# Cross-validation and performance measures

## How to avoid prediction blunders

Filippo Biscarini
*Senior Scientist*
*CNR, Milan (Italy)*

Nelson Nazzicari
*Senior scientist*
*CREA, Lodi (Italy)*

# Overfitting

# What is overfitting?

You may fit a deep learning model to your data and then measure the "accuracy" of predictions on the same data: **would this be correct?**

# What is overfitting?

You may fit a deep learning model to your data and then measure the "accuracy"of predictions on the same data: **would this be correct?**

- short answer: **NO!**
- main reason: **overfitting**

# What is overfitting?

Overfitting:

Fitting too well the data: $R^2$ too large (≈1)

# What is overfitting?
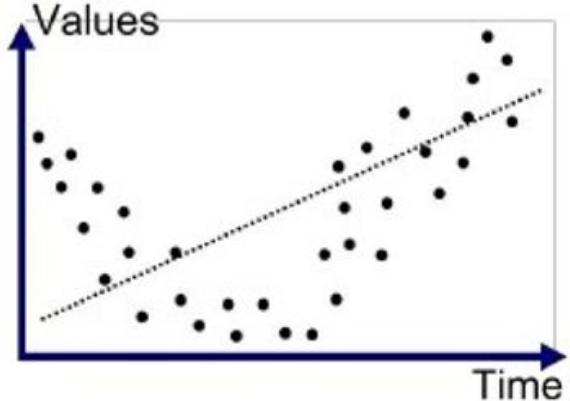
Overfitting:

Fitting too well the data: $R^2$ too large (≈1)

overfitting happens with:

- using the same data to fit the model and to make predictions
- overparameterization of the model (e.g. too many effects)
- flexible methods (e.g. polynomial functions, splines, … and **deep learning!**)
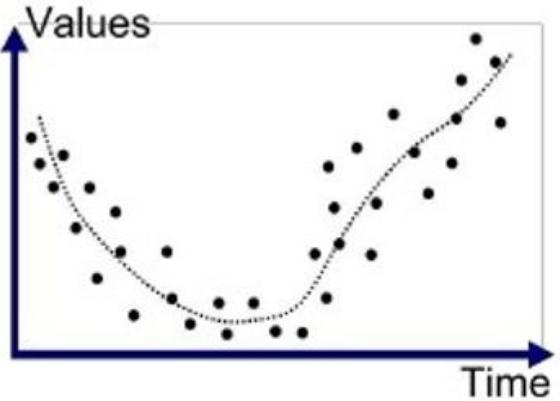
# What is overfitting?
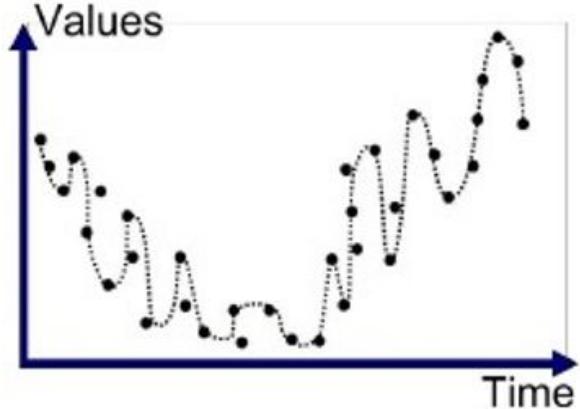


| Underfitted | Good Fit/Robust | Overfitted |
|---|---|---|
| Linear regression | Polynomial (?) | Highly non-linear model |
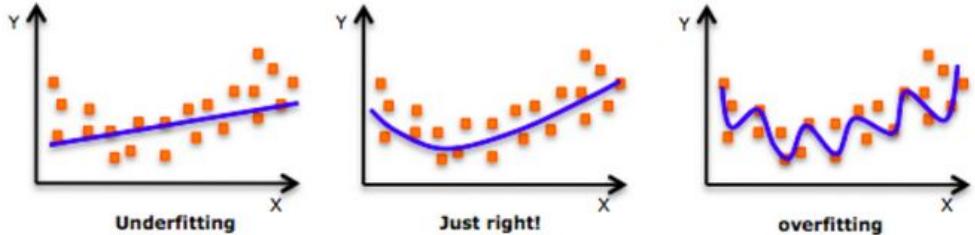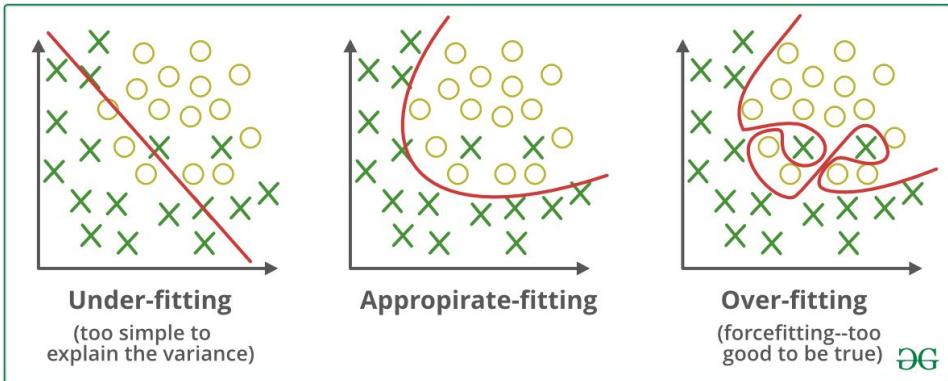
# What is overfitting?



