

STATG019 – Selected Topics in Statistics 2018

Lecture 1

An Introduction to Model Validation

Dr Franz J. Király

Course organization

On-line course

moodle ID 16602

on-line activities, course info and materials

Lecture

Wednesdays, 9 - 11 am

Attendance is **mandatory**

First lecture Feb 21, last lecture Mar 21 (no lecture in reading week)

Slides are available on moodle

Lecture is videotaped and available on Lecturecast/moodle

Feel free to suggest content you want to learn about

Tutorials and/or practical sessions

Thursday, 11am - 1pm

location varies

May take place irregularly, depending on demand

Attendance is **non-compulsory**

Discussion of theoretical or practical exercises, group presentations,
etc...

Your choice.

Course organization

In-Course-Assessment

Two take-home ICA, one on clustering, one on model validation

Each counts 50% towards your final grade

Handing out of ICA no.2: Feb 21 (*today*, on moodle)

Submission deadline of ICA no.2: Apr 25 (via moodle/Turnitin)

Further details will be announced via the moodle news forum

Important ICA rules:

Part group work, part individual work

Group work rule: no two people in same group for ICA1 and ICA2

To ensure this: registration of ICA2 group, by deadline, on moodle

Do *not* discuss with group members before you have registered!

Separate submission portals for group work and individual work

Table of contents

Lecture 1: Introduction to (predictive) model validation

- Scientific validity in the context of the data analytics workflow
- Basic model-agnostic assessment of supervised/predictive models
- Performance quantification of regression and classification models
- Predictive model validation in R/mlr and python/sklearn

Lecture 2: Theory of estimating the generalization error

- Full statistical formulation of the supervised learning setting
- Bias-variance trade-off, cross-validation and re-sampling estimators
- Estimators of the generalization loss and the loss's variance
- Hypothesis testing for pairwise and portmanteau model comparison

Lecture 3: Model selection and model tuning meta-strategies

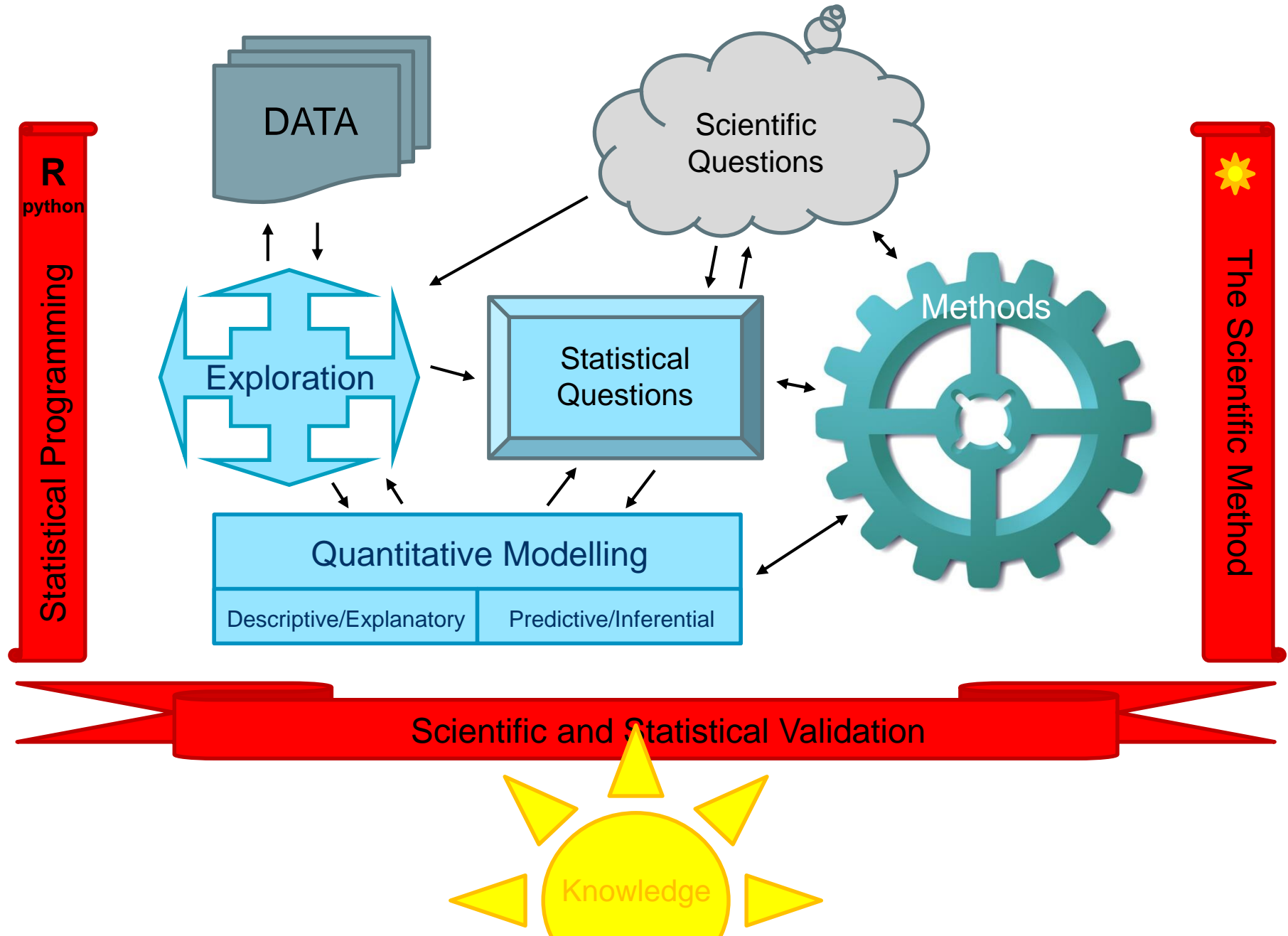
- Meta-strategies for automated model improvement
- Interaction of model tuning and model validation workflows

Potential further lecture topics (moodle vote):

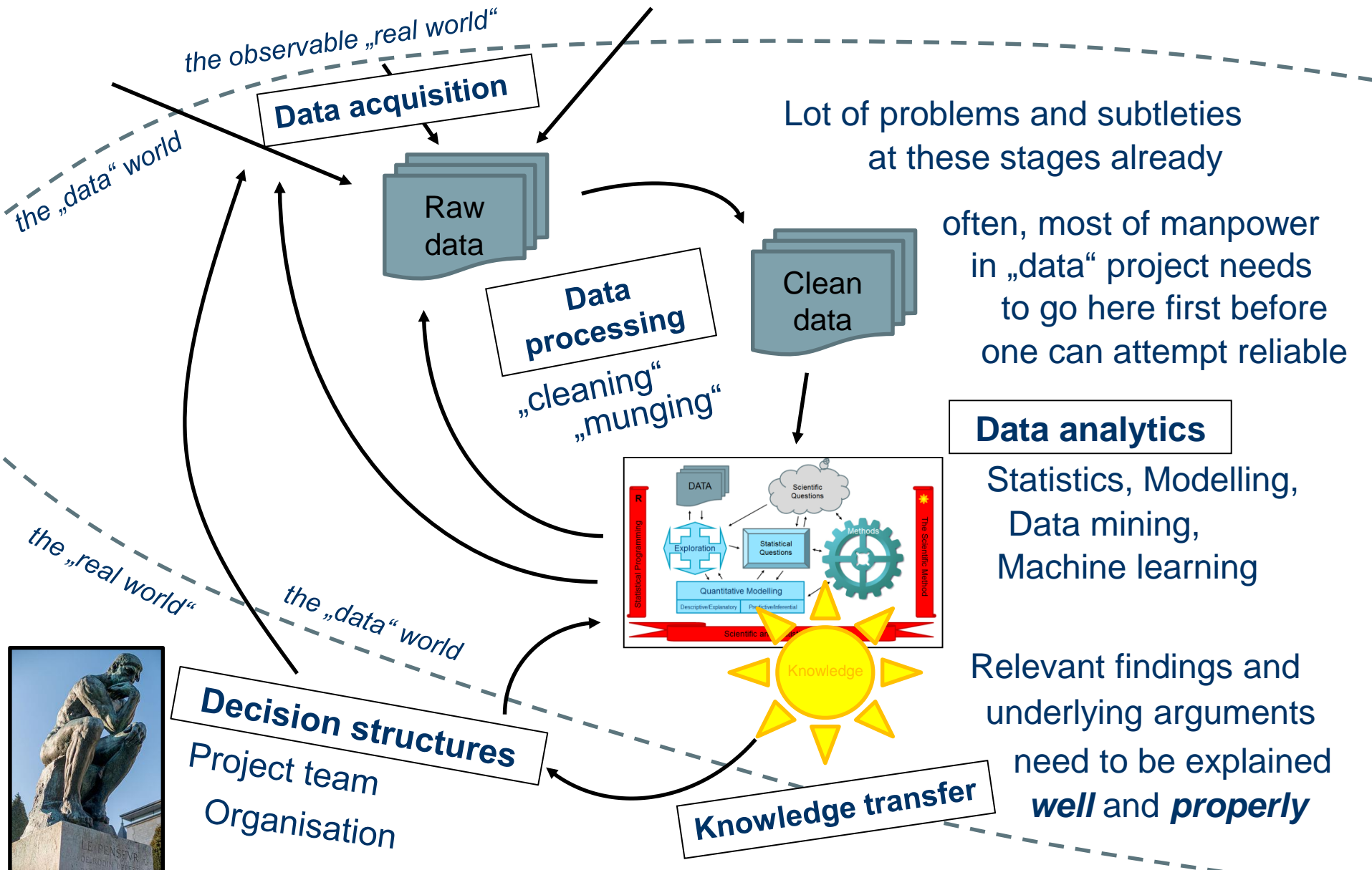
- Advanced tasks: probabilistic; temporal; unsupervised modelling
- Usage of and architecture design for machine learning toolbox environments

An overview of data analytics

(practical)



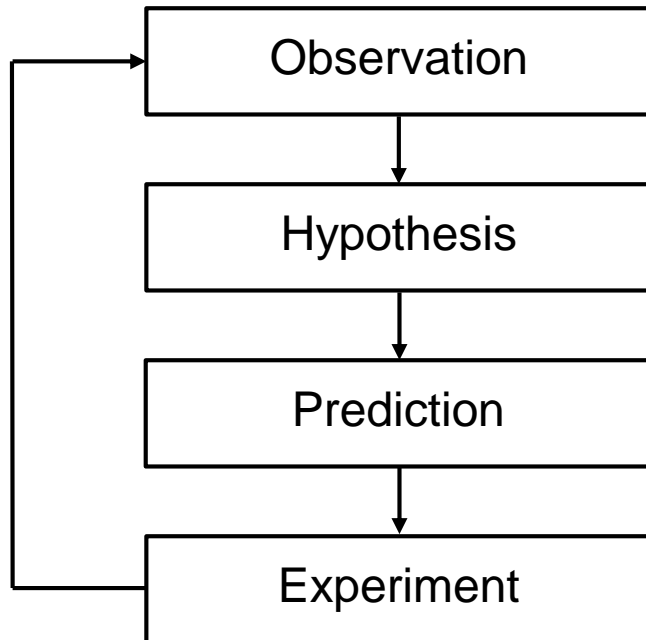
challenges from a broader context



The Scientific Method

quantative version

with Statistics!



Why did this apple fall from the tree?

