Experimental Techniques in Physics Supported with AI/ML

Automation of decision-making based on short and noisy measurement data sets

Lecture 12, summer 2023/2024

Agenda

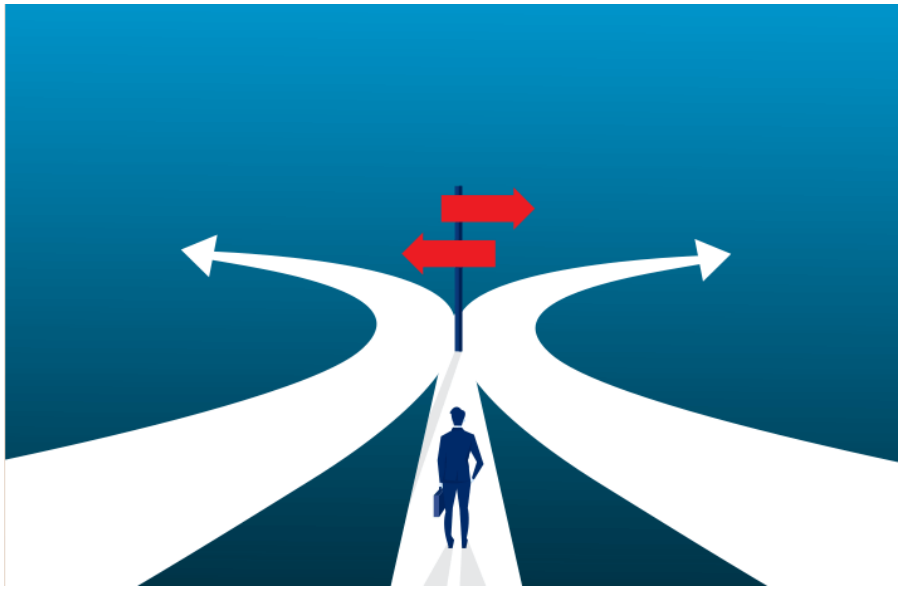
1. Explored systems and data sets
2. Decision making
3. Automation of decision-making based on short and noisy measurement data sets - an example

Explored systems and data sets

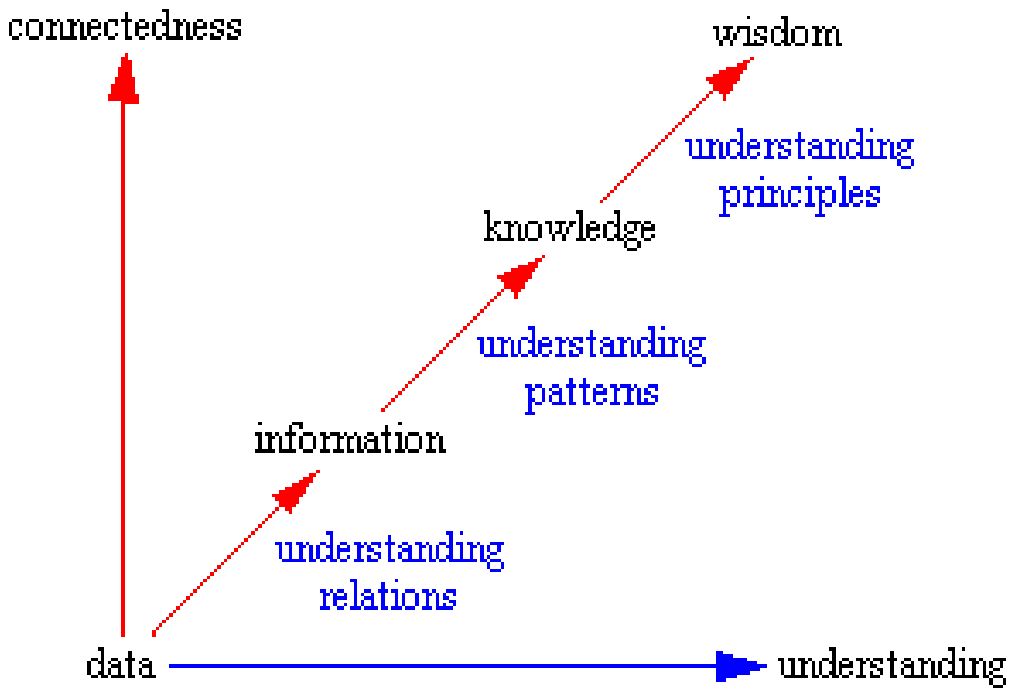
1. Physiological data, e.g. respiratory, ECG, EEG, AM, CM, etc.
2. Data from technical systems, e.g. smart building, etc.
3. Environmental data, e.g., concentration of particulate matter, ozone, CO, etc.
4. Financial data, e.g., stock market indexes, company profit, customer sentiment, etc.

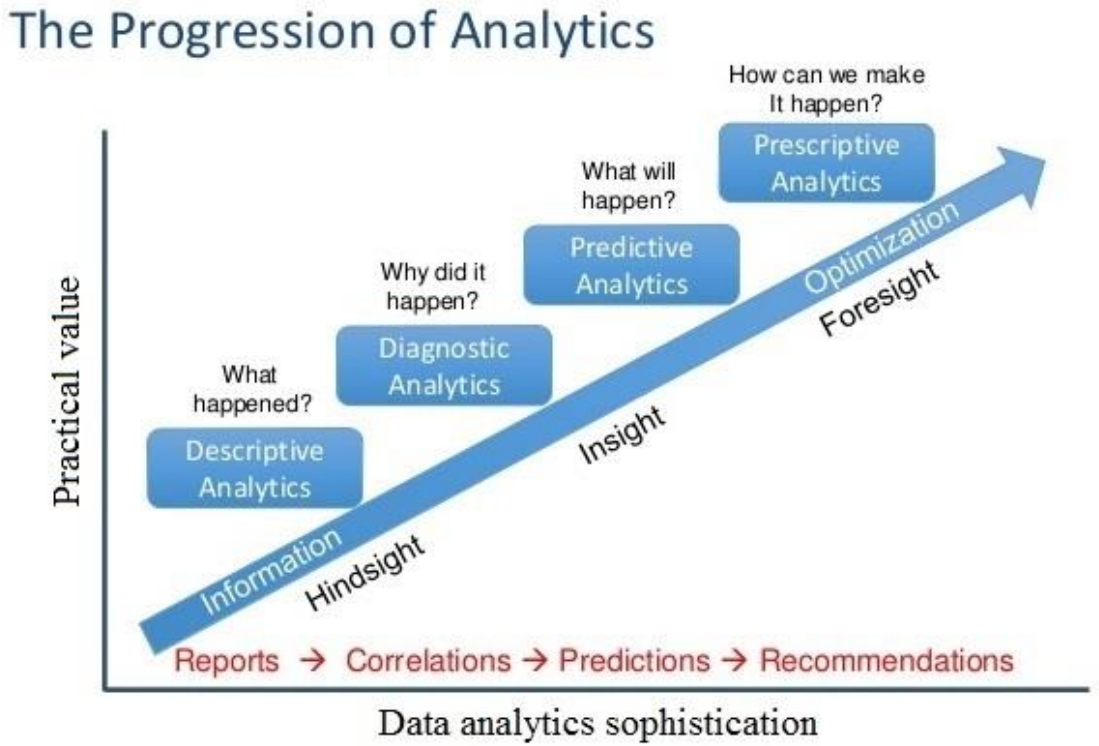
Goal: To obtain as much information about the system (its structure and/or functions) as possible from a limited amount of data that may be distorted

