

# Bias/variance trade-off, model assessment and model selection

Pierre Geurts and Louis Wehenkel

Institut Montefiore, University of Liège, Belgium



ELEN062-1  
Introduction to Machine Learning  
July 2020

## Supervised learning (i)

Let us assume that one is given:

- ▶ A learning sample of  $N$  input-output pairs

$$LS = \{(x_i, y_i) | i = 1, \dots, N\}, \quad x_i \in \mathcal{X}, y_i \in \mathcal{Y}$$

independently and identically drawn (*i.i.d.*) from an unknown distribution  $p(x, y)$ .

- ▶ A loss function

$$L : \mathcal{Y} \times \mathcal{Y} \rightarrow \mathbb{R}$$

measuring the discrepancy between its arguments.

One wants to find a function  $f : \mathcal{X} \rightarrow \mathcal{Y}$  that minimizes the following expectation (*generalization error*):

$$E_{x,y} \{L(y, f(x))\}$$

## Supervised learning (ii)

There are two types of problems:

- Classification:

$$L(y, y') = 1(y \neq y') \quad (\text{error rate})$$

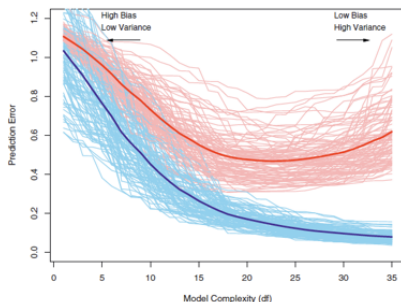
- Regression:

$$L(y, y') = (y - y')^2 \quad (\text{square error})$$

# Learning set randomness

Let us denote by  $\hat{f}_{LS}$  the function learned from a learning sample  $LS$  by a given learning algorithm.

The function  $\hat{f}_{LS}$  (its prediction at some point  $x$ ) is a random variable.



Error on the **learning sample** and **test sample** for 100 different samples.

Source: [1]

## Two quantities of interest

Given a model  $\hat{f}_{LS}$  built from some learning sample, its **generalization** error is given by:

$$Err_{LS} = E_{x,y} \left\{ L \left( y, \hat{f}_{LS}(x) \right) \right\}$$

⇒ useful for model assessment and model selection.

Given a learning algorithm, its **expected generalization** error over random learning samples of size  $N$  is given by:

$$E_{LS} \{ Err_{LS} \} = E_{LS} \left\{ E_{x,y} \left\{ L \left( y, \hat{f}_{LS}(x) \right) \right\} \right\}$$

⇒ useful to characterize a learning algorithm.

- ▶ Bias/variance trade-off: decomposition of the expected error  $E_{LS} \{Err_{LS}\}$  that helps to understand overfitting.
- ▶ Performance evaluation: procedures to estimate  $Err_{LS}$  or  $E_{LS} \{Err_{LS}\}$ .
- ▶ Performance measures: common loss functions  $L$  for classification and regression.

# Outline

- ① Bias/variance trade-off
- ② Performance evaluation
- ③ Performance measures
- ④ Further reading

- ① Bias/variance trade-off
  - Bias and variance definitions
  - Parameters that influence bias and variance
  - Bias and variance reduction techniques
- ② Performance evaluation
- ③ Performance measures
- ④ Further reading



- ▶ Bias/variance decomposition for a simple regression problem with no input.
- ▶ Extension to regression problems with inputs
- ▶ Bias/variance trade-off in classification problems

# A simple regression problem with no input (i)

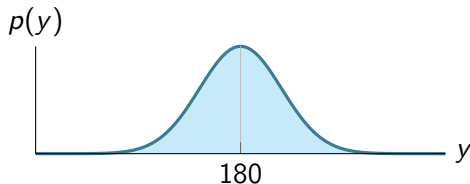
**Goal:** predict as well as possible the height of a Belgian male adult.

More formally:

- Choose an error measure (e.g. square error).
- Find an estimation  $\hat{y}$  such that

$$E_y \left\{ (y - \hat{y})^2 \right\}$$

over the whole population of Belgian male adults is minimized.



## A simple regression problem with no input (ii)

The estimation that minimizes the error can be computed by cancelling its derivative:

$$\begin{aligned}\frac{\partial}{\partial y'} E_y \{ (y - y')^2 \} &= 0 \\ \Leftrightarrow E_y \{ -2 (y - y') \} &= 0 \\ \Leftrightarrow E_y \{ y \} - E_y \{ y' \} &= 0 \\ \Leftrightarrow y' &= E_y \{ y \}\end{aligned}$$

Hence, the estimation that minimizes the error is  $E_y \{ y \}$ , which is called the **Bayes model**.

**However**, in practice, it is impossible to compute the exact value of  $E_y \{ y \}$ , as this would imply to measure the height of every Belgian male adult.

## Learning algorithms (i)

As  $p(y)$  is unknown, one needs to find an estimation  $\hat{y}$  from a sample of individuals i.i.d. from the Belgian male adult population:

$$LS = \{y_1, \dots, y_N\}$$

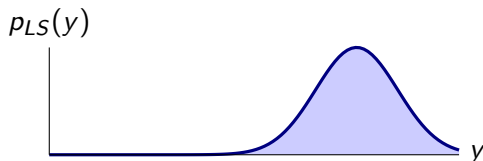
Examples:

$$- \hat{y}_1 = \frac{1}{N} \sum_{i=1}^N y_i$$

$$- \hat{y}_2 = \frac{\lambda 180 + \sum_{i=1}^N y_i}{\lambda + N}, \quad \lambda \in [0, +\infty[ \quad (\text{if it is known that the height is close to 180})$$

## Learning algorithms (ii)

As learning samples  $LS$  are randomly drawn, the prediction  $\hat{y}$  will also be a random variable:



A good learning algorithm should not be **good** only on a single learning sample but **on average** over all learning samples of size  $N$ . One thus seeks to minimize

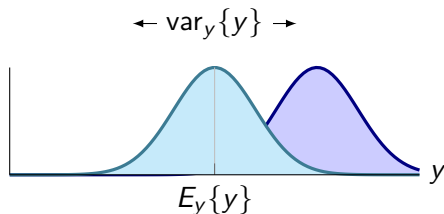
$$E = E_{LS} \left\{ E_y \left\{ (y - \hat{y})^2 \right\} \right\}$$

## Bias/variance decomposition (i)

Let us analyse this error in more details:

$$\begin{aligned} & E_{LS} \left\{ E_y \left\{ (y - \hat{y})^2 \right\} \right\} \\ = & E_{LS} \left\{ E_y \left\{ (y - E_y\{y\} + E_y\{y\} - \hat{y})^2 \right\} \right\} \\ = & E_{LS} \left\{ E_y \left\{ (y - E_y\{y\})^2 \right\} \right\} + E_{LS} \left\{ E_y \left\{ (E_y\{y\} - \hat{y})^2 \right\} \right\} \\ + & E_{LS} \left\{ E_y \left\{ 2(y - E_y\{y\})(E_y\{y\} - \hat{y}) \right\} \right\} \\ = & E_y \left\{ (y - E_y\{y\})^2 \right\} + E_{LS} \left\{ (E_y\{y\} - \hat{y})^2 \right\} \\ + & E_{LS} \left\{ 2(E_y\{y\} - E_y\{y\})(E_y\{y\} - \hat{y}) \right\} \\ = & E_y \left\{ (y - E_y\{y\})^2 \right\} + E_{LS} \left\{ (E_y\{y\} - \hat{y})^2 \right\} \end{aligned}$$

## Bias/variance decomposition (ii)



$$E = \underbrace{E_y \left\{ (y - E_y\{y\})^2 \right\}}_{= \text{residual error} = \text{var}_y\{y\}} + E_{LS} \left\{ (E_y\{y\} - \hat{y})^2 \right\}$$

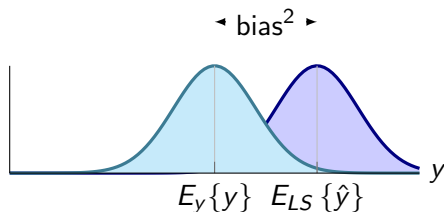
The residual error is the **minimal attainable** error.

## Bias/variance decomposition (iii)

$$\begin{aligned} & E_{LS} \left\{ (E_y\{y\} - \hat{y})^2 \right\} \\ = & E_{LS} \left\{ (E_y\{y\} - E_{LS}\{\hat{y}\} + E_{LS}\{\hat{y}\} - \hat{y})^2 \right\} \\ = & E_{LS} \left\{ (E_y\{y\} - E_{LS}\{\hat{y}\})^2 \right\} + E_{LS} \left\{ (E_{LS}\{\hat{y}\} - \hat{y})^2 \right\} \\ + & E_{LS} \left\{ 2(E_y\{y\} - E_{LS}\{\hat{y}\})(E_{LS}\{\hat{y}\} - \hat{y}) \right\} \\ = & (E_y\{y\} - E_{LS}\{\hat{y}\})^2 + E_{LS} \left\{ (\hat{y} - E_{LS}\{\hat{y}\})^2 \right\} \\ + & 2(E_y\{y\} - E_{LS}\{\hat{y}\})(E_{LS}\{\hat{y}\} - E_{LS}\{\hat{y}\}) \\ = & (E_y\{y\} - E_{LS}\{\hat{y}\})^2 + E_{LS} \left\{ (\hat{y} - E_{LS}\{\hat{y}\})^2 \right\} \end{aligned}$$



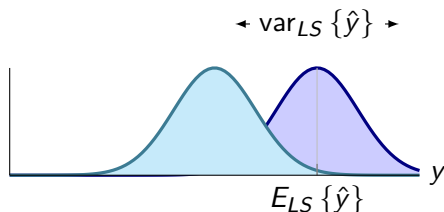
## Bias/variance decomposition (iv)



$$E = \text{var}_y\{y\} + \underbrace{\left( E_y\{y\} - \overbrace{E_{LS}\{\hat{y}\}}^{\text{= average model}} \right)^2}_{\text{= bias}^2} + \dots$$

where the bias is the **difference** between the average model and the Bayes model.

