Machine Learning

Statistical Learning Framework

Souhaib Ben Taieb

March 8, 2020

University of Mons

Table of contents

Learning from data

"Machine learning is a **scientific discipline** that explores the **construction and study of algorithms** that can **learn from data**."

- The essence of machine learning
  - A pattern exists
  - We cannot pin it down mathematically
  - We have data on it
- Learning examples
  - Spam Detection
  - Product Recommendation
  - Credit Card Fraud Detection
  - Medical Diagnosis
- The goal of **supervised learning** is to find an underlying relationship between one or several **input** variables and one (or several) **output** variable(s) given **data**.

Table of contents

Components of supervised learning

- The **input** variables are typically denoted using the symbol X. It is also called *predictors*, *independent variables*, *features*, *variables* or just *inputs*. If we observe p different variables, we write $X = (X_1, X_2, \ldots, X_p)$. The **inputs** belong to an input space $\mathcal{X}$. We typically assume that $\mathcal{X} \subseteq \mathbb{R}^p$.
- The **output** variable is typically denoted using the symbol Y. It is often called the *response* or *dependent variable*. The **output** belongs to an output space $\mathcal{Y}$.
  - Regression: $\mathcal{Y} \subseteq \mathbb{R}$
  - Binary classification: $\mathcal{Y} \in \{-1, 1\}$ or $\mathcal{Y} \in \{0, 1\}$
  - Multi-class classification (with K categories): $\mathcal{Y} \in \{1, 2, \ldots, K\}$

Components of supervised learning

- The **data**, also called *training set*, is a set of n input-output pairs

$$\mathcal{D} = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\} = \{(x_i, y_i)\}_{i=1}^{n},$$

  where $x_i = (x_{i1}, \ldots, x_{ip})$. Each pair is also called an *example* or *sample* or *data point*. The space $\mathcal{X} \times \mathcal{Y}$ is called the *data space*.
- In order to learn (or estimate) the input-output relation, we have to postulate the existence of a model for the data.

Probabilistic data model

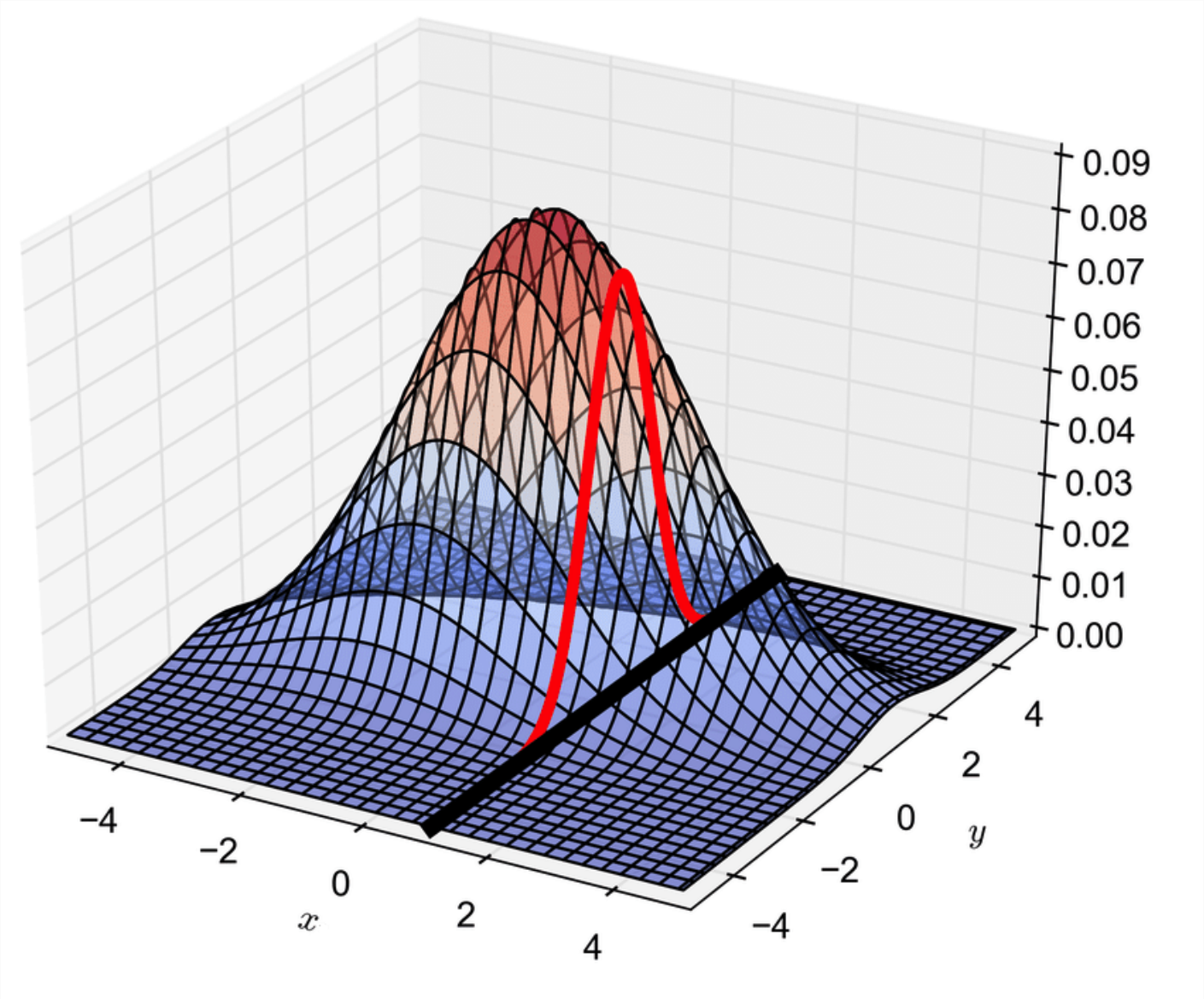
- We assume there exists a fixed unknown data distribution $p_{X,Y}(x, y)$ according to which the data are identically and independently distributed (i.i.d.).
- The probability distribution $p_{X,Y}(x, y)$ models different *sources of uncertainty*.
- We assume that it factorizes as

$$p_{X,Y}(x, y) = p_X(x)p_{Y|X}(y|x)$$

where

  - the marginal distribution $p_X(x)$ models uncertainty in the sampling of the input points.
  - the conditional distribution $p_{Y|X}(y|x)$ describes a stochastic (non-deterministic) relation between inputs and output.

Conditional distribution (continuous case)



Source: https://tinyurl.com/tvhn5v6

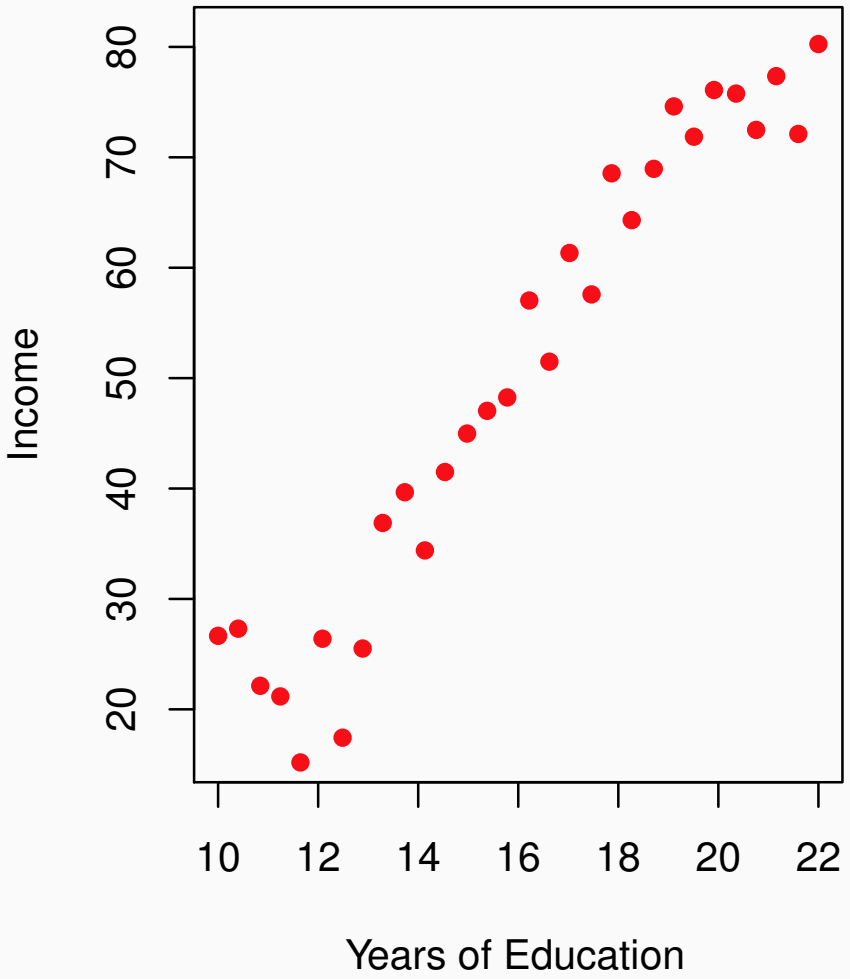
Probabilistic data model - Regression

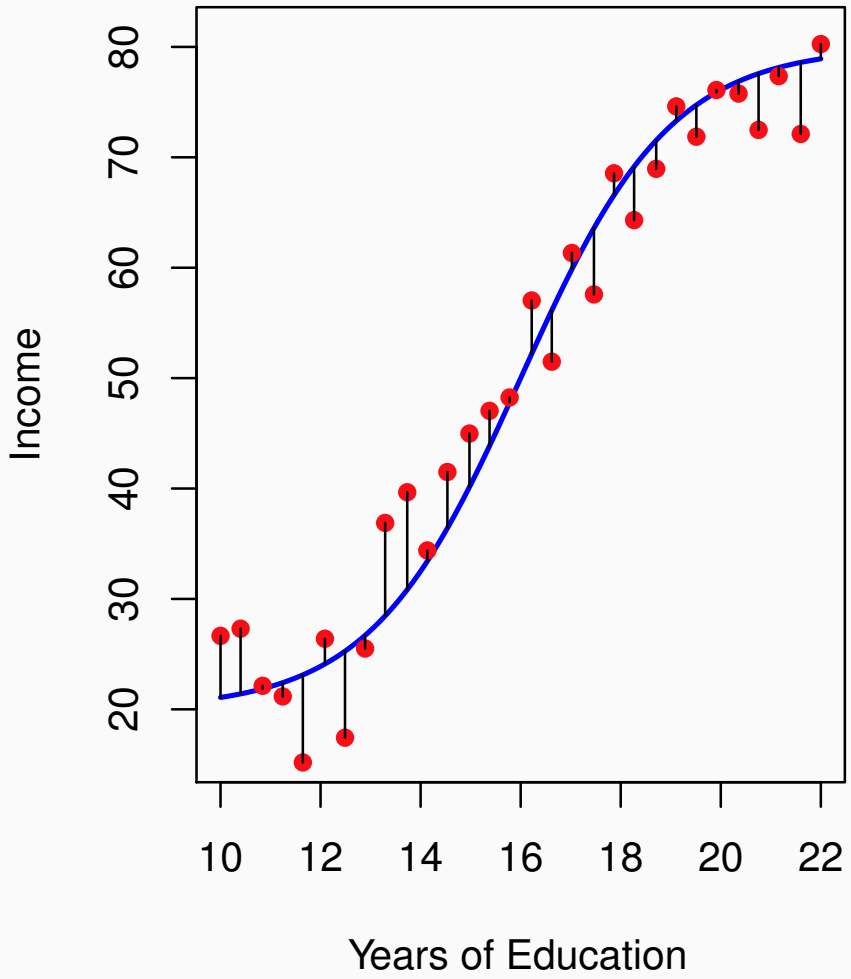
- In regression, the following (additive error) model is often considered:

