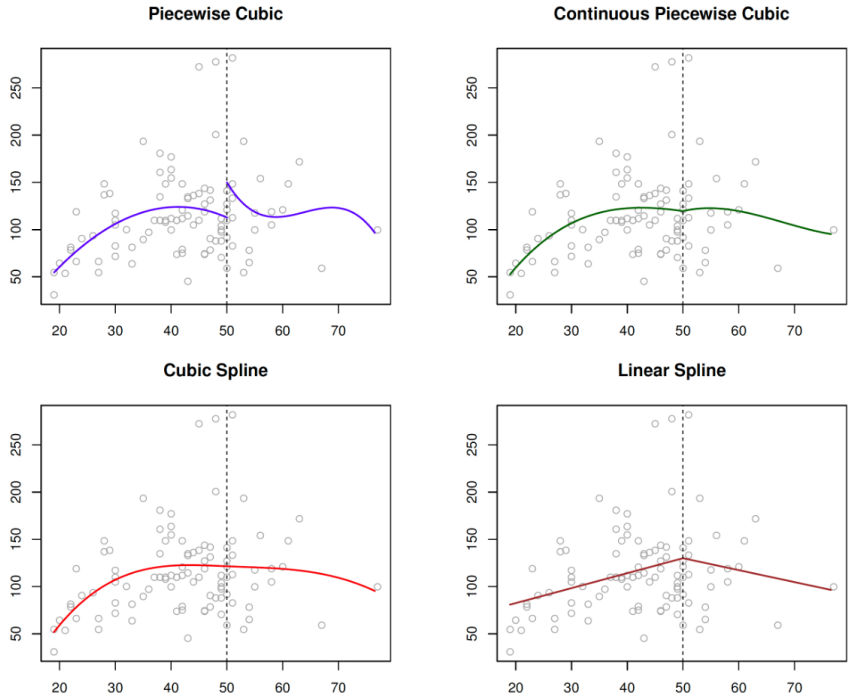
We build:

- a $K = 3$ knots step function model for life expectancy against fertility, using the (arbitrary) knot values at 2, 4, and 6;
- a $K = 4$ knots step function model for life expectancy against infant mortality, using the (arbitrary) knot values 10, 20, 40, 70:
- a $K = 3$ knots step function model for life expectancy against the log of gdp per capita, using the (arbitrary) knot values at 6, 8, 10:
- a $K = 6$ knots step function model for life expectancy against the log of gdp per capita, using the (arbitrary) knot values at 5, 6, 7, 8, 9, 10:


Step Function Regression - K=3
life_expectancy
fertility
Step Function Regression - K=4
life_expectancy
infant_mortality
Step Function Regression - K=3
life_expectancy
lgdppc
Step Function Regression - K=6
life_expectancy
lgdppc

2.6.2 – Splines

We can combine polynomial regression and step functions to obtain a more **flexible curve fitting** approach.



Regression Splines

Instead of fitting a **single polynomial** over $\mathrm{range}(X)$, we can use **multiple simple polynomials** (with $\deg \leq 3$, usually) in **regions** R_k, such as:

$$y_i = \begin{cases} \beta_{0,1} + \beta_{1,1} x_i + \beta_{2,1} x_i^2 + \beta_{3,1} x_i^3 + \varepsilon_i, & \text{if } x_i \in R_1 \\ \beta_{0,2} + \beta_{1,2} x_i + \beta_{2,2} x_i^2 + \beta_{3,2} x_i^3 + \delta_i, & \text{if } x_i \in R_2 \end{cases}$$

Various constraints can be imposed on the polynomials:

- **none**;
- **continuous** at each ∂R_k;
- C^1 **(continuously differentiable)** at each ∂R_k, etc.

In a sense, **splines** have the "**maximum**" amount of continuity; using **more** regions leads to **more flexible** fits. Say we split a 1-D domain into $K+1$ **regions**, bounded by K **knots**.

- **No restriction:** we are trying to fit $K+1$ **piecewise cubic functions** to Tr; we need to estimate 4 parameters per polynomial $\implies$ $4(K+1)$ **effective parameters**.
- C^0 **fit:** the polynomials must **agree at the knots**, which **reduces** the number of **effective parameters** (**simpler** model) $\implies$ $3(K+1)$ effective parameters.
- C^1 **fit:** the **derivatives** must also **agree at the knots**, further **reducing** the model **complexity** $\implies$ $2(K+1)$ effective parameters.
- C^2 **fit:** the **second derivatives** must also **agree at the knots**; the resulting **cubic spline** only has $K+4$ parameters to fit.