Decision making

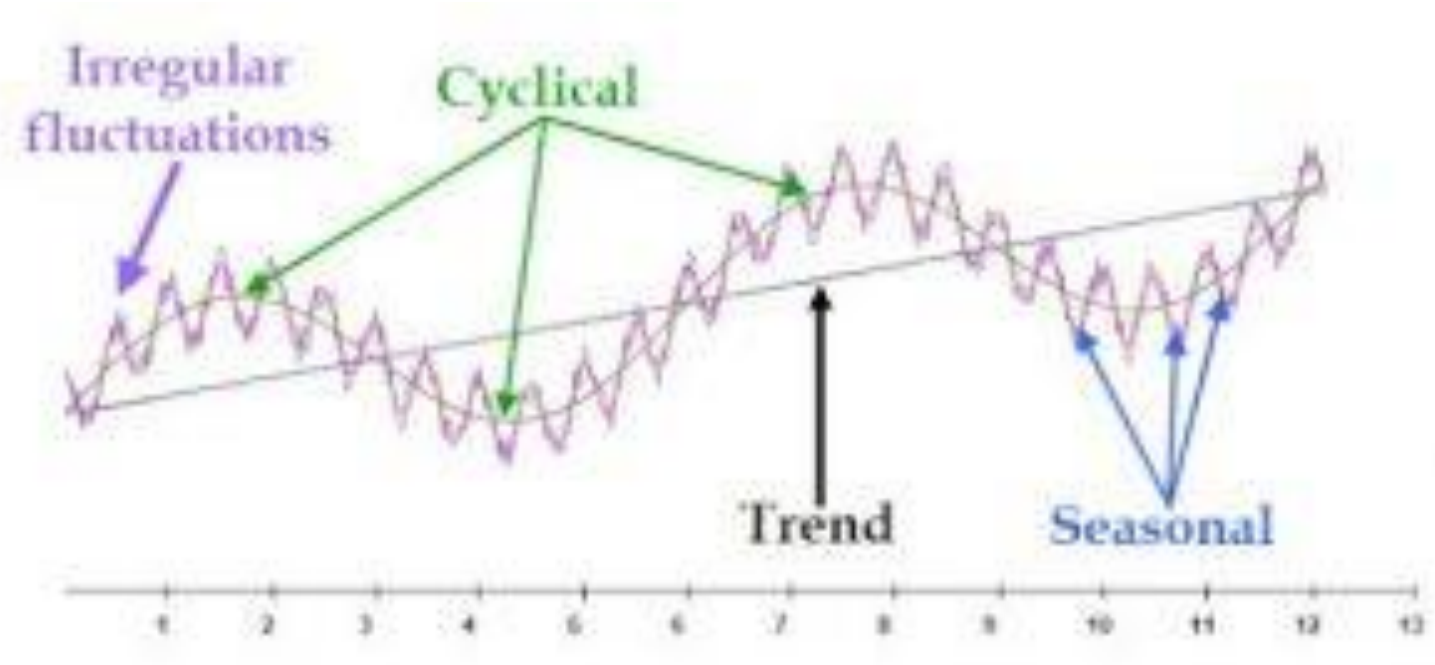


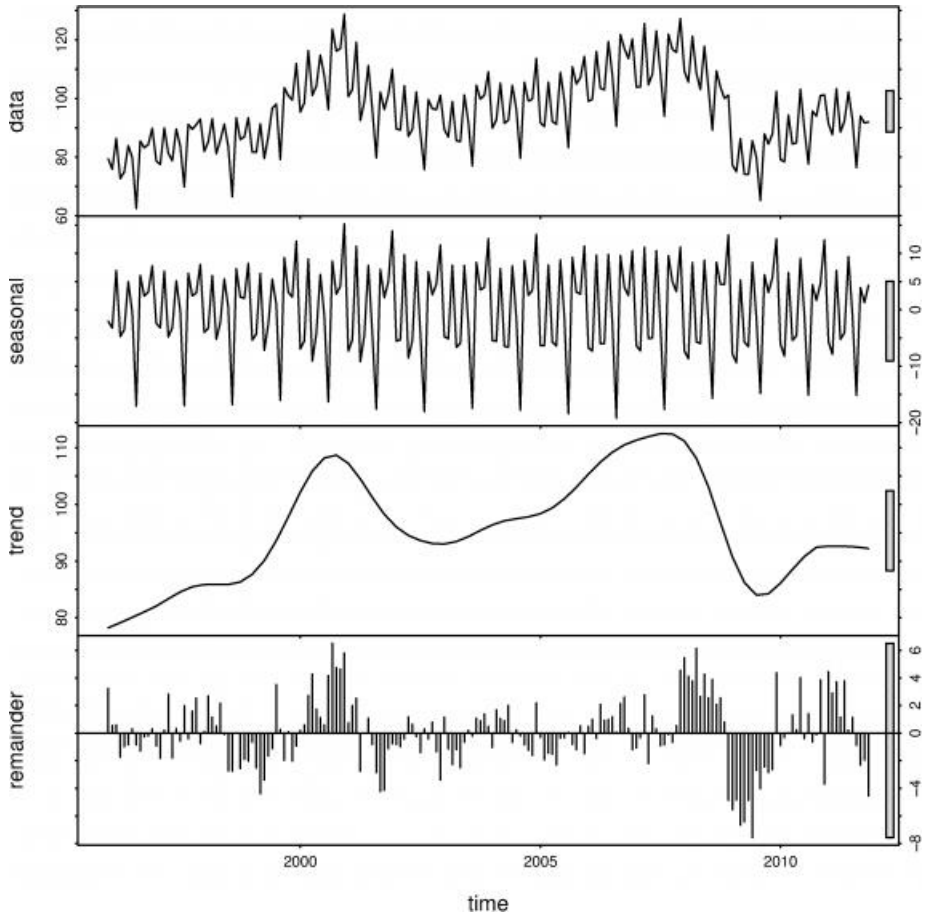
Measurement data, measurement data analysis, automation…