$$y = f(x) + \varepsilon,$$

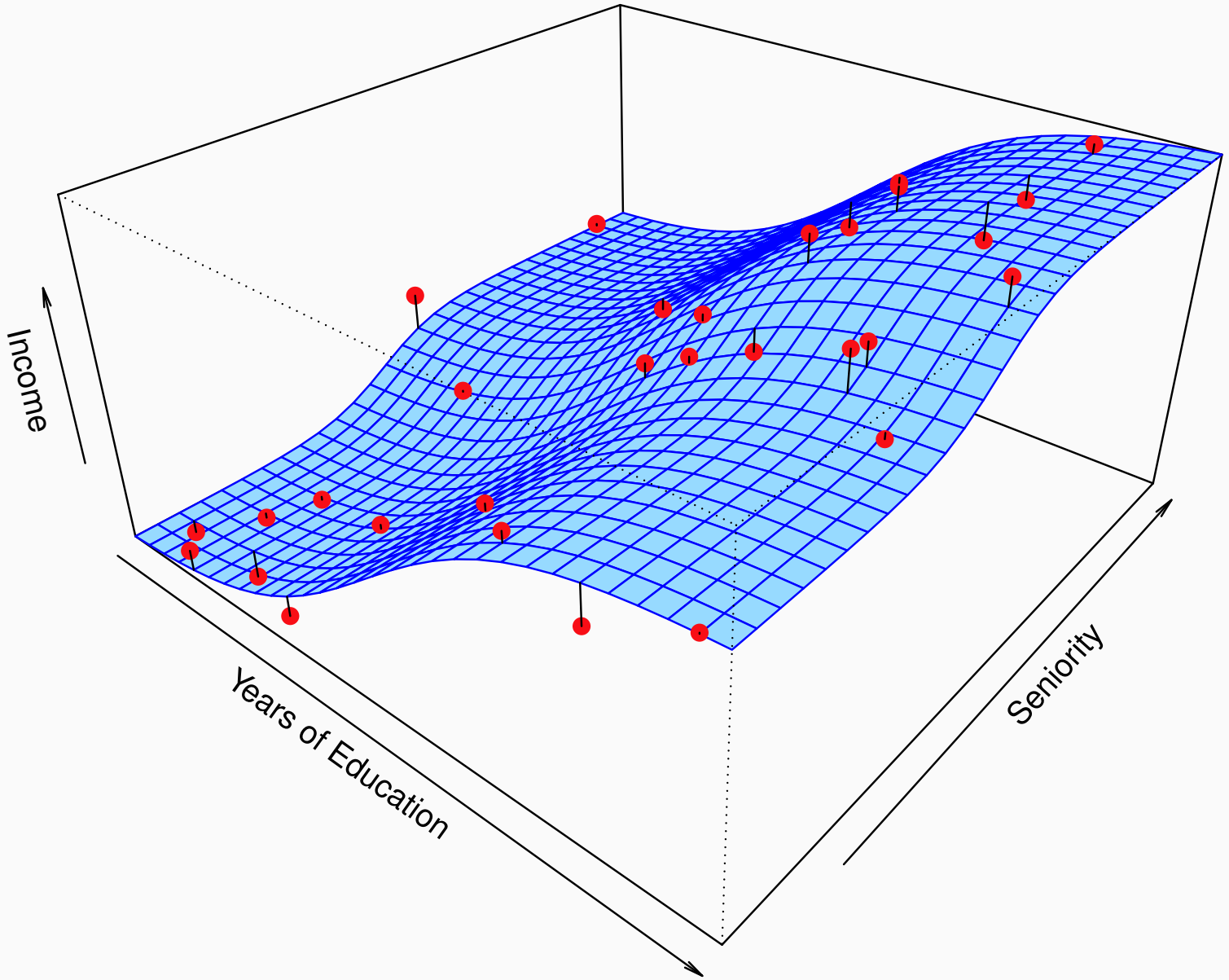
  where
  - f is a fixed unknown function
  - ε is random noise (independent of x) with $\mathbb{E}[\varepsilon] = 0$ and $\mathrm{Var}(\varepsilon) = \sigma^2$, with $\sigma \in [0, \infty)$.
- The model implies that the conditional distribution depends on x only through the conditional mean.
  - $\mathbb{E}[y|x] = f(x)$ and $\mathrm{Var}[y|x] = \sigma^2$
- For example, f can be a linear function $f(x) = \beta_0 + \sum_{j=1}^{p} \beta_j x_j$ ($\beta_j \in \mathbb{R}$, $j = 0, 1, \ldots, p$) and $\varepsilon \sim \mathcal{N}(0, \sigma^2)$.
  - $y|x \sim \mathcal{N}(f(x), \sigma^2)$

Probabilistic data model - Regression





Probabilistic data model - Regression



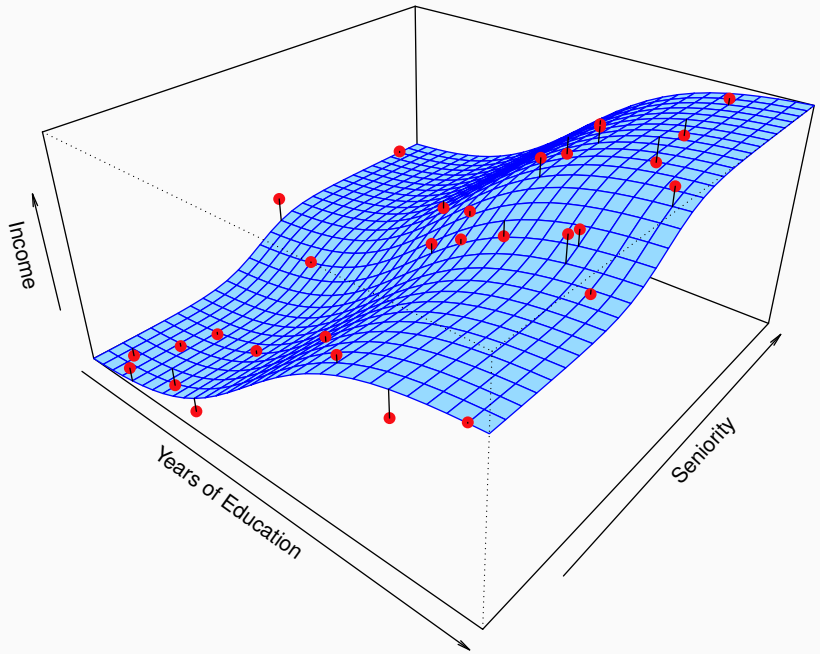
Why estimate f?

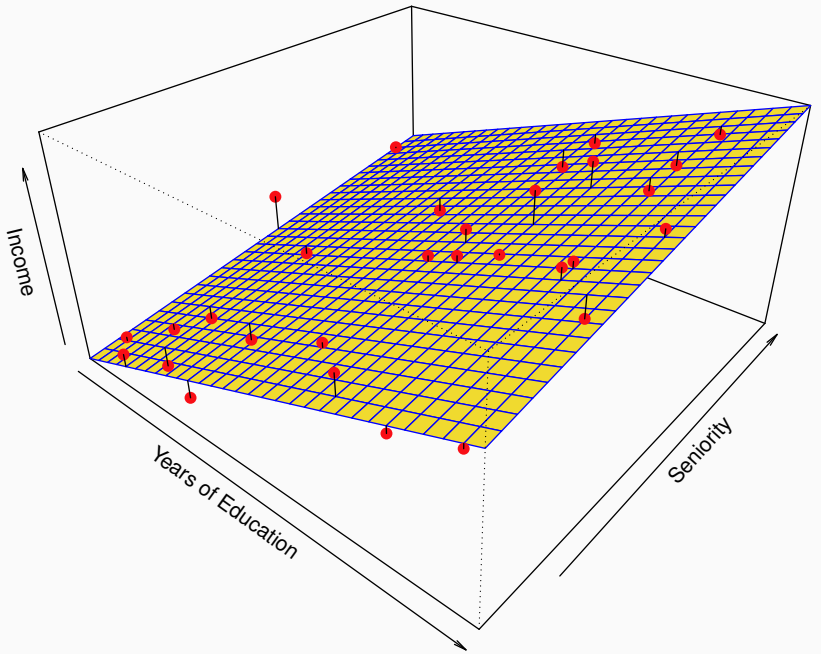
- **Prediction**:
  - We want to predict the output for a new input x^*, i.e.

$$\hat{y}^* = \hat{f}(x^*)$$

    where $\hat{f}$ represents our estimate for f, and $\hat{y}^*$ represents the resulting prediction for y^*.
- **Inference (or explanation)**:
  - Which predictors are associated with the response?
  - What is the relationship between the response and each predictor?

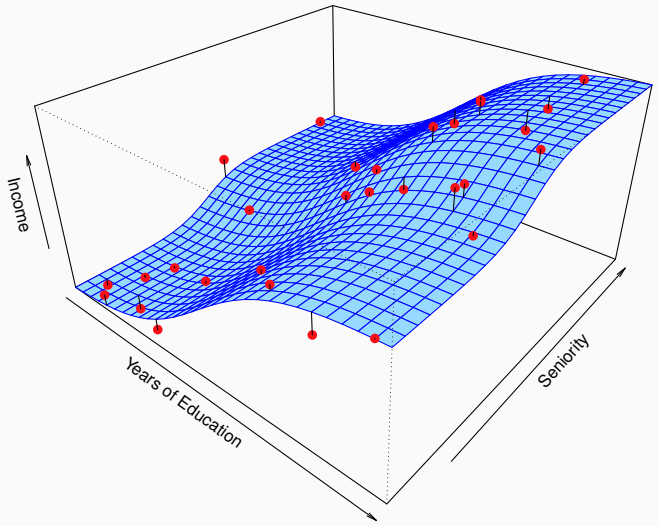
How to estimate f?