Positive part:

$$w_+ = \begin{cases} w & \text{if } w > 0 \\ 0 & \text{else} \end{cases}$$

Formally, a **linear spline** requires $\xi_1, \ldots, \xi_K$ knots and has $K+1$ **effective parameters**. The model can be expressed simply using **positive parts**:

$$y_i = \beta_0 + \beta_1 x_i + \beta_2 (x_i - \xi_1)_+ + \cdots + \beta_{K+1} (x_i - \xi_K)_+ + \varepsilon_i.$$

The **cubic spline** is:

$$y_i = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \beta_3 x_i^3 + \beta_4 (x_i - \xi_1)_+^3 + \cdots + \beta_{K+3} (x_i - \xi_K)_+^3 + \varepsilon_i.$$

The **natural cubic spline** is a **cubic spline** between ξ_1 and ξ_K, with **linear extrapolation** beyond ξ_1 and ξ_K $\implies$ **more** knots for the **same number** of effective parameters as a **linear spline**.

The OLS **machinery** remains available: **predictions**, **diagnostics**, **remedial measures**, **confidence intervals**, and extension to **logistic regression**, as needed.

Gapminder Example

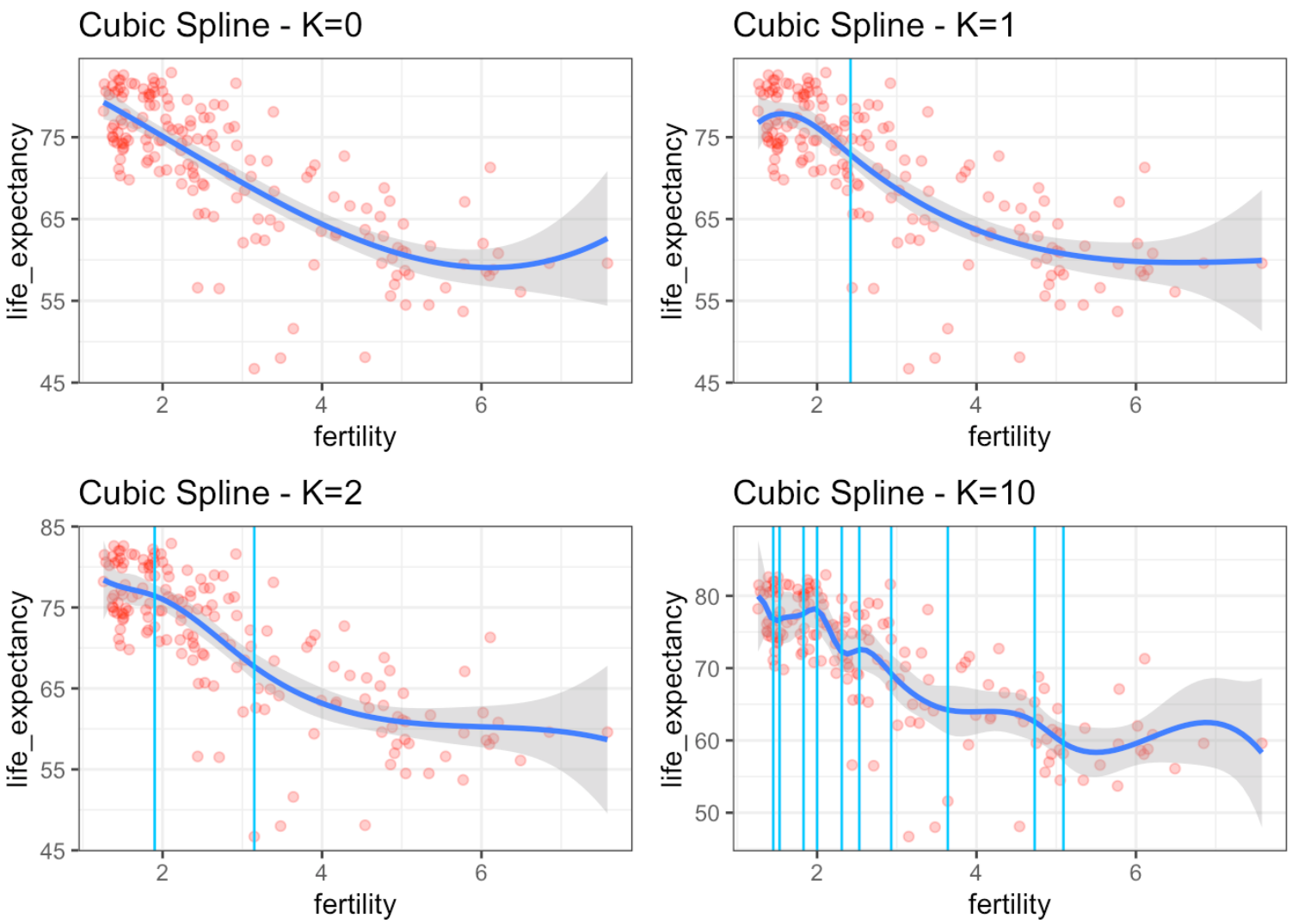
We plot **cubic splines** and **natural cubic splines** for life expectancy against fertility in the 2011 Gapminder data **using the entire set as** Tr (DUDADS, 20.5, $Splines$).