## Bias/variance decomposition (v)



$$E = \text{var}_y\{y\} + \text{bias}^2 + \underbrace{E_{LS} \left\{ (\hat{y} - E_{LS} \{\hat{y}\})^2 \right\}}_{= \text{estimation variance} = \text{var}_{LS} \{\hat{y}\}}$$

where  $\text{var}_{LS} \{\hat{y}\}$  is a consequence of **overfitting**.

The **expected generalization** error can thus be rewritten as:

$$E = \text{var}_y\{y\} + \text{bias}^2 + \text{var}_{LS}\{\hat{y}\}$$

## Application to a simple example (i)

Let us first consider a model  $\hat{y}_1$  computing the average height over the learning sample:

$$\hat{y}_1 = \frac{1}{N} \sum_{i=1}^N y_i$$

One can easily compute:

$$\begin{aligned} \text{bias}^2 &= (E_y\{y\} - E_{LS}\{\hat{y}_1\})^2 = 0 \\ \text{var}_{LS}\{\hat{y}_1\} &= \frac{1}{N} \text{var}_y\{y\} \end{aligned}$$

From statistics,  $\hat{y}_1$  is the best estimate with zero bias.

## Application to a simple example (ii)

Let us now consider a model where it is known that the height is close to 180:

$$\hat{y}_2 = \frac{\lambda 180 + \sum y_i}{\lambda + N}$$

One can compute:

$$\begin{aligned} \text{bias}^2 &= \left( \frac{\lambda}{\lambda + N} \right)^2 (E_y\{y\} - 180)^2 \\ \text{var}_{LS} \{\hat{y}_2\} &= \frac{N}{(\lambda + N)^2} \text{var}_y\{y\} \end{aligned}$$

From these, it can be observed that  $\hat{y}_1$  may not be the best estimator because of its variance. There is a **bias/variance trade-off** with respect to  $\lambda$ .

## Bayesian approach (i)

Let us assume that:

- ▶ The average height is close to 180cm:

$$P(\bar{y}) = A \exp\left(-\frac{(\bar{y} - 180)^2}{2\sigma_{\bar{y}}^2}\right)$$

- ▶ The height of one individual is Gaussian around the mean:

$$P(y_i | \bar{y}) = B \exp\left(-\frac{(y_i - \bar{y})^2}{2\sigma_y^2}\right)$$

What is the most probable value of  $\bar{y}$  after having seen the learning sample?

$$\hat{y} = \arg \max_{\bar{y}} P(\bar{y} | LS)$$

## Bayesian approach (ii)

$$\begin{aligned}\hat{y} &= \arg \max_{\bar{y}} P(\bar{y} | LS) \\ &= \arg \max_{\bar{y}} P(LS | \bar{y}) P(\bar{y}) \quad (\text{Bayes theorem and } P(LS) \text{ is constant}) \\ &= \arg \max_{\bar{y}} P(y_1, \dots, y_N | \bar{y}) P(\bar{y}) \\ &= \arg \max_{\bar{y}} \prod_{i=1}^N P(y_i | \bar{y}) P(\bar{y}) \quad (\text{independence of learning cases}) \\ &= \arg \min_{\bar{y}} - \sum_{i=1}^N \log(P(y_i | \bar{y})) - \log(P(\bar{y}))\end{aligned}$$

## Bayesian approach (iii)

$$= \arg \min_{\bar{y}} \sum_{i=1}^N \frac{(y_i - \bar{y})^2}{2\sigma_y^2} + \frac{(\bar{y} - 180)^2}{2\sigma_{\bar{y}}^2}$$

= ...

$$= \frac{\lambda 180 + \sum_i y_i}{\lambda + N} \text{ with } \lambda = \frac{\sigma_y^2}{\sigma_{\bar{y}}^2}$$



## Extension to a problem with inputs (i)

In this case,  $\hat{y}(\underline{x})$  is a function of several inputs  $\Rightarrow$  average over the whole input space.

The error becomes:

$$E_{\underline{x},y} \left\{ (y - \hat{y}(\underline{x}))^2 \right\}$$

When averaging over all learning sets:

$$\begin{aligned} E &= E_{LS} \left\{ E_{\underline{x},y} \left\{ (y - \hat{y}(\underline{x}))^2 \right\} \right\} \\ &= E_{\underline{x}} \left\{ E_{LS} \left\{ E_{y|\underline{x}} \left\{ (y - \hat{y}(\underline{x}))^2 \right\} \right\} \right\} \\ &= E_{\underline{x}} \left\{ \text{var}_{y|\underline{x}} \{y\} \right\} + E_{\underline{x}} \left\{ \text{bias}^2(\underline{x}) \right\} + E_{\underline{x}} \left\{ \text{var}_{LS} \{ \hat{y}(\underline{x}) \} \right\} \end{aligned}$$

## Extension to a problem with inputs (ii)

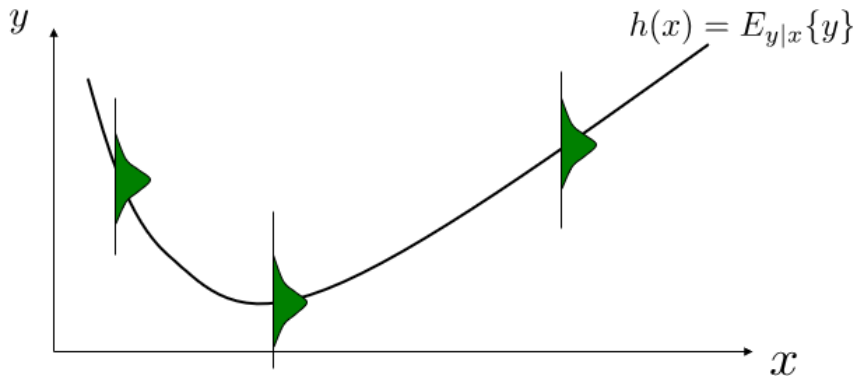
$$E_{LS} \{ E_{y|\underline{x}} \{ (y - \hat{y}(\underline{x}))^2 \} \} = \text{noise}(\underline{x}) + \text{bias}^2(\underline{x}) + \text{variance}(\underline{x})$$

- ▶  $\text{noise}(\underline{x}) = E_{y|\underline{x}} \{ (y - h_B(\underline{x}))^2 \}$ :  
Quantifies how much  $y$  varies from  $h_B(\underline{x}) = E_{y|\underline{x}}\{y\}$  (*Bayes model*).
- ▶  $\text{bias}^2(\underline{x}) = (h_B(\underline{x}) - E_{LS} \{ \hat{y}(\underline{x}) \})^2$ :  
Measures the error between the Bayes model and the average model.
- ▶  $\text{variance}(\underline{x}) = E_{LS} \{ (\hat{y}(\underline{x}) - E_{LS} \{ \hat{y}(\underline{x}) \})^2 \}$ :  
Quantifies how much  $\hat{y}(\underline{x})$  varies from one learning sample to another.

## Illustration (i)

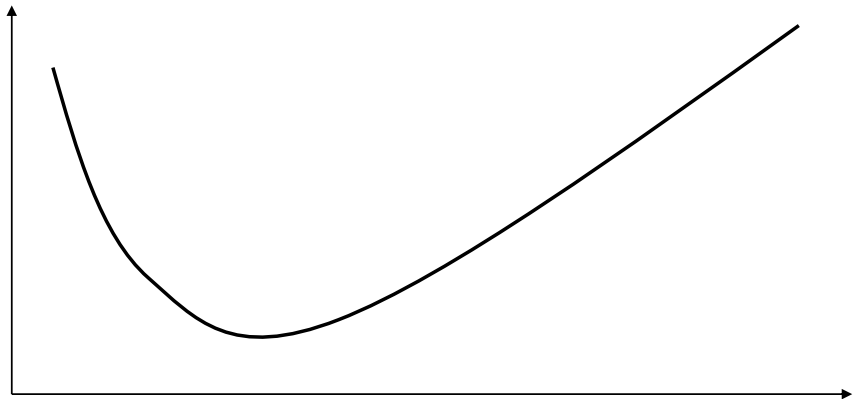
Let us consider the following problem:

- A single input  $x$ , which is a uniform random variable drawn in  $[0, 1]$ .
- $y = h(x) + \varepsilon$  where  $\varepsilon \sim \mathcal{N}(0, 1)$ .