regression

Underfitting     Just right!     overfitting

classification

Under-fitting
(too simple to
explain the variance)

Appropirate-fitting

Over-fitting
(forcefitting--too
good to be true)

Linear regression     Polynomial (?)     Highly
non-linear
model
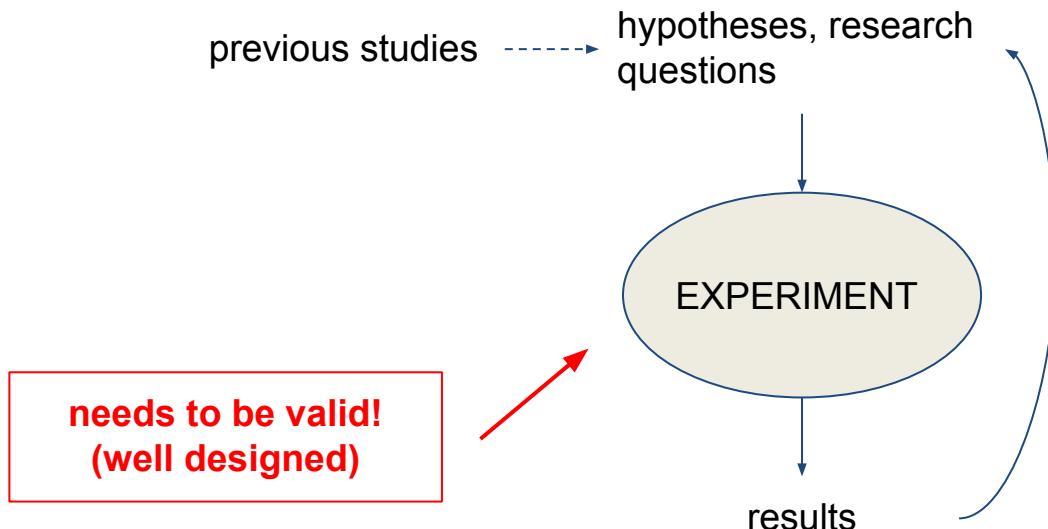
# Training and validation sets

## A tool to control overfitting
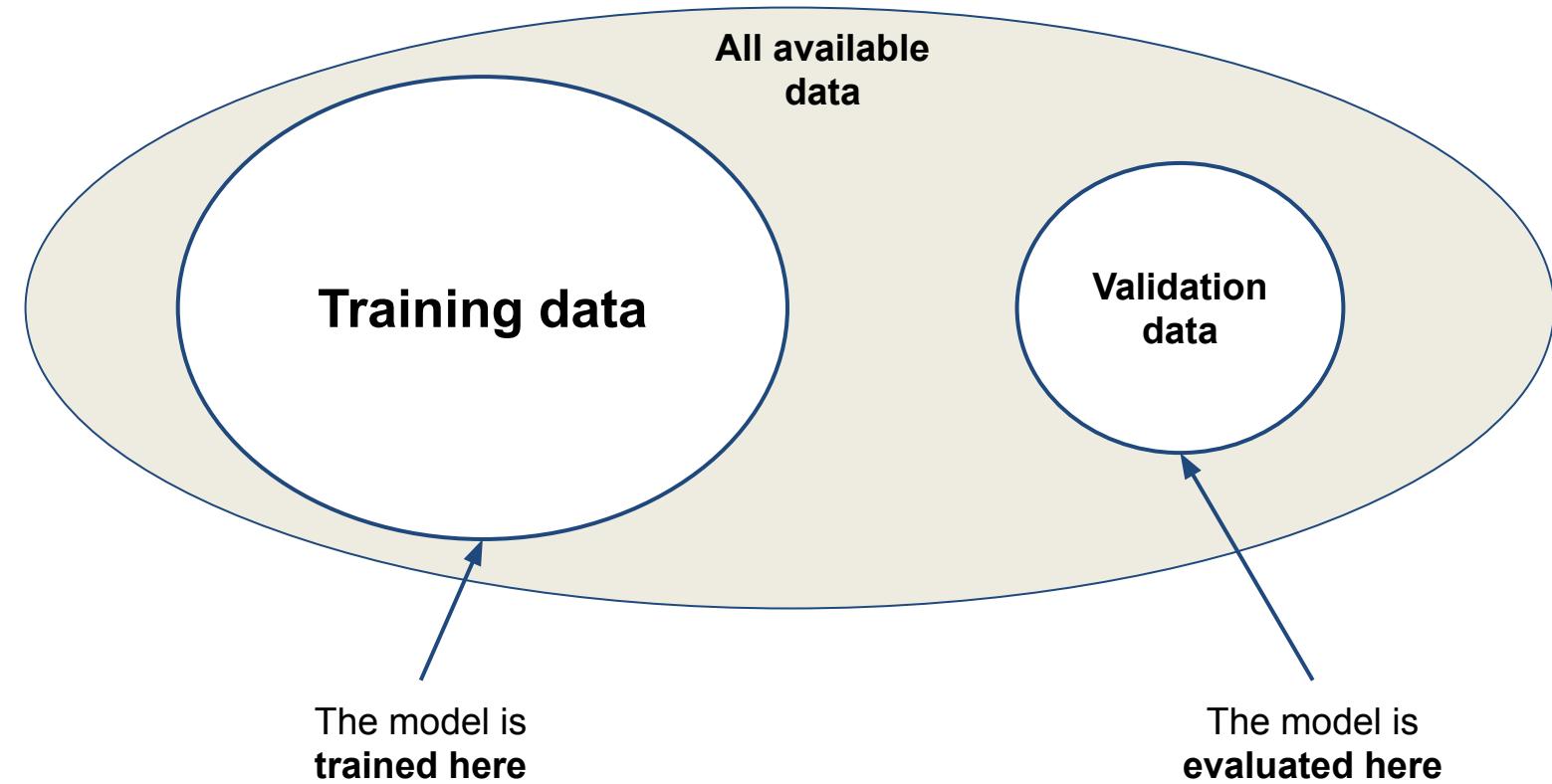
# Training and validation sets

"*The test of all knowledge is experiment*"

