

Resampling techniques for statistical modeling

Gianluca Bontempi

Département d'Informatique
Boulevard de Triomphe - CP 212
<http://www.ulb.ac.be/di>

Prediction error

- So far we have focused on measures of statistical accuracy for parameters of a model: bias, variance, confidence interval and so on.
- *Prediction error* is a different quantity that measures how well a model predicts the response value of a future observations.
- This quantity is used in **model selection**, that is the procedure that aims to choose a model that has the lowest prediction error among a set of candidates.
- Before studying prediction error, we introduce the notion of model in the problems of regression and classification.

Input/output problems

Consider this set of prediction problems

- Predict whether a patient, hospitalized due to a heart attack, will have a second heart attack, on the basis of demographic, diet and clinical measurements.
- Predict the price of a stock in 6 months from now, on the basis of company performance measures and economic data.
- Identify the risk factors for breast cancer, based on clinical and demographic variables.
- Classify the category of a text email (spam or not) on the basis of its text content.
- Characterize the mechanical property of a steel plate on the basis of its physical and chemical composition.

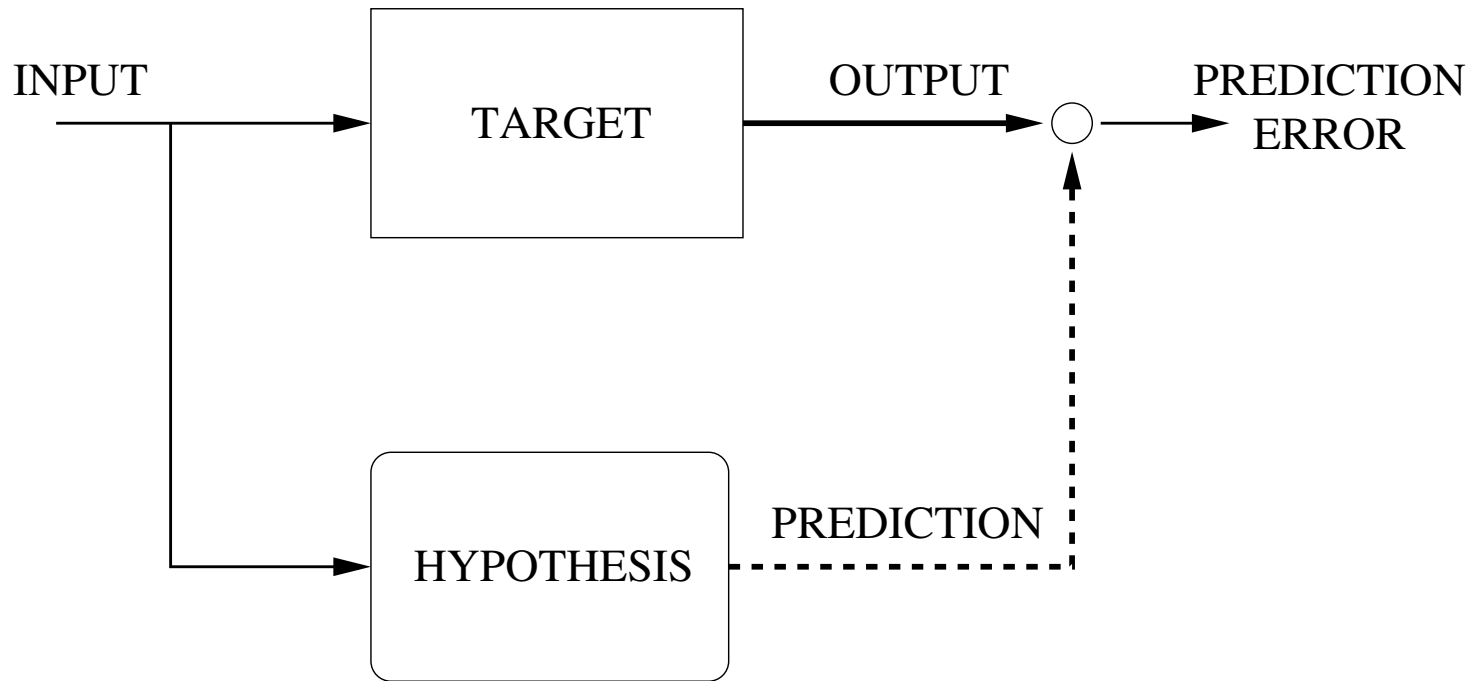
Input/output problems

All the previous examples are characterized by

1. An outcome measurement, also called **output**, usually quantitative (like a stock price) or categorical (like heart attack/no heart attack).
2. a set of **features** or **inputs**, also quantitative or categorical, that we wish to use to predict the output.