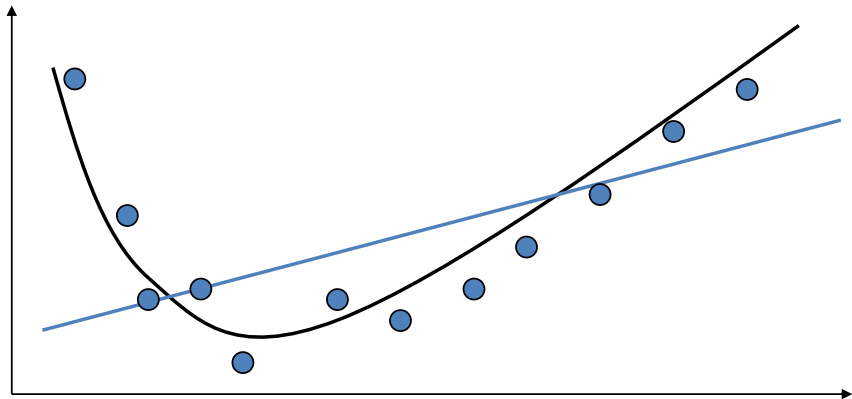
## Illustration (ii)

A **low variance** and **high bias** method leads to **underfitting**.



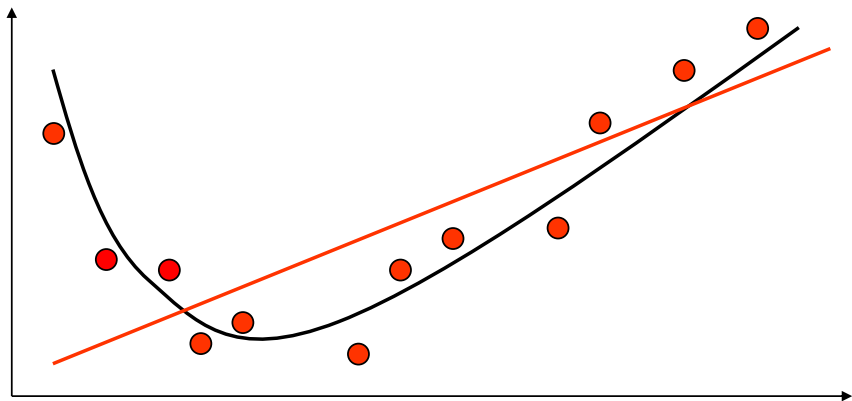
## Illustration (ii)

A **low variance** and **high bias** method leads to **underfitting**.



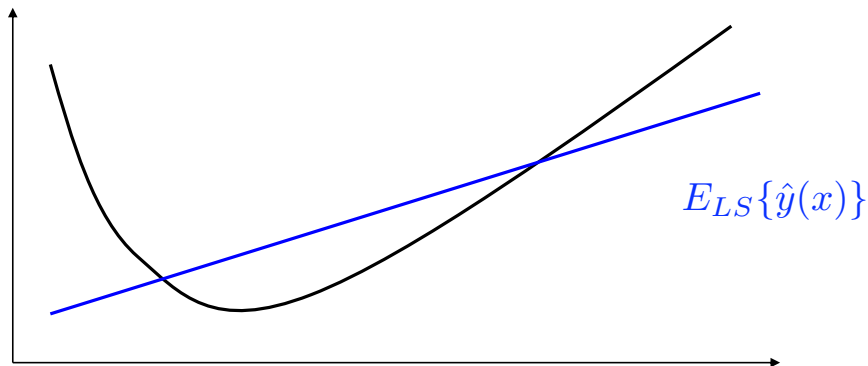
## Illustration (ii)

A **low variance** and **high bias** method leads to **underfitting**.



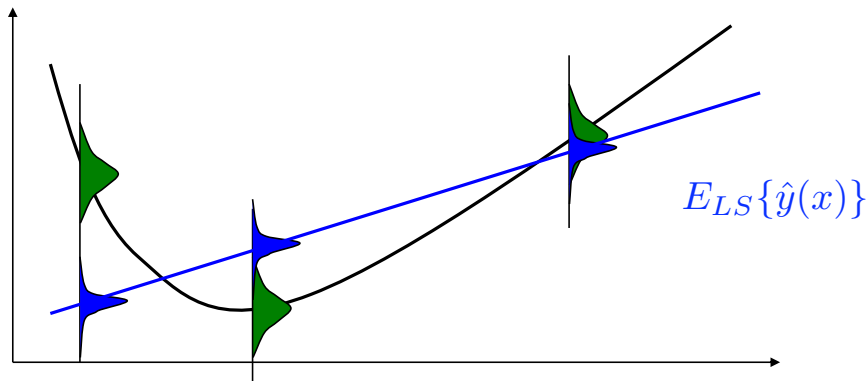
## Illustration (ii)

A **low variance** and **high bias** method leads to **underfitting**.



## Illustration (ii)

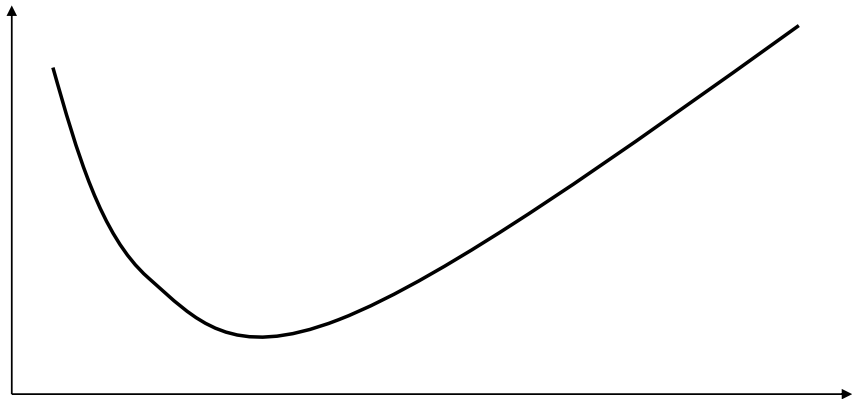
A **low variance** and **high bias** method leads to **underfitting**.





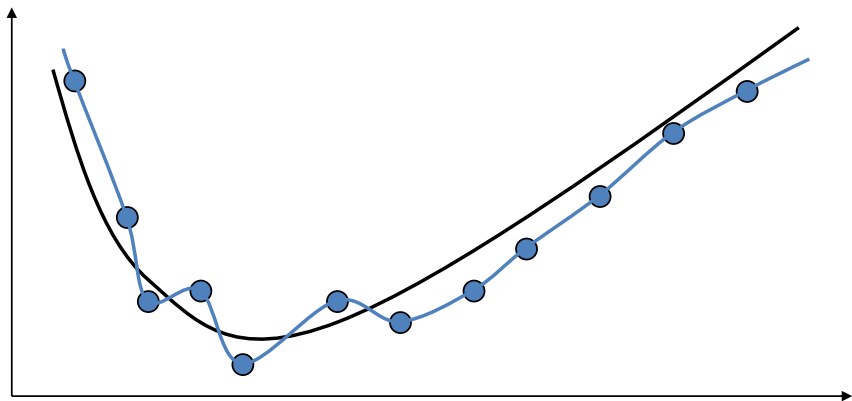
## Illustration (iii)

A **low bias** and **high variance** method leads to **overfitting**.



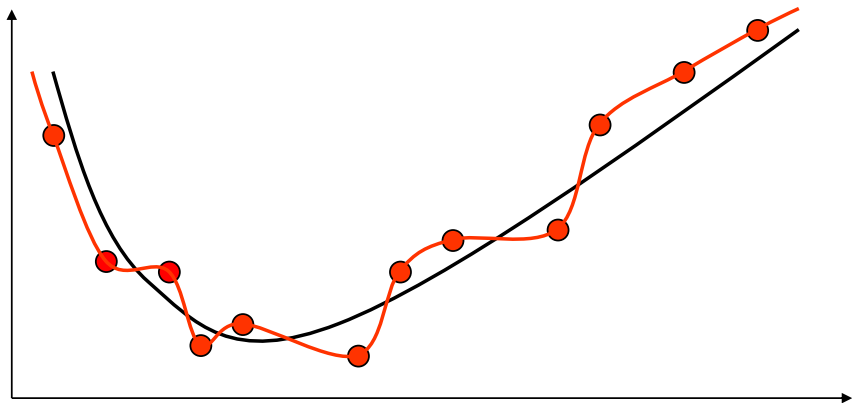
## Illustration (iii)

A **low bias** and **high variance** method leads to **overfitting**.



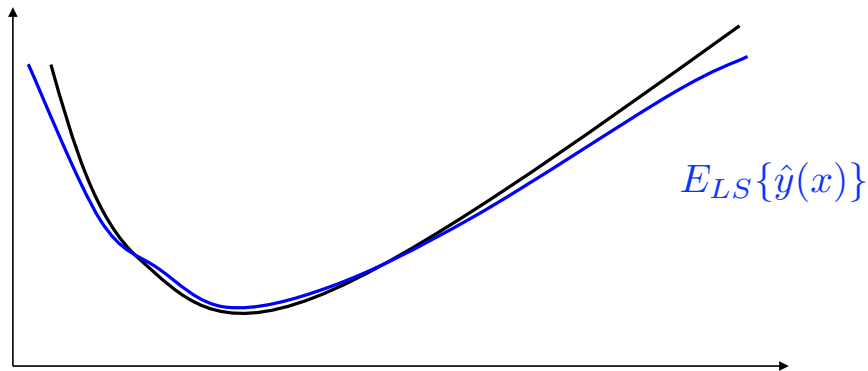
## Illustration (iii)

A **low bias** and **high variance** method leads to **overfitting**.