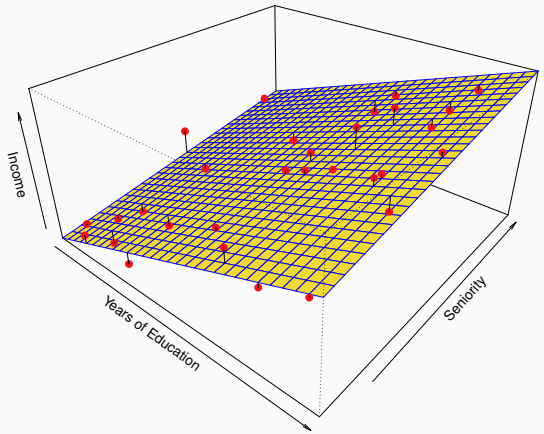


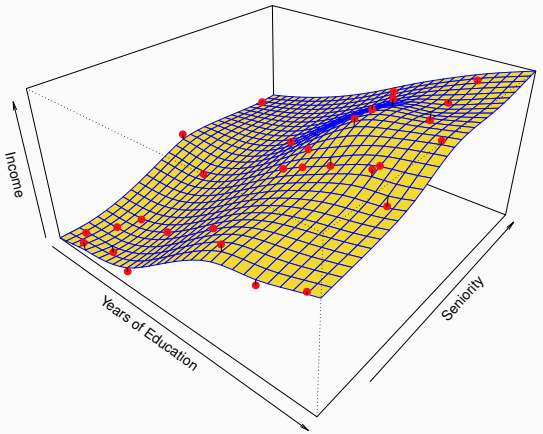


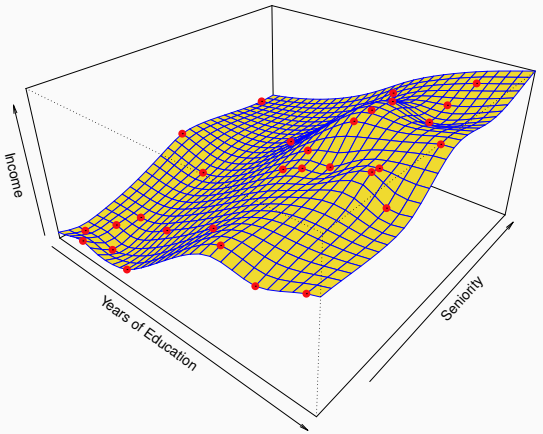
$$\hat{f}(\text{education}, \text{seniority}) = \hat{\beta}_0 + \hat{\beta}_1 \times \text{education} + \hat{\beta}_2 \times \text{seniority}$$

How to estimate f?





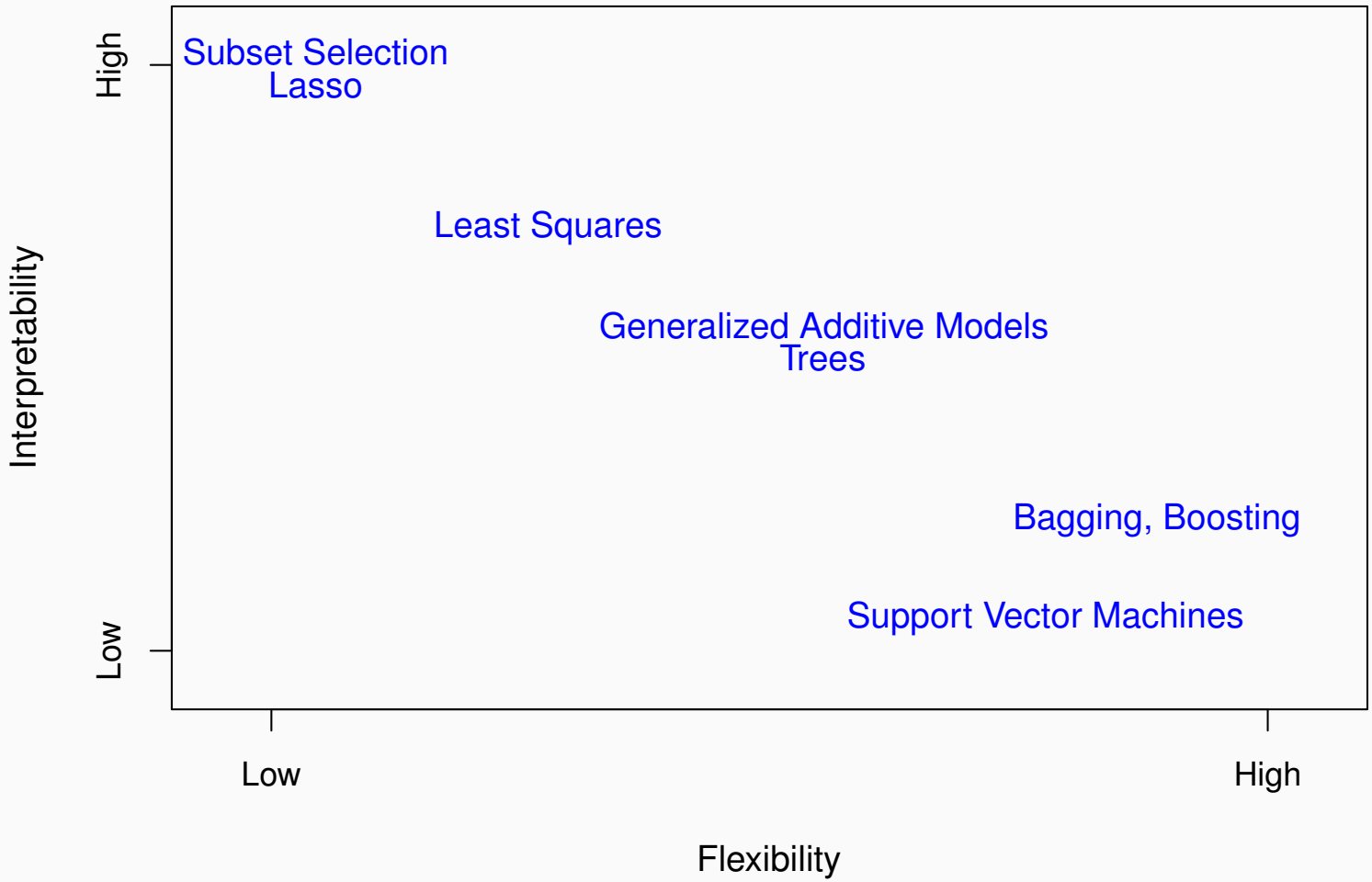




Two classes of estimation methods

- **Parametric methods**
  - Assumption about the form of f, e.g. linear
  - The problem of estimating f reduces to estimating a set of parameters
  - Usually a good starting point for many learning problems
  - Poor performance if linearity assumption is wrong
  - Example: $\hat{f}(x) = \hat{\beta}_0 + \hat{\beta}_1 x_1 + \cdots + \hat{\beta}_p x_p$
- **Non-parametric methods**
  - No *explicit* assumptions about the form of f
  - High flexibility: it can potentially fit a wider range of shapes for f
  - A large number of observations is required to estimate f with good accuracy
  - Example: $\hat{f}(x) = \frac{1}{k} \sum_{x_i \in N_k(x)} y_i$ where $N_k(x)$ is the set with the k nearest neighbors of x.

Model Interpretability vs flexibility



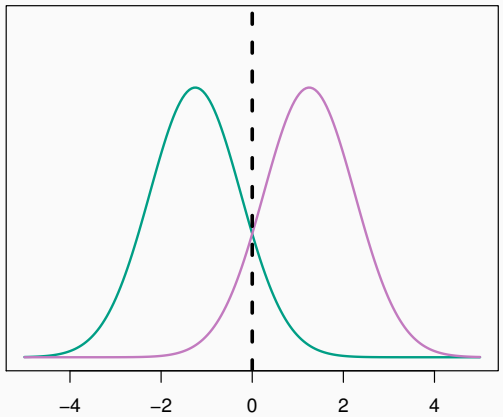
See "Explainable AI" for a modern view of this tradeoff.

Probabilistic data model - Classification

- In classification, y is a discrete random variable ($p(y|x)$ is a conditional pmf). We cannot use the previous additive error model. The notion of "noise" is different.
- Using Bayes' rule, we can write

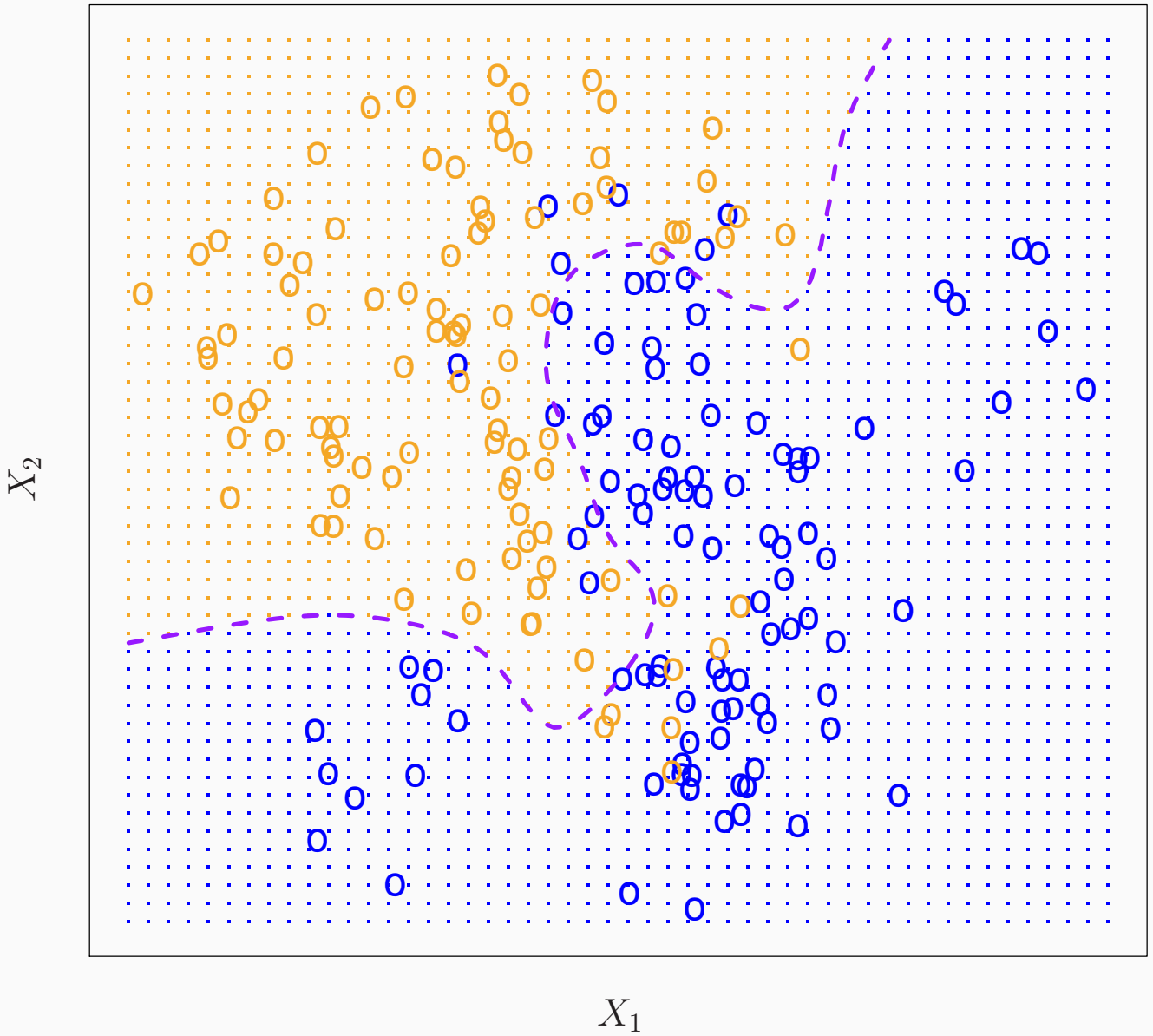
$$p(y|x) = \frac{p(x|y)p(y)}{p(x)} \propto p(x|y)p(y) \overset{y \text{ uniform}}{\propto} p(x|y)$$

- Let us consider $K = 2$ and $p = 1$.