If we suppose that we have available a set of input/output data, also called **training set**, we could use statistical methods to build a **prediction model** or **learner** which could enable us to predict the outcome for **new unseen objects**.

Supervised learning



The prediction problem is also known as the **supervised learning** problem, because of the presence of the outcome variable which guides the learning process.

Collecting a set of training data is analogous to the situation where a teacher suggests the correct answer for each input configuration.

Regression and classification

According to the type of output we can distinguish between two types of prediction tasks:

Regression when we predict quantitative outputs, e.g. real or integer numbers

Classification (or pattern recognition) where we predict qualitative or categorical outputs which assume values in a finite set of classes (e.g. black, white and red) where there is no explicit ordering. Qualitative variables are also referred to as **factors**.

The regression statistical setting

- Let $\mathbf{x} \in \mathbb{R}^n$ denote a real valued random input vector and y a real valued random output variable.
- Let $p_{\mathbf{x},y}(x, y)$ be their joint distribution function.
- We seek a function $h(\cdot)$ for predicting y given values of the input x .
- We define a *loss function* $C(y, h(x))$ for penalizing error in prediction.
- We seek for the function $h(\cdot)$ that minimizes

$$J(h) = E_{\mathbf{x},y}[C(y, h(\mathbf{x}))] = \int C(y, h(x))p(x, y)dx dy$$

The regression statistical setting (II)

- The function that minimizes the above cost is the one that minimizes the pointwise version

$$J(x, h) = E_{\mathbf{y}}[C(\mathbf{y}, h(x))|x]$$

- It can be shown that if $C(y, h) = (y - h)^2$ the optimal solution is the so-called *regression function*

$$E[\mathbf{y}|x]$$

- Unfortunately the quantity $p_{\mathbf{x},\mathbf{y}}(x, y)$ is unknown as the regression function $E[\mathbf{y}|x]$.
- The only available information is a set of N samples i.i.d. distributed according to $p_{\mathbf{x},\mathbf{y}}(x, y)$. Regression techniques aim at estimating $E[\mathbf{y}|x]$ on the basis of the observed data.

The classification statistical setting

- Let $\mathbf{x} \in \mathbb{R}^n$ denote a real valued random input vector and \mathbf{y} random output variable that takes values in the set $\{c_1, \dots, c_K\}$.
- Let $\text{Prob}\{\mathbf{y} = c_k | \mathbf{x}\}$ the probability that the output belongs to the k th class given the set of measurements \mathbf{x} . It follows

$$\sum_{k=1}^K \text{Prob}\{\mathbf{y} = c_k | \mathbf{x}\} = 1$$

- An estimate $c(\mathbf{x})$ of the class takes values in $\{c_1, \dots, c_K\}$. We define a *loss matrix* $L_{(k \times k)}$, being null on the diagonal and non negative elsewhere.
- The element $L_{(jk)} = L_{(c_j, c_k)}$ denotes the cost of the misclassification when the predicted class is c_j and the correct class is c_k .

- The goal of the classification procedure is to find the predictor c that minimizes for all x

$$\sum_{k=1}^K L(c(x), c_k) \text{Prob} \{y = c_k | x\}$$

- It can be shown that the optimal classifier (also known as the *Bayes classifier*) is the one that returns for all x

$$\arg \min_j \sum_{k=1}^K L(j, k) \text{Prob} \{y = c_k | x\}$$

- In the case of a 0-1 loss function the optimal classifier returns

$$\arg \max_j \text{Prob} \{y = c_j | x\}$$

- Unfortunately, we can derive the Bayes classifier only if the quantities $\text{Prob} \{y = c_j | x\}$ are known.