Drop height/ strange British length unit	1.0	3.9	9.1	16.2	24.9
Drop time/ weird British time unit	.1	.2	.3	.4	.5

Looks like $\text{height} = 100 \text{ time}^2 + \text{residual}$
(I ran polynomial regression in R)

I hypothesize that this will be the case
for any other height/time pair.

Drop height/ strange British length unit	35.9	49.2	64.5	80.5	101
Drop time/ weird British time unit	.6	.7	.8	.9	1.0

Average predictive
relative error = 1%

A good fit!

Conclusion: Hypothesis confirmed. I think the reason
might be an unspecified invisible force. Further
investigation is needed. Increase height? Live specimens?

A primer on Good Data Science

Three questions to ask *first*

1. *What is the scientific question?*

this should be *precisely* formalized

(of course there can be multiple)

Observation and/or **Experiment?** (falsifiable/validable!, how?)

„Let's see what's in the data“

„Can we predict? Which variables are important?“

Summaries and visual exploration

„Associative relation? Causal relation?“

2. *Can this be answered from the data? Why/why not?*

Are there any limitations in measurement/acquisition?

for example, „wasn't measured“

or „it is not clear whether [value] measures it“

Does the data allow to judge causation, or only association?

usually only carefully designed intervention study can evidence causation

3. *What is the statistical question or learning task the scientific question can/should be phrased as?*

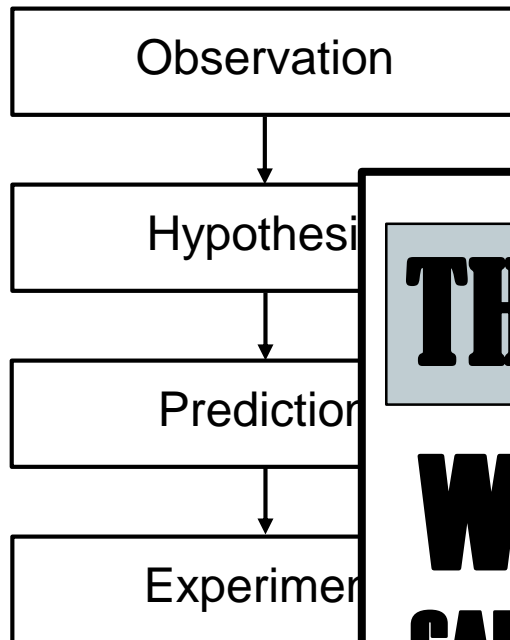
Or are we doing X to do Y to do Z which addresses something similar to the original question?

This may be fine if the original question cannot be directly addressed, but it should be clear.

Supervised prediction? Unsupervised exploration? Model-specific inference?

Which variables can we sensibly use in prediction or analysis?

On experiments and causality



... we examined various clinical parameters in representative populations of **football fans** (N = 1342) versus **non-fans** (N = 1313). Our study, heavily based on deep learning and big data analysis, suggests...

THE INFORMED OBSERVER

WATCHING FOOTBALL CAUSES HAIR LOSS AND BALDNESS

Further dangers include prostate cancer and early death
Famous statistician suspects “confounding” as mechanism



“I’m shocked”

Potential reasons for observed association

If A (age of customer) and B (say repair frequency) are associated, this can be due to the following:

1. A causes B

e.g., people watching soccer experience hair loss because of it

2. B causes A

e.g., people experiencing hair loss watch more football, because of hair loss

3. There is a common cause C for A and B

e.g., people experiencing hair loss watch more football, both because of...?

4. Combinations of the above

5. The observed association is coincidental

Point 5 can (and should) be quantified statistically

p-value = probability the observation arises randomly (from a certain null process)

Points 1-4 can only be distinguished by careful experimental setups!

Three questions to ask *first*

1. *What is the scientific question?*
2. *Can this be answered from the data? Why/why not?*
3. *What is the statistical question or learning task the scientific question can/should be phrased as?*

Case study A:

Retailer wants to understand whether coupons increase customer spending

So complete purchase records are (properly) cleaned, statistically explored
predictive model building is done on a „big data analytics platform“

a boosted tree model is able to predict spending from coupon usage pattern

(improve on uninformed baseline is significant)

precise analysis reveals that indeed more frequent coupon users spend more

Conclusion: Handing out coupons leads to increased customer spending

(anything wrong with this?)

Case study B:

Manufacturer wants to find optimal settings for their machines

Machine settings and manufacturing quality (= target variable)
from many machines are collected in a database and

then analysed by a professional data analytics company in open source software
benchmarking reveals an excellent predictive model for the output quality

(prediction of quality from input setting possible with extremely low error)

Model is then used to find and prescribe best settings for the next 2 month period
(= time it takes to acquire enough data to check the new settings)

Case study C:

The regional police would like to broadly assess known criminals' risk of relapse

Forensic psychiatrists' precise assessment is available in the urban regions
but may be (randomly) missing for more rural areas

The statistics department is tasked to build a model based on demographics
which in the end is able to predict the forensic assessment with high accuracy
(and high sensitivity and high specificity)

Model is then used to screen for high-risk criminals

4. Exploratory Data Analysis

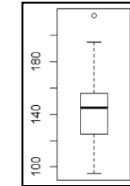
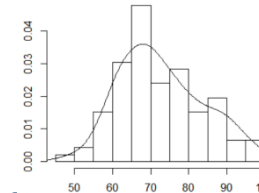
(good practice before doing any modelling)

(a) Inspection of single variables – *of all* (if there's not too many)

Summary statistics: 5-number summary, mean, variance

Stem-and-leaf display, box-plot, histograms, density plots

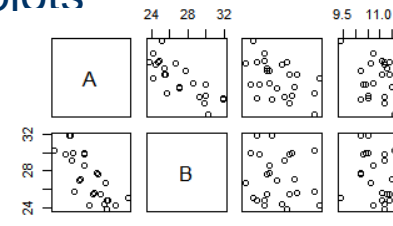
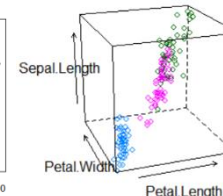
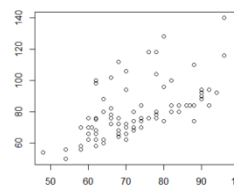
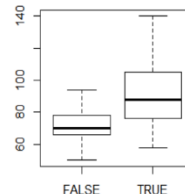
Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
Min. :4.300	Min. :2.000	Min. :1.000	Min. :0.100	setosa :50
1st Qu.:5.100	1st Qu.:2.800	1st Qu.:1.600	1st Qu.:0.300	versicolor:50
Median :5.800	Median :3.000	Median :4.350	Median :1.300	virginica :50
Mean :5.843	Mean :3.057	Mean :5.375	Mean :1.199	
3rd Qu.:6.400	3rd Qu.:3.300	3rd Qu.:5.100	3rd Qu.:1.800	
Max. :7.900	Max. :4.400	Max. :6.900	Max. :2.500	



(b) Bi- and multivariate inspection

Contingency tables, mosaic plot, scatter-plot, Group stratified plots

	Sex	
Class	Male	Female
1st	57	140
2nd	14	80
3rd	75	76
Crew	192	20



(c) Quantification of the above – *quantify relevant statements*

Association: suitable measures, e.g. covariance, Pearson or rank correlation

Difference: suitable hypothesis tests, e.g. t-test, Wilcoxon signed rank test

Blurred line towards modelling:

Advanced descriptive and predictive models, unsupervised learning

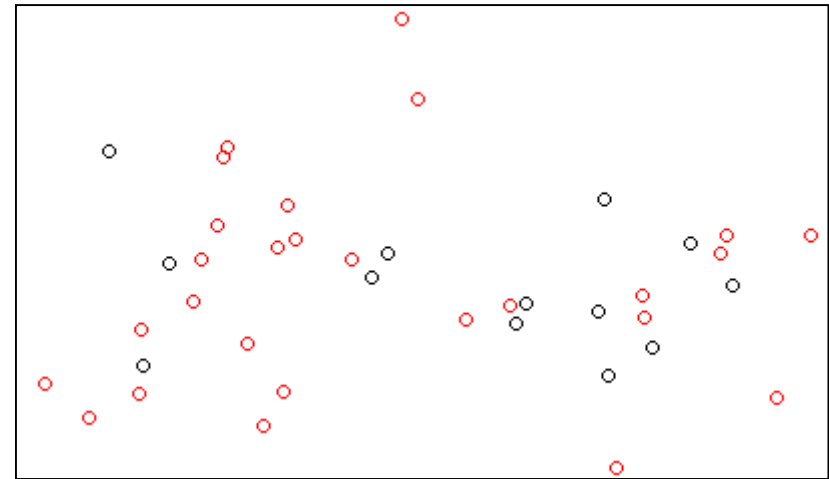
Is the pattern really there?

Humans tend to see patterns where there are none

*„there is an evident pattern“
or „there is clearly a difference“*

is **not** a scientific argument
(just a meaningless claim)

But how do we know
whether there really is a pattern?



These points are coloured completely at random

Solution (E. Pearson, Neyman): ***Quantification!***

First the actual outcome of the experiment

Then how likely that outcome would arise at random

(if unlikely one argues there is a pattern w.r.t. the specific model of random)

Leads to: modern statistics and statistical modelling

Quantifier of pattern/difference: statistical hypothesis testing

5. Common modelling approaches

Association testing and causal inference

Answers „is there relation between X and Y“ and „does change in X cause Y to change?“

Does usually not answer what exactly the relation is, beyond that there is one.

Important: causal relation = association plus careful experimental (interventional) set-up

Predictive modelling and predictive inference

Answers „can we predict Y from X“ together with providing an algorithm to do so

Without the prediction algorithm, „can we predict“ is equivalent to „is there relation“

Important: many algorithms produce a prediction without guarantee, validation is needed

Descriptive modelling and model-specific inference

Answers questions on „how“ of data and relations, given specific model assumptions

E.g., „how many clusters are there“ or „by how much does Y change if X does“

Important: assumption of model is usually not checked by model itself, validation is needed

Exploratory and unsupervised modelling

Answers no question in particular, but often useful in hypothesis generation

Without question, makes no testable statements, so not very helpful on its own

Handle with care, as unsuitable for a scientifically valid argumentation chain

5. Common modelling approaches

Many modelling strategies have multiple aspects/interfaces

Example: ordinary least squares regression (has all)

Association/testing: F-test result, „set of covariates associates with target“

Predictive mode: fitted linear model usable for making predictions

Model-specific descriptive and inferential mode:

coefficients quantitatively describe how covariates and target relate
confidence intervals and t-test results give variable importance

„**White-box models**“ have an easily accessible inferential mode

Stylized advantage: excellent interpretability through parsimony

Stylized disadvantage: don't have model testing or validatory mode

„**Black-box models**“ do not have in-built inferential modes

Stylized advantage: usually yield far more accurate predictions

Stylized disadvantage: no one knows what the model does, or why

Generic validation is usually easiest via „predictive mode“

Recognizing „non-standard“ issues

Non-standard tasks

Does this fit with common approaches, e.g., supervised/unsupervised framework?

Imbalanced class labels, missing labels? Labels revealed through specific process?

Goto on-line, reinforcement learning, semi-supervised; anomaly detection, time series...

Non-standard size

Too many variables? Too many to load, or too many to run specific algorithms?

Too many features? More than 50 may give problems in a number of strategies.