previous studies ----→ hypotheses, research
questions

EXPERIMENT

needs to be valid!
(well designed)

results

# Training and validation sets



All available
data

**Training data**

Validation
data

The model is
**trained here**

The model is
**evaluated here**
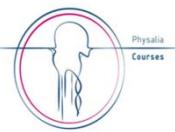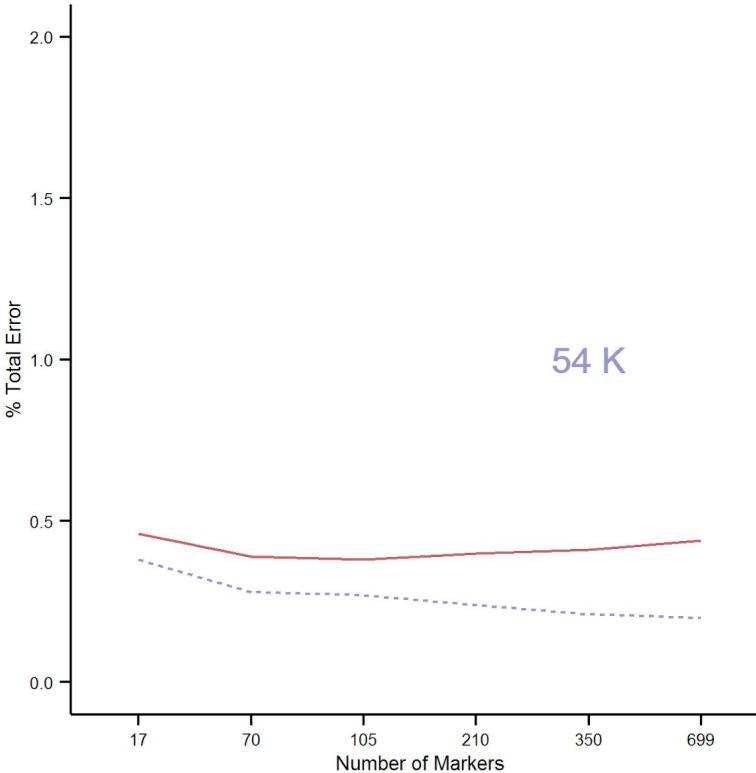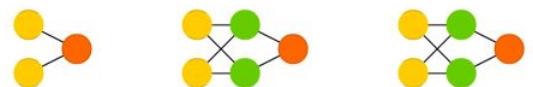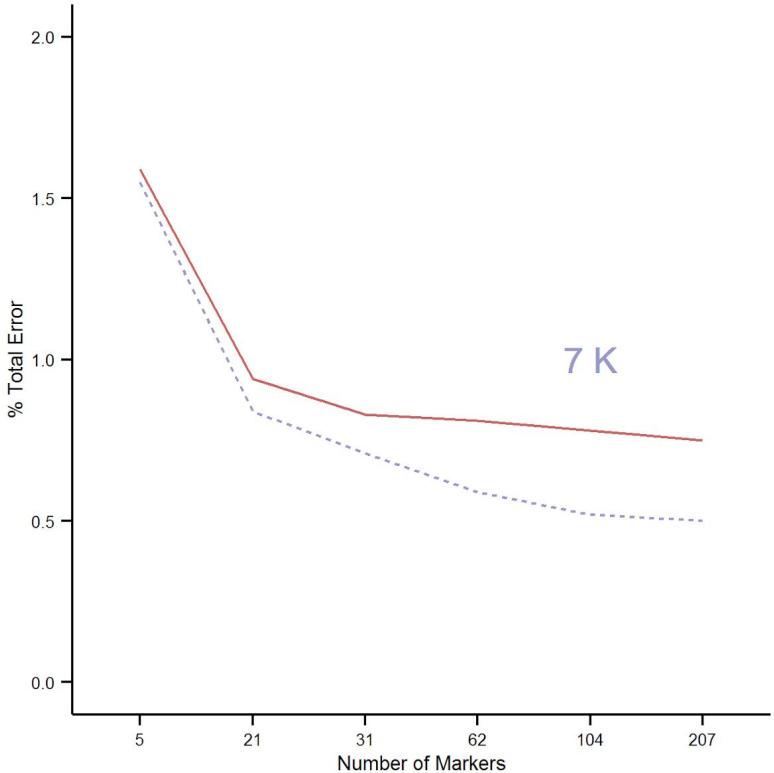
# Training and validation sets