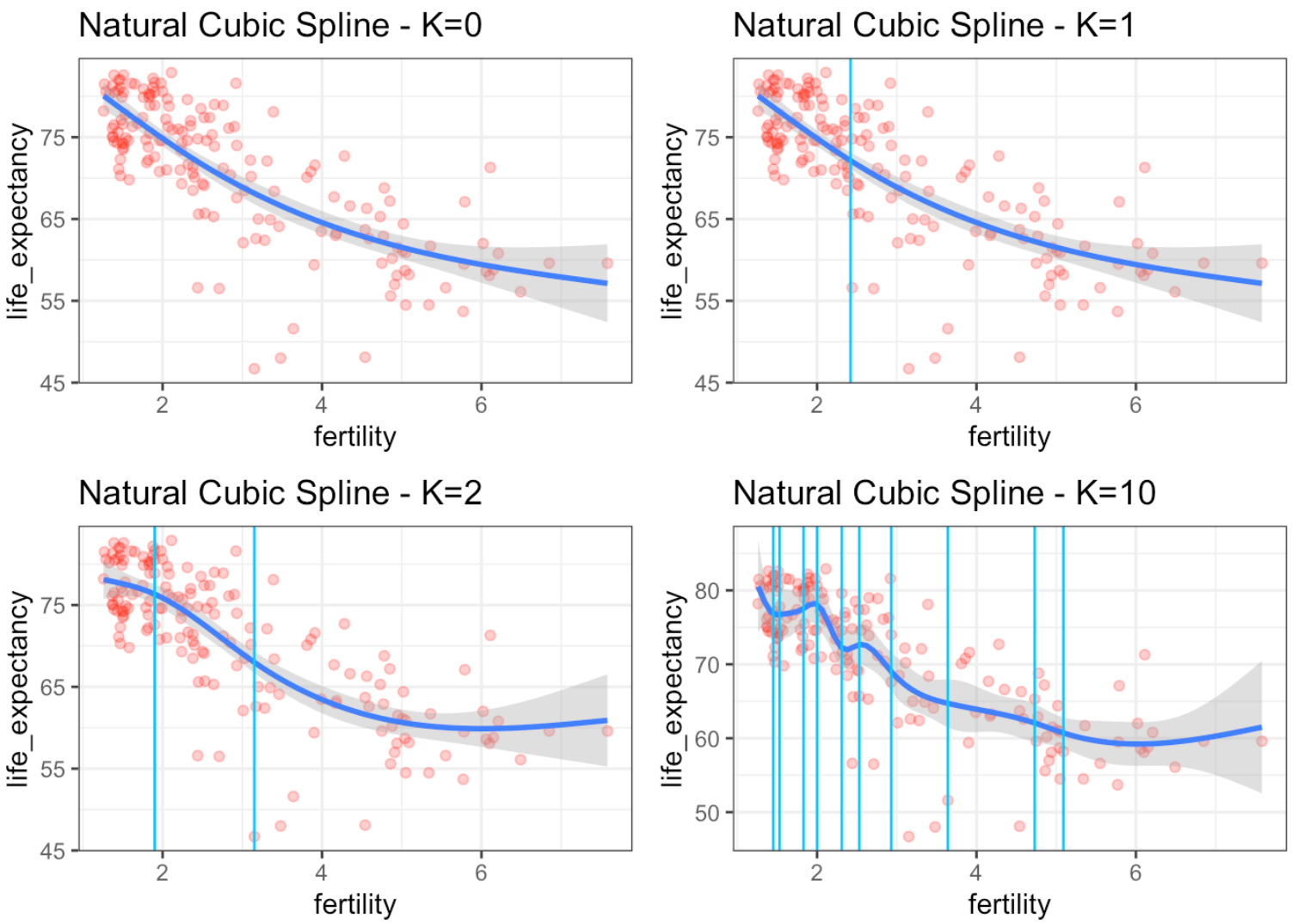
In theory, we place **more** knots in locations where the spline function is thought to **vary more rapidly**, and **fewer** where it is **more stable**.