Goto feature extraction, dimension reduction, sub-sampling, „big data“ strategies

Non-standard data format

Are the features „normal“ vectors of discrete categories and continuous numbers?

Text, images, shapes, audio? Series, matrices? Bags-of-features, distributions?

Goto structure-specific methods, feature extraction, kernels, bag-of-methods

Non-standard data properties and sampling process

Strong associations/correlations in the data, particularly between training/test splits?

Failure of i.i.d. assumption in sampling (not all data are in essence/potentially the same)

Goto forecasting, transfer learning, on-line learning, signal processing ... ??

most difficult of all issues, much is open

Validity of Data Analytics Process

Reproducibility – *if it cannot be checked, it cannot be trusted*

Availability of data, documentation of acquisition process and (study) protocol

Availability of code for data cleaning, models and *complete analyses*

Internal validity – *the data admits solid conclusions about itself*

Guaranteed absence of systematic errors in the data acquisition process

Sufficient data quality and quantity for appropriate statistical analyses

External validity – *the analysis admits conclusions that generalize*

Representativeness of the conclusions for the data generating process

Representativeness/relevance of the conclusions for the scientific question

State-of-art model validation and outcome quantification

All models are wrong, but we need to know *how* wrong exactly...

...for the given modelling task.

Technical state-of-art = rest of the lecture

Good data scientific reporting

Awareness of the data context

Why collected? Scientific question? Limitations?

Necessary domain knowledge to spot obvious errors or mistakes

How acquired? Known/potential error sources?

Reproducibility – if it cannot be checked, it cannot be trusted

Clear documentation: summary report, available well-commented code

Clear project structure: in description and in the code

Clear and reproducible workflow: raw data should never be changed

An experienced statistician should be able to re-do everything

Clear discussion and interpretation

Report/paper with short abstract, discussion of the data, question, results

Statistical report with clear argumentation, explanation of analyses

Relevant scientific and domain conclusions, limitations of data and findings

Findings and their interpretation should be separate

Clear discussion and interpretation

The argumentative chain between analyses and scientific questions needs to be clear and precise

Which **quantitative** finding implies
which **scientific** conclusion
implying which **domain** conclusion?

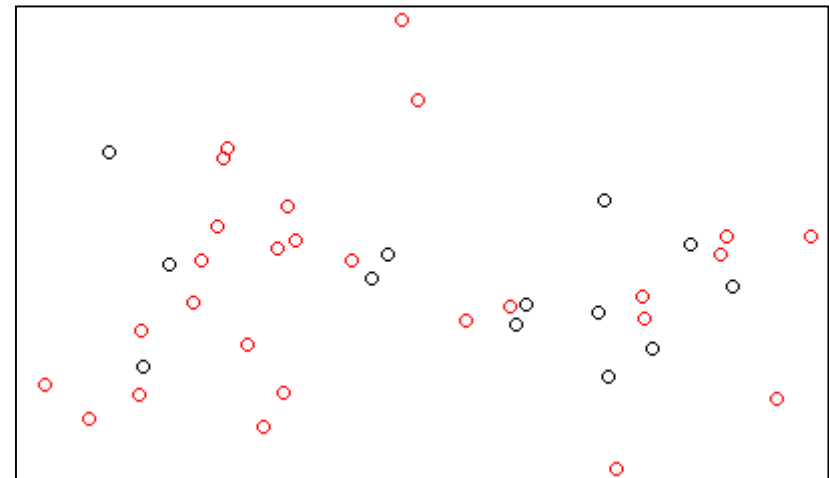
E.g., does association coupons/spending really *imply* causation/likeliness?
(of course not, without the right experimental set-up)

Relevant statements and findings should be quantified

„there is an evident pattern“
or „there is clearly a difference“

is not supported by a plot

There needs to be some quantifier
which could falsify the statement
(e.g., an appropriate hypothesis test)



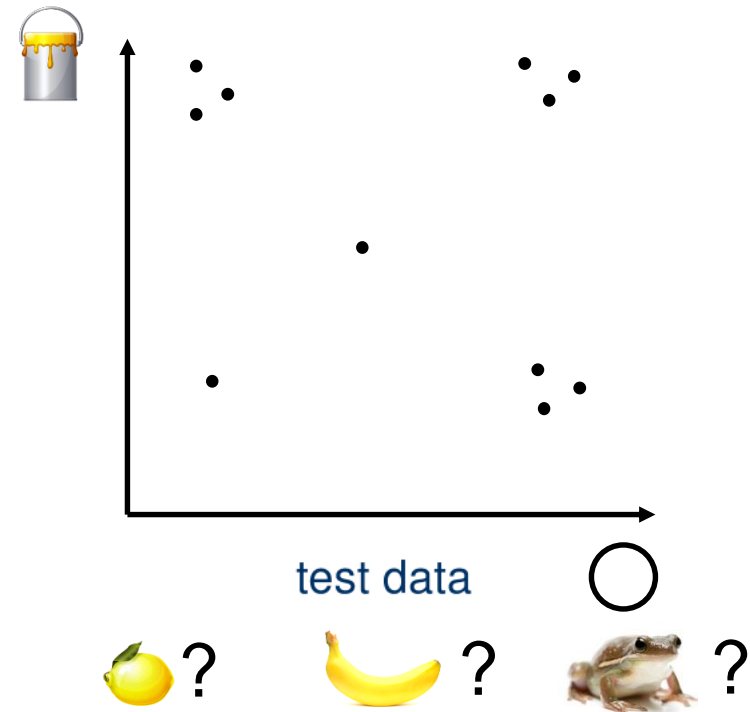
This is a plot which shows nothing, really

External Validation of Predictive/Supervised Models

Overview of Supervised Learning

Supervised Learning:

some data is labelled by expert/oracle



Task: predict label from covariates

statistical models are usually discriminative

Examples: regression, classification

Supervised learning task: Regression

= predicting a continuous target/outcome

Mathematical setting:

Feature variables $x \in \mathbb{R}^n$

Also called: covariates, independent variables

Target variable $y \in \mathbb{R}$ (labels)

Also called: outcome, dependent variable

Training:

Given **Training data** $x_1, \dots, x_N \in \mathbb{R}^n$
Training labels $y_1, \dots, y_N \in \mathbb{R}$

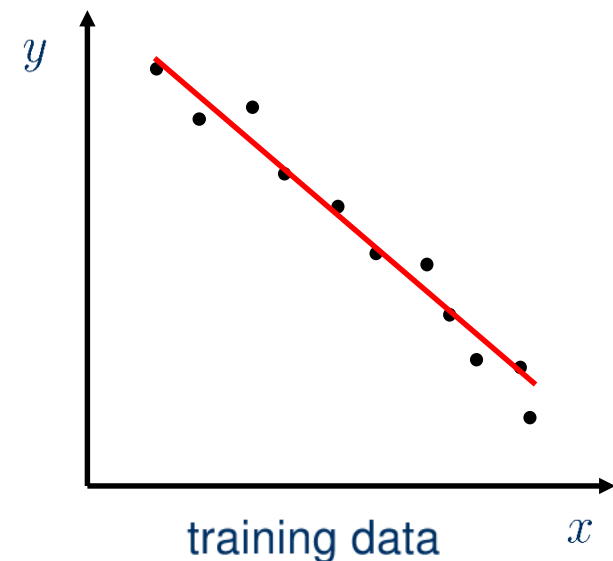
Estimate f such that $y_i \approx f(x_i)$

Prediction:

Given **test data** $x_* \in \mathbb{R}^n$
 predict a **label** $y_* \in \mathbb{R}$

Evaluation:

For true label y_* , difference of y_* and $f(x_*)$



Example models for classification: linear regression,

LASSO, principal components regression, support vector regression,
 kernel ridge regression, random forest, boosted trees, neural networks

Ordinary Least Squares Regression

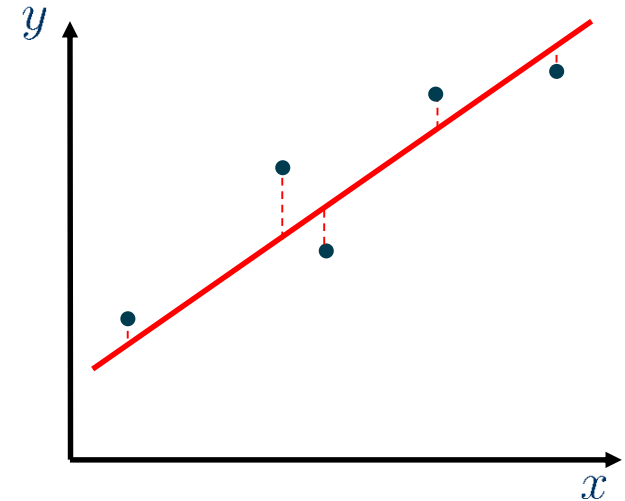
Learning task: *Regression*

Predict target $y \in \mathbb{R}$ **from** features $x \in \mathbb{R}^n$

Main ideas: prediction function $f: \mathbb{R}^n \rightarrow \mathbb{R}$

parametric, linear: $y \approx f(x) = \hat{\alpha} + \langle \hat{\beta}, x \rangle$ $\hat{\alpha} \in \mathbb{R}$
 $\hat{\beta} \in \mathbb{R}^n$

fitted to training data by least-squares principle



Fitting to data $x_1, \dots, x_N \in \mathbb{R}^n$
labels $y_1, \dots, y_N \in \mathbb{R}$

$$(\hat{\beta}, \hat{\alpha}) = (\tilde{X}^\top \tilde{X})^{-1} \tilde{X}^\top \cdot y$$

$X \in \mathbb{R}^{N \times n}$ data matrix $\tilde{X} = (X, 1)$

Analytic minimizer of residual sum-of-squares

Predicting from features $x_* \in \mathbb{R}^n$
a label $y_* \in \mathbb{R}$

Substitute into fitted prediction function

$$\hat{y}_* = f(x_*) = \hat{\alpha} + \langle \hat{\beta}, x_* \rangle$$

Tuning parameters

Whether there
is an intercept yes/no

$$\hat{\alpha} = 0?$$

Advantages

classical, well-studied, often-used
informed baseline, parsimonious
works well quite often, must-do

Disadvantages

unsuitable for non-linear data
does not select features
sensitive to outliers

Variants non-linear & kernel regression
the LASSO robust linear regression

Ref's EoSL chapter 3 and section 3.2
Pearson 1930 Galton 1880s

Interpolation with Multidimensional Splines

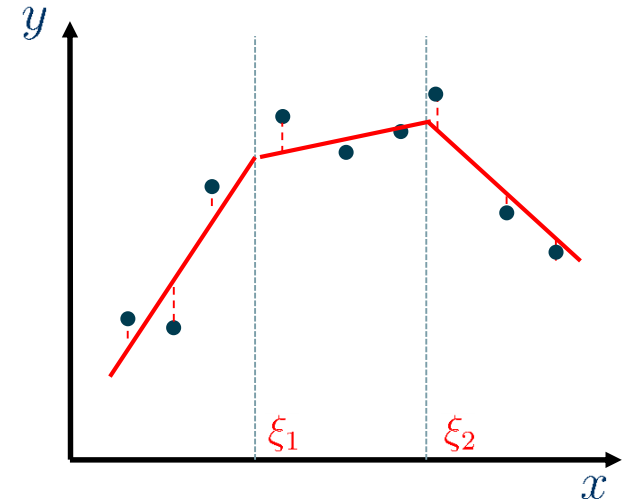
Learning task: *Regression*

Predict target $y \in \mathbb{R}$ **from** features $x \in \mathbb{R}^n$

Main ideas: Piecewise cubic prediction $f : \mathbb{R}^n \rightarrow \mathbb{R}$

f polynomial in piecewise basis functions

least-squares-type penalty, derivative compatibility



Fitting to data $x_1, \dots, x_N \in \mathbb{R}^n$
labels $y_1, \dots, y_N \in \mathbb{R}$

