ML02 — Risk, Model Selection and k-NN example

Romain Gautron

Big Data Platform

March 5, 2019

In that chapter

→ notions of loss, real and empirical risk

→ bias-variance trade-off

→ model tuning and model training

→ k-NN algorithm illustration

→ k-fold cross validation and nested cross validation

Notations

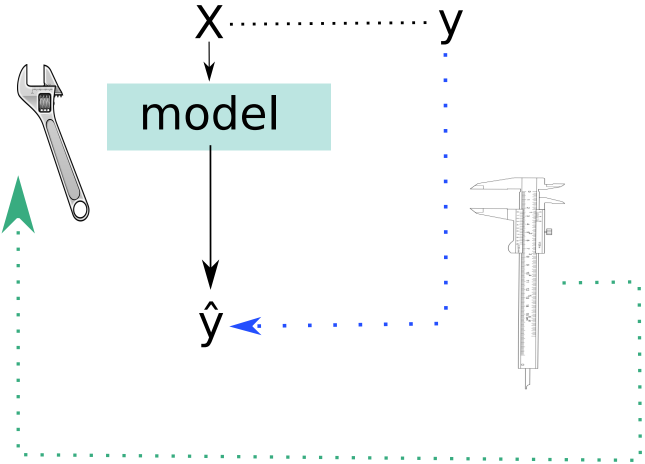
h the hypothesis i.e. the model

$$\mathbf{X} = \begin{pmatrix} 1 & x_{11} & \dots & x_{1d} \\ \vdots & \vdots & \vdots & \vdots \\ 1 & x_{n1} & \dots & x_{nd} \end{pmatrix}, \mathbf{X}_i = \begin{pmatrix} 1 \\ x_{i1} \\ \vdots \\ x_{id} \end{pmatrix}$$

$$\mathbf{Y} = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}$$

$$\Theta = \begin{pmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_n \end{pmatrix}$$

Training process



Loss function & Risk

Loss function (Cost function)

$$\hat{y} = h(X, \Theta), \mathbf{L}(y, \hat{y}) \text{ loss function}$$

Regression example

$y \in \mathbb{R}$

$L(y, \hat{y}) = (y - \hat{y})^2$

Classification example

$y \in \{-1, +1\}$

$L(y, \hat{y}) = \frac{1}{4}(y - \hat{y})^2$

"How do cost diverging from real value predictions ?"

Risk and Empirical Risk

$\mathcal{R}_{real} = \mathbb{E}(L) = \int_{X \in \mathcal{X}, y \in \mathcal{Y}} \mathbf{L}(y, h(X, \Theta)) dP_{\mathcal{X}\mathcal{Y}}, dP_{\mathcal{X}\mathcal{Y}}$ joint distribution (X, y)

We only have $(X_1, y_1), ..., (X_n, y_n)$ drawn from $P_{\mathcal{X}\mathcal{Y}}$:

$\mathcal{R}_{emp} = \frac{1}{N} \sum_{i=1}^{n} \mathbf{L}(y_i, h(X_i, \Theta))$

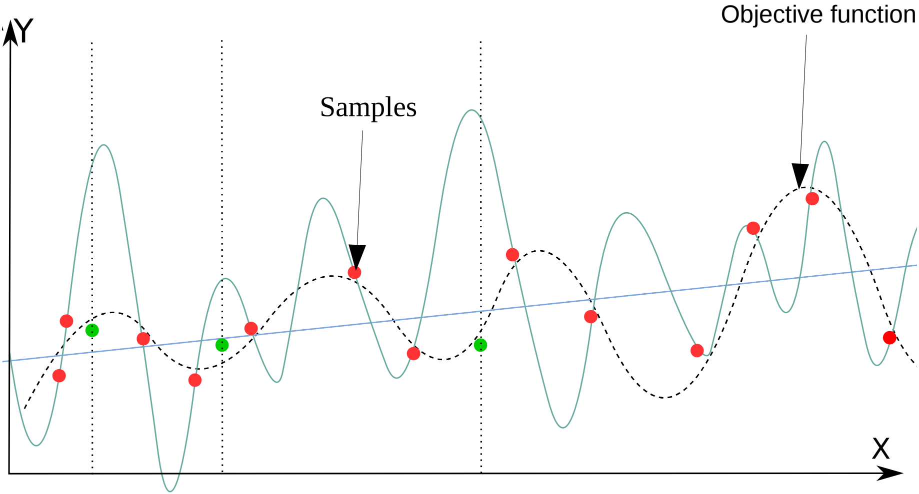
In other words, $\mathcal{R}_{emp} \equiv$ apparent error

