

Machine Learning for Physics

Introduction

Lisbon School on Machine Learning for Physics 2025, LIP Lisboa, Portugal

Pietro Vischia
pietro.vischia@cern.ch
[@pietrovischia](https://twitter.com/pietrovischia)



If you are reading this as a web page: have fun! If you are reading this as a PDF: please visit

https://www.hep.uniovi.es/vischia/persistent/2025-03-12_LisbonMLSchoolPhysics_Intro.html

to get the version with working animations

Organizational bits

Organization: thanks Michele!!!



- Also thanks to the administrative staff (Natália Antunes, João Pedro Santos)
- Also thanks to the IT staff (Mário David, Nuno Dias, José Aparicio,)
- Not many thanks to the building administration, which makes finding a proper room and all IT things ultradifficult by centralizing many aspects

Organization: thanks Michele!!!



- Also thanks to the administrative staff (Natália Antunes, João Pedro Santos)
- Also thanks to the IT staff (Mário David, Nuno Dias, José Aparicio,)
- Not many thanks to the building administration, which makes finding a proper room and all IT things ultradifficult by centralizing many aspects

Instructors



Pietro Vischia

Ramón y Cajal senior researcher at the Universidad de Oviedo and ICTEA (Spain), Adjunct Professor at IITM. Graduated in 2016 from IST. He is the coordinator of the MODE (Machine-Learning-Optimized Design of Experiments) Collaboration, and the Machine Learning Coordinator of the CMS Experiment at CERN. Specialist in Machine Learning applied to High Energy Physics. Researcher in high-dimensional spaces via gradient descent, eventually powered by quantum algorithms, and on the extension of machine learning methods to realistic neurons with spiking networks, to be then implemented in neuromorphic hardware devices. Within CMS, he focusses on plugging inductive bias in machine learning algorithms for standard model Higgs physics (including the 2018 observation of the ttH process) and beyond-the-standard-model new physics searches in the Top, Higgs, and vector boson sectors. More info at <https://vischia.github.io/>.



Inês Ochoa

Researcher at the Laboratory of Instrumentation and Experimental Particle Physics in Portugal. Graduated in 2015 from University College London. Particle physicist in the ATLAS Collaboration, with a focus on searches for new physics via unsupervised learning techniques and developing new algorithms for measuring Higgs boson properties. Expertise in b-tagging and jet substructure, in online and offline systems. Newly appointed co-coordinator of the HEP Software Foundation Reconstruction & Software Triggers group. More info at <https://inesochoa.github.io/>.



Cristóvão Beirão da Cruz e Silva

Researcher at the Laboratory of Instrumentation and Experimental Particle Physics in Portugal. Graduated in 2016 from IST. Currently a particle physicist in the CMS collaboration. His research interests focus on detector R&D and the development of precision timing detectors, particularly for the PPS2 upgrade for the HL-LHC. He has additional expertise in data analysis using machine learning techniques, having contributed to the search for the Higgs boson decaying to two photons and SUSY searches with LHC data, particularly the search for the supersymmetric partner of the tau lepton and the search for the supersymmetric partner of the top quark in the compressed mass scenario.

Practical information

- For any inquiries, ask Michele or one of the instructors during coffee break, or write to lisbon-ml-workshop@cern.ch
- In line with yesterday's email, we assume you have downloaded the data files for the exercises using the instructions at <https://indico.lip.pt/event/1909/page/232-course-material-please-read>

VERY IMPORTANT

- **Please check if you have wifi connection via eduroam:** if you do not, please approach us at the beginning of the coffee break, so that during the rest of the morning we can have a temporary wifi account created for you, that you will use for the exercises.

class: section

Why Machine Learning in Physics

Nobel Prize in Physics 2040

8 October 2024

The Royal Swedish Academy of Sciences has decided to award the Nobel Prize in Physics 2024 to

John J. Hopfield

Princeton University, NJ, USA

Geoffrey E. Hinton

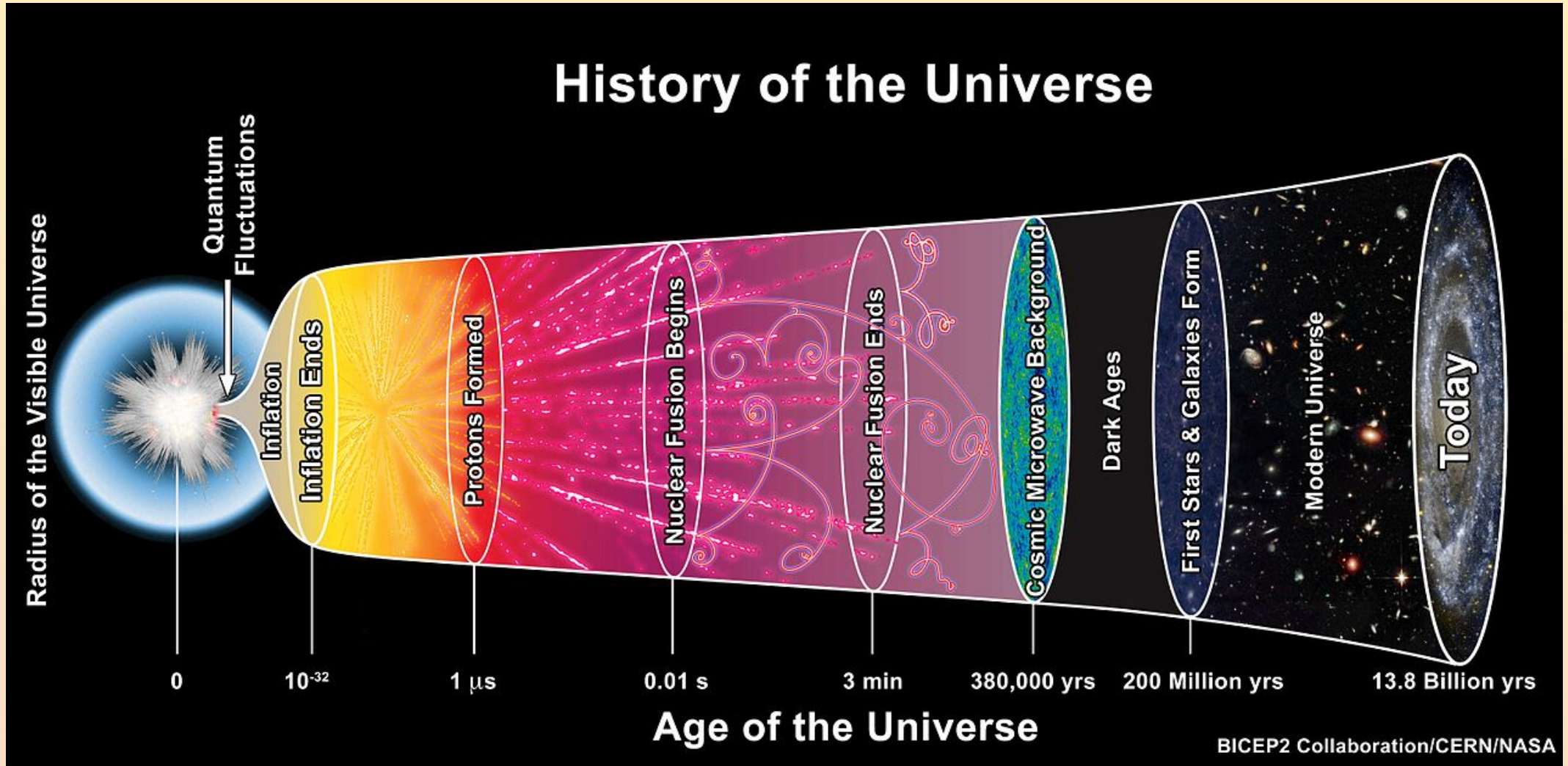
University of Toronto, Canada

“for foundational discoveries and inventions that enable machine learning with artificial neural networks”

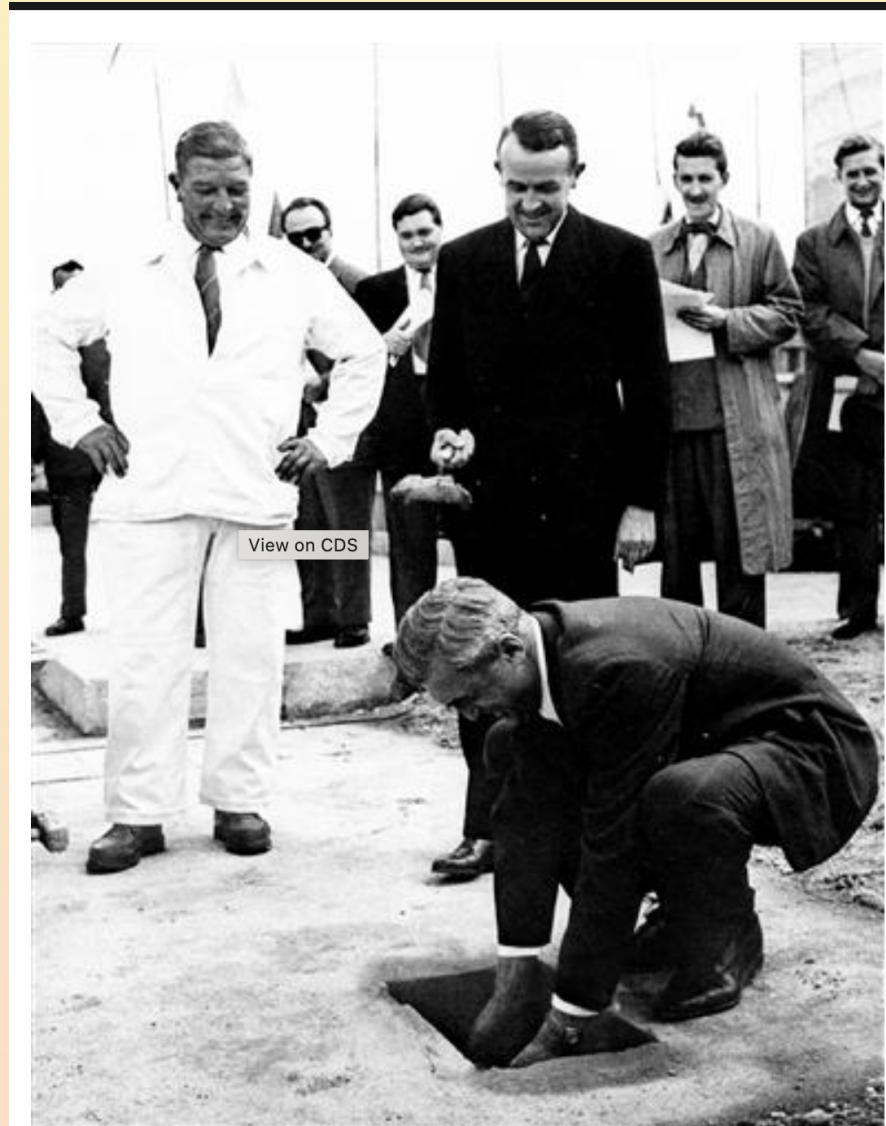
They trained artificial neural networks using physics

This year's two Nobel Laureates in Physics have used tools from physics to

We Try to Understand the Universe



1954: CERN is founded



On 10 June 1955, CERN Director-General, Felix Bloch, laid the foundation stone on the Laboratory site, watched by Max Petitpierre, the President of the Swiss Confederation. (Image: CERN)

1964: CERN's nursery



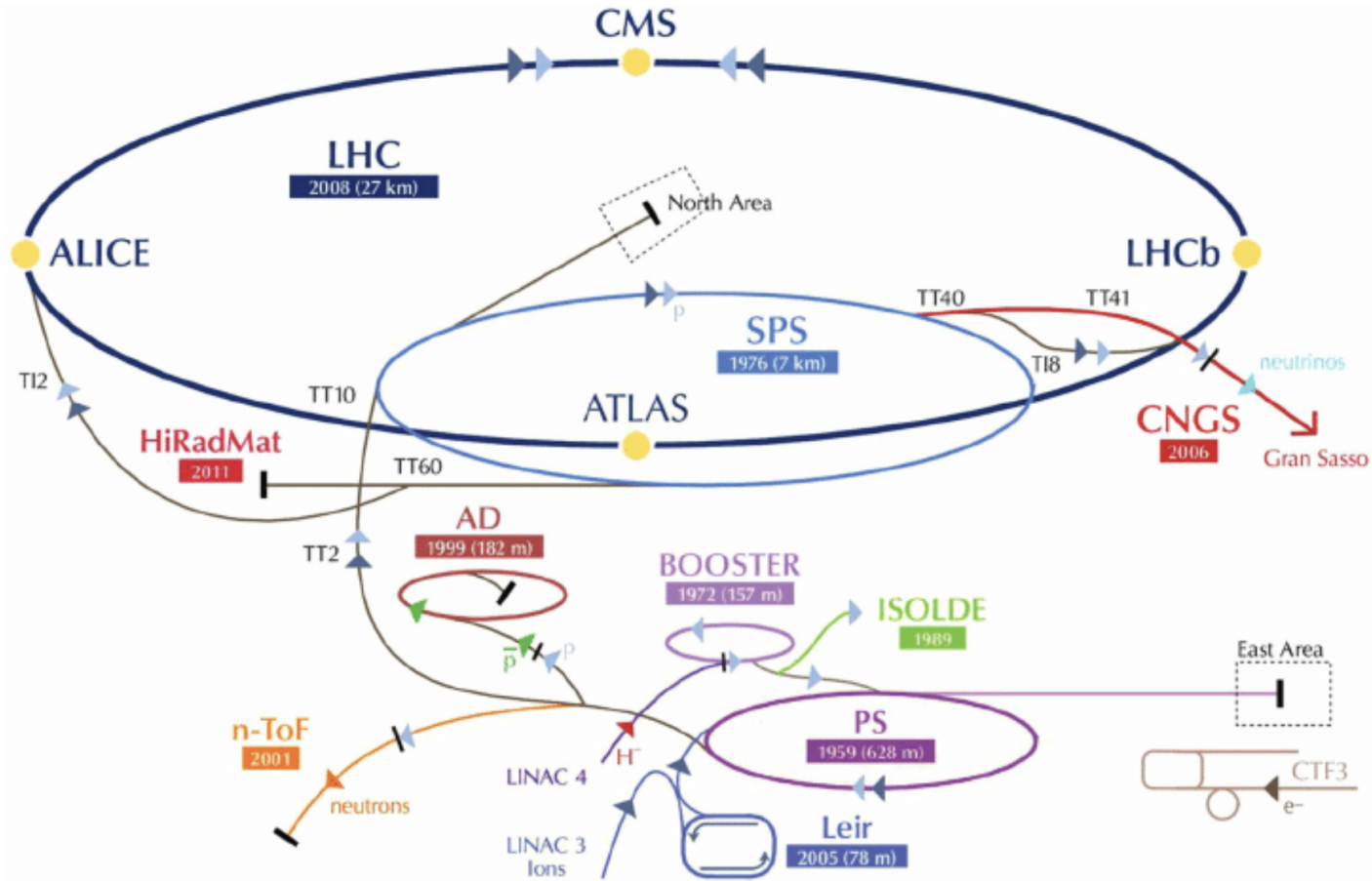
As well as firemen, cleaners and transport staff, teachers from the CERN nursery school and girl guides from the international troupe at Ferney-Voltaire helped to look after the children. The Geneva authorities kindly lent the playground equipment.

Our mission

Our mission is to:

- perform **world-class research** in fundamental physics.
- provide a unique range of **particle accelerator facilities** that enable research at the forefront of human knowledge, in an environmentally responsible and sustainable way.
- **unite people** from all over the world to push the frontiers of science and technology, for the benefit of all.
- **train new generations** of physicists, engineers and technicians, and **engage all citizens** in research and in the values of science.

A vocation for recycling

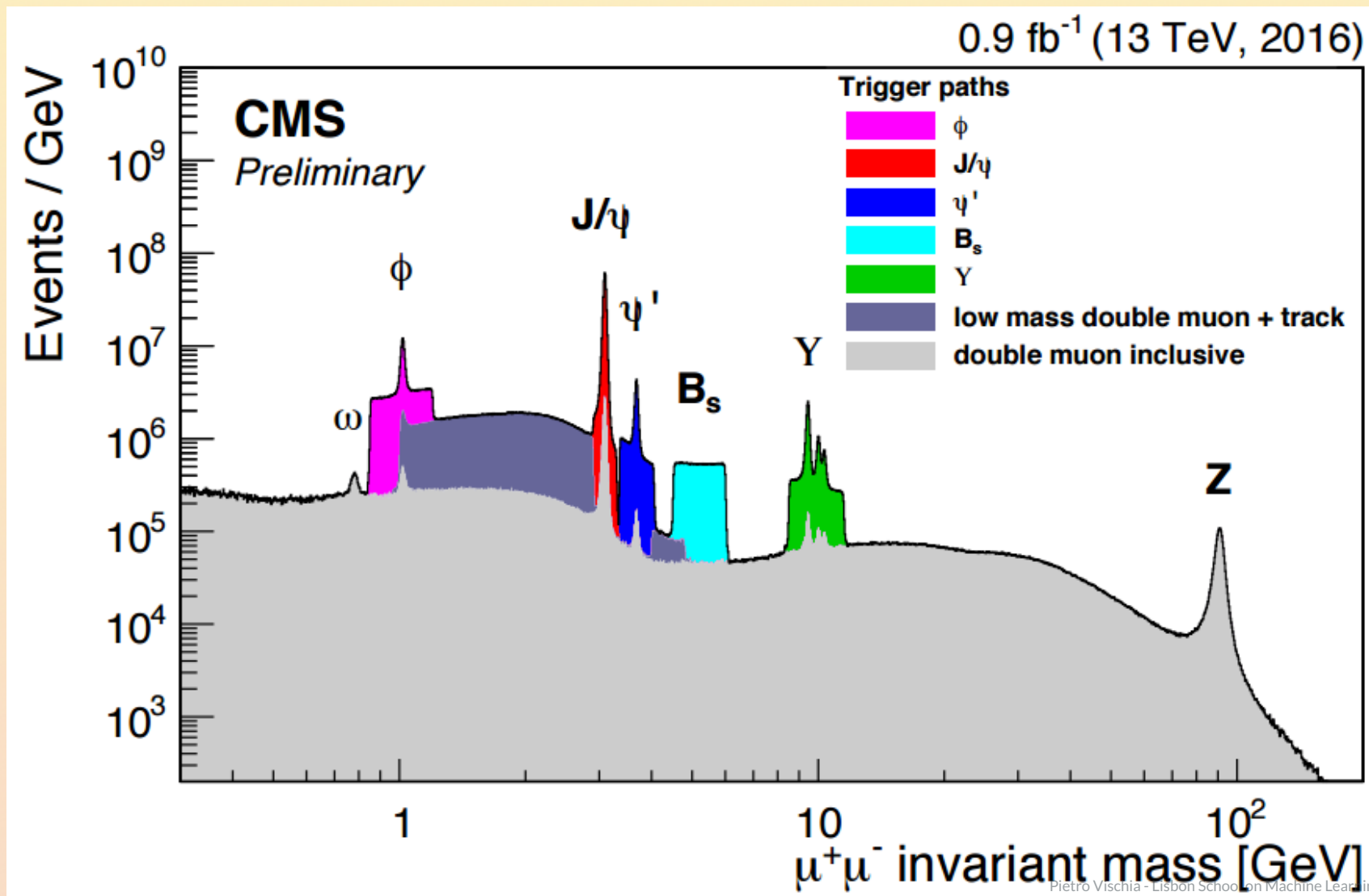


Existing CERN accelerator complex with Large Hadron Collider (LHC), Super Proton Synchrotron (SPS), Proton Synchrotron (PS), Antiproton Decelerator (AD), Low Energy Ion Ring (LEIR), Linear Accelerators (LINAC), CLIC Test Facility (CTF3), CERN to Gran Sasso (CNGS), Isotopes Separation on Line (ISOLDE), and neutrons Time of Flight (n-ToF).

Discover!

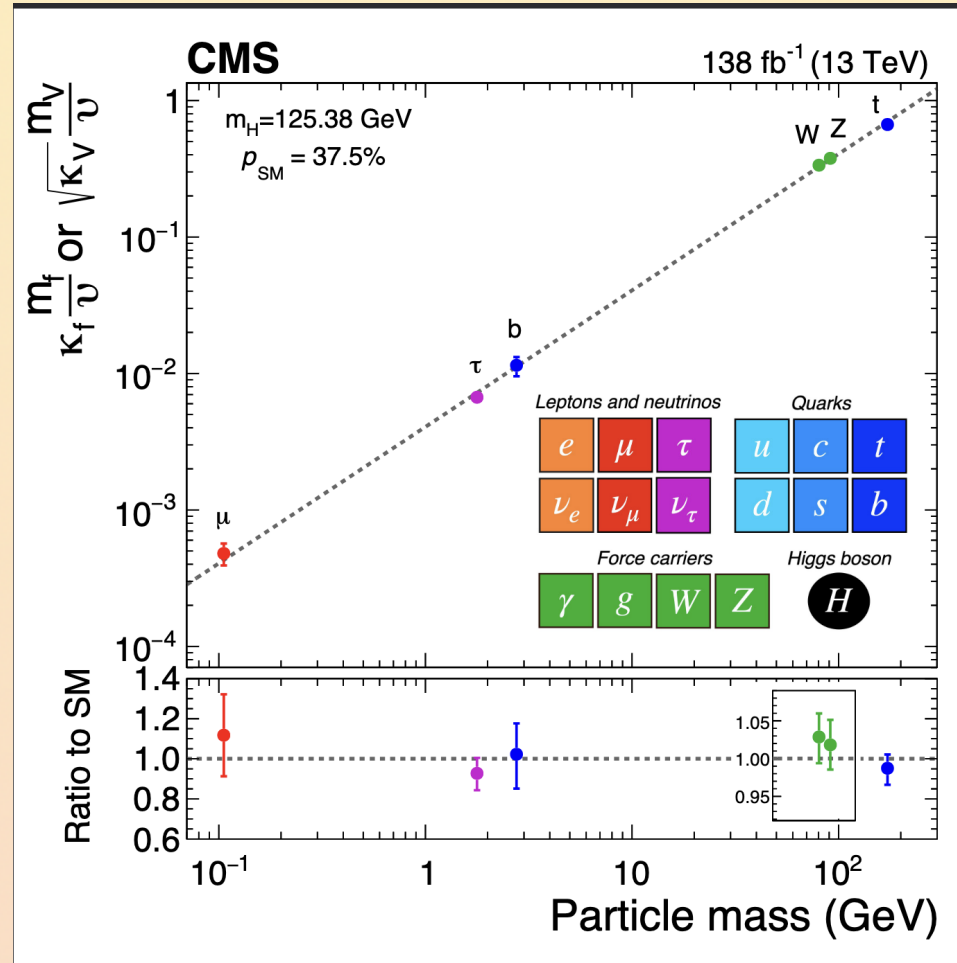


Keep rediscovering!



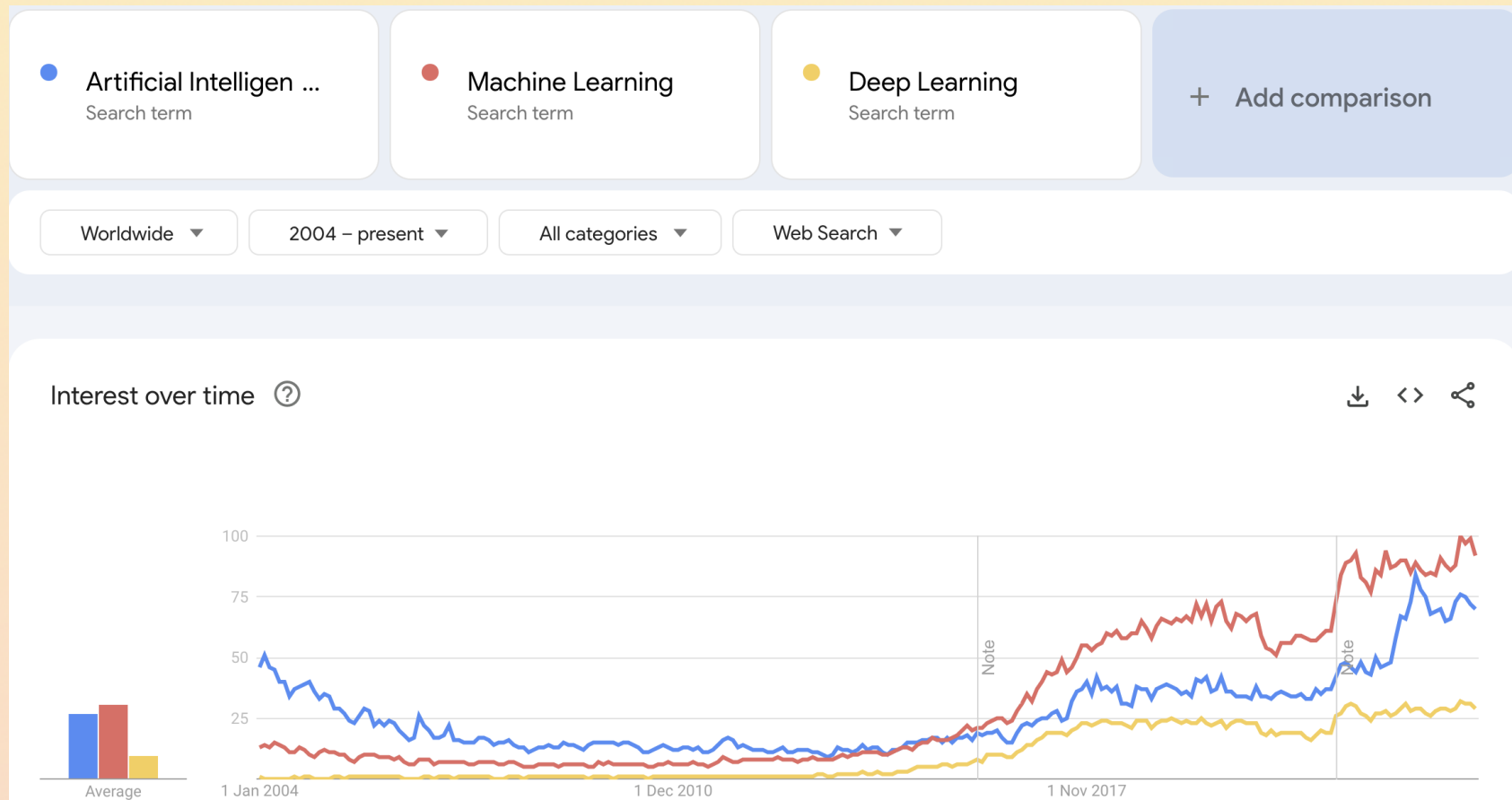
Our latest Nature paper

(strong ICTEA contrib!)



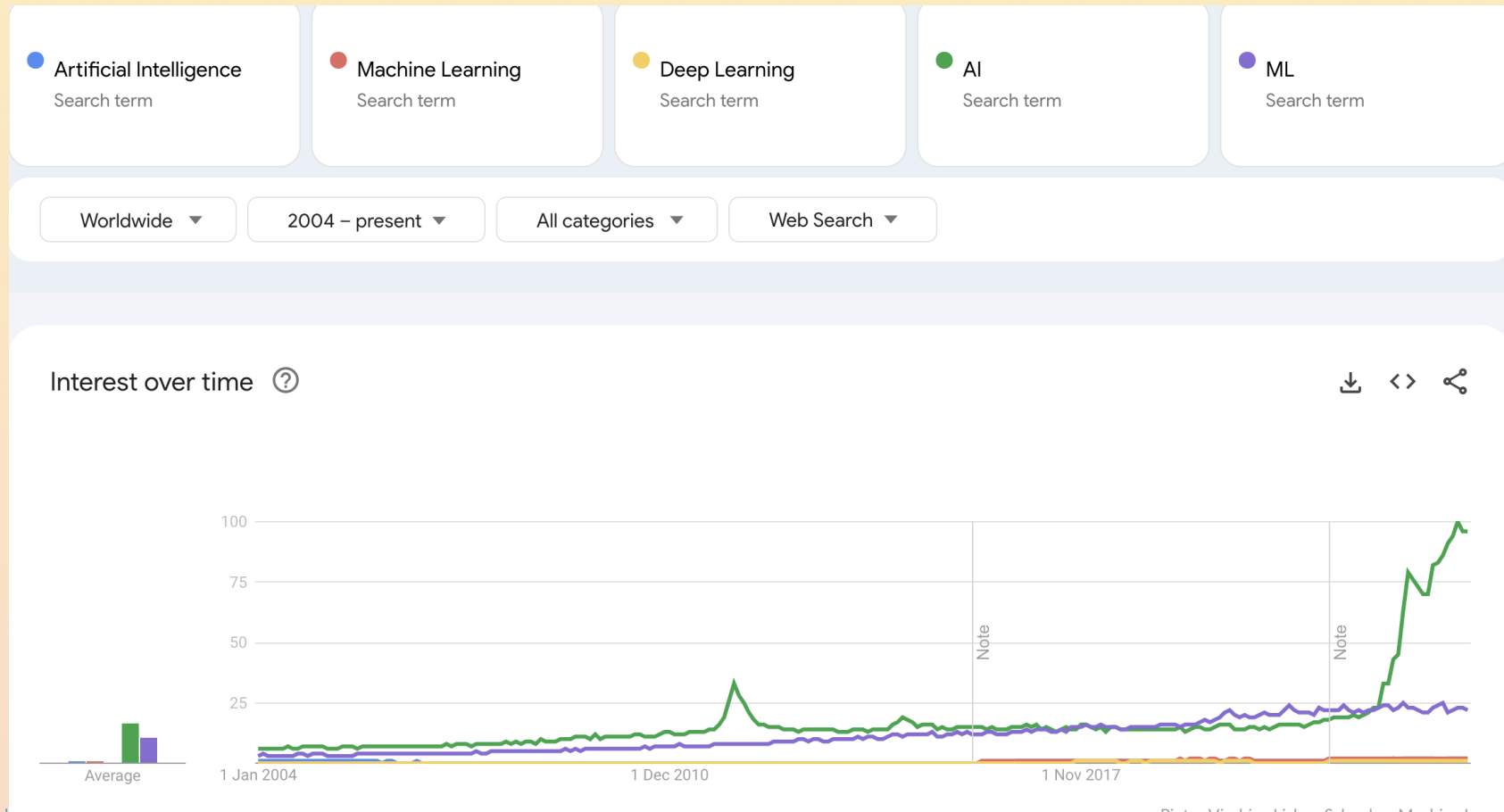
AI, the eternal buzzword?

- Artificial Intelligence (AI)
- Machine Learning (ML)
- Statistical Learning

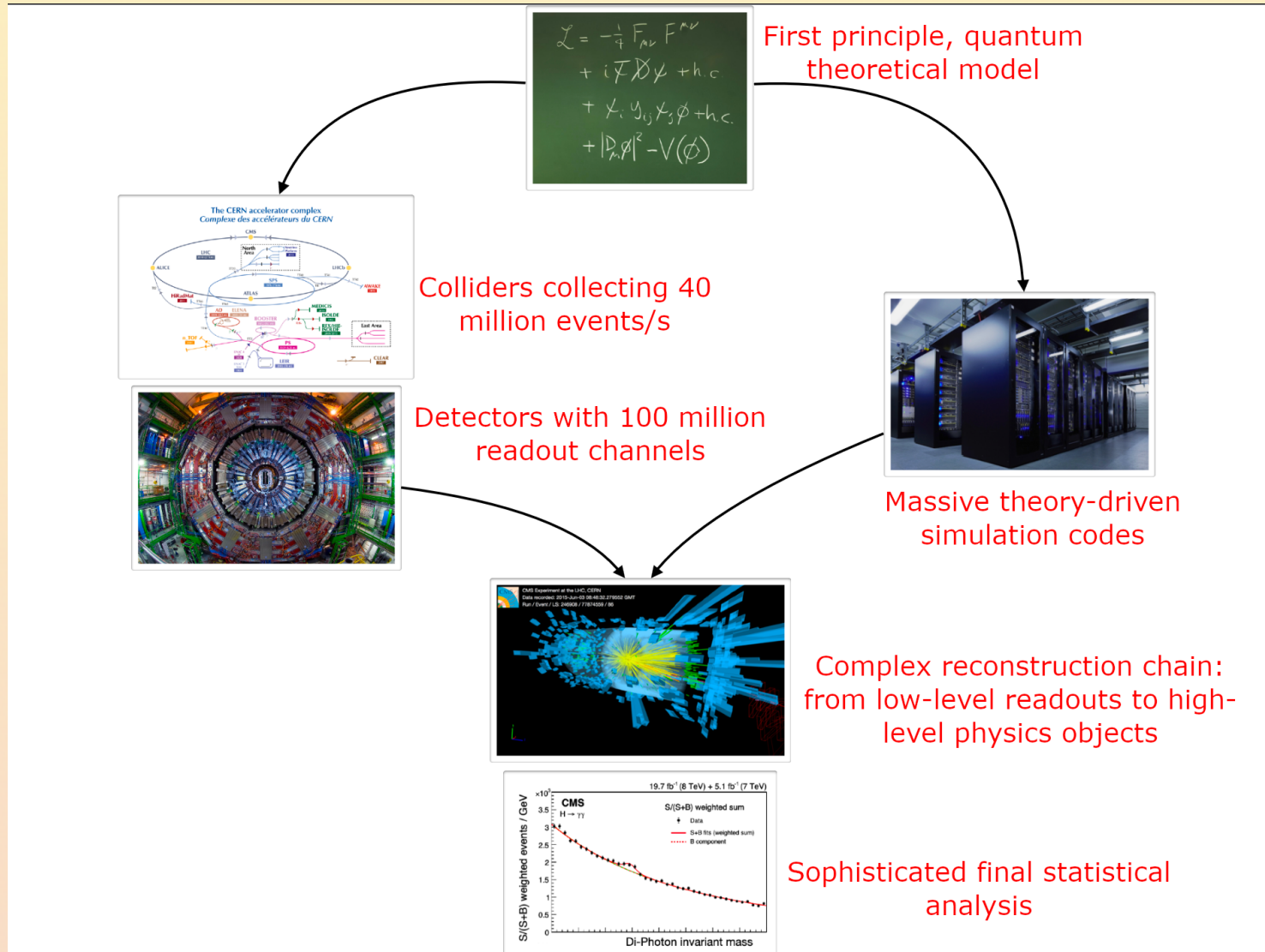


AI, the eternal buzzword?