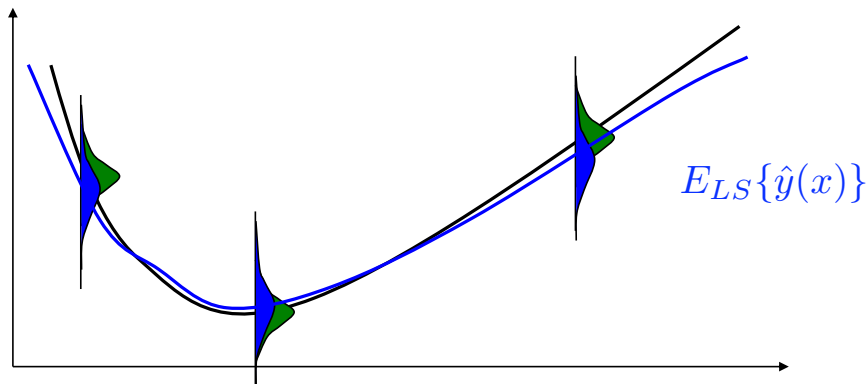
## Illustration (iii)

A **low bias** and **high variance** method leads to **overfitting**.



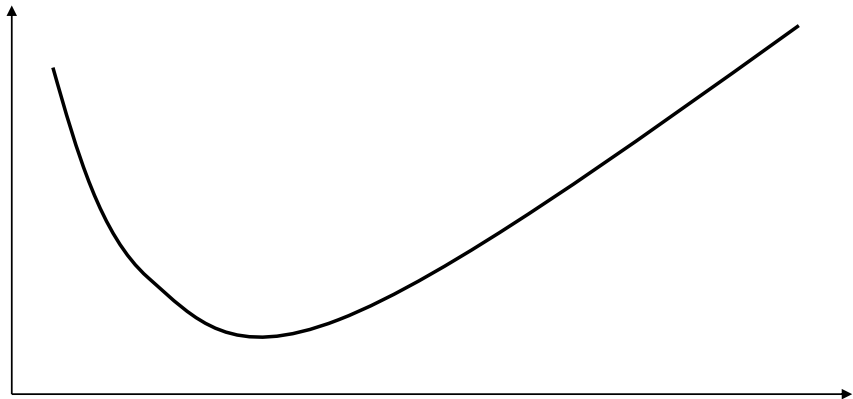
## Illustration (iii)

A **low bias** and **high variance** method leads to **overfitting**.



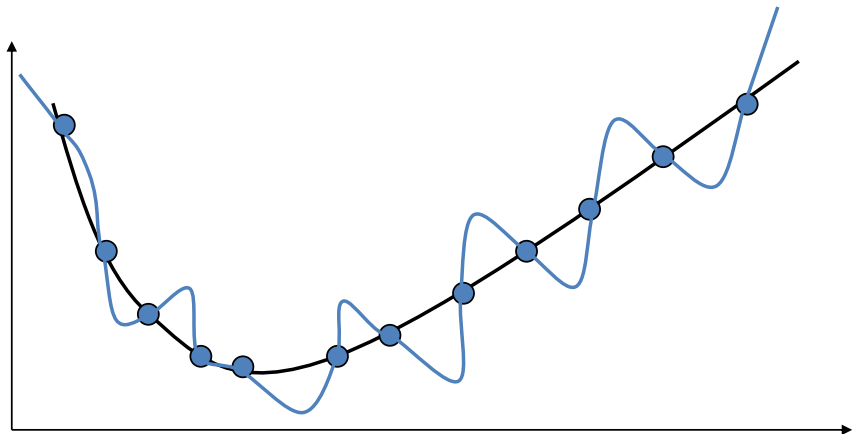
## Illustration (iv)

**No noise** doesn't imply no variance, rather **less variance**.



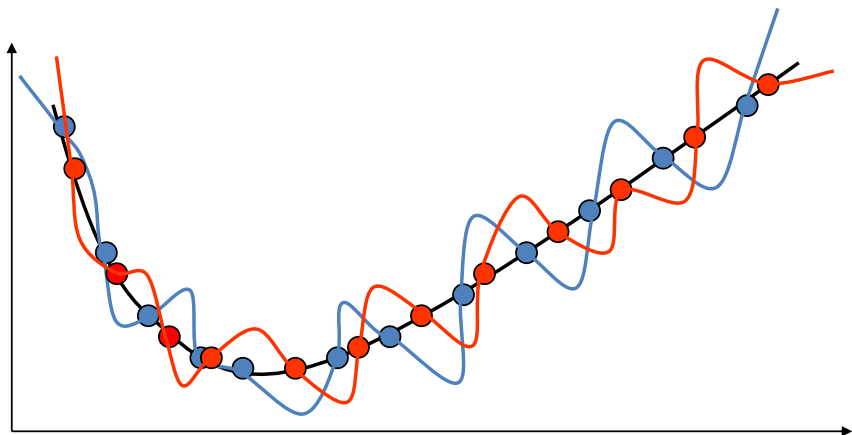
## Illustration (iv)

**No noise** doesn't imply no variance, rather **less variance**.



## Illustration (iv)

No noise doesn't imply no variance, rather **less variance**.





## Bias/variance trade-off in classification problems (i)

The mean misclassification error corresponds to:

$$E = E_{LS} \{ E_{\underline{x}, y} \{ 1(y \neq \hat{y}(\underline{x})) \} \}$$

The best possible model is the **Bayes model**:

$$h_B(\underline{x}) = \arg \max_c P(y = c | \underline{x})$$

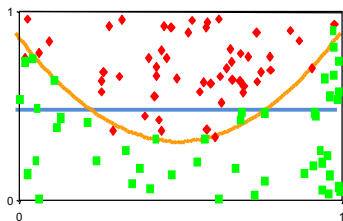
The “average” model is:

$$\arg \max_c P(\hat{y}(\underline{x}) = c | \underline{x})$$

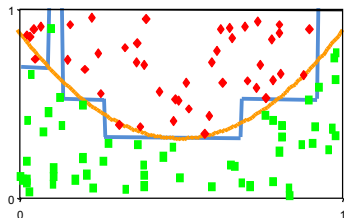
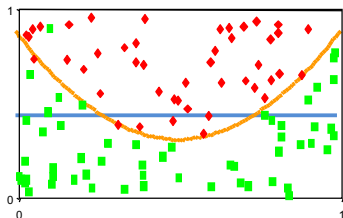
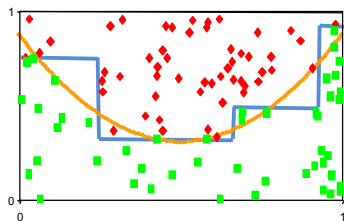
**Unfortunately**, there is no such decomposition of the mean misclassification error into a bias and variance terms. Nevertheless, the same phenomena can be observed.

## Bias/variance trade-off in classification problems (ii)

Single test node

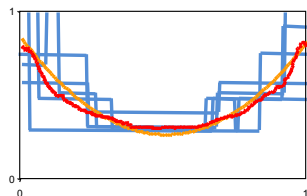
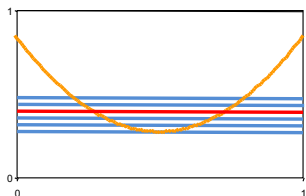


Fully grown tree



Decision tree classifiers on two different data sets (top and bottom).

## Bias/variance trade-off in classification problems (iii)



The **bias** is a **systematic** error component, which is independent of the learning sample. Rather, it depends on the **learning algorithm**.

The **variance** is the error due to the **variability** of the model with respect to the **learning sample** randomness.

There are errors due to bias and errors due to variance.

## Parameters that influence bias and variance - Outline

- ▶ Complexity of the model
- ▶ Complexity of the Bayes model
- ▶ Noise
- ▶ Learning sample size
- ▶ Learning algorithm

# Illustrative problem

Let us consider the following artificial problem:

- 10 inputs, all uniform random variables drawn in  $[0, 1]$ .
- The true function depends only on 5 inputs:

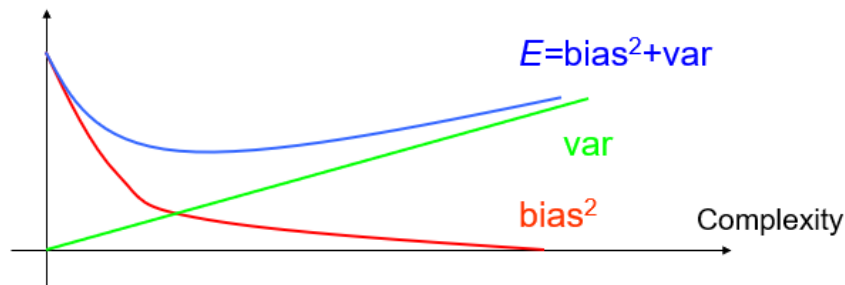
$$y(\underline{x}) = 10 \sin(\pi x_1 x_2) + 20(x_3 - 0.5)^2 + 10x_4 + 6x_5 + \varepsilon$$

where  $\varepsilon$  is a random variable drawn in  $\mathcal{N}(0, 1)$ .