So we want to minimize $\mathcal{R}_{emp}$? ...

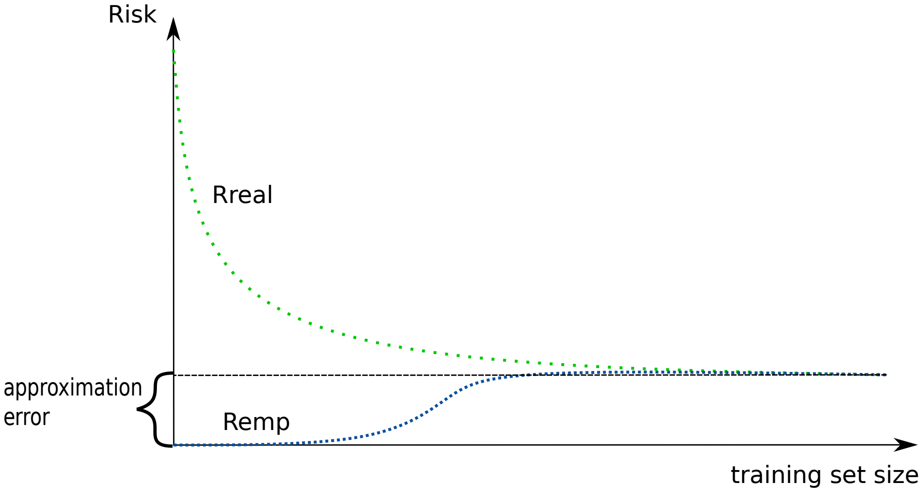
For a given X in the training set, we want to predict $\hat{y}$ as close as possible of its real value, so we want to minimize the empirical risk?

⚠ **Minimizing $\mathcal{R}_{emp}$ is not sufficient!**

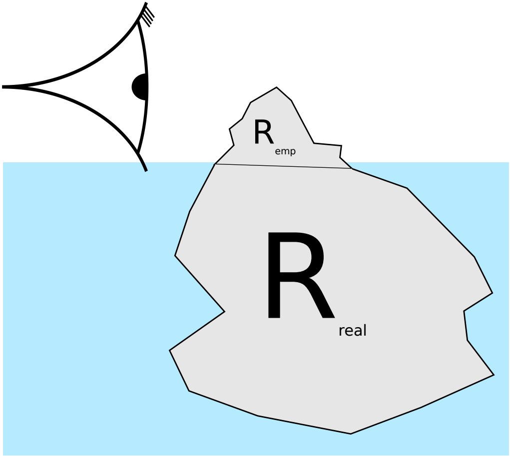
Minimizing $\mathcal{R}_{emp}$ is not sufficient!