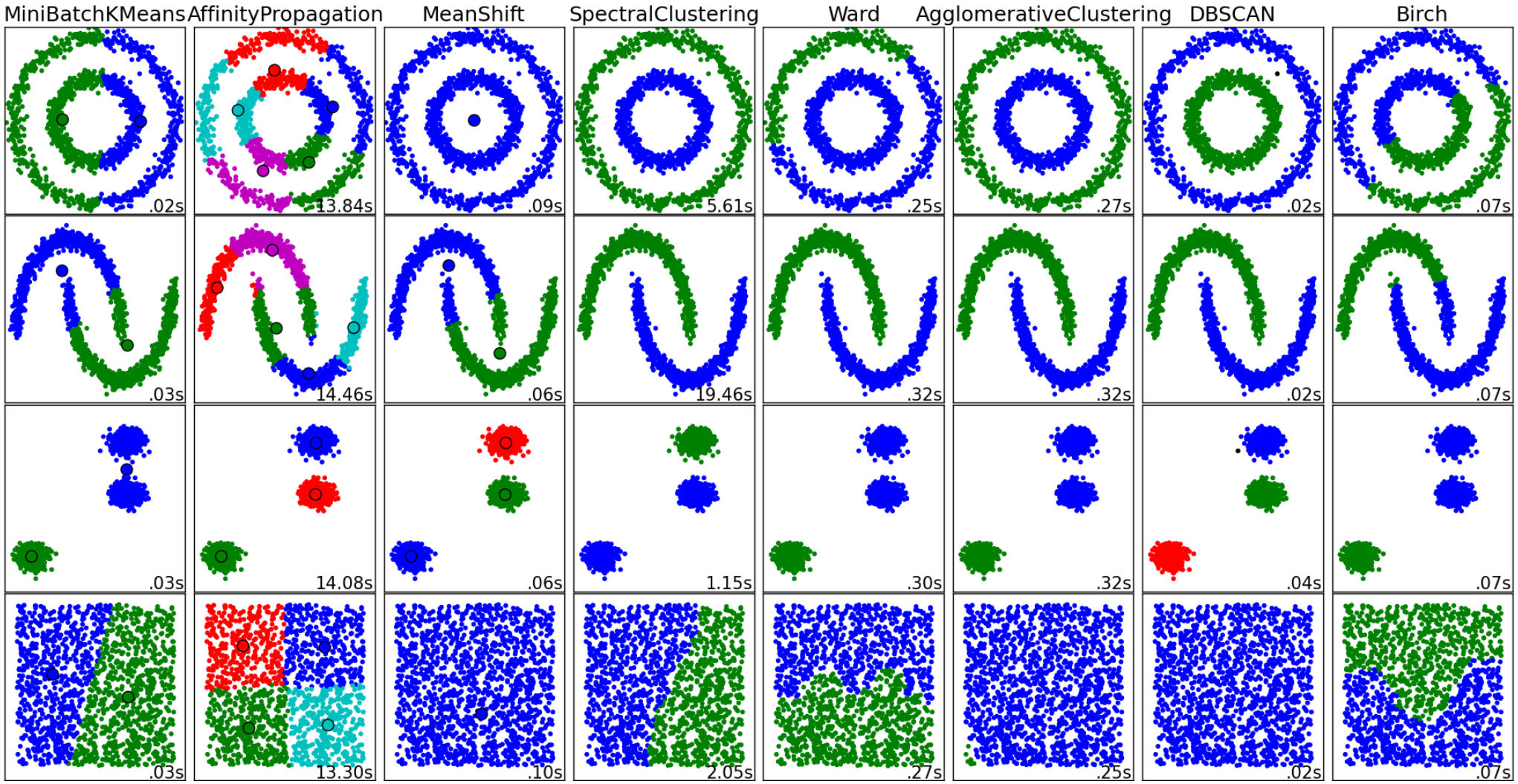


Measurement data in the form of a time series

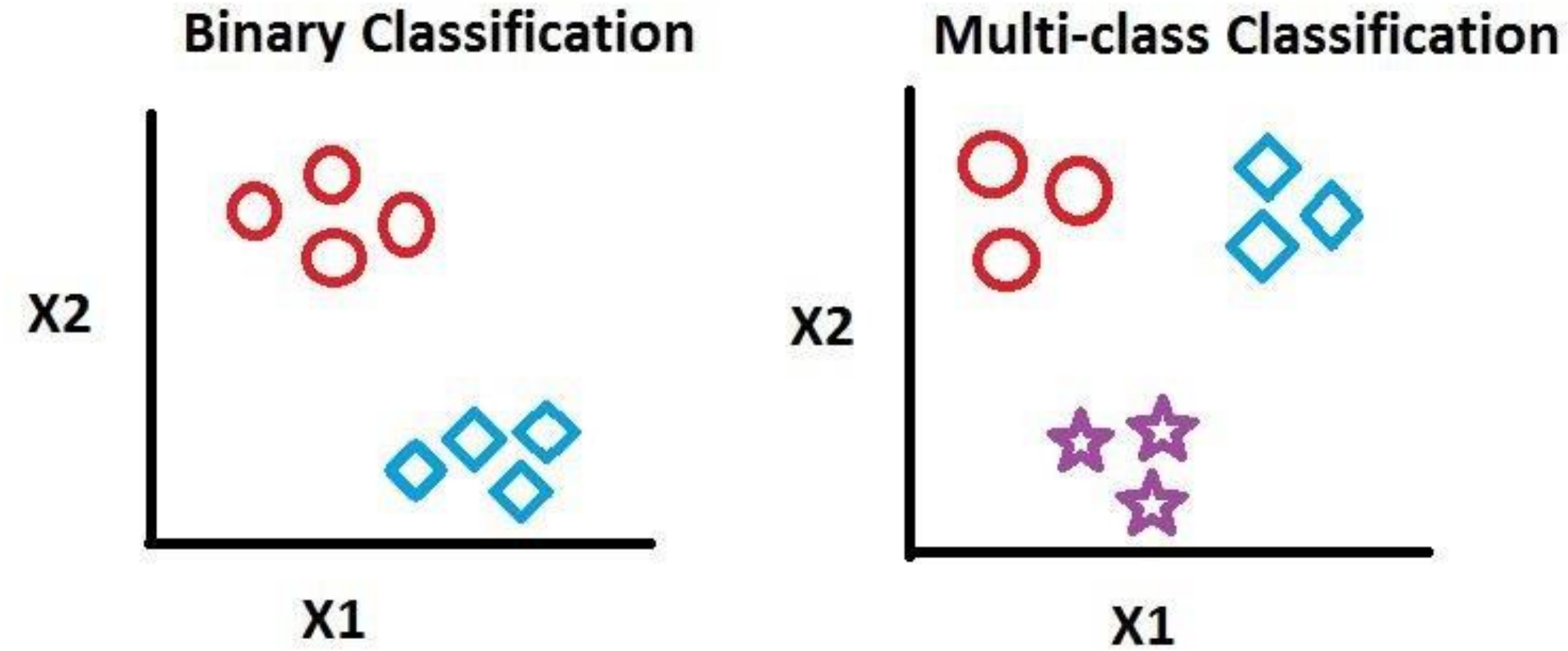




Comparison of clustering methods



Binary classification and multiclassification



Example: automation of decision-making

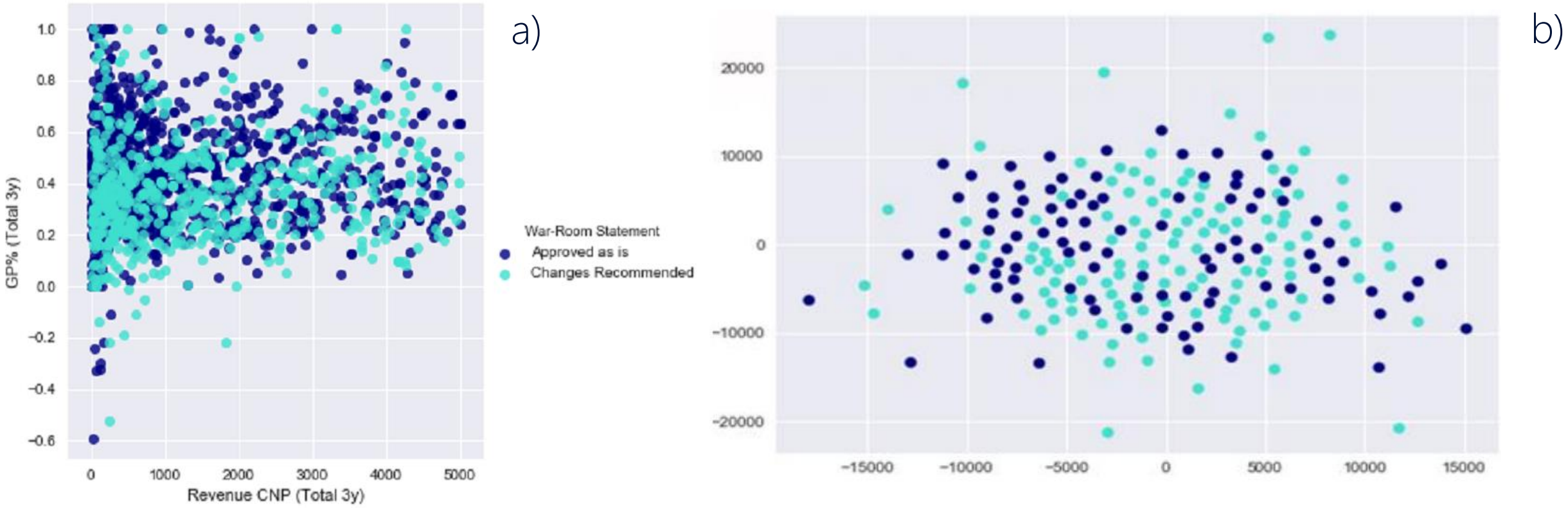


Fig. a) Scatterplot of decisions-"approved"/"changes recommended"-observed for values of two parameters (deal's revenue and sales margin percentage SM%) recorded over a 3-year period; b) 2-D projection of decisions conditioned on the complete set of inputs used in the experiment.