The following experimentations are conducted:

1.  $E_{LS} \rightarrow$  Average over 50 learning sets of size 500.
  2.  $E_{\underline{x}, y} \rightarrow$  Average over 2000 cases
- $\Rightarrow$  Estimate bias and variance, as well as the residual error.

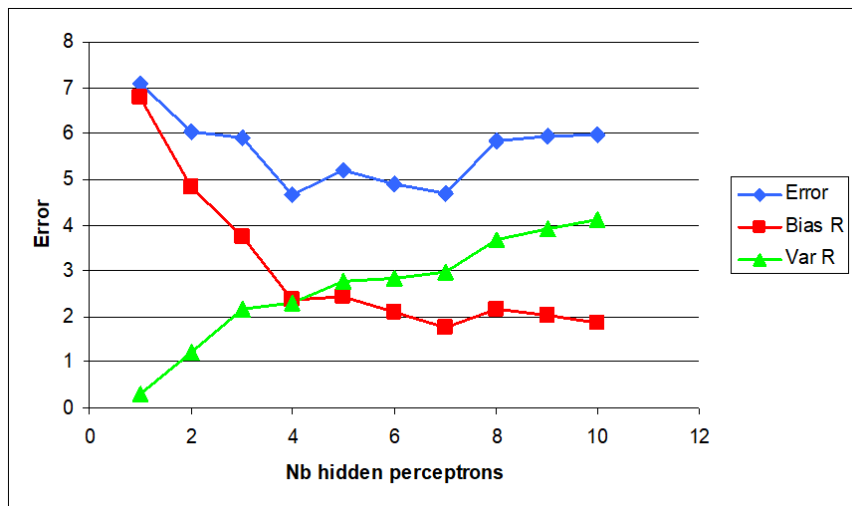
## Complexity of the model



Usually, the bias is a decreasing function of the complexity, while the variance is an increasing function of the complexity.

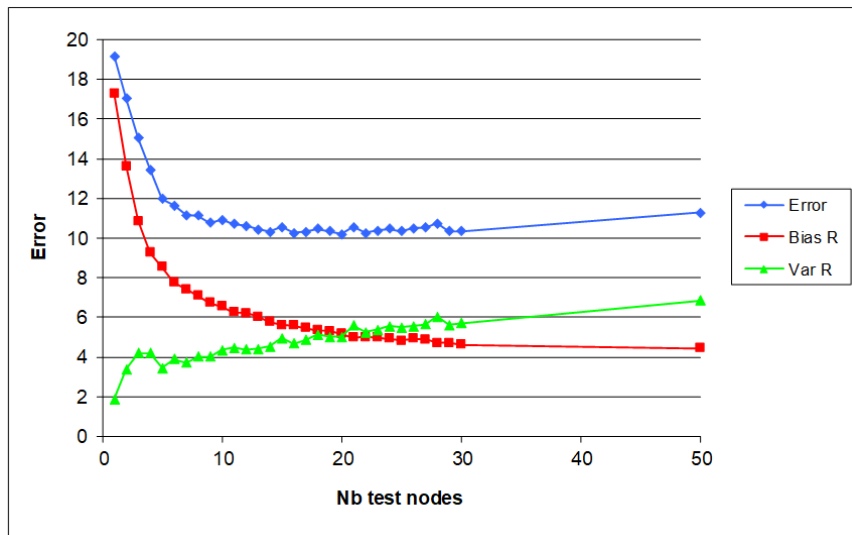
*Note:* the residual error is included in the bias term.

# Complexity of the model - Neural networks



Error, bias and variance w.r.t. the number of neurons in the hidden layer.

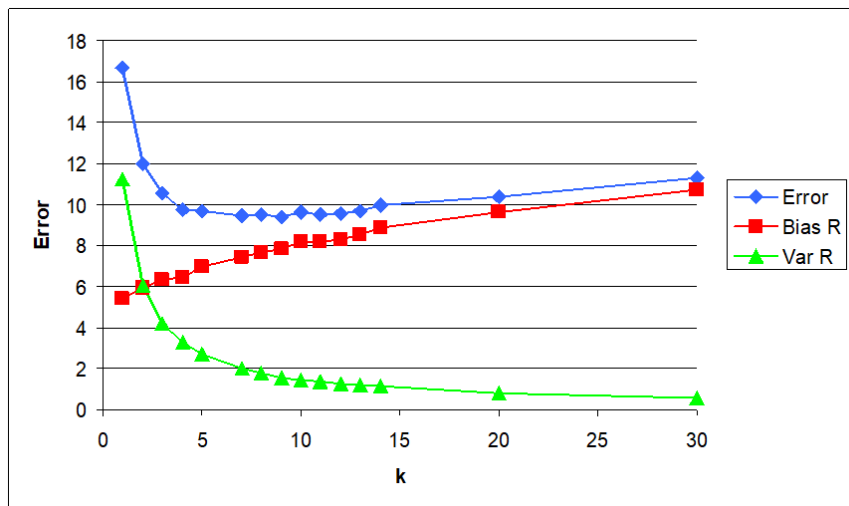
# Complexity of the model - Regression trees



Error, bias and variance w.r.t. the number of test nodes.



## Complexity of the model - kNN



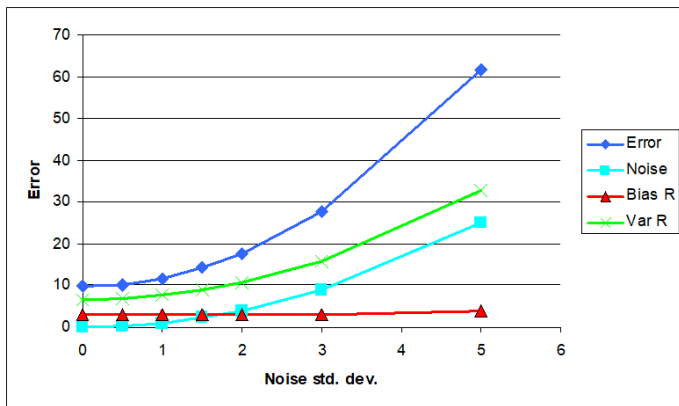
Error, bias and variance w.r.t. the number of neighbors.

## Complexity of the Bayes model

At fixed model complexity, the bias **increases** with the complexity of the Bayes model. Nevertheless, the effect on variance is difficult to predict.

# Noise

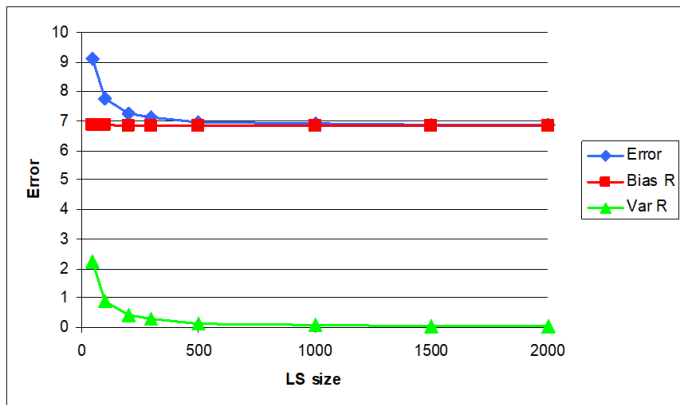
Variance **increases** with noise while bias remains mainly unaffected.



Influence of noise on full regression trees.

## Learning sample size (i)

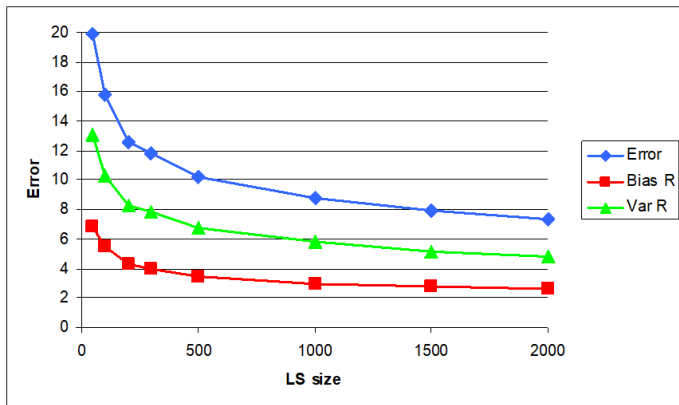
At fixed model complexity, bias remains **constant** and variance **decreases** with the learning sample size.



Influence of the learning sample size on linear regression.

## Learning sample size (ii)

When the complexity of the model is dependent on the learning sample size, **both** bias and variance **decrease** with the learning sample size.



Influence of the learning sample size on regression trees.

## Learning algorithms - Linear regression

Method	$E^2$	bias <sup>2</sup> + noise	var
Linear regression	7.0	6.8	0.2
kNN ( $k = 1$ )	15.4	5.0	10.4
kNN ( $k = 10$ )	8.5	7.2	1.3
MLP (10)	2.0	1.2	0.8
MLP (10 - 10)	4.6	1.4	3.2
Regression tree	10.2	3.5	6.7

Linear regression has few parameters. Therefore, it has a small variance. However, since the true function is non-linear, it has a high bias.