In practice, the knots are placed uniformly at **quantiles** of X, based on the pre-selected number of knots.

In the charts on the next page, we display cubic splines with $K = 0, 1, 2, 10$ knots (or $3, 4, 5, 13$ **degrees of freedom**).

In the charts two pages hence, we display natural cubic splines with $K = 0, 1, 2, 10$ knots (or $1, 2, 3, 11$ **degrees of freedom**).


Cubic Spline - K=0
Cubic Spline - K=1
Cubic Spline - K=2
Cubic Spline - K=10
life_expectancy
fertility


Natural Cubic Spline - K=0
Natural Cubic Spline - K=1
Natural Cubic Spline - K=2
Natural Cubic Spline - K=10
life_expectancy
fertility
45
55
65
75
50
60
70
80
2
4
6

Do you notice any difference in the **shapes** of the cubic splines vs. those of the natural cubic splines?

How do we determine the optimal K? **Cross-validation**:

1. compute the estimated error for various K (10-fold CV, say)

2. pick the K^* that **minimizes** the error.

Regression splines often provide "**better**" models than polynomial regression because they induce **flexibility** *via* **large** K with low $d \leq 3$, rather than through high d of polynomial regression (and their **wild variability** near the boundaries).

See also: DUDADS, 20.5, *Multivariate Adaptive Regression Splines*.

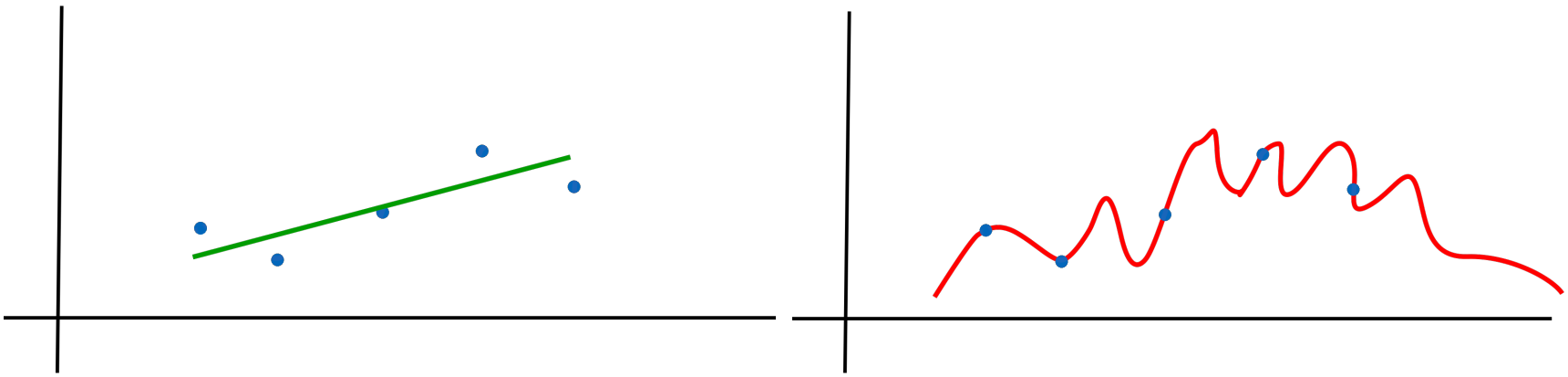
Smoothing Splines

We can use a **mathematical** approach to produce a spline. We seek a function g that provides a **good fit** for Tr; in other words, we are looking for a g for which

$$\mathrm{SSE} = \sum_{i=1}^{N} (y_i - g(x_i))^2 \quad \text{is "}\textbf{small}\text{".}$$

But g must also be **constrained**: any smooth function interpolating **exactly** (x_i, y_i), $i = 1, \ldots, N$ would yield $\mathrm{SSE} = 0$, at the cost of **severe overfitting** and loss of **interpretability**.