Data: various sources of problems

Human errors

- The data came from manually filled Excel files, hence containing a large number of errors (e.g. OP% was sometimes filled as "ABCDE", sometimes as 0-100%)
- Not all contracts were included in the database used
- There are cases of contracts that, according to the process, should not be sent to the so-called Extended War Room (e.g., contracts that are too large)

Data specifics

- Small data set
- Data is unbalanced - only 20% of contracts are sent for manual verification
- This 20% is the most relevant - it is safer to send good contracts for manual verification than to accept bad contracts
- The most relevant parameters are: Revenue, SM%, OP%

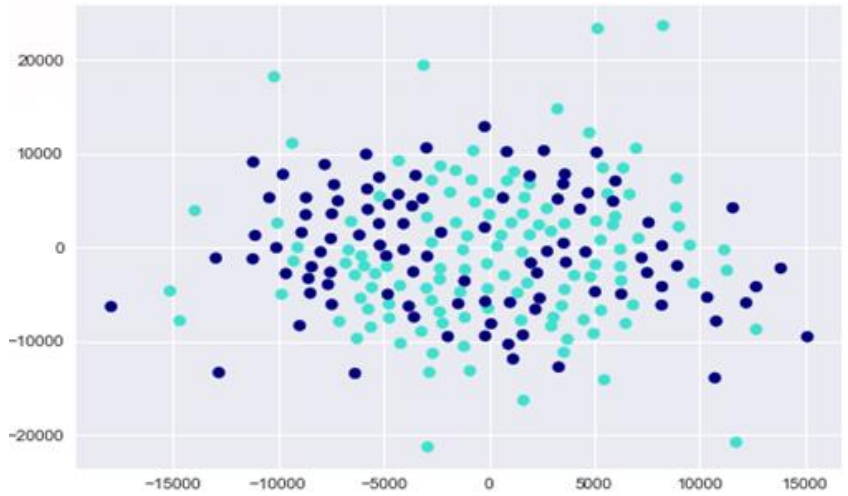
Process specifics

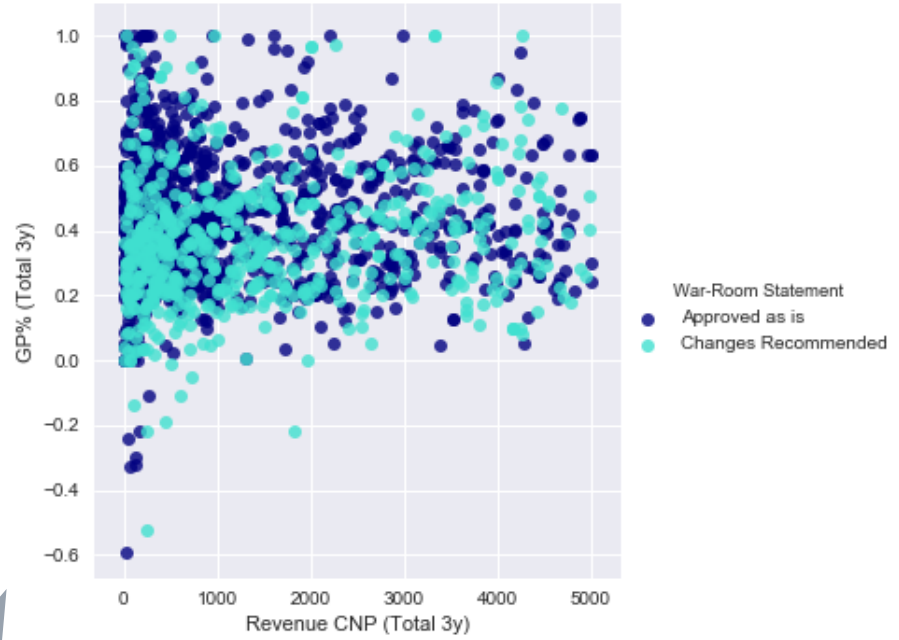
- LoA levels (determine the complexity of the contract) assigned to contracts change over time
- Sometimes contracts with very similar (though not identical) parameter values are sent back
- The goal is to get the highest possible acceptance rate for contracts, while maintaining good decision quality

Choice of data mining method

A large number of cases were not subject to any rule

- Experts were able to provide only a few "hard" rules
- They made decisions based on some unformalized knowledge and intuition
- Preliminary as well as after statistical analysis, no rule applicable to the dataset was observed

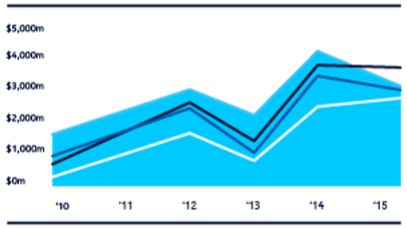




The use of ML methodology will make it possible to extract rules, "invisible" to other methods, applicable to the dataset?

Why machine learning?

A large number of contracts are not subject to any (clear) rules when making decisions

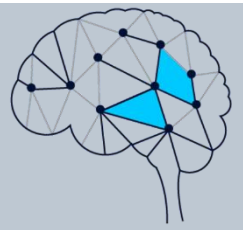


Historical data (parameters due to contracts)

Training set
Approved as is
For manual verification

Decisions made by experts for most contracts

Input values:
SM%,
Revenue,
Market
...
etc.