$p(x|y = -1)$ $p(x|y = +1)$

Probabilistic data model - Classification



Components of supervised learning (continued)

- The goal of learning is to estimate the "best" input-output relation, rather than the whole distribution $p(x, y)$. In other words, we want the "best" prediction (value) for a given input x.
- Given the conditional distribution $p(y|x)$, which **value** should we use as output prediction for an input x?
- To do that, we need to define "best" by specifying a **loss function**

$$\mathsf{L} : \mathcal{Y} \times \mathcal{Y} \to [0, \infty),$$

which is a (pointwise) measure of the error $L(y, f(x))$ we incur in when predicting $f(x)$ in place of y.

  - $L(y, f(x)) = (y - f(x))^2$ (square error loss)
  - $L(y, f(x)) = |y - f(x)|$ (absolute error loss)
  - $L(y, f(x)) = \mathbb{1}\{y \neq f(x)\}$ (zero-one loss)
  - ...

Components of supervised learning (continued)

- Given a loss function, the "best" input-output relation is the **target function** $f^* : \mathcal{X} \to \mathcal{Y}$ which minimizes the **expected loss** (or **expected risk**)

$$\mathbb{E}[L(y, f(x))] = \int L(y, f(x))\ p(x, y)\ dx\ dy,$$

  also called the **out-of-sample error** or **generalization error**.

Table of contents

Target function (optimal prediction) in regression

In regression (with squared error loss), the target function f^* minimizes

$$\mathbb{E}_{x,y}[(y - f(x))^2] = \mathbb{E}_x[\mathbb{E}_{y|x}[(y - f(x))^2|x]].$$

It suffices to minimize the error pointwise, i.e.

$$f(x) = \text{argmin}_{c \in \mathbb{R}} \ \mathbb{E}_{y|x}[(y - c)^2|x = x].$$

The solution is

$$f(x) = \mathbb{E}_{y|x}[y|x],$$

the conditional expectation, also known as the regression function.

Thus the best prediction of y at any point x is the conditional mean, when *best is measured by average squared error*.

The target (regression) function has the lowest mean squared errors (MSE) given by σ^2, the variance of $y|x$.

Proof

We want to minimize

$$r(c; x) = \mathbb{E}_{y|x}[(y - c)^2 | x = x].$$

The necessary condition for optimality is given by

$$\begin{aligned}
\frac{\partial r(c; x)}{\partial c} = 0 \iff \frac{\partial \mathbb{E}_{y|x}[(y - c)^2 | x = x]}{\partial c} &= 0 \\
\iff \mathbb{E}_{y|x}\left[\frac{\partial (y - c)^2}{\partial c} | x = x\right] &= 0 \\
\iff \mathbb{E}_{y|x}\left[-2(y - c) | x = x\right] &= 0 \\
\iff c = \mathbb{E}_{y|x}\left[y | x = x\right]
\end{aligned}$$

The sufficient condition for optimality (a minimum) is given by

$$\frac{\partial^2 r(c; x)}{\partial c^2} > 0 \iff 2 > 0.$$

Target function (optimal prediction) in classification

Let us consider multi-class classification with K categories where $y \in \mathcal{C} = \{C_1, \ldots, C_K\}$.

With the zero-one loss, the target function f^* minimizes

$$\mathbb{E}_{x,y}[\mathbb{1}\{y \neq f(x)\}] = p(y \neq f(x)) = \mathbb{E}_x[\mathbb{E}_{y|x}[\mathbb{1}\{y \neq f(x)\}|x].$$

It suffices to minimize the error pointwise, i.e.

$$f(x) = \text{argmin}_{c \in \mathcal{C}} \ \mathbb{E}_{y|x}[\mathbb{1}\{y \neq c\}|x = x].$$

Proof

We have that

$$
\begin{aligned}
&\mathbb{E}_{y|x}[\mathbb{1}\{y \neq c\}|x = x] \\
&= \sum_{k=1}^{K} \mathbb{1}\{C_k \neq c\} p(y = C_k|x = x) \\
&= \sum_{k:C_k \neq c} 1 \times p(y = C_k|x = x) + 0 \times p(y = c|x = x) \\
&= \sum_{k:C_k \neq c} p(y = C_k|x = x) \\
&= \sum_{k:C_k \neq c} p(y = C_k|x = x) + p(y = c|x = x) - p(y = c|x = x) \\
&= \sum_{k=1}^{K} p(y = C_k|x = x) - p(y = c|x = x) \\
&= 1 - p(y = c|x = x).
\end{aligned}
$$

Target function (optimal prediction) in classification

This implies that

$$\begin{aligned} f(x) &= \text{argmin}_{c \in \mathcal{C}} \ \mathbb{E}_{y|x}[\mathbb{1}\{y \neq f(x)\}|x = x] \\ &= \text{argmin}_{c \in \mathcal{C}} \ 1 - p(y = c|x = x) \\ &= \text{argmax}_{c \in \mathcal{C}} \ p(y = c|x = x). \end{aligned}$$

Equivalently, we have that

$$f(x) = C_k \text{ if } p(y = C_k|x = x) = \ \max_{c \in \mathcal{C}} \ p(y = c|x = x).$$

This target function (classifier) is called the Bayes classifier, which has the following error rate at x:

$$1 - \max_{k=1,\ldots,K} \ p(y = C_k|x = x),$$

also called the Bayes error rate, which gives the lowest possible error rate that could be achieved if we knew the true probability distribution of the data.

Supervised learning in practice

- In practice, the target function cannot be computed exactly since we do not know $p(x, y)$.
- Instead, we observe a dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$ where $(x_i, y_i) \overset{\text{i.i.d.}}{\sim} p(x, y)$, i.e. $x_i \overset{\text{i.i.d.}}{\sim} p(x)$ and $y_i \overset{\text{i.i.d.}}{\sim} p(y|x = x_i)$.
- Using the **dataset** $\mathcal{D}$, the goal of the **learning algorithm** is to pick the "best" hypothesis or function, denoted $\hat{f}_{\mathcal{D}}$ or $\hat{f}$, from a **hypothesis set** $\mathcal{H}$, where "best" is defined by the **loss function** L.
- The **hypothesis set** $\mathcal{H}$ contains all the hypotheses (functions) we consider. It is often implicitly defined.
- In other words, the learning algorithm will **estimate** the target function.

Table of contents

In-sample and out-of-sample errors

Since we do not know $p(x, y)$, we cannot solve the following optimization problem:

$$f = \text{argmin}_{h:\mathcal{X}\to\mathcal{Y}} \; E_{\text{out}}(h),$$

where

$$E_{\text{out}}(h) = \mathbb{E}_{x,y}[L(y, h(x))] = \int L(y, h(x)) \; p(x, y) \; dx \; dy$$

is the **out-of-sample error** of h.

In-sample and out-of-sample errors

Since we do not know $p(x, y)$, we cannot solve the following optimization problem:

$$f = \text{argmin}_{h:\mathcal{X}\to\mathcal{Y}} \; E_{\text{out}}(h),$$

where

$$E_{\text{out}}(h) = \mathbb{E}_{x,y}[L(y, h(x))] = \int L(y, h(x)) \; p(x, y) \; dx \; dy$$

is the **out-of-sample error** of h.

However, using $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$ where $(x_i, y_i) \overset{\text{i.i.d.}}{\sim} p(x, y)$, we can solve the following optimization problem:

$$\hat{f} = \text{argmin}_{h\in\mathcal{H}} \; E_{\text{in}}(h),$$

where

$$E_{\text{in}}(h) = \frac{1}{n} \sum_{i=1}^{n} L(y_i, h(x_i))$$

is the **in-sample error** of h.

In-sample and out-of-sample errors

Unfortunately, the in-sample error is often **not a good estimate** of the out-of-sample error, as it does not properly account for model complexity. In fact, the dataset $\mathcal{D}$ has been used **twice**: (1) to select $\hat{f}$ in $\mathcal{H}$, and (2) to evaluate $\hat{f}$.

In-sample and out-of-sample errors

Unfortunately, the in-sample error is often **not a good estimate** of the out-of-sample error, as it does not properly account for model complexity. In fact, the dataset $\mathcal{D}$ has been used **twice**: (1) to select $\hat{f}$ in $\mathcal{H}$, and (2) to evaluate $\hat{f}$.

A better estimate of the out-of-sample error can be obtained by using a training set $\mathcal{D}_{\text{train}}$ to compute $\hat{f}$, and a **seperate** test set $\mathcal{D}_{\text{test}}$ to evaluate $\hat{f}$.

In-sample and out-of-sample errors

Unfortunately, the in-sample error is often **not a good estimate** of the out-of-sample error, as it does not properly account for model complexity. In fact, the dataset $\mathcal{D}$ has been used **twice**: (1) to select $\hat{f}$ in $\mathcal{H}$, and (2) to evaluate $\hat{f}$.

A better estimate of the out-of-sample error can be obtained by using a training set $\mathcal{D}_{\text{train}}$ to compute $\hat{f}$, and a **seperate** test set $\mathcal{D}_{\text{test}}$ to evaluate $\hat{f}$.