Classification strategies

Optimal classification is possible if the quantities $\text{Prob}\{y = c_j|x\}$ are known. Two strategies are generally used.

Density estimation via the Bayes theorem. Since

$$\text{Prob}\{y = c_j|x\} = \frac{\text{Prob}\{x|y = c_j\} \text{Prob}\{y = c_j\}}{\text{Prob}\{x\}}$$

an estimation of $\text{Prob}\{x|y = c_j\}$ allows an estimation of $\text{Prob}\{y = c_j|x\}$.

Direct estimation via regression techniques. If the classification problem has $K = 2$ classes and if we denote them by $y = 0$ and $y = 1$

$$E[y|x] = \text{Prob}\{y = 1|x\} + 0\text{Prob}\{y = 0|x\} = \text{Prob}\{y = 1|x\}$$

Then the classification problem can be put in the form of a regression problem where the output takes value in $\{0, 1\}$.

Linear relationship

Linear regression is a very old technique in statistics and traces back to the work of Gauss.

Consider a linear relation between an independent variable $x \in \mathcal{X} \subset \mathbb{R}^n$ and a dependent random variable $y \in \mathcal{Y} \subset \mathbb{R}$

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n + \mathbf{w}$$

where \mathbf{w} represents a random variable with mean zero and constant variance σ_w^2 . In matrix notation the equation can be written as:

$$y = x^T \beta + \mathbf{w}$$

where x stands for the $[p \times 1]$ vector $x = [1, x_1, x_2, \dots, x_n]^T$ and where $p = n + 1$ is the total number of model parameters.

The multiple linear regression model

Consider N observations $\{\langle x_i, y_i \rangle : i = 1, \dots, N\}$, where $x_i = (x_{i1}, \dots, x_{in})$, generated according to the previous model. The multiple linear regression model can be written as

$$Y = X\beta + w$$

where Y the $[N \times 1]$ response vector, X is the $[N \times p]$ *data matrix*, whose j^{th} column of X contains readings on the j^{th} regressor, β is the $[p \times 1]$ vector of parameters

$$Y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix} \quad X = \begin{bmatrix} 1 & x_{11} & x_{12} & \dots & x_{1n} \\ 1 & x_{21} & x_{22} & \dots & x_{2n} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_{N1} & x_{N2} & \dots & x_{Nn} \end{bmatrix} \quad \beta = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_n \end{bmatrix} \quad W = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_N \end{bmatrix}$$

where w_i are assumed uncorrelated, with mean zero and constant variance σ_w^2 (homogeneous variance). Then $\text{Var}[\mathbf{w}_1, \dots, \mathbf{w}_N] = \sigma_w^2 I_N$.

The least-squares solution

We seek the the least-squares estimator $\hat{\beta}$ such that

$$\hat{\beta} = \arg \min_b \sum_{i=1}^N (y_i - x_i^T b)^2 = \arg \min_b ((Y - Xb)^T (Y - Xb))$$

Given $\hat{\beta}$ we obtain

$$\widehat{\text{SSE}}_{\text{emp}} = \left((Y - X\hat{\beta})^T (Y - X\hat{\beta}) \right) = e^T e$$

the **residual sum of squares** where e is the $[N \times 1]$ vector of residuals.

The vector $\hat{\beta}$ must satisfy

$$\frac{\partial}{\partial \hat{\beta}} [(Y - X\hat{\beta})^T (Y - X\hat{\beta})] = 0$$

Normal equations

- Differentiating the residual sum of squares we obtain the *least-squares normal equations*

$$(X^T X)\hat{\beta} = X^T Y$$

- As a result, assuming X is of full column rank

$$\hat{\beta} = (X^T X)^{-1} X^T Y$$

where the $X^T X$ matrix is a symmetric $[p \times p]$ matrix which plays an important role in multiple linear regression.

- Note that the computation of $\hat{\beta}$ represents the **parametric identification step** when the class of hypothesis is linear.

Analysis of the LS estimate

This section reviews some results of linear regression analysis.