Choose „nodes“ $\xi_1^{(i)}, \dots, \xi_k^{(i)} \in \mathbb{R}$ $i = 1, \dots, n$

$$f(x) = \sum_h \theta_h \cdot h(x) \quad \begin{array}{l} \text{with basis functions} \\ h \text{ of form } (x_i - \xi)_+^d \end{array}$$

find θ minimizing some loss $L(\theta)$

Predicting from features $x_* \in \mathbb{R}^n$
a label $y_* \in \mathbb{R}$

Substitute into fitted prediction function

$$\hat{y}_* = f(x_*) = \sum_h \theta_h \cdot h(x_*)$$

Tuning parameters

The nodes $\xi_1^{(i)}, \dots, \xi_k^{(i)} \in \mathbb{R}$

The loss function $L(\theta)$

Degree and form of polynomials

Explicit/implicit smoothness

Advantages

classical, well-studied, often-used

Lot of choice in parameter settings

Excellent in low dimensions $n \leq 3$

Disadvantages

Lot of choice in parameter settings

Computational explosion and

„curse of dimensionality“ for $n \geq 3$

Variants Cubic splines (special case)

Wavelets

Smoothing splines

Ref's EoSL chapter 5 and section 5.2

De Boor 1978

Daubechies 1992

Supervised learning task: Classification

= predicting a categorical target/outcome

Mathematical setting:

Feature variables $x \in \mathbb{R}^n$
 Target variable $y \in L$ (labels)
 e.g., $L = \{\text{banana}, \text{lemon}\}$

Training:

Given **Training data** $x_1, \dots, x_N \in \mathbb{R}^n$
Training labels $y_1, \dots, y_N \in \mathbb{R}$

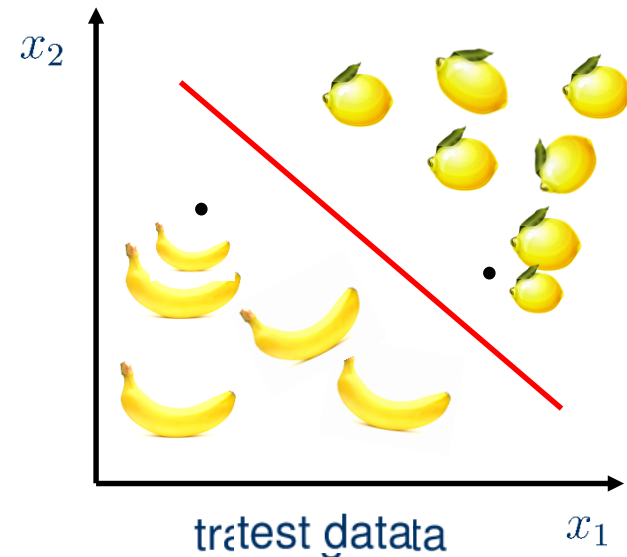
Estimate f such that $y_i \approx f(x_i)$

Prediction:

Given **test data** $x_* \in \mathbb{R}^n$
 predict a **label** $y_* \in \mathbb{R}$

Evaluation:

For true label y_* , whether $y_* = f(x_*)$



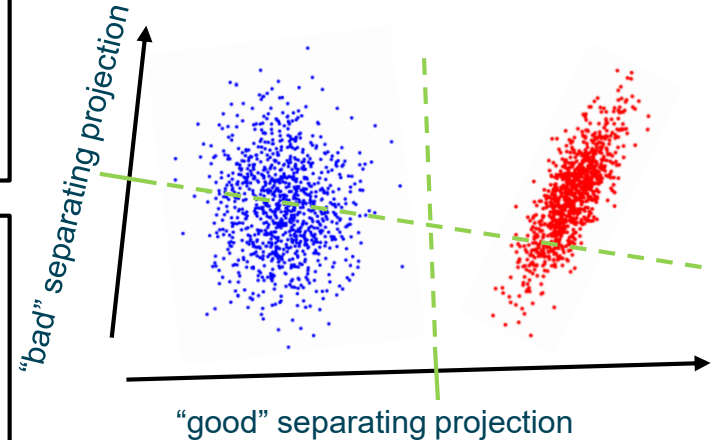
Example models for classification:

linear discriminant analysis, logistic regression,
 support vector machine, random forest, neural networks

Linear Discriminant Analysis

Learning task: *Classification* $L = \{-1, 1\}$
(two-class)
Predict target $y \in L$ **from** features $x \in \mathbb{R}^n$

Main ideas: prediction function $f: \mathbb{R}^n \rightarrow L$
Linear, thresholded: $y \approx f(x) = \text{sgn}(\hat{\alpha} + \langle \hat{\beta}, x \rangle)$ $\hat{\alpha} \in \mathbb{R}$
 $\hat{\beta} \in \mathbb{R}^n$
fitted to data by minimizing „separation“ measure



Fitting to data labels $x_1, \dots, x_N \in \mathbb{R}^n$
 $y_1, \dots, y_N \in L$
Min. of „separation“
$$S(\beta) = \frac{(\beta^\top \mu_1 - \beta^\top \mu_{-1})^2}{\beta^\top (\Sigma_{-1} + \Sigma_1) \beta}$$

 $\mu_i = \text{mean}(\text{class } i)$
 $\Sigma_i = \text{cov}(\text{class } i)$
yields analytic solution
 $\beta = (\Sigma_{-1} + \Sigma_1)^{-1} \cdot (\mu_1 - \mu_{-1})$
common offset:
 $\alpha = \beta^\top \cdot \frac{1}{2} (\mu_1 + \mu_{-1})$

Predicting from features $x_* \in \mathbb{R}^n$
a label $y_* \in L$
Substitute into fitted prediction function
$$\hat{y}_* = f(x_*) = \text{sgn}(\hat{\alpha} + \langle \hat{\beta}, x_* \rangle)$$

Tuning parameters
Choice of offset, e.g.,
whether there
is an offset yes/no

Advantages

classical, well-studied, often-used
informed baseline, parsimonious
works well quite often

Disadvantages

unsuitable for non-linear data
does not select features
sensitive to outliers

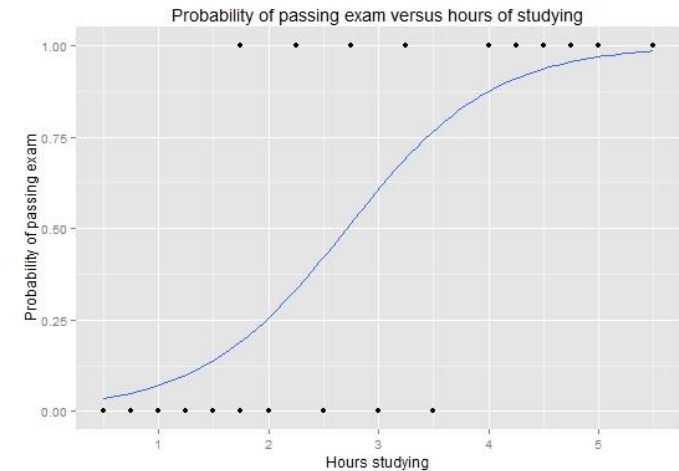
Variants non-linear & kernel LDA
Regularized LDA Parametric LDA

Ref's EoSL section 4.3
Fisher 1936

Logistic Regression

Learning task: *Classification* $L = \{0, 1\}$
(two-class)
Predict target $y \in L$ **from** features $x \in \mathbb{R}^n$

Main ideas: prediction function $f: \mathbb{R}^n \rightarrow \mathbb{R}$
Linear, thresholded: $\log\text{-odds}(y) \approx f(x) = \hat{\alpha} + \langle \hat{\beta}, x \rangle$ $\hat{\alpha} \in \mathbb{R}$
 $\hat{\beta} \in \mathbb{R}^n$
fitted to data by maximum likelihood estimation



Fitting to data $x_1, \dots, x_N \in \mathbb{R}^n$
labels $y_1, \dots, y_N \in L$
obtain α, β numerically maximizing the likelihood
$$\log \mathcal{L}(\alpha, \beta | x, y) = \sum_{i=1}^N y_i \log p_i + (1 - y_i) \log(1 - p_i)$$

where $p_i = \text{logistic}(\langle \beta, x_i \rangle + \alpha)$
= inverse-log-odds

Predicting from features $x_* \in \mathbb{R}^n$
a label $y_* \in L$
Substitute into fitted prediction function
$$\hat{y}_* = f(x_*) = \text{round}(\hat{\alpha} + \langle \hat{\beta}, x_* \rangle)$$

(without rounding: probability of label)

Tuning parameters
Choice of offset, e.g.,
whether there
is an offset yes/no
Class cut-off inside $[0, 1]$

Advantages

classical, well-studied, often-used
informed baseline, parsimonious
works well quite often

Disadvantages

unsuitable for non-linear data
does not select features
non-analytic, non-convex fitting

Variants Fully Bayesian log.reg.
Multinomial log.reg. Neural networks

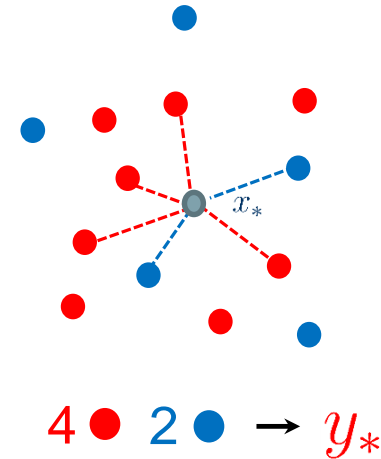
Ref's EoSL section 4.4
Cox 1958

K-Nearest Neighbor Classifier

Learning task: *Classification* $L = \{-1, 1\}$
(two-class)
Predict target $y \in L$ **from** features $x \in \mathbb{R}^n$

Main ideas:

Predict majority label of the K closest previously seen points
„closest“ is measured by a distance function



Fitting to **data** $x_1, \dots, x_N \in \mathbb{R}^n$
labels $y_1, \dots, y_N \in L$

all data and labels are „remembered“
since required in the prediction step

$K = N$ is special case of „uninformed“ majority

Predicting from **features** $x_* \in \mathbb{R}^n$
a label $y_* \in L$

among x_1, \dots, x_N , determine the K closest to x_*
„nearest neighbors“

w.r.t. some pre-specified distance $d(x_i, x_*)$
e.g. Euclidean

predict majority label of nearest neighbors

Tuning parameters

K (= the number of neighbors)

the distance function

optional: weighting rule

Advantages

classical, well-studied, often-used
parsimonious, quick to compute
often good accuracy, baseline

Disadvantages

Highly sensitive to the distance
does not capture „global“ features
there is no „model“ to easily store