Empirical risk, Real risk and training set size



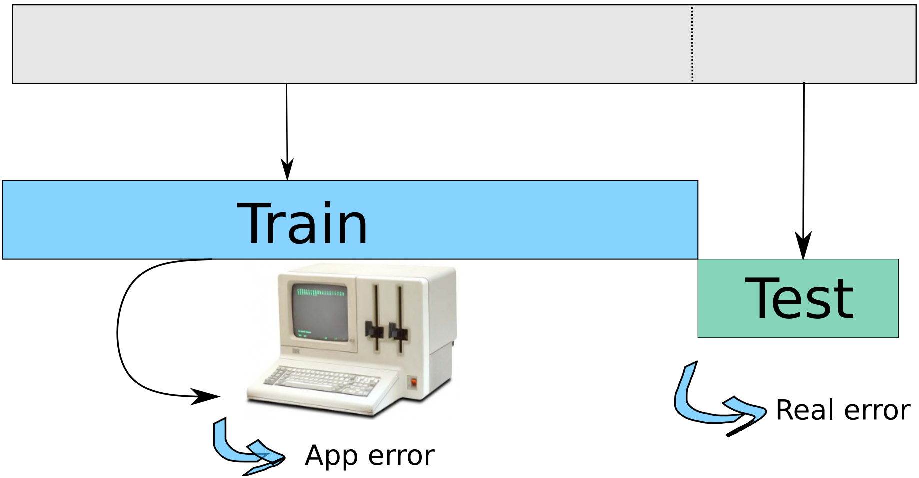
Minimizing $\mathcal{R}_{emp}$ is not sufficient!



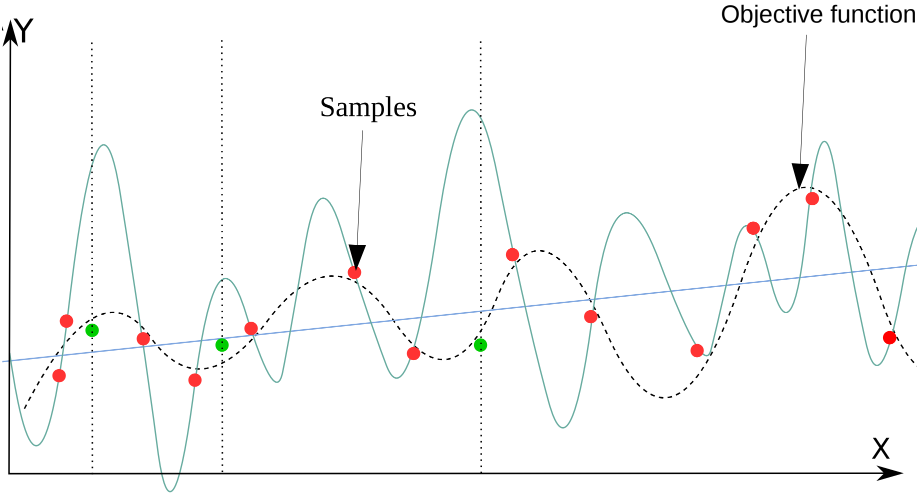
Minimizing $\mathcal{R}_{emp}$ is not sufficient!

How to measure $\mathcal{R}_{real}$???

We want an unbiased estimator of $\mathcal{R}_{real}$: basic idea



Bias-Variance trade-off: complexity and risk



Bias-Variance Decomposition (1)

Square Loss $\Rightarrow$ Empirical Risk $\equiv$ Mean Square Error

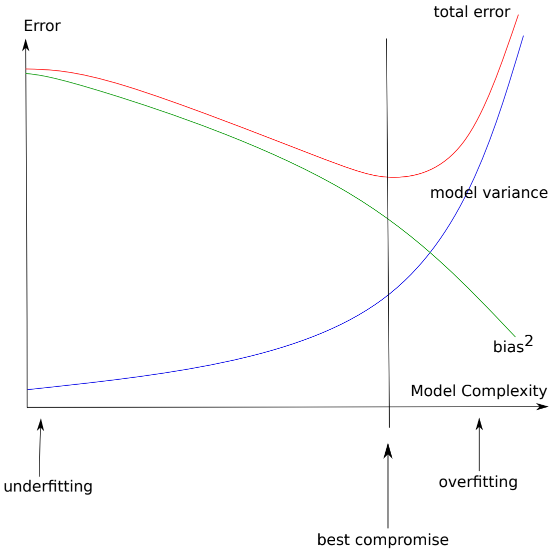
$$MSE = \frac{1}{N}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

$$\mathbb{E}((y_i - \hat{y}_i)^2) = \mathbb{V}(y_i) + \mathbb{V}(\hat{y}_i) + [\mathbb{E}(y_i) - \mathbb{E}(\hat{y}_i)]^2$$

generalization error = intrinsic error + model variance + bias^2
= intrinsic error + confidence interval + empirical error