## Learning algorithms - kNN

Method	$E^2$	bias <sup>2</sup> + noise	var
Linear regression	7.0	6.8	0.2
kNN ( $k = 1$ )	15.4	5.0	10.4
kNN ( $k = 10$ )	8.5	7.2	1.3
MLP (10)	2.0	1.2	0.8
MLP (10 - 10)	4.6	1.4	3.2
Regression tree	10.2	3.5	6.7

For small values of  $k$ , there is a high variance and a moderate bias. For high values of  $k$ , the variance decreases but bias increases.

## Learning algorithms - MLP

Method	$E^2$	bias <sup>2</sup> + noise	var
Linear regression	7.0	6.8	0.2
kNN ( $k = 1$ )	15.4	5.0	10.4
kNN ( $k = 10$ )	8.5	7.2	1.3
MLP (10)	2.0	1.2	0.8
MLP (10 - 10)	4.6	1.4	3.2
Regression tree	10.2	3.5	6.7

In both cases, the bias is low. However, variance increases with the model complexity.



## Learning algorithms - Regression tree

Method	$E^2$	bias <sup>2</sup> + noise	var
Linear regression	7.0	6.8	0.2
kNN ( $k = 1$ )	15.4	5.0	10.4
kNN ( $k = 10$ )	8.5	7.2	1.3
MLP (10)	2.0	1.2	0.8
MLP (10 - 10)	4.6	1.4	3.2
Regression tree	10.2	3.5	6.7

Regression trees have a small bias. Indeed, a complex enough tree can approximate any non-linear function. However, it has a high variance.

# Bias and variance reduction techniques - Outline

- ▶ Introduction
- ▶ Dealing with the bias/variance trade-off of one algorithm
- ▶ Ensemble methods

# Bias and variance reduction techniques

In the context of a given method, adapting the learning algorithm to find the best trade-off between bias and variance is a solution, but not a panacea.

**Examples:** pruning, weight decay.

Ensemble methods change the bias/variance trade-off, but destroy some features of the original method.

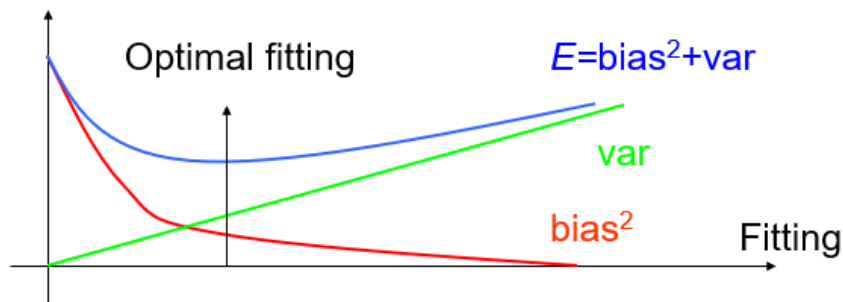
**Examples:** bagging, boosting.

## Variance reduction for a given model (i)

**Idea:** reduce the ability of the learning algorithm to fit the learning sample.

- ▶ Pruning: reduces the model complexity explicitly
- ▶ Early stopping: reduces the amount of search
- ▶ Regularization: reduces the size of the hypothesis space.  
**Example:** weight decay in neural networks consists in penalizing high weight values.

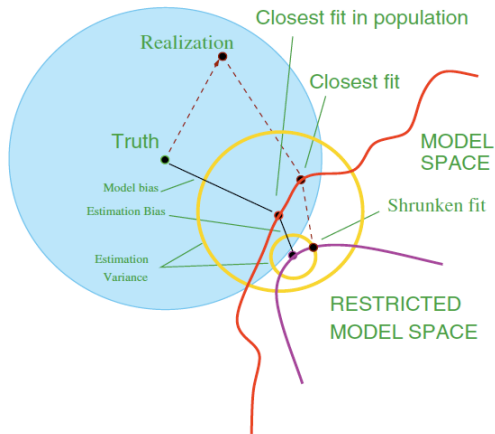
## Variance reduction for a given model (ii)



The selection of the optimal level of fitting can be done:

- a priori (**not optimal**)
- by cross-validation (less efficient):
  - $\text{bias}^2 \approx$  error on the learning set.
  - $E \approx$  error on an independent test set.

# Variance reduction for a given model (iii)



Source: [1]

## Variance reduction for a given model (iv)

### Examples:

- Post-pruning of regression trees.
- Early stopping of MLP by cross-validation.

Method	E	Bias	Variance
Full regression tree (250)	10.2	3.5	6.7
Pruned regression tree (45)	9.1	4.3	4.8
Full learned MLP	4.6	1.4	3.2
Early stopped MLP	3.8	1.5	2.3

As expected, variance decreases but bias increases.

## Ensemble methods (i)

Ensemble methods combine the predictions of several models built with a learning algorithm in order to improve with respect to the use of a single model.

There are two main families:

1. **Averaging techniques**: they grow several models independently and average their predictions. They mainly decrease **variance**.  
**Examples**: bagging, random forests.
2. **Boosting type algorithms**: they grow several models sequentially. They mainly decrease **bias**.  
**Examples**: adaboost, MART.



## Ensemble methods (ii)

**Examples:** bagging, boosting, random forests.

Method	E	Bias	Variance
Full regression tree	10.2	3.5	6.7
Bagging	5.3	3.8	1.5
Random forests	4.9	4.0	0.9
Boosting	5.0	3.1	1.9

# Discussion

The notions of bias and variance are very useful to predict how changing the (learning and problem) parameters will affect accuracy. This explains why very simple methods can work much better than more complex ones on very difficult tasks.

Variance reduction is a very important topic: reducing bias is easy, while keeping variance low is not as easy. It is especially important when machine learning is applied in domains that require complex models: time series analysis, computer vision, natural language processing, . . .

All learning algorithms are **not** equal in terms of variance. Trees are among the worse methods according to this criterion.

# Outline

- ① Bias/variance trade-off
- ② Performance evaluation
  - Model assessment and selection
  - Cross-validation
  - Bootstrap
  - CV-based model selection
- ③ Performance measures
- ④ Further reading

# Estimating the performance of a model

Given a model learned from some data set of size  $N$ , how to estimate its performance from this data set?

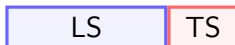
What for?

- ▶ Model selection: choosing the best model among several models.  
**Example**: determining the right complexity of a model or choosing between different learning algorithms.
- ▶ Model assessment: having chosen a final model, it consists in estimating its performance on new data.

# Large data sets: test set method

**Idea:** randomly divide the data set into two parts: a **learning set** and a **test set**.

**Example:** 70% – 30%



Method:

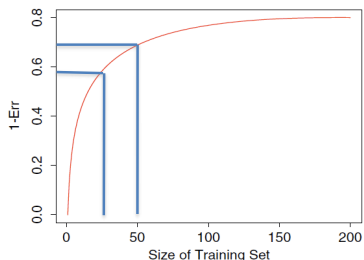
1. Fit the model on the learning set
2. Test it on the test set

The resulting estimate is an estimate of the error of a model learned on the **whole** data set.

## Small data sets

In this case, the test set error is **unreliable** because it is based on a small sample: 30% of an already small data set.

It is also pessimistically biased as an estimate of the error of a model built on the whole data set: for small sample sizes, a model learned on 70% of the data is significantly less good than a model learned on the whole data.



Learning curve: performance vs learning set size

# k-fold cross-validation

**Idea:** randomly divide the data set into  $k$  subsets (e.g.  $k = 10$ ).



Method:

- For each subset:
  1. Learn the model on the objects that are not in the subset.
  2. Compute a prediction with this model for the points in the subset.
- Report the mean error over these predictions.

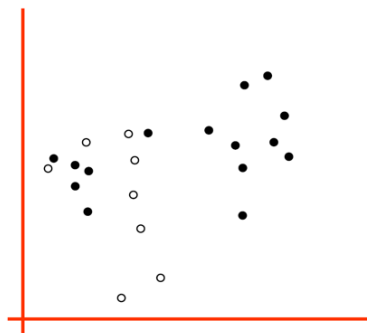
When  $k = N$ , the method is called **leave-one-out** cross-validation.

## Choosing a value for $k$

- ▶  $k = N$ :
  - Unbiased: removing one object does not change much the size of the learning sample.
  - High variance: highly data set dependent.
  - Slow: requires to train  $N$  models.
- ▶  $k = 5, 10$ :
  - Lower variance and faster: only 5 – 10 models on fewer data.
  - Potentially biased: see learning curve.



# Exercise



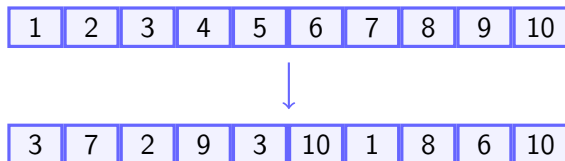
Source: [2]

In this classification problem with two inputs:

- ▶ What is the resubstitution error (LS error) of 1-NN?
- ▶ What is the LOO error of 1-NN?
- ▶ What is the LOO error of 3-NN?
- ▶ What is the LOO error of 22-NN?

## Bootstrap (i)

Bootstrap sampling consists in sampling **with replacement**.