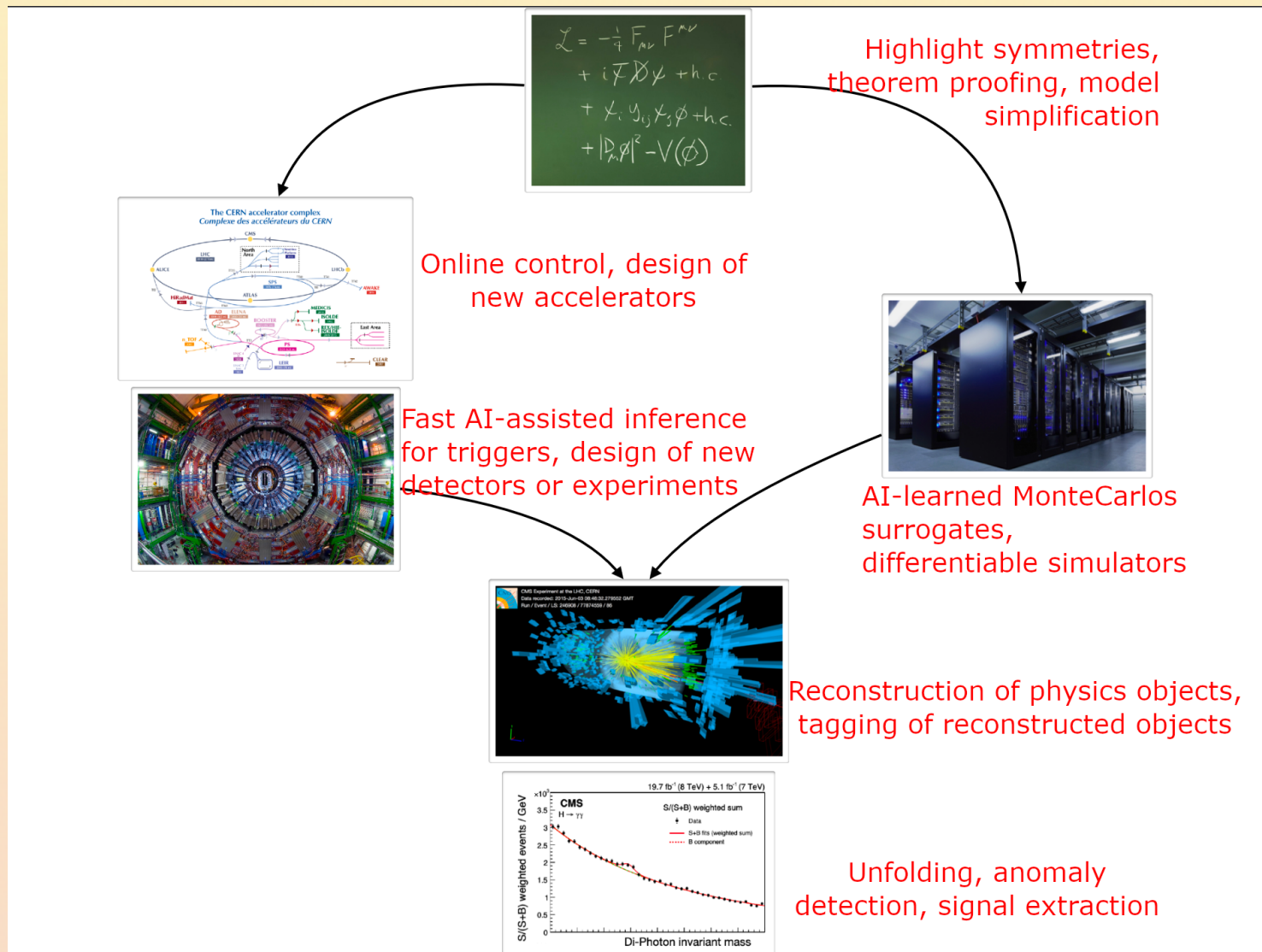
- Artificial Intelligence (AI)
- Machine Learning (ML)
- Statistical Learning



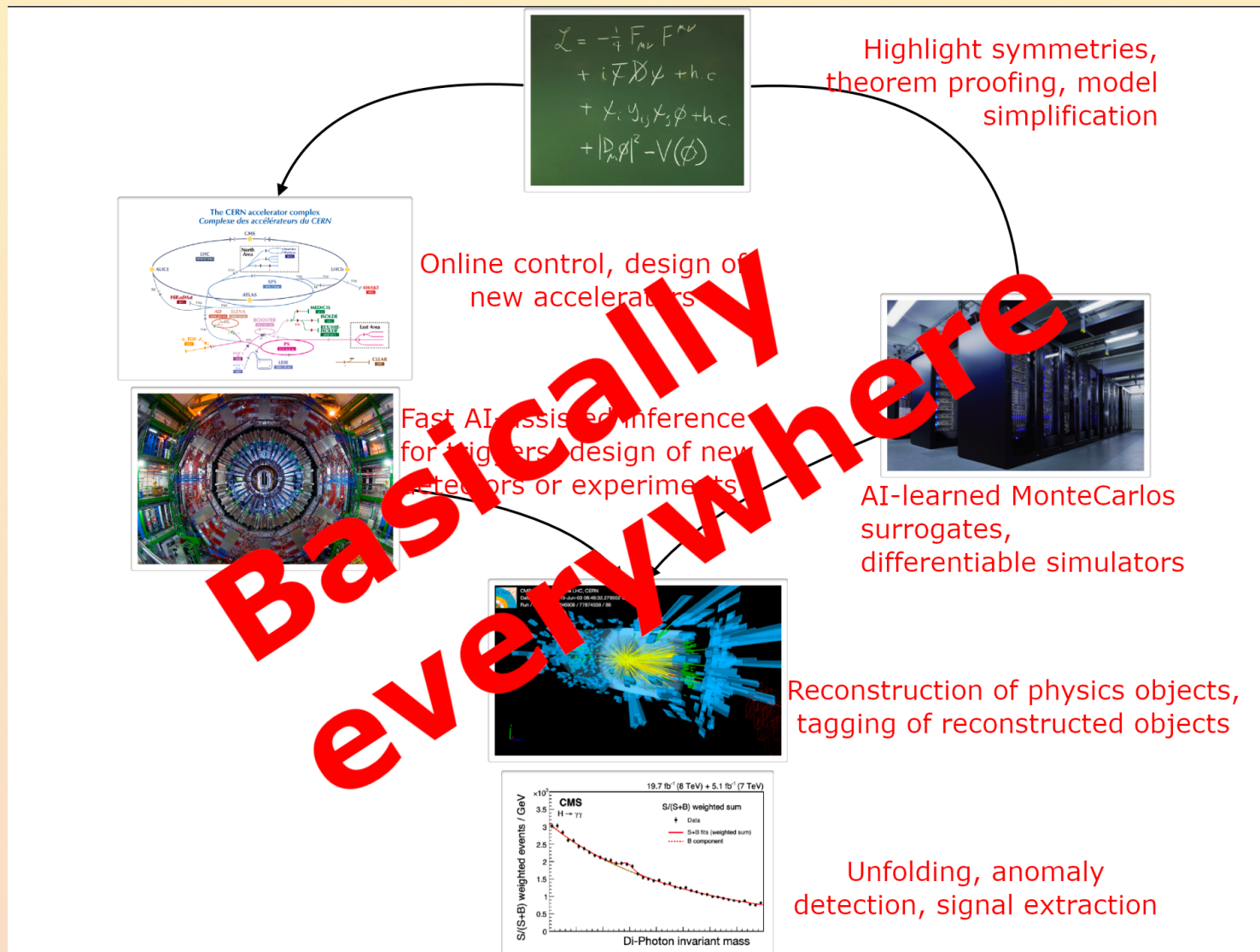
What do we do



Where we can plug AI




Where we can plug AI



Most theory papers are symbolic

- AI-assisted theorem proving



Lean [de Moura et al., 2015] **Coq** [Barras et al., 1997] **Isabelle** [Nipkow et al., 2002]

<https://machine-learning-for-theorem-proving.github.io/> (NeurIPS 2023)

- LLMs to solve mathematical problems

Article | [Open access](#) | Published: 14 December 2023

Mathematical discoveries from program search with large language models

[Bernardino Romera-Paredes](#) ✉, [Mohammadamin Barekatin](#), [Alexander Novikov](#), [Matej Balog](#), [M. Pawan](#)

- Simplify polylogarithms (no classical algorithm available, LLMs 91% success!)

Dutch:
naamsveranderingsdocumentenbriefgeheel

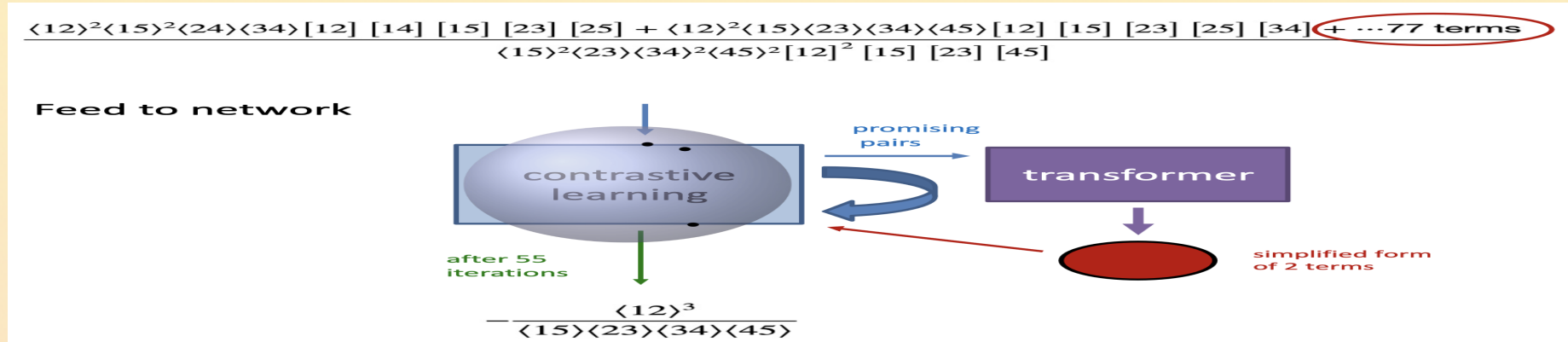
$$f(x) = 9 \left(-\text{Li}_3(x) - \text{Li}_3\left(\frac{2ix}{-i+\sqrt{3}}\right) - \text{Li}_3\left(-\frac{2ix}{i+\sqrt{3}}\right) \right) + 4 \left(-\text{Li}_3(x) + \text{Li}_3\left(\frac{x}{x+1}\right) + \text{Li}_3(x+1) - \text{Li}_2(-x) \ln(x+1) \right) - 4 \left(\text{Li}_2(x+1) \ln(x+1) + \frac{1}{6} \ln^3(x+1) + \frac{1}{2} \ln(-x) \ln^2(x+1) \right)$$

translate → **English: dossier**

$$f(x) = -\text{Li}_3(x^3) - \text{Li}_3(x^2) + 4\zeta_3$$

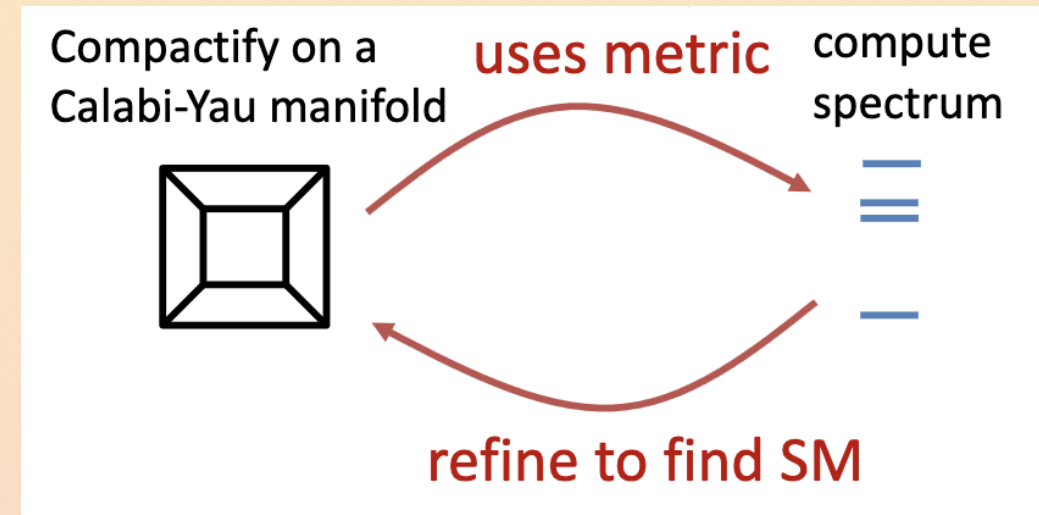
Most theory papers are symbolic

- 5-point MHV amplitude w/ Feynman diagrams: from 1990 tokens to 27 tokens



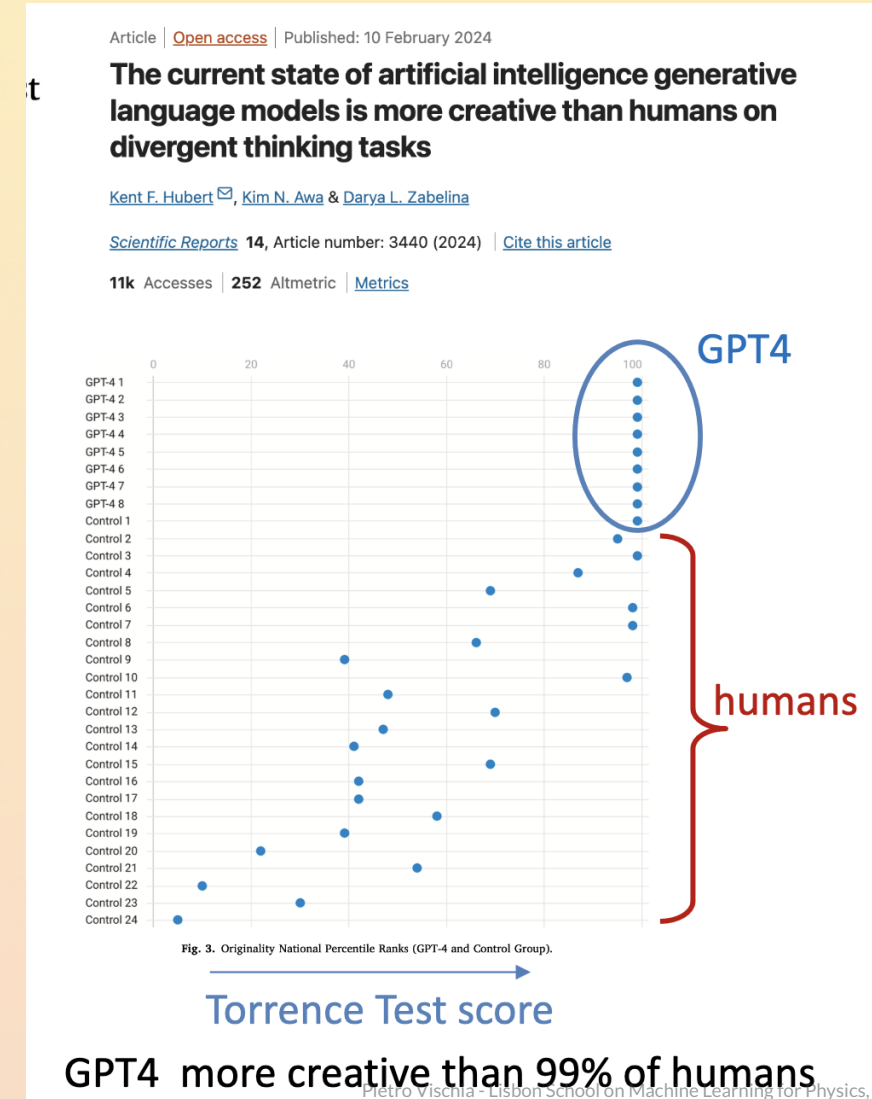
Solve string theory 🤪

- Find nontrivial Calabi-Yau metrics (1910.08605)
- Look for fixed points of metric flows (2310.19870)
- Predict rank of gauge group (1707.00655, prediction later proven)



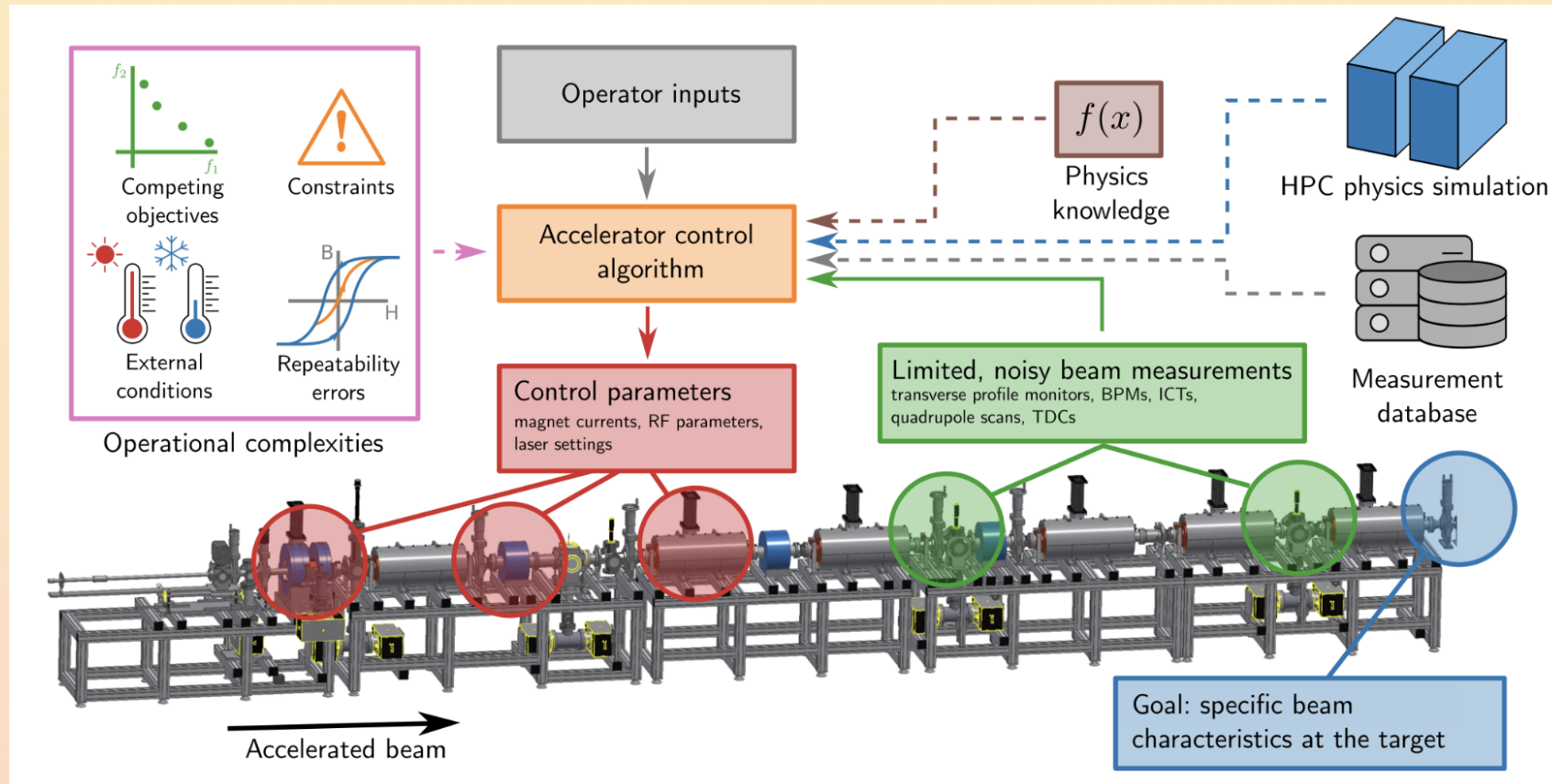
Beyond symbolic manipulation

- Can AI find interesting questions?
- Can AI models teach themselves to be good physicists using data?
- **If AI understands physics (can calculate everything) but we do not, do we consider it an acceptable "understanding"?**



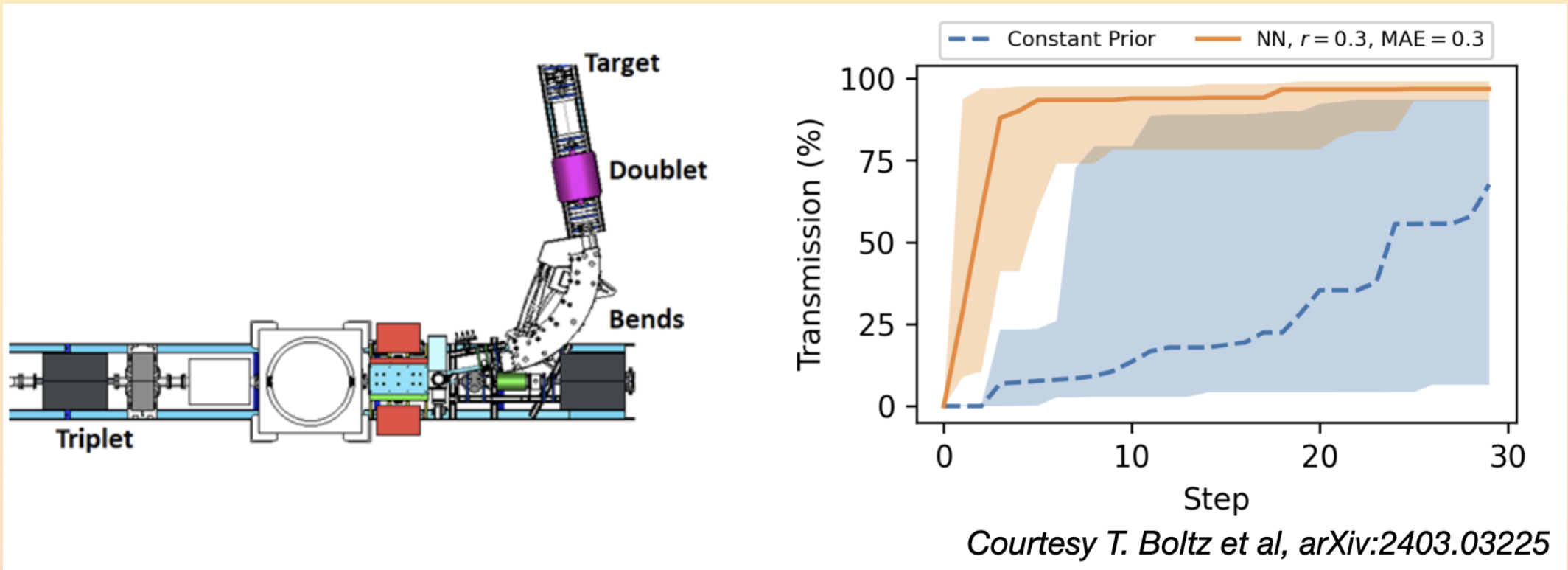
Accelerate accelerators

- Daily operation and control have huge impact on resources and efficiency
 - Beam scheduling: changing supercycle requires 20-100 clicks (2-25min) about 60 times/day
 - 15% of the yearly cost of SPS fixed target cycle employed for "waste" cycles to mitigate hysteresis problems
- What if we could make them fully automatic (like e.g. Space telescopes)?



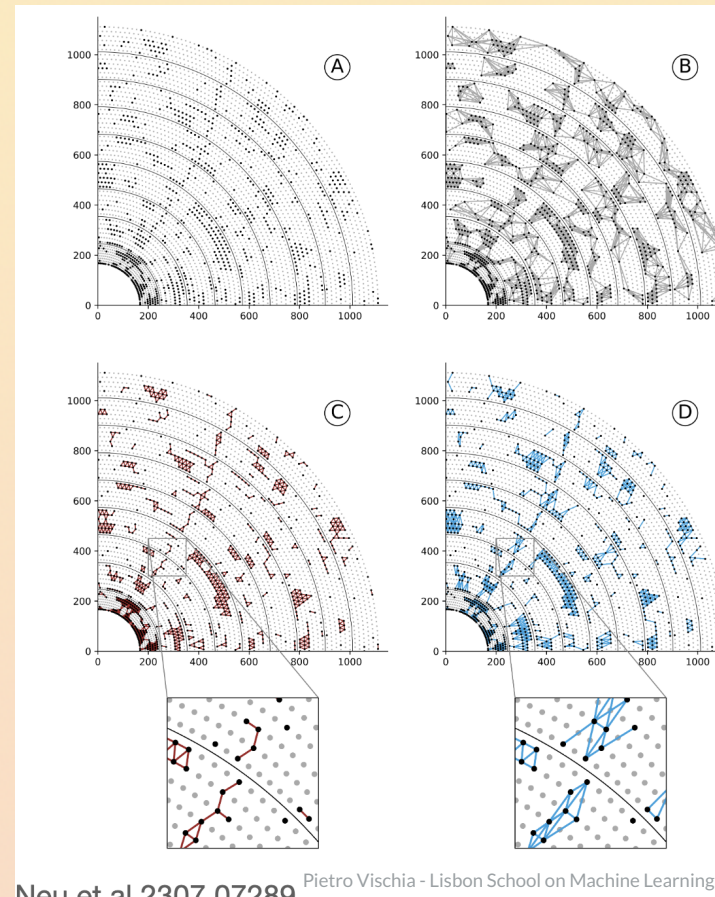
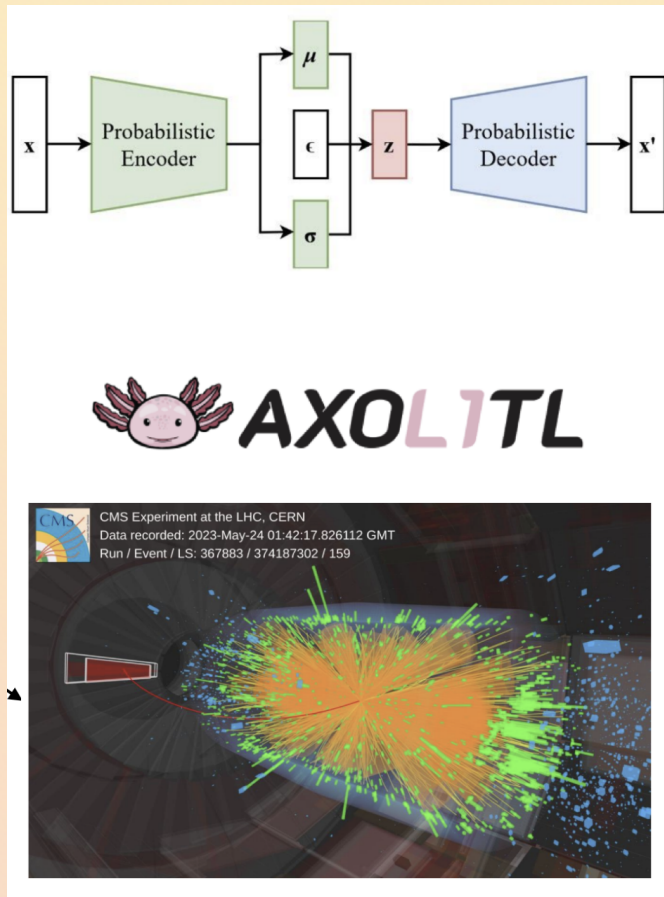
Accelerate accelerators

- Hierarchical, AI-controlled autonomous systems
- Optimize transmission to target in a system with 5 DoF, using Bayesian Optimization



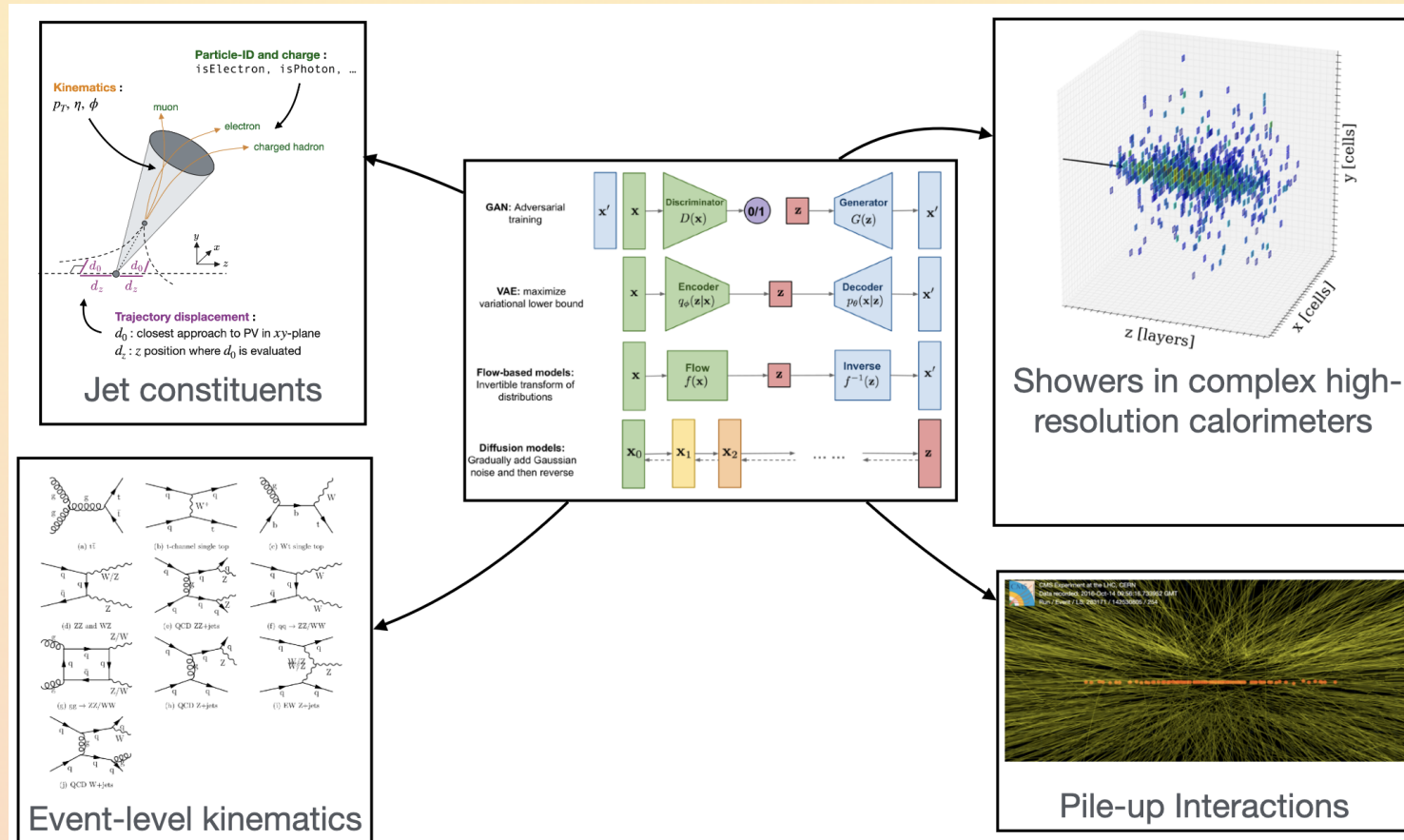
Trigger

- Pack AI models into the L1 trigger → improve selection criteria
 - At ICTEA!
- Can do e.g. anomaly detection, and online graph building



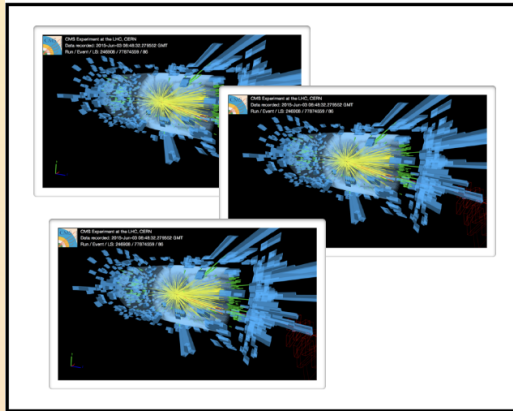
Simulations: the problem

- Monte Carlo simulations are very costly
- The more data we collect, the more simulated events we need

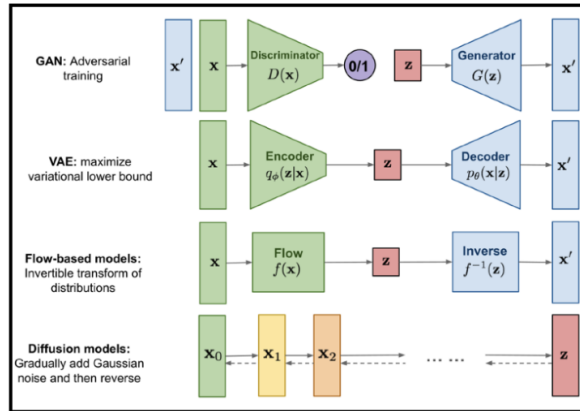


Simulation: two solutions

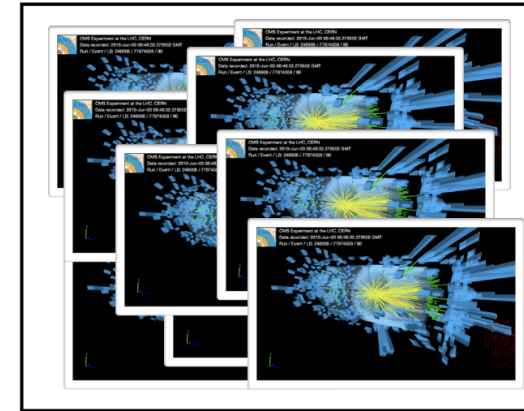
1. Use classical simulation or collider data as input



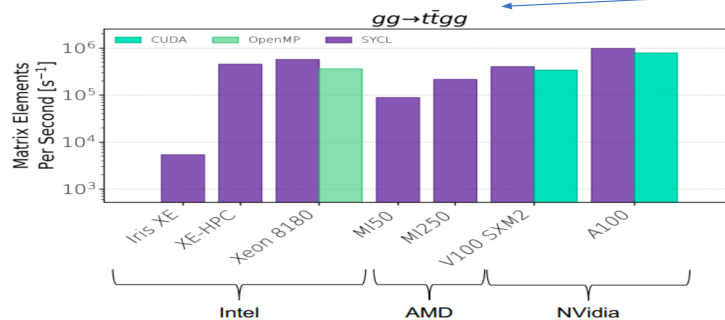
2. Train generative surrogate



3. Oversample



- Very recently, Madgraph5_aMC@NLO authors deployed a version of their code that can run on GPUs.
- This version significantly improves computation times (see [this talk](#)).



Talk by C. Vico Villalba

So our idea is: can we do this on hardware based accelerators?

- **FPGAs** are:
 - **Highly** parallelizable
 - In some cases not as fast as GPU.
 - But less power consuming.
 - Hardware based! really versatile.

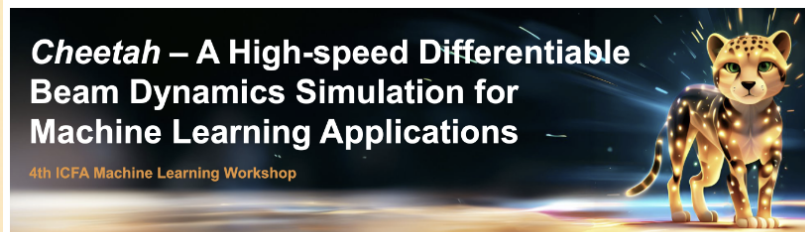


See talk by P. Leguina.

Simulation: long term solution

- Make everything differentiable, exploiting [differentiable programming](#)

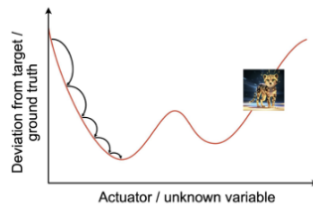
Cheetah



Gradient-based Tuning

Transverse beam tuning at ARES

- Tune magnet settings or lattice parameters using the **gradient of the beam dynamics model** computed through **automatic differentiation**.
- Seamless **integration with PyTorch** tools tuning neural networks.
- Becomes very useful for **high-dimensional tuning tasks** (see neural network training).



```
ares_ea.AREAMQZM1.k1 = nn.Parameter(0.0)
ares_ea.AREAMQZM2.k1 = nn.Parameter(0.0)
ares_ea.AREAMCVMI.angle = nn.Parameter(0.0)
ares_ea.AREAMQZM3.k1 = nn.Parameter(0.0)
ares_ea.AREAMCHM1.angle = nn.Parameter(0.0)

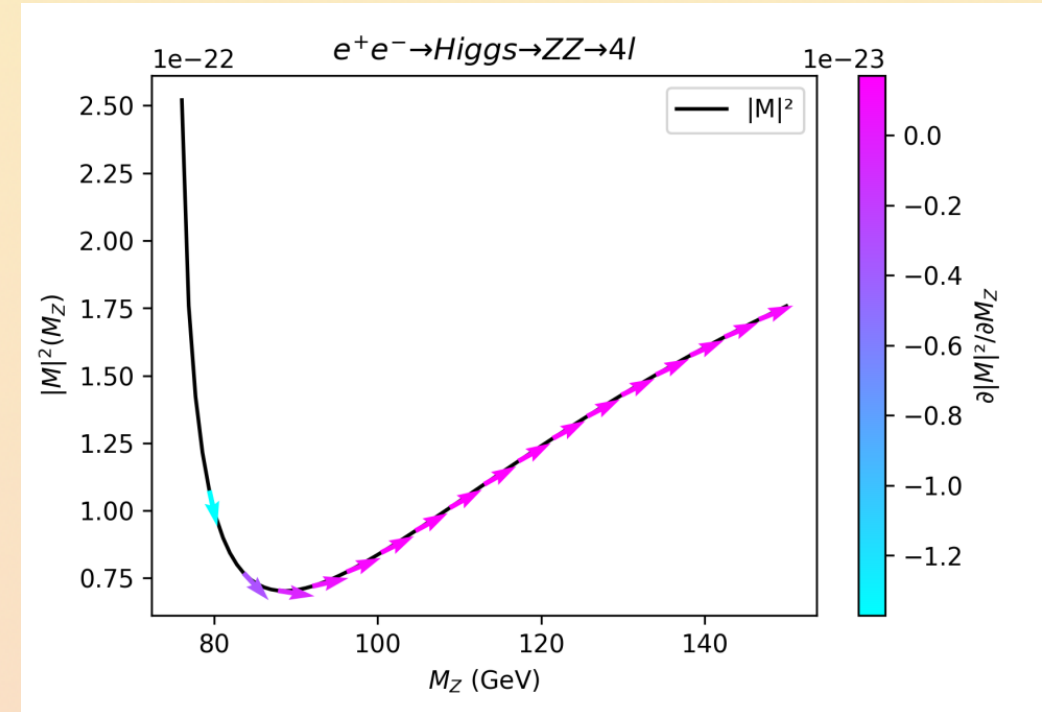
optimizer = Adam(ares_ea.parameters())

for _ in range(42):
    outgoing = ares_ea.track(incoming)
    loss = loss_fn(outgoing)

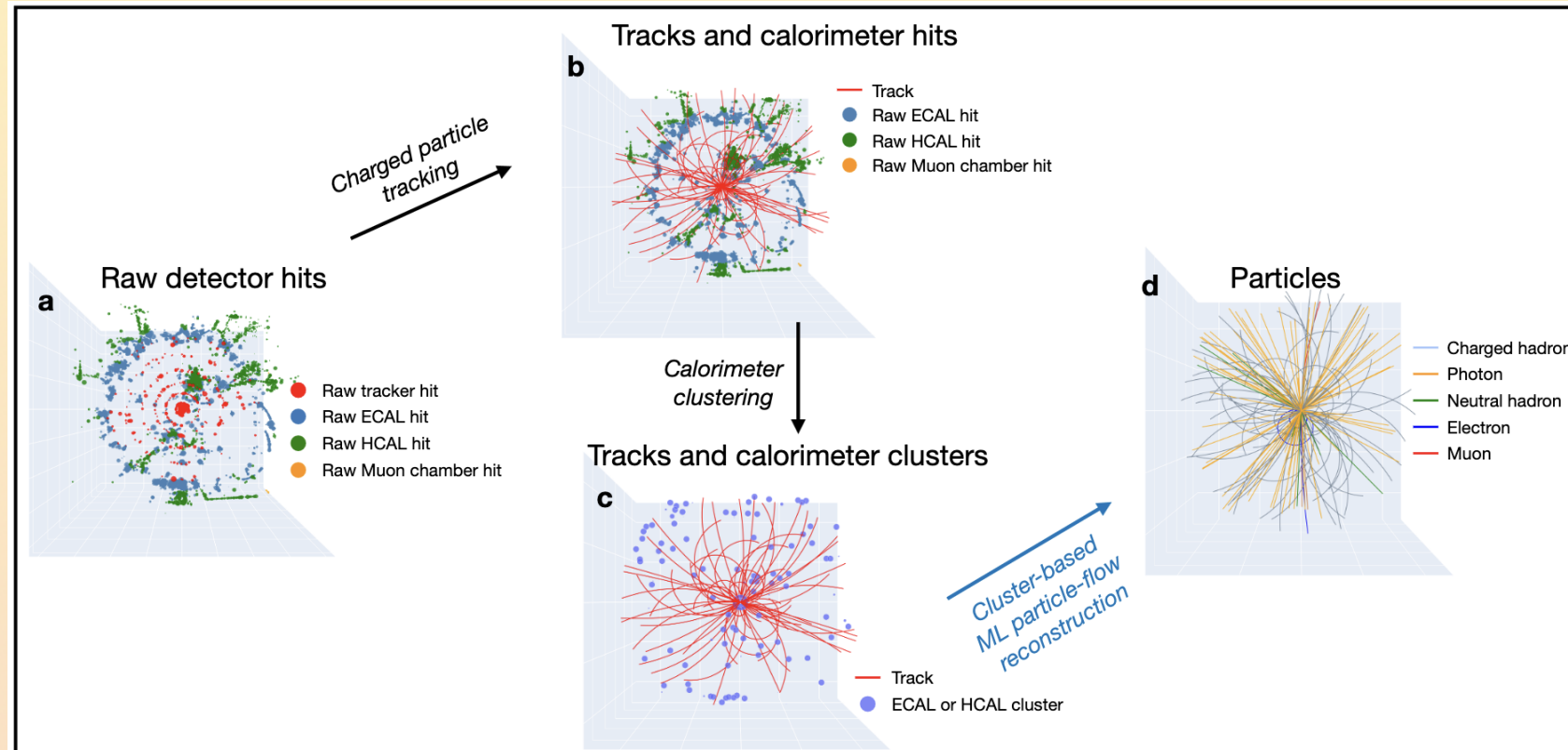
    loss.backward()
    optimizer.step()
    optimizer.zero_grad()
```

MadJax

(differentiable matrix element computation)



Reconstruction...