See Vapnik–Chervonenkis theory

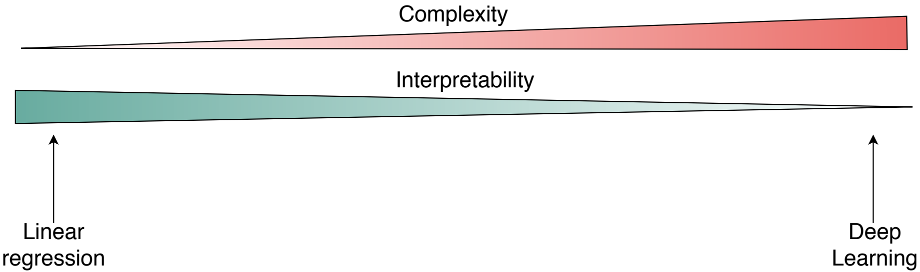
Bias-Variance Decomposition (2)



To have in mind

- $\rightarrow$ measuring an apparent error is not sufficient
- $\rightarrow$ the more complex the model the more prone to overfitting
- $\rightarrow$ an high complexity has to be compensated by a high number of learning example

Remark: complexity vs interpretability



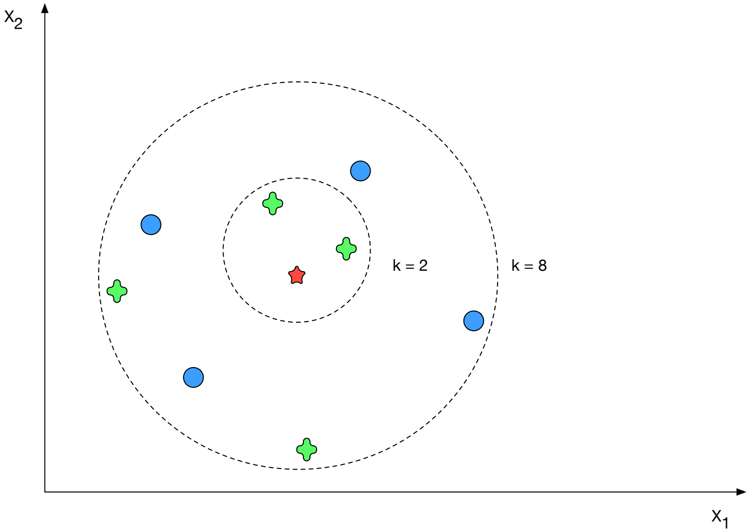
The k-NN algorithm

k-nearest neighbours classifier

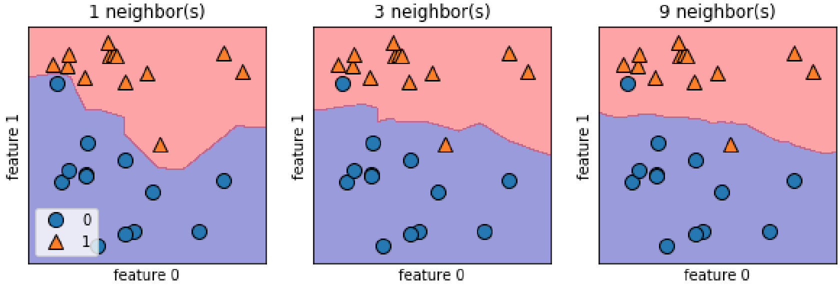
→ For each new point, take the majority of labels of the k-nearest points (given norm such as euclidean) in the training set.