- accuracy (model performance) on the training set is "optimistic" (biased upward ← *overfitting*)
- a better estimate of model performance can be obtained from independent data
- usually we are interested in the predictive performance on new data
- accuracy in the validation set is usually lower than in the training set
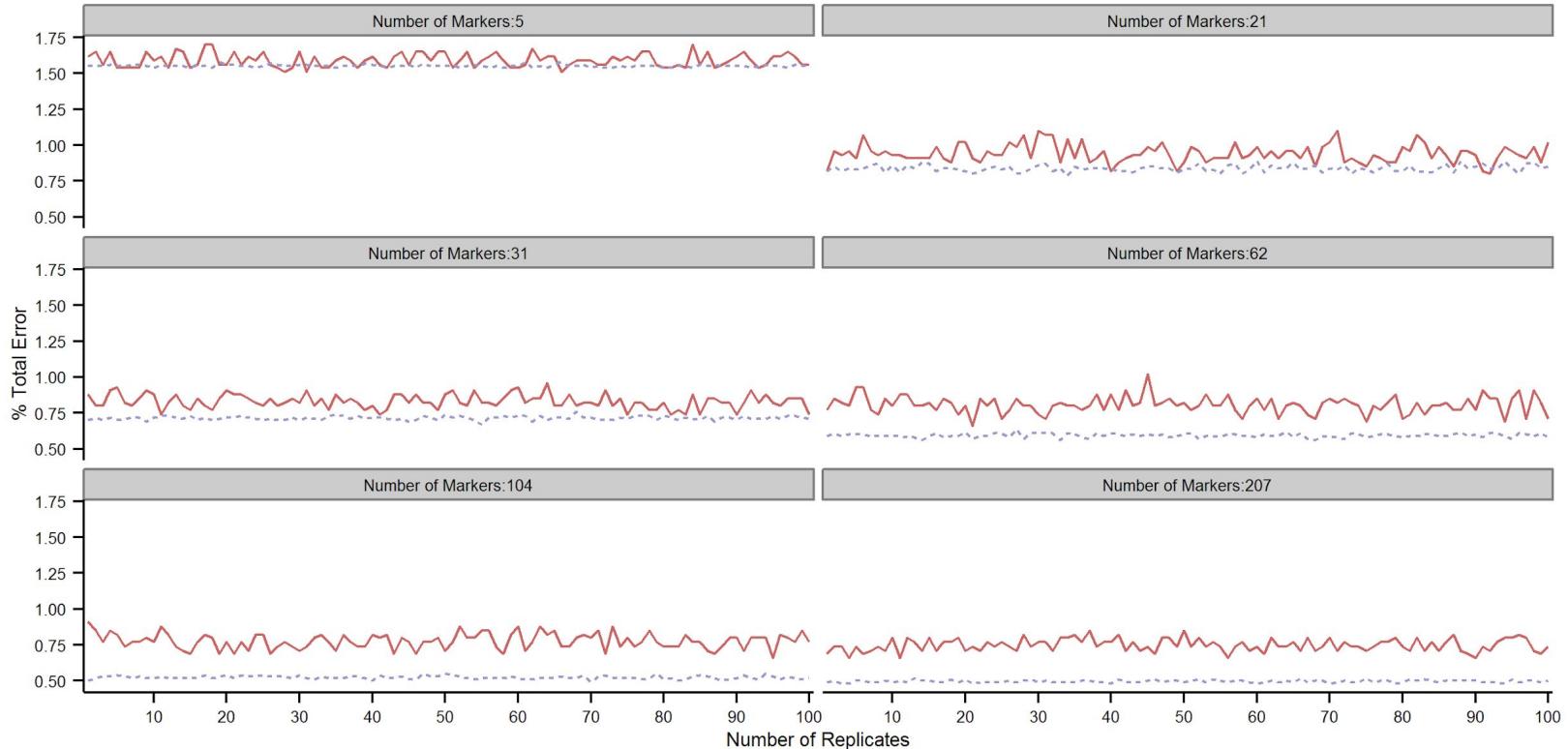
# An example from genomics

# An example from genomics
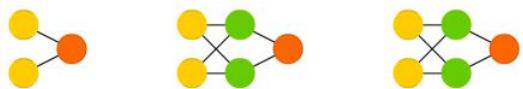
# And the test set?

- "Test" and "Validation" are often considered and used as synonyms
  - But they are not! (strictly speaking)
- A test set would be <u>a third</u> set
  - Completely new data
  - Used only once when you finished everything else
  - An estimate of performances in real world
- We (try to) stick to Keras convention, and never talk about test sets :)
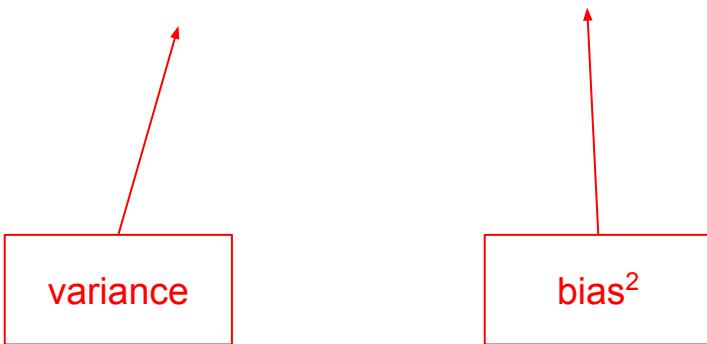
# Prediction error