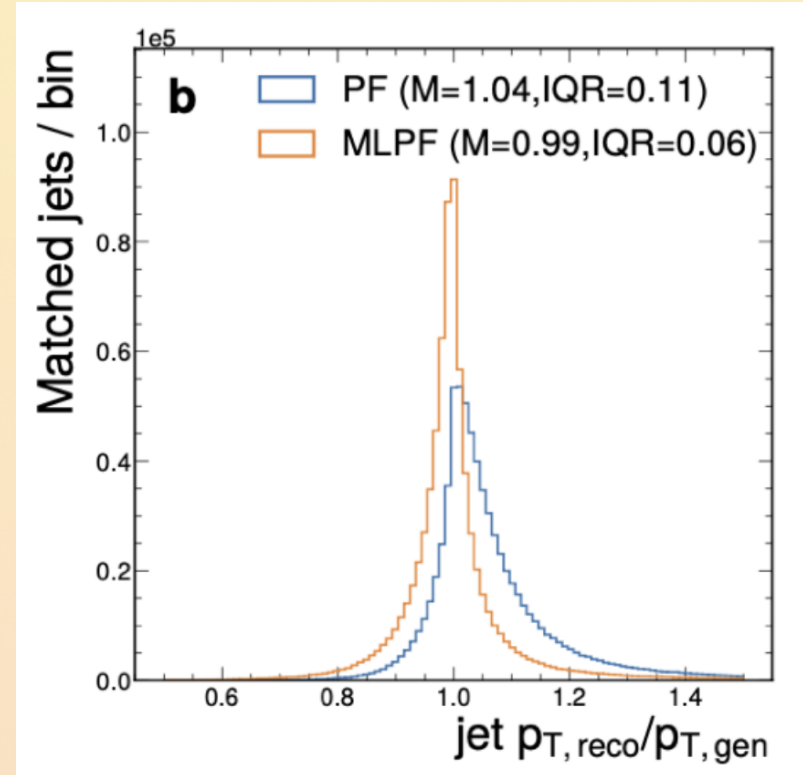
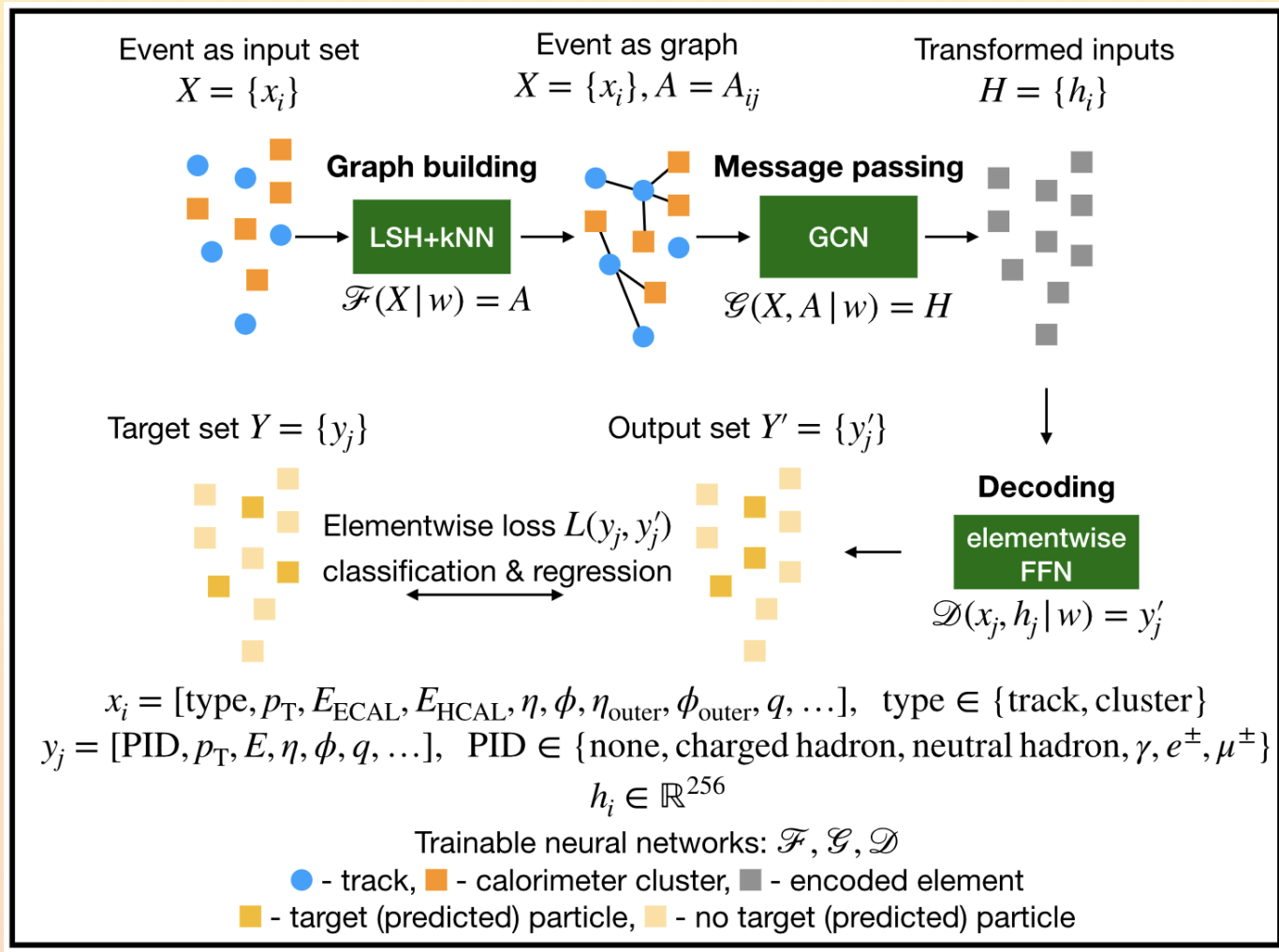


Reconstruction maps **low-level** detector read-outs

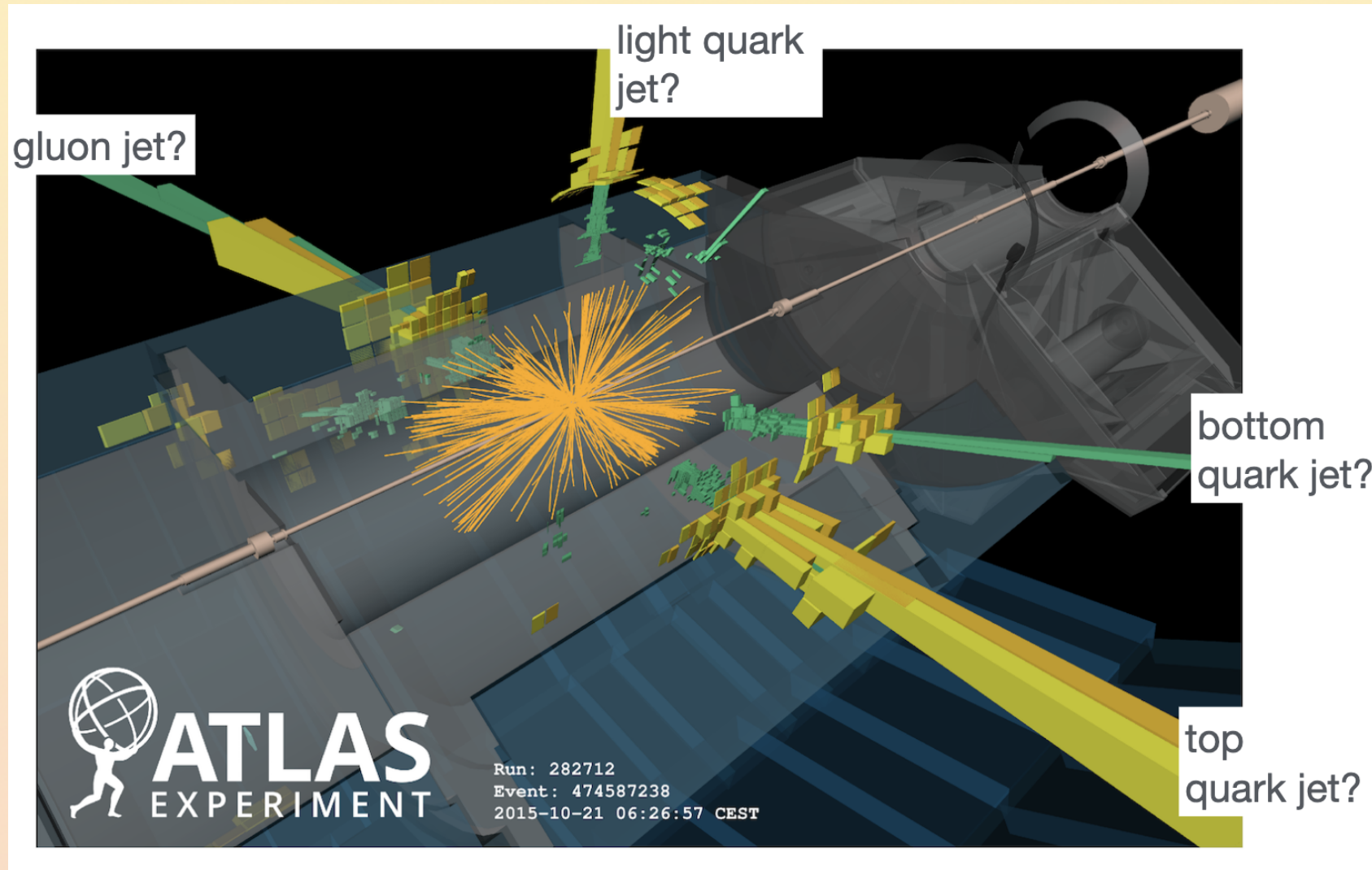
to **physical particles**

(Which in-turn are the basis of higher-level interpretation)

...with AI

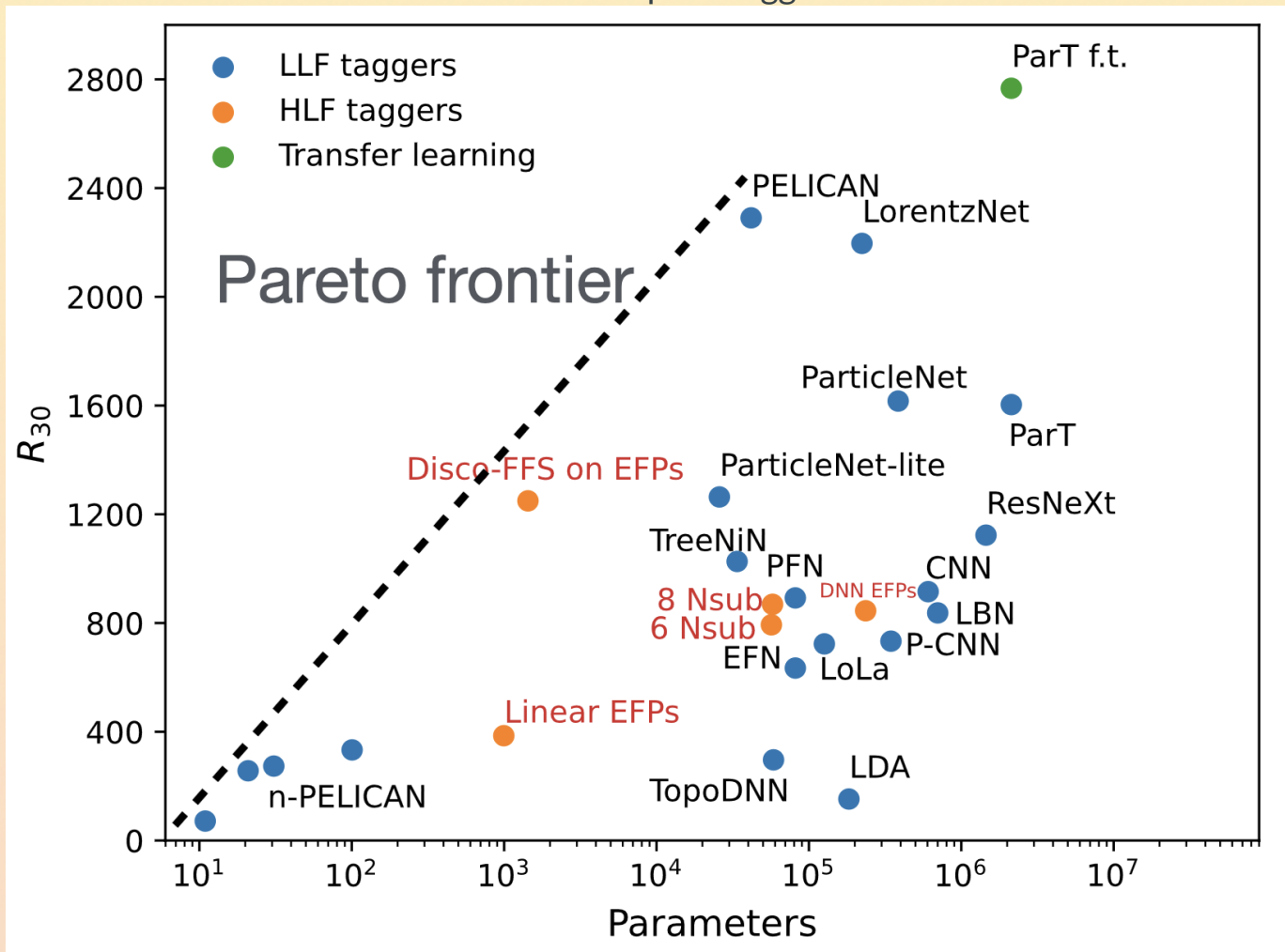


Identification...

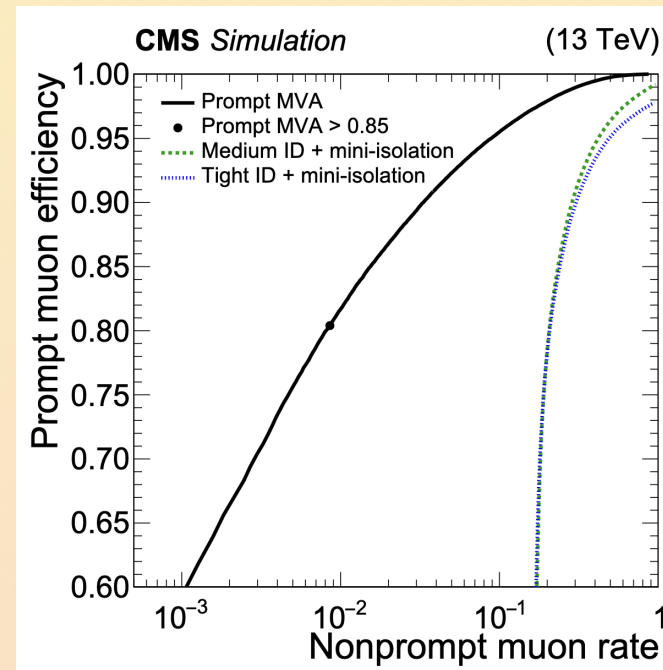


...with AI

Vast landscape of taggers

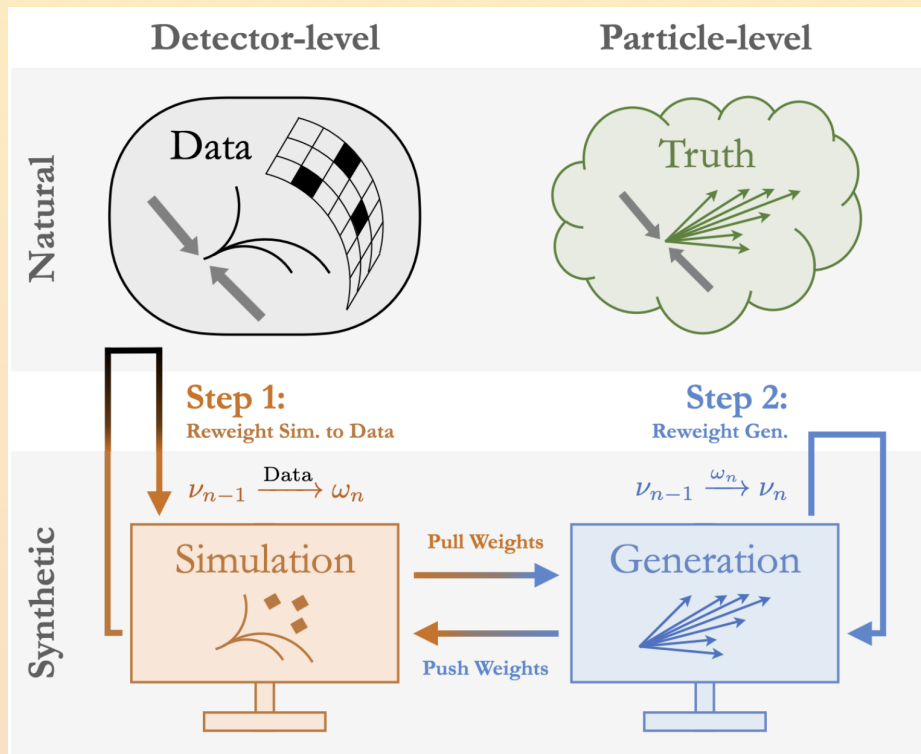


CMS Muon ID: **made in ICTEA!**

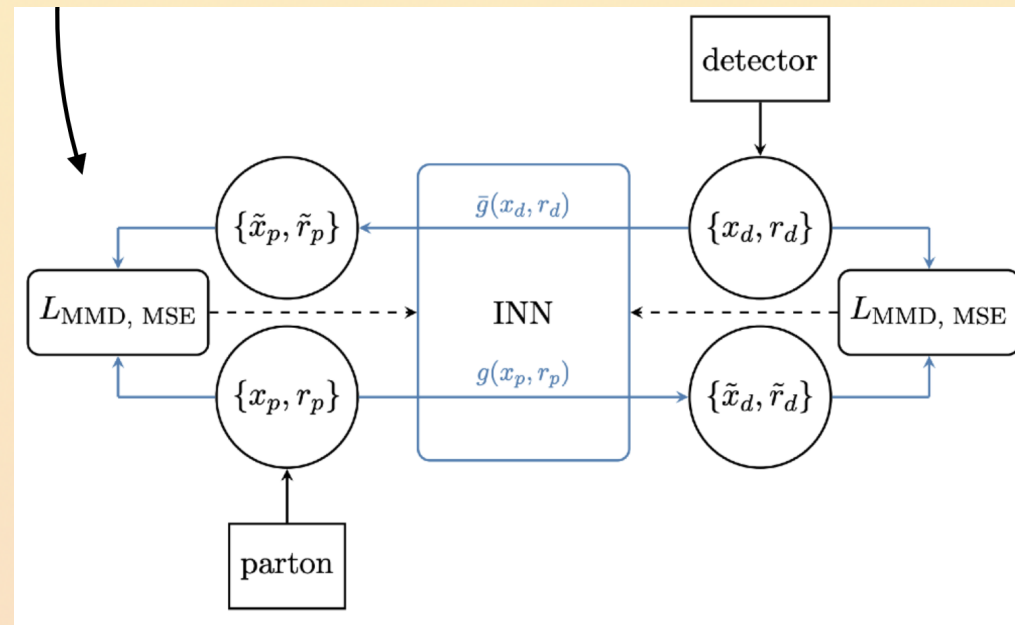


Inference: unfolding

- Use classifiers to learn appropriate weights



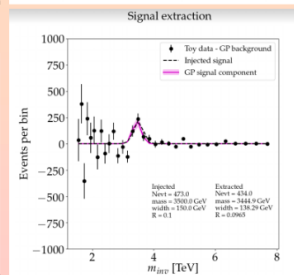
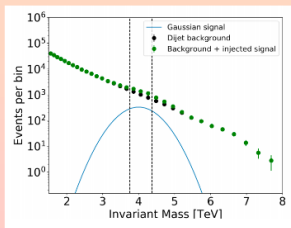
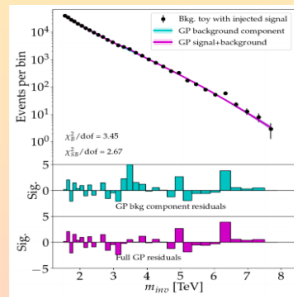
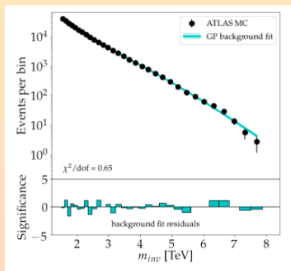
- Morph distributions one into the other using diffusion models



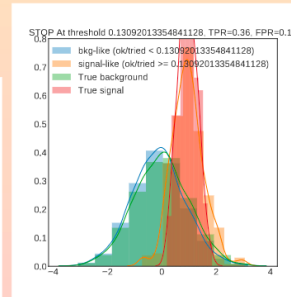
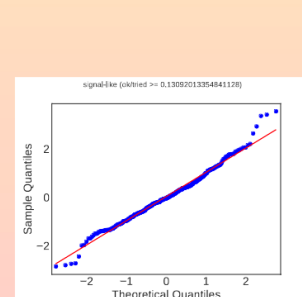
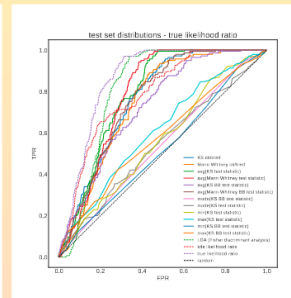
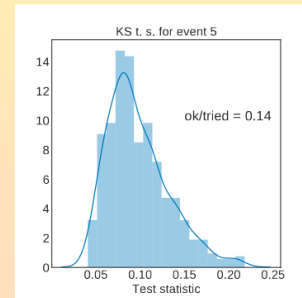
Inference: anomaly detection

Gaussian processes

- Multivariate gaussian associated to a set of random variables ($N_{dim} = N_{random\ variables}$)
 - *Kernel* as a similarity measure between bin centers (counts) and a *averaging function*
- $\mu(x) = 0,$
- $\Sigma_B(x, x') = A \exp\left(\frac{d - (x + x')}{2a}\right) \sqrt{\frac{2l(x)l(x')}{l(x)^2 + l(x')^2}} \exp\left(\frac{-(x - x')^2}{l(x)^2 + l(x')^2}\right),$
- $\Sigma_S(x, x') = C \exp\left(-\frac{1}{2}(x - x')^2/k^2\right) \exp\left(-\frac{1}{2}((x - m)^2 + (x' - m)^2)/t^2\right),$
- Signal is not parameterized
- Hyperparameters fixed by the B-only fit
- S: residual of B-subtraction



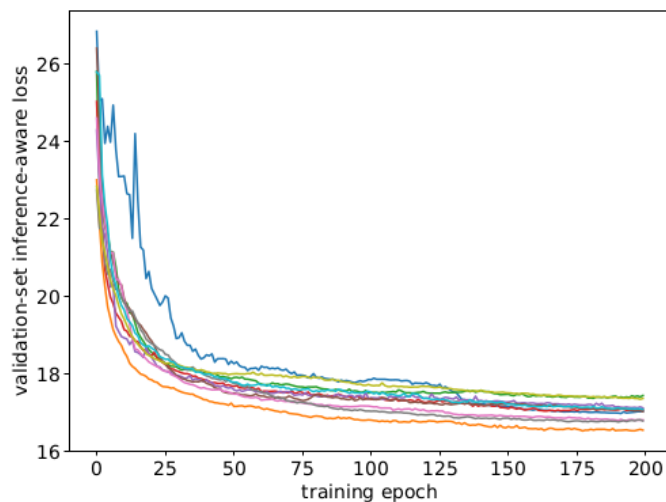
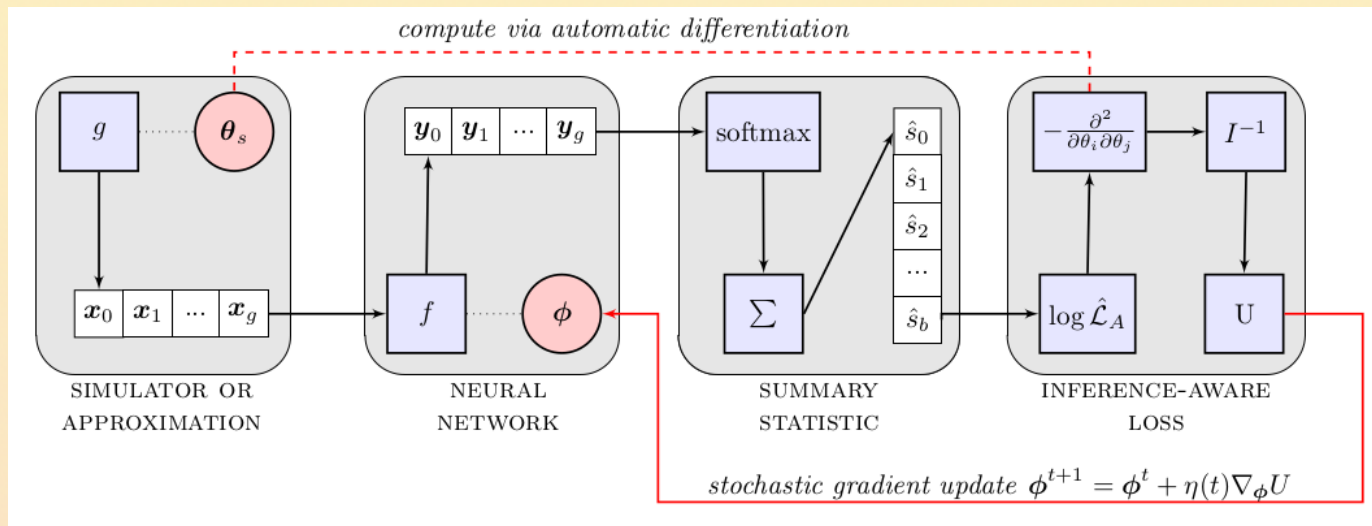
- Data: mixture model with small S
- Classification based on sample properties
 - Compare bootstrapped samples with reference (pure B)
 - Use Metodiev theorem to translate inference into signal fraction
- Validate with LR y LDA **At ICTEA!**
 - Promising results



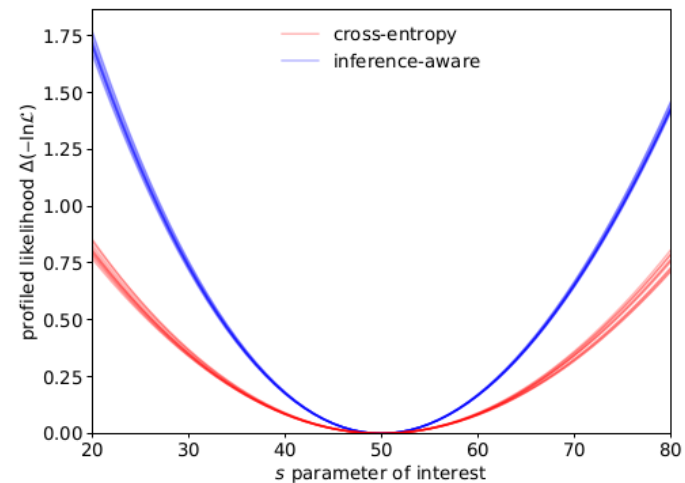
Vischia-Dorigo arXiv:1611.08256, doi:10.1051/epjconf/201713711009, and P.

Vischia's talk at EMS2019

Go to INFERNO: syst-aware inference opt.

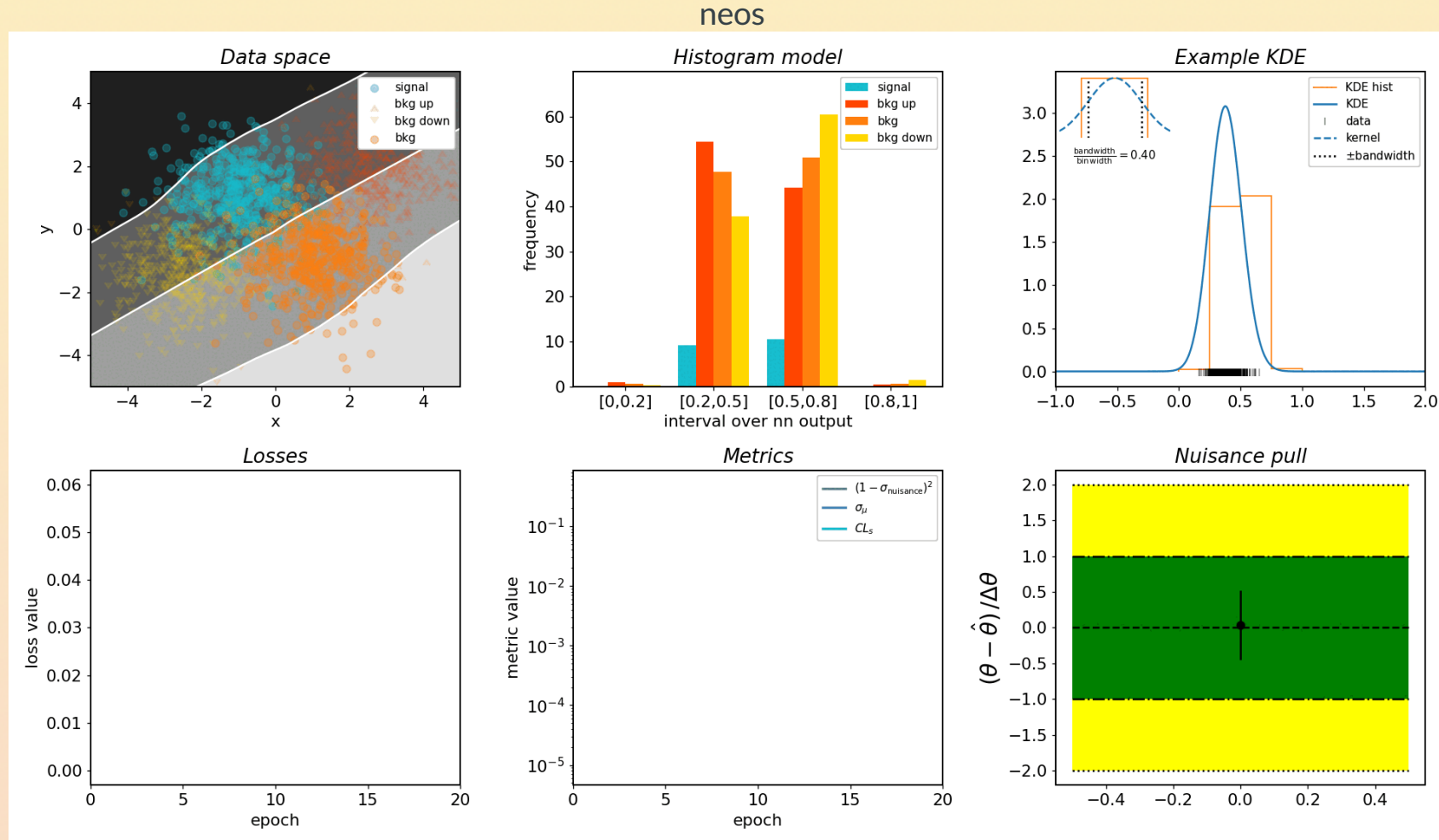


(a) inference-aware training loss



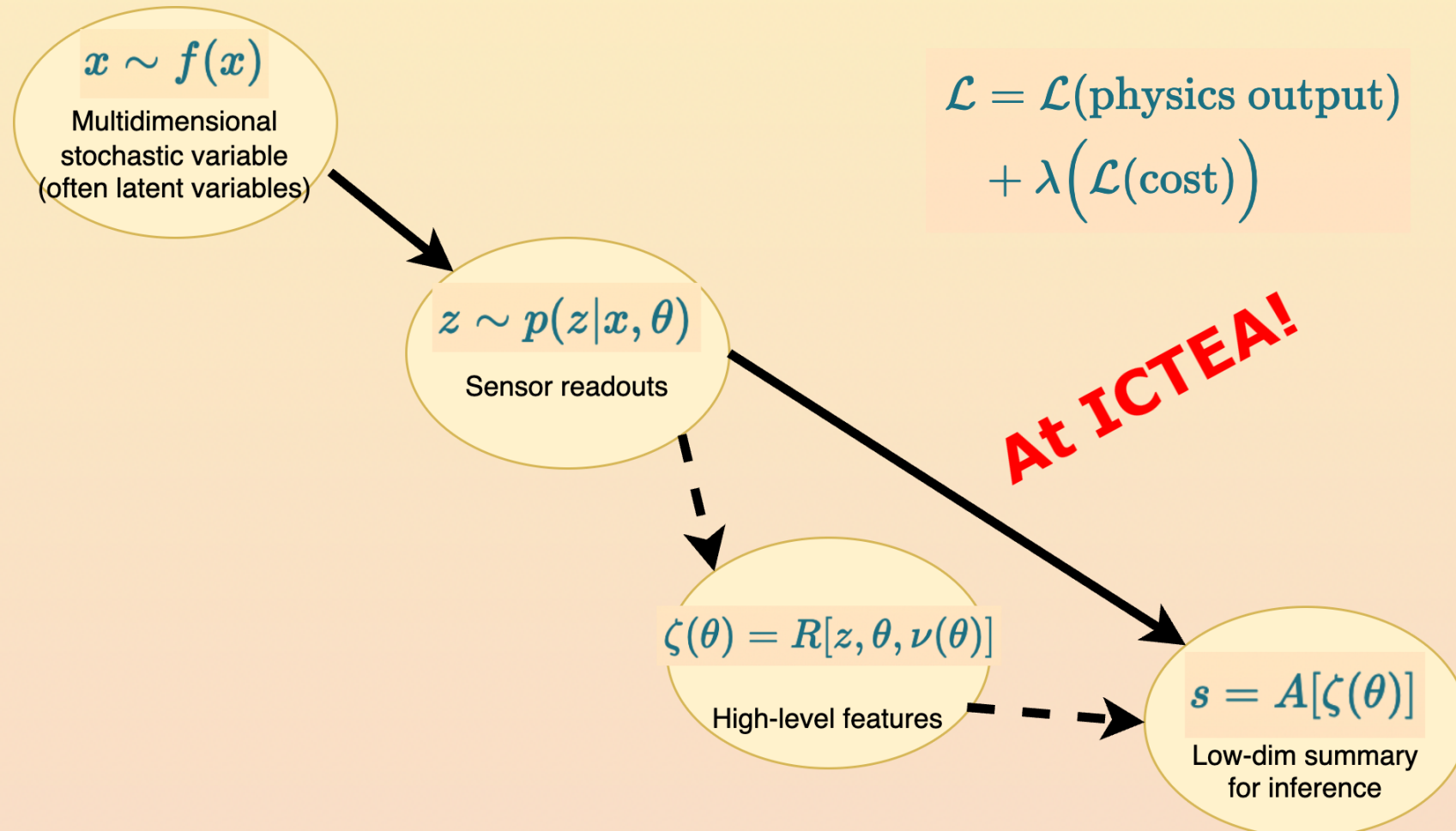
(b) profile-likelihood comparison

Measurement-aware analysis opt.