→ If there is no majority, random drawing of the class within the k-NN

The k-NN algorithm



The k-NN algorithm
How to choose an optimal k (hyperparameter)???



Hyperparameters defintion

Hyperparameters

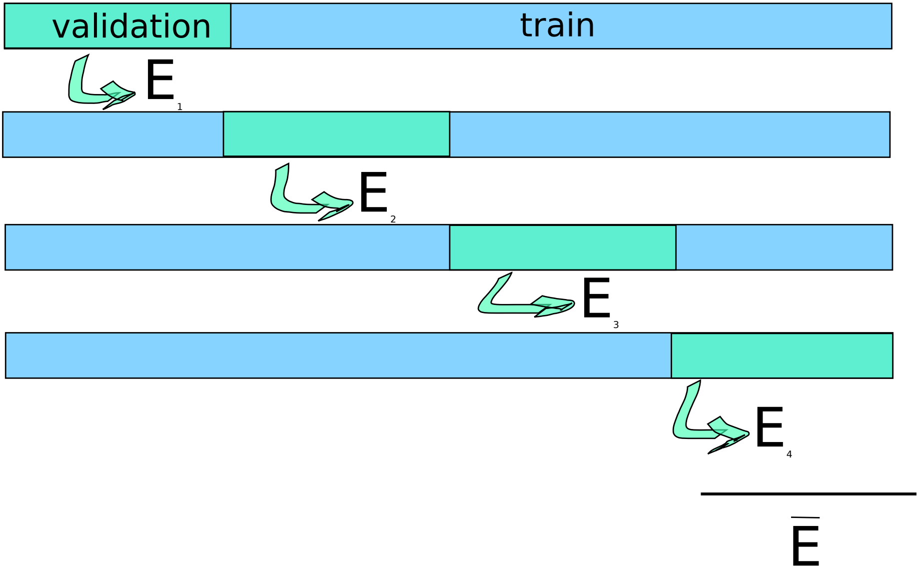
Some parameters defining the 'setup' of an algorithm and fixed before training

Model tuning and model training

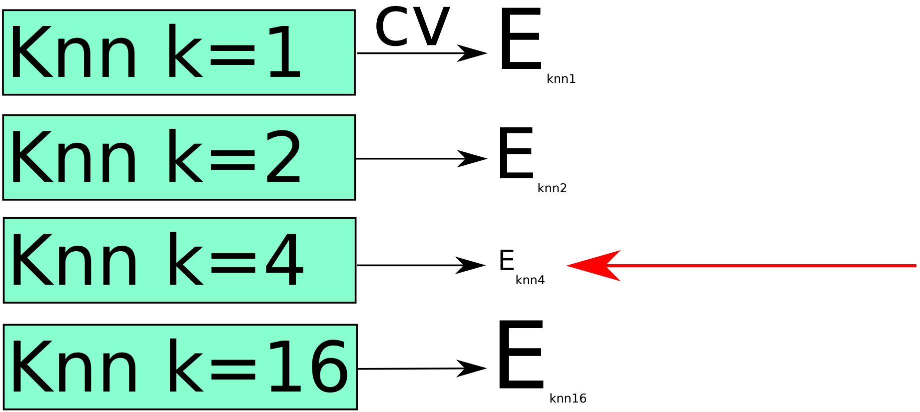
→ Model tuning: Adjusting hyperparameters of the model

→ Model training: For given hyperparameters, run the algorithm (once or several iterations) to minimize loss

k-fold cross-validation



k-fold cross-validation and model tuning



If we want to compare different models?