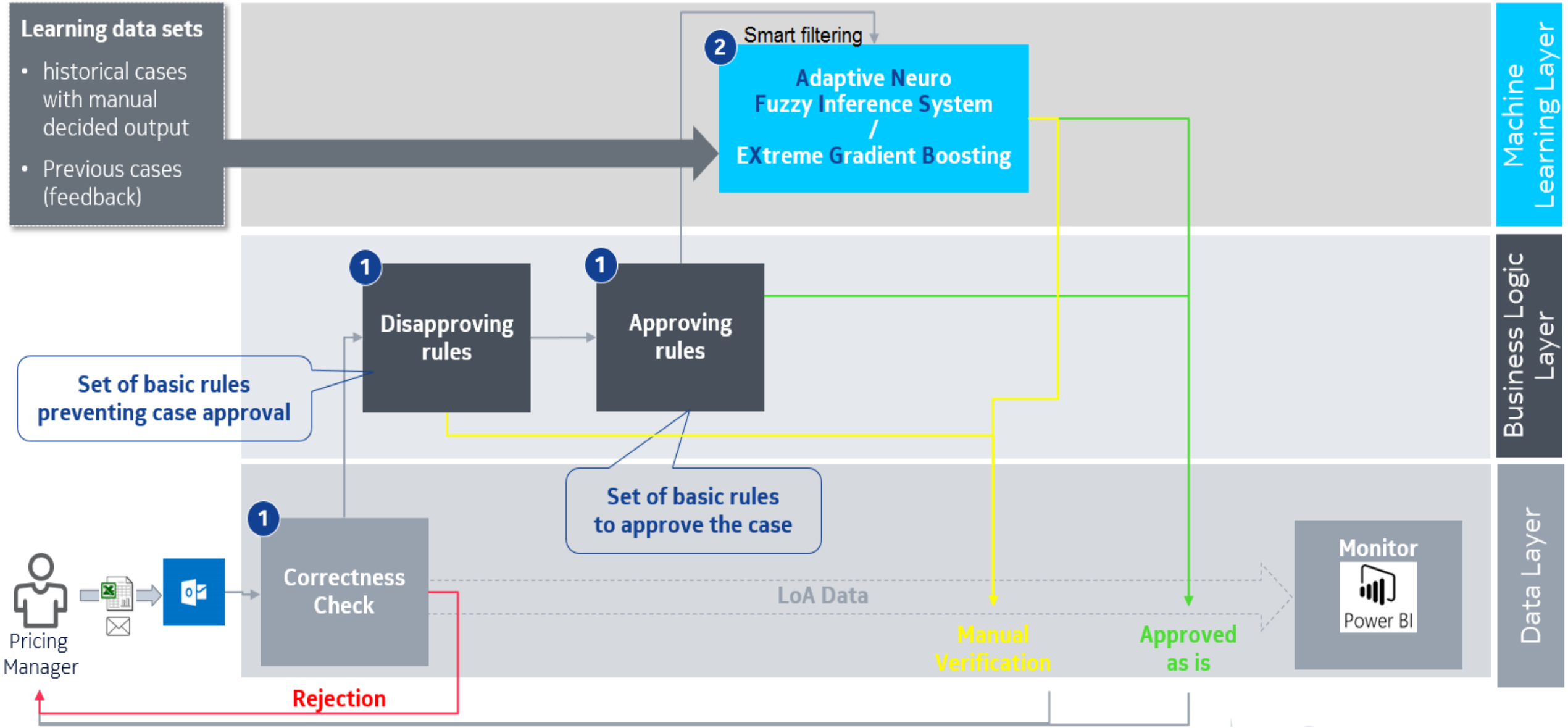
Machine learning algorithm

Prediction

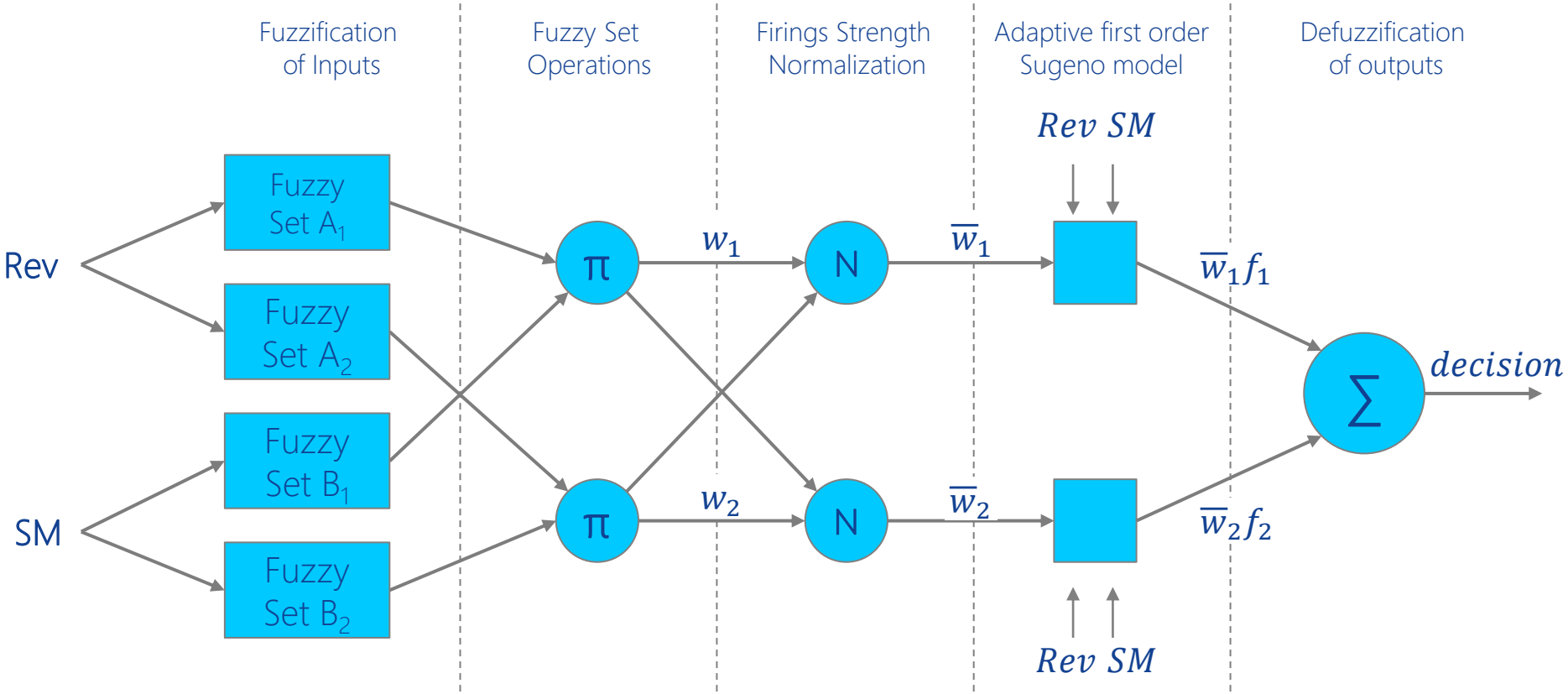
Decision

Supervised learning problem - binary classification

Example of decision-making automation



Smart filter1: Decision Support System - ANFIS (simplified model)



Adaptive Neuro-Fuzzy Inference System (ANFIS)

ANFIS is an intelligent Neuro-Fuzzy technique used for the modeling and control of ill-defined and uncertain systems.
It is an adaptive learning system that performs adaptation based on a learner model and updates it with the newly derived facts.

Step 1 – Learning

Based on historical data, ANFIS works by applying Neural Network learning methods to tune the parameters of a Fuzzy Inference System (FIS)

Step 2 – Fuzzy Decision Support System

Based on fuzzified input it supports decisions in accordance with fuzzy rules.

Rule 1: IF Rev is A1 AND SM is B1;THEN

$$f_1 = p_1 Rev + q_1 SM + r_1$$

Rule 2: IF Rev is A2 AND SM is B2;THEN

$$f_2 = p_2 Rev + q_2 SM + r_2$$

"*Can* a **collection of** so-called **weak learners**
be the basis for generating
a **single** so-called **strong learner**?"