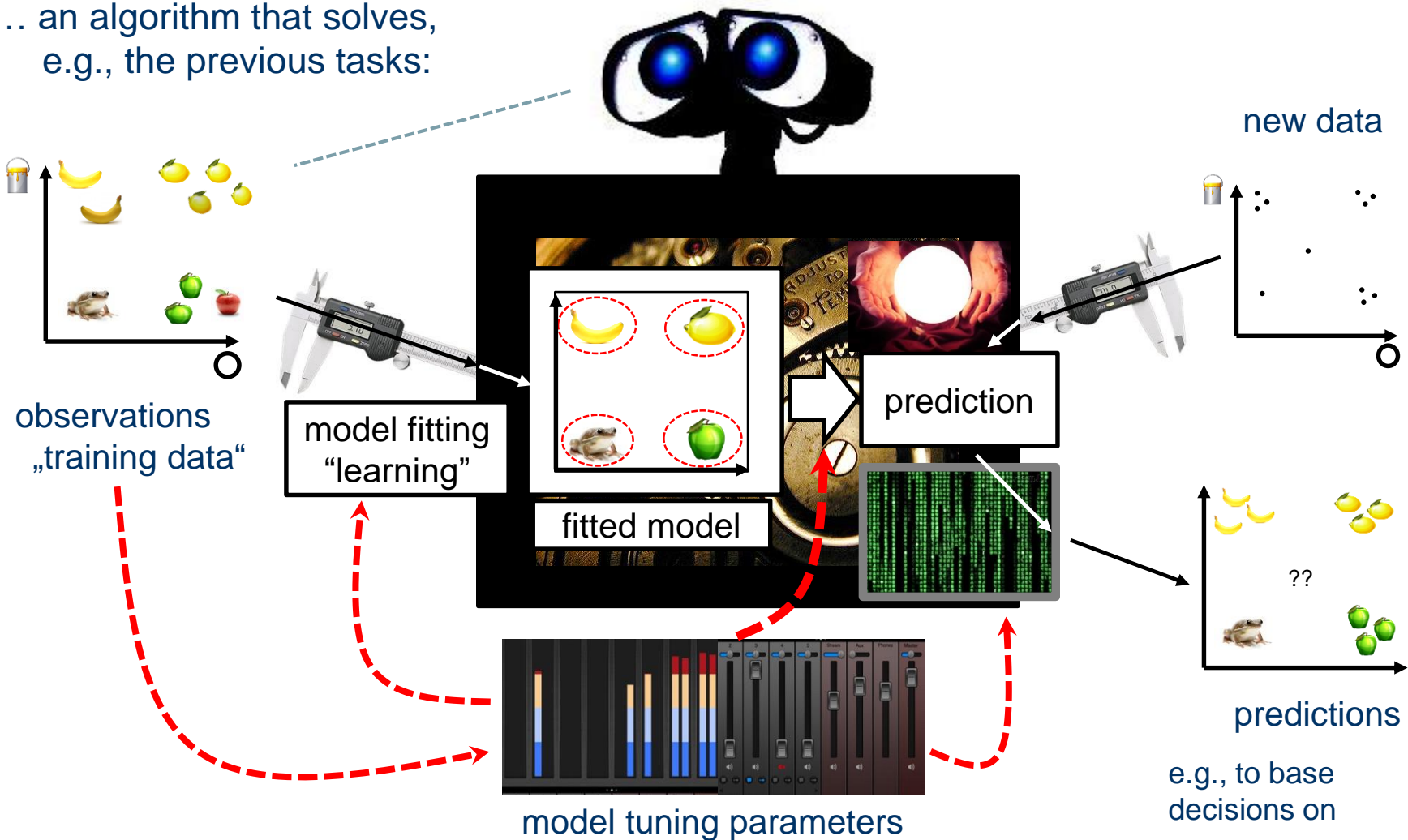
Variants non-linear & kernel k-NN

Smoothed k-NN Adaptive k-NN

Ref's EoSL chapter 13 and section 13.3

Fix, Hodges 1951 Dasarthy1991

... an algorithm that solves,
e.g., the previous tasks:



Validating predictive models

How does one check whether a predictive model makes sense?

Model validation and model selection

= *data-centric and data-dependent modelling*

a scientific *necessity* implied by the scientific method and the following:

1. *There is no model that is good for all data.*

(otherwise the concept of a model would be unnecessary)

2. *For given data, there is no a-priori reason to believe that a certain type of model will be the best one.*

(any such belief is not empirically justified hence pseudoscientific)

3. *No model can be trusted unless its validity has been verified by a model-independent argument.*

(otherwise the justification of validity is circular hence faulty)

Machine learning provides algorithms & theory for meta-modelling and powerful algorithms motivated by meta-modelling optimality.

Which model/method is best?

Occam's razor = law of parsimony:

Frustra fit per plura quod potest fieri per pauciora.

William of Ockham, in: Summa Totius Logicae (ca 1323)

Entia non sunt multiplicanda praeter necessitatem.

A more popular rephrasing by John Punch

= the simplest model/most intuitive method is best

Problem: what does „simple“ or „intuitive“ mean?



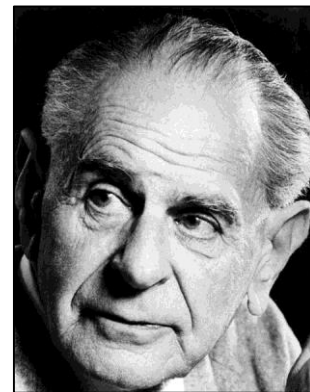
Falsifiability/prediction strength:

[a good model] must accurately describe a large class of observations on the basis of a model that contains only a few arbitrary elements, and it must make definite predictions about the results of future observations.

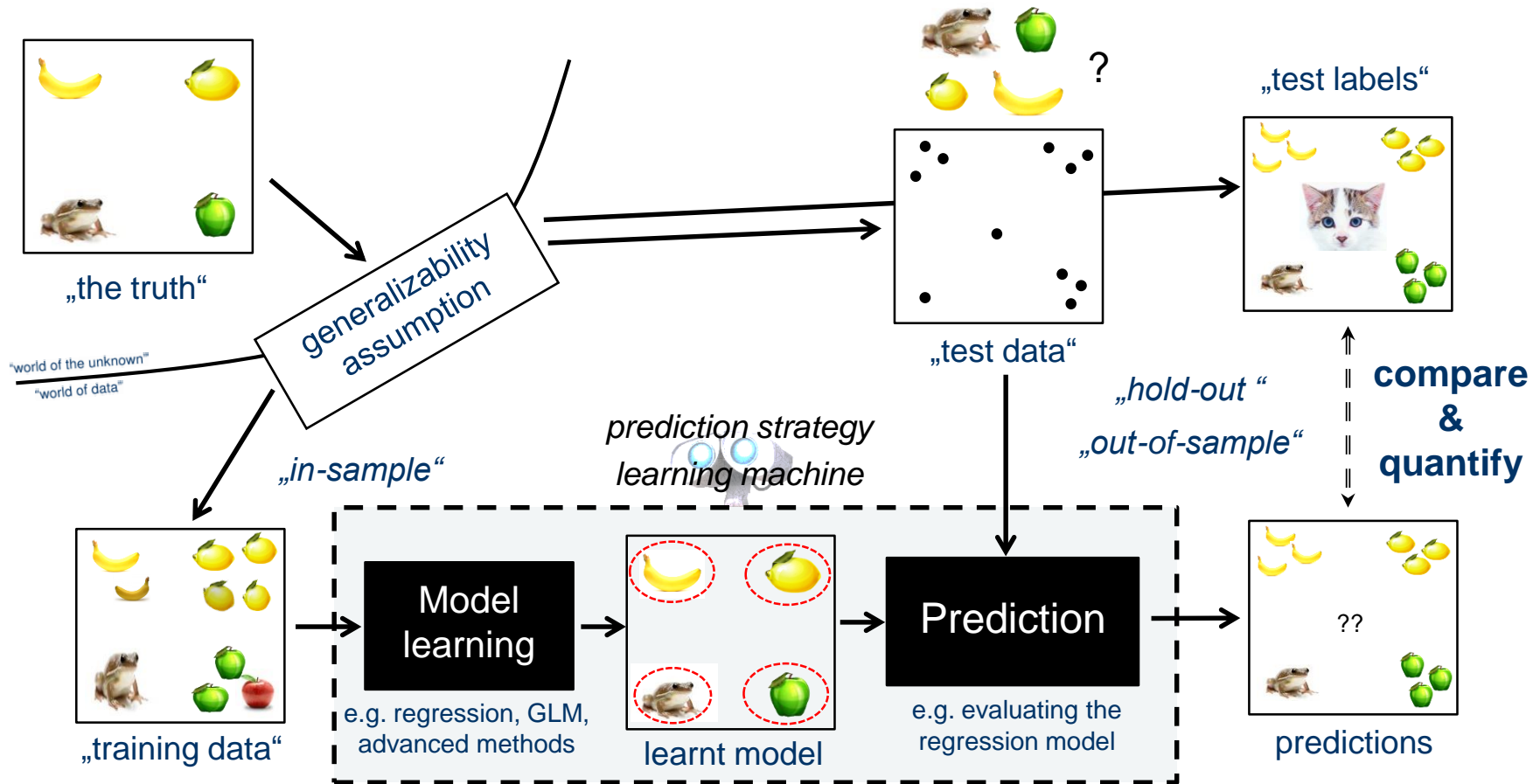
S. Hawking (after K. Popper), in: A Brief History of Time (1988)

= the simple model with lowest (prediction) error is best

Problem: how to predict (with guarantees!) the prediction error?



Model validation: does the model make sense?



Predictive models need to be validated on unseen data!

The only (general) way to test goodness of prediction is actually observing prediction!

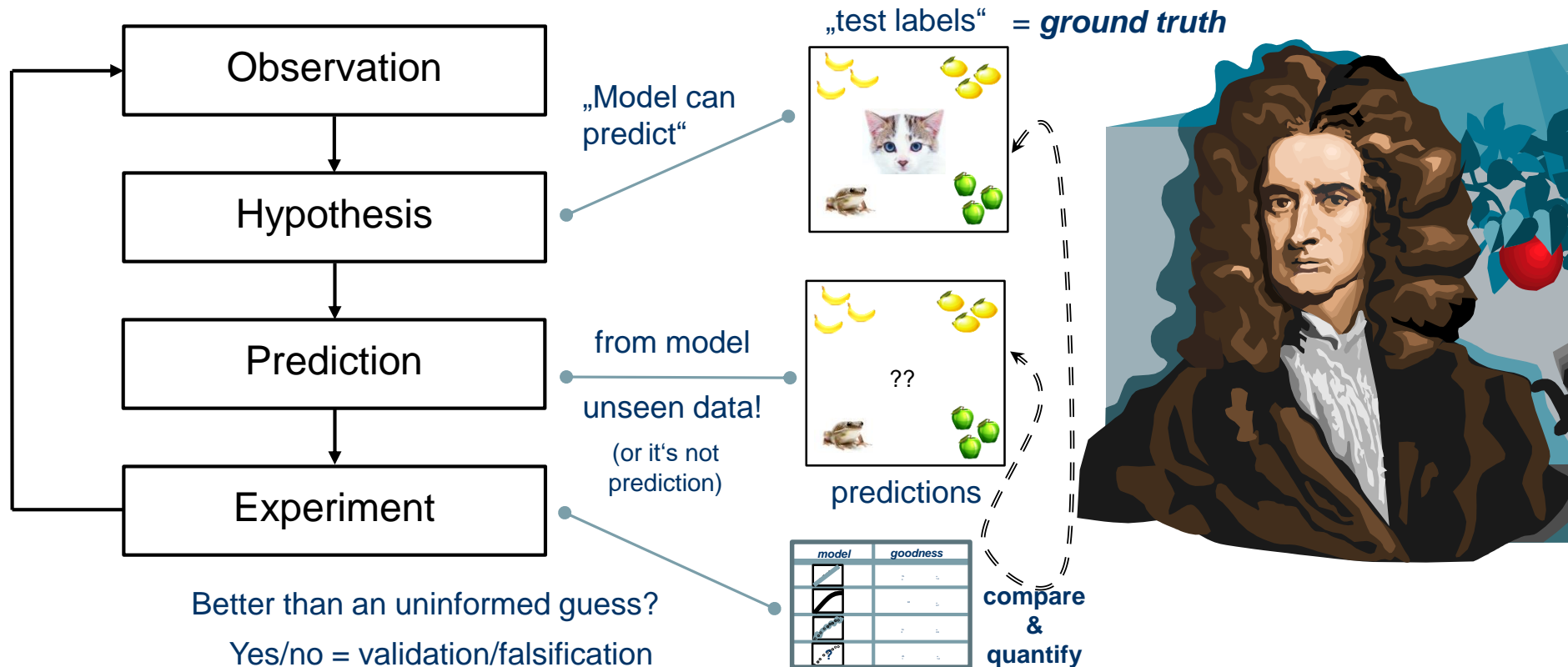
Which means the part of data for testing has **not** been seen by the algorithm before!