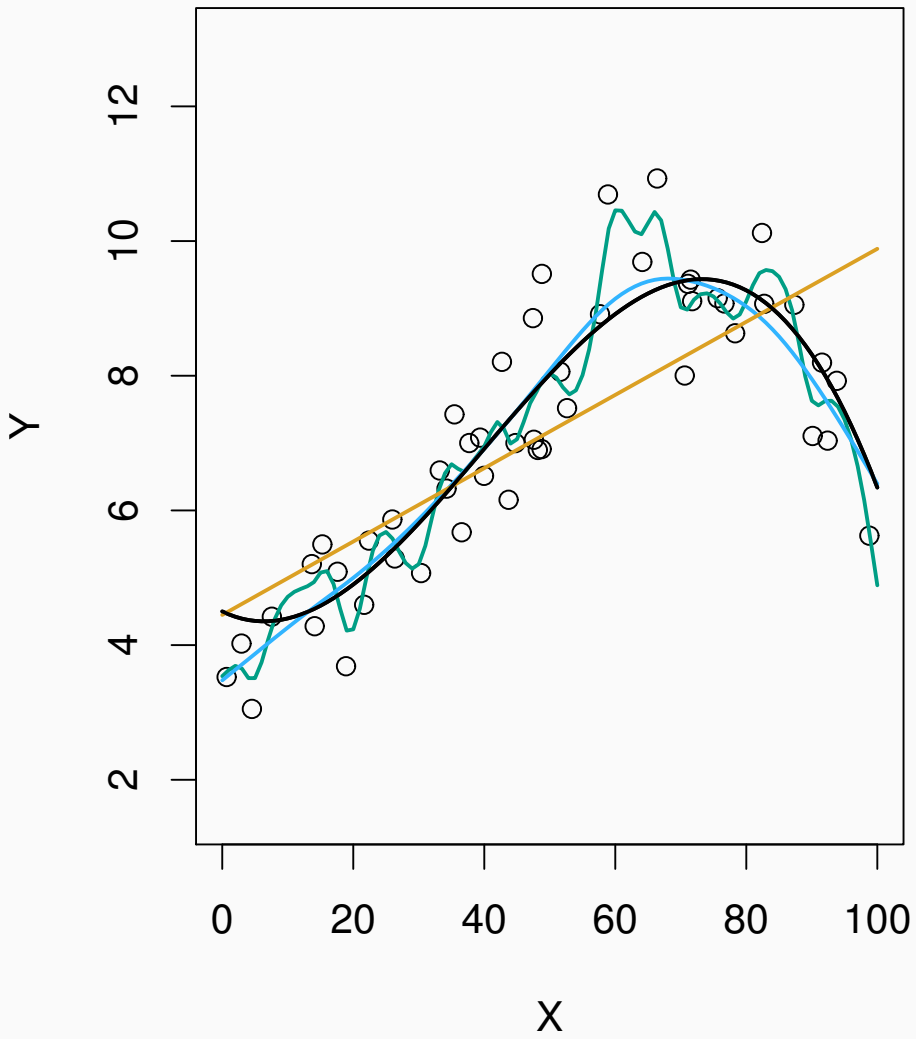
In other words, $\hat{f}$ is computed by minimizing the training errors, i.e.

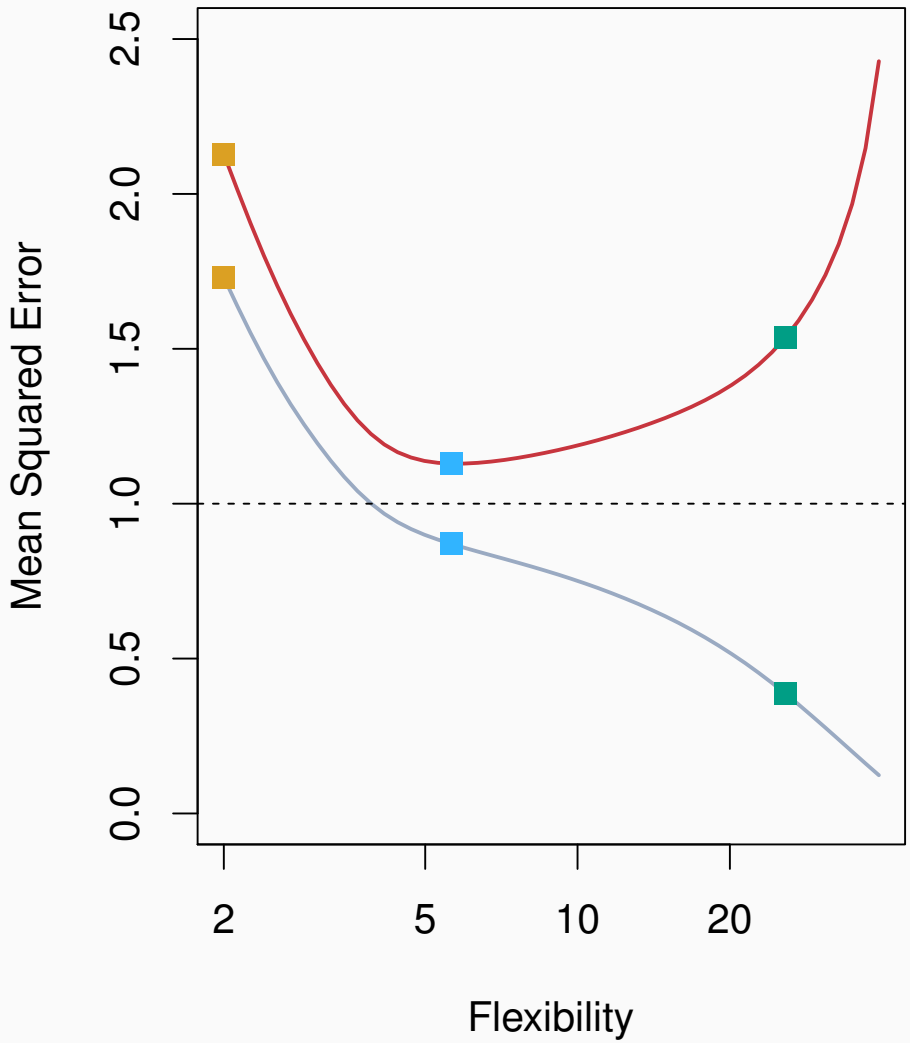
$$\hat{f} = \text{argmin}_{h \in \mathcal{H}} \frac{1}{n_{\text{train}}} \sum_{(x_i, y_i) \in \mathcal{D}_{\text{train}}} L(y_i, h(x_i)).$$

An estimate of the out-of-sample error is given by averaging the test errors:

$$\frac{1}{n_{\text{test}}} \sum_{(x_i, y_i) \in \mathcal{D}_{\text{test}}} L(y_i, h(x_i)).$$

Training and test errors in regression (with splines)





Black: true curve

Orange: linear regression

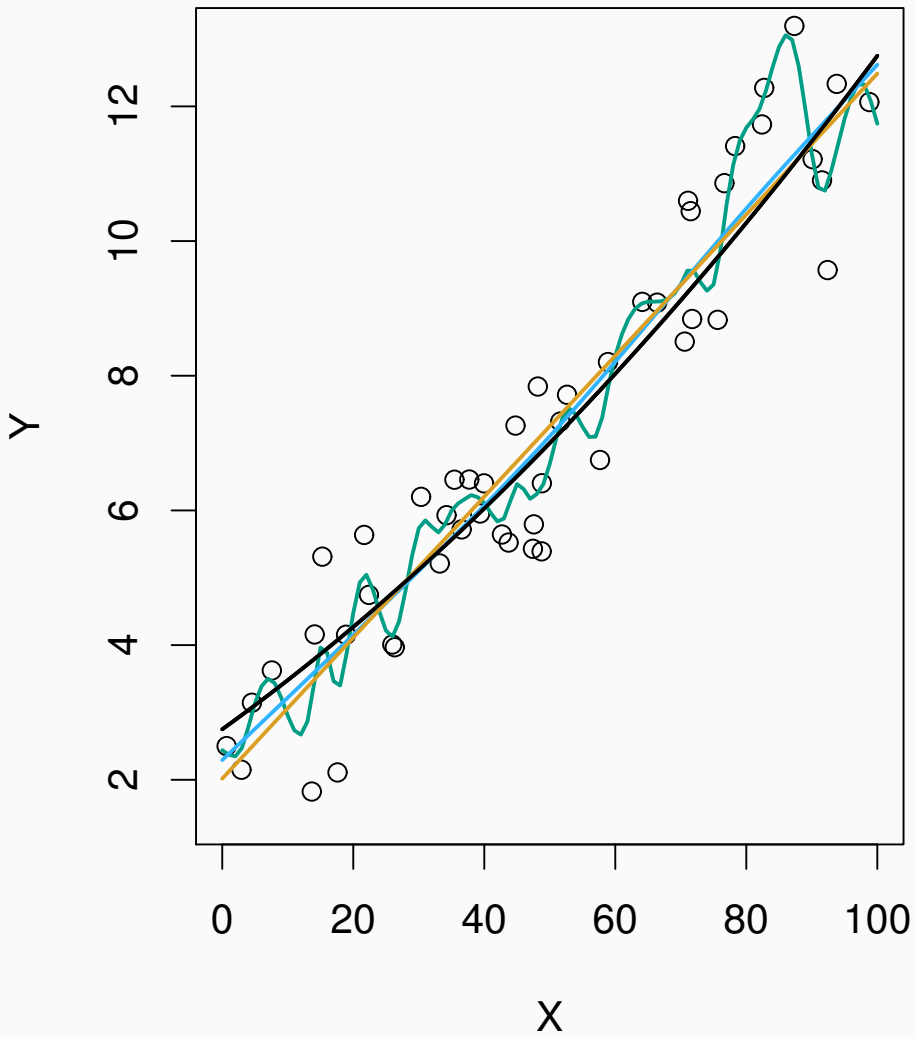
Blue/green: Smoothing splines

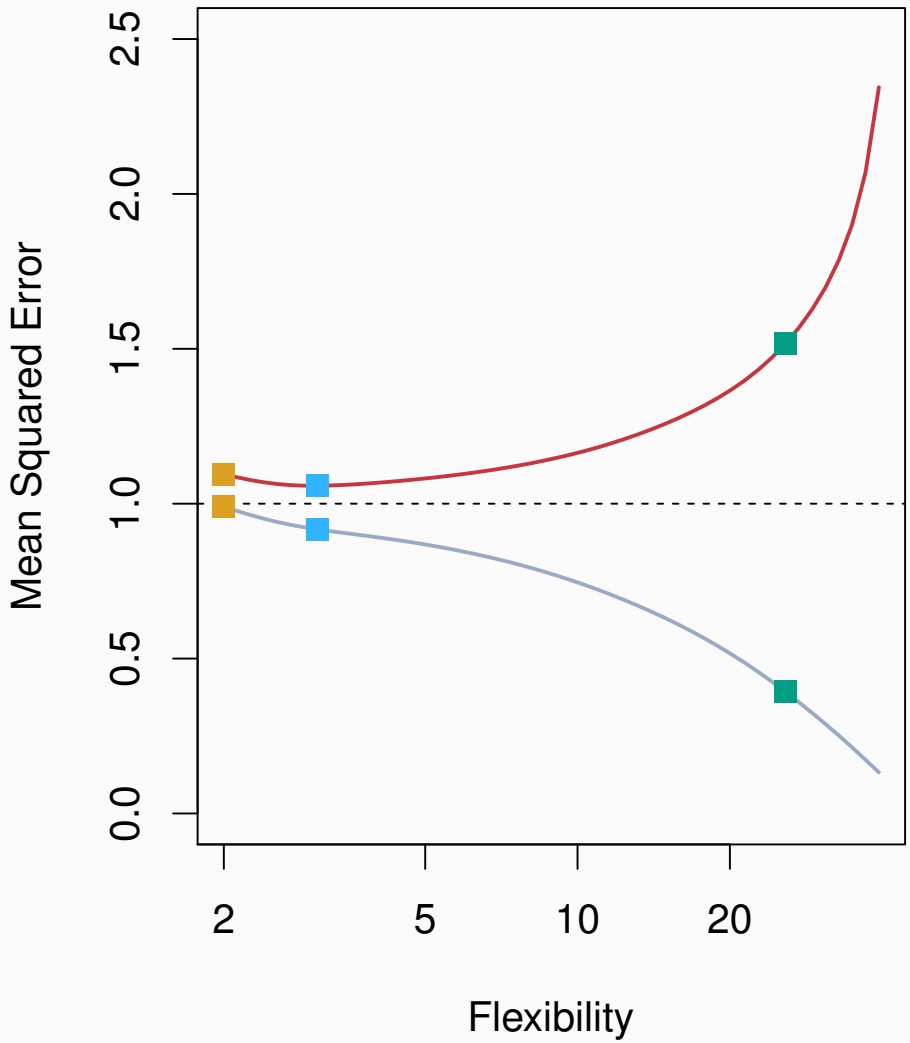
Grey: Training MSE

Red: Test MSE

Dashed: Minimum test MSE

Training and test errors in regression (with splines)