The flip side, of course, is that **too many constraints** can result in the data being **underfit**.



A spline with too few constraint overfits the data (right).

The **smoothing spline** approach seeks to solve the following problem:

$$g_\lambda = \arg\min_h \left\{ \underbrace{\sum_{i=1}^{N} (y_i - h(x_i))^2}_{\text{SSE loss}} + \lambda \underbrace{\int_{\Omega(X)} [h''(t)]^2 \, dt}_{\text{penalty term}} \right\},$$

where $\lambda \geq 0$ is a **tuning parameter**; $\Omega(X)$ is the **range** of the predictor X.

The **penalty term** measures the **roughness** of the spline function h – if h is "**wiggly**", the penalty is **large**, and $vice\text{-}versa$; if h represents a straight line, the penalty term is 0.

When $\lambda \to 0$, the penalty term has **little effect**, so we expect g_λ to be "**jumpy**" leading to smooth functions that interpolate the observations exactly ($\Longrightarrow$ **overfitting**).

When $\lambda \to \infty$, the penalty term **dominates** and $\int [g''_\lambda(t)]^2 dt \to 0$ over $\Omega(X)$; thus $g_\lambda \to$ OLS solution over $\Omega(X)$ ($\Longrightarrow$ **underfitting**).

The tuning parameter λ controls the **bias-variance trade-off**, expressed as a tussle between **rigidity** and **model complexity**.

Notice the **structural link** between the **bias-variance trade-off**, the concept of **regularization**, and **smoothing splines**?

The **optimal smoothing spline** g_λ is a **natural cubic spline** with a knot at **every** data point $\xi_i = x_i$, $i = 1, \ldots, N$, with continuous 0th, 1st, and 2nd derivatives throughout the range $\Omega(X) = [\min \xi_i, \max \xi_i]$, and is **linear** outside $\Omega(X)$.

IMPORTANT: g_λ **is not the natural cubic spline one that would be obtained from the regression spline approach**, as g_λ also depends on the **turning parameter** λ.

What is the **best choice** for λ? At first glance, this would seem to be another job for **cross-validation**.

There is another option: we can specify the smoothing spline through the **effective degrees of freedom**, which decrease from N to 2 as λ goes from 0 to ∞. There might be **external requirements** for these to take on a **pre-specified** value.