(note: this includes the case where machine = linear regression, generalized linear models, etc)

Cherchez la Ground Truth!

Crucial argument to conclude „the model is useful“.

Here: model genuinely predicts [data] which it could not have known about



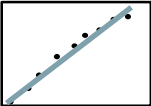
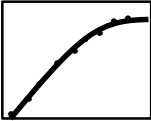
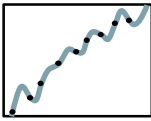

Ground truth is necessary for carrying out any experiment!

If a claim cannot be checked (=experiment can say „no“), it is not scientific!

Said with Wolfgang Pauli: it's not even wrong...

Quantitative model comparison

Concretely, one would like to have a benchmark table such as this:

<i>model</i>	<i>statistic of model error (or model goodness)</i>	
	15.3	± 1.4
	9.5	± 0.7
	13.6	± 0.9
	20.1	± 1.2

Confidence regions (or paired tests) to compare models to each other:

A is better than B / B is better than A / A and B are equally good

Uninformed model (stupid model/random guess) needs to be included

otherwise a statement „is better than an uninformed guess“ cannot be made.

Today: How to (correctly) obtain and (correctly) interpret such a table.

Lecture 2: Statistical rationale for today's choices and formal guarantees

The predictive model validation workflow

1. Define the prediction task which needs to be addressed:

what are the learners supposed to do?

regression (= target continuous)? classification (= target discrete)?

which variables/data to predict from?  

which variables/data are predicted?   

2. Select appropriate measure(s) of goodness/error:

when is the result considered good?

 $\leftarrow = \rightarrow$  e.g. mean squared error out-of-sample

3. Estimate out-of-sample performance for all strategies and baselines:



how well is the model actually solving the task?

residual diagnostics

= „benchmarking experiment“

benchmark table, with error bars

4. Quantitatively compare prediction strategie(s) and baselines:

which models are best?

is the model better than an uninformed guess?

is the model better than much simpler models?

e.g. comparing error bars

or paired sample test
on predictions/residuals

model	goodness
	7.5
	8.2
	8.8
	9.1

Always validate out-of-sample!

Always compare to naive baselines and state-of-the-art!

2. Measuring prediction goodness of regressors

given **test data** $x_1, \dots, x_N \in \mathbb{R}^n$ „**true**“ **test labels** $y_1, \dots, y_N \in \mathbb{R}$ (held out)
predictor $f : \mathbb{R}^n \rightarrow \mathbb{R}$ **model predictions** $f(x_1), \dots, f(x_N) \in \mathbb{R}$

Measures derived from *distribution of prediction residuals* $\rho_i := y_i - f(x_i)$
 (usually) or $\rho_i := 1 - \frac{f(x_i)}{y_i}$ (relative version)

$$\text{MSE} = \frac{1}{N} \sum_{i=1}^N \rho_i^2$$

mean squared error
 = residual variance

$$\text{MAE} = \frac{1}{N} \sum_{i=1}^N |\rho_i|$$

mean absolute error
 = mean absolute residual

$$\text{RMSE} = \sqrt{\text{MSE}}$$

root mean squared error
 = residual standard deviation

RMSE and MAE are equal for Gaussian residuals; RMSE is more sensitive to outliers
 (in expectation)

Error bars/confidence regions can be obtained from re-sampling

how precisely is subject of ongoing research, there are a number of subtle issues
but trust regions should be obtained for any important summary

Less frequently used but sensible measures: R-squared out-of-sample
 Medians of residual samples (MedSE, MedAE) Variance (un)explained

Other measures are uncommon and possibly problematic

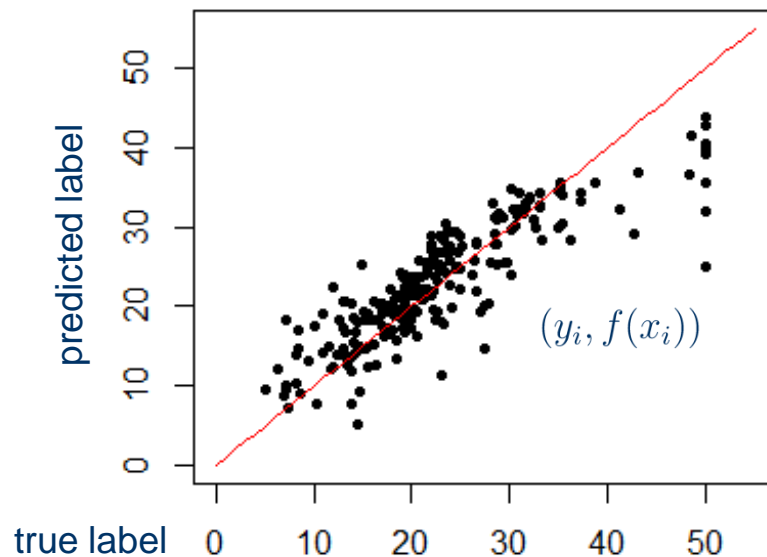
2. Model diagnostics for regressors

given **test data** $x_1, \dots, x_N \in \mathbb{R}^n$ „true“ test labels $y_1, \dots, y_N \in \mathbb{R}$ (held out)

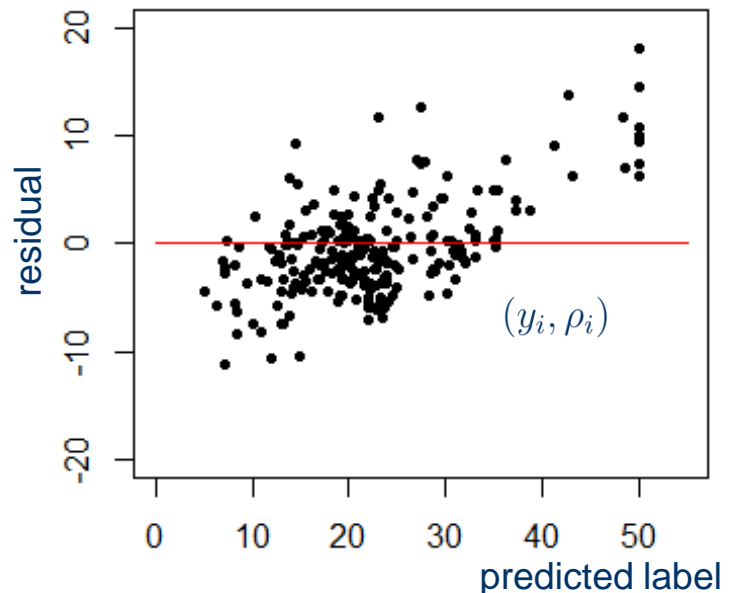
predictor $f : \mathbb{R}^n \rightarrow \mathbb{R}$ **model predictions** $f(x_1), \dots, f(x_N) \in \mathbb{R}$

prediction residuals $\rho_i := y_i - f(x_i)$

„**cross-plot**“: true vs predicted



residual plot: predicted vs residuals



Look for: residual structure, non-linearity, outliers

(analogous to residual diagnostics for linear regression)

for re-sampling (point 3): check per-fold, and aggregated

2. Measuring prediction goodness of classifiers

for two labels $L = \{\text{yes}, \text{no}\}$

classifier $f: \mathbb{R}^n \rightarrow L$
(already fitted)

Test covariates $x_1, \dots, x_M \in \mathbb{R}^n$

Test labels $c_1, \dots, c_M \in L$





Test set can be equal to training set for in-sample fit
should be out-of-sample for predictive validation

1. Make predictions for the test set:

Predicted labels $y_1 = f(x_1), \dots, y_M = f(x_M)$

2. Compute contingency table of test labels vs predicted labels

<i>entries are total counts</i>		<i>true labels</i>	
		yes „positive“	no „negative“
<i>predicted labels</i>	yes	no. of true positives TP	no. of false positives FP „type I error“
	no	no. of false negatives FN „type II error“	no. of true negatives TN

Example:		<i>true</i>	
			
<i>predicted</i>		22	4
		2	14

3. Most measures of predictive goodness are obtained from contingency table

Sensitivity = $TP / (TP + FN)$ fraction of positives correctly predicted

Specifity = $TN / (TN + FP)$ fraction of negatives correctly predicted

Accuracy = $(TP + FN) / \text{all}$ fraction of correct predictions

Precision = $TP / (TP + FP)$

fraction of positives among
those predicted positive

2. Measuring prediction goodness of classifiers

for two labels $L = \{\text{yes}, \text{no}\}$

classifier $f : \mathbb{R}^n \rightarrow L$
(already fitted)

Test covariates $x_1, \dots, x_M \in \mathbb{R}^n$

Test labels $c_1, \dots, c_M \in L$

Test set can be equal to training set for in-sample fit
should be out-of-sample for predictive validation

3. Most measures of predictive goodness are obtained from contingency table

Sensitivity = $TP / (TP + FN)$ fraction of positives correctly predicted

Specificity = $TN / (TN + FP)$ fraction of negatives correctly predicted

Accuracy = $(TP + TN) / \text{all}$ fraction of correct predictions

Precision = $TP / (TP + FP)$

fraction of positives among
those predicted positive

Important: at least *two* measures above should be computed for classifiers

which to compute may depend on the scientific question, common examples:

medical prediction (e.g. screening)

data mining (e.g. information retrieval)

machine learning (e.g. screening)

sensitivity, specificity

precision, recall (=sensitivity)

misclassification rate (= 1 minus accuracy),
precision, or F-beta values

Advanced remarks

Multiple measures from cross-validation can be aggregated via mean

Standard errors can be obtained by bootstrapping (e.g., Jackknife, re-sampling)

ROC curves should be obtained for classifiers with threshold/tuning parameter

2. Model diagnostics for classifiers

Important: most classifier admit trading-off parameters by tuning (e.g. threshold)