Black: true curve

Orange: linear regression

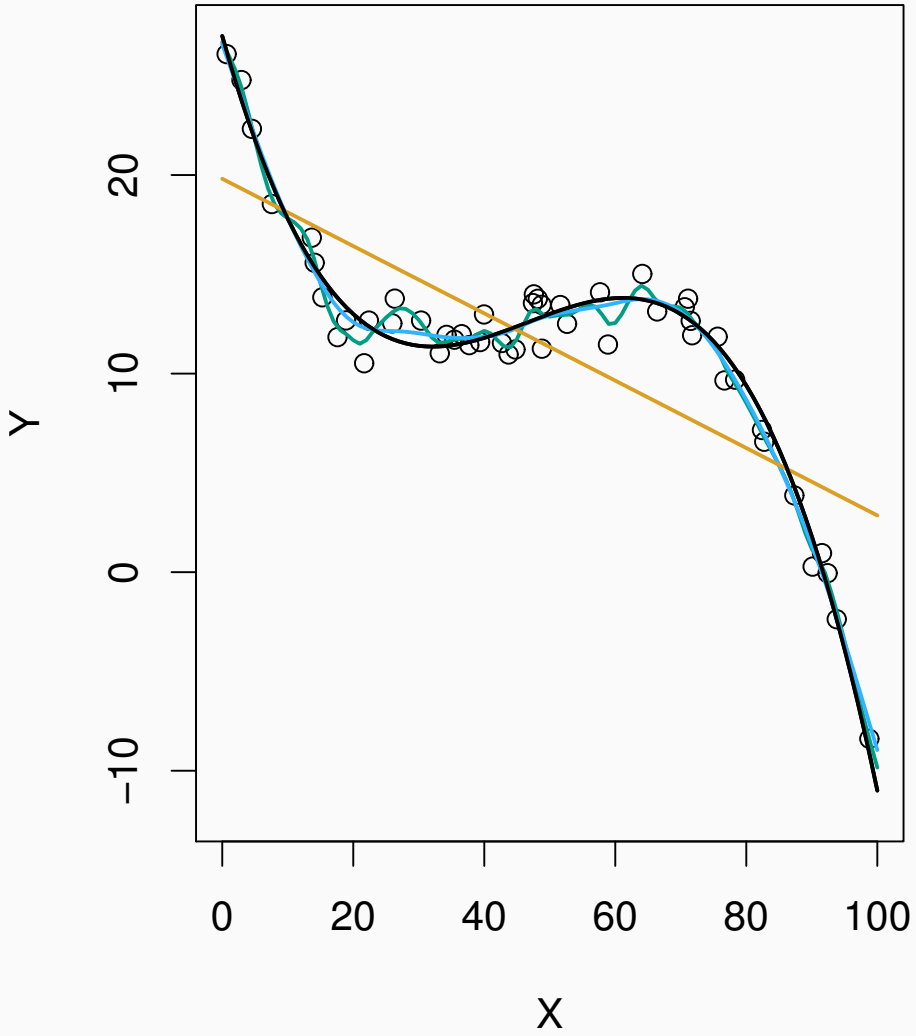
Blue/green: Smoothing splines

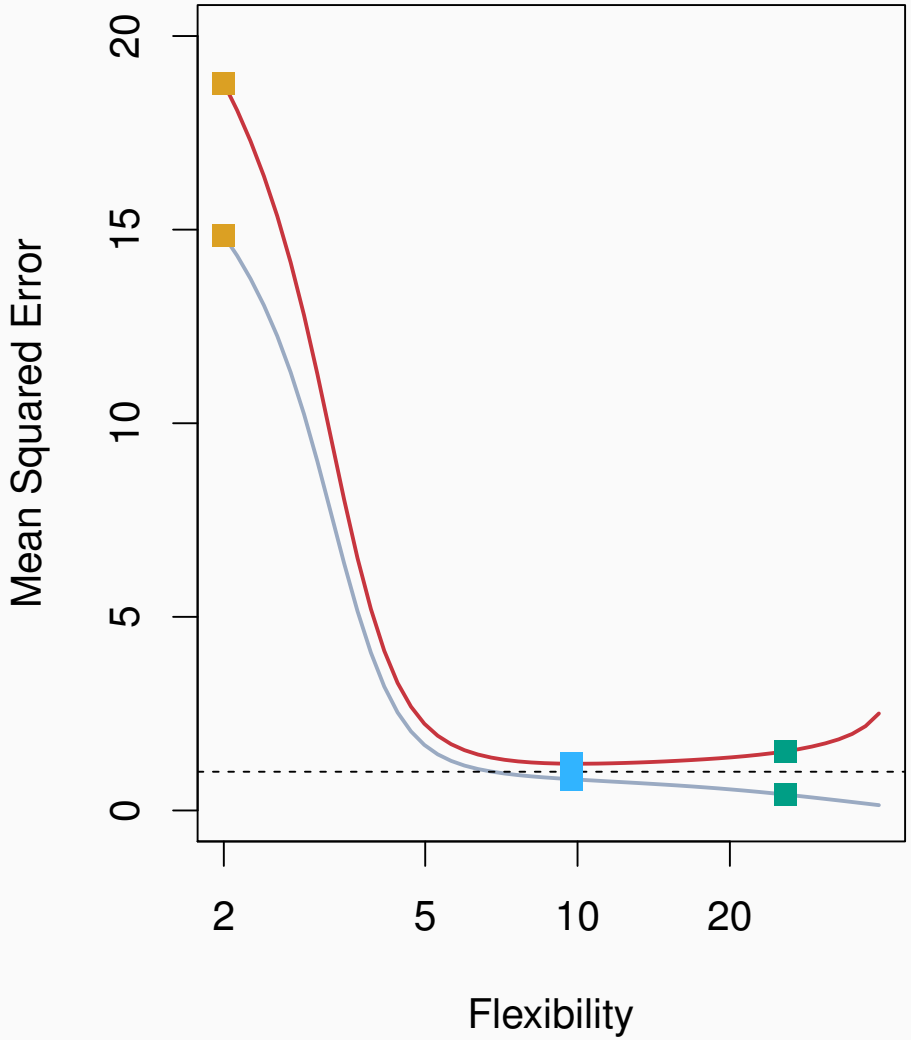
Grey: Training MSE

Red: Test MSE

Dashed: Minimum test MSE

Training and test errors in regression (with splines)





Black: true curve

Orange: linear regression

Blue/green: Smoothing splines

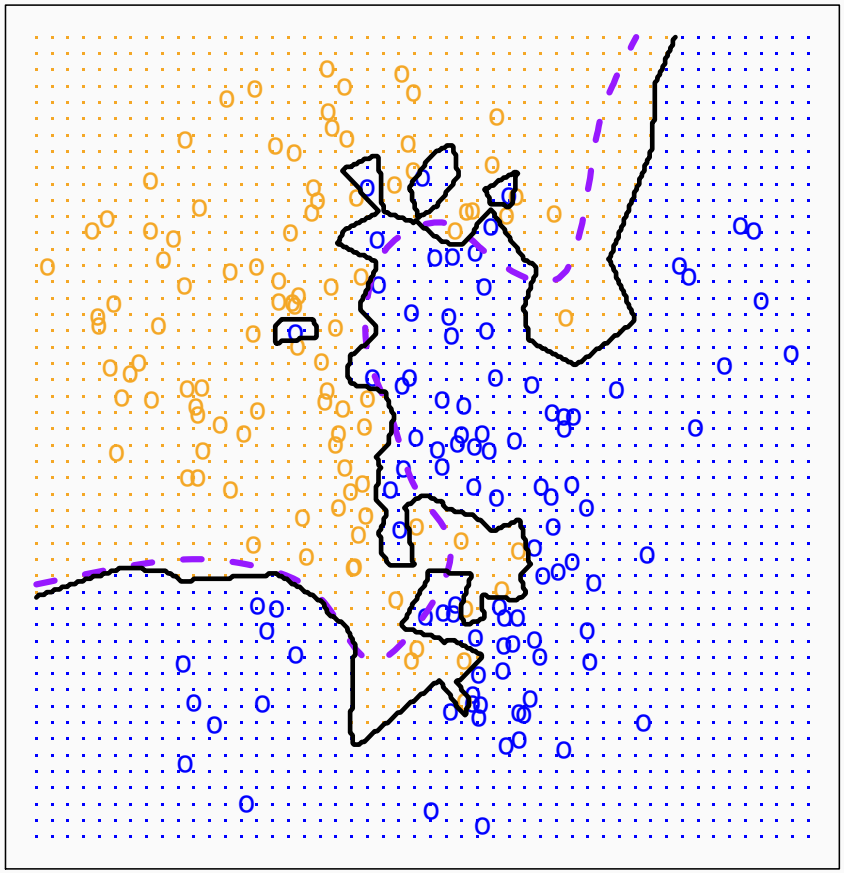
Grey: Training MSE

Red: Test MSE

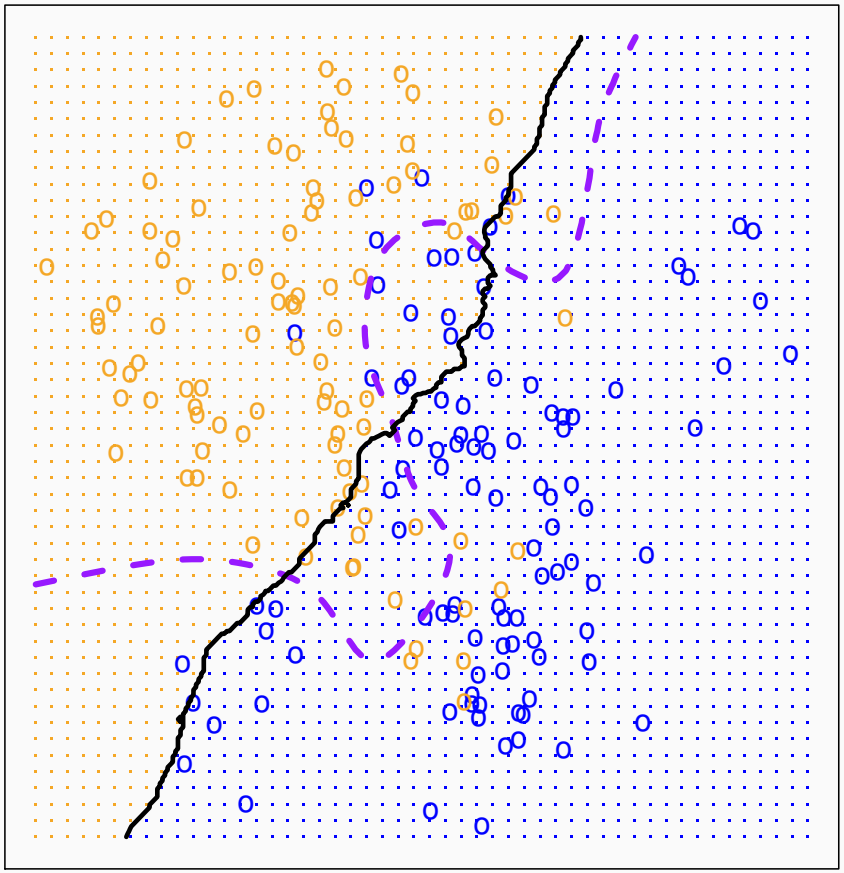
Dashed: Minimum test MSE

Training and test errors in classification (with KNN)