- The *residual mean square* estimator

$$\hat{\sigma}^2 = \frac{(Y - X\hat{\beta})^T (Y - X\hat{\beta})}{N - p}$$

is an unbiased estimator of the error variance $\sigma_{\mathbf{w}}^2$, assuming the underlying model is linear.

- Under the condition that $E[\mathbf{w}] = 0$, $\hat{\beta}$ is an unbiased estimator of β .
- Under the assumption that the w_i are uncorrelated and have common variance, the variance-covariance matrix of $\hat{\beta}$ is given by

$$\text{Var}[\hat{\beta}] = \sigma_{\mathbf{w}}^2 (X^T X)^{-1}$$

Variance of the prediction

- The variance of the prediction \hat{y} for a generic input value $x = x_0$ is given by

$$\text{Var}[\hat{y}|x_0] = \sigma_{\mathbf{w}}^2 x_0^T (X^T X)^{-1} x_0$$

- Assuming normal errors, the $100(1 - \alpha)\%$ confidence bound for the regression value $E[\mathbf{y}|x = x_0]$ is given by

$$\hat{y}(x_0) \pm t_{\alpha/2, n-p} \hat{\sigma} \sqrt{x_0^T (X^T X)^{-1} x_0}$$

where $t_{\alpha/2, N-p}$ is the upper $\alpha/2$ percent point of the \mathcal{T} -distribution with $N - p$ degrees of freedom and the quantity

$$\hat{\sigma} \sqrt{x_0^T (X^T X)^{-1} x_0}$$

is the *standard error of prediction* for multiple regression.

The HAT matrix

The Hat matrix is defined as

$$H = X(X^T X)^{-1} X^T$$

It is a symmetric, idempotent $[N \times N]$ matrix that transforms the output values Y of the training set in the regression predictions \hat{Y} :

$$\hat{Y} = X\hat{\beta} = X(X^T X)^{-1} X^T Y = HY$$

Using the above relation, the vector of residuals can be written as:

$$e = Y - X\hat{\beta} = Y - X(X^T X)^{-1} X^T Y = [I - H]Y$$

and the residual sum of squares as

$$e^T e = Y^T [I - H]^2 Y = Y^T [I - H] Y = Y^T P Y$$

where P is a $[N \times N]$ matrix, called the *projection* matrix.

The expected empirical error

The expectation of the residual sum of squares can be written as

$$E_{\mathbf{D}_N}[\mathbf{e}^T \mathbf{e}] = E_{\mathbf{D}_N}[\mathbf{Y}^T P \mathbf{Y}] = \sigma_{\mathbf{w}}^2 \text{tr}(P) + E[\mathbf{Y}^T] P E[\mathbf{Y}]$$

Since

$$\begin{aligned} \text{tr}(P) &= \text{tr}(I - H) = N - \text{tr}(X(X^T X)^{-1} X) = \\ &= N - \text{tr}(X^T X (X^T X)^{-1}) = N - p \end{aligned}$$

we have

$$\begin{aligned} E[\mathbf{e}^T \mathbf{e}] &= (N - p)\sigma_{\mathbf{w}}^2 + (X\beta)^T P(X\beta) = \\ &= (N - p)\sigma_{\mathbf{w}}^2 + \beta^T X^T (I - X(X^T X)^{-1} X^T) X \beta = (N - p)\sigma_{\mathbf{w}}^2 \end{aligned}$$

This is the expectation of the error made by a linear model trained on D_N to predict the value of the output in D_N .

MISE error

Let us compute now the expected prediction error of a linear model trained on D_N when this is used to predict for the same training inputs X a set of outputs Y_{ts} distributed according to the same linear law but independent of the training output Y . We call this quantity **Mean Integrated Squared Error**

$$\begin{aligned} \text{MISE} &= E_{\mathbf{D}_{N,\mathbf{y}}}[(\mathbf{Y}_{ts} - X\hat{\beta})^T (\mathbf{Y}_{ts} - X\hat{\beta})] = \\ &= E_{\mathbf{D}_{N,\mathbf{y}}}[(\mathbf{Y}_{ts} - X\beta + X\beta - X\hat{\beta})^T (\mathbf{Y}_{ts} - X\beta + X\beta - X\hat{\beta})] = \\ &= N\sigma_w^2 + E_{\mathbf{D}_N}[(X\beta - X\hat{\beta})^T (X\beta - X\hat{\beta})] \end{aligned}$$