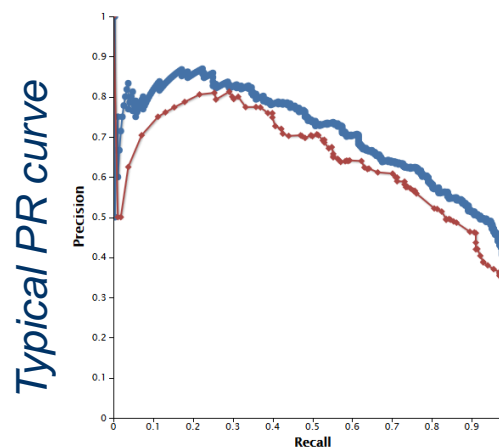
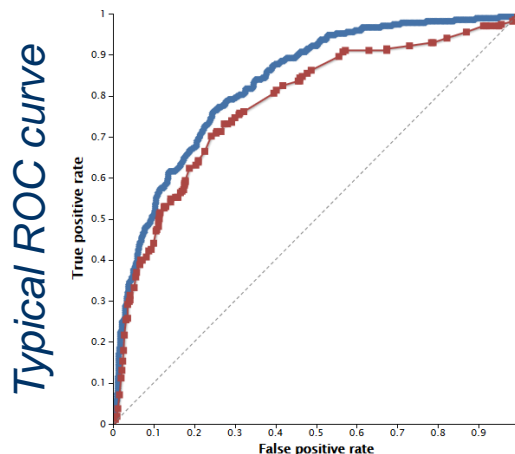
In practice, interesting trade-offs are:

↪ Sensitivity = $TP/(TP + FN)$	fraction of positives correctly predicted	„ROC curve“ (= receiver-operator-characteristic)
↪ Fallout = $FP/(FP + TN)$	fraction of negatives predicted as positive	
↪ Recall = Sensitivity	fraction of positives correctly predicted	„Sens/Spec curve“ = 1 minus ROC
↪ Specifity = $TN/(TN + FP)$	fraction of negatives correctly predicted	

Describe trade-off between type I and type II error, relative to true labels

↪ Recall = Sensitivity	fraction of positives correctly predicted	„Precision-Recall curve“
↪ Precision = $TP/(TP + FP)$	fraction of positives among those predicted positive	

Describes trade-off between type I and type II error by fractions of true positives

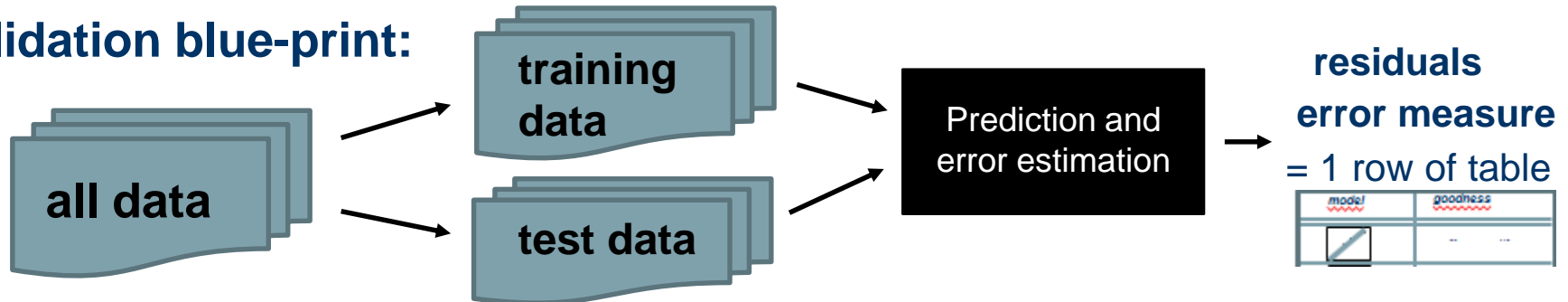


AUC = „area under ROC curve“ statistic of prediction goodness

Confidence bands for AUC and curves can be computed as for other error measures

3. Estimating out-of-sample performance

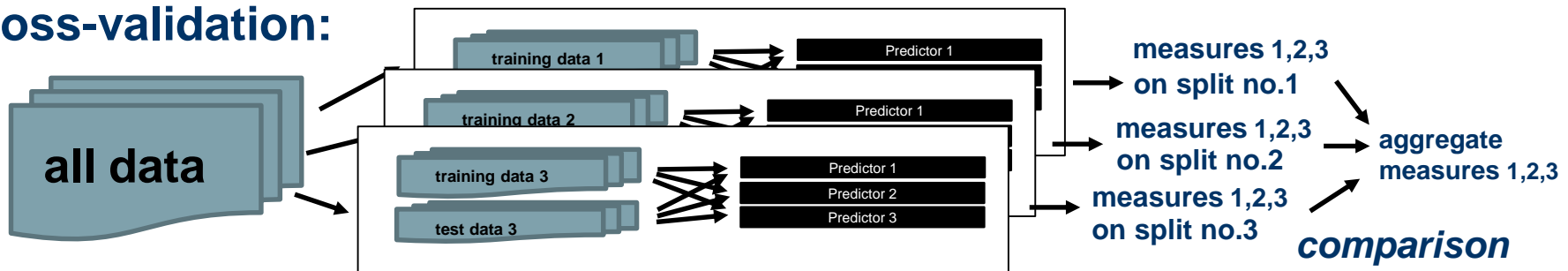
Validation blue-print:



Model comparison:

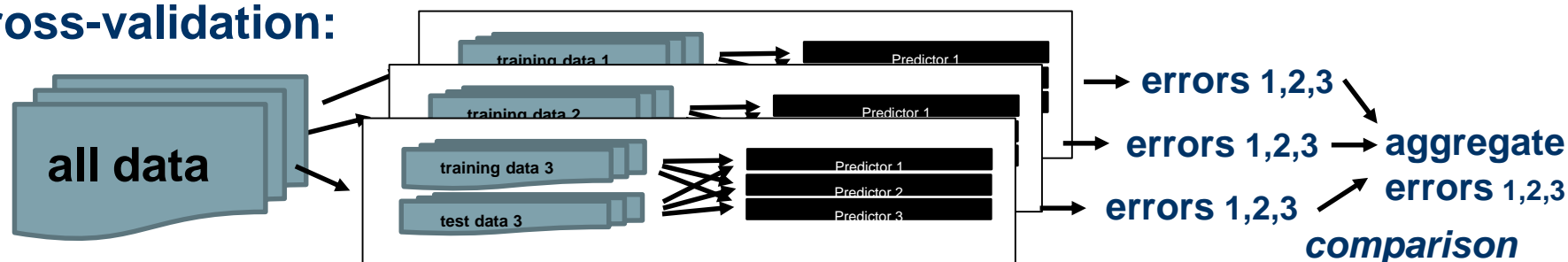


Cross-validation:



re-sampling of multiple train/test splits allows more accurate and stable estimation
all methods need to be evaluated on the same cross-validation splits

Cross-validation:



Aggregation: errors by mean cave: means *inside* MSE are computed per re-sampling
error variances by mean (conservative) or sample-corrected (may underestimate)

<i>type of cross-v.</i>	<i>how to obtain training/test splits</i>	<i>pros/cons</i>
k-fold often: $k=5$	<ol style="list-style-type: none"> 1. divide data in k (almost) equal parts 2. obtain k train/test splits via: each part is test data exactly once the rest of data is the training set 	good compromise between runtime and accuracy when k is small compared to data size
leave-one-out	= [number of data points]-fold	very accurate, high run-time
repeated sub-sampling parameters: training/test size # of repetitions	<ol style="list-style-type: none"> 1. obtain a random sub-sample of training/test data of specified sizes (train/test need not cover all data) 2. repeat 1. desired number of times 	can be arbitrarily quick can be arbitrarily inaccurate (depending on parameter choice) can be combined with k-fold

4. Quantitative comparison of models

Caveat: despite 25+ years of machine learning, much is still open

Problem 1: accurately estimating confidence region for error statistic
is an „*estimate for the error of an error estimate*“
similarly, comparisons are comparisons of errors

Problem 2: cross-validation re-samples are correlated
usually leads to type I error, „spotting differences where there are none“
in the applications often more severe than the type II error
„not spotting a difference between methods which is there“
since falsely concluding usefulness of a method is error of type I
(„there is a significant improvement above the stupid baseline“)

Ad-hoc strategy:

Bootstrap (e.g. Jackknife) statistic on each test fold, aggregation by mean
consistent estimate which avoids type I errors but may incur type II errors

more details on properties and issues of the cross-validation estimate in lecture 2

Important types of comparisons and implications

(with the usual caveats around statistical modelling and significance)

„strategy X predicts better than strategy Y“ ... implies what it says

„strategy X predicts better than the uninformed baselines“

implies that strategy X does something sensible

„strategy X predicts not better than uninformed baselines“

implies that on the data, strategy X does not seem to help

can have many reasons (e.g. not enough data, wrong model, bad tuning)

can mean (but not necessarily) that strategy X was a bad idea

Even if so, knowing of the issue is the first step towards fixing it...

„strategy X with data D added is better than strategy X without“

implies that data D contains information useful for strategy X

„prediction with data D added is better than prediction without“

implies that (assuming a lot of strategies have been explored)

Data D contains useful information for the prediction task addressed

Remark: Knowing the above, the „useful“ models can be further studied

Possibly revealing features of the data that were implicitly captured

Approach 1: „Opening“ black-box models – a widely open area of research

Approach 2: finding interpretable „white box“ model that is best in prediction

Popular ways to hide a useless method

1. Don't benchmark. At all.

After all, if there's no experiment, it can't prove you wrong. Big Data. BIG DATA.

2. Compute *only* prediction goodness for your method.

Perhaps people won't find out that, without comparison, it's the same as 1.

3. „Forget“ to compare to state-of-art baselines.

A cancer treatment is good enough if better than bloodletting and exorcism.

4. If worse than a random/uninformed guess, compare only to a method which is even worse.

Homeopathy is great because it's better than being hit on the head with a stone.

5. Have small size of test sample, no hypothesis test.

No one will find out that the improvement of your new method is completely explained by randomness if you do not quantify whether it is.

Model validation with mlr

Extended tutorial:

<http://mlr-org.github.io/mlr-tutorial/release/html/index.html>

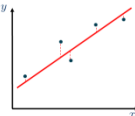
Benchmarking and validation with mlr

`library(mlr)` loads the **mlr** package.