„*Can* a **collection of** so-called **weak learners**
be the basis for generating
a **single** so-called **strong learner**?“

Answer: **YES**

Robert Schapire: "The Strength of Weak Learnability”, 1990

Boosting

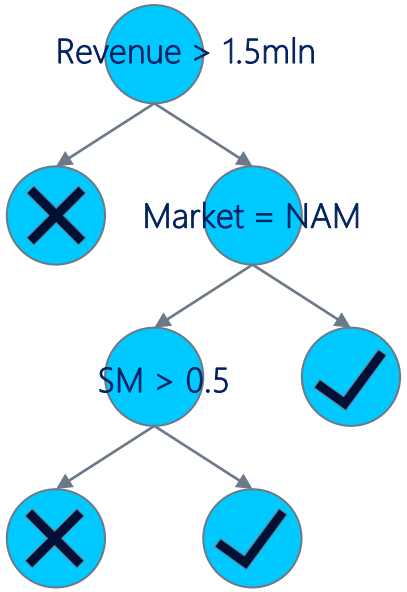
Introduction

Why does it work?	Every poor classifier brings errors	You can use the information about these errors to generate another classifier in following iteration, which is more accurate
How is this implemented?	There are two main approaches:: • Weighting • Gradient Boosting	A decision tree was used to solve the problem as a weak learner

Introduction to XGB

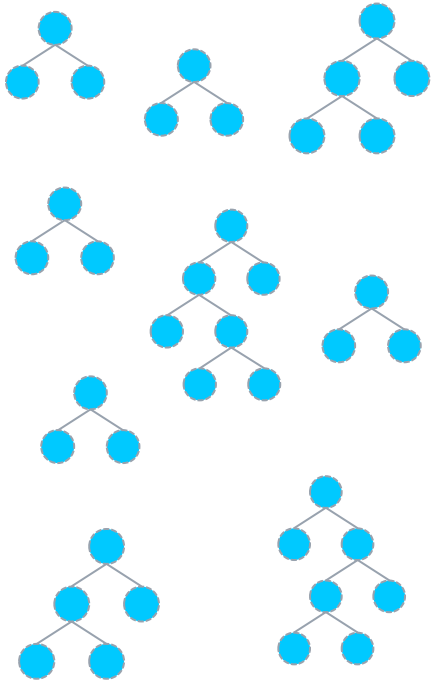
From decision tree to Gradient Boosting

Single **decision tree**



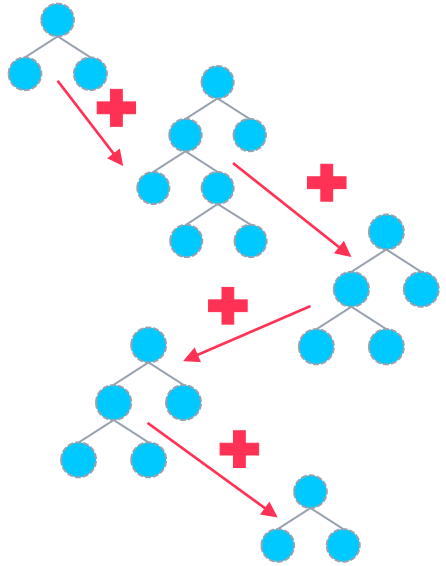
Easy to interpret but prone to overfitting

Random Forest as Ensemble Method



Average outputs of multiple randomly created trees

Additive training – each new tree tries to minimize error of its predecessors



Average outputs weighting them by their contribution to error minimalization

Training XGB

$$F_1(x) = y$$

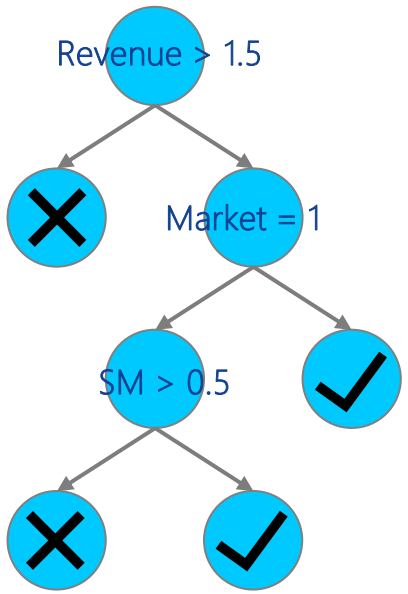
$$h_1(x) = y - F_1(x)$$

$$F_2(x) = F_1(x) + h_1(x)$$

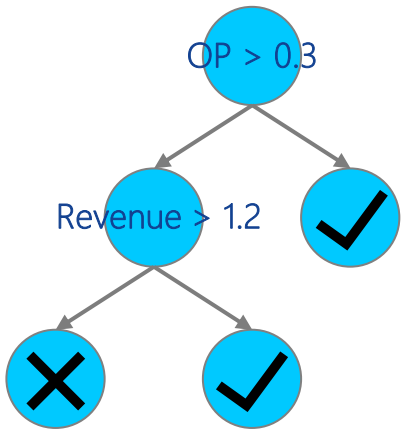
$$F_M(x) = F_{M-1}(x) + h_{M-1}(x)$$

$$F_m(x) = F_{m-1}(x) + \gamma_m h_m(x)$$

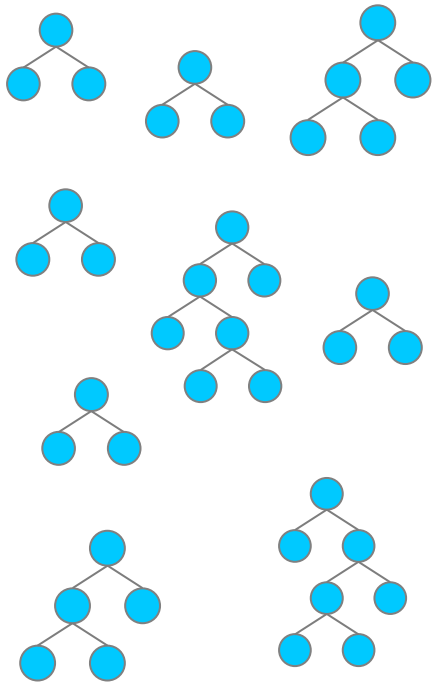
$Tree_1$



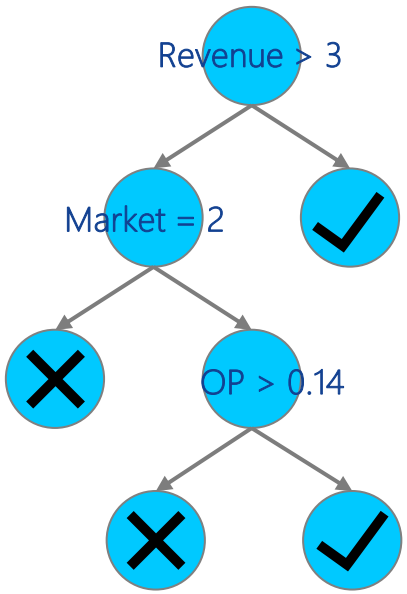
$Tree_2$



$Tree_{...}$



$Tree_{500}$



Extreme Gradient Boosting (XGB):

Step 1 – Learning

- Trained in an iterative way (grows one tree at a time)
- Each tree tries to improve predictions of its predecessors
- A single tree learns greedily
- Regularization techniques are added to the process to avoid overfitting

Step 2 – Predicting

New observations are passed through all trees, and final decision is based on their aggregated votes.