# Prediction error

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

variance

bias$^2$
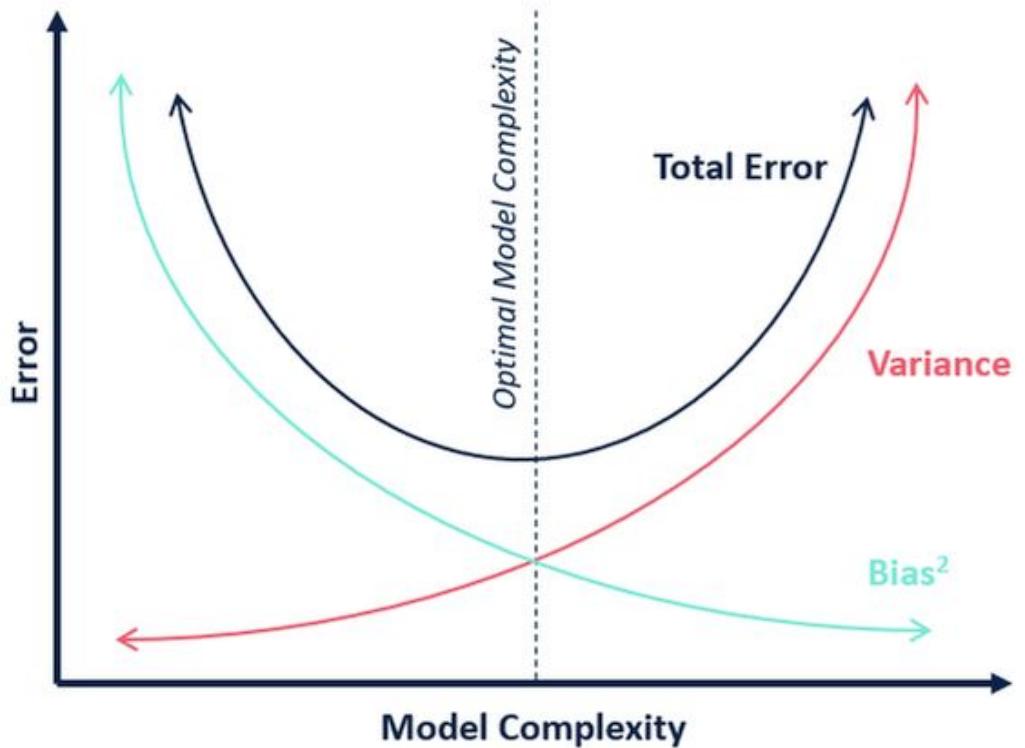
# Prediction error

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

- **variance** refers to the change of the predictor if estimated using different training data
- **bias** refers to the approximation of a real problem by a simpler model

# Bias-variance trade-off



- models with low bias and high variance (e.g. KNN with k=1)

- models with high bias and low variance (e.g. horizontal line crossing the data)

- → find models/methods with both low variance and low bias

Source: https://ai-pool.com/a/s/bias-variance-tradeoff-in-machine-learning

# Bias-variance trade-off

Methods

**low variance and high bias**

- Linear regression
- Logistic regression
- Penalised regression
- SVM (linear kernel)
- Naive Bayes
- etc.

**high variance and low bias**

- Random Forest
- Boosting
- Polynomial regression
- Regression splines
- GAM
- **Deep learning**
- KNN
- SVM (Rbf)
- Loess/Lowess (local regression)
- etc.

# Bias-variance trade-off

Related trade-offs
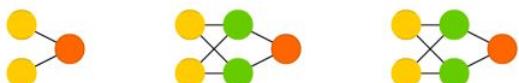
1.  Prediction accuracy vs model interpretability:

    ■   e.g. linear regression is easy to interpret, splines are not

2.  Parsimony vs "black-box":

    ■   e.g. variable selection, all-variables models (e.g. RF), Occam's razor

# Bias-variance trade-off

Important for:
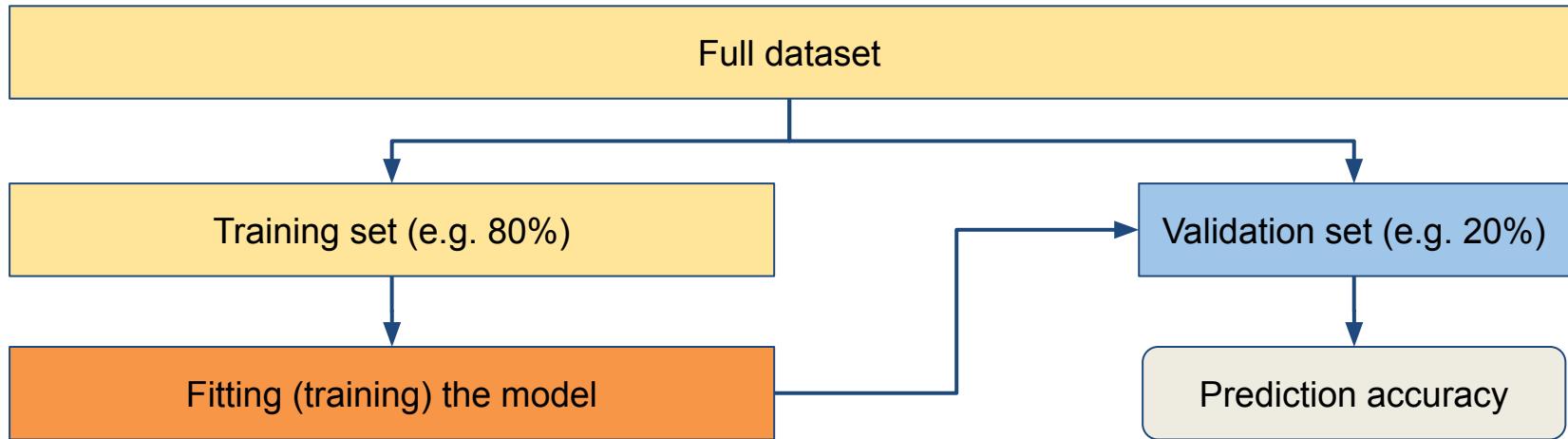
1.  **Correctly estimating the performance of a predictive machine**

