Image analysis

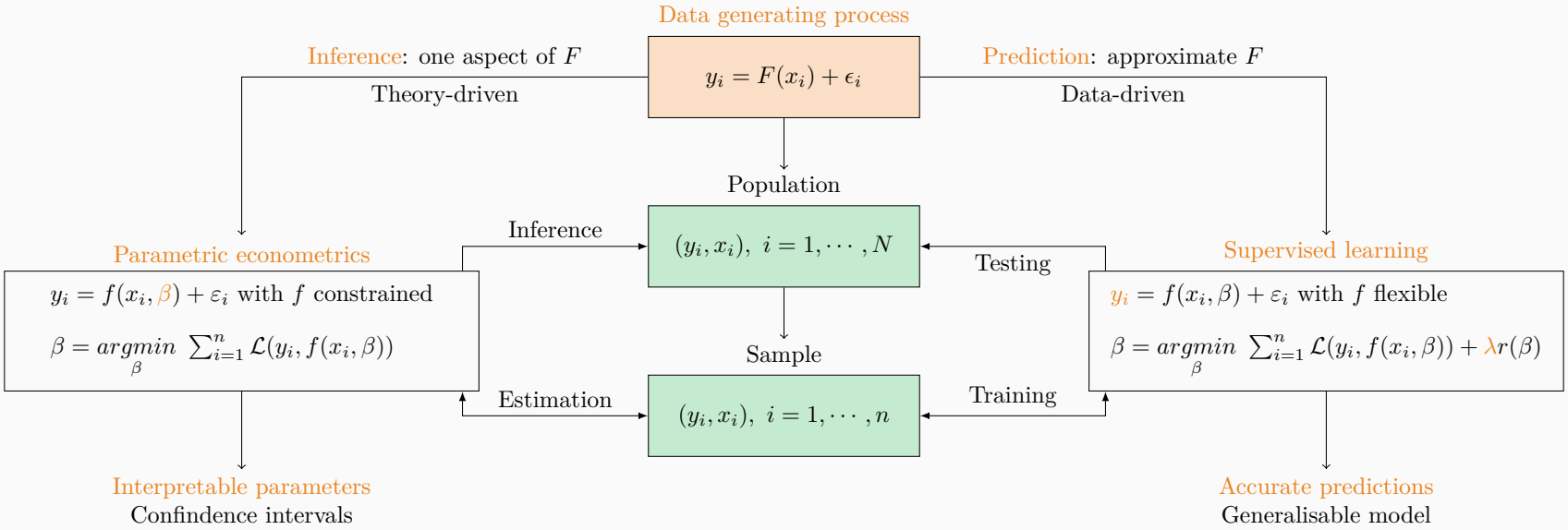
1. Predictive modelling

Clément Gorin
clement.gorin@univ-paris1.fr

Sorbonne School of Economics
Masters in Development Economics

Overview

This lecture introduces supervised learning models and their connection with traditional econometrics



We provide a conceptual framework to understand the similarities and differences between these approaches

Framework

Assume an unknown data generating process F that maps an input x_i to an output y_i with additive error ϵ_i

$$y_i = F(x_i) + \epsilon_i \tag{1}$$
$$i = 1, \ldots, N$$

with a population of N observable (x_i, y_i) pairs "generated" by the this function

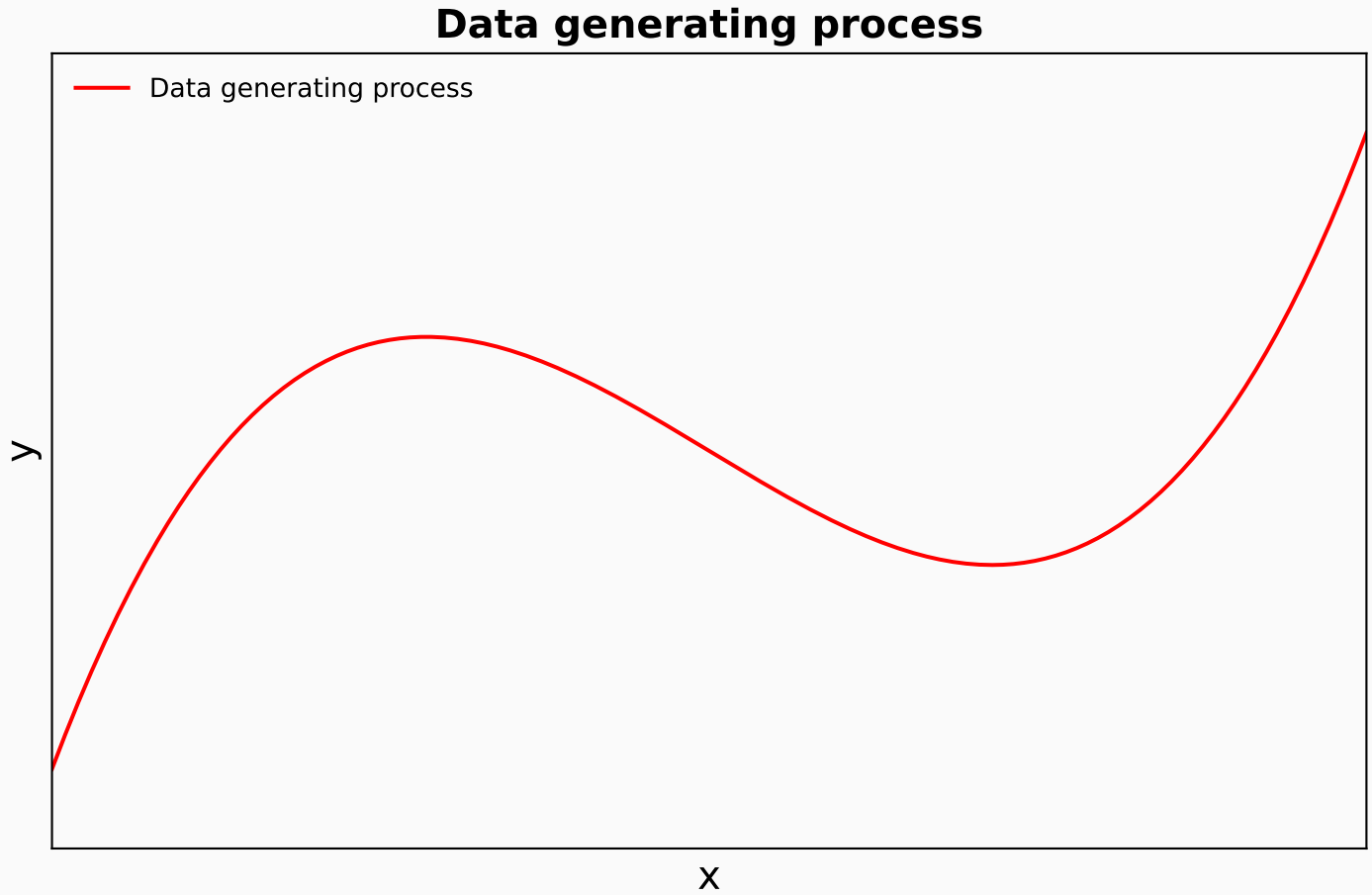
- Many tasks can be formulated this way e.g. estimating one aspect or the entire data-generating process
- E.g. x_i may be the pixel intensities in an image or the characters in a text and y_i a probability

To illustrate the methods, let's simulate a data-generating process and some data