Measurement-aware detector opt.!

- Joint optimization of design parameters w.r.t. inference made with data
- MODE White Paper, [10.1016/j.revip.2023.100085 \(2203.13818\)](https://arxiv.org/abs/2203.13818), 117-pages document, physicists + computer scientists

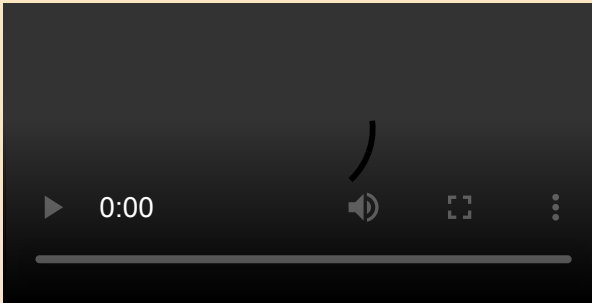


Prototype for muon tomography

TomOpt: Differential optimisation for task- and constraint-aware design of particle detectors in the context of muon tomography

Giles C. Strong, Maxime Lagrange, Aitor Orio, Anna Bordignon, Florian Bury, Tommaso Dorigo, Andrea Giammanco, Mariam Heikal, Jan Kieseler, Max Lamparth, Pablo Martínez Ruíz del Árbol, Federico Nardi, Pietro Vischia, Haitham Zaraket

We describe a software package, TomOpt, developed to optimise the geometrical layout and specifications of detectors designed for tomography by scattering of cosmic-ray muons. The software exploits differentiable programming for the modeling of muon interactions with detectors and scanned volumes, the inference of volume properties, and the optimisation cycle performing the loss minimisation. In doing so we provide the first demonstration of end-to-end-differentiable and inference-aware optimisation of particle physics instruments. We study the performance of the software on a relevant benchmark scenarios and discuss its potential applications.










CERN AI structures

- CERN Interexperimental Machine Learning Working Group, <https://iml.web.cern.ch>

The IML working group holds regular meetings open to all interested parties and maintains a discussion forum to facilitate the exchange of information among the LHC experiments in machine learning. The IML working group also fosters connections with other HEP experiments and the ML community at large.

Coordinators

**ICTEA
(2020-2024)**

Fabio Catalano (ALICE) 	Lorenzo Moneta (SFT) 	Pietro Vischia (CMS) 
Stefano Carrazza (TH) 	Anja Butter (TH) 	Julian Garcia Pardinias (LHCb) 
Daniel Whiteson (ATLAS) 		

CMS AI structures

- Shared coordination area between **Physics** and **Offline&Computing**
- Informs the collaboration, promotes new techniques, review AI-based analyses
- Real impact on steering AI applications in a 5000k members collaboration

ML Group Coordinator: Pietro Vischia

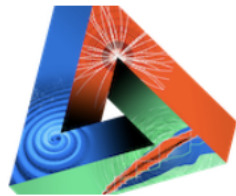
- CMS member since 2009
- Ph.D. in 2016 from Instituto Superior Técnico (Lisboa)
 - Bachelor and Master's: Università degli Studi di Padova
- "Ramón y Cajal" Senior Researcher at Universidad de Oviedo and ICTEA since 01/2023
 - Postdoc: UCLouvain (BE, 2018-2022), Universidad de Oviedo (ES, 2016-2018)
 - Predoc: research fellow at LIP Lisboa (PT, 2011-2016)
- Analysis highlights: top mass and xsec, charged Higgs, WZ, ttH multilepton
- Roles in CMS
 - CERN **IML coordinator** for CMS (2020-2024)
 - **Statistics Committee** member (2015-ongoing)
 - **HiggsWW** L3 (2021-2023), LHC EWK multiboson WG convener (2018-2020)
 - **Combine contact** (SMP and TOP), Computing coordinator and PT Grid T2 admin (2013-2016), **ARC chair** (3x) and member (13x), **CCLC** (6x), MC contact (HIG-Exo, HIG, HWW), CMSDAS facilitator (3x), **Trigger shifter** (online and offline)
- ML in several papers and preprints (various CMS analyses, anomaly detection, design of experiments, neuroscience), badly edited the photo to the right with ML
- Other: PI of NeuroMODE (neuromorphic computing for design of experiments and trigger applications), steering board of MODE (Machine-learning Optimized Design of Experiments), steering group of EUCAIF (European Coalition for AI in Fundamental physics). Past: partner node PI of AMVA4NewPhysics (Advanced MultiVariate Analyses for New Physics)
- Regular courses and lectures on ML; book on Stat and ML near completion (or at least this is what I told the editor :D)



European AI structures

- European initiative for advancing the use of Artificial Intelligence (AI) in Fundamental Physics"
 - **ICTEA** (PV) in the Steering Board!

EUROPEAN COALITION FOR AI IN
FUNDAMENTAL PHYSICS



JENAA

Joint ECFA-NuPECC-APPEC Activities

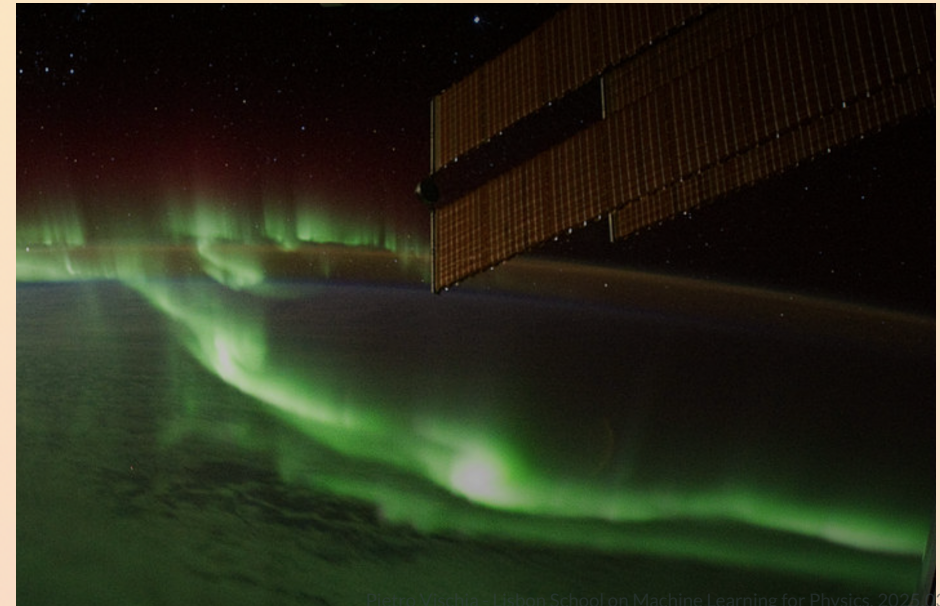
Fundamental → Applied

- Industries (e.g. in Asturias) can profit from AI developed at CERN!
 - **Just contact us!**
- - X/ γ detectors (Xrays, PET)
 - Hadron therapy and proton CT
 - Vacuum technology
 - Cryogenics
 - Art
 - WWW



Many industrial applications of CERN AI technology

- MRI
- GPS
- Satellites
- Solar panels
- Airport security scanners
- Space watch (avoid asteroids)



We have been doing "AI"-assisting since thousands of years



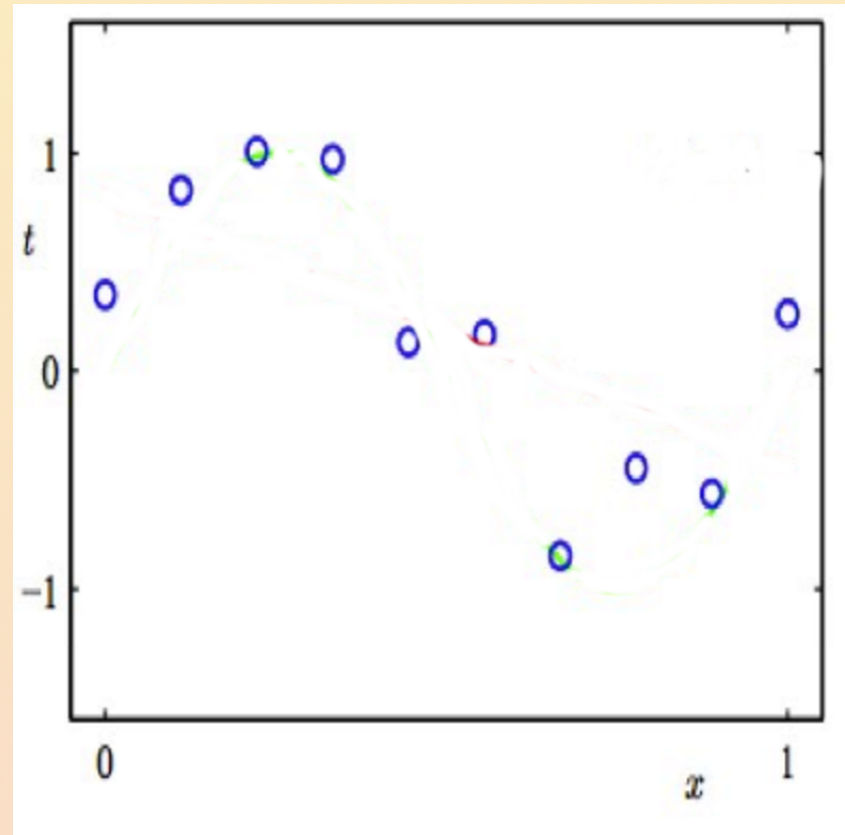
Be prepared for the next thousand!

(maybe not always at CERN :D)

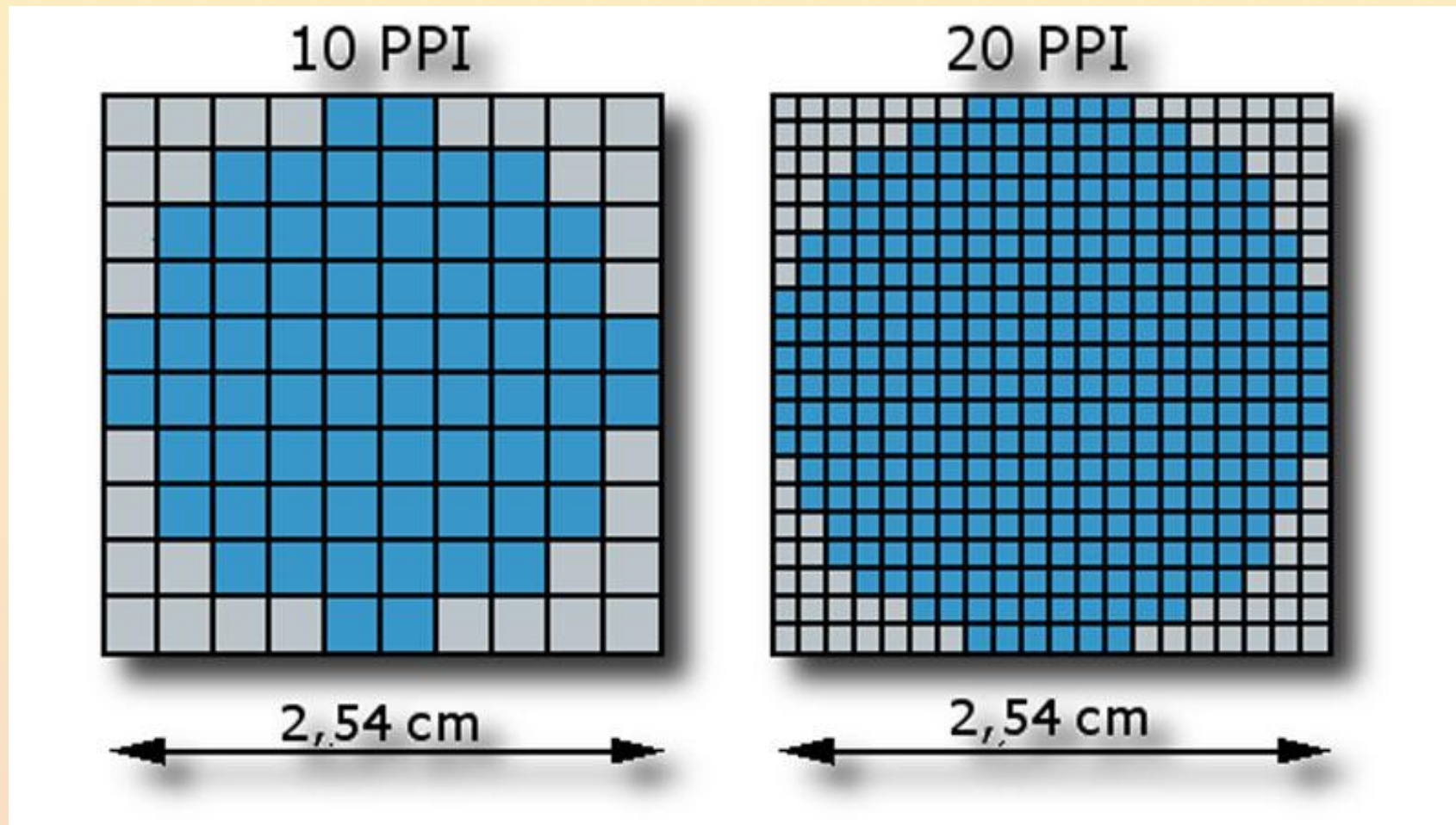
Understanding Data

Vast amounts of data are being generated in many fields, and the statistician's job is to make sense of it all: to extract important patterns and trends, and understand "what the data says." We call this learning from data.

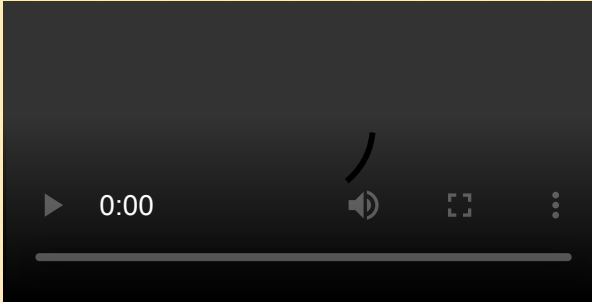
(Hastie, Tibshirani, Friedman, Springer 2017)



Think in Millions of Dimensions

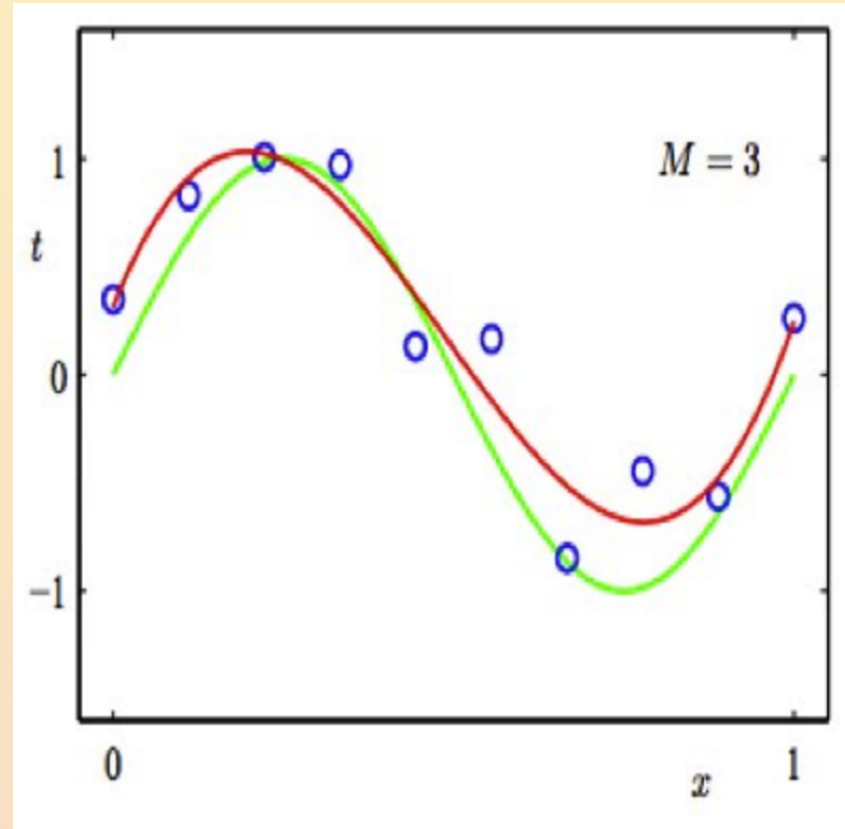


Why does Statistics work?



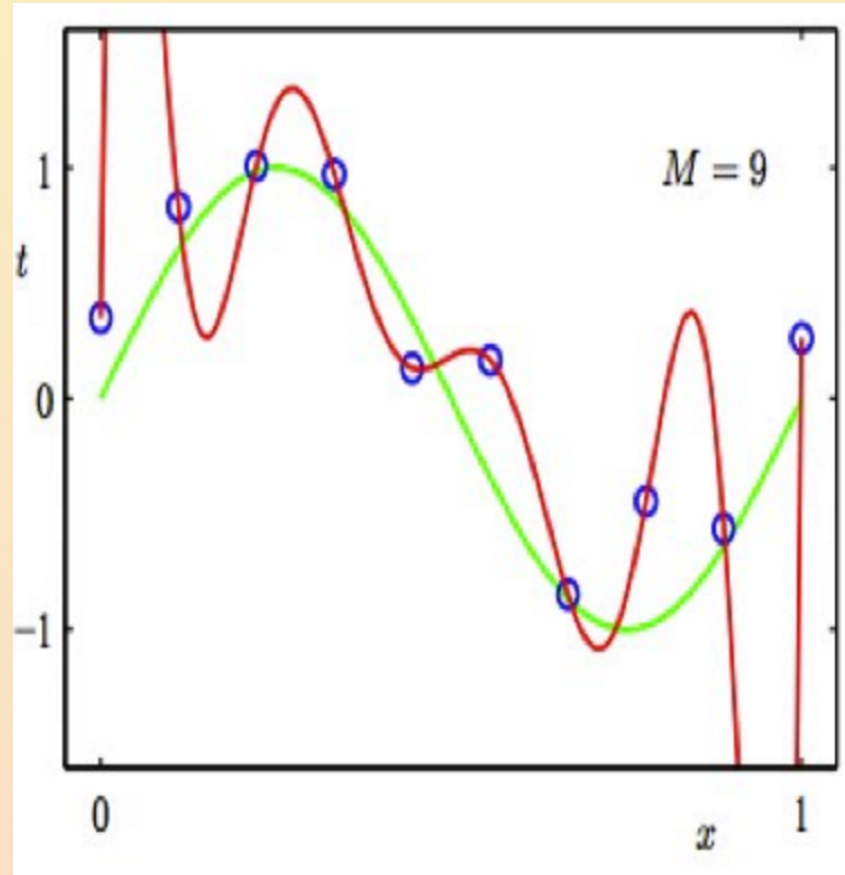
Functions Describe the World

- Interpolation



Sometimes too well

- Generalization



Easy or difficult?

```
[vischia@lxplus797 ~]$ python dump_cpTree.py
Info in <TCanvas::MakeDefCanvas>: created default TCanvas with name c1
*****
* Row * Instance * Lep1_pt.L * Lep2_pt.L * Lep3_pt.L * Lep1_eta. * Lep2_eta. * Lep3_eta. * Lep1_phi. * Lep2_phi. * Lep3_phi. * met.met * met_phi.ml * weight_CP * HTT_score *
*****
* 56350 * 0 * 280.82870 * 93.600341 * 59.040779 * -1.360351 * 0.7332763 * -0.819335 * -2.826171 * 0.7969970 * -3.100097 * 118.03358 * 2.5146484 * 40.820312 * 0.8255668 *
* 56350 * 1 * 280.82870 * 93.600341 * 59.040779 * -1.360351 * 0.7332763 * -0.819335 * -2.826171 * 0.7969970 * -3.100097 * 118.03358 * 2.5146484 * 40.820312 * 0.8255668 *
* 56350 * 2 * 280.82870 * 93.600341 * 59.040779 * -1.360351 * 0.7332763 * -0.819335 * -2.826171 * 0.7969970 * -3.100097 * 118.03358 * 2.5146484 * 40.820312 * 0.8255668 *
* 56350 * 3 * 280.82870 * 93.600341 * 59.040779 * -1.360351 * 0.7332763 * -0.819335 * -2.826171 * 0.7969970 * -3.100097 * 118.03358 * 2.5146484 * 40.820312 * 0.8255668 *
* 79791 * 0 * 67.791183 * 42.294036 * 17.310911 * -1.846923 * 1.4458007 * -0.762207 * -0.628540 * -2.910644 * 2.9912109 * 8.8895807 * -1.773925 * 94.398437 * -99 *
*****
=> 5 selected entries
-----
* Row * Instance * Lep1_pt.L * Lep2_pt.L * Lep3_pt.L * Lep1_eta. * Lep2_eta. * Lep3_eta. * Lep1_phi. * Lep2_phi. * Lep3_phi. * met.met * met_phi.ml * weight_CP * HTT_score *
*****
* 58751 * 0 * 86.053443 * 32.593345 * 13.077508 * -1.346435 * 2.0512695 * -0.503906 * -0.392944 * -2.925781 * 0.9309082 * 17.544691 * 0.1735229 * 3.478e-08 * 0.9726203 *
* 58751 * 1 * 86.053443 * 32.593345 * 13.077508 * -1.346435 * 2.0512695 * -0.503906 * -0.392944 * -2.925781 * 0.9309082 * 17.544691 * 0.1735229 * 3.478e-08 * 0.9726203 *
* 58751 * 2 * 86.053443 * 32.593345 * 13.077508 * -1.346435 * 2.0512695 * -0.503906 * -0.392944 * -2.925781 * 0.9309082 * 17.544691 * 0.1735229 * 3.478e-08 * 0.9726203 *
* 58751 * 3 * 86.053443 * 32.593345 * 13.077508 * -1.346435 * 2.0512695 * -0.503906 * -0.392944 * -2.925781 * 0.9309082 * 17.544691 * 0.1735229 * 3.478e-08 * 0.9726203 *
* 58751 * 4 * 86.053443 * 32.593345 * 13.077508 * -1.346435 * 2.0512695 * -0.503906 * -0.392944 * -2.925781 * 0.9309082 * 17.544691 * 0.1735229 * 3.478e-08 * 0.9726203 *
* 58751 * 5 * 86.053443 * 32.593345 * 13.077508 * -1.346435 * 2.0512695 * -0.503906 * -0.392944 * -2.925781 * 0.9309082 * 17.544691 * 0.1735229 * 3.478e-08 * 0.9726203 *
*****
=> 6 selected entries
```

Signal

Background

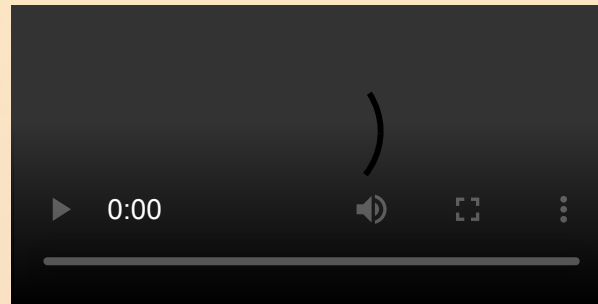
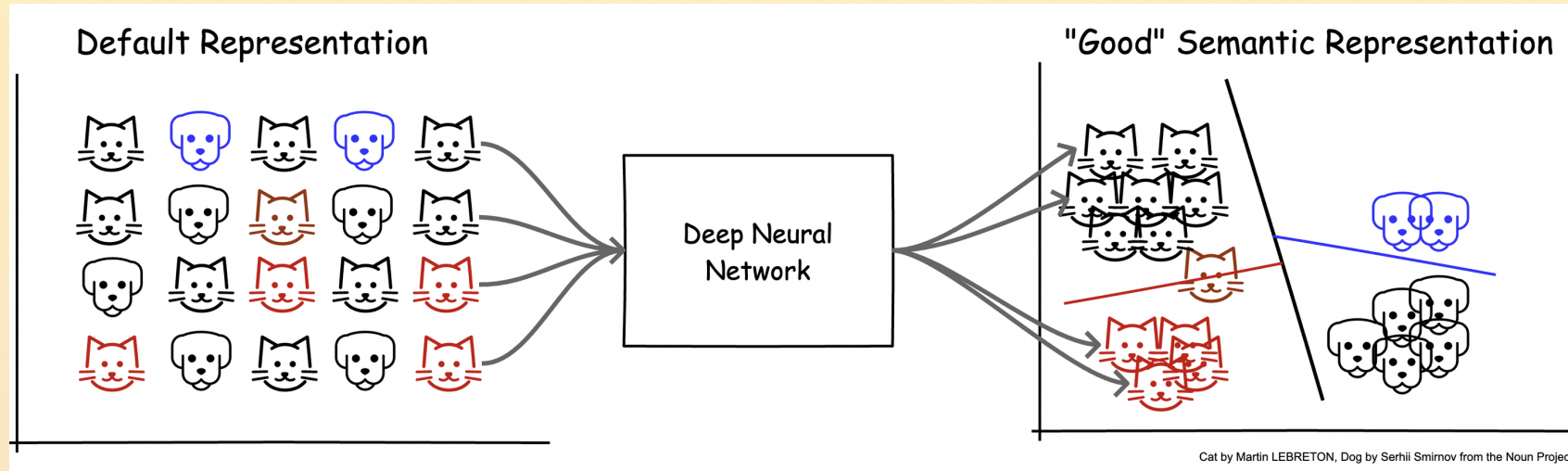
Easy or difficult?



Mapping Improves Understanding

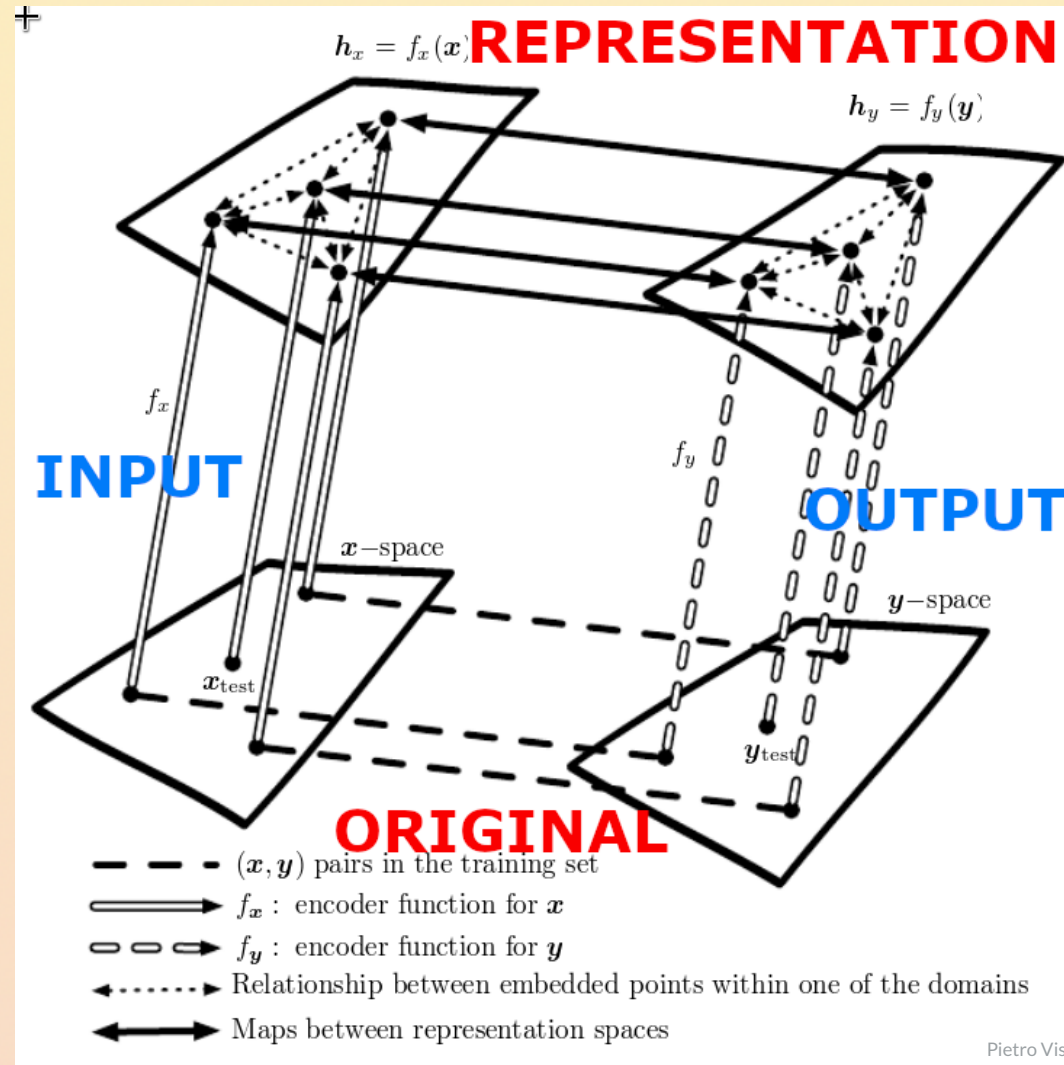


Representations Make Tasks Easier



Learn Representations

- Like AdS/CFT, but actually demonstrated 🤪

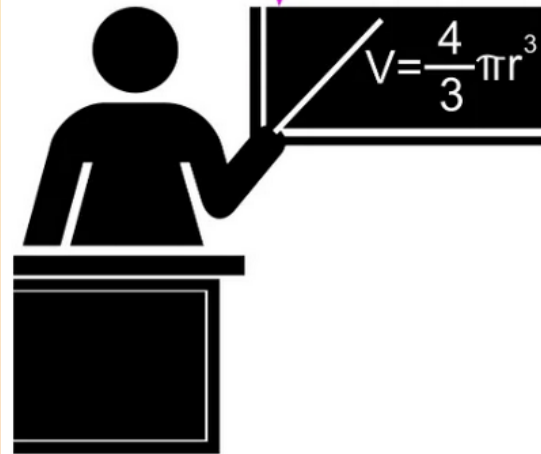


Learning from data:

the mathematical formalism

Learn in different ways (today: supervised learning)

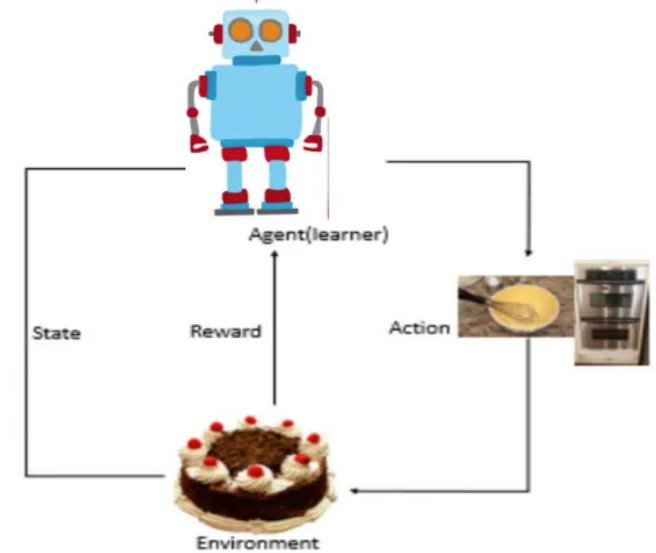
Machine Learning



Supervised Learning



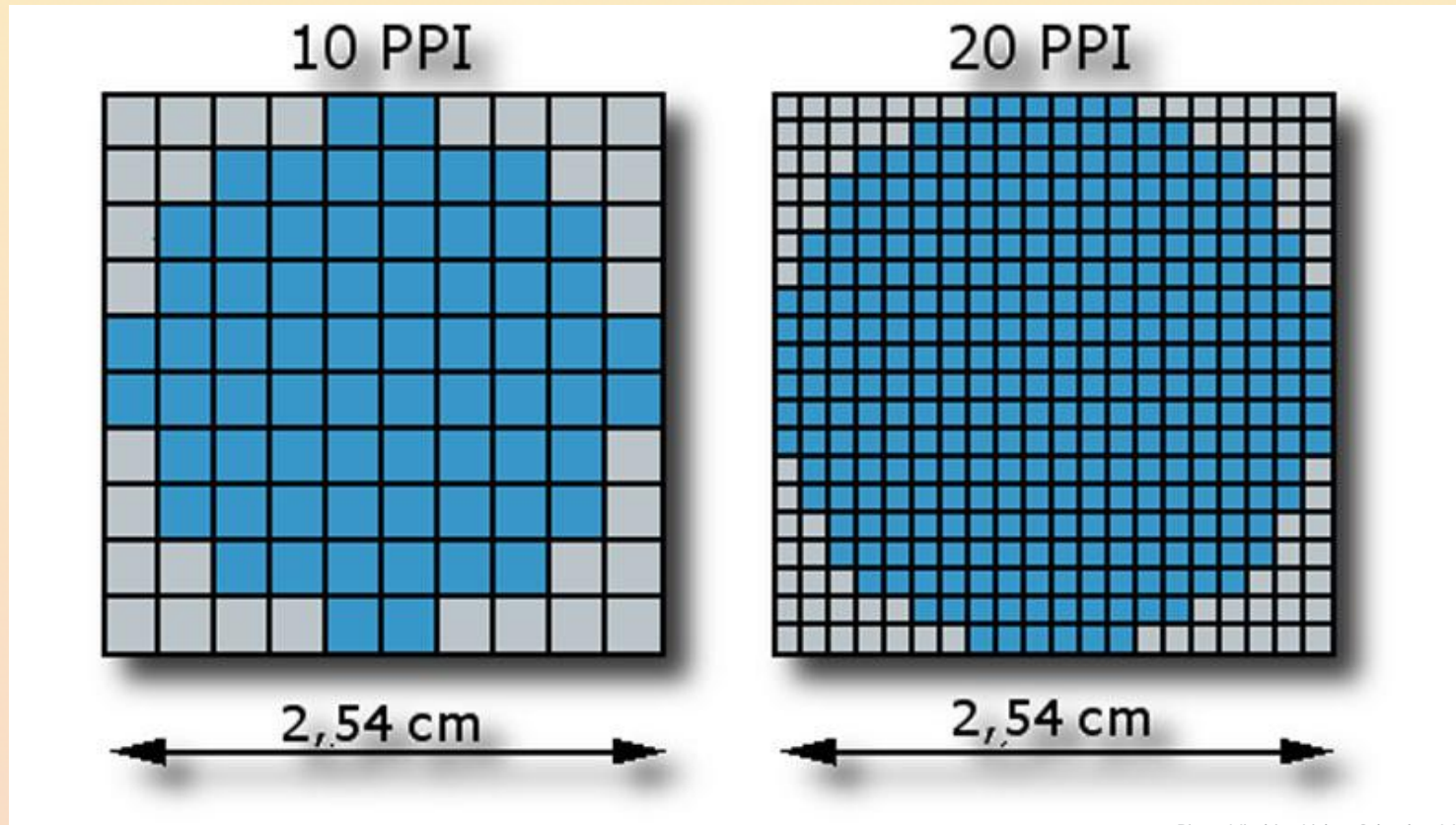
Unsupervised Learning



Reinforcement Learning

Input space

- \mathcal{X} : a high-dimensional input space
 - The challenges come from the high dimensionality!
- If all dimensions are real-valued, \mathbb{R}^d
 - For square images of side \sqrt{d} , space is $\mathcal{X} = \mathbb{R}^d$, and $d \sim \mathcal{O}(10^6)$



Data probability distribution

- ν : unknown data probability distribution
 - We can sample from it to obtain an arbitrary amount of data points
 - We are not allowed to use any analytic information about it in our computations

The target function

- $f^* : \mathcal{X} \rightarrow \mathbb{R}$, unknown target function
 - In case of multidimensional output to a vector of dimension k , $f^* : \mathcal{X} \rightarrow \mathbb{R}^k$
 - Some loose assumptions (e.g. square-integrable with respect to the ν measure, i.e. finite moments, bounded...)