2.  Correctly estimating model parameters

3.  Selecting between models

# Resampling methods

# Sampling the training and the validation sets



| Full dataset |
|:---:|

Training set (e.g. 80%) → Validation set (e.g. 20%)

Training set → Fitting (training) the model

Fitting (training) the model → Validation set

Validation set → Prediction accuracy

- To correctly assess the performance of a predictive model we measure it on independent data → validation data
- However we can sample many different training and test sets!
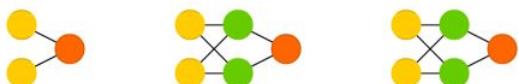
# Resampling the data

- Resampling involves **repeatedly sampling** the training and validation datasets: each time, the model is **refitted** in the training set and **evaluated** in the validation set
- You can e.g. estimate the **variability** of a predictive model or the effect of modifying the model or method:
    - **Model assessment**
    - **Model selection**

# Resampling the data

- Several resampling methods exist
- We will examine two such methods:
  1. **validation set approach**
  2. **cross-validation**
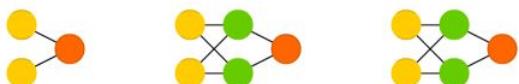
# The validation set approach

| training set | validation set |
|:---:|:---:|

- We split the data in **two random subsets**: training and validation (test)

- 10%/90%, 20%/80%, 30%/70% etc.

- This is what we already did!

- Repeat this *n times* and you get **robust estimates** of the model performance

# The validation set approach
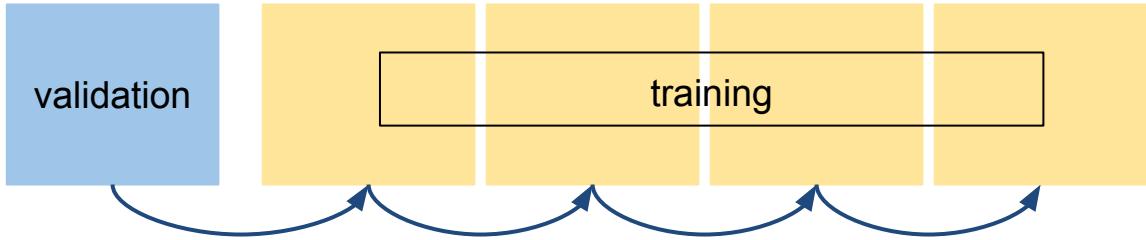
| training set | validation set |
| --- | --- |

Drawbacks:

- **highly variable** (depending on the random partition of the data)
- only a subset of the data is used to train (fit) the model → **potentially underestimate model performance**
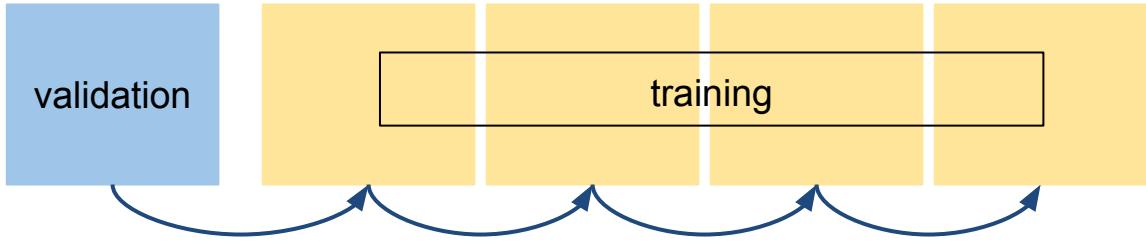
# k-fold cross-validation



- *k* random partitions of equal size
- each partition in turn is used for validation, the rest for training
- *k* estimates of model performance

# k-fold cross-validation



- *k* random partitions of equal size
- each partition in turn is used for validation, the rest for training
- ***k* estimates** of model performance $\longrightarrow$ $CV_{(k)} = \frac{1}{k} \sum_{i=1}^{k} MSE_i$

# k-fold cross-validation

- Lower variability than the validation set approach
- cross-validation works well in **finding the minimum point** in the estimated test MSE curve → model selection
- In cross-validation each observation/record is used both to train the model and to test it → more data are used here than in the validation set approach → lower bias
- cross-validation is therefore expected to have **both lower variance** and **lower bias** than the validation set approach → more accurate estimate of model performance
- typical values for *k* are **k=5** and **k=10** (k = n → LOOCV)

# Cross-validation: right and wrong

- Consider a **regression problem**: **100 samples**, **50,000 features** (variables, e.g. 'omics data):

  - <u>Step 1</u>: Find the 100 features with the **strongest correlation** with the response variable

  - <u>Step 2</u>: Apply a **predictor** (e.g. multiple linear regression) with only these 100 **selected features**
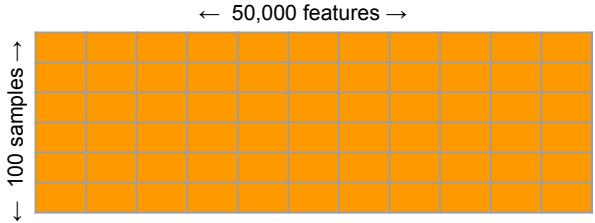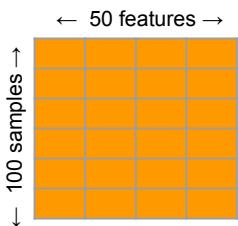
Estimate the **prediction error**: can we apply cross-validation in step 2?