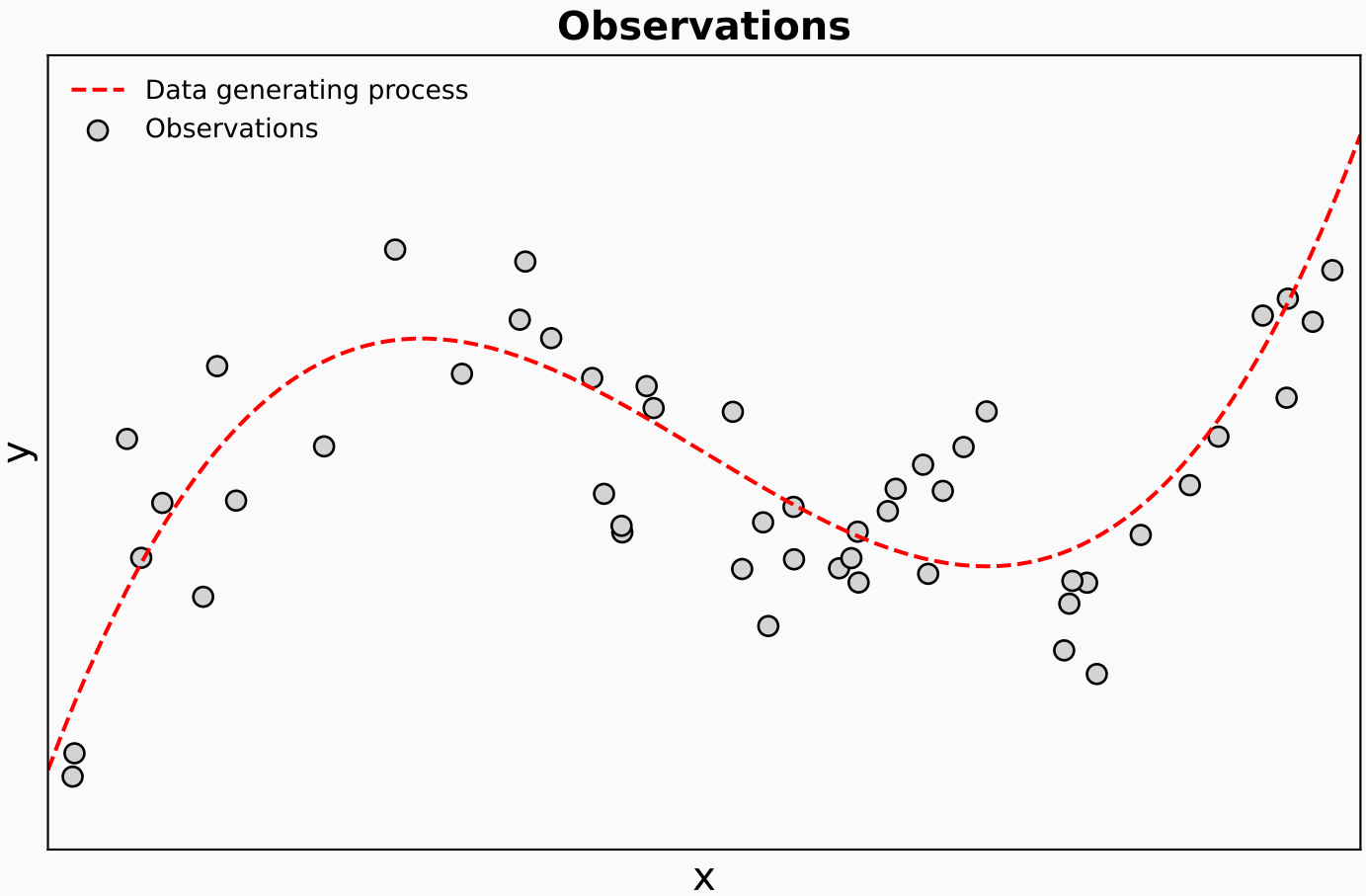
Target function

```
import numpy as np

# DGP
def F(x): return x**3 - x * 5

# Data
np.random.seed(0) # For reproductibility
n = 50
x = np.random.uniform(-3, 3, n)
e = np.random.normal(0, 3, n)

y = F(x) + e
```

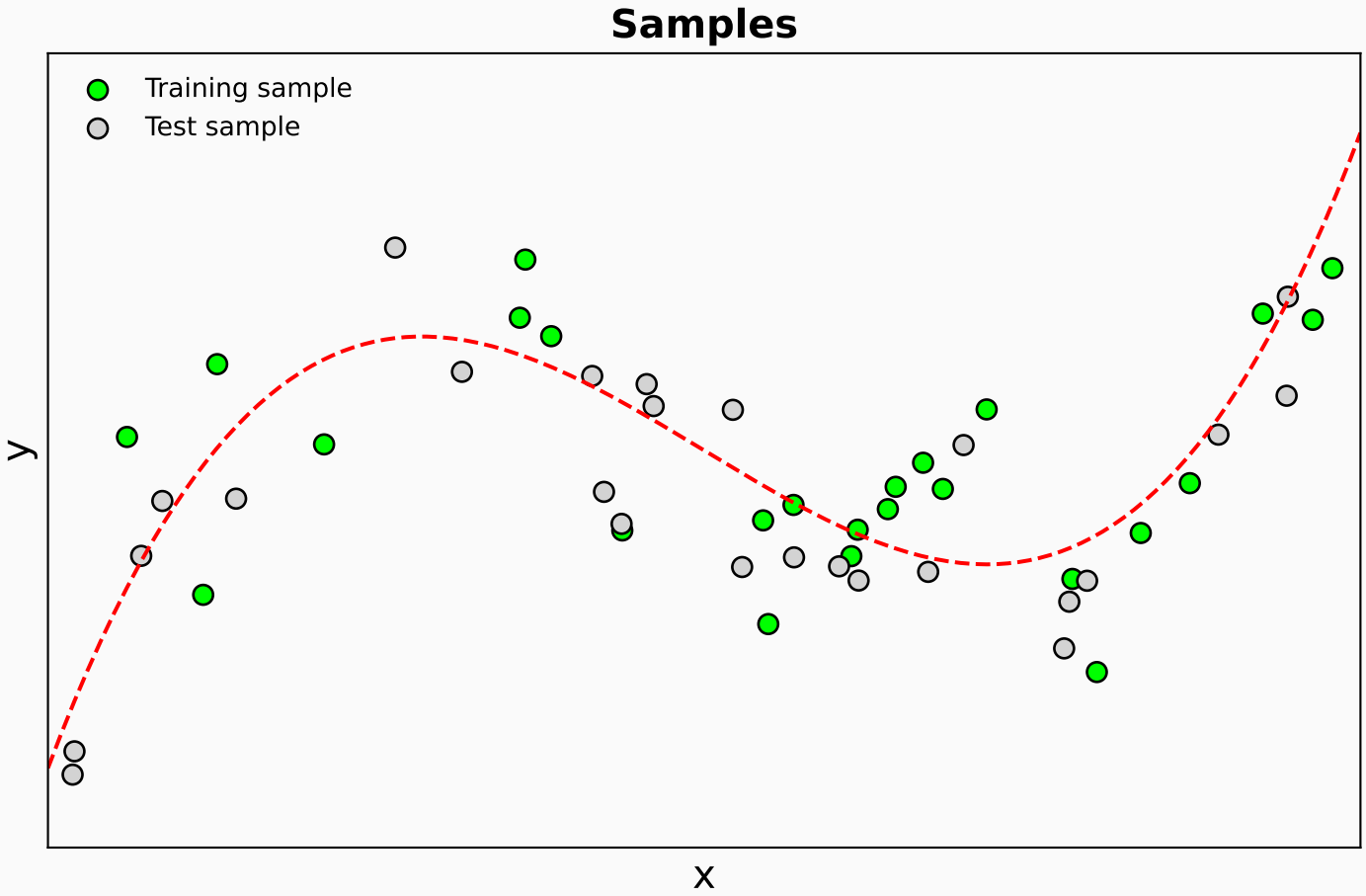
Consider a data generating process of interest e.g. relationship between income and education, or nightlights



This process is not observable in practice, we want to learn about it using some of the observable data points



We can only observe a representative sample from that population called the training sample



▸ Random sample

In practice, there are two reasons why we may want to compute an estimate $\hat{F}$ for this function

- To interpret the relationship between x_i and $y_i \rightarrow$ focus on estimated parameters $\hat{\beta}$ and explanatory power
- To predict accurately y_i using $x_i \rightarrow$ focus on estimated output $\hat{y}_i$ and predictive power (out-of-sample)
- Parametric econometrics is used for the former while the latter is best solved using supervised learning

There is an infinity of functions passing through the observed data points so function approximation is unsolvable

$$y_i = f(x_i, \beta) + \varepsilon_i \qquad (2)$$
$$i = 1, \ldots, n$$

where f is an empirical model, β a set of parameters, ε_i the empirical error and n the number of observations

- We constrain the problem by restricting the search space to a parametric family of functions
- The model structure incorporates those contraints and depend on the application and prior knowledge