Since

$$\begin{aligned} X\beta - X\hat{\beta} &= X\beta - X(X^T X)^{-1} X^T Y = \\ &= X\beta - X(X^T X)^{-1} X^T (X\beta + w) = -X(X^T X)^{-1} X^T w \end{aligned}$$

we have

$$\begin{aligned} N\sigma_{\mathbf{w}}^2 + E_{\mathbf{D}_N}[(X\beta - X\hat{\beta})^2] &= \\ &= N\sigma_{\mathbf{w}}^2 + E_{\mathbf{D}_N}[\mathbf{w}^T X(X^T X)^{-1} X^T X(X^T X)^{-1} X^T \mathbf{w}] = \\ &= N\sigma_{\mathbf{w}}^2 + E_{\mathbf{D}_N}[\text{tr}(\mathbf{w}^T \mathbf{w})] = \sigma_{\mathbf{w}}^2(N + p) \end{aligned}$$

Then, we obtain that the empirical error returns a biased estimate of MISE, that is

$$E_{\mathbf{D}_N}[\widehat{\text{SSE}}_{\text{emp}}] = E_{\mathbf{D}_N}[\mathbf{e}^T \mathbf{e}] \neq \text{MISE}$$

As a consequence, if we replace $\widehat{\text{SSE}}_{\text{emp}}$ with

$$\mathbf{e}^T \mathbf{e} + 2\sigma_{\mathbf{w}}^2 p$$

we obtain an unbiased estimator.

Nevertheless, this estimator requires an estimate of the noise variance.

The PSE and the FPE

Given an a priori estimate $\hat{\sigma}_{\mathbf{w}}^2$ we have the **Predicted Square Error (PSE)** criterion

$$PSE = \widehat{SSE}_{\text{emp}} + 2\hat{\sigma}_{\mathbf{w}}^2 p$$

Taking as estimate of $\sigma_{\mathbf{w}}^2$

$$\hat{\sigma}_{\mathbf{w}}^2 = \frac{1}{N - p} \widehat{SSE}_{\text{emp}}$$

we have the **Final Prediction Error (FPE)**

$$FPE = \frac{1 + p/N}{1 - p/N} \widehat{SSE}_{\text{emp}}$$

Nonlinear input/output problems

- so far we have considered input/output problems where the relation between input and output is linear

$$\mathbf{y} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n + \mathbf{w}$$

- The advantage of linear models are numerous:
 - the least-squares $\hat{\beta}$ estimate can be expressed in an analytical form
 - the least-squares $\hat{\beta}$ estimate can be easily calculated through matrix computation.
 - statistical properties of the estimator can be easily defined.
 - recursive formulation for sequential updating are available.
- Unfortunately in real problem it is extremely unlikely that the input and output variables are linked by a linear relation.
- Moreover, the form of the relation is often unknown and only a limited amount of samples is available.

Some examples of hypothesis functions

Nearest-neighbor regression

Radial basis function

Feedforward neural network

Regression tree

Support vector machine

Hierarchical mixtures of experts

The regression plus noise form

- A typical way of representing the unknown input/output relation is the **regression plus noise form**

$$\mathbf{y} = f(x) + \mathbf{w}$$

where f is a deterministic function and the term \mathbf{w} represents the noise or random error. It is typically assumed that \mathbf{w} is independent of \mathbf{x} and $E[\mathbf{w}] = 0$.

- Suppose that we have available a **training set** $\{\langle x_i, y_i \rangle : i = 1, \dots, N\}$, where $x_i = (x_{i1}, \dots, x_{in})$, generated according to the previous model.
- The goal of a learning procedure is to find a model $h(x)$ which is able to give a good approximation of the unknown function $f(x)$.
- But how to choose h , if we do not know the probability distribution underlying the data and we have only a limited training set?