Boosting

Theory

Key formulas

Combined learner at step *m*

Loss function

Weak learner at step *m*

$$F_m(\mathrm{x}) = \mathrm{F}_{m-1}(x) + \underset{h_m \epsilon H}{\operatorname{argmin}} \sum_{i=1}^{n} L(y_i, F_{m-1}(x_i) + h_m(x_i))$$

Family of weak learner functions

Weighting factor

Gradient of loss function w. r. t. combined learner

$$F_m(\mathrm{x}) = \mathrm{F}_{m-1}(x) - \gamma_m \sum_{i=1}^{n} \nabla_{F_{m-1}} L(y_i, F_{m-1}(x_i))$$

$$\gamma_m = \underset{\gamma}{\operatorname{argmin}} \sum_{i=1}^{n} L(y_i, F_{m-1}(x_i) - \gamma \nabla_{F_{m-1}} L(y_i, F_{m-1}(x_i))$$

Boosting

Theory

Pseudocode

1. Initialize first learner with a constant value

$$F_0(\mathrm{x}) = \underset{c}{\operatorname{argmin}} \sum_{i=1}^{n} L(y_i, c)$$

2. For *m = 1* to *M* : (for chosen number of classifiers)
   A. Compute pseudo residuals

$$r_{im} = -\left[\frac{\partial L\big(y_i, F(x_i)\big)}{\partial F(x_i)}\right]_{F(x)=F_{m-1}(x)} \quad \text{for i} = 1, \ldots, \text{n.}$$

   B. Fit a weak learner to pseudo-residuals, classify samples:

$$\{(x_i, r_{im})\}_{i=1}^{n}$$

   C. Compute factor γ_m by solving

$$\gamma_m = \underset{\gamma}{\operatorname{argmin}} \sum_{i=1}^{n} L(y_i, F_{m-1}(x_i) + \gamma h_m(x_i))$$

   D. Update

$$F_m(\mathrm{x}) = F_{m-1}(x) + \gamma_m h_m(x)$$

Algorithm tuning

Hyperparameters optimization

Among algorithm hyperparameters the most important ones are:

Learning rate – how big steps does gradient descent take

Regularization parameter lambda - prevents overfitting

Weighting of positive samples – in case of imbalanced dataset

Minimal child weight – controls overfitting

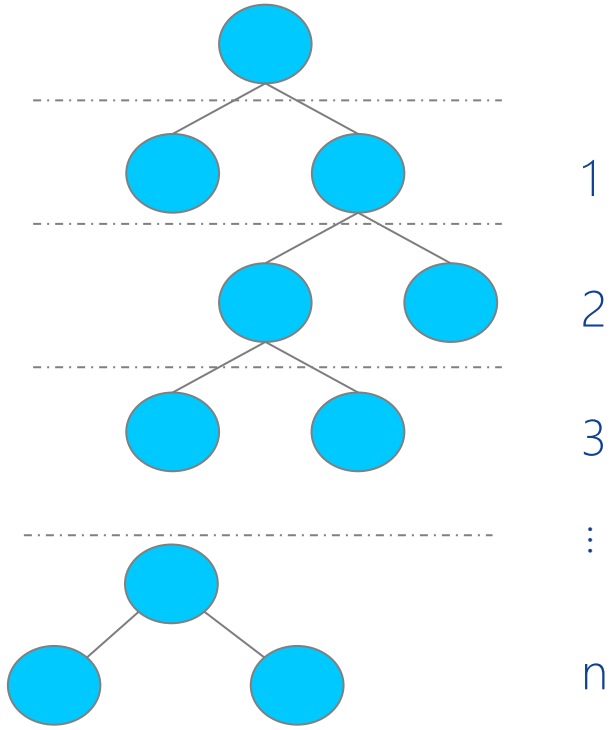
Maximum depth of each tree - bigger the depth, bigger the modeling abilities

Number of estimators (trees) – more trees lead to better error minimization (on training set)

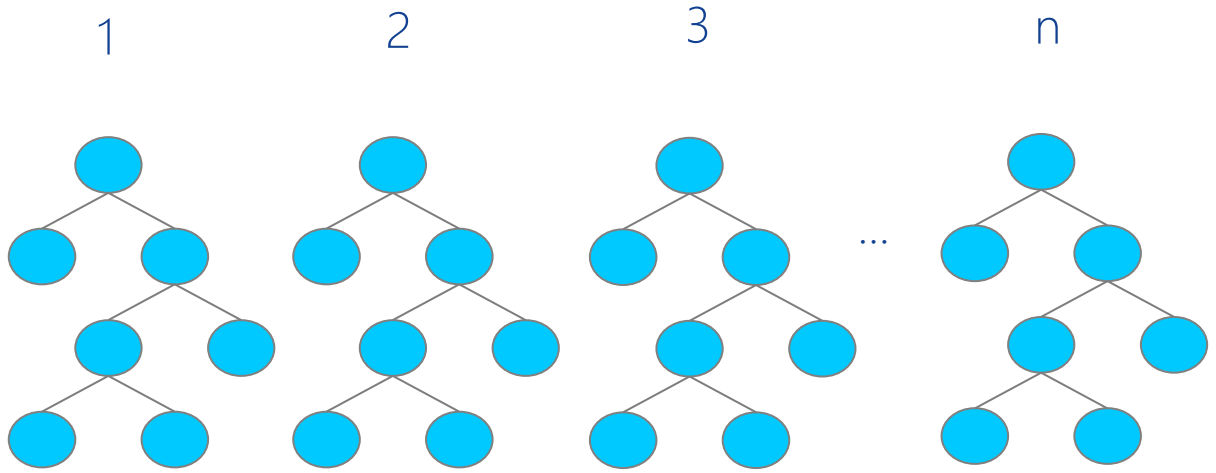
Algorithm tuning

Hyperparameters examples

- **Maximum depth of each tree** - bigger the depth, bigger the accuracy for training dataset, too deep trees may cause overfitting

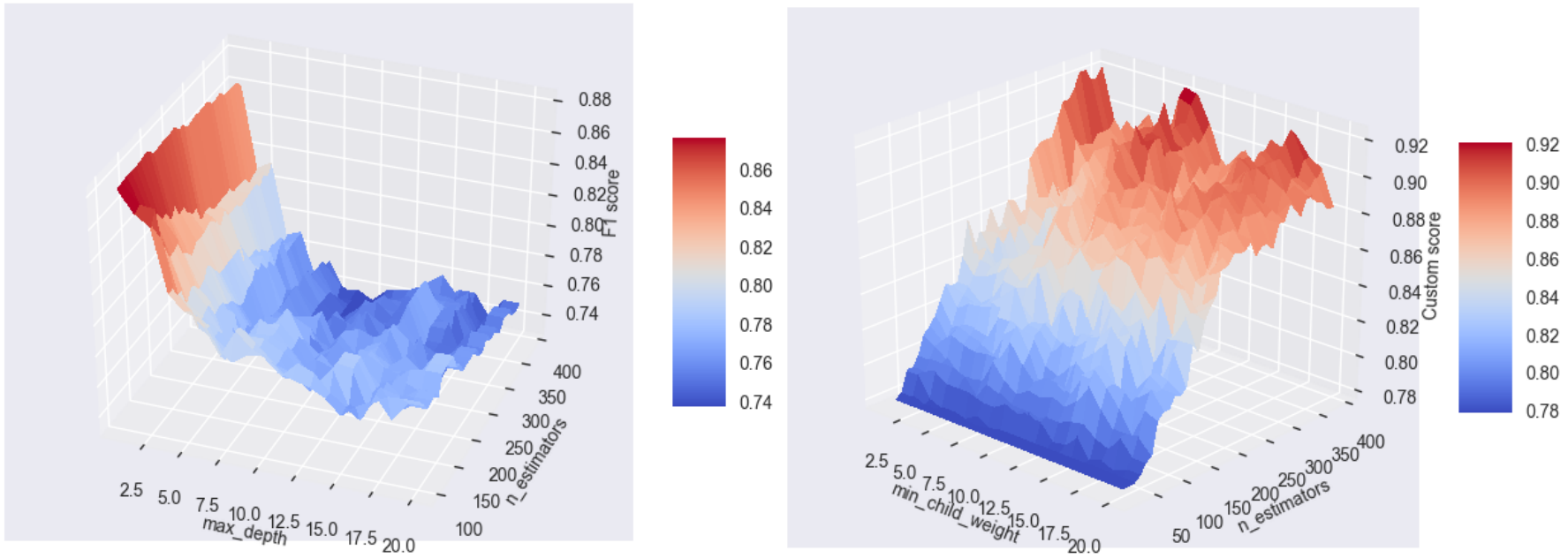


- **Number of estimators (trees)** – more trees, more accuracy on training set, too large number may also lead to overfitting