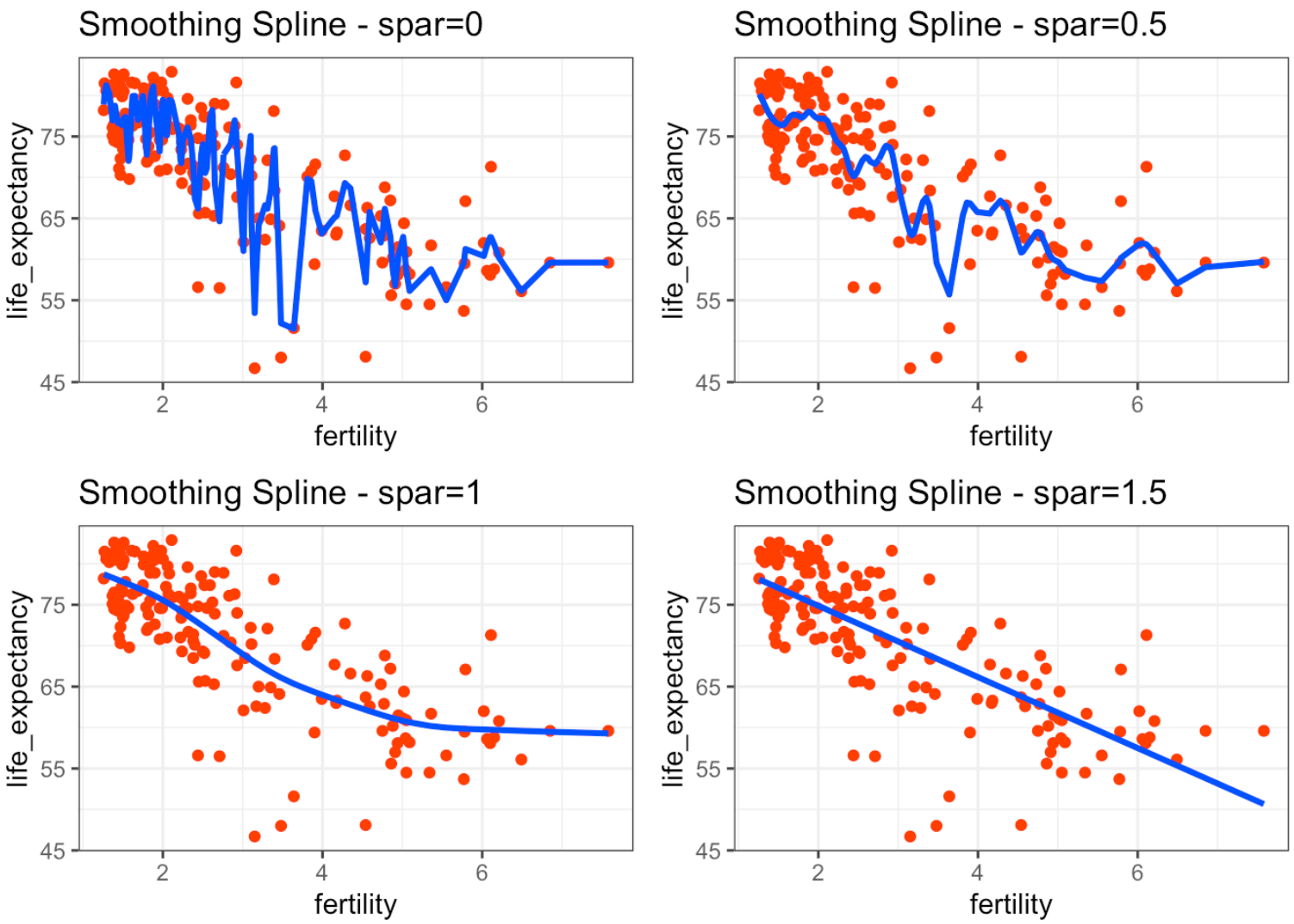
Gapminder Example

We plot **smoothing splines** for life expectancy against fertility in the 2011 Gapminder data **using the entire set as** Tr (DUDADS, 20.5, $Splines$).

We use 4 different smoothing parameter values: note the evolution of a **flexible** but highly **non-interpretable** model (the wiggly curve associated to `spar=0`, top left) into to a rigid but highly interpretable model (the line associated to `spar=1.5`, bottom right) as the `spar` values increase.

Other smoothing procedures also exist: **moving average**, **local regression** (LOESS, LOWESS), etc. – it pays to collect many arrows for your quiver.

It is also possible to conduct smoothing over $p-$dimensional predictor spaces; the basic ideas are pretty much the same, but the implementation is a tad more complex.


Smoothing Spline - spar=0
Smoothing Spline - spar=0.5
Smoothing Spline - spar=1
Smoothing Spline - spar=1.5
life_expectancy
fertility
45
55
65
75
2
4
6

2.6.3 – Generalized Additive Models

Polynomial regression and splines can apply to **predictor sets** $\{X_1, \ldots, X_p\}$, but they are **best-suited** to predict a response Y from a **single predictor** X (the model complexity **quickly increases** with **more than one** predictor).

Generalized additive models (GAM) allow for **flexible non-linearities in several variables** while retaining the **additive structure** of linear models:

$$y_i = \beta_0 + f_1(x_{i,1}) + \cdots + f_p(x_{i,p}) + \varepsilon_i, \quad i = 1, \ldots, N$$

where each of the f_j can be derived using any of the methods previously discussed.

GAM provide a compromise between **OLS** and fully **non-parametric** models.

For instance, it may be that

$$f_1(x_i) = \beta_{1,1} b_{1,1}(x_{i,1}) + \cdots + \beta_{1,L_1} b_{1,L_1}(x_{i,1})$$
$$\vdots$$
$$f_p(x_i) = \beta_{p,1} b_{p,1}(x_{i,p-1}) + \cdots + \beta_{p,L_p} b_{p,L_p}(x_{i,p-1}),$$

say, we would fit the data using OLS (but this cannot be done if one of the components is a **smoothing spline**, for instance, or if it is **non-linear** in some other way).