The loss functional

- $L[f] = \mathbb{E}_\nu \left[l\left(f(x), f^*(x)\right) \right]$
 - The metric that tells us how good our predictions are
- The function $l(\cdot, \cdot)$ is a given expression, e.g. regression loss, logistic loss, etc
 - In this lecture, typically it is the L^2 norm: $\|f - f^*\|_{L^2(\mathcal{X}, \nu)}$

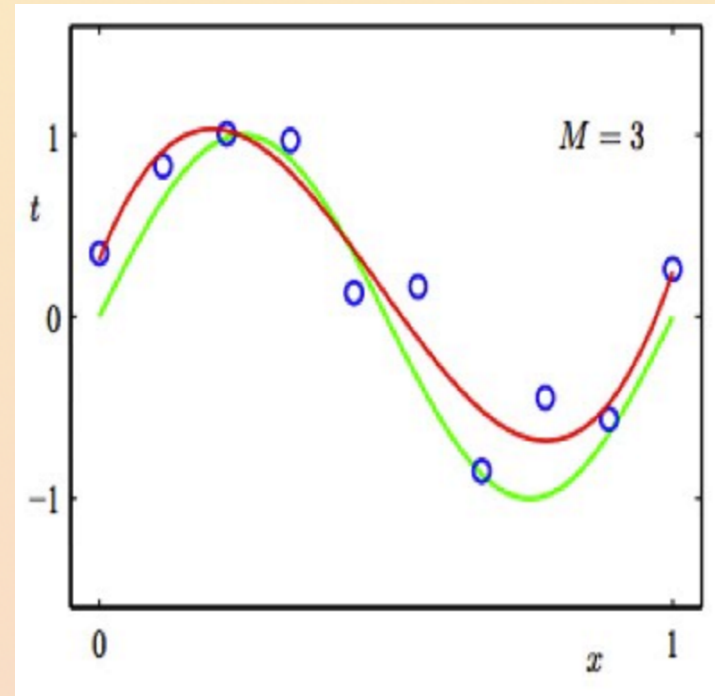
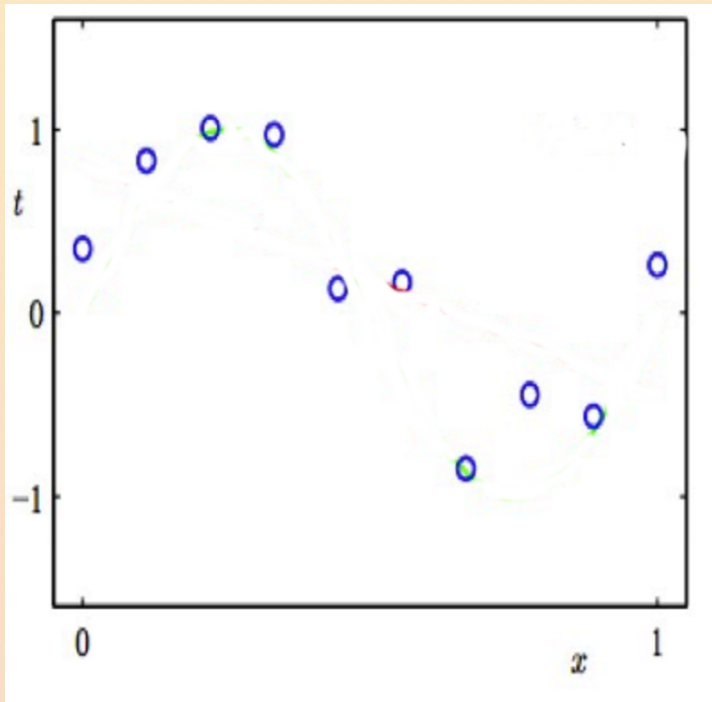
Loss function comes from inference

- Decision-theoretic approach (C.P. Robert, "The Bayesian Choice")
 - \mathcal{X} : observation space
 - Θ : parameter space
 - \mathcal{D} : decision (action) space
- **Statistical inference** take a decision $d \in \mathcal{D}$ related to parameter $\theta \in \Theta$ based on observation $x \in \mathcal{X}$, under $f(x|\theta)$
 - Typically, d consists in estimating $h(\theta)$ accurately

$$U(\theta, d) = \mathbb{E}_{\theta, d} [U(r)]$$

Learning goal

- Goal: predict f^* from a finite i.i.d. sample of points sampled from ν
- Sample $\{x_i, f^*(x_i)\}_{i=1, \dots, n}, x_i \sim \nu$
 - For each of the points x_i , we know the value of the unknown function (our true labels)
 - We want to **interpolate** for any arbitrary x inbetween the labelled x_i ...
 - ...in million of dimensions!

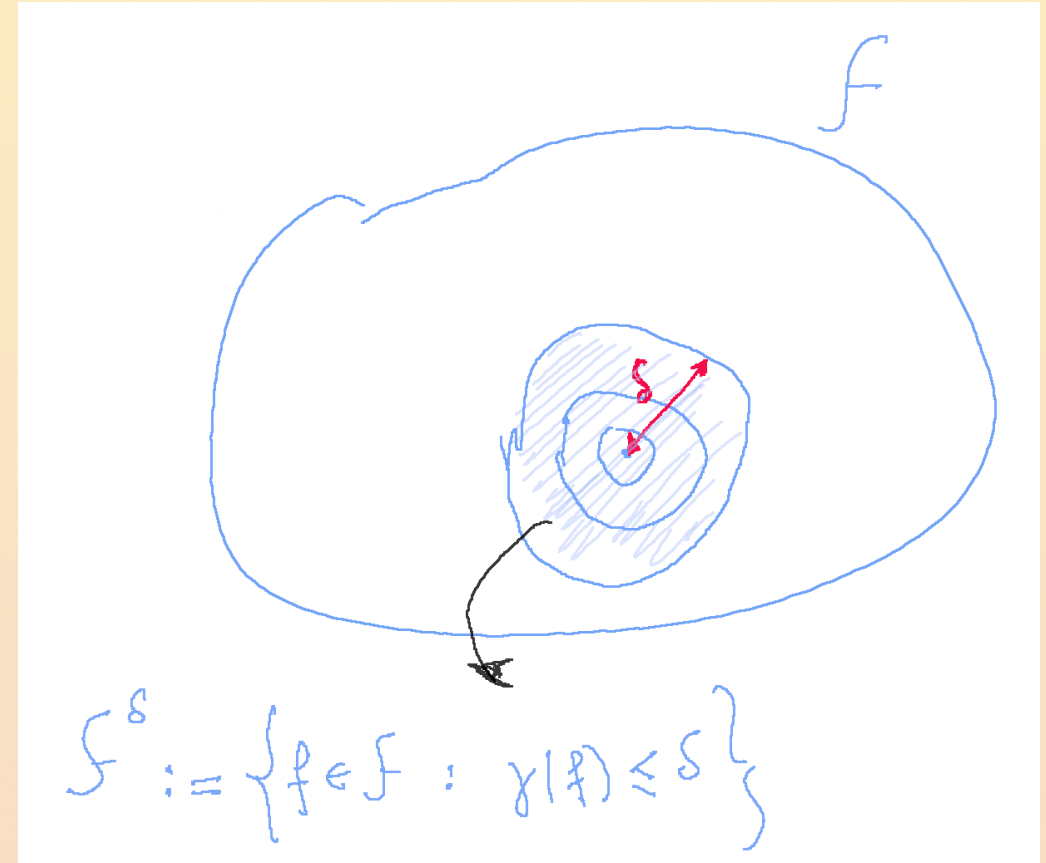


The space of possible solutions

- The space of possible functional solutions is vast: $\mathcal{F} \subseteq \{f : \mathcal{X} \rightarrow \mathbb{R}\}$ (hypothesis class)
- We need a notion of complexity to "organize" the space
- $\gamma(f), f \in \mathcal{F}$: **complexity** of f
 - It can for example be the norm, i.e. we can augment the space \mathcal{F} with the norm

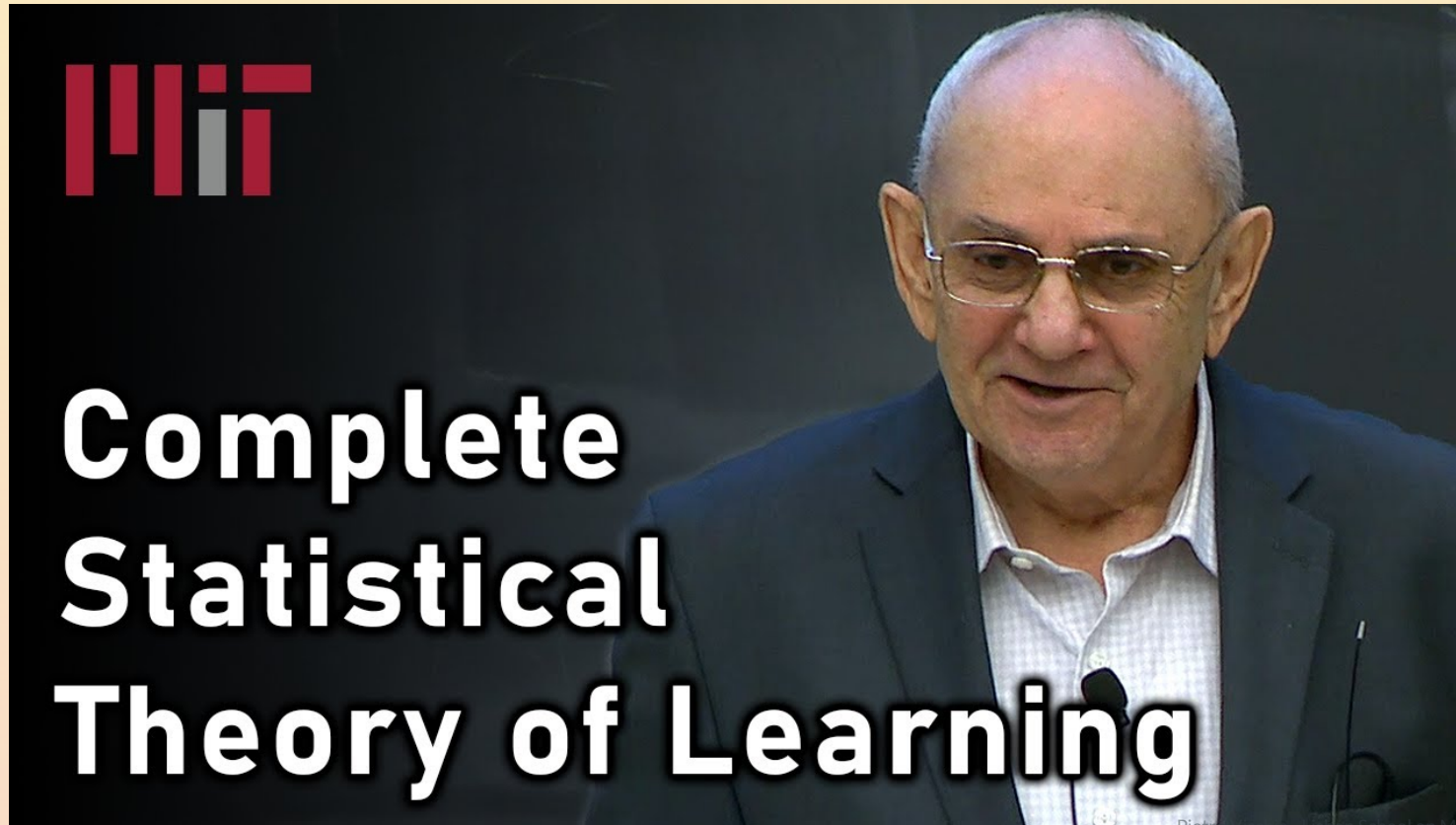
Organizing the space

- When the complexity is defined via the norm, \mathcal{F} is highly organized: Banach space!
- The simplest function according to the norm criterion is the 0 function
- If we increase the complexity by increasing the norm, we obtain **convex balls** $\{f \in \mathcal{F}; \gamma(f) \leq \delta\} =: \mathcal{F}^\delta$
- Convex minimization is considerably easier than non-convex minimization



Empirical Risk Minimization

- For each element of \mathcal{F} , a measure of how well it's interpolating the data
- Empirical risk: $\hat{L}(f) = \frac{1}{n} \sum_{i=1}^n |f(x_i) - f^*(x_i)|^2$
 - $|\cdot|$ is the empirical loss. If it's the norm, then $\hat{L}(f)$ is the empirical Mean Square Error
 - If you find an analogy with least squares method, it's because for one variable it's exactly that!



Formalizing the minimization of a functional in a given space}

- **Constraint form:** $\min_{f \in \mathcal{F}^\delta} \hat{L}(f)$.
 - Not trivial
- **Penalized form:** $\min_{f \in \mathcal{F}} \hat{L}(f) + \lambda \gamma(f)$.
 - More typical
 - λ is the price to pay for more complex solutions. Depends on the complexity measure
- **Interpolant form:** $\min_{f \in \mathcal{F}} \gamma(f)$ s.t. $\hat{L}(f) = 0 \iff f(x_i) = f^*(x_i) \quad \forall i$
 - In ML, most of the times there is no noise, so $f(x_i)$ is exactly the value we expect there (i.e. we really know that x_i is of a given class, without any uncertainty)

Are the three forms equivalent?

- The interpolant form exploits the no-noise assumption (*"give me the least complex elements in \mathcal{F} that interpolates"*)
- These forms are **not completely equivalent**. The penalized form to be solved requires averaging a full set of penalized forms, so it's not completely equivalent
- There is certainly an implicit correspondence between δ and λ
 - (the larger λ , the smaller δ and viceversa)

The Fundamental Theorem of Machine Learning

- We want to relate the result of the empirical risk minimization (ERM) with the prediction
 - Let's use the constraint form
- Let's assume we have solved the ERM at a precision ϵ (we are ϵ -away from...). We then have $\hat{f} \in \mathcal{F}^\delta$ such that $\hat{L}(\hat{f}) \leq \epsilon + \min_{f \in \mathcal{F}^\delta} \hat{L}(f)$
- How good is \hat{f} at predicting f^* ? In other words, what's the true loss?
 - Can use the triangular inequality

$$L(\hat{f}) - \inf_{f \in \mathcal{F}} L(f) \leq \inf_{f \in \mathcal{F}^\delta} L(f) - \inf_{f \in \mathcal{F}} L(f)$$

$$+ 2 \sup_{f \in \mathcal{F}^\delta} |L(f) - \hat{L}(f)|$$

+ ϵ

Approximation error

(how appropriate is my measure of complexity)

Statistical error

(having the empirical loss instead of the true loss)

Optimization error

The Error Market

- The minimization is regulated by the parameter δ (the size of the ball in the space of functions)
- Changing δ results in a **tradeoff** between the different errors
 - Very small δ makes the statistical error blow up
- We are better at doing convex optimization (easier to find minimum), but even then the optimization error ϵ will not be negligible
 - ϵ : how much are you willing to spend in resources to minimize $\hat{L}(f)$
 - We kind of control it!
 - If the other errors are smaller than ϵ , then it makes sense to spend resources to decrease it
 - Otherwise, don't bother

The big questions

- **Approximation**: we want to design "good" spaces \mathcal{F} to approximate f^* in high-dimension
 - Rather profound problem, on which we still struggle
- **Optimization**: how to design algorithms to solve the ERM in general
 - We essentially have ONE answer
 - **Gradient Descent!**

The Curse of Dimensionality

- How many samples do we need to estimate f^* , depending on assumptions on its regularity?

The Curse of Dimensionality

- How many samples do we need to estimate f^* , depending on assumptions on its regularity?
 - f^* constant \rightarrow need only 1 sample
 - f^* linear \rightarrow need d samples
- Space of functionals is $\mathcal{F} = \{f : \mathbb{R}^d \rightarrow \mathbb{R}; f(x) = \langle x, \theta \rangle\} \simeq \mathbb{R}^d$ (isomorphic)
 - It's essentially like solving a system of linear equations for the linear form $\langle x, \theta^* \rangle$
 - d equations, d degrees of freedom
- The reason why it's so easy is that linear functions are regular at a global level
 - Knowing the function locally tells us automatically the properties everywhere

Locally linear functions

- Assume f^* locally linear, i.e. f^* is Lipschitz
 - $|f^*(x) - f^*(y)| \leq \beta \|x - y\|$
 - $Lip(f^*) = \inf \{ \beta; |f^*(x) - f^*(y)| \leq \beta \|x - y\| \text{ is true} \}$
 - $Lip(f^*)$ is a measure of smoothness
- Space of functionals that are Lipschitz: $\mathcal{F} = \{ f : \mathbb{R}^d \rightarrow \mathbb{R}; f \text{ is Lipschitz} \}$
- We want a normed space to parameterize complexity, so we convert to a Banach space
 - $\gamma(f) := \max(Lip(f), |f|_\infty)$
 - The parameterization of complexity is the Lipschitz constant

Formalization of the prediction problem

- $\forall \epsilon > 0$, find $f \in \mathcal{F}$ such that $\|f - f^*\| \leq \epsilon$ from n i.i.d. samples
 - n : sample complexity, "how many more samples to I need to make the error a given amount of times smaller"
- If f^* is Lipschitz, it can be demonstrated that $n \sim \epsilon^{-d}$
 - Upper bound: approximate f with its value in the closest of the sampled data points, find out expected error $\sim \epsilon^2$, upper bound is exponential
 - Lower bound: maximum discrepancy (the worst case scenario): unless you sample exponential number of data points, knowing $f(x_i)$ for all of them doesn't let you well approximate outside

PAC (Probably Approximately Correct) learning

DEFINITION 3.1 (PAC Learnability) A hypothesis class \mathcal{H} is PAC learnable if there exist a function $m_{\mathcal{H}} : (0, 1)^2 \rightarrow \mathbb{N}$ and a learning algorithm with the following property: For every $\epsilon, \delta \in (0, 1)$, for every distribution \mathcal{D} over \mathcal{X} , and for every labeling function $f : \mathcal{X} \rightarrow \{0, 1\}$, if the realizable assumption holds with respect to $\mathcal{H}, \mathcal{D}, f$, then when running the learning algorithm on $m \geq m_{\mathcal{H}}(\epsilon, \delta)$ i.i.d. examples generated by \mathcal{D} and labeled by f , the algorithm returns a hypothesis h such that, with probability of at least $1 - \delta$ (over the choice of the examples), $L_{(\mathcal{D}, f)}(h) \leq \epsilon$.

- ϵ ("approximately correct"): how far from optimality the model is
- δ ("probably"): how likely the model is to meet the accuracy requirement
- $m_{\mathcal{H}}$ determines sample complexity (how many examples to guarantee PAC?)

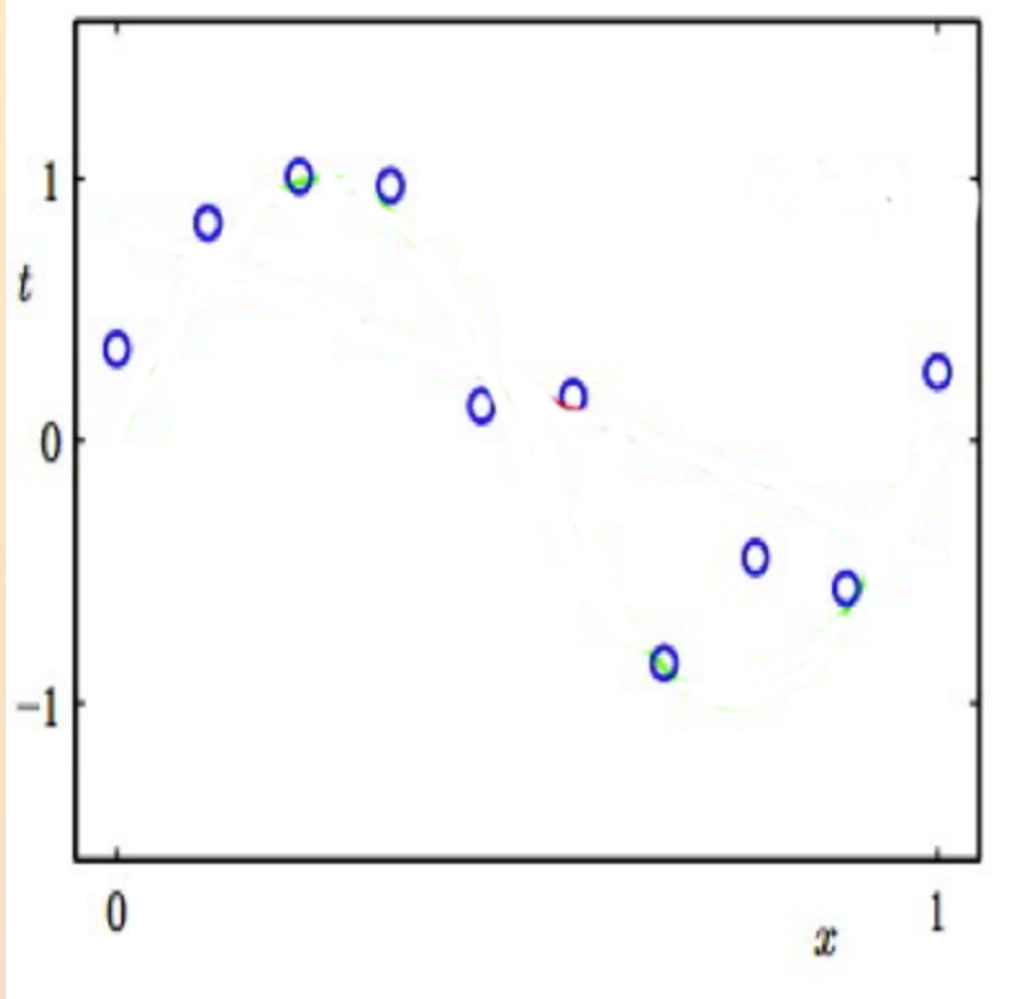
COROLLARY 3.2 *Every finite hypothesis class is PAC learnable with sample complexity*

$$m_{\mathcal{H}}(\epsilon, \delta) \leq \left\lceil \frac{\log(|\mathcal{H}|/\delta)}{\epsilon} \right\rceil.$$

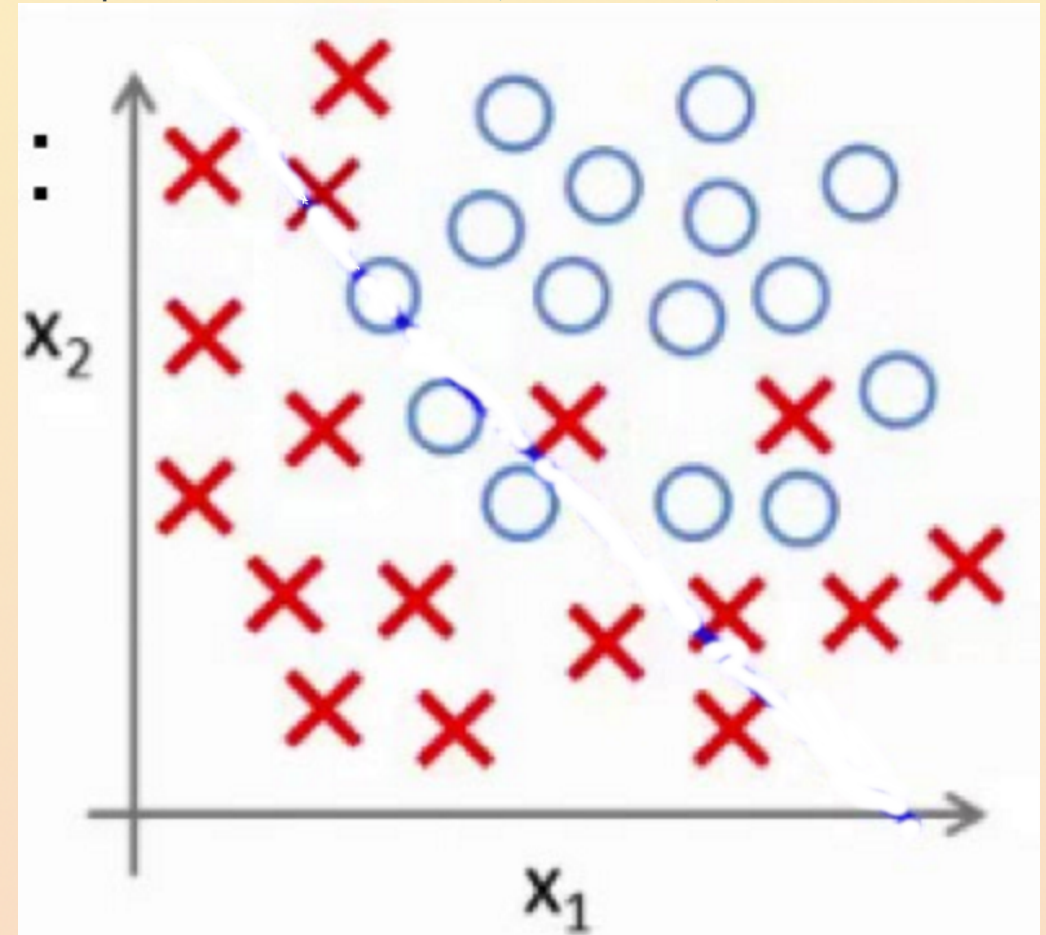
Enough of the math?

What's the best function

To describe the data points? ([regression](#))

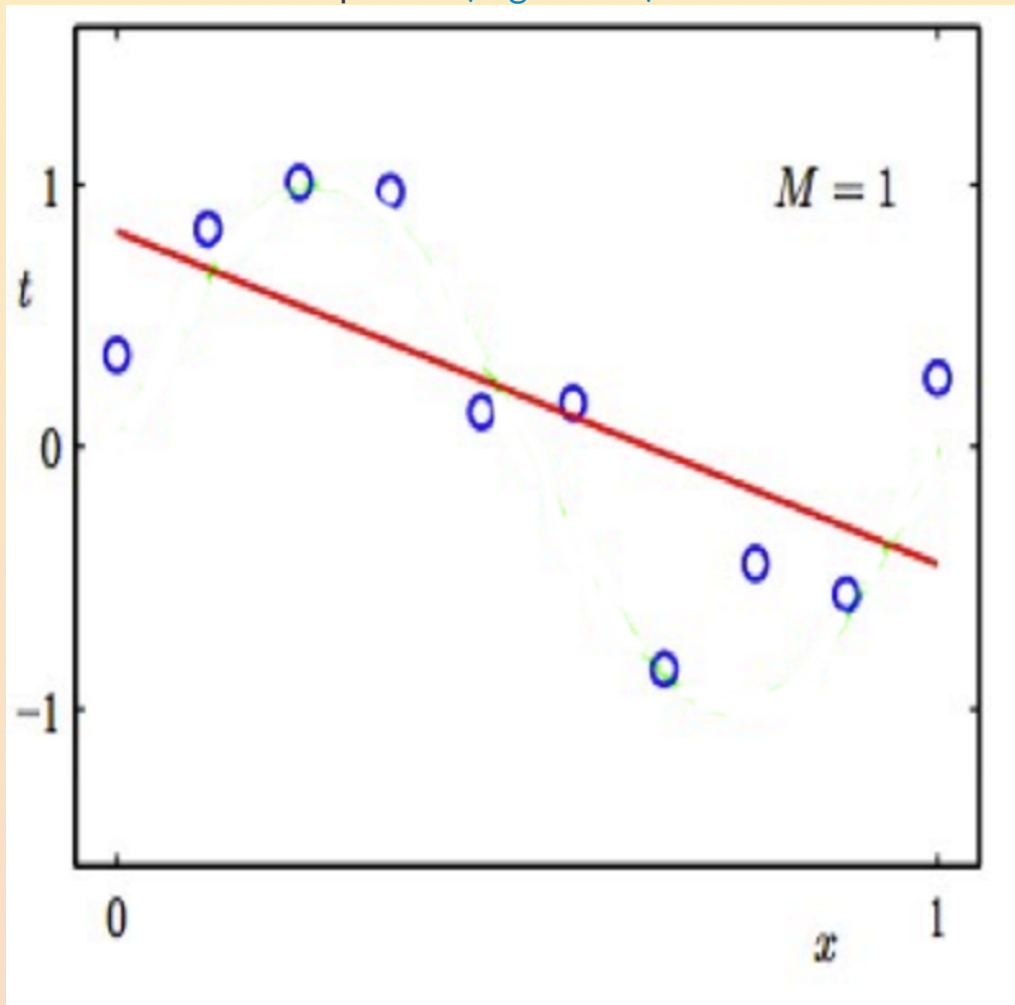


To separate into two classes? ([classification](#))

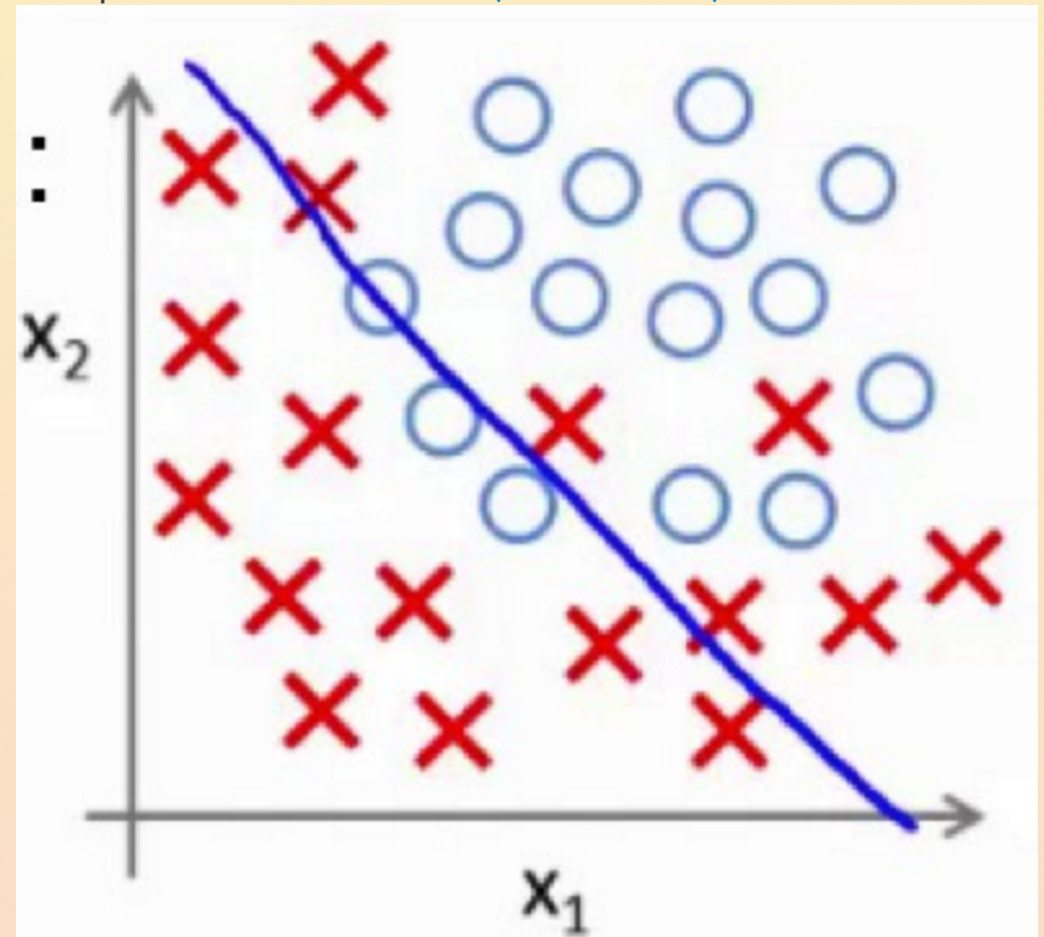


What's the best function

To describe the data points? (regression)

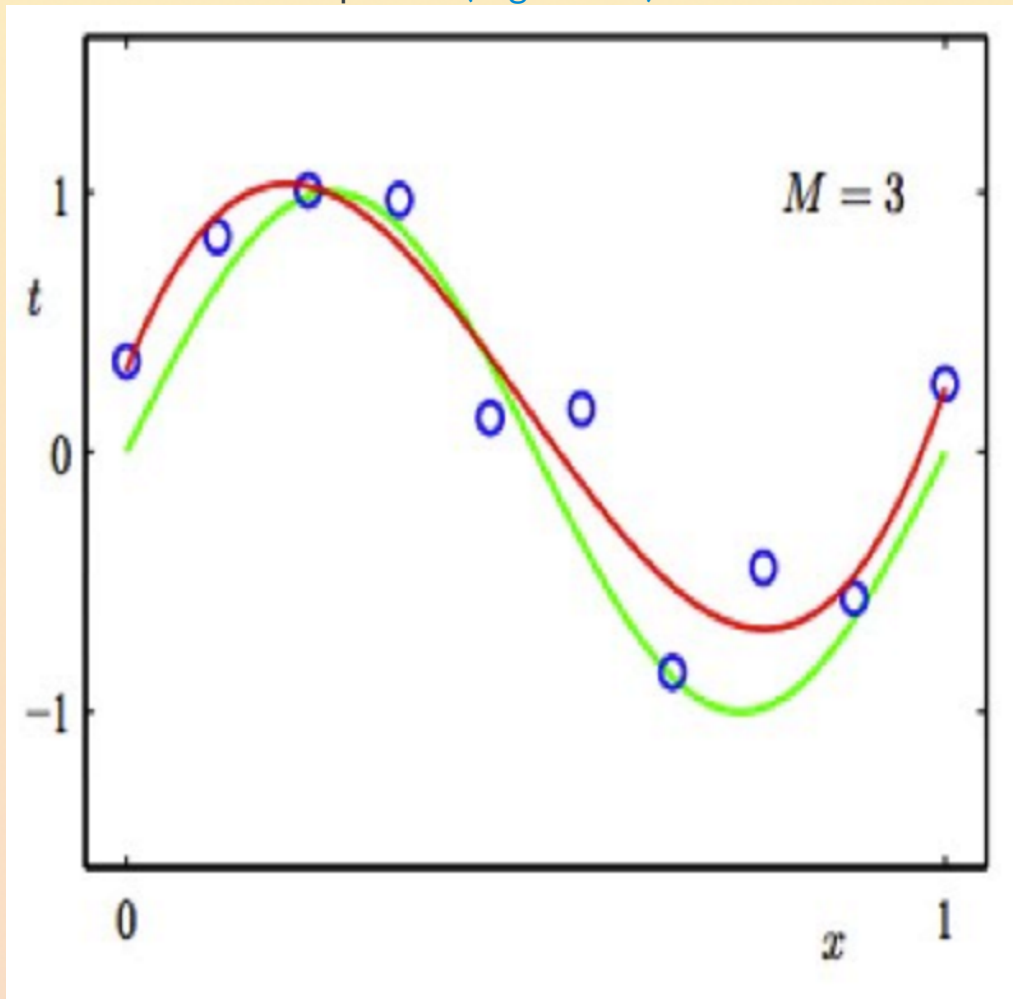


To separate into two classes? (classification)

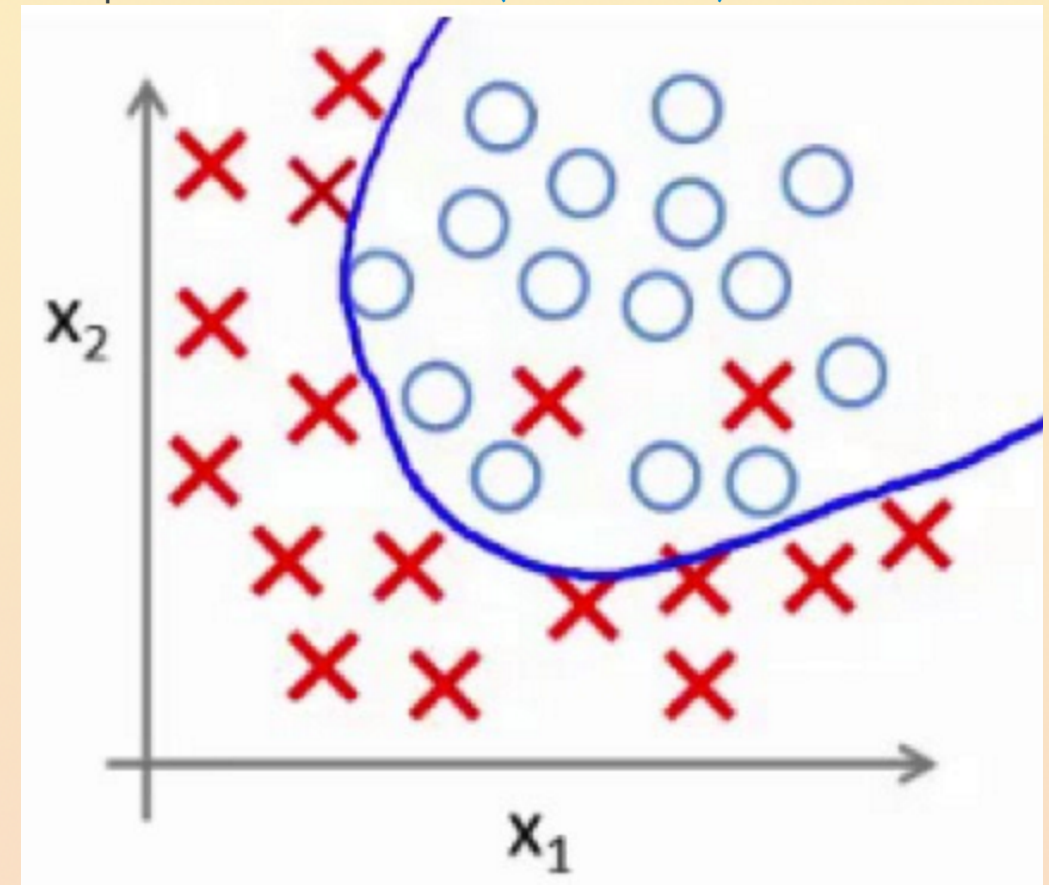


What's the best function

To describe the data points? (regression)