Some objects do not appear and some others appear several times:

$$P(o_i \in \text{bootstrap}) = 1 - \left(1 - \frac{1}{N}\right)^N \approx 1 - \frac{1}{e} = 0.632$$

## Bootstrap (ii)

Let us define the **bootstrap error estimate** method:

- For  $i = 1$  to  $B$ :
  1. Take a bootstrap sample  $B_i$  from the data set.
  2. Learn a model  $f_i$  on it.
- For each object, compute the expected error of all models that were built without it (about 30%).
- Average over all objects.

Some improvements:

- ▶ “.632 bootstrap”, which corrects for the learning curve.
- ▶ “.632+ bootstrap”, which corrects for overfitting.

## Conditional vs Expected test errors

Conditional test error (for a given model  $\hat{f}_{LS}$ ):

$$\text{Err}_{LS} = E_{x,y} \left\{ L \left( y, \hat{f}_{LS}(x) \right) \right\}$$

Expected test error:

$$E_{LS} \{ \text{Err}_{LS} \} = E_{LS} \left\{ E_{x,y} \left\{ L \left( y, \hat{f}_{LS}(x) \right) \right\} \right\}$$

Only the test set method estimates the first error. Cross-validation estimates the second one (even leave-one-out).

## Model selection: typical scenario

Given a data set of  $N$  objects (input-output pairs), how to best exploit this data set to obtain:

- ▶ The best possible model (e.g. among regression trees and k-NN)  
→ **model selection**
- ▶ An estimate of its prediction error → **model assessment**

Again, the solution depends on the size  $N$  of the data set.

# Large data sets: test set method

**Idea:** randomly divide the data set into 3 parts:

1. A learning set  $LS$
2. A validation set  $VS$
3. A test set  $TS$

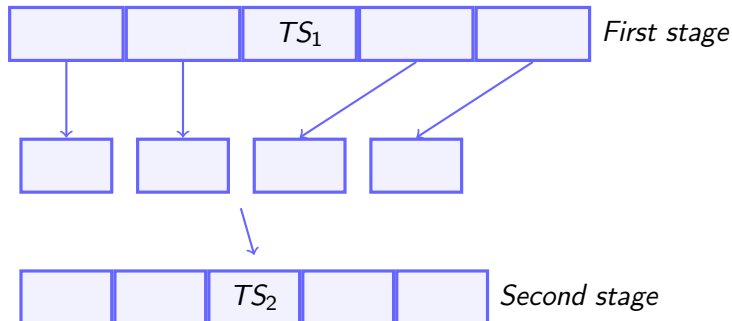
**Example:** 50% – 25% – 25%



1. Fit the models to compare on the learning set, using different algorithms or different complexity values.
2. Select the best one based on its performance on the validation set.
3. Retrain this model on  $LS + VS$ .
4. Test it on the test set → **performance estimate**.
5. Retrain this model on  $LS + VS + TS$ . This yields the finally chosen model.

## Small data sets: cross-validation

**Idea:** use two stages of  $k$ -fold cross-validation.



The **first** stage is used for the **assessment** of the final model, while the **second** one is used for **model selection**.

*Note:* we could also combine test set and cross-validation.

## Why do we need a validation set/second stage? (i)

How well does the second stage (resp. validation set) error estimate the true error?

When comparing many complex models, the probability of finding a good only one by chance is high.

Therefore, this estimated error is expected to be **overly optimistic**.



## Why do we need a validation set/second stage? (ii)

Let us consider the following example:

- $N = 50$
- 1 000 input variables
- Input variables are unrelated to the class. Their values are i.i.d. from  $\mathcal{N}(0, 1)$ .

⇒ any model should have a 50% generalization error rate.

Let us now compare 1 000 learning algorithms: the  $i$ -th algorithm learns a decision tree on the  $i$ -th feature only. For each of them, the 10-fold cross-validation error is computed.

→ 10-fold cross-validation error of the best model  $\approx 16\%$

→ Its error on a test sample of 5 000 cases  $\approx 48\%$

## Why do we need a validation set/second stage? (iii)

This is the idea behind **selection bias**.

**General rule:** any choice made using the output should be inside a cross-validation loop.

Let us consider another example on the same data set:

- ▶ Select the 10 attributes that are the most correlated with the output in the learning set.
- ▶ Estimate the error rate of a tree built with these 10 attributes using 10-fold cross-validation on the same learning set: 20%
- ▶ Estimate the error of this model on a sample of 5 000 cases: 51%

# Analytical methods for model selection

**Idea:** find the model that minimizes a criterion, typically of the form:

$$\text{Err}(\text{LS}) + G(\text{Complexity})$$

where  $G$  is a monotonically increasing function.

The criterion is derived from theoretical arguments.

**Example:** the minimum description length approach is motivated from coding theory.

Advantage:

- Cheap: no need for retraining

Drawbacks:

- OK for model selection but not for model assessment.
- May miss the true optimum in the finite sample case.

# Outline

- ① Bias/variance trade-off
- ② Performance evaluation
- ③ Performance measures
  - Classification
  - Regression
  - Loss functions for learning
- ④ Further reading

# Performance criteria

	True class	Model 1	Model 2
1	Negative	Positive	Negative
2	Negative	Negative	Negative
3	Negative	Positive	Positive
4	Negative	Positive	Negative
5	Negative	Negative	Negative
6	Negative	Negative	Negative
7	Negative	Negative	Positive
8	Negative	Negative	Negative
9	Negative	Negative	Negative
10	Positive	Positive	Positive
11	Positive	Positive	Negative
12	Positive	Positive	Positive
13	Positive	Positive	Positive
14	Positive	Negative	Negative
15	Positive	Positive	Negative

Which of these two models is the best? The choice of an error or quality measure is highly **application-dependent**.

## Binary classification (i)

Results can be summarized in a **contingency table**, also called confusion matrix.

Actual class	Predicted class		Total
	Positive	Negative	
Positive	<b>T</b> True <b>P</b> Positive	<b>F</b> alse <b>N</b> egative	P
Negative	<b>F</b> alse <b>P</b> ositive	<b>T</b> True <b>N</b> egative	N

$$\rightarrow \text{Error rate} = \frac{FP+FN}{P+N}$$

$$\rightarrow \text{Accuracy} = \frac{TP+TN}{P+N} = 1 - \text{Error rate}$$

## Binary classification (ii)

The simplest criterion to compare model performances is the error rate or the accuracy:

Example:

Actual class	Predicted class		Total
	Positive	Negative	
Positive	5	1	6
Negative	3	6	9

Model 1

$$\rightarrow \text{Error rate} = \frac{4}{15} = 27\%$$

$$\rightarrow \text{Accuracy} = \frac{11}{15} = 73\%$$

Actual class	Predicted class		Total
	Positive	Negative	
Positive	3	3	6
Negative	2	7	9

Model 2

$$\rightarrow \text{Error rate} = \frac{5}{15} = 33\%$$

$$\rightarrow \text{Accuracy} = \frac{10}{15} = 66\%$$

# Limitations of the error rate

Actual class	Predicted class		Total
	Positive	Negative	
Positive	0	10	P
Negative	0	90	N

Model 1

Error rate = 10%

Actual class	Predicted class		Total
	Positive	Negative	
Positive	10	0	P
Negative	10	80	N

Model 2

Error rate = 10%

Actual class	Predicted class		Total
	Positive	Negative	
Positive	0	50	P
Negative	0	50	N

Model 3

Error rate = 50%

This criterion does not convey any information about the error distribution across classes: the first two models have the same error rates but different error distributions. It is also sensitive to changes in the class distribution in the test sample.



## Sensitivity and specificity (i)

For medical diagnosis, more appropriate measures are:

- Sensitivity (or recall) =  $\frac{TP}{P}$
- Specificity =  $\frac{TN}{TN+FP} = 1 - \frac{FP}{N}$

## Sensitivity and specificity (ii)

Actual class	Predicted class		Total
	Positive	Negative	
Positive	0	10	10
Negative	0	90	90

Model 1

- Error rate = 10%
- Sensitivity =  $\frac{0}{10} = 0\%$
- Specificity =  $\frac{90}{90} = 100\%$

Actual class	Predicted class		Total
	Positive	Negative	
Positive	10	0	10
Negative	10	80	90

Model 2

- Error rate = 10%
- Sensitivity =  $\frac{10}{10} = 100\%$
- Specificity =  $\frac{80}{90} = 89\%$

Actual class	Predicted class		Total
	Positive	Negative	
Positive	0	50	P
Negative	0	50	N

Model 3

- Error rate = 50%
- Sensitivity =  $\frac{0}{50} = 0\%$
- Specificity =  $\frac{50}{50} = 100\%$

## Sensitivity and specificity (iii)

	True class	Model 1	Model 2
1	Negative	Positive	Negative
2	Negative	Negative	Negative
3	Negative	Positive	Positive
4	Negative	Positive	Negative
5	Negative	Negative	Negative
6	Negative	Negative	Negative
7	Negative	Negative	Positive
8	Negative	Negative	Negative
9	Negative	Negative	Negative
10	Positive	Positive	Positive
11	Positive	Positive	Negative
12	Positive	Positive	Positive
13	Positive	Positive	Positive
14	Positive	Negative	Negative
15	Positive	Positive	Negative

Actual class	Predicted class		Total
	Positive	Negative	
Positive	5	1	6
Negative	3	6	9

Model 1

$$\rightarrow \text{Sensitivity} = \frac{5}{6} = 83\%$$

$$\rightarrow \text{Specificity} = \frac{6}{9} = 66\%$$

Actual class	Predicted class		Total
	Positive	Negative	
Positive	3	3	6
Negative	2	7	9

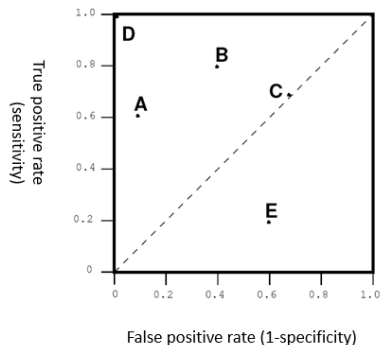
Model 2

$$\rightarrow \text{Sensitivity} = \frac{3}{6} = 50\%$$

$$\rightarrow \text{Specificity} = \frac{7}{9} = 78\%$$

Determining which model is the best one depends on the application.