In practice, using **natural cubic splines** for the quantitative components seem to work as well as smoothing spline, when it comes to making predictions.

GAM can also be used for **classification** via log-odds (see Chapter 3).

A typical GAM call in R might look like:

```
mgcv::gam(y ~ s(x1,df=5) +
              lo(x2,spar=0.5) +
              bs(x3,df=4) +
              ns(x4,df=5):ns(x5,df=5) +
              x6, data=dat)
```

which would indicate that the contribution of:

- x_1 is given by **smoothing spline** with 5 degrees of freedom,
- x_2 is given by a **local regression** with `spar`$=0.5$,
- x_3 is given by a **cubic spline** with 4 degrees of freedom,
- the fourth component is an **interaction term** based on **natural splines** for x_4 and x_5 (each with 5 degrees of freedom), and
- x_6 is **directly** added to the model.

Advantages:

- can fit a **non-linear** f_j to each predictor $X_j \implies$ may capture trends missed by OLS;
- can reduce the number of **data transformations** to try out **manually** on each X_j;
- non-linear fits may improve **accuracy of predictions** for the response Y;
- useful for inference due to their **additivity** – the effect of X_j on Y (while keeping other predictors fixed) can be analyzed separately;
- the overall **smoothness** of the model can be summarized *via* **effective degrees of freedom/parameters**.

Disadvantages:

- may still be affected by the **curse of dimensionality**;
- restricted to **additive** models – **interaction terms** can be added manually by introducing new predictors $X_j \times X_k$, as can interaction functions $f_{j,k}(X_j, X_k)$ (using local regression or MARS, say), but they quickly get out of hand (again, **CoD**).