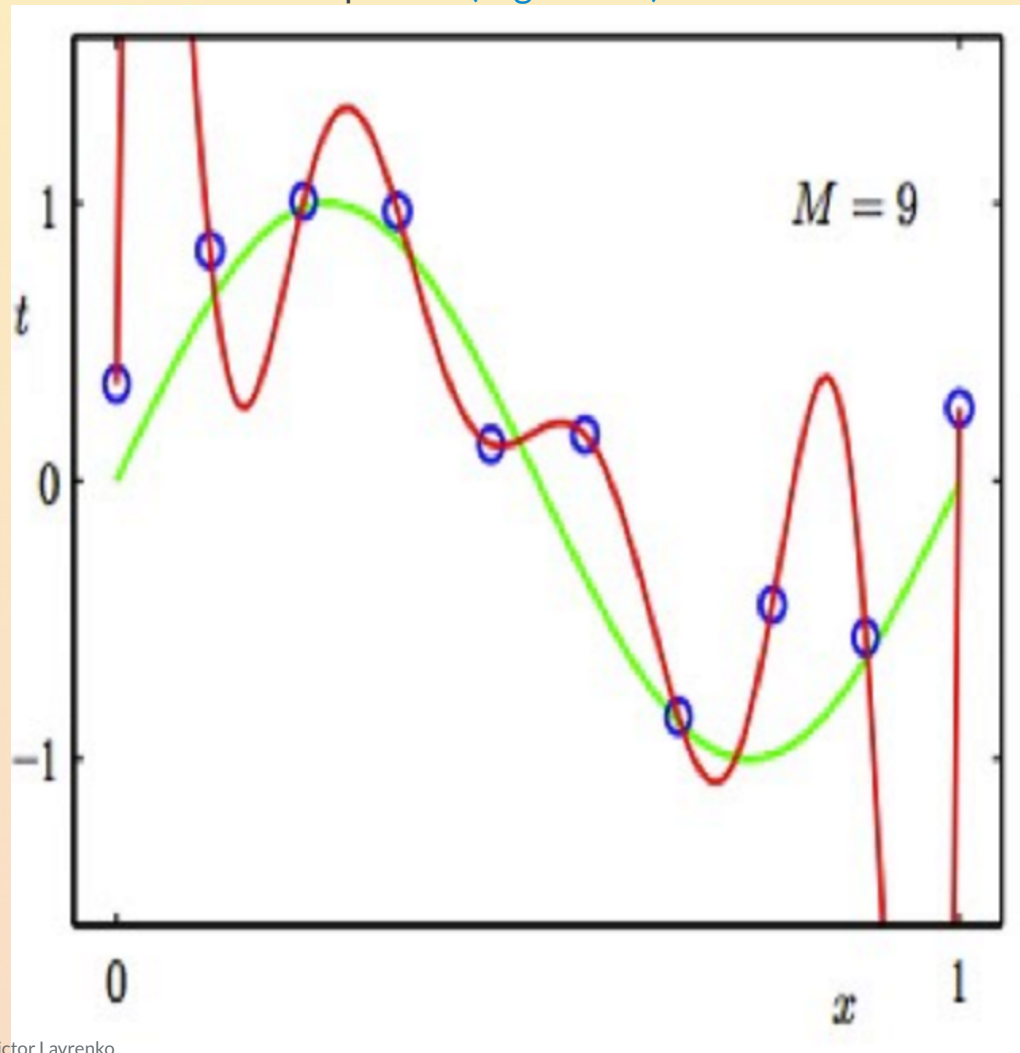


To separate into two classes? (classification)

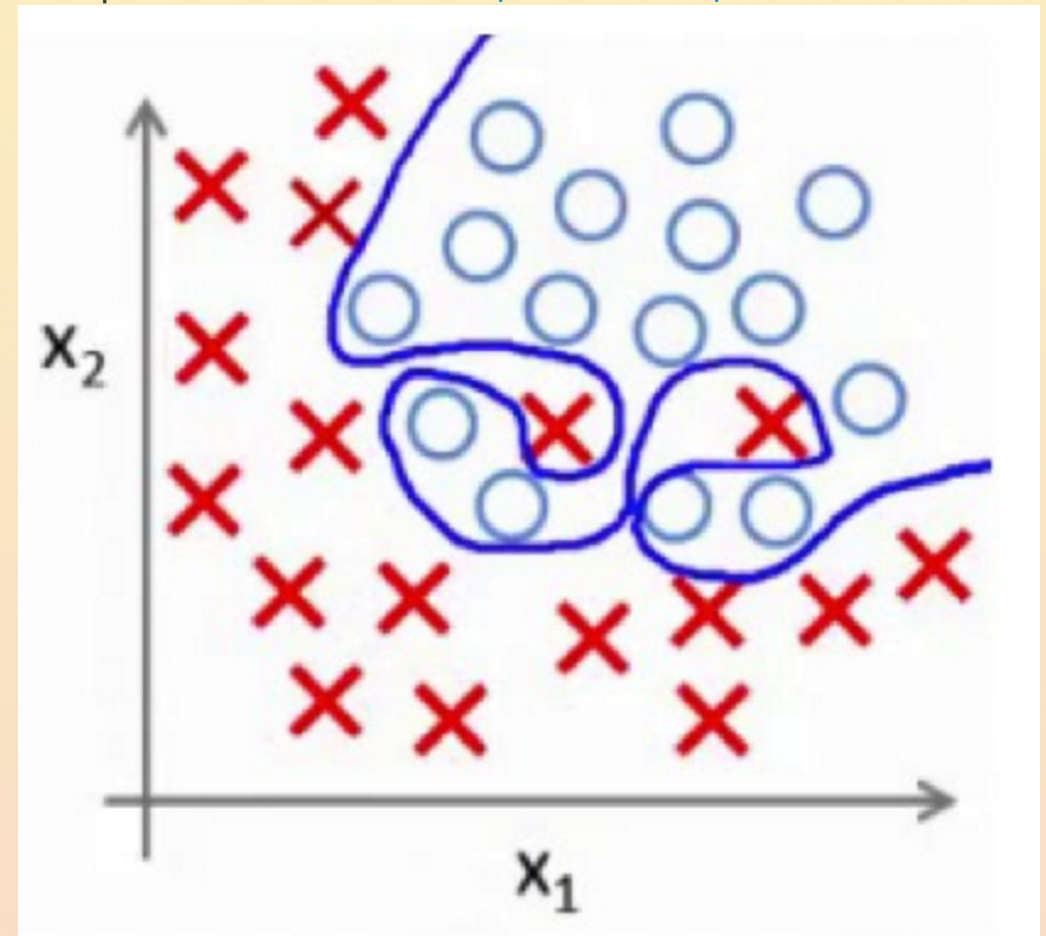


What's the best function

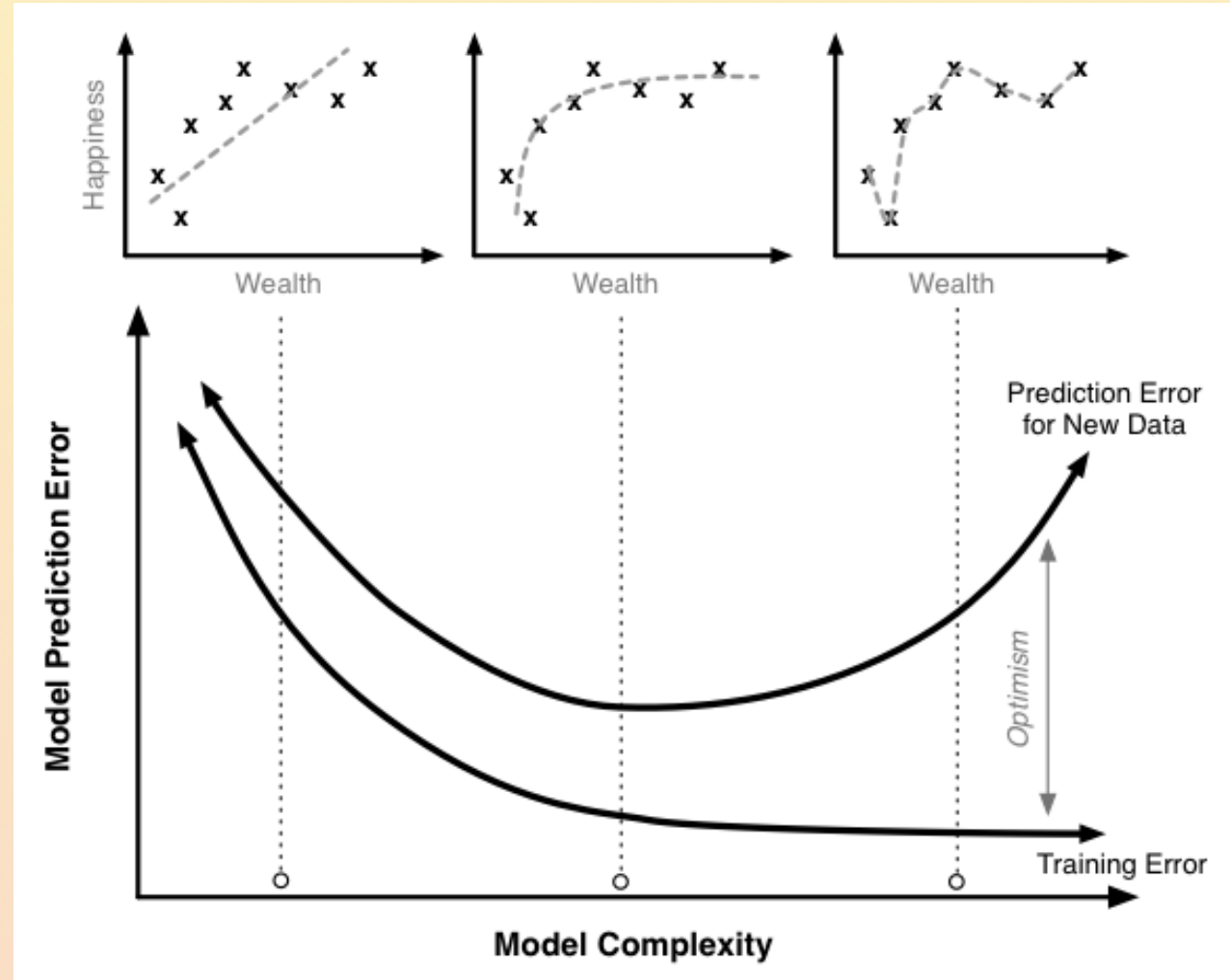
To describe the data points? (regression)



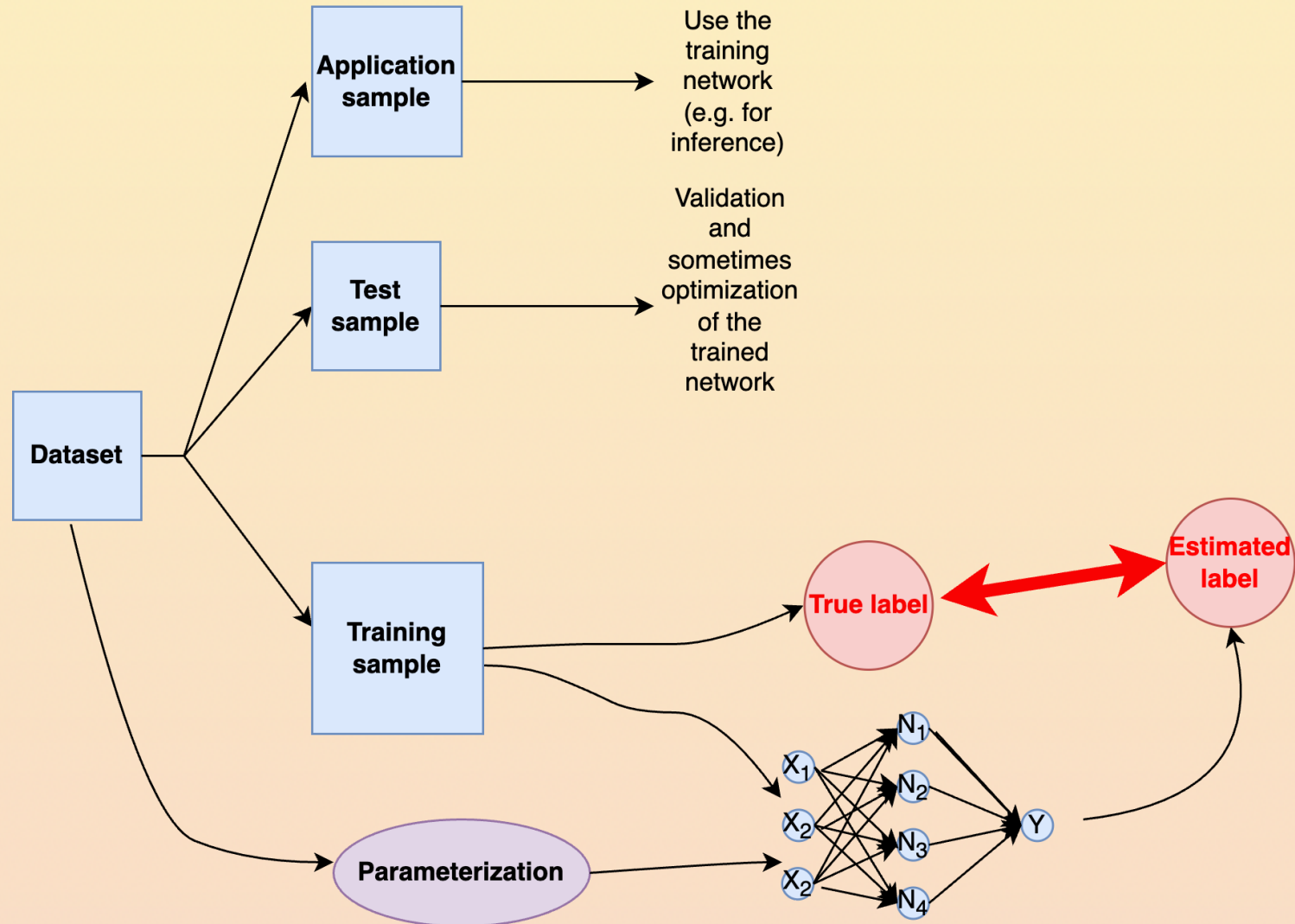
To separate into two classes? (classification)



Avoid overtraining



Training a model



Before looking into ML models...

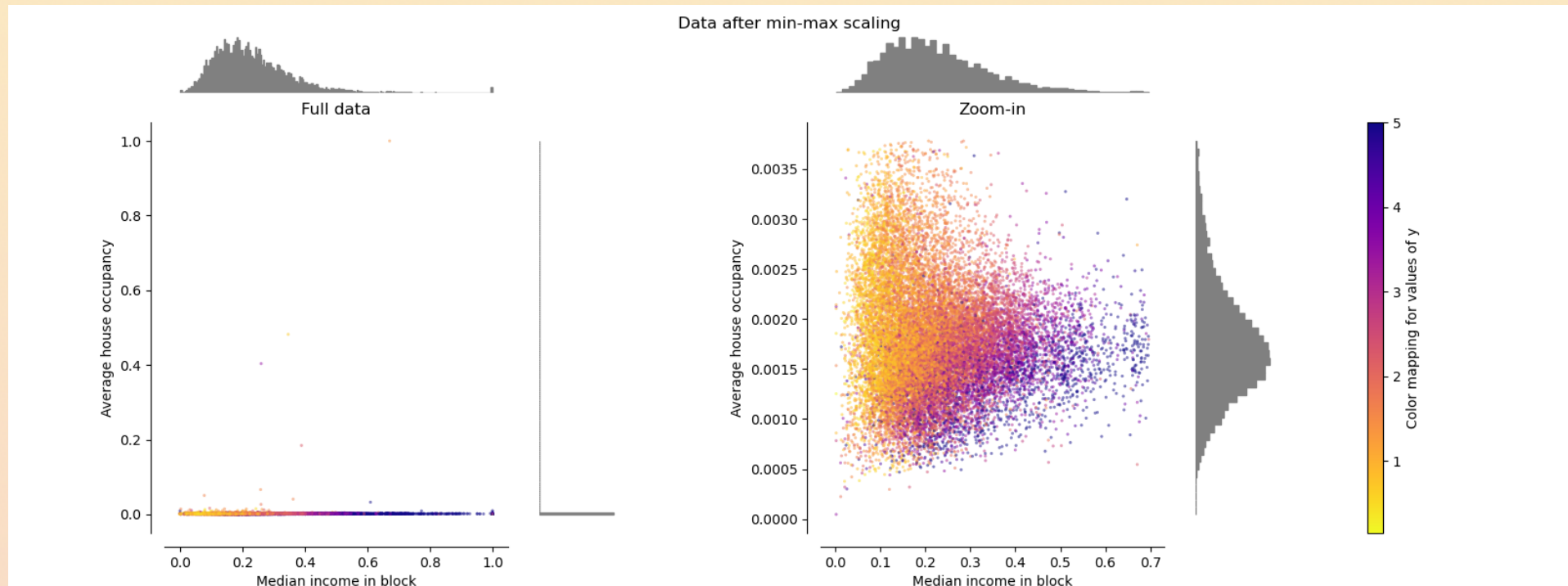
Before looking into ML models... Look at the data!!!

Data preparation: MinMax scaling

$$k = \frac{(x - \min(x))}{\text{abs}(\max(x) - \min(x))}$$

$$z = k * (\text{Max} - \text{Min}) + \text{Min}$$

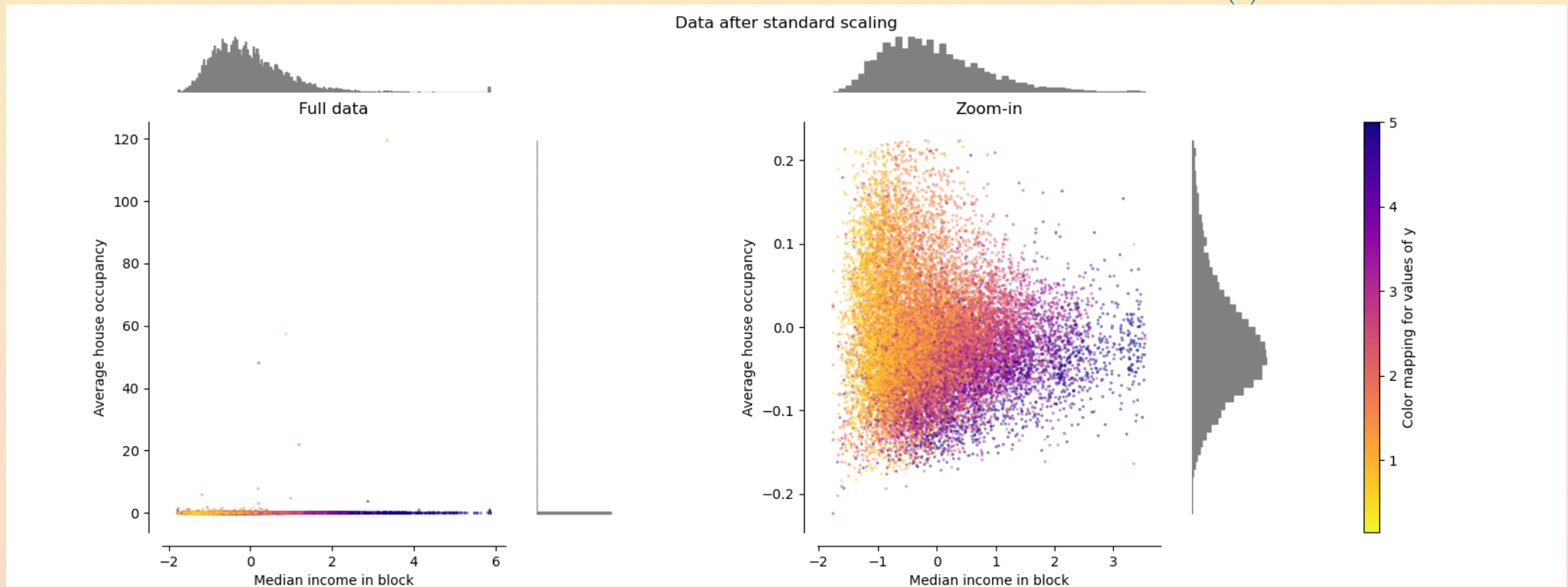
- Very sensitive to outliers (it scales them linearly)
- Scale to different variance and different mean



Data preparation: standardization

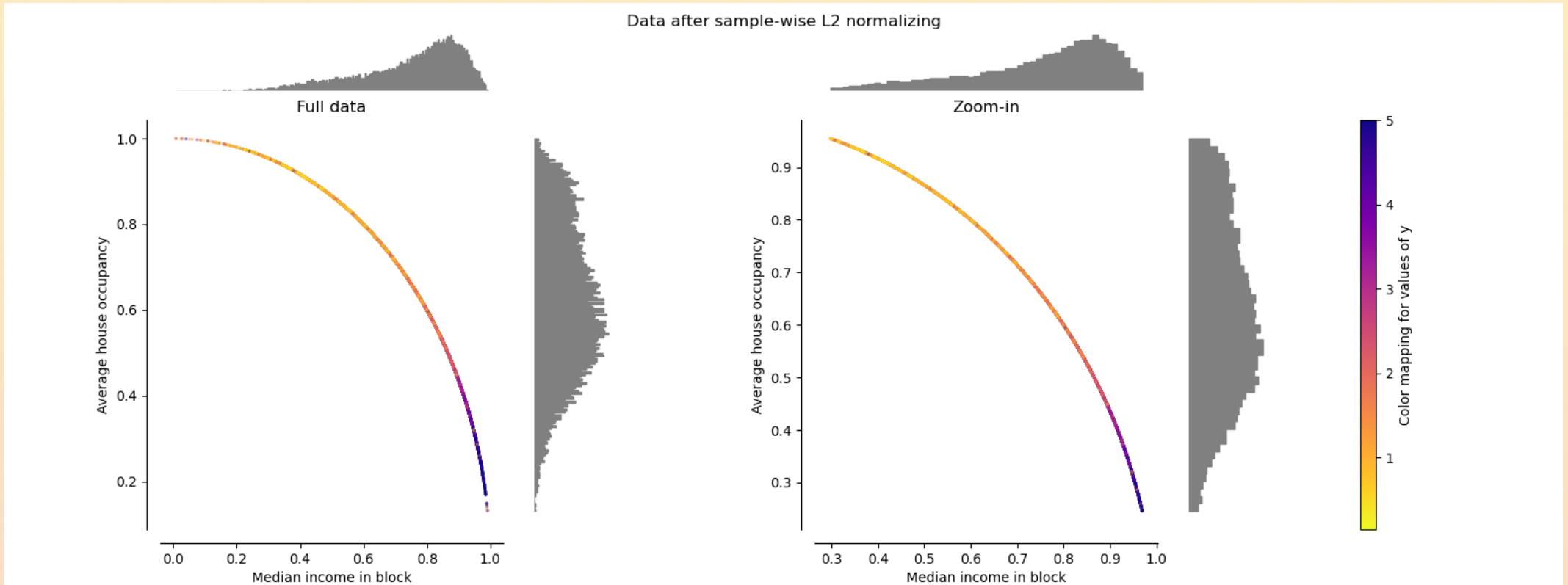
$$z = \frac{(x - \text{mean}(x))}{\text{var}(x)}$$

- Direct comparison between weights assigned to different features
- Easier, more effective numerical minimization, but still sensitive to outliers
- Scale to same variance and same mean (can also scale to same variance but different mean $z = \frac{x}{\text{var}(x)}$)



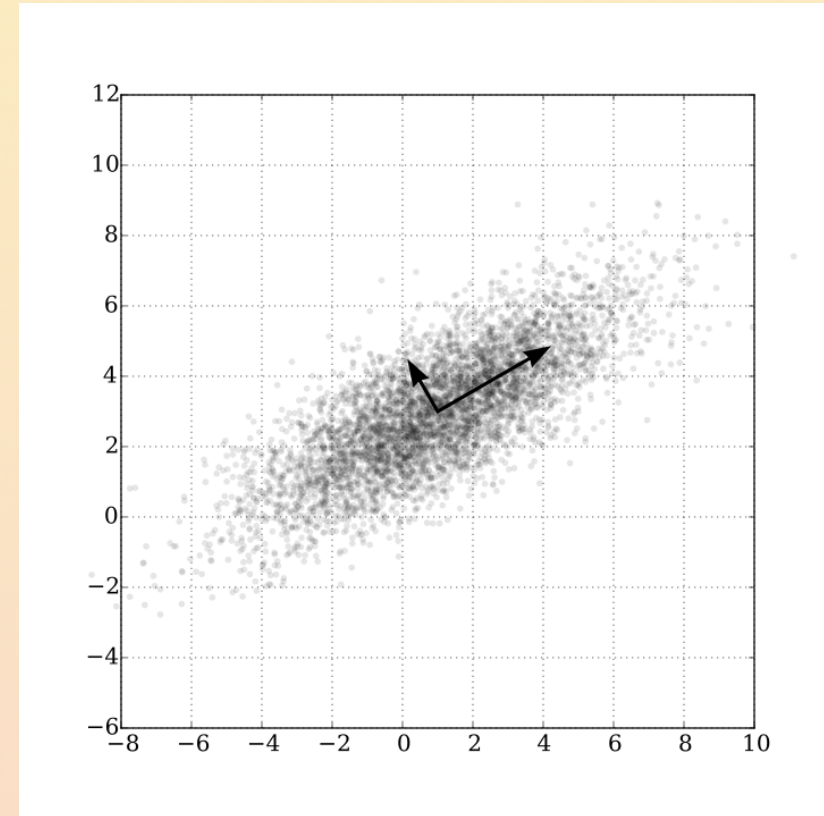
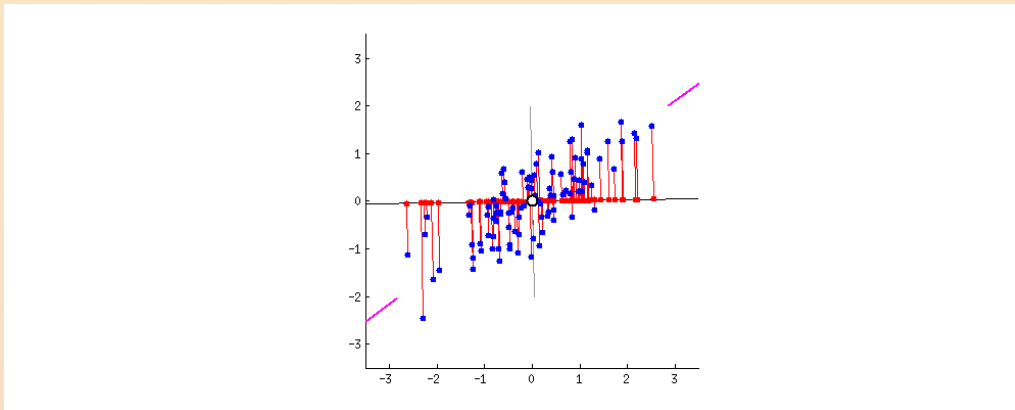
Data preparation: normalisation

- Normalize each data point to have unit norm
- Useful if using dot-product or other kernels to quantify similarity



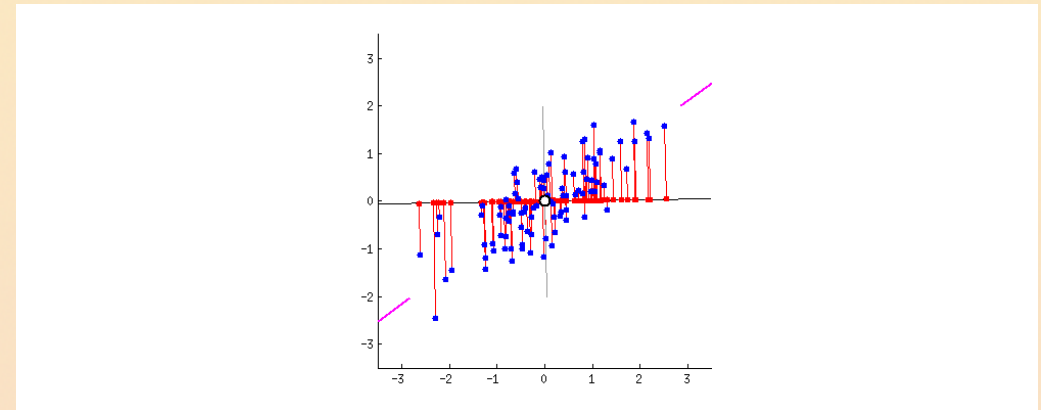
Data preparation: PCA

- Principal Component Analysis: find orthonormal basis where dimensions are linearly uncorrelated
 - Found iteratively finding the direction (linear combination of features) explaining the most variance
- Principal components are the **eigenvectors of the data covariance matrix**
 - Can be found by Singular Value Decomposition (SVD)
- Somehow analogous to finding axes of ellipsoid
 - Features with different units \rightarrow arbitrariness (scale them first)
- Can retain a few dimensions: dimensionality reduction
 - Drop directions least explaining the variance



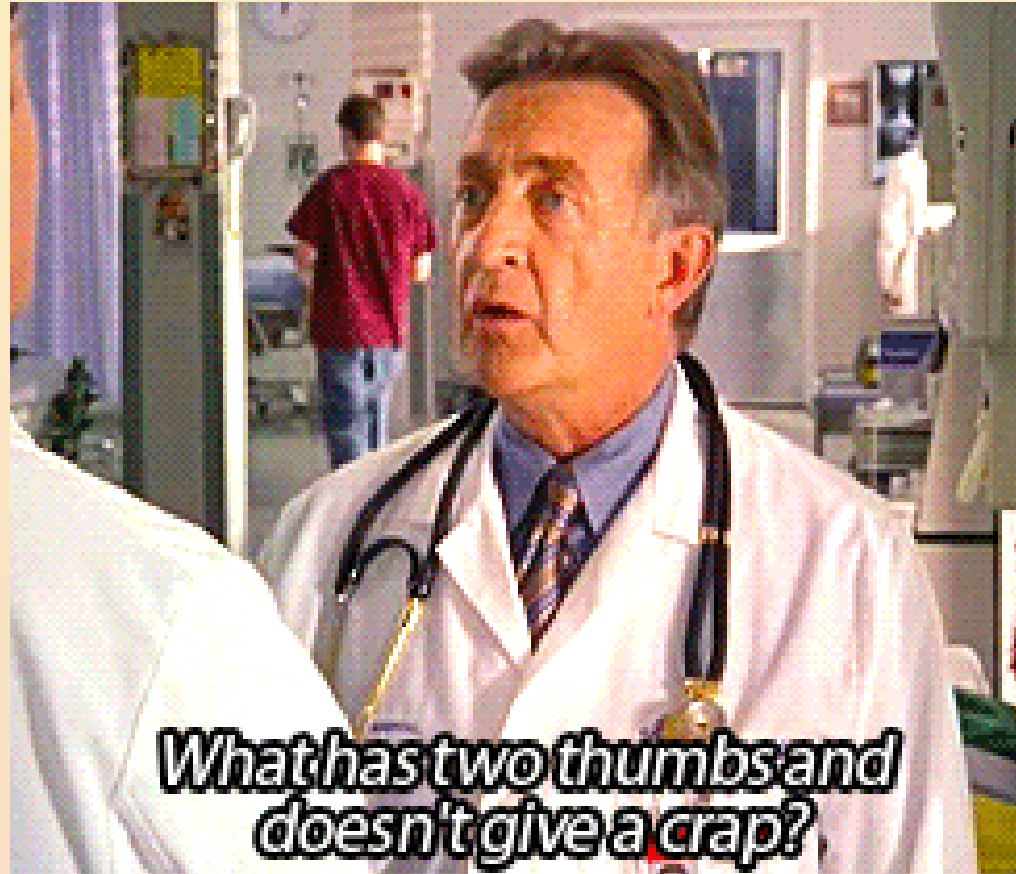
Data preparation: PCA

- Iteratively, find the first one, then find the next one conditioned on being orthogonal to the previous one
 - Or simply do Single Value Decomposition (SVD)
- SVD: 2D case (multivariate is just iteratively the same)
 - find the best linear fit: this shows the direction of maximum variance in the dataset
 - the eigenvector is the direction of that line
 - the eigenvalue expresses how much the data set is spread out on that line
- Steps for PCA
 - Standardize each variable
 - Compute covariance matrix
 - Compute eigenvectors of covariance matrix
 - Order them by eigenvalue
 - Select components you want to keep
 - Transform data in the new coordinate system



Data preparation: missing values

- LHC data are of **extremely good quality** (unlike e.g. medicine, social sciences)



- Use proxy feature (**pT of the two b jets** → **pT of the two jets with highest b-tagging discriminator**, in a region without b jets)
- Use average over other data points (**pT of the third jet** → ****mean of the third jet pT for data points that have it, if some data points don't have three jets**)

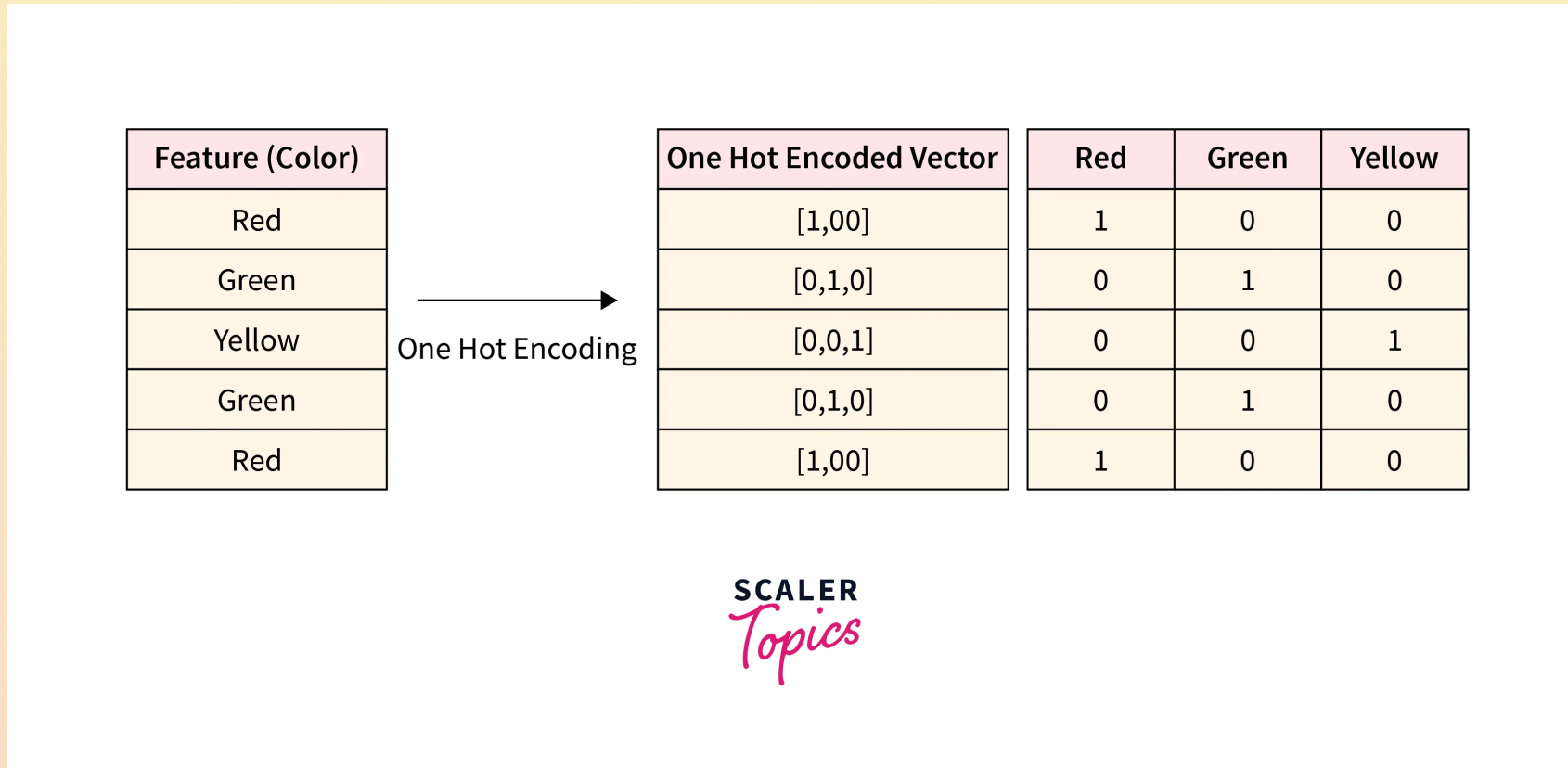
Data preparation: Types of categorical features

- **Nominal:** these are names, encoded as numbers (e.g. country of birth, assigning a certain number to each country). Operations do not make any sense whatsoever
- **Ordinal numbers:** express ordering, so comparisons (greater-than and less-than) make sense, but other operations (addition, subtraction, multiplication, etc) do not
- **Interval numbers:** express discrete measurements (e.g. year today). Subtraction has meaning (e.g., to calculate age), but 0 is arbitrary
- **Ratio numbers:** interval numbers, but where the 0 has a special meaning (e.g. length, time). Subtraction, multiplication, division have meaning

When the feature is categorical, how to tell a model that operations on a label don't make sense?