Empirical model (B-Spline)

```
1  import pasty
2  from statsmodels import api
3
4  class BSpline:
5
6      def __init__(self, d:int, df:int, alpha:float=0.05):
7          self.d     = d
8          self.df    = df
9          self.alpha = alpha
10         self.model = None
11
12     def transform(self, x:np.ndarray, d:int, df:int) -> np.ndarray:
13         X = patsy.dmatrix(f'bs(x, degree={d}, df={df})', {'x': x})
14         return X
15
16     def fit(self, x:np.ndarray, y:np.ndarray) -> None:
17         X = self.transform(x, d=self.d, df=self.df)
18         self.model = api.OLS(y, X).fit()
19
20     def predict(self, x:np.ndarray) -> np.ndarray:
21         X  = self.transform(x, d=self.d, df=self.df)
22         yh = self.model.get_prediction(X)
23         yh = yh.summary_frame(alpha=self.alpha)
24         return yh
```

Within this parameter-space, optimisation searches for the values $\hat{\beta}$ that minimise a loss function over the training sample

$$\hat{\beta} = \underset{\beta}{argmin} \sum_{i=1}^{n} \mathcal{L}\left(y_i, f(x_i, \beta)\right) \tag{3}$$

$$i = 1, \ldots, n$$

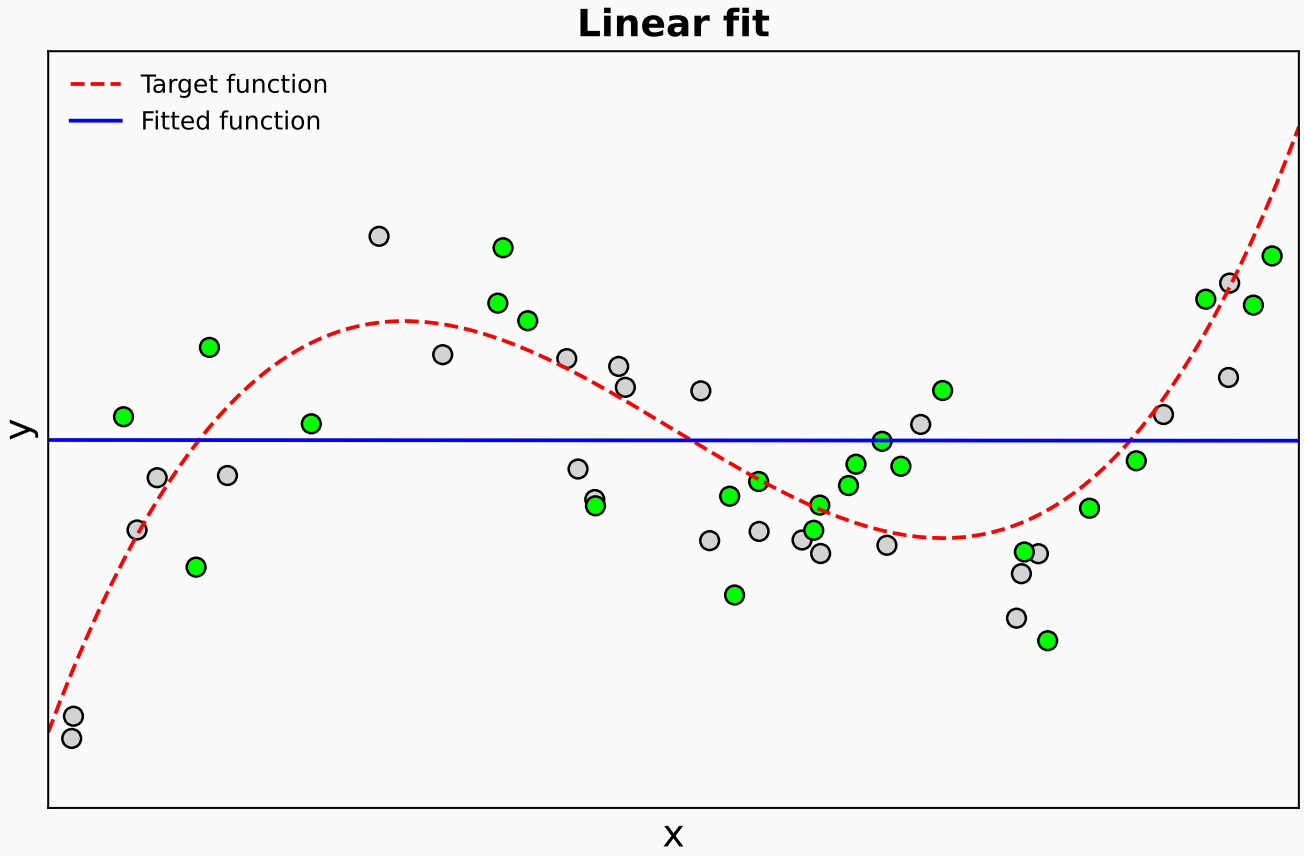
where $\mathcal{L}$ is a loss function, a measure of distance between the observed y_i and estimated $\hat{y}_i = f(x_i, \hat{\beta})$

- The form of the loss function depends on the distribution of the output e.g. continuous, categorical, counts
- Optimisation may have an analytical solution (e.g. OLS) or require iterative routines (e.g. maximum likelihood)

▸ Spline model

To compute interpretable parameters estimates, we impose strong prior assumptions on the model

- We must define which inputs enter the model and their functional relationship with the output
  - → **Additivity**: separate interpretation e.g. *ceteris paribus*
  - → **Linearity**: simple interpretation e.g. unit increase
  - → **Transformations**: interpretable ones e.g. logs, interaction
- This approach is sensible for low-dimensional problems with an established theoretical background

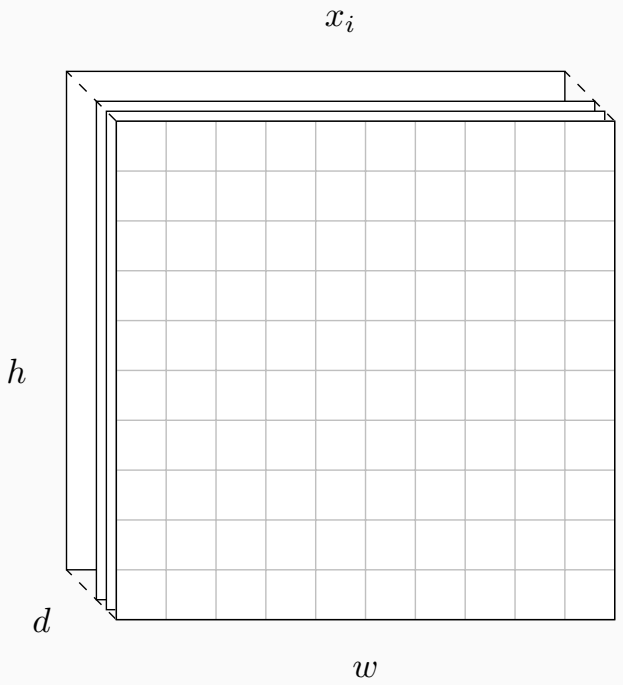



```
model = BSpline(d=1, df=1)
model.fit(x_train, y_train)
yh_test = model.predict(x_test)
```

Under restrictive conditions on the form of F and the conditional distribution of ε e.g. $\varepsilon_i|x \sim \mathcal{N}(0, \sigma^2)$

- Parametric econometrics deliver interpretable and meaningful average parameter estimates
- Statistical inference provides confidence intervals to establish generality beyond the sample data
- The optimisation has a closed-form solution (e.g OLS) or can be solved efficiently (e.g Newton-Raphson)