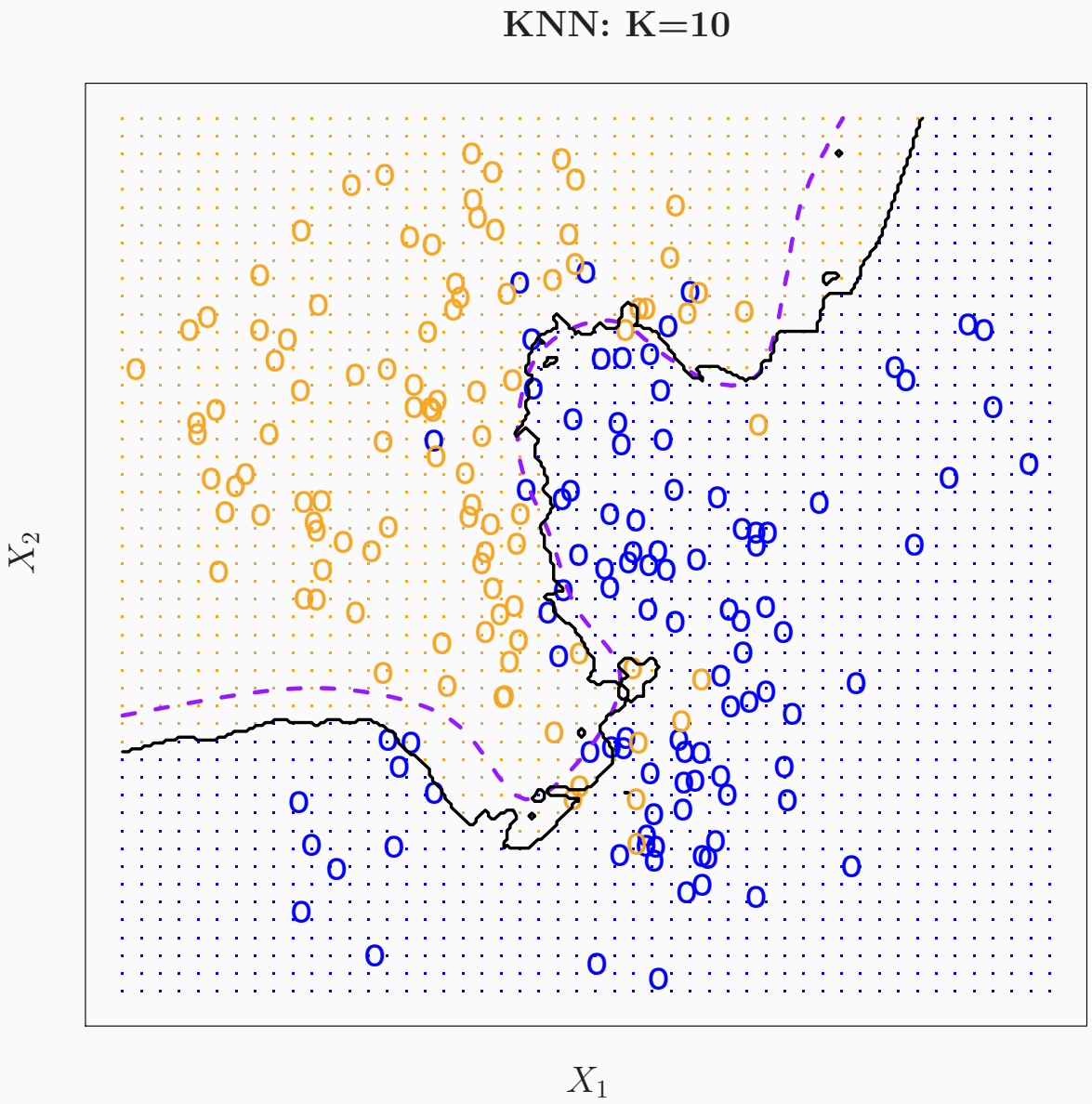
KNN: K=1



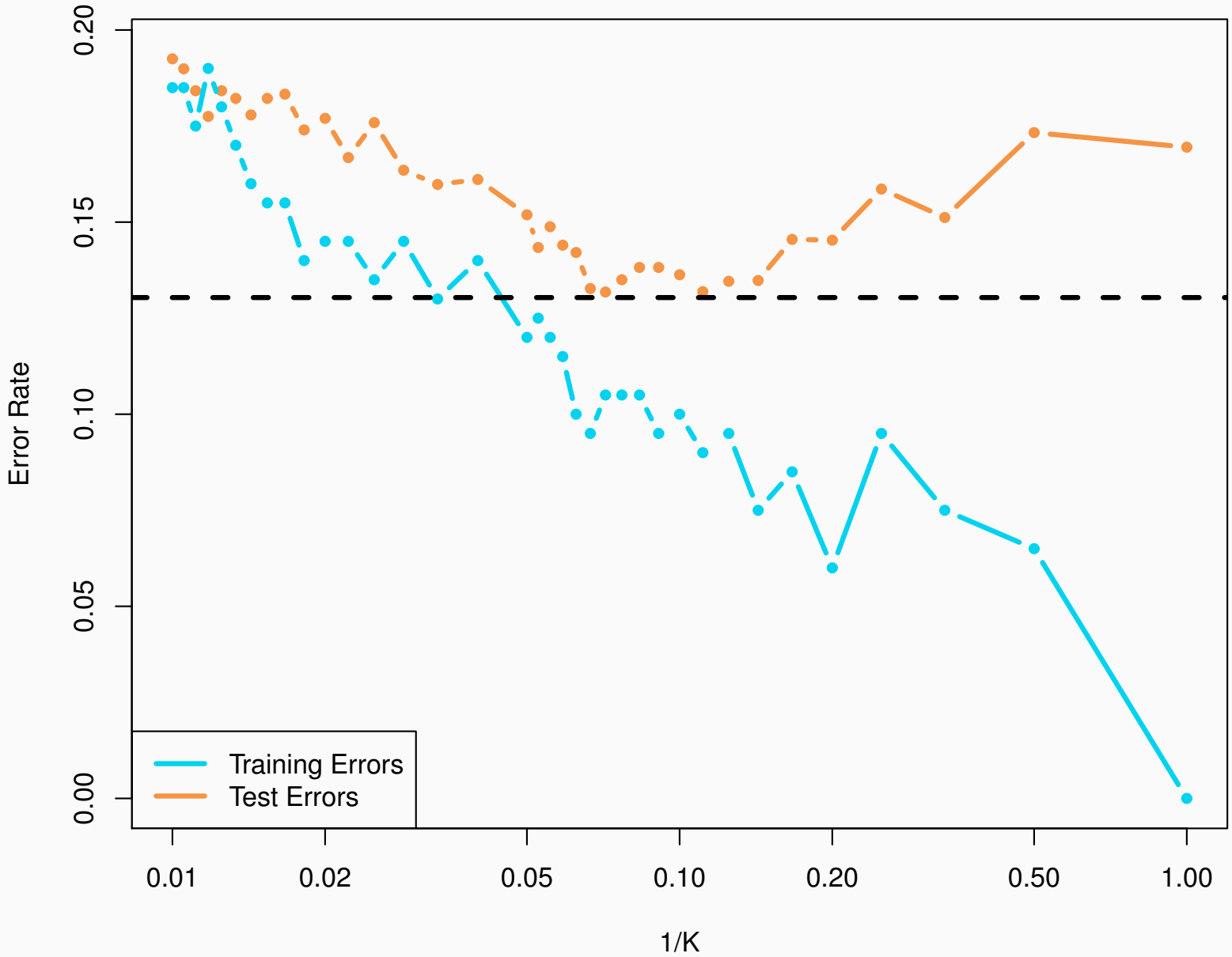
KNN: K=100



Training and test errors in classification (with KNN)



Training and test errors in classification (with KNN)