What is a good model?

- In order to assess a model $h(x)$ we need to define a **cost (or loss) function** C associated with $f(x)$ and $h(x)$.
- The cost function measures the discrepancy between the output of the function $f(\cdot)$ and the output of the function $h(\cdot)$, given an input x .
- The cost function can have different forms. In the regression problem the cost function is typically supposed to be quadratic

$$C(y(x), h(x, \alpha)) = (y(x) - h(x, \alpha))^2$$

- In the classification case

$$C(y(x), h(x, \alpha)) = I(y(x) \neq h(x, \alpha))$$

What is a good model? (II)

- If the input/output phenomenon were deterministic it could be easy to define the error of a model $h(x, \alpha)$ by

$$\int_{\mathcal{X}} C(f(x), h(x, \alpha)) dx$$

- Unfortunately, the stochastic case is more complex.

The regression error: functional risk and MSE

In the regression problem, two statistical measure of discrepancy can be defined for a given x

Functional risk: it averages over the \mathcal{Y} -domain the cost C for a given hypothesis $h(\cdot, \alpha_N)$:

$$R(\alpha_N)(x) = E_{\mathbf{y}}[\mathbf{C}(x)] = \int_{\mathcal{Y}} (y - h(x, \alpha_N))^2 p_{\mathbf{y}}(y|x) dy$$

In this expression, \mathbf{C} is intended as a function of the random variable \mathbf{y} , while the hypothesis α_N is kept fixed.

Mean-squared error: it averages a quadratic cost C for a given input x over the ensemble of training sets with N samples:

$$\text{MSE}(x) = E_{\mathbf{D}_N, \mathbf{y}}[\mathbf{C}|x] = \int_{\mathcal{Z}^N, \mathcal{Y}} (y - h(x, \alpha_N(\mathbf{D}_N)))^2 dP_f(y|x) dP^N(\mathbf{D}_N)$$

In this expression α_N is a function of the random variables \mathbf{D}_N .

The classification error

In classification the output variable is a class label. We can define

Misclassification error: it averages over the \mathcal{Y} -domain the cost C for a given hypothesis $h(\cdot, \alpha_N)$:

$$E_{\mathbf{y}}[\mathbf{C}(x)] = \int_{\mathcal{Y}} I(y \neq h(x, \alpha_N)) p_{\mathbf{y}}(y|x) dy$$

In this expression, \mathbf{C} is intended as a function of the random variable \mathbf{y} , while the hypothesis α_N is kept fixed.

Mean misclassification error: it averages the cost C for a given input x over the ensemble of training sets with N samples:

$$\text{MME}(x) = E_{\mathbf{D}_N, \mathbf{y}}[\mathbf{C}|x] = \int_{\mathcal{Z}^N, \mathcal{Y}} I(y \neq h(x, \alpha_N(D_N))) dP_f(y|x) dP^N(D_N)$$

In this expression α_N is a function of the random variables \mathbf{D}_N .

The learning procedure

A learning procedure aims at two main goals:

1. to choose a parametric family of hypothesis $h(x, \alpha)$ which contains or gives good approximation of the unknown function f (**structural identification**).
2. within the family $h(x, \alpha)$, to estimate on the basis of the training set D_N the parameter α_N which best approximates f (**parametric identification**).

In order to accomplish that, a learning procedure is made of two *nested loops*:

1. an external structural identification loop which goes through different model structures
2. an inner parametric identification loop which searches for the best parameter vector within the family structure.

Parametric identification

The parametric identification of the hypothesis is done according to ERM (Empirical Risk Minimization) principle where

$$\alpha_N = \alpha(D_N) = \arg \min_{\alpha \in \Lambda} \widehat{\text{SSE}}_{\text{emp}}(\alpha)$$

minimizes the **empirical risk or training error**

$$\widehat{\text{SSE}}_{\text{emp}}(\alpha) = \sum_{i=1}^N (y_i - h(x_i, \alpha))^2$$

constructed on the basis of the data set D_N .