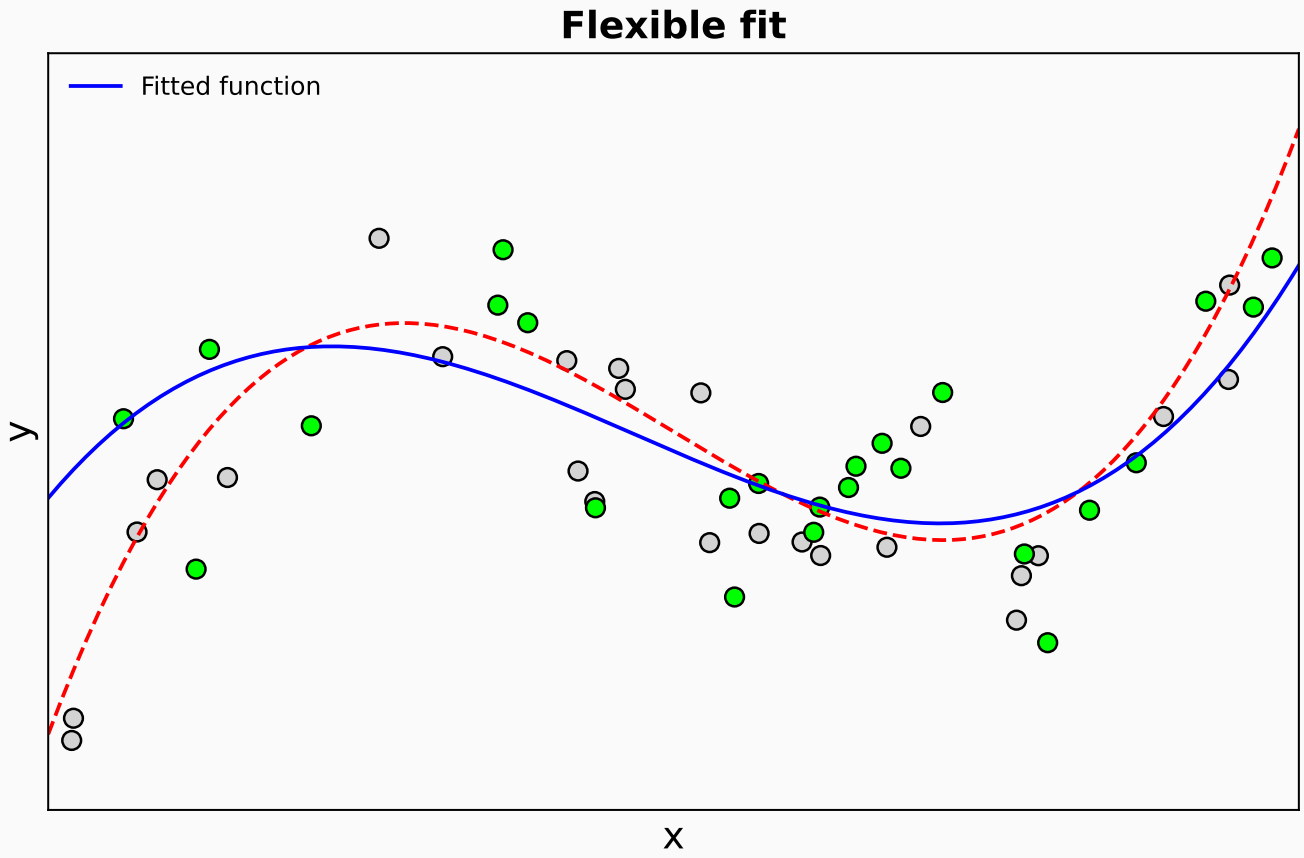
Assume we predict the probability that an image contains a house, much like a logistic regression



An image is represented as an array of pixel intensities with dimensions $h \times w \times d$ e.g. $1024 \times 1024 \times 3$

For these tasks parameter interpretability is not a strong requirement and flexible methods are available

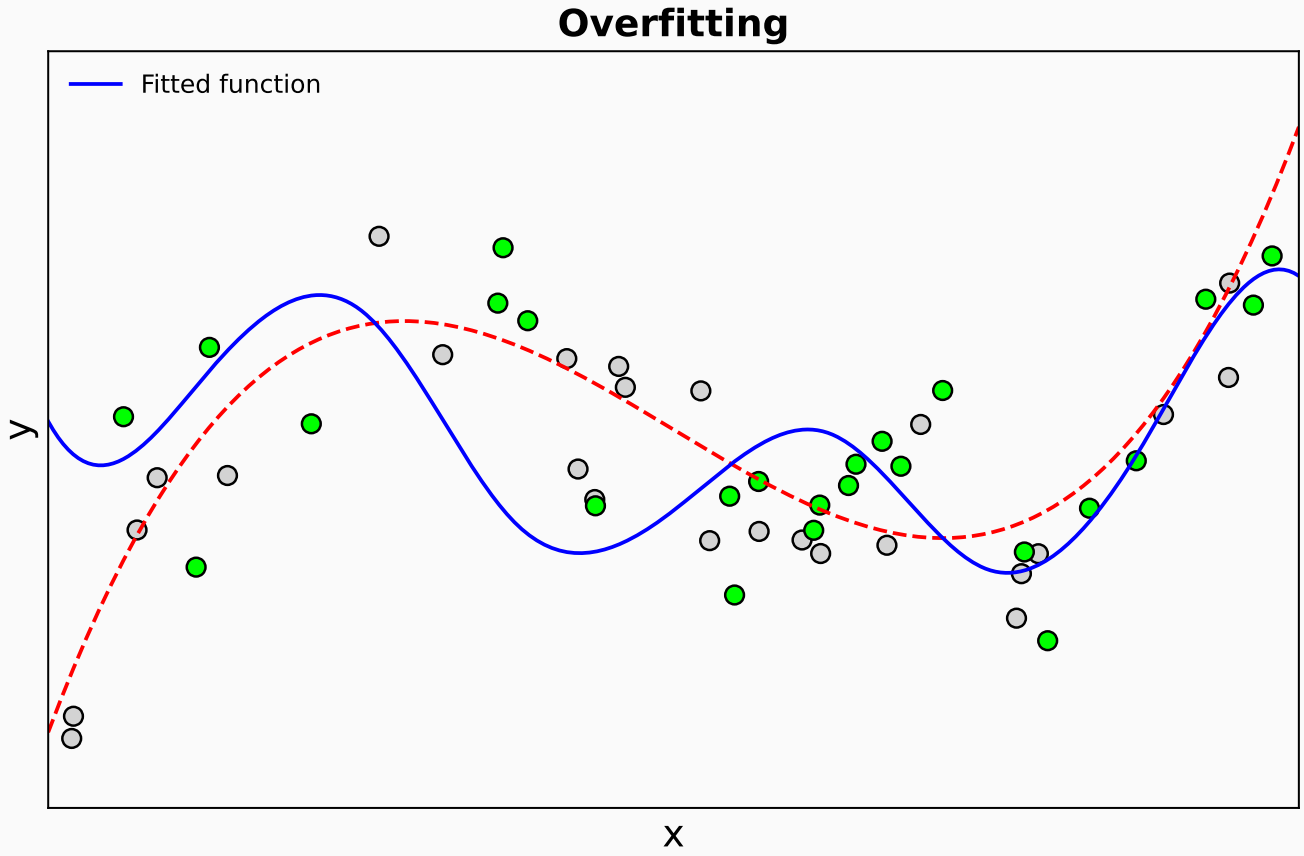
- The target function is so complex that we cannot devise a sensible parameteric model
  - → **Dimensionality** of the image
  - → **Interactions** among pixels
  - → **Non-linearities**
- Supervised learning builds an empirical model directly from the sample data using many parameters
- The model learns the target function when it approximates it with sufficient accuracy and generality




```
model = BSpline(d=3, df=3)
model.fit(x_train, y_train)
yh_test = model.predict(x_test)
```

For predictive applications, the model must be assessed using observations outside the (training) sample

- Flexible methods approximate not only the target function but also the observational error (i.e. overfitting)
- The training sample loss provides a biased estimate of predictive performance (i.e. tends toward 0)
- The model is assessed on another random sample that has not been used for training i.e. test sample




```
model = BSpline(d=3, df=10)
model.fit(x_train, y_train)
yh_test = model.predict(x_test)
```

During optimisation, we minimise the training sample error but we really care about the test sample error

- The empirical model must be flexible enough to approximate complex functions (i.e. low bias)
- However the estimated function must also generalise to out-of sample observations (i.e. low variance)
- This fundamental trade-off is addressed by adding a regularisation term to the loss function

A regularisation term is added to the loss to encourage more general models. Equation (3) becomes