# Cross-validation: right and wrong
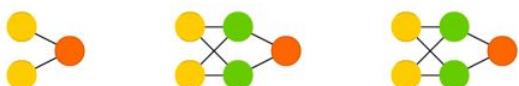
Consider a **regression problem**: **100 samples**, **50,000 features** (variables, e.g. 'omics data):

- <u>Step 1</u>: Find the 50 features with the **strongest correlation** with the response variable (y)

← 50,000 features →

100 samples

← 50 features →

100 samples

<u>Step 2</u>: Do the train/validation split (or k-fold CV) and build a **predictive model** (e.g. multiple linear regression) with only these 50 **selected features**
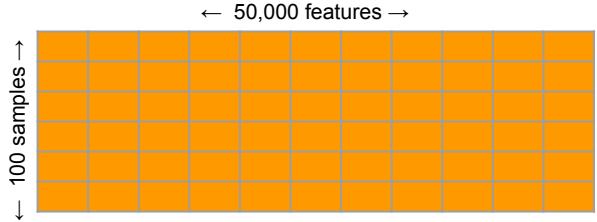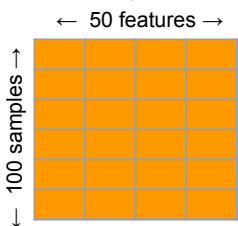
Is this a **right way to apply cross-validation** and estimate the prediction error?

# Cross-validation: right and wrong

Consider a **regression problem**: **100 samples**, **50,000 features** (variables, e.g. 'omics data):

- <u>Step 1</u>: Find the 50 features with the **strongest correlation** with the response variable (y)

← 50,000 features →

100 samples

← 50 features →

100 samples

<u>Step 2</u>: Do the train/validation split (or k-fold CV) and build a **predictive model** (e.g. multiple linear regression) with only these 50 **selected features**

Is this a **right way to apply cross-validation** and estimate the prediction error? → NO!

# Cross-validation: right and wrong

Can we apply cross-validation only in step 2? → **NO!**

- in Step 1, the **model has already used the response** of the training data

- Features have been **"cherry picked"** based on the data: this is already **training**, and the correlation with the response may be a result of the specific configuration of this dataset (a "quirk" in the data)
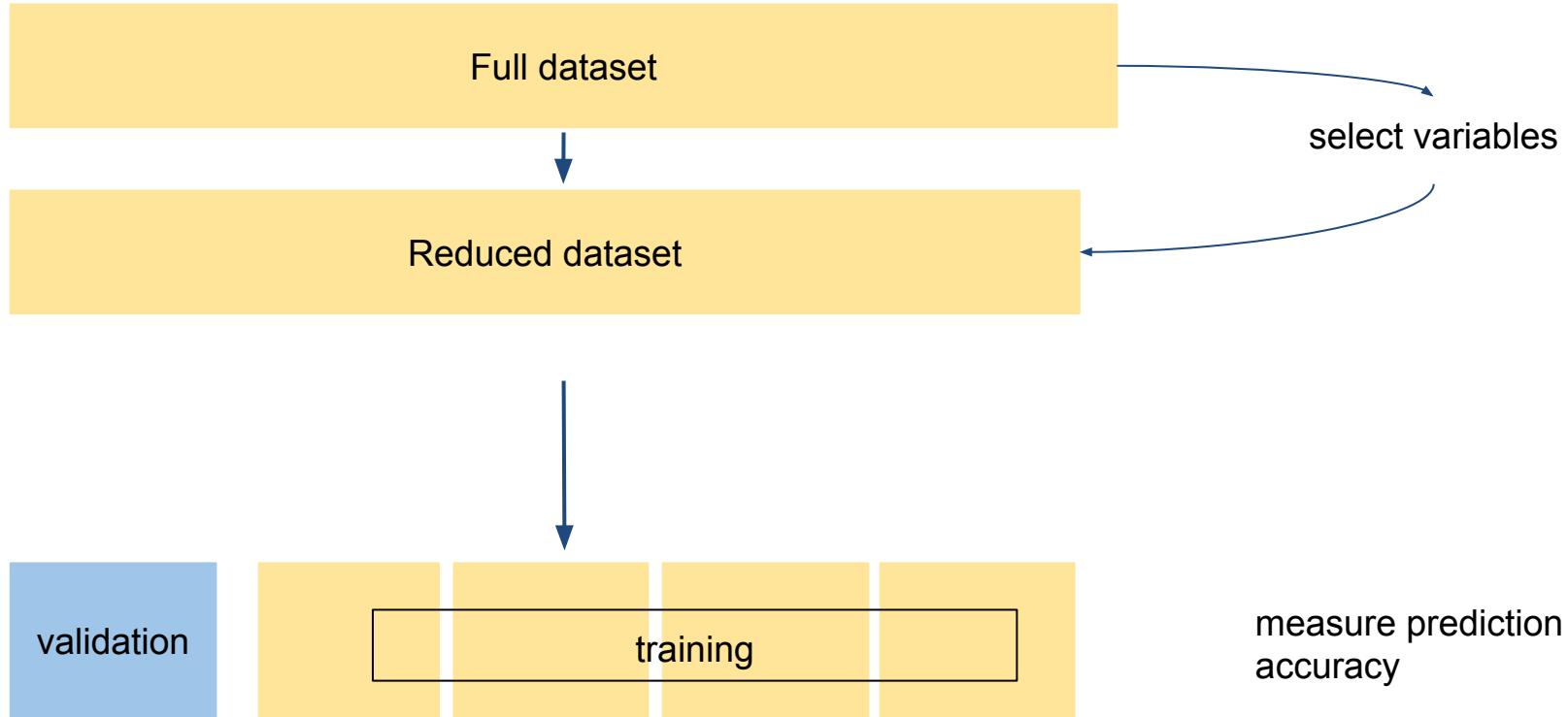
# Cross-validation: right and wrong