(scikit-learn has similar structure)

modelling strategies are encoded in **Learner** objects

Learner object

Ordinary Least Squares Regression		
<p>Learning task: Regression Predict target $y \in \mathbb{R}$ from features $x \in \mathbb{R}^n$</p>		
<p>Main ideas: prediction function $f: \mathbb{R}^n \rightarrow \mathbb{R}$ parametric, linear: $y \approx f(x) = \hat{\alpha} + (\hat{\beta}, x)$ $\hat{\alpha} \in \mathbb{R}$ fitted to training data by least-squares principle</p>		
<p>Fitting to data data: $x_1, \dots, x_n \in \mathbb{R}^n$ labels: $y_1, \dots, y_n \in \mathbb{R}$ $(\hat{\beta}, \hat{\alpha}) = (X^T X)^{-1} X^T y$ $X \in \mathbb{R}^{n \times n}$ non-singular $E = (I, 1)$ Analytic minimizer of residual sum-of-squares</p>	<p>Predicting from features features: $x_* \in \mathbb{R}^n$ label: $y_* \in \mathbb{R}$ Substitute into fitted prediction function: $\hat{y}_* = f(x_*) = \hat{\alpha} + (\hat{\beta}, x_*)$</p>	<p>Tuning parameters Whether there is an interest: yes/no $\hat{\alpha} = 0?$</p>
<p>Advantages classical, well studied, often used informed baseline, parsimonious works well and quite often, must do</p>	<p>Disadvantages unsuitable for non-linear data does not select features sensitive to outliers</p>	<p>Variants: non-linear & kernel regression the LASSO robust linear regression Ref's: ExSL, chapter 3 and section 3.2 Pearson 1901 Galton 1889</p>

modular structure



regr.lm
train(regr.lm, task)
predict(regr.lm, task)
metadata & model info

```
learners <- list(makeLearner("regr.lm", id = "lm"),
                 makeLearner("regr.randomForest", id = "RF"))
```

Two predictive learners are selected: linear regression and random forest

(= advanced non-linear model)

1. Define the task that the learners should address

```
task <- makeRegrTask(id = "mtcars", data = mtcars,
```

```
task = predict mpg after in mtcars dataset
```

```
target = "mpg")
```

Variables must be factor or numerical

2. Select appropriate measure(s) of goodness/error:

```
measures <- list(rmse, mae, timetrain)
```

RMSE MAE runtime

Measures of prediction goodness are also Measure type objects in their own right

Benchmarking and validation with mlr

3. Estimate error out-of-sample for all models and baselines:

```
valsetup <- makeResampleDesc("CV", iters = 10)
```

initializer object for 10-fold cross-validation

```
bmresults <- benchmark(learners,task,valsetup,measures)
```

computes object containing out-of-sample predictions and benchmark results

now contains all results from the benchmarking experiment

4. Quantitatively compare model(s) and baselines:

bmresults to obtain the table of benchmarking results

	task.id	learner.id	rmse.test.sqrt.of.mean	mae.test.mean	timetrain.test.mean
1	pulse	lm	3.095076	7.296039	0.004
2	pulse	RF	2.970952	6.511462	0.052

predictive model used

goodness-of-model statistics

Results can be obtained from the **bmresults** object

```
getBMRAggrPerformances(bmresults)
```

Aggregate performance table

```
getBMRPerformances(bmresults)
```

Performance per re-sample split

```
getBMRPredictions(bmresults)
```

Out-of-sample predictions

Model validation with scikit-learn

Extended tutorial:

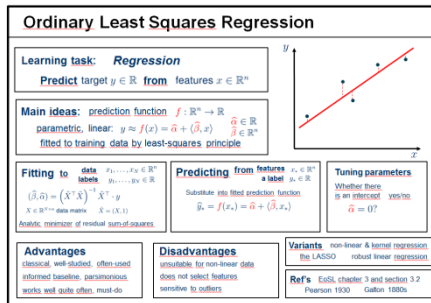
<http://scikit-learn.org/stable/tutorial/basic/tutorial.html>

Benchmarking and validation with sklearn

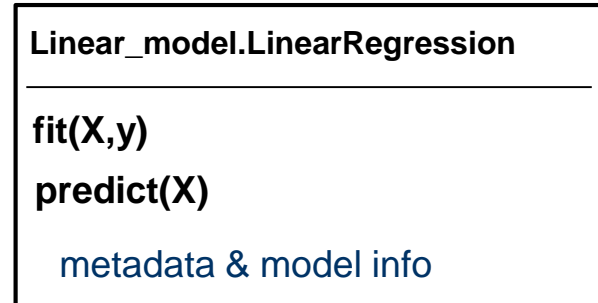
from **sklearn** import etc to load an object/function (mlr has similar structure)

modelling strategies are encoded in **Estimator** objects

Estimator object



modular structure



est_lm = Linear_model.LinearRegression

est_RF = ensemble.RandomForestRegressor

Two predictive learners are selected: linear regression and random forest

(= advanced non-linear model)

1. Split the dataset in a feature array and a target vector

task_data, task_targets = mtfame[:,2:11], mtfame['mpg']

task = predict mpg in mtcars dataset

features must be 2D array, target must be 1D vector

2. Select appropriate measure(s) of goodness/error:

measure_rmse <- metrics.mean_absolute_error

= MAE

Measures of prediction goodness are also **metrics** type objects in their own right

Benchmarking and validation with sklearn

3. Estimate error out-of-sample for all models and baselines:

cvsetup = cross_validation.KFold(n = 32, n_folds = 10)

initializer object for 10-fold cross-validation

or: directly set the **cv** parameter to 10 for k-fold cross-validation

**preds = cross_validation.cross_validation_predict(
est_lm,task_data,task_targets,cvsetup)**

to obtain prediction results; use fixed **cvsetup** to use identical splits

**scores = cross_validation.cross_val_score(
est_lm,task_data,task_targets,measure_rmse,cvsetup)**

instead of scorer object, **measure_rmse** can also be a string

4. Quantitatively compare model(s) and baselines:

scores contains the results for a single method/score combination

loop over methods and metrics, with identical splits, for benchmarking experiment

metrics.classification_report(task_targets,preds)

gives a table of classification metrics if the task was classification

Hands-on workshop: benchmarking experiments

What are there best learners?

Depends...

For *unsupervised* learners, the definition of „good“ is not entirely clear

(more about this later)

For *supervised* learners, good = accurate prediction:

**Any sensible answer to this question would require a
systematic quantitative meta-study**

comparing a larger number of learners
on a larger number of data sets

To my knowledge, there is only one larger benchmarking study which
compares 179 **classifiers** w.r.t. accuracy on 121 data sets (UCI repository)

No such study seems to exist for **regression learners** (as of May 2016)

The plural of anecdotes is not „data“

The plural of analysis is not „meta-study“

The Fernández-Delgado et al study

Do we need hundreds of classifiers to solve real world classification problems?

(actual name of the publication)

JMLR 2014

compares 179 **classifiers** (most in R/caret) on 121 data sets (from UCI repository)

all of „classical table form“ samples/features

Error measures: MMCE, Cohen's kappa (aggregates), Friedman rank

Error estimation: 2-fold cross-validation, class-balanced

(except when data set comes with its own splits)

Model tuning: 4-fold cross-validation, on standard grids

Rank	Acc.	κ	Classifier	Rank	Acc.	κ	Classifier
32.9	82.0	63.5	parRF.t (RF)	45.5	79.5	61.8	adaboost.R (BST)
33.1	82.3	63.6	rf.t (RF)				
36.8	81.8	62.2	svm.C (SVM)				
38.0	81.2	60.1	svmPoly.t (SVM)				
39.4	81.9	62.5	rforest.R (RF)				
39.6	82.0	62.0	elm_kernel_lm (NN)				
40.3	81.4	61.1	svmRadialCost.t (SVM)				
42.5	81.0	60.0	svmRadial.t (SVM)				
42.9	80.6	61.0	C5.0.t (BST)				
44.1	79.4	60.5	avNNet.t (NNET)	57.6	79.1	59.3	adaboost.R (BST)

Problem: tuning and evaluation are done on the same data batch, no nesting!
Hence interpretability of all results is limited, overfitting algorithms (such as neural networks) are possibly optimistically represented

Table 3 from the paper, 20 classifiers with highest Friedman rank

The Fernández-Delgado et al study

Rank	Acc.	κ	Classifier	Rank	Acc.	κ	Classifier
32.9	82.0	63.5	parRF_t (RF)	45.5	79.5	61.0	nnet_t (NNET)
33.1	82.3	63.6	rf_t (RF)	47.0	78.7	59.4	pcaNNet_t (NNET)
36.8	81.8	62.2	svm_C (SVM)	47.1	80.8	53.0	BG_LibSVM_w (BAG)
38.0	81.2	60.1	svmPoly_t (SVM)	47.3	80.3	62.0	mlp_t (NNET)
39.4	81.9	62.5	rforest_R (RF)	47.6	80.6	60.0	RotationForest_w (RF)
39.6	82.0	62.0	elm_kernel_lm (NNET)	50.1	80.9	61.6	RRF_t (RF)
40.3	81.4	61.1	svmRadialCost_t (SVM)	51.6	80.7	61.4	RRFglobal_t (RF)
42.5	81.0	60.0	svmRadial_t (SVM)	52.5	80.6	58.0	MAB_LibSVM_w (BST)
42.9	80.6	61.0	C5.0_t (BST)	52.6	79.9	56.9	LibSVM_w (SVM)
44.1	79.4	60.5	avNNet_t (NNET)	57.6	79.1	59.3	adaboost_R (BST)

Table 3 from the paper, 20 classifiers with highest Friedman rank

The highest ranking classifiers fall in *three* broad classes:

Random forests, boosted/bagged tree methods

Kernel SVM classifiers with polynomial/Gaussian kernel

Shallow neural networks

(roughly in this order)

Caveat when reading the paper:

Certain parts of the manuscript claim that a single C5.0 tree achieves top performance (e.g. Fig.6)

however, that is likely not a single tree but a boosted ensemble of C5.0 trees, compare classifier no.90

or Table 7

Which learners to try? (personal recommendation)

valid for the „standard“ setting of small-to-medium sized data (100.000 or so samples)

no missing values, tabular format

Classification

„Best performers“

Random forest, gradient boosted trees
standard setting ok usually needs to be tuned

Kernel SVC (gauss, poly, linear)
Data normalization, parameter tuning!

Shallow neural networks
Parameter tuning, perhaps bagging

Usually one or several of these will „win“

Only this part is validated in a meta-study!
Unfortunately in a faulty way...

„Uninformed baselines“

Predicting majority
 Predicting one class with fixed probability
(for two-class ROC only)

„informed“ learners will need to be better

State-of-art domain models

Necessary and good scientific practice

„Parsimonious models“

Linear discriminant analysis

Logistic regression and GLM
Logit or binomial link function

K-nearest neighbor, naive Bayes
K tuned

*If one of these wins, they may be preferable
 (parsimony)*

„interesting choices“

Ensembles on the „best“ from above
 Tuned unsupervised pre-processing
 Bagging/boosting wrappers

*Good ideas, may work/lead to improvement
 (or not)*

Which learners to try? (personal recommendation)

valid for the „standard“ setting of small-to-medium sized data (100.000 or so samples)
no missing values, tabular format

Regression

„Best performers“

Random forest, gradient boosted trees
standard setting ok usually needs to be tuned

= Kriging
Kernel r. regression & SVR (gauss, poly, linear)
Data normalization, parameter tuning!

Shallow neural networks
Parameter tuning, perhaps bagging

One or several of these may „win“?

Just based on personal educated guess

Why are there no meta-studies...

„Uninformed baselines“

Predicting the training mean

Predicting the training median

„informed“ learners will need to be better

State-of-art domain models

Necessary and good scientific practice

„Parsimonious models“

Ordinary least squares regression

Local smoothing splines
Gaussian or Laplace kernel

K-nearest neighbor, possibly weighted
K tuned weighting scheme tuned

*If one of these wins, they may be preferable
(parsimony)*

„interesting choices“

Ensembles on the „best“ from above

The LASSO, sparsity-inducing methods

PCA/SVD as pre-processing or PCR

*Good ideas, may work/lead to improvement
(or not)*