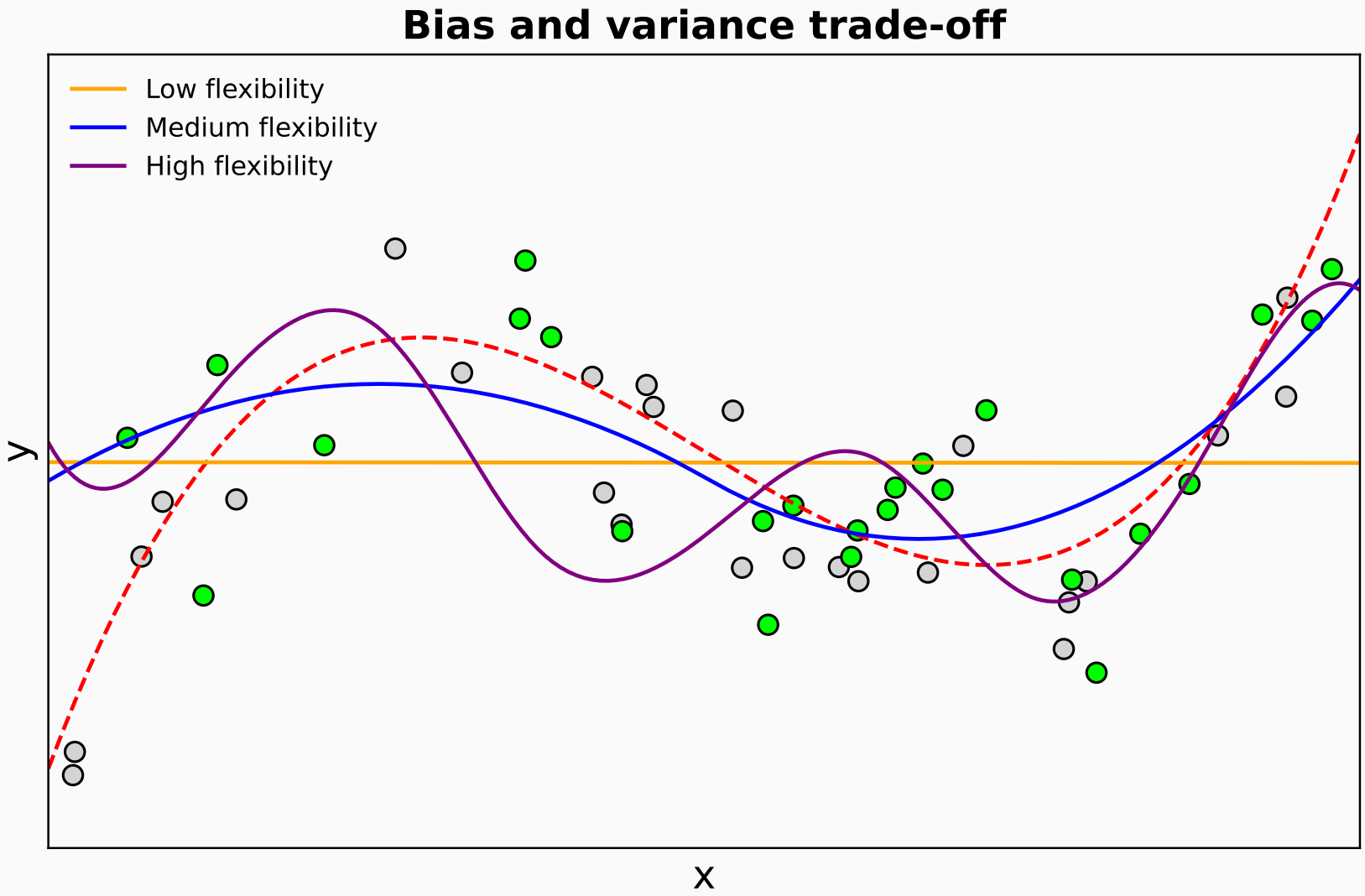
$$\hat{\beta} = \underset{\beta}{argmin} \sum_{i=1}^{n} \mathcal{L}\left(y_i, f(x_i, \beta)\right) + \lambda r(\beta) \tag{4}$$

$$i = 1, \ldots, n$$

where r is a regularisation function that increases with large parameter values and λ is the regularisation parameter

- Large parameter values allow the estimated function to change its behaviour over a small space
- The relative strength of the two counteracting effects is governed by the tuning parameter λ

Bias and Variance

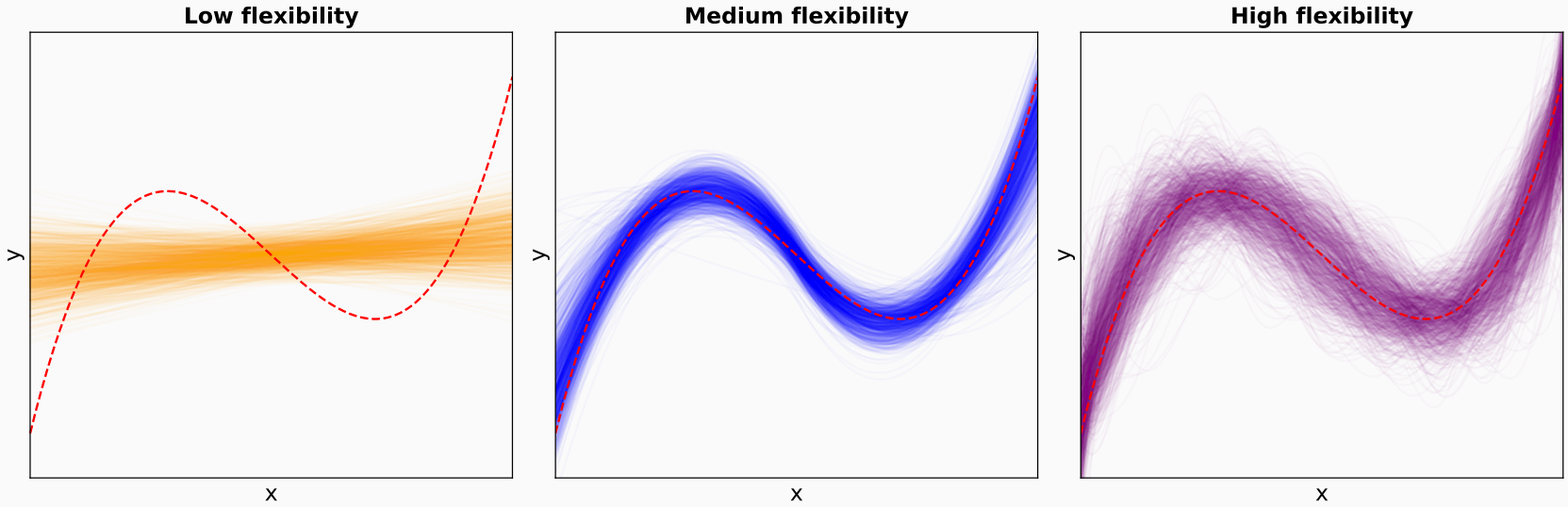

Bias and variance trade-off
Low flexibility
Medium flexibility
High flexibility
y
x

To produce accurate preditions our model must minimise simultaneously prediction bias and variance

- These statistics refer to the distribution of $\hat{y}$ when the same model is estimated on different random samples
- Estimators can be considered as random variables with the estimated output being its realisation
- For predictions, we consider the integrated effect of bias and variance over the entire function space

▸ Random variable

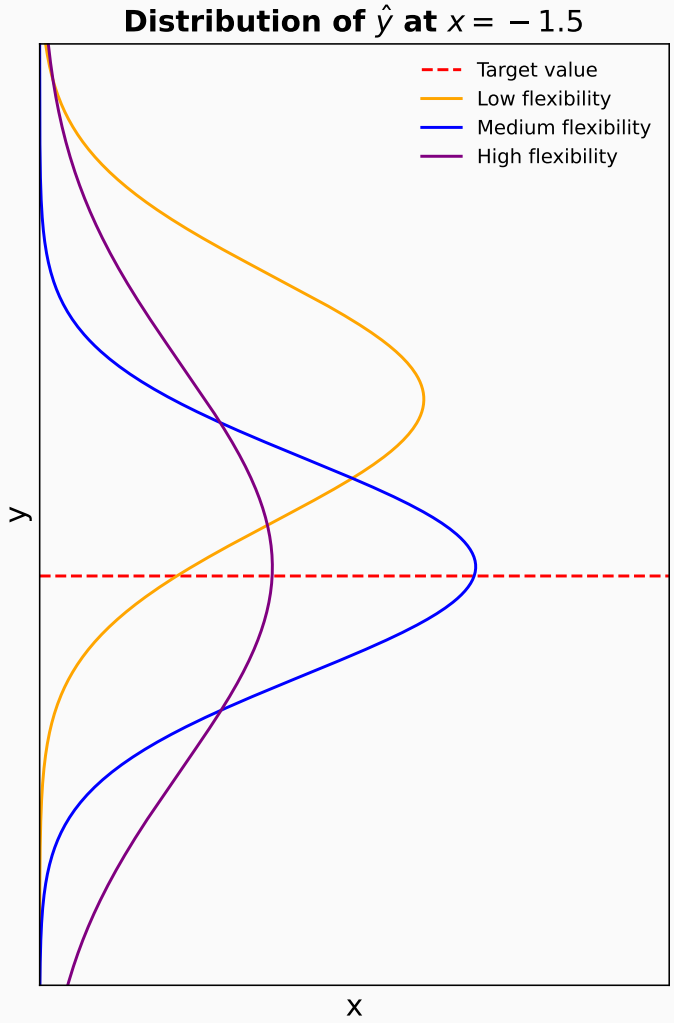
Bootstrapped predictions for different estimators