A fundamental picture

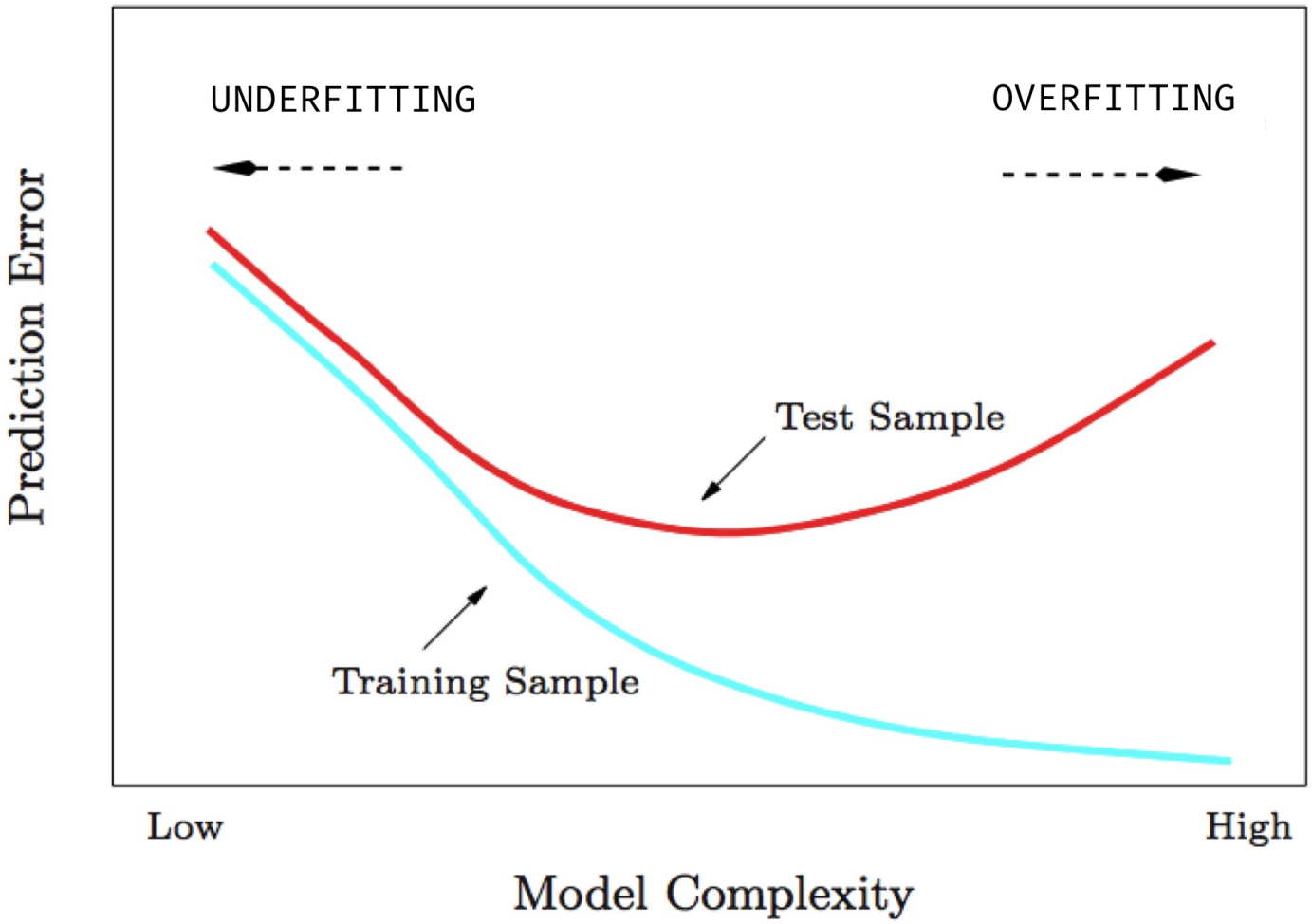


Table of contents

The bias-variance tradeoff

Let $\hat{f}$ denote the function estimated using $\mathcal{D}$, and $y = f(x) + \varepsilon$, then the expected out-of-sample error of $\hat{f}$ for a new input x_0 can be decomposed as follows:

$$\mathrm{E}[(y - \hat{f}(x_0))^2] = [\mathrm{Bias}(\hat{f}(x_0))]^2 + \mathrm{Var}(\hat{f}(x_0)) + \mathrm{Var}(\varepsilon)$$

Out-of-sample MSE = Bias^2 + Variance + Irreducible variance

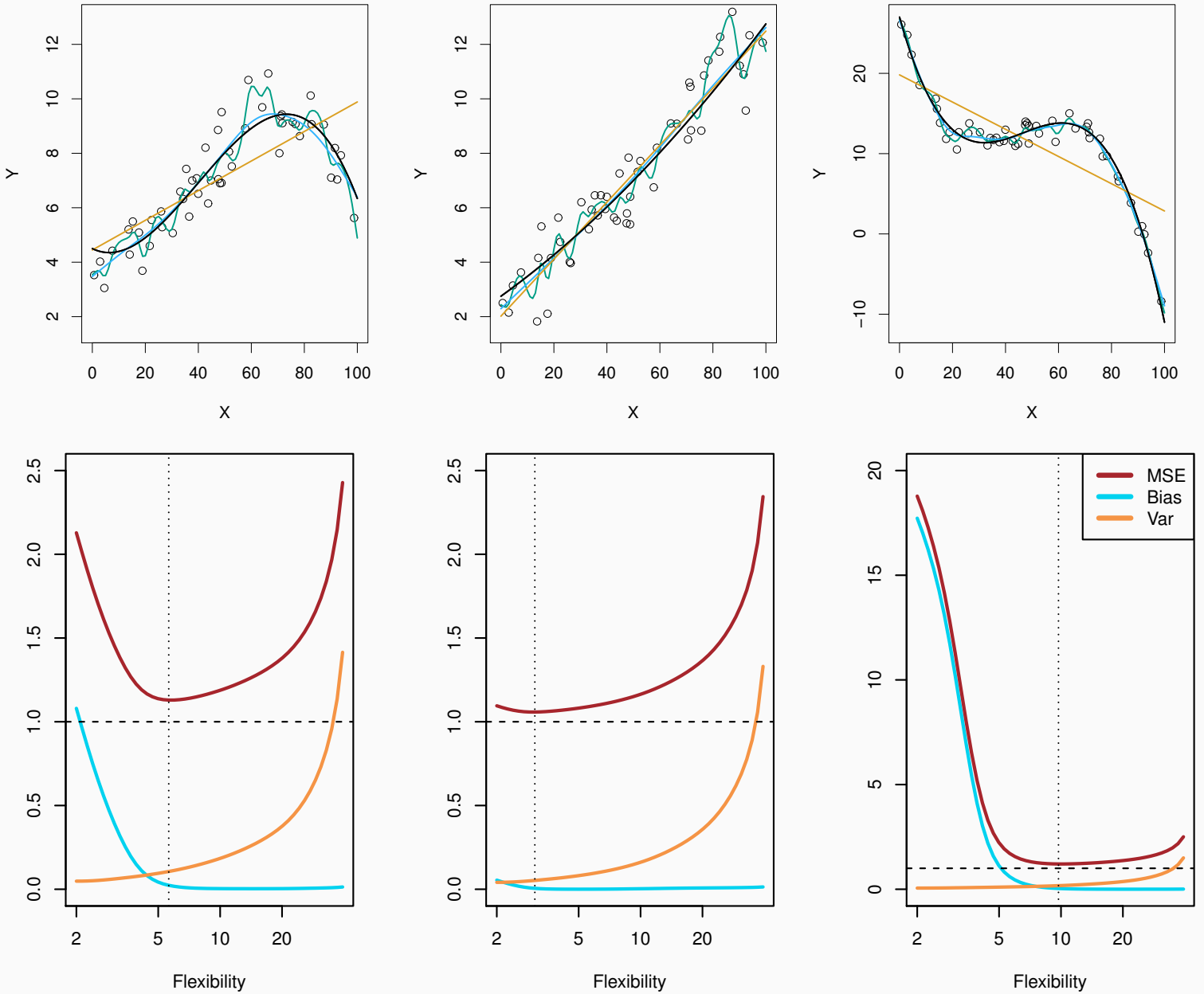
- The expectation averages over the variability of y as well as the variability in the training data.
- As the flexibility of $\hat{f}$ increases, its variance increases and its bias decreases.
- Choosing the flexibility based on average out-of-sample MSE amounts to a **bias-variance trade-off**.

Bias-variance tradeoff

There are two competing forces that govern the choice of learning method: **bias** and **variance**.

- **Bias** is the error that is introduced by modeling a complicated problem by a simpler problem.
  - For example, linear regression assumes a linear relationship when few real relationships are exactly linear.
  - In general, the more flexible a method is, the less bias it will have.
- **Variance** refers to how much your estimate would change if you had different training data.
  - In general, the more flexible a method is, the more variance it has.
  - The size of the training data has an impact on the variance.

Bias-variance tradeoff



The bias-variance tradeoff

Let us assume the data generating process (DGP) is given by $y = f(x) + \varepsilon$ where ε is a random noise term with zero mean and variance σ^2. An estimate of f, denoted $\hat{f}$, is computed using a dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$ sampled from the DGP, i.e. $y_i = f(x_i) + \varepsilon_i, i = 1, \ldots, n$.

Then the expected MSE for a new x_0 will be equal to

$$\mathbb{E}[(y - \hat{f}(x_0))^2] = \mathbb{E}[(y - \hat{f}(x_0))^2] = [\text{Bias}(\hat{f}(x_0))]^2 + \text{Var}(\hat{f}(x_0)) + \sigma^2$$

where

$$\text{Bias}(\hat{f}(x_0)) = \mathbb{E}[\hat{f}(x_0)] - f(x_0)$$

$$\text{and} \qquad \text{Var}(\hat{f}(x_0)) = \mathbb{E}\left[\left(\hat{f}(x_0) - \mathbb{E}[\hat{f}(x_0)]\right)^2\right].$$

Proof

We will abbreviate $f = f(x_0)$ and $\hat{f} = \hat{f}(x_0)$.

Since f is deterministic, $\mathbb{E}[f] = f$ and $\mathrm{Var}[f] = 0$.

$$\begin{aligned}
\mathbb{E}[(y - \hat{f})^2] &= \mathbb{E}[(y \underline{-f + f} - \hat{f})^2] \\
&= \mathbb{E}[(y - f)^2 + (f - \hat{f})^2 + 2(y - f)(f - \hat{f})] \\
&= \sigma^2 + \mathbb{E}[(\hat{f} - f)^2] + 2\mathbb{E}[(y - f)(f - \hat{f})]
\end{aligned}$$

Now

$$\begin{aligned}
\mathbb{E}[(\hat{f} - f)^2] &= \mathbb{E}[(\hat{f} \underline{-\mathbb{E}[\hat{f}] + \mathbb{E}[\hat{f}]} - f)^2] \\
&= \mathbb{E}[(\hat{f} - \mathbb{E}[\hat{f}])^2] + (\mathbb{E}[\hat{f}] - f)^2 + 2\mathbb{E}[(\hat{f} - \mathbb{E}[\hat{f}])(\mathbb{E}[\hat{f}] - f)] \\
&= \mathrm{Var}[\hat{f}] + \mathrm{Bias}^2[\hat{f}] + 2\mathbb{E}[(\hat{f} - \mathbb{E}[\hat{f}])(\mathbb{E}[\hat{f}] - f)]
\end{aligned}$$

Proof

Both cross-product terms are equal to zero as can be shown by expansion:

$$\begin{aligned}
\mathbb{E}[(y - f)(f - \hat{f})] &= \mathbb{E}[yf - f^2 - Y\hat{f} + f\hat{f}] \\
&= f^2 - f^2 - \mathbb{E}[y\hat{f}] + f\mathbb{E}[\hat{f}] \\
&= -\mathbb{E}[(f + \varepsilon)\hat{f}] + f\mathbb{E}[\hat{f}] \\
&= -\mathbb{E}[f\hat{f}] - \mathbb{E}[\varepsilon\hat{f}] + f\mathbb{E}[\hat{f}] \\
&= 0
\end{aligned}$$

$$\begin{aligned}
\mathbb{E}[(\hat{f} - \mathbb{E}[\hat{f}])(\mathbb{E}[\hat{f}] - f)] &= \mathbb{E}\left[\hat{f}\mathbb{E}[\hat{f}] - \mathbb{E}[\hat{f}]\mathbb{E}[\hat{f}] - \hat{f}f + \mathbb{E}[\hat{f}]f\right] \\
&= \mathbb{E}[\hat{f}]\mathbb{E}[\hat{f}] - \mathbb{E}[\hat{f}]\mathbb{E}[\hat{f}] - \mathbb{E}[\hat{f}]f + \mathbb{E}[\hat{f}]f \\
&= 0
\end{aligned}$$