Parametric identification and optimization

- The computation of α_N is formulated as an optimization problem.
- The computation of α_N requires a procedure of multivariate optimization in the space of parameters.
- The complexity of the optimization depends on the form of the model h .
- Examples of parametric identification procedure are linear least-squares for linear models and backpropagated gradient-descent for feedforward neural networks.

Generalization and overfitting

- How to estimate the quality of a model? Is the empirical risk a good measure of MSE?
- The goal of learning is to find a model which is able to **generalize**, i.e. able to return good predictions for input sets independent of the training set.
- In a nonlinear setting, it is possible to find models with such a complicate structure that they have a null empirical risk. Are these models good?
- Typically NOT. Since doing very well on the training set could mean doing badly on new data.
- This is the phenomenon of **overfitting**.
- Using the same data for training a model and assessing it is typically a wrong procedure, since this returns an over optimistic assessment of the model generalization capability.

Bias and variance of a model

The output of a parametric identification algorithm is an estimator $h(x, \alpha_N)$ of the unknown quantity $f(x)$.

Like for all estimators we can define its bias

$$\text{Bias}[h(x, \alpha_N)] = E[h(x, \alpha_N)] - f(x)$$

and its variance $\text{Var}[h(x, \alpha_N)]$ and it can be shown that

$$\begin{aligned} \text{MSE}(x) &= E_{\mathbf{D}_N, \mathbf{y}} [\mathbf{C}(x, y)] = E_{\mathbf{D}_N, \mathbf{y}} [(\mathbf{y} - h(x, \alpha_N))^2] = \\ &= E_{\mathbf{y}} [(\mathbf{y} - E_{\mathbf{y}}[\mathbf{y}|x])^2] + E_{\mathbf{D}_N} [(h(x, \alpha_N) - E_{\mathbf{y}}[\mathbf{y}|x])^2] = \\ &= \sigma_{\mathbf{w}}^2 + E_{\mathbf{D}_N} [(h(x, \alpha_N) - E_{\mathbf{y}}[\mathbf{y}|x])^2] = \\ &= \sigma_{\mathbf{w}}^2 + \qquad \qquad \qquad \text{“noise”} \\ &+ (E_{\mathbf{D}_N} [h(x, \alpha_N)] - E_{\mathbf{y}}[\mathbf{y}|x])^2 + \qquad \qquad \text{“squared bias”} \\ &+ E_{\mathbf{D}_N} [(h(x, \alpha_N) - E_{\mathbf{D}_N} [h(x, \alpha_N)])^2] \qquad \text{“variance”} \end{aligned}$$

The bias/variance trade-off

- The first term is the variance of y around its true mean $f(x)$ and cannot be avoided no matter how well we estimate $f(x)$, unless $\sigma_{\mathbf{w}}^2 = 0$.
- The bias measures the difference in x between the average of the outputs of the hypothesis functions over the set of possible D_N and the regression function value $f(x)$
- The variance reflects the variability of the guessed $h(x, \alpha_N)$ as one varies over training sets of fixed dimension N . This quantity measures how sensitive the algorithm is to changes in the data set, regardless to the target.