- Bias is the tendency of an estimator to systematically produce an over-estimate or under-estimate
- Variance is the amount by which the estimates change when fitting the model on another random sample

The shape of the PDF gives information about the estimator's properties

- The first moment measures the central tendency (bias)
- The second moment measures the dispersion (variance)
- Integrating over a range of the PDF provides the probability



To derive formally bias-variance trade-off, consider the mean squared error loss function

$$\mathcal{L}_{mse}(y_i, \hat{y}_i) = \frac{1}{n}\sum_{i=1}^{n}(y_i - f(x_i, \hat{\beta}))^2 \tag{5}$$

in predictive applications, the error is typically expressed at the observation level, rathen than the sample

- The squared term ensures that positive and negative estimated errors do not cancel out
- Observations whose predicted output lie far from the true output are given more weight

The expected training mean squared error can be expressed as the sum of reducible and irreducible error

$$E\left[(y-\hat{f})^2\right] = \underbrace{(f-\hat{f})^2}_{\text{reducible error}} + \underbrace{Var[\epsilon]}_{\text{irreducible error}} \tag{6}$$

where $f = F(x_i)$, $\hat{f} = f(x_i, \hat{\beta})$, and the subscript i is dropped to simplify notation

- We can only minimise reducible error, the distance between the target and the estimated function
- Even with a perfect estimate for F, the training error is non-zero because ϵ_i cannot be predicted using x_i

▸ Proof

The reducible error in equation (6) can be further decomposed as the sum of bias (squared) and variance

$$E\left[(y-\hat{f})^2\right] = \underbrace{\left(E[\hat{f}]-f\right)^2}_{\text{bias}^2} + \underbrace{E\left[\left(\hat{f}-E[\hat{f}]\right)^2\right]}_{\text{variance}} + Var[\epsilon] \quad (7)$$

- Minimising the test sample error implies minimising both the bias and variance on the training sample
- Increased regularisation can be seen as reducing the estimator's variance at the cost of introducing bias

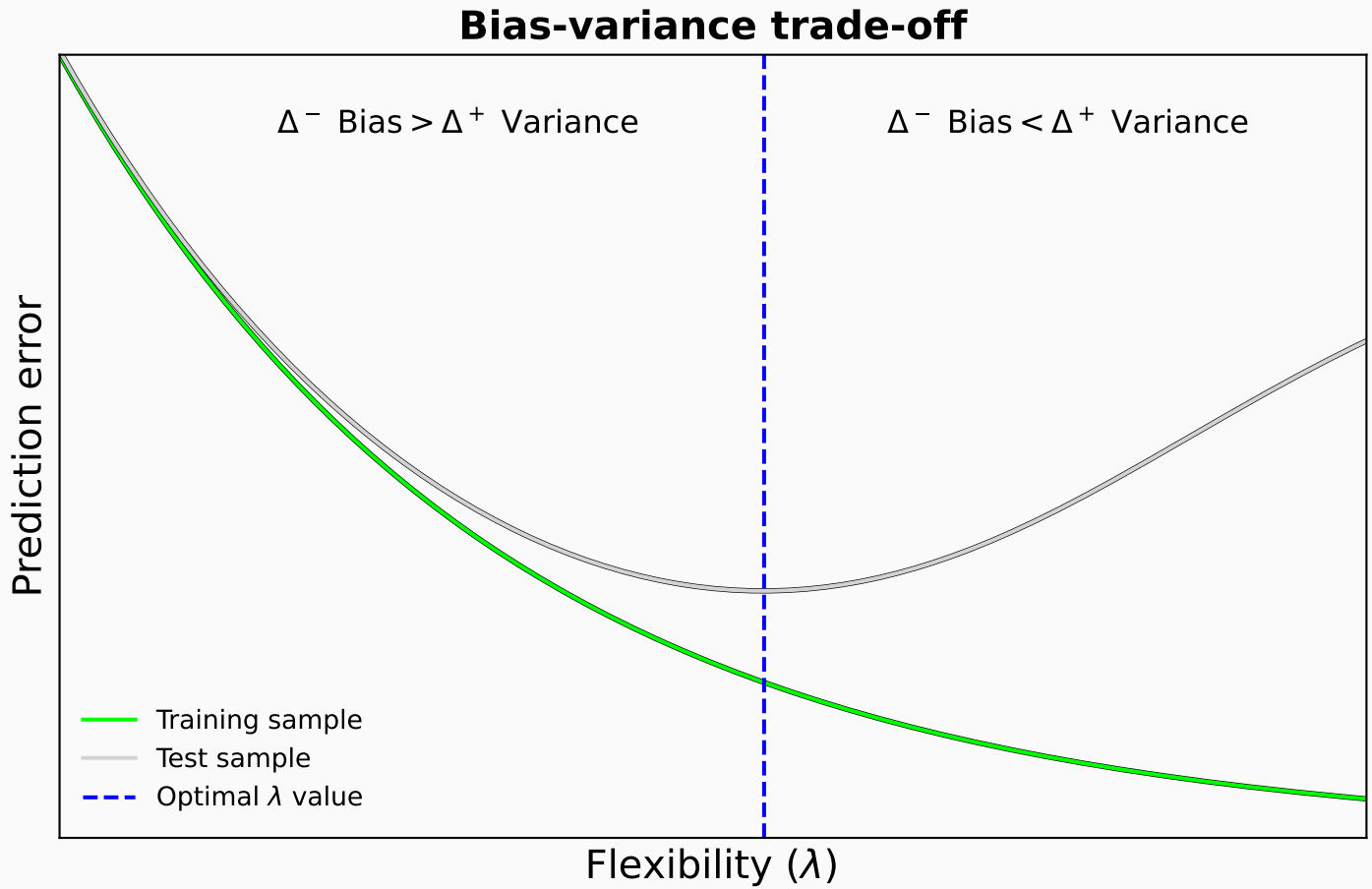
▸ Proof

As we add flexibility to the model (i.e. more parameters) the bias decreases and the variance increases

Bootstrapped bias and variance

Flexibility	MSE	Bias^2	Var.	Δ MSE	Δ Bias^2	Δ Var.
Low	18.10	16.87	1.23			
Medium	2.53	0.90	1.63	-15.57	-15.97	0.40
High	6.87	0.52	6.36	4.34	-0.38	4.73

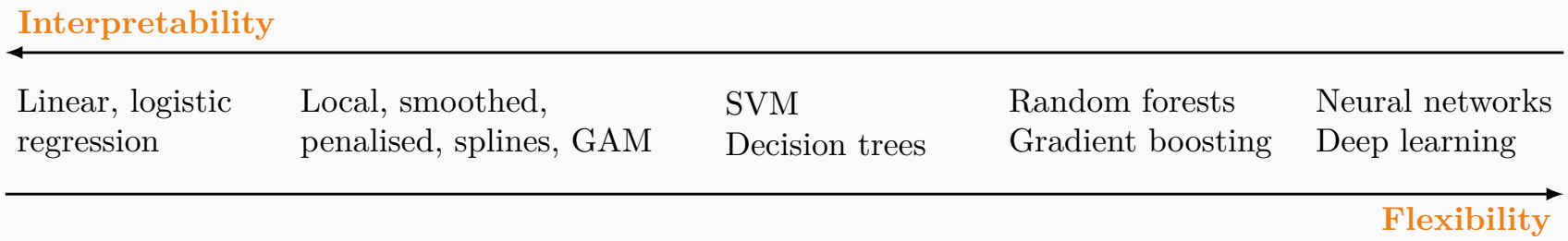
The challenge of regularisation is to find the model that strikes the right balance between bias and variance



Belkin et al. (2019) showed evidence of a double-descent performance curve, showing that increasing model capacity beyond the point of interpolation can improve generalisation.