Can we apply cross-validation only in step 2? → **NO!**
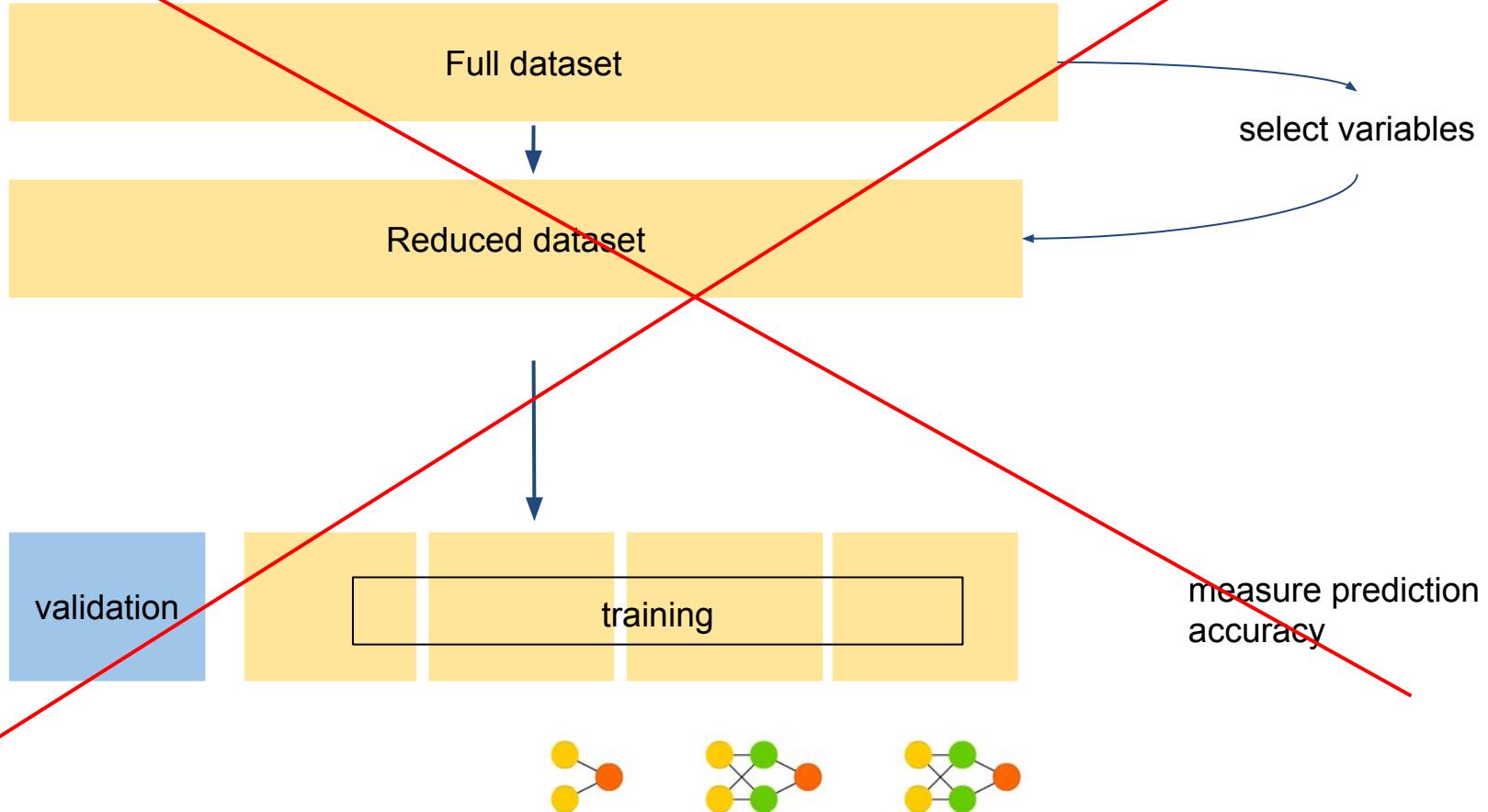
- **Wrong!** → select variables on the whole dataset, then apply cross-validation

- **Right!** → first split the data in training and test sets, then select variables (part of training)

# Cross-validation: wrong way

Full dataset

select variables

Reduced dataset

validation

training

measure prediction accuracy

# Cross-validation: wrong way

Full dataset

select variables

Reduced dataset

validation    training

measure prediction accuracy

# Cross-validation: right way

Full dataset

validation | training

repeat k times

measure
prediction
accuracy

select variables

reduced dataset

# Cross-validation: right way