Gapminder Example

We plot the **individual contributions** of fertility, infant mortality, GDP, and continental membership to life expectancy in the 2011 Gapminder data **using the entire set as** Tr (DUDADS, 20.5, GAM).

```
Parametric coefficients:
                  Estimate Std. Error t value Pr(>|t|)
(Intercept)        68.1186     0.7470  91.190  < 2e-16 ***
continentAmericas   4.4787     1.1161   4.013 9.30e-05 ***
continentAsia       4.7110     0.9993   4.714 5.35e-06 ***
continentEurope     3.4179     1.3209   2.588   0.0106 *
continentOceania   -0.2891     1.3798  -0.210   0.8343
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

We see in the outcome that the intercept is $\beta_0 = 68.1186$; life expectancy predictions take the form

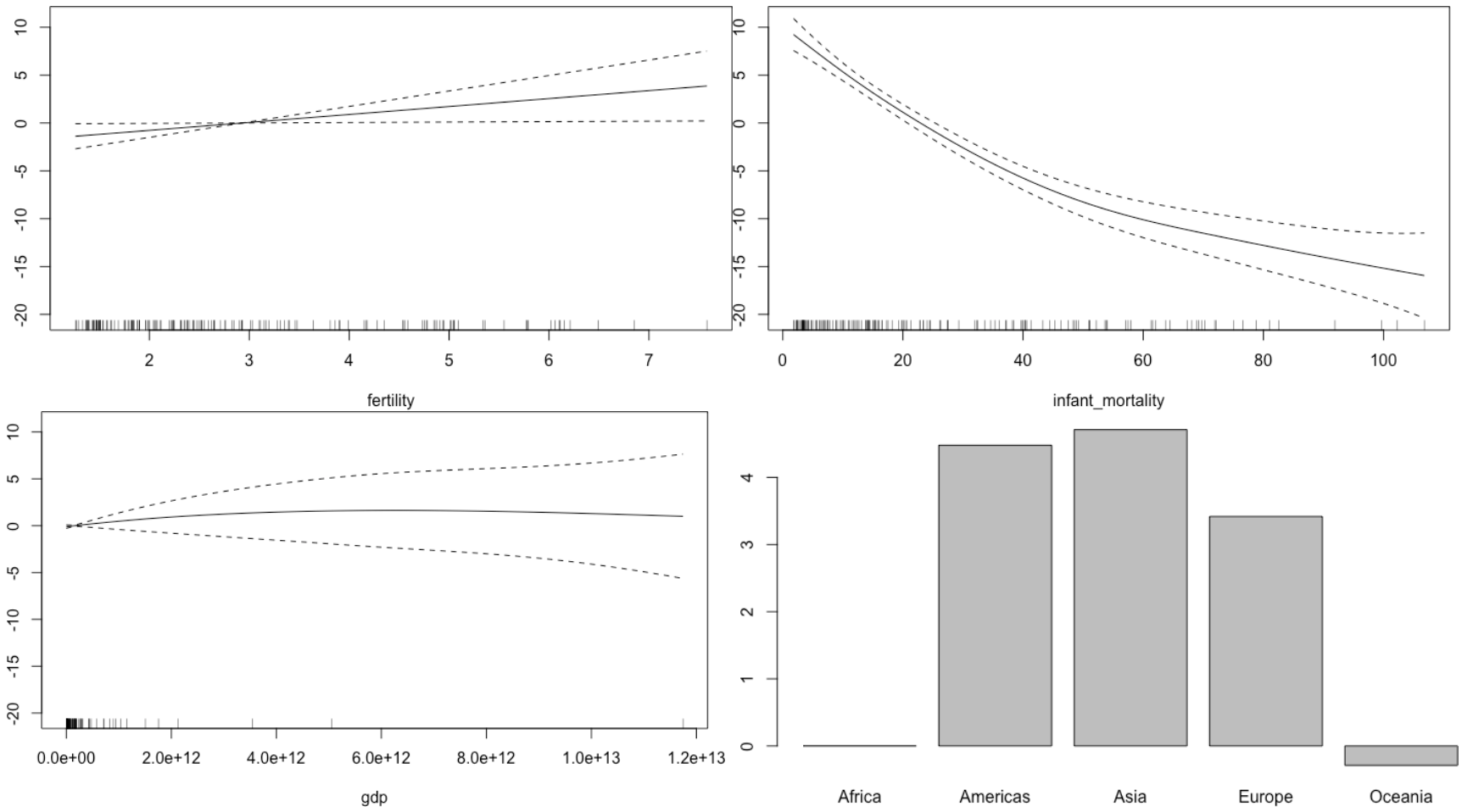
$$\text{life expectancy} \approx \beta_0 + f_1(\text{fertility}) + f_2(\text{infant mortality}) + f_3(\text{gdp}) + \beta_{\text{continent}}.$$

For instance, the life expectancy for:

- an **American** country with
- fertility $= 3$, infant mortality $= 1$, and GDP $= 6 \times 10^{12}$

would be approximately

$$68.1 + 0 + 10 + 2 + 4.5 = 84.6.$$


10
5
0
-5
-10
-15
-20
2
3
4
5
6
7
0
20
40
60
80
100
fertility
0.0e+00
2.0e+12
4.0e+12
6.0e+12
8.0e+12
1.0e+13
1.2e+13
gdp
infant_mortality
0
1
2
3
4
Africa
Americas
Asia
Europe
Oceania

The Rest of the RVE Picture

Something about the algae blooms example? What else?

References

T.Hastie, R.Tibshirani, and J.Friedman, The Elements of Statistical Learning: Data Mining, Inference, and Prediction, 2nd ed. Springer, 2008.

G.James, D.Witten, T.Hastie, and R.Tibshirani, An Introduction to Statistical Learning: With Applications in R. Springer, 2014.

C.C.Aggarwal and C.K.Reddy, Eds., Data Clustering: Algorithms and Applications. CRC Press, 2014.

C.C.Aggarwal, Data Mining: The Textbook. Springer, 2015.

B.Boehmke and B.Greenwell, Hands on Machine Learning with R. CRC Press.

H.Rosling, The health and wealth of nations. Gapminder Foundation, 2012.

D.Robinson, "What's the difference between data science, machine learning, and artificial intelligence?". Variance Explained, Jan. 2018.

M.H.Kutner, C.J.Nachtsheim, J.Neter, and W.Li, Applied Linear Statistical Models. McGraw Hill Irwin, 2004.

G.E.P.Box, "Use and abuse of regression," Journal of Technometrics, vol. 8, no. 4, pp. 625–629, Nov. 1966.

T.Hastie, R.Tibshirani, and M.Wainwright, Statistical Learning with Sparsity : the LASSO and Generalizations. CRC Press, 2015.

L.Torgo, Data Mining with R, 2nd ed. CRC Press, 2016.