Data preparation: One-hot encoding

- One-hot encoding works both for input features and for output labels
 - Eliminates order, each category is orthogonal (independent)
 - Improves model performance: allows the model to capture structure and relationship between features that would not be capturable if encoded as a single number
 - Makes non-numerical data compatible with algorithms that require numerical inputs



Data preparation: One-hot encoding

- "Yellow" $[0, 0, 1]$ can be predicted as "0 for red and 0 for green"
 - One-hot-encoded features highly correlated ("multicollinearity")
- Dummy variable trap
 - Drop one of the "dimensions"

Feature (Color)
Red
Green
Yellow
Green
Red

→ One Hot Encoding

Red	Green
1	0
0	1
0	0
0	1
1	0

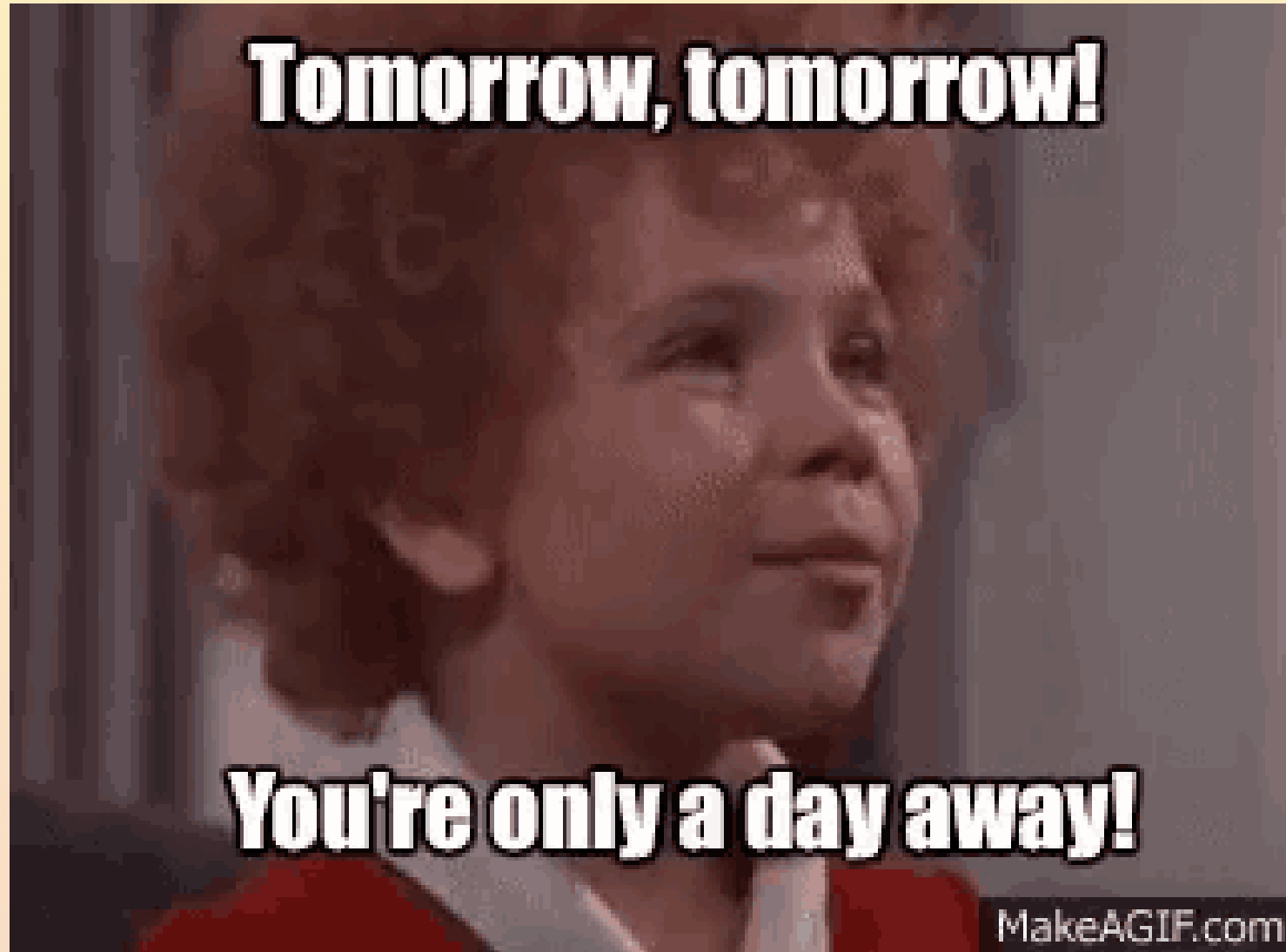
Yellow Column dropped to avoid the Dummy Variable Trap

Other types of encoding for nominal features

- **Label Encoding:** assume there is some ordering (treat nominal feature as an ordinal feature)
- **Binary Encoding:** convert categories into binary numbers and then creates binary columns. Preserves information, but compresses it
- **Target Encoding:** conflates regression and classification, replacing each category with the mean of a target feature for that category. Can be useful, but risk of [leakage](#) (averages done on full dataset imply information leak from test to training data set)

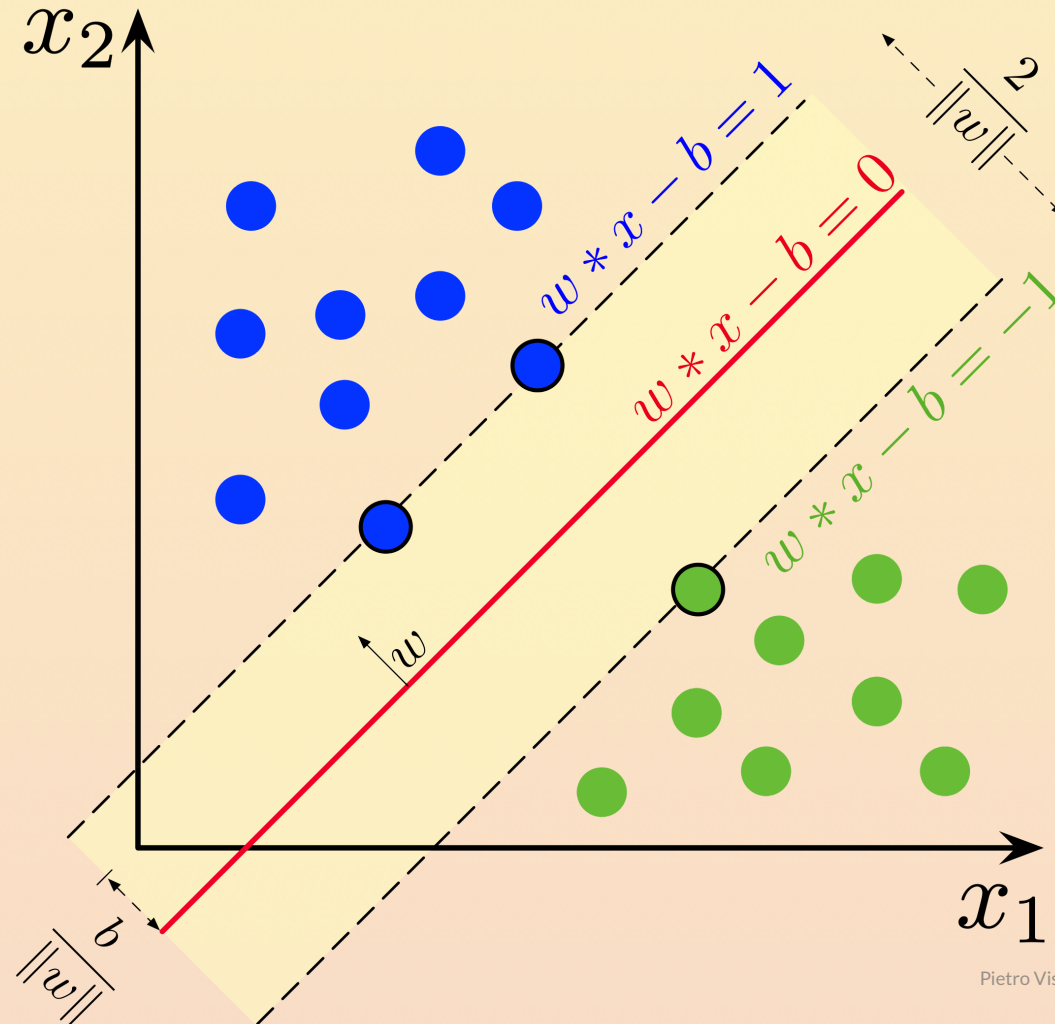
- That said, depending on the model you choose, maybe one-hot encoding is not even necessary. Some loss functions, for instance, are implemented in such a way that no implicit ordering is enforced (see in the next lecture)

Class weights vs event weights



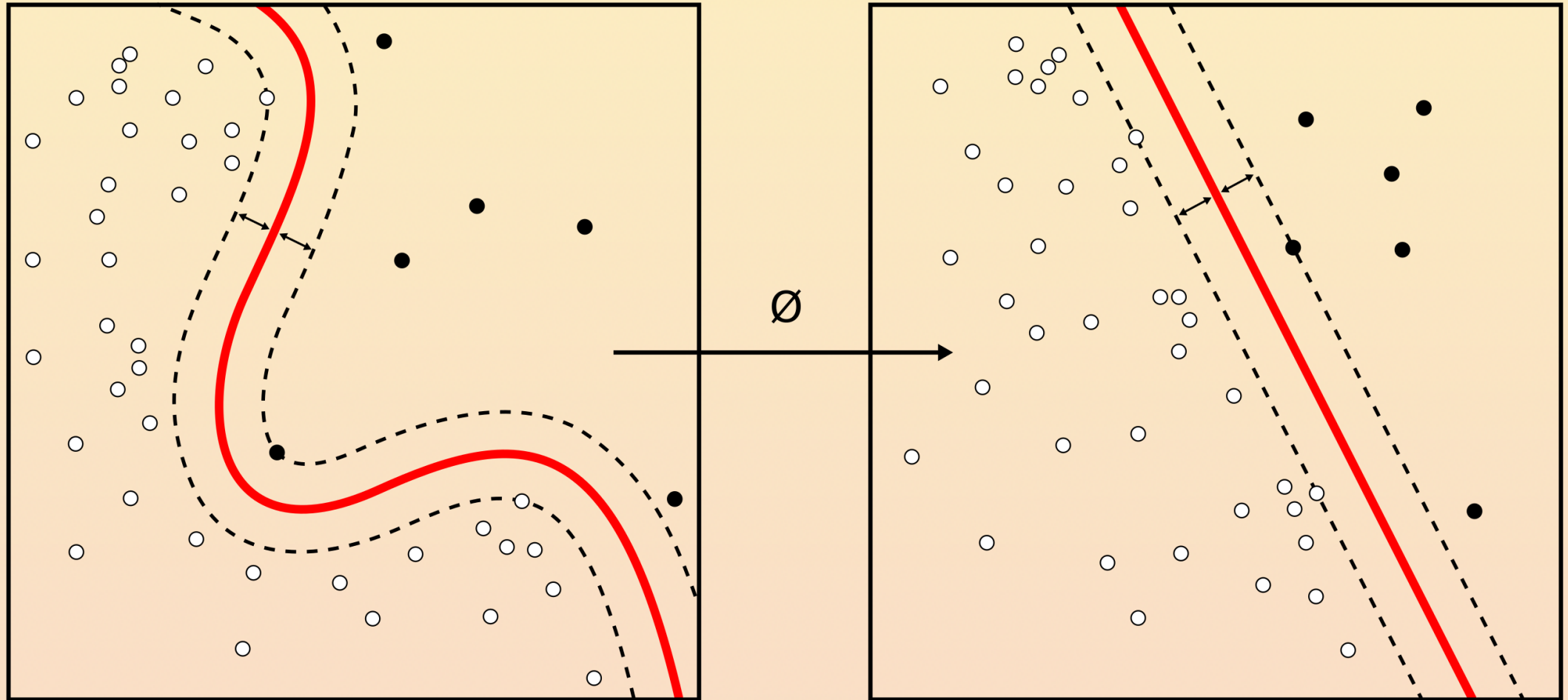
Support Vector Machines (Vapnik)

- Similar to a "cut-based" analysis, but sophisticated strategy for optimal class separation
- Minimize empirical classification error + maximize geometric margin



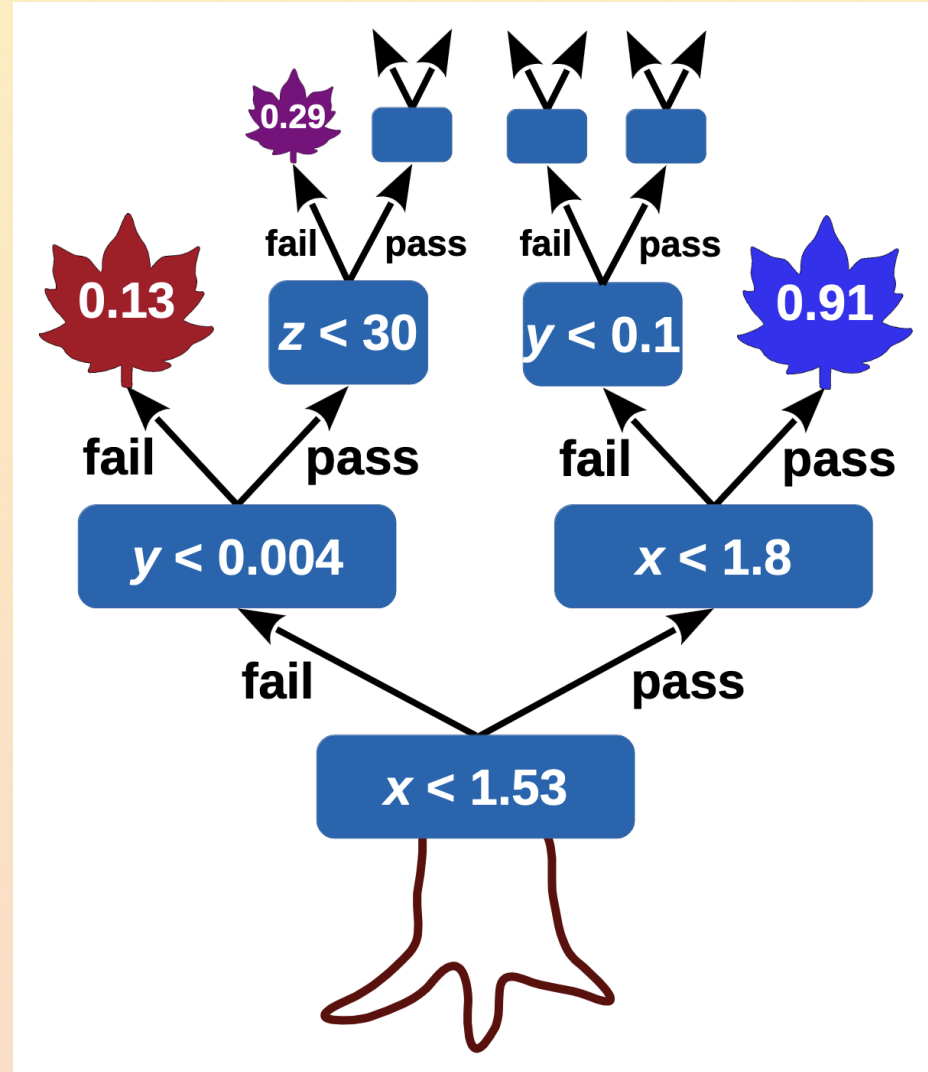
Support vector Machines

- Nonlinearity by kernel trick (deform the metric of the space until separation is linear)



Decision Trees

- "Cut-based" analysis on steroids
- Ordering is key
 - 1) check stopping criterion
 - 2) sort according to each feature
 - 3) compute all separations
 - 4) if best separation improves, split
 - 5) back to 1



Decision Trees: hyperparameters

- Class balance: normalize classes $\sum_i w_i = \sum_j w_j, \forall(i, j)$ at root node
 - In practice, early splitting provides balance anyway
- Impurity $i(t)$, and stopping criterion
 - Minimum leaf (end node) size (maximum error $\sqrt{N_{min}}$ ensures significance of purity estimate)
 - Perfect separation
 - Insufficient improvement
 - Maximum tree depth (purely computational requirement)

Decision Trees: splitting

- Impurity decrease: $\Delta i(S, t) = i(t) - p_P i(t_P) - p_F i(t_F)$
- Optimization problem: $\Delta i(S^*, t) = \max_{S \in \text{splits}} \Delta i(S, t)$

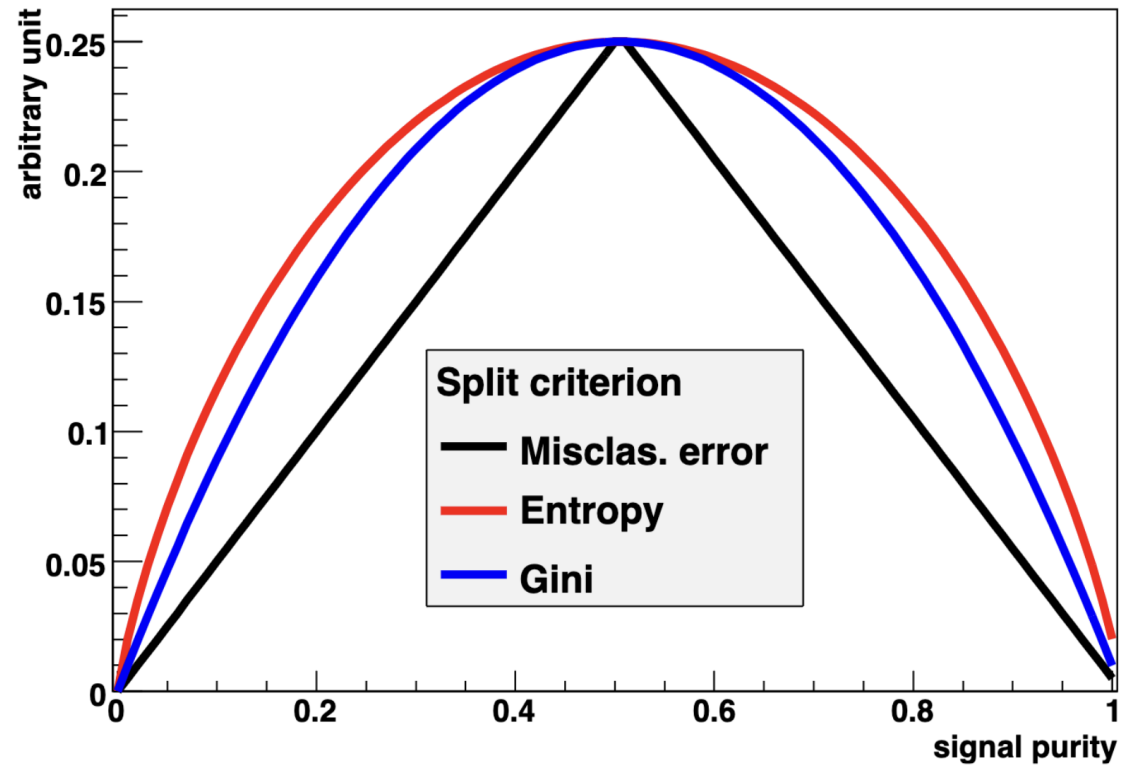
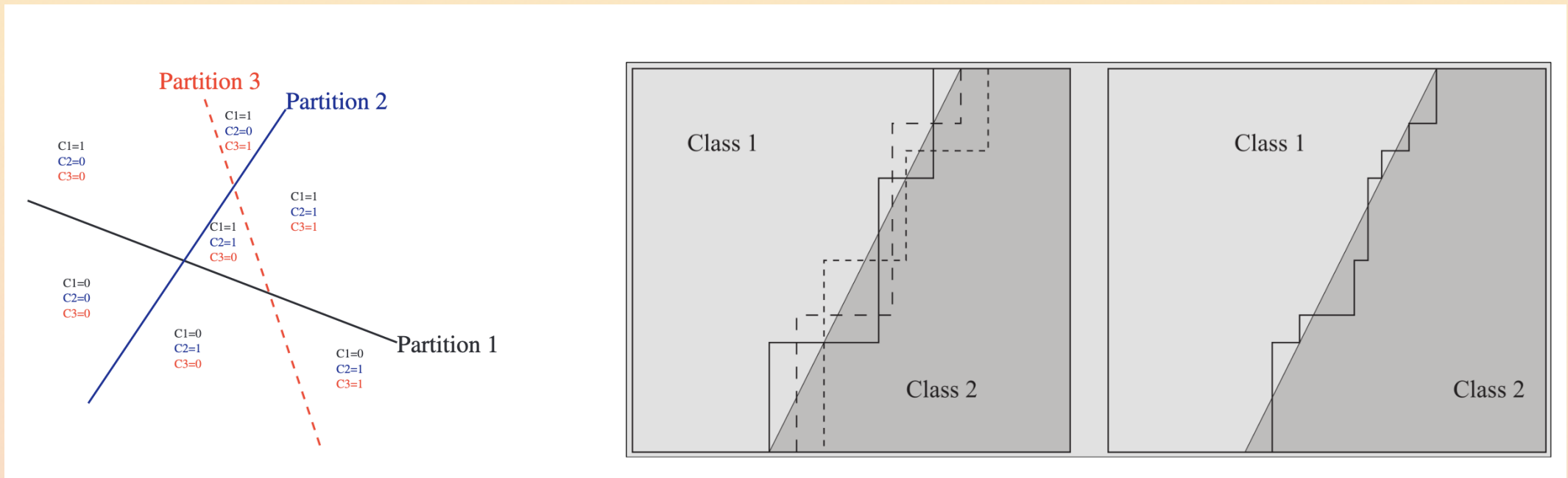


Fig. 5. Impurity measures as a function of signal purity.

Decision Trees: limitations

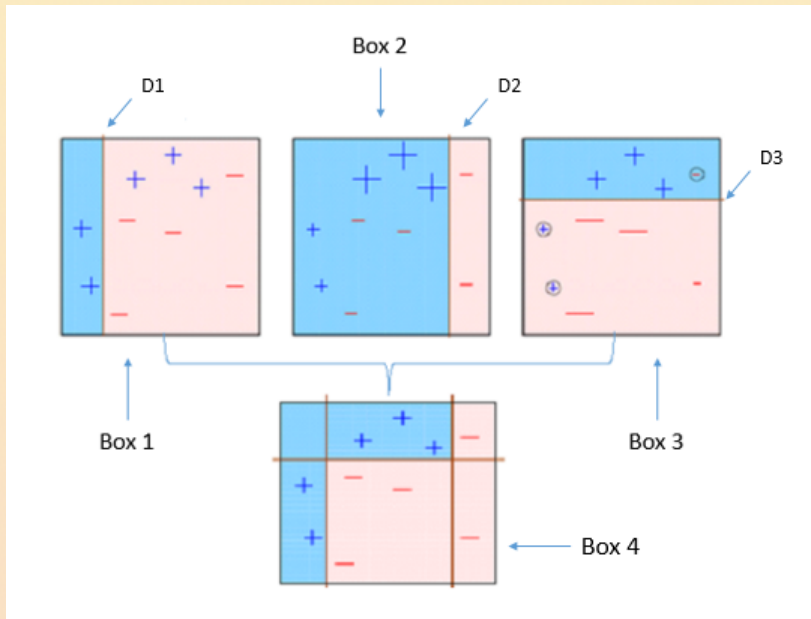
- Sensitivity to training sample
 - Decision trees are not robust: high variance for small changes in training sample
- Tree depth results in higher statistical uncertainty in the split purity
 - Can be mitigated by [pruning](#)
- Ensemble learning fixes all of this
 - Richer description when intersecting partitions
 - More accurate description when averaging partitions



Boosted decision trees

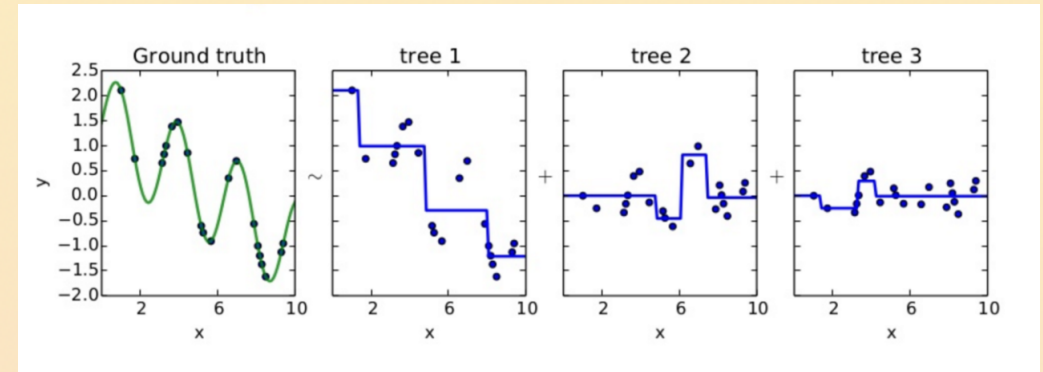
- Ada(ptive) Boost

- Increase at each iteration the importance of events incorrectly classified in the previous iteration



- Gradient Boost

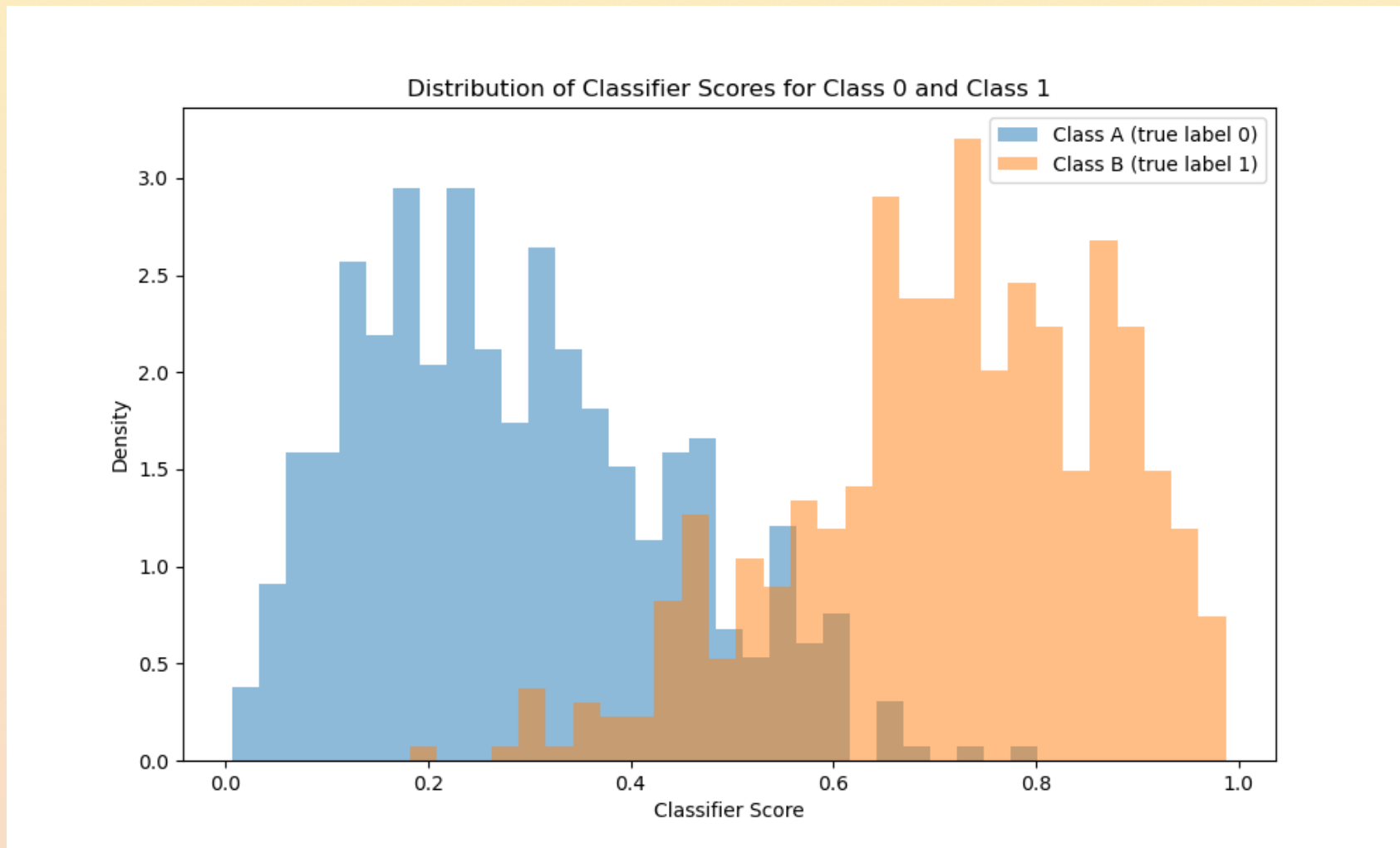
- fit the new predictor to the residual errors of the previous one



- **Bagging:** training trees on bootstrap replicas, average all trees
- **Random forest:** bagging + pick only a random subset of features at each step

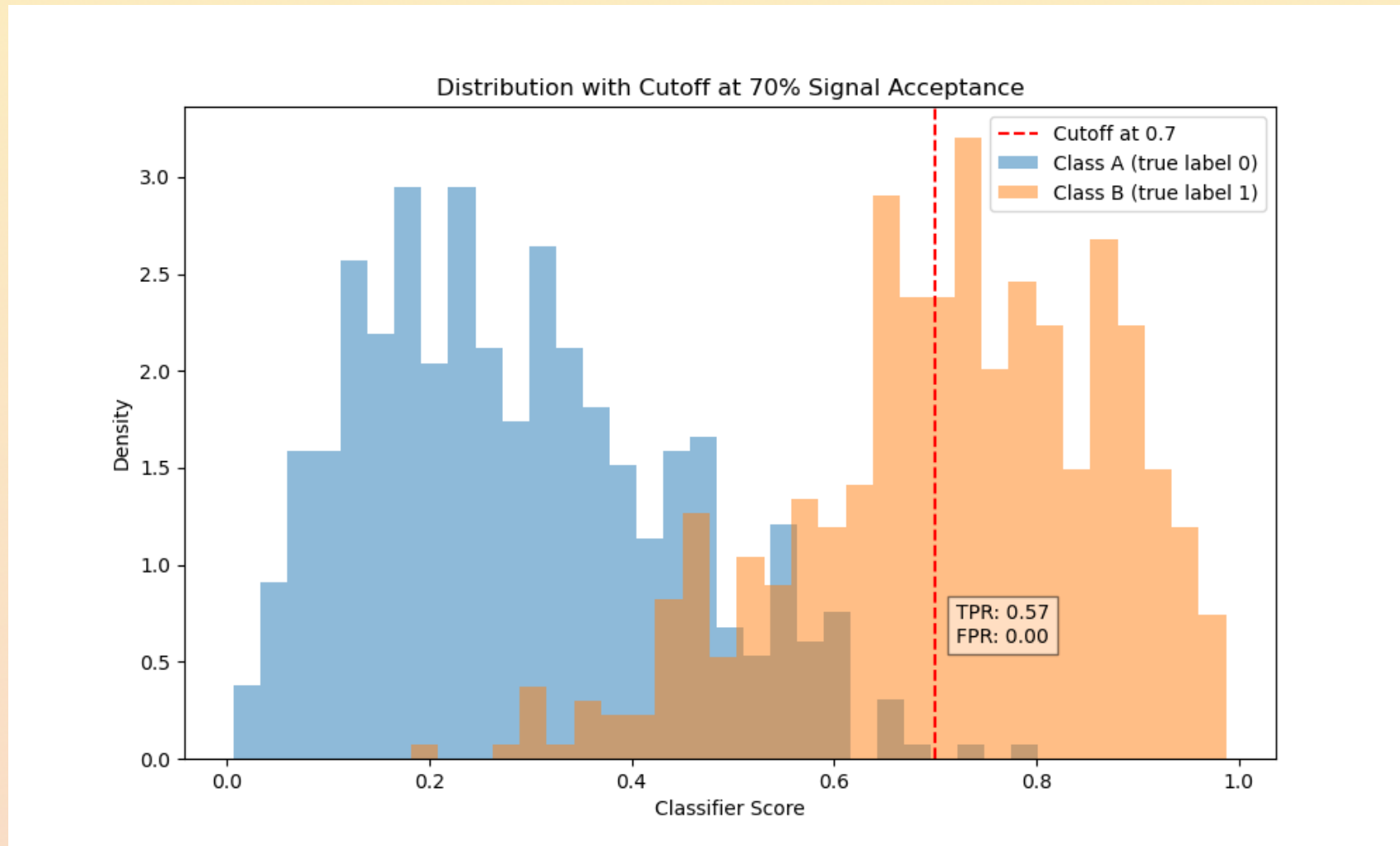
Performance (classification): ROC Curve explained

- Depending on how well the classifier works, each class will have a score distribution close to its true label



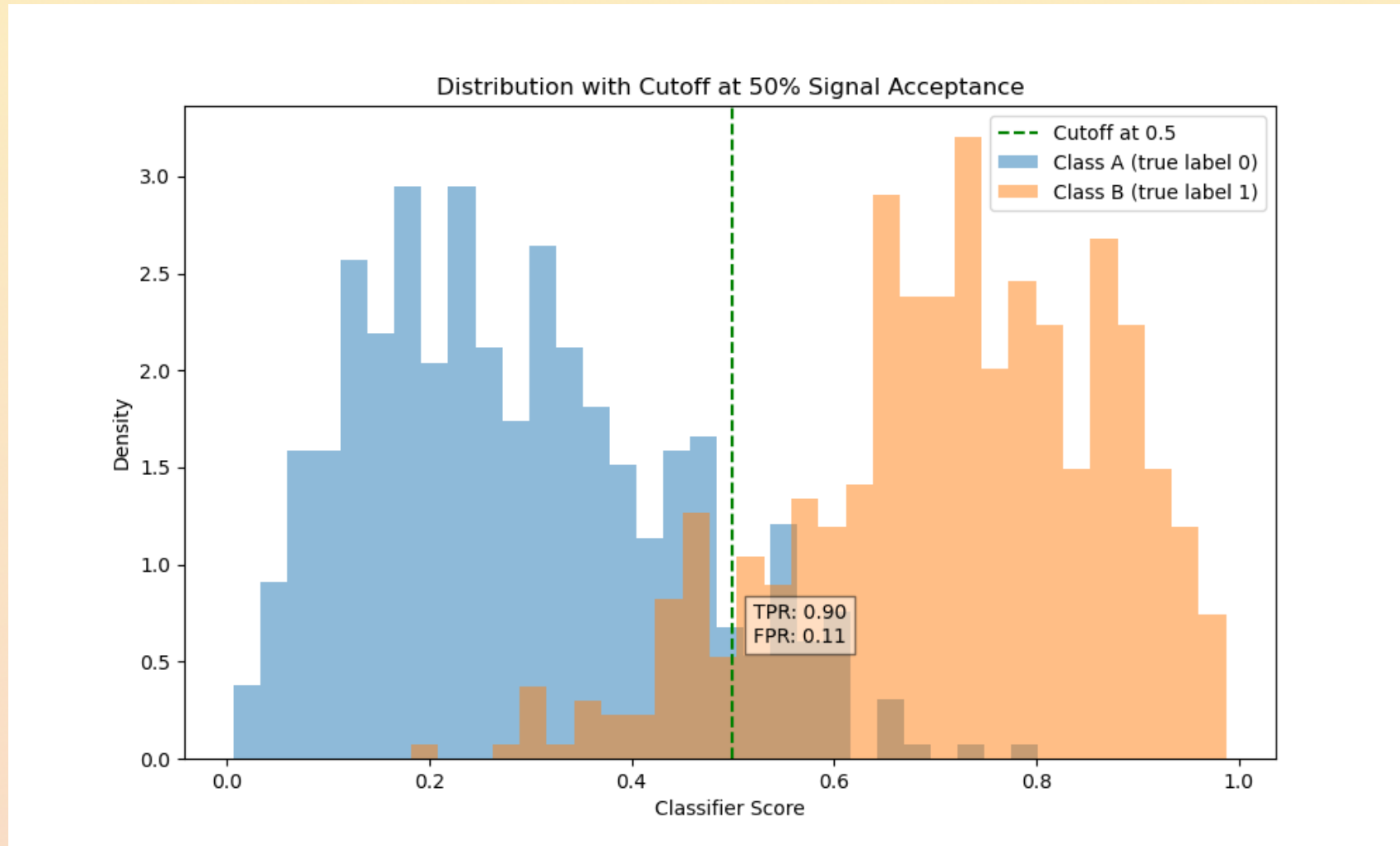
Performance (classification): ROC Curve explained

- To use the score for categorization, decide "I label as Class B everything with a score larger than X"