# Receiver Operating Characteristic (ROC) Curve (i)



Where are:

- ▶ The best classifier?
- ▶ A classifier that always says positive?
- ▶ A classifier that always says negative?
- ▶ A classifier that randomly guesses the class?

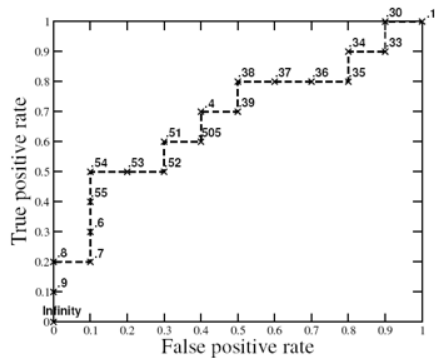
## ROC Curve (ii)

Often, the output of a classification algorithm is a number, *e.g.* a class probability. In this case, a threshold may be chosen in order to balance sensitivity and specificity.

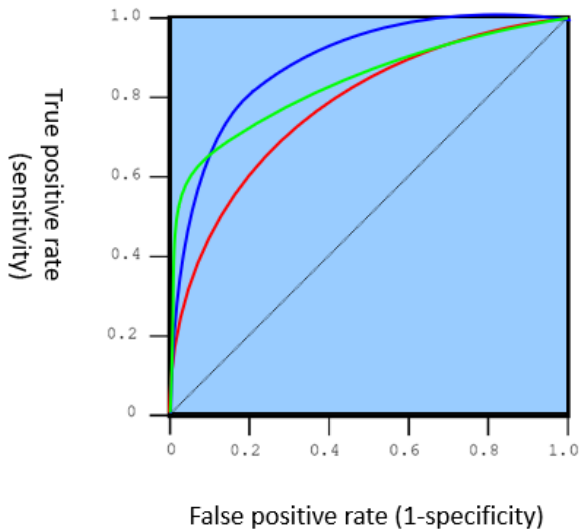
A ROC curve plots sensitivity versus  $1 - \text{specificity}$  values for different thresholds.

# ROC Curve (iii)

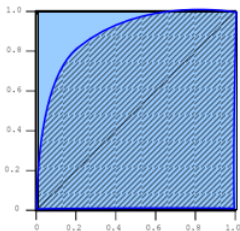
Inst#	Class	Score	Inst#	Class	Score
1	p	.9	11	p	.4
2	p	.8	12	n	.39
3	n	.7	13	p	.38
4	p	.6	14	n	.37
5	p	.55	15	n	.36
6	p	.54	16	n	.35
7	n	.53	17	p	.34
8	n	.52	18	n	.33
9	p	.51	19	p	.30
10	n	.505	20	n	.1



## ROC Curve (iv)



## Area under the ROC curve



The area under the ROC curve summarizes a ROC curve by a single number. It can be interpreted as the **probability** that two objects randomly drawn from the sample are well ordered by the model, *i.e.* the positive has a higher score than the negative.

**However**, it does not tell the whole story.



## Precision and recall (i)

Other frequently used measures are:

- ▶ Precision =  $\frac{TP}{TP+FP}$  = proportion of good predictions among all the positive predictions
- ▶ Recall =  $\frac{TP}{TP+FN}$  = proportion of positives that are detected
- ▶ F-measure =  $2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$

## Precision and recall (ii)

Actual class	Predicted class		Total
	Positive	Negative	
Positive	10	0	10
Negative	50	950	1000

Model 1

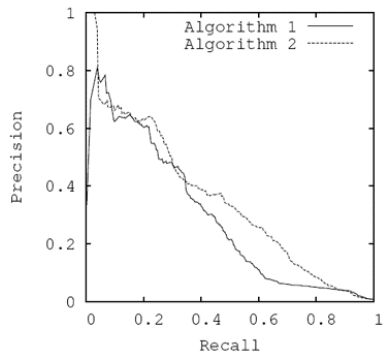
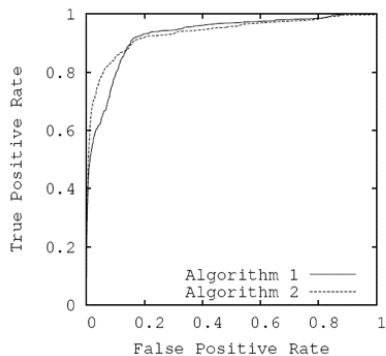
- Sensitivity =  $\frac{10}{10} = 100\%$
- Specificity =  $\frac{950}{1000} = 95\%$
- Precision =  $\frac{10}{60} = 17\%$
- Recall =  $\frac{10}{10} = 100\%$
- F-measure = 29%

Actual class	Predicted class		Total
	Positive	Negative	
Positive	10	0	10
Negative	10	990	1000

Model 2

- Sensitivity =  $\frac{10}{10} = 100\%$
- Specificity =  $\frac{990}{1000} = 99\%$
- Precision =  $\frac{10}{20} = 50\%$
- Recall =  $\frac{10}{10} = 100\%$
- F-measure = 66%

# Precision/recall vs ROC curve



## Regression performance (i)

	True Y	Model 1	Model 2
1	59.43	63.08	78.12
2	33.15	36.66	37.28
3	11.8	13.28	19.62
4	77.11	78.38	90.01
5	40.6	42.92	60.19
6	81.25	85	86.13
7	83.01	86.37	102.32
8	0.55	1.67	16.6
9	76.72	80.92	94.74
10	29.18	33.28	52.54
11	20.37	22.43	36.94
12	95.13	98.86	104.24
13	11.66	15.97	28.16
14	42.95	46.12	65.82
15	94.55	94.75	104.02

### ► Mean squared error

$$\frac{1}{N} \sum_{i=1}^N (y_i - \hat{y}_i)^2$$

$$\rightarrow MSE_1 = 53.38$$

$$\rightarrow MSE_2 = 249.6$$

### ► Mean absolute error

$$\frac{1}{N} \sum_{i=1}^N |y_i - \hat{y}_i|$$

$$\rightarrow MAE_1 = 4.3$$

$$\rightarrow MAE_2 = 14.6$$

## Regression performance (ii)

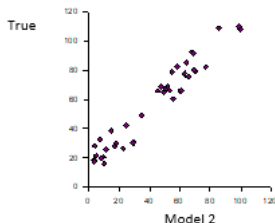
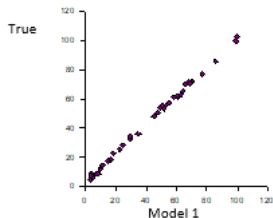
- ▶ Pearson correlation

$$\frac{\sum_i \left( y_i - \frac{1}{N} \sum_j y_j \right) \left( \hat{y}_i - \frac{1}{N} \sum_j \hat{y}_j \right)}{(N-1)s_y s_{\hat{y}}}$$

- ▶ Spearman rank correlation

$$1 - \frac{6 \sum_i d_i^2}{N(N^2 - 1)}$$

with  $d_i$  the difference of rank of  $y_i$   
and  $\hat{y}_i$



# Performance measures for training

Performance measures for training can be different from performance measures for testing. There are several reasons for that:

▶ Algorithmic:

- A differentiable measure is amenable to gradient optimization.
- A decomposable measure is amenable to online training.

**Examples:** the error rate and MAE are not derivable, the AUC is not decomposable.

▶ Overfitting:

- For training, the loss function often incorporates a penalty term for model complexity, which is irrelevant at test time.
- Some measures are less prone to overfitting (e.g. margin).



# Outline

- 1 Bias/variance trade-off
- 2 Performance evaluation
- 3 Performance measures
- 4 Further reading**

- ▶ Hastie et al., chapter 2 & 7, [1]:
  - Bias/variance trade-off (2.5, 2.9, 7.2, 7.3)
  - Model assessment and selection (7.1, 7.2, 7.3, 7.10, 7.11)



# References I

-  Trevor Hastie, Robert Tibshirani, and Jerome Friedman.  
*The elements of statistical learning: data mining, inference, and prediction.*  
Springer Science & Business Media, 2009.
-  Cross-validation for detecting and preventing overfitting.  
<https://www.autonlab.org/resources/tutorials>.  
Accessed: 2020-10-21.