The trade-off between bias and variance is often seen as a balance between interpretability and flexibility

- Linear models are interpretable but lack the flexibility to approximate complex non-linear functions
- Flexibility means that a model with different sets of parameters can produce similar predictions



Uncertainty

Uncertainty

The statistical properties (e.g. finite sample, asymptotic) of flexible models can often not be derived analytically

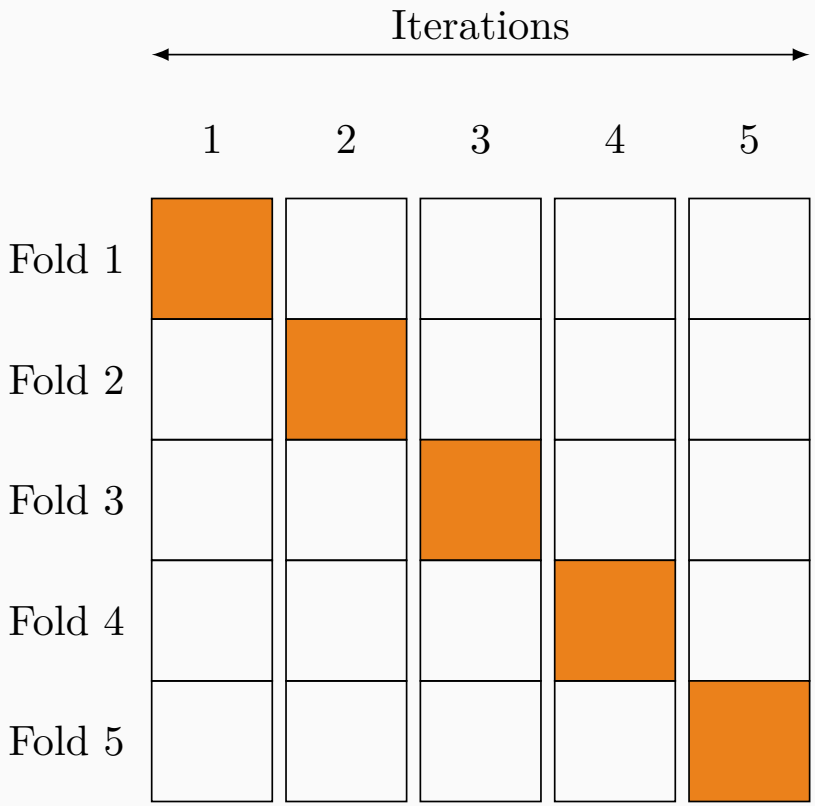
- **Resampling** are techniques to repeatedly draw samples from a dataset to assess the properties of a model
- **Cross-validation** estimates the preformance and generalisability of predictive models
- **Bootstrap** estimates the distribution of a statistic e.g. bias, confidence intervals for the predictions

Predictive models are estimated on a training sample and assessed on a separate test sample

- There is a trade-off between the performance of the model and the assessment of that performance
- The best model is trained using the entire dataset, but the performance estimate would be biased and inefficient
- An accurate performance estimate requires a large test sample, which could be used to train a better model

Cross-validation (Stone 1974) is a resampling method that addresses both these issues simultaneously

- The observations are randomly partitioned into k distinct groups or "folds" without replacement
- The model is repeatedly trained on $k-1$ partitions and assessed on the remaining partition
- There are k iterations, so that each observation is used $k-1$ times for training and once for testing




```
1  eh_cv = model_selection.cross_validate(model, x, y, cv=5, scoring='
       neg_mean_squared_error')
2  eh_cv = np.mean(-eh_cv['test_score'])
```

For instance, the cross-validated mean squared error is calculated by averaging the MSE on the k test partitions

$$\mathcal{L}_{mse}^{cv}(k) = \frac{1}{k}\sum_{1}^{k}\left[\frac{1}{n_t}\sum_{i=1}^{n_t}\left(y_i - \hat{f}(x_i)\right)^2\right]$$

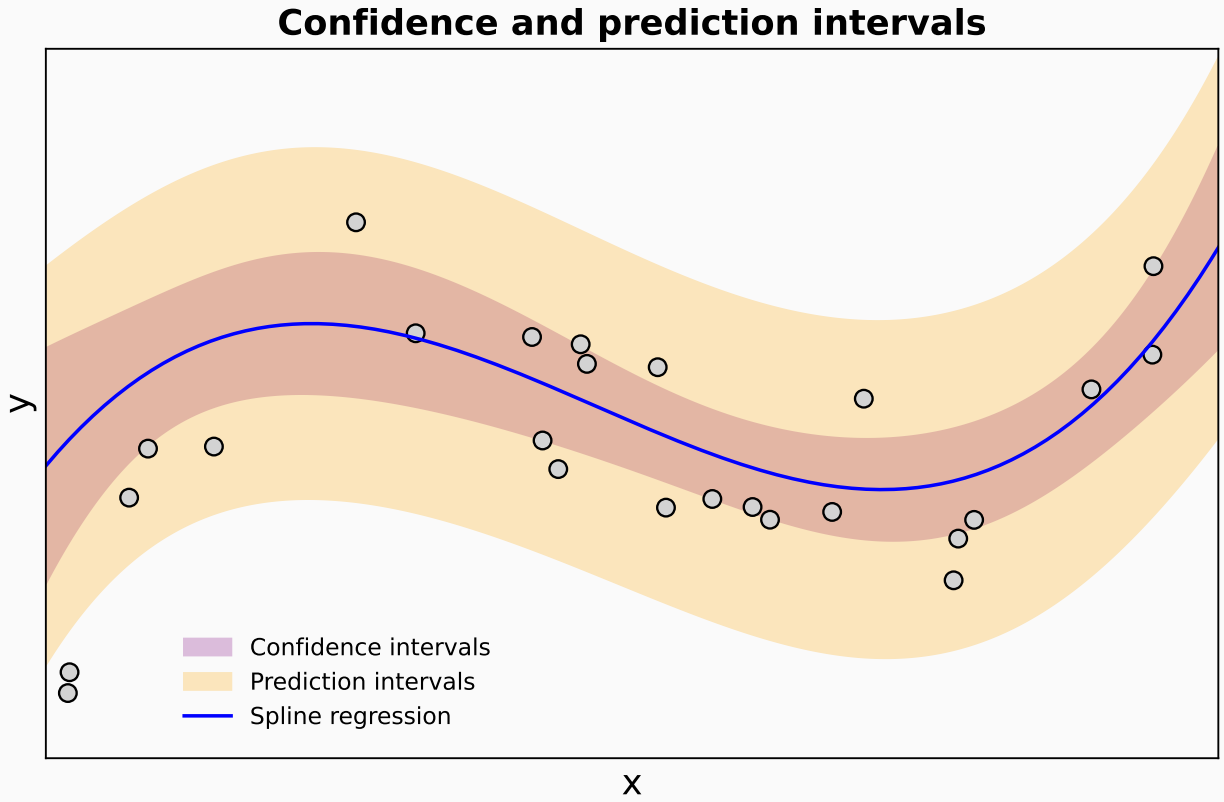
where $i = 1, \ldots, n_t$ are the observations of the test fold

- The value of k is a trade-off between the computational cost and the bias of the estimator
- "Leave-one-out" cross-validation involves $k = n$ folds. In practice, $k = \{5, 10\}$ provide good estimates (Breiman and Spector 1992). See Hastie et al. (2009) for guidance

The bootstrap is a non-parametric method used to estimate model bias and variance (Efron 1979)

- Estimates uncertainty in the predicted response by constructing intervals with a level of confidence
  - → Confidence intervals i.e. mean response[1]
  - → Prediction intervals i.e. individual predictions
- These intervals accounts for both the sampling and the model variability – as well as observation variability for PIs

[1] In econometrics, confidence intervals refer to intervals for the estimated parameters. Parametric intervals are less suited, as they rely on specific model structures and assumptions about the underlying data distribution.