$\rightarrow$ k-fold CV for hyperparameter tuning ✓

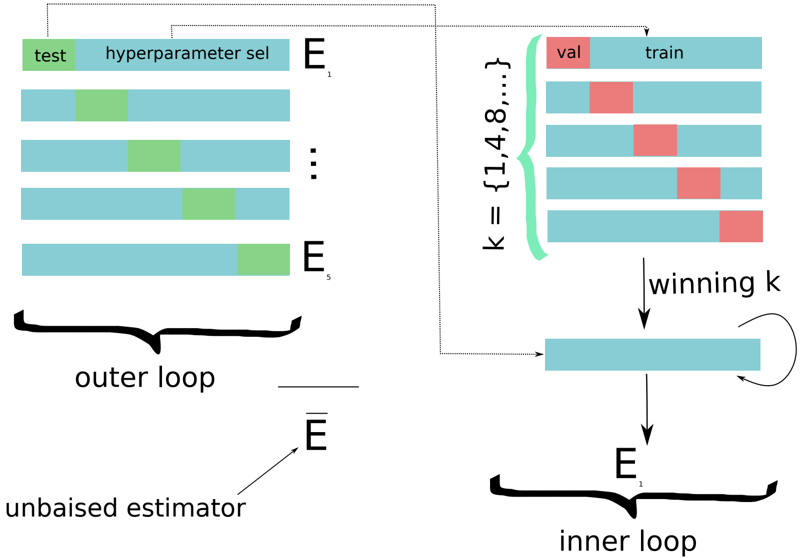
$\rightarrow$ k-fold CV for model selection ✗

Are we safe?

$\rightarrow$ CV for tuning hyper-parameters **BUT** we biased the estimator by over-fitting data (Cawley, Talbot 2010)

- $\hookrightarrow$ Generalization performance will be optimistic!
- $\hookrightarrow$ We need unbiased estimations **to compare algorithms**

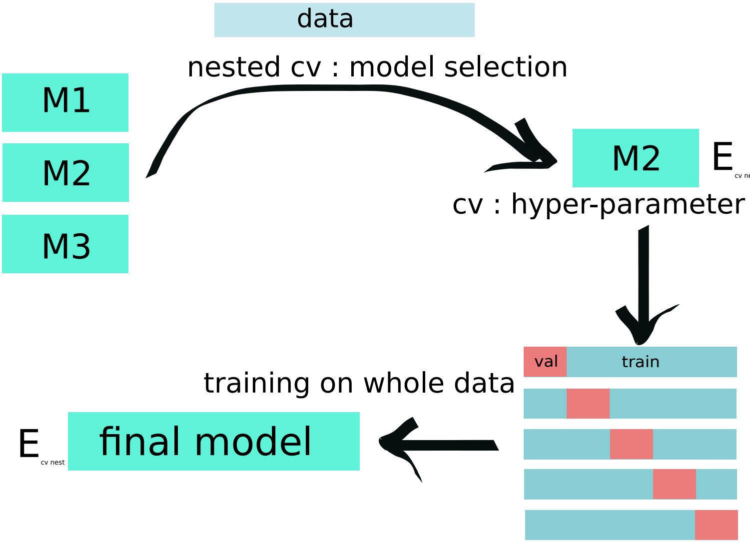
Nested CV: unbiased generalization measure



Use of Nested CV

- $\rightarrow$ Measure generalization error with hyper-parameter tuning
- $\rightarrow$ Fair comparison of different algorithms
- $\rightarrow$ Having a good idea of how model will perform

Model selection



Good practices

→ never present performances based on training set (resubstitution)

→ use at least a hold out test set

→ better use CV

→ best use nested cross validation performances if hyperparameter tuning needed

→ choose relevant metrics

What did we learn?

- $\rightarrow$ minimizing the empirical risk is not enough
- $\rightarrow$ the more complex a model the more prone to overfiting
- $\rightarrow$ a hyper-parameter is a 'structural' parameter to determine before training
- $\rightarrow$ we adjust hyper-parameters by cross-validation
- $\rightarrow$ an unbiased generalization performance measure for a model with hyper-parameters requires nested cross-validation
- $\rightarrow$ we compare models with nested-cross-validation