Efficiency maximization

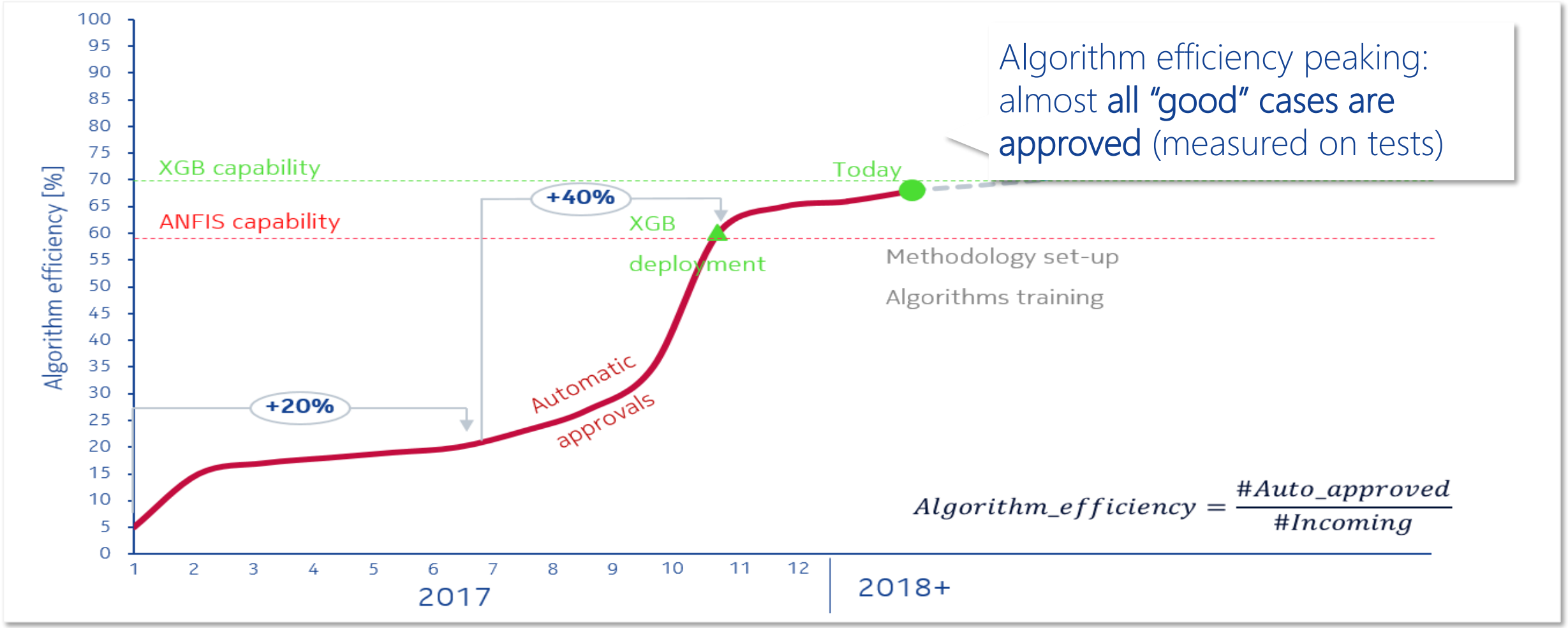
Hyperparameter adjustments



Critical success factors in implementing the ML solution were:

- Tuning algorithm hyperparameters (e.g. tree depth, number of trees)
- Weighting of "changes recommended" samples
- Careful regularization

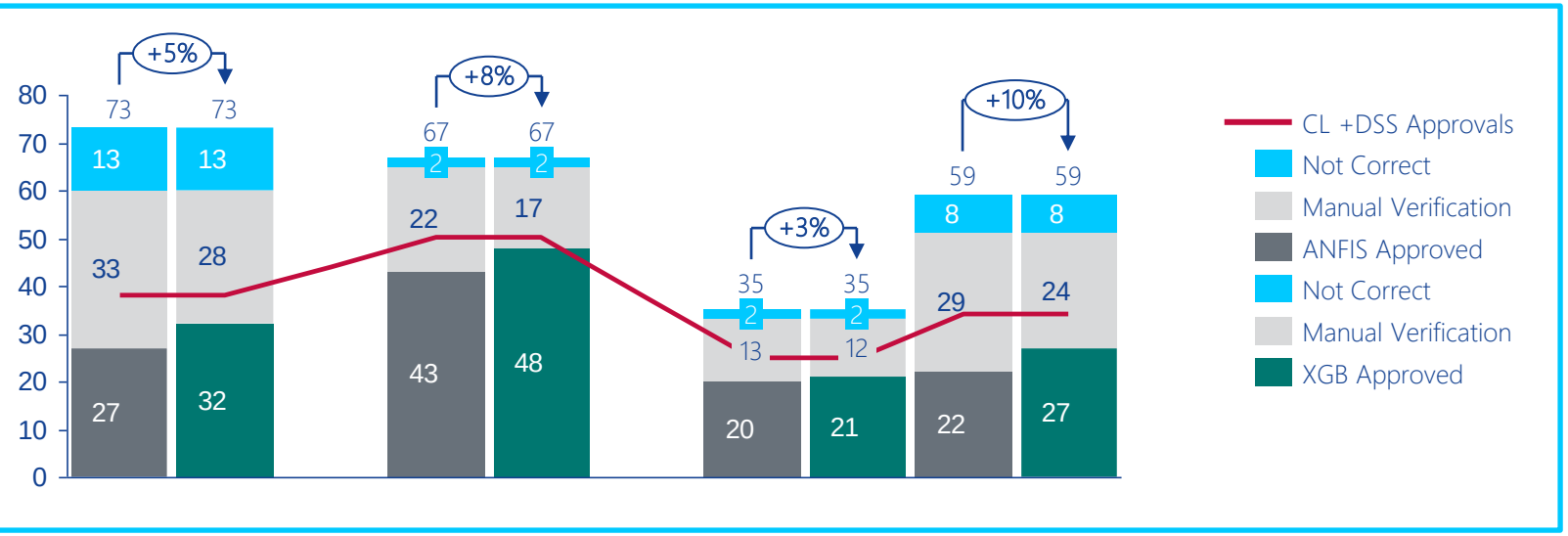
Results – Algorithm efficiency evolution in time



Algorithm efficiency peaking: almost **all "good" cases are approved** (measured on tests)

On initial scope algorithm **reached the maximum amount of validated cases** (~70%)

XGB vs ANFIS: Comparison



	Good approval decision	BAD approval decision
Total No of cases	120	
ANFIS	59	7
XGB	71	2
Value of all cases	k€ 108,570	k€ 108,570
Value of ANFIS-approved	k€ 71,451	k€ 12,519
Value of XBG-approved cases	k€ 75,708	k€ 2,651

XGB has better approval efficiency by

10 p.p.

Even if making mistakes, XGB makes them on cases with much lower value.

Summary

- A decision automation methodology is proposed for the case of a limited number of multi-parameter observations, heterogeneous structure of the decision space, distortion of observations due to proces properties and caused by observer
- Described the relationship between functional values and algorithm accuracy, and the effect of input coding on the accuracy of estimates
- A decision support system using the developed methodology was built, tested and implemented
- Decision time for contract acceptance/rejection reduced from ~2 weeks to ~1 minute
- Objectivity and reproducibility of decisions relative to those made by humans, especially in situations where rapid decision-making is required based on multi-parameter data and with numerous constraints
- Ability to implement automation methodologies for systems of different nature
- The next step: a recommendation system