Confidence interval: 95% of the true mean prediction for a given input is likely to fall within the interval, based on the model and sampling variability. **Prediction interval**: 95% of new individual observations for the given input are likely to fall within the interval, accounting for both model uncertainty and the natural variability of the data.

We would like to draw additional samples from the population, but we only observe a single dataset in practice

- Bootstrap repeatedly resamples observations with replacement from the dataset i.e. independence
- Bootstrapped samples have the same size as the dataset i.e. so some observations are repeated
- The bootstrapped distribution approximates the sampling distribution for the population for a large number of bootstraps (Efron and Tibshirani 1994)

▸ Robust prediction intervals

Quantile confidence and (naive) prediction intervals

```
n_boot = 1000
n_obs  = X.shape[0]
yh_confidence = np.zeros((n_boot, n_obs))
yh_prediction = np.zeros((n_boot, n_obs))

for i in range(n_boot):
    X_boot, y_boot = utils.resample(X, y, replace=True, n_samples=n_obs)
    model.fit(X_boot, y_boot)
    yh_boot = model.predict(X).flatten()
    eh_boot = y_boot - model.predict(X_boot)
    eh_boot = np.random.choice(eh_boot.flatten(), n_obs, replace=True)
    yh_confidence[i] = yh_boot
    yh_prediction[i] = yh_boot + eh_boot

conf_low, conf_high = np.quantile(yh_confidence, q=[0.025, 0.975], axis=0)
pred_low, pred_high = np.quantile(yh_prediction, q=[0.025, 0.975], axis=0)
yh_mean = np.mean(yh_confidence, axis=0)
```

Summary

Summary

Supervised learning and econometrics can be seen as function approximation methods. You may use the former when

- Prediction accuracy is more important than the interpretability of estimated parameters
- No functional form is suggested by theory and no sensible parametric model can be written
- The target function is non-linear, potentially high-dimensional and there are many observations

Summary

Every statistical model can be seen as striking a different balance between prediction bias and variance

- This trade-off can also be interpreted as a compromise between flexibility and interpretability
- The model choice depends on the research problem and the nature of the target function
- Re-sampling methods provide additional information about the statistical properties of a model

Thank you for your attention!

References i

- Stone, Mervyn (1974). **“Cross-validatory choice and assessment of statistical predictions”.** In: *Journal of the Royal Statistical Society* 36.2, pp. 111–147 (cit. on p. 37).
- Efron, Bradley (1979). **“Bootstrap methods: Another look at the jackknife”.** In: *The Annals of Statistics* 7.1, pp. 1–26 (cit. on p. 40).
- Breiman, Leo and Philip Spector (1992). **“Submodel Selection and Evaluation in Regression. The X-Random Case”.** In: *International Statistical Review* 60.3, pp. 291–319 (cit. on p. 39).
- Efron, Bradley and Robert Tibshirani (1994). ***An introduction to the bootstrap*.** CRC Press (cit. on p. 42).

References ii

- Breiman, Leo (2001). **“Statistical modeling: The two cultures”.** In: *Statistical Science* 16.3, pp. 199–231.
- Hastie, Trevor, Robert Tibshirani, and Jerome Friedman (2009). ***The elements of statistical learning*.** Springer (cit. on pp. 39, 57).
- Mullainathan, Sendhil and Jann Spiess (2017). **“Machine learning: An applied econometric approach”.** In: *Journal of Economic Perspective* 31.2, pp. 87–106.
- Belkin, Mikhail et al. (2019). **“Reconciling modern machine[?]learning practice and the classical bias–variance trade[?]off”.** In: *Proceedings of the National Academy of Sciences* 116.32, pp. 15849–15854 (cit. on p. 32).

Appendix

A random sample is representative of the population, every observation has an equal probability of being sampled

- This implies that the sample's characteristics are distributed similarly as those of the population
- Unrepresentative or insufficient samples causes models to produce biased and/or inefficient estimates
- Supervised learning models are assessed on another random sample called the test sample (more on this)

Estimators can be considered as random variables with the estimated output being its realisation

- A random variable is a series of realisations (i.e. estimates) that take real values with a probability
- With one random sample we observe one realisation, one value out of the set of possible values i.e. other draws
- These values and their associated probabilities are respesented using a probability density function

Splines fit polynomial regressions to intervals of data while ensuring continuity and smoothness at the boundaries

$$y_i = \underbrace{\sum_{d=0}^{D} \beta_d \, x_i^d}_{polynomial} + \sum_{p}^{P} \theta_p \, \underbrace{I(x_i \geq p)}_{indicator} \underbrace{(x_i - p)^{\alpha}}_{continuity} + \epsilon_i$$

where d is the degree of the polynomial, p the interval boundary (i.e. knot) and α the continuity at p

- $\alpha = 0$ is a piecewise polynomial, only the indicator is left and there is no continuity or smoothness at p
- $\alpha = 1, 2, 3$ enforces continuity in the function, the first derivative and second derivative, respectively

◂ Presentation

$$
\begin{aligned}
E\left[(y-\hat{f})^2\right] &= E\left[(f+\epsilon-\hat{f})^2\right] \\
&= E\left[(f-\hat{f}+\epsilon)^2\right] \\
&= E\left[(f-\hat{f})^2+2\epsilon(f-\hat{f})+\epsilon^2\right] \\
&= E\left[(f-\hat{f})^2\right]+E\left[2\epsilon(f-\hat{f})\right]+E[\epsilon^2] \\
&= E\left[(f-\hat{f})^2\right]+2\underline{E[\epsilon]}E[f-\hat{f}]+E[\epsilon^2] \\
E\left[(y-\hat{f})^2\right] &= \underbrace{(f-\hat{f})^2}_{\text{reducible error}} + \underbrace{Var[\epsilon]}_{\text{irreducible error}}
\end{aligned}
$$

Note that f is constant and $\hat{f}$ is assumed to be fixed. If the error is truly random $E(\epsilon) = 0$

◂ Presentation

$$
\begin{aligned}
E\left[\left(y-\hat{f}\right)^2\right] &= E\left[\left(f+\epsilon-\hat{f}\right)^2\right] \\
&= E\left[\left(f+\epsilon-\hat{f}+E[\hat{f}]-E[\hat{f}]\right)^2\right] \\
&= E\left[\left(f-E[\hat{f}]\right)+\epsilon+\left(E[\hat{f}]-\hat{f}\right)^2\right] \\
&= E\left[\left(f-E[\hat{f}]\right)^2\right]+E(\epsilon^2)+E\left[\left(E[\hat{f}]-\hat{f}\right)^2\right] \\
&+2E\left[\left(f-E[\hat{f}]\right)\epsilon\right]+2E\left[\epsilon\left(E[\hat{f}]-\hat{f}\right)\right] \\
&+2E\left[\left(f-E[\hat{f}]\right)\left(E[\hat{f}]-\hat{f}\right)\right]
\end{aligned}
$$

$$= \left(f - E[\hat{f}]\right)^2 + E(\epsilon^2) + E\left[\left(E[\hat{f}] - \hat{f}\right)^2\right]$$
$$+ 2\left(f - E[\hat{f}]\right)\underline{E[\epsilon]}$$
$$+ 2\underline{E[\epsilon]}E\left[E[\hat{f}] - \hat{f}\right]$$
$$+ 2E\left[\left(f - E[\hat{f}]\right)\underline{\left(E[\hat{f}] - \hat{f}\right)}\right]$$

$$E\left[\left(y - \hat{f}\right)^2\right] = Bias^2[\hat{f}] + Var[\hat{f}] + Var[\epsilon]$$

Note that f is constant and $\hat{f}$ is assumed to be fixed so $E\left(\hat{f} - E(\hat{f})\right) = 0$. If the error is truly random $E(\epsilon) = 0$

◂ Presentation

Computing robust prediction intervals requires additional steps, beyond the scope of this class

- Correlation between the bootstrap training and test samples (entire dataset) $\rightarrow$ LOO bootstrap estimator
- Bias in the size of the training and the test sample i.e. as in cross-validation $\rightarrow$ ".632 estimator"
- Robustness to overfitting $\rightarrow$ ".632+" estimator. See Hastie et al. (2009) for additional guidance