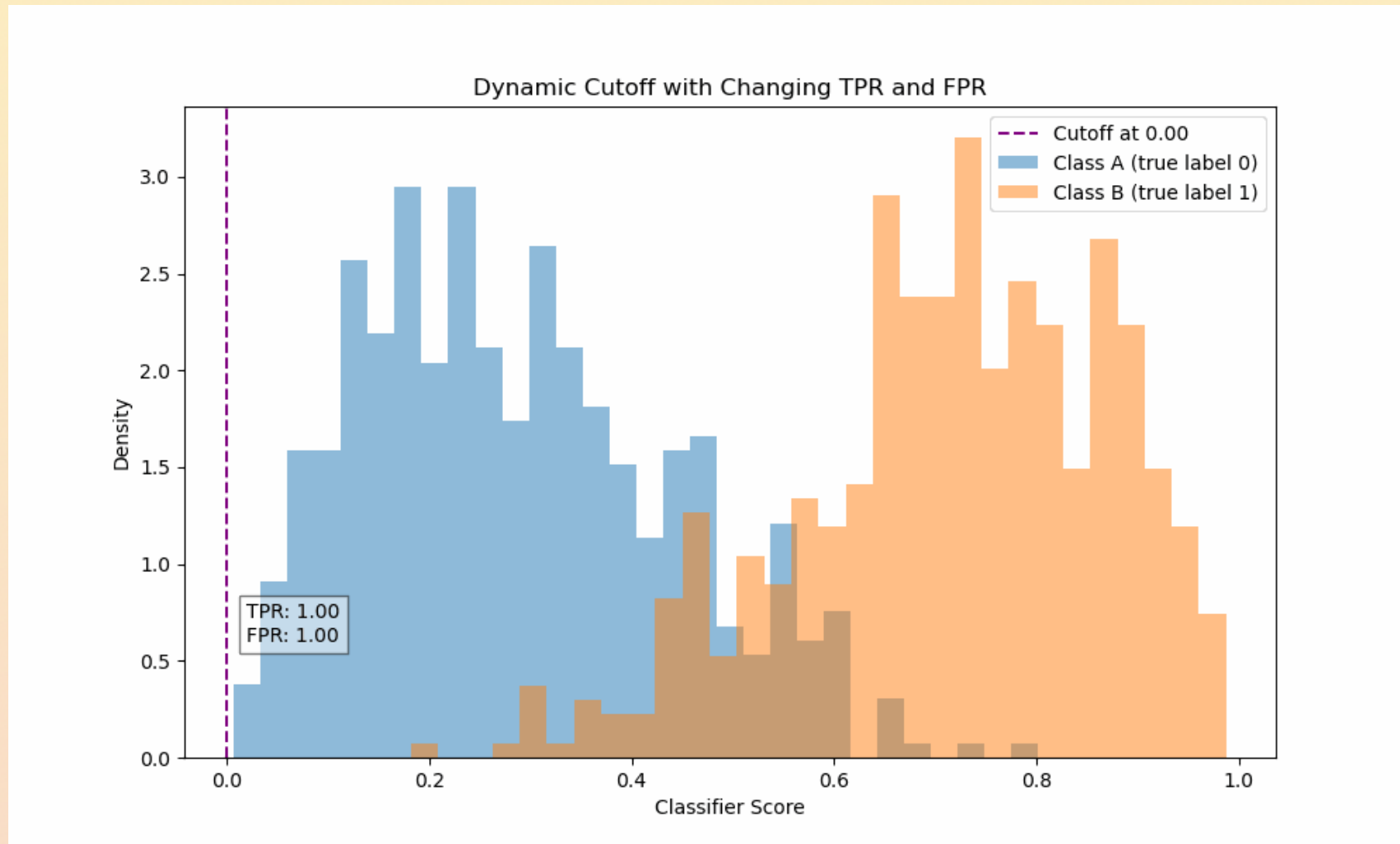
Performance (classification): ROC Curve explained

- Changing the cutoff results in tradeoff between larger TPR at the price of a larger FPR



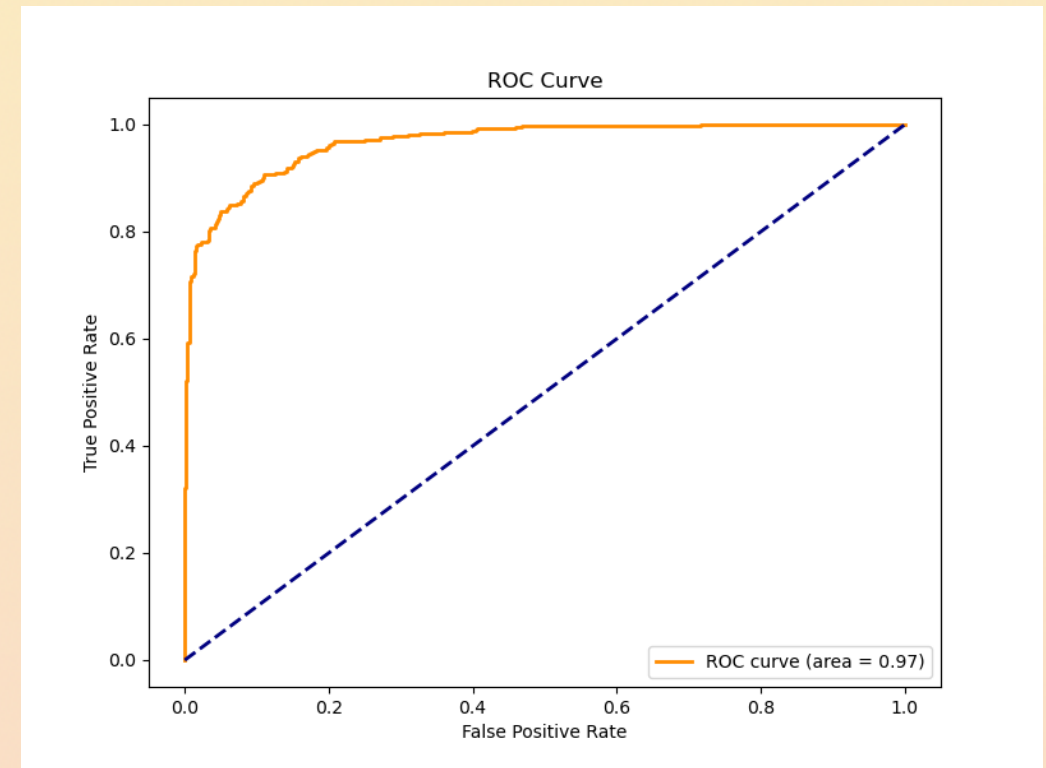
Performance (classification): ROC Curve explained

- Changing the cutoff continuously results in a set of pairs (TPR , FPR) that describes a continuous line



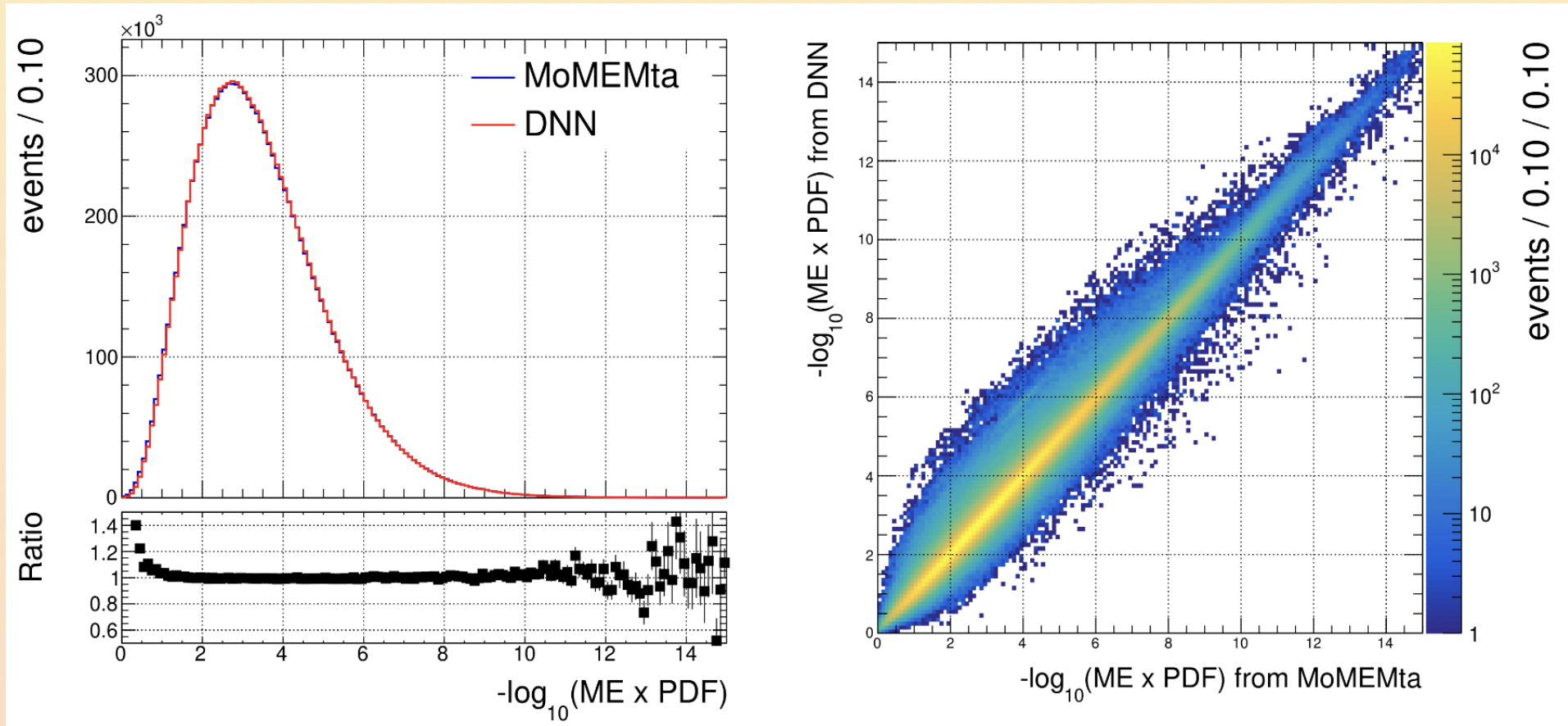
Performance (classification) ROC Curve explained

- The line described by the pairs (TPR, FPR) is the **Receiver Operating Characteristic (ROC) curve**
 - The diagonal is the worst performance possible (throwing a coin)
 - The closer the integral is to 1, the better. If the integral is smaller than 0.5, then you can invert the decision to have a good classifier
- Usually then you choose your decision cutoff by deciding which FPR you accept or which TPR you want
- For two classes (signal, background), it is signal efficiency vs background efficiency (rejection := 1-efficiency)
- For multiple classes: either pairwise or one-vs-all



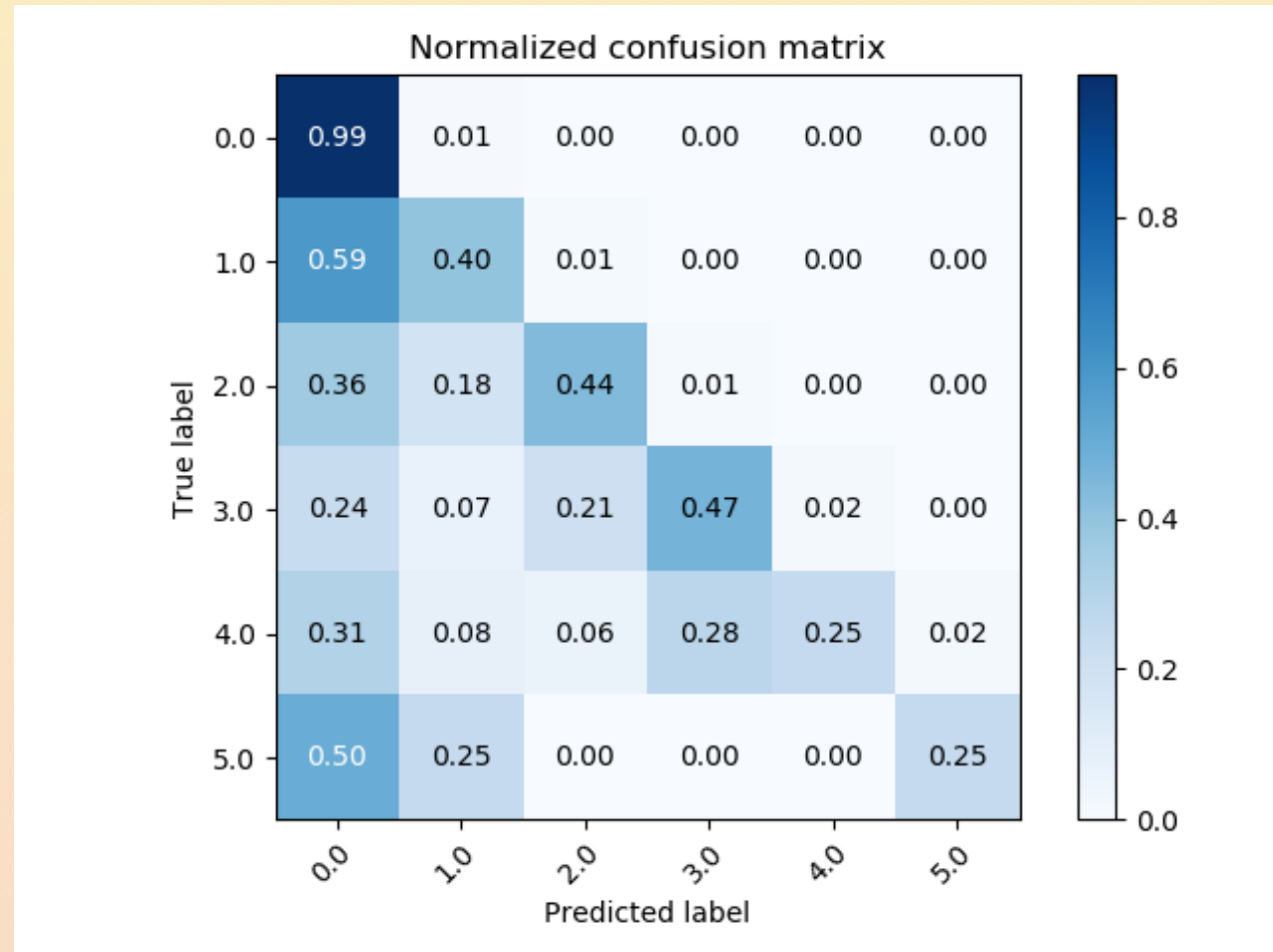
Performance (regressio): scatter plots

- For regression problems
- Can compute linear pearson coefficient as an estimate of linearity
 - Formulas exist also for weighted events



Confusion matrix

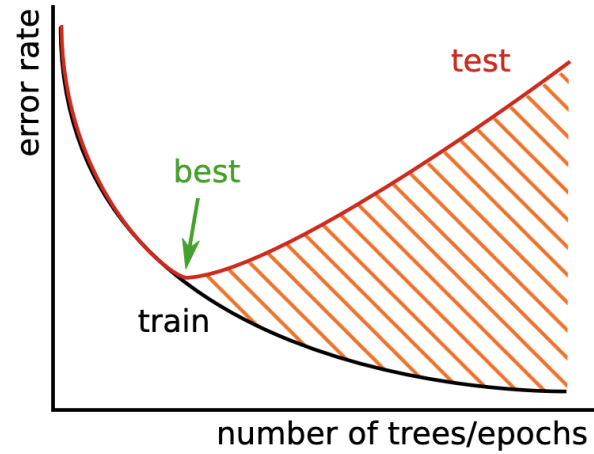
- For classification, can also use it to discretize regression (e.g. in the case of histograms)
- Note the normalization (each true label row sums up to 1)



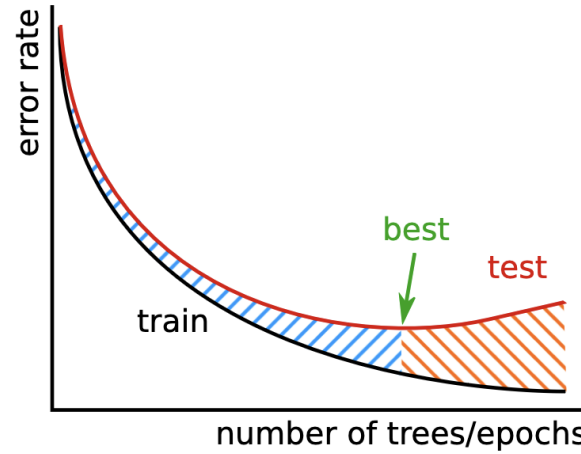
Error rates (for reference)

		CONDITION determined by "Gold Standard"			
		CONDITION POS	CONDITION NEG	PREVALENCE $\frac{\text{CONDITION POS}}{\text{TOTAL POPULATION}}$	
TEST OUT- COME	TEST POS	True Pos TP	<i>Type I Error</i> False Pos FP	<i>Precision</i> Pos Predictive Value $\text{PPV} = \frac{\text{TP}}{\text{TEST P}}$	False Discovery Rate $\text{FDR} = \frac{\text{FP}}{\text{TEST P}}$
	TEST NEG	<i>Type II Error</i> False Neg FN	True Neg TN	False Omission Rate $\text{FOR} = \frac{\text{FN}}{\text{TEST N}}$	Neg Predictive Value $\text{NPV} = \frac{\text{TN}}{\text{TEST N}}$
ACCURACY ACC $\text{ACC} = \frac{\text{TP} + \text{TN}}{\text{TOT POP}}$		<i>Sensitivity (SN), Recall</i> Total Pos Rate TPR $\text{TPR} = \frac{\text{TP}}{\text{CONDITION POS}}$	<i>Fall-Out</i> False Pos Rate FPR $\text{FPR} = \frac{\text{FP}}{\text{CONDITION NEG}}$	Pos Likelihood Ratio LR + $\text{LR} + = \frac{\text{TPR}}{\text{FPR}}$	Diagnostic Odds Ratio DOR $\text{DOR} = \frac{\text{LR} +}{\text{LR} -}$
		<i>Miss Rate</i> False Neg Rate FNR $\text{FNR} = \frac{\text{FN}}{\text{CONDITION POS}}$	<i>Specificity (SPC)</i> True Neg Rate TNR $\text{TNR} = \frac{\text{TN}}{\text{CONDITION NEG}}$	Neg Likelihood Ratio LR - $\text{LR} - = \frac{\text{TNR}}{\text{FNR}}$	

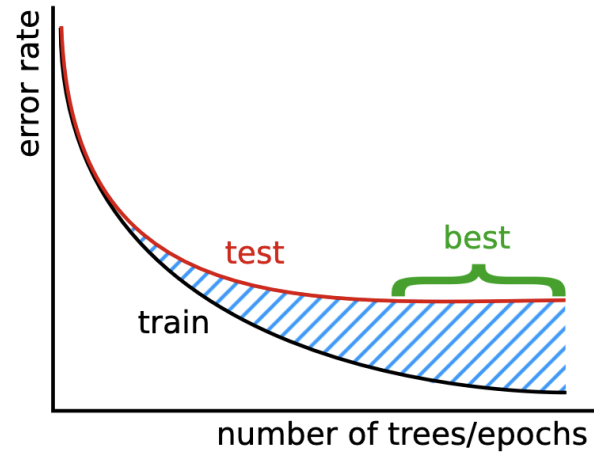
Model complexity



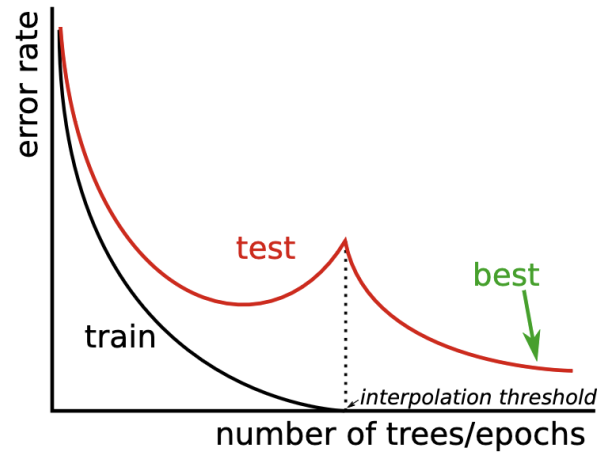
(a)



(b)



(c)

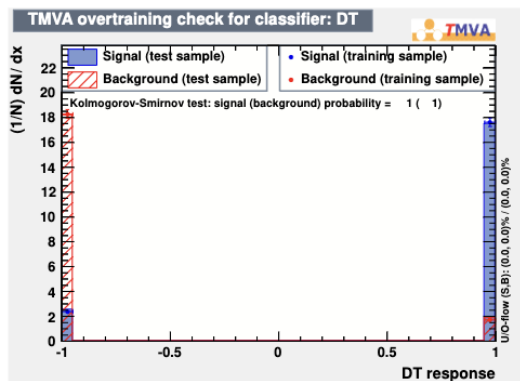


(d)

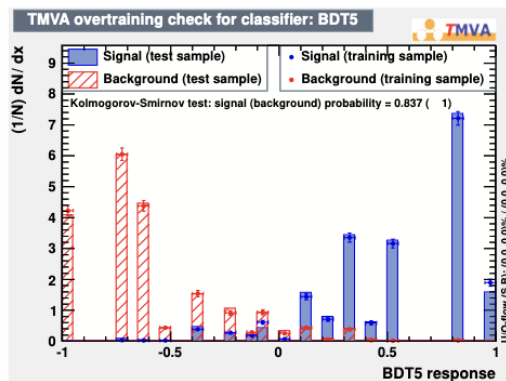
Overtraining check

- KS test done mostly for BDTs. For neural networks, much handier ways

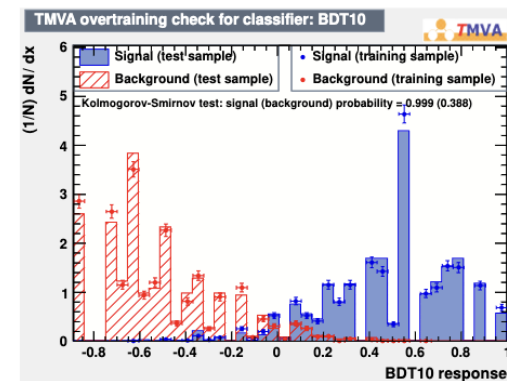
Do the training and test output distributions come from the same underlying p.d.f.?



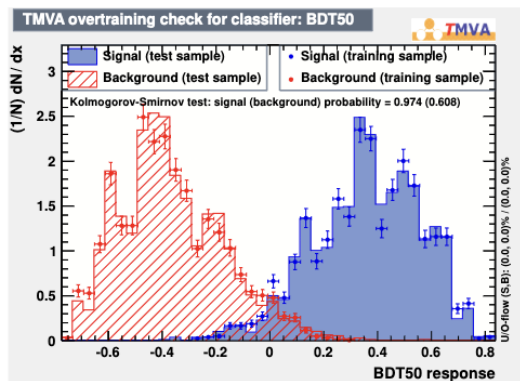
(a) Single decision tree



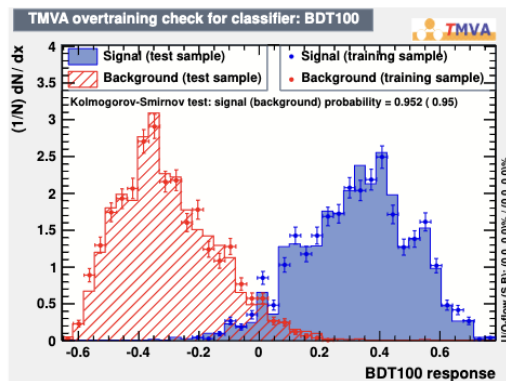
(b) 5 trees



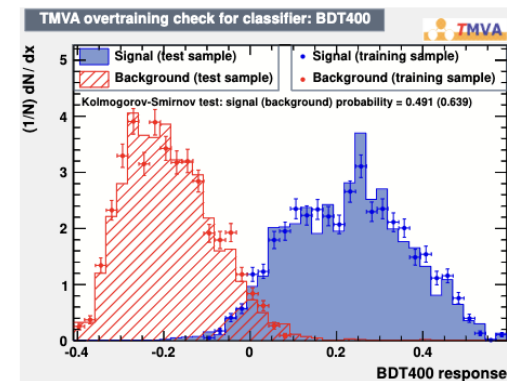
(c) 10 trees



(d) 50 trees



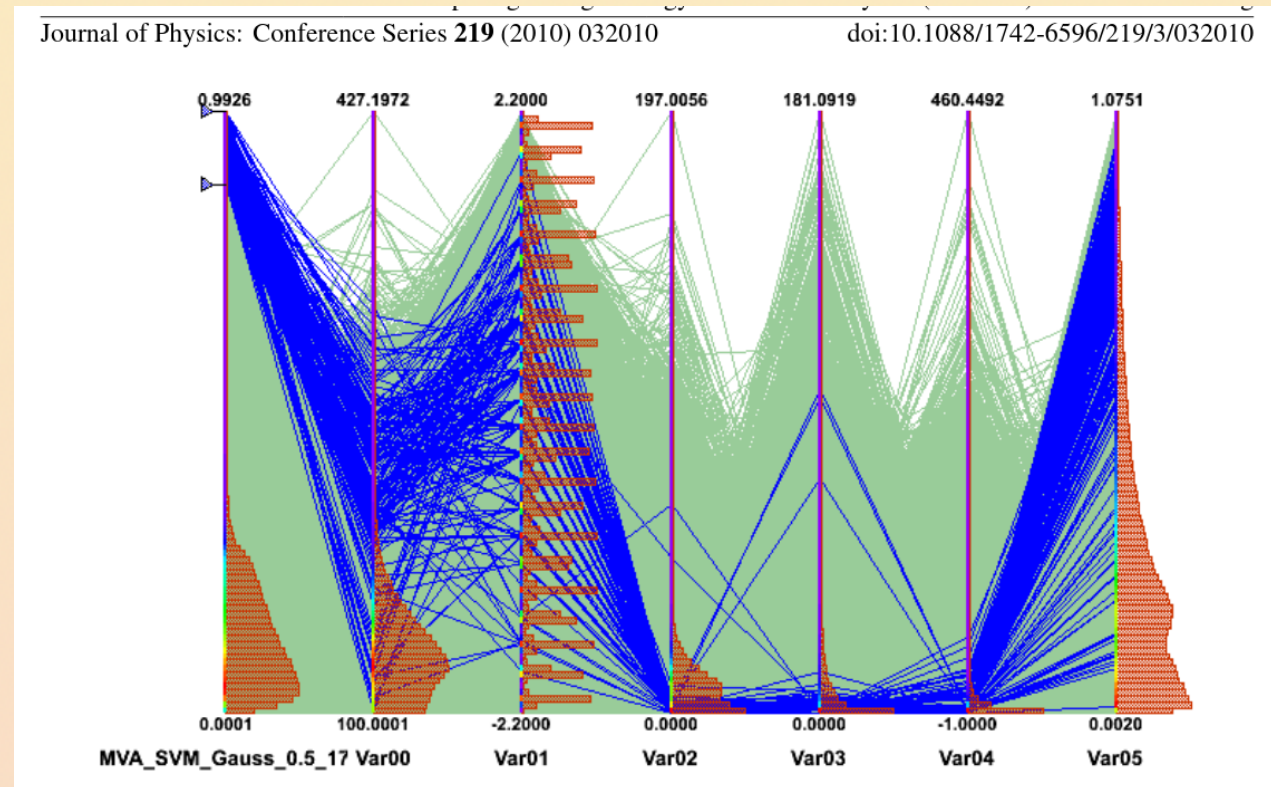
(e) 100 trees



(f) 400 trees

Interpretability, explainability

- **Permutation Importance**: the decrease in a model score when a single feature value is randomly shuffled (see [scikit-learn documentation](#)) (akin to impacts for profile likelihood fits)
- **Shapley Values**: based on game theory (will use them this afternoon)
- **Correlation-based**: e.g. parallel coordinates in TMVA: look where each variable is mapped to/correlated with

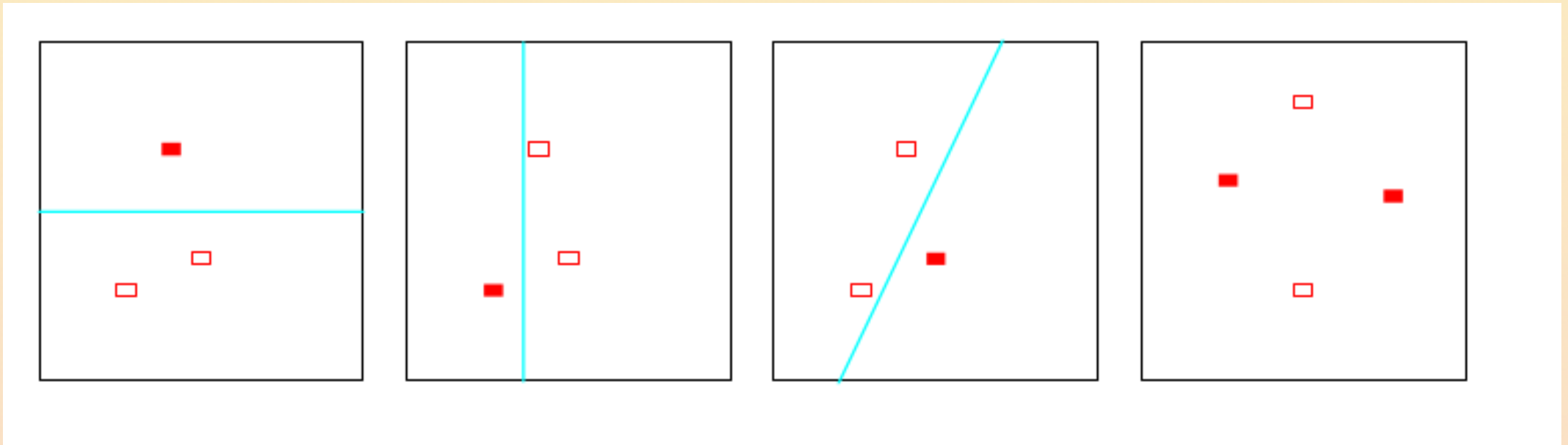


Model assessment by comparing models

- **Bayesian Information Criterion:** $BIC = n_{free\ params} \ln(n_{data}) / 2 \ln(\hat{L})$
 - Parameter θ predicted by two models M_0 and M_1 : $P(\theta|\vec{x}, M) = \frac{P(\vec{x}|\theta, M)P(\theta|M)}{P(\vec{x}|M)}$
 - Apply Bayes theorem to Bayesian evidence (Model likelihood): $P(\vec{x}|M) = \int P(\vec{x}|\theta, M)P(\theta|M)d\theta$
 - Posterior odds: $\frac{P(M_0|\vec{x})}{P(M_1|\vec{x})} = \frac{P(\vec{x}|M_0)\pi(M_0)}{P(\vec{x}|M_1)\pi(M_1)}$
 - Can rewrite posteriors in terms of BIC, equivalent
- **Minimum Description Length (MDL):** Kolmogorov complexity (length of minimum program needed to describe the data)
 - *for i = 1 to 2500; do print'0001'; halt*
 - *print'101001010100010111001000010000101110011100001010100101...'; halt*

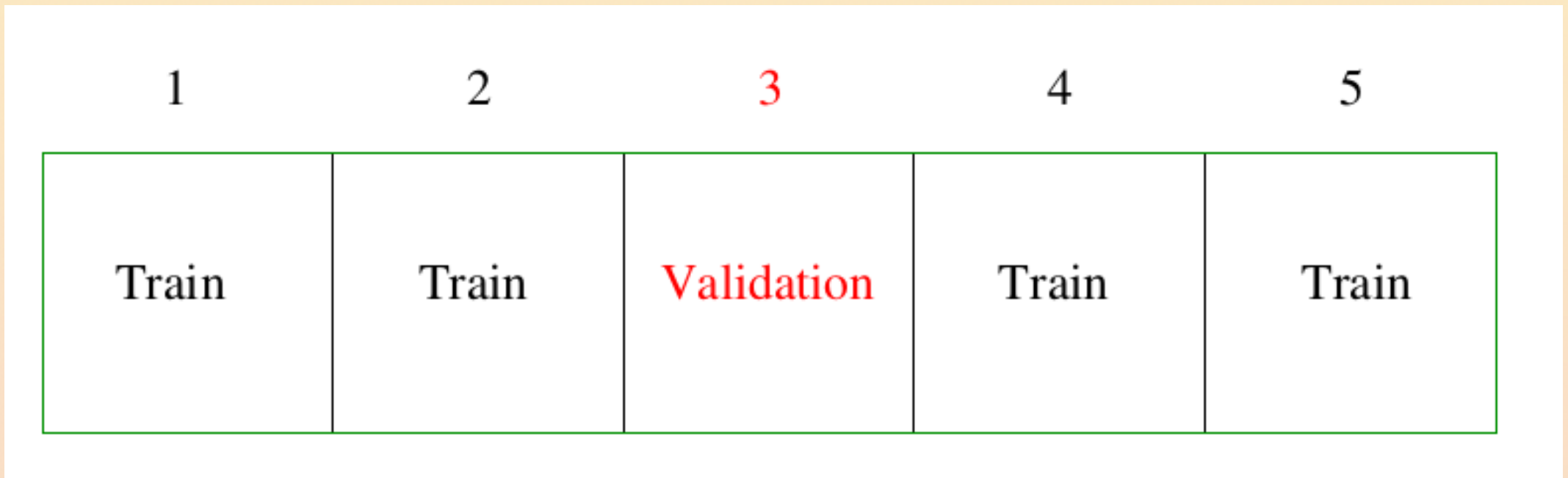
Model assessment by comparing models

- **Structural Risk Minimization:** complexity as Vapnik-Chervonkensis class (largest number of shattered points)
 - Build a nested sequence of models with increasing VC complexity h
 - Write a probabilistic upper bound for the regression error: $err \leq f(h/N)$
 - Choose model with smallest value of the upper bound



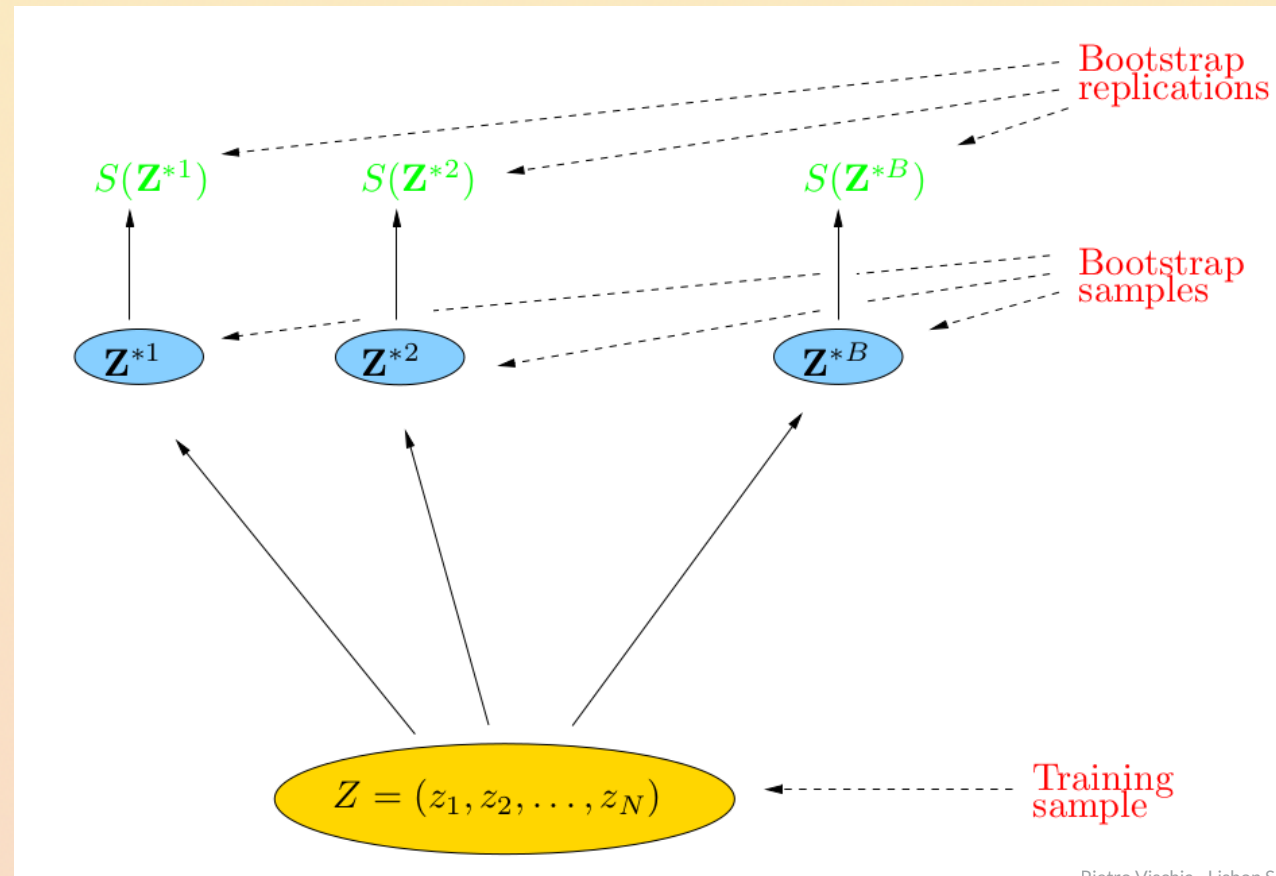
Model assessment: focus on prediction error

- **Cross-validation:** useful when data are scarce
 - Split the data into K parts ("folds")
 - For the k th part, fit the model to the other $K-1$ folds, and calculate test error as error on predicting the k th part data
 - Do this for all k , then combine the K estimates of the prediction error
 - Choose K
 - $K=N$ (leave-one-out), unbiased but high variance (training sets are basically the same)
 - Low K (5--10): Lower variance, but maybe bias (folds not representative of the data set)



Model assessment: focus on prediction error

- **Bootstrap**: a general tool to assessing statistical accuracy
 - Estimate the variance on the statistic $S(Z)$ (Z are the data)
 - Can be used as model assessment tool, or to improve an estimator
 - **Bagging** to combine weak learners (ensemble learning)



Next: supervised learning

Don't forget to let us know NOW if you need a wifi account (i.e. if you don't have an eduroam account or if your account doesn't work)