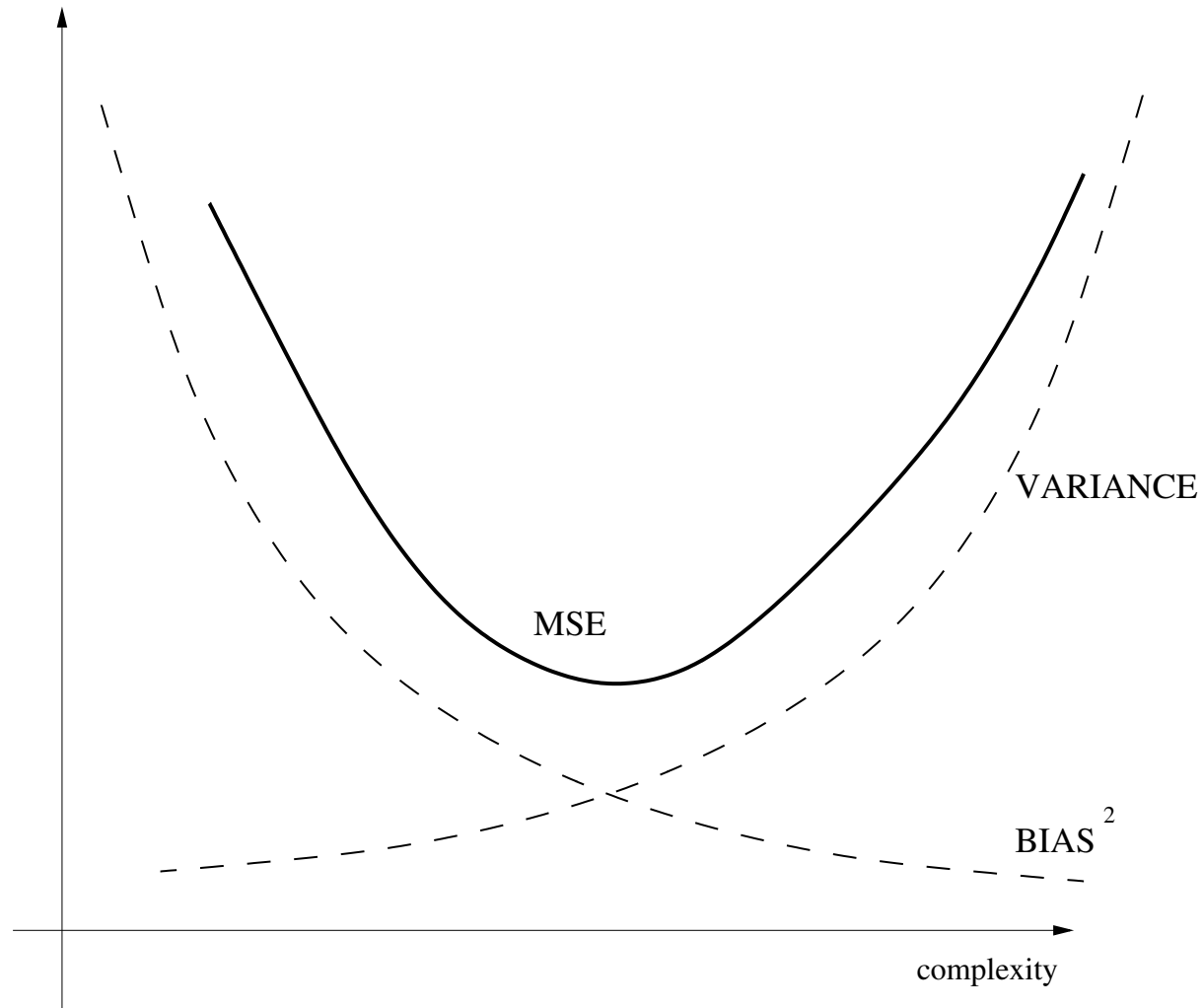
The bias/variance trade-off (II)

- In qualitative terms

$$\text{MSE} = \sigma_{\mathbf{w}}^2 + \text{squared bias} + \text{variance}$$

where the intrinsic noise term reflects the target alone, the bias reflects the target's relation with the learning algorithm and the variance term reflects the learning algorithm alone.

The bias/variance trade-off



The bias/variance dilemma

- The designer of a learning machine has not access to the term MSE but can only estimate it on the basis of the training set. Hence, the bias/variance decomposition is relevant in practical learning since it provides a useful hint about the features to control in order to make the error MSE small.
- The bias term measures the lack of representational power of the class of hypotheses. To reduce the bias term we should consider complex hypotheses which can approximate a large number of input/output mappings.
- The variance term warns us against an excessive complexity of the approximator. This means that a class of too powerful hypotheses runs the risk of being excessively sensitive to the noise affecting the training set; therefore, our class could contain the target but it could be practically impossible to find it out on the basis of the available dataset.

- In other terms, it is commonly said that an hypothesis with large bias but low variance *underfits* the data while an hypothesis with low bias but large variance *overfits* the data.
- In both cases, the hypothesis gives a poor representation of the target and a reasonable trade-off needs to be found.
- The task of the model designer is to search for the optimal trade-off between the variance and the bias term, on the basis of the available training set.