Lecture I: Machine Learning

Outline:

Lecture I: → Motivation
→ Machine Learning basics
→ Example of polynomial regression

Lecture II: → Training procedure
→ Neural networks

Lecture III: → ML in high energy physics

Motivation: Why using ML in physics?

→ Deep Learning revolution in many fields outside of physics, such as:
↳ Computer vision:
- image ~~recognition~~ classification
- image generation
- image segmentation: locate objects and boundaries

↳ Speech recognition
⋮

→ We should take advantage of this and apply it to physics!

example: LHC physics ✓ (via simulations) ✓

What do we need?

→ large data sets
(→ truth labels)

ML basics

What is ML?

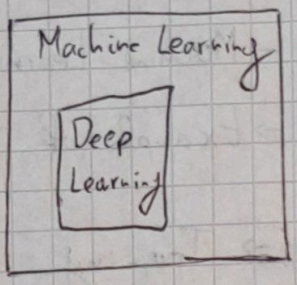
Study of algorithms which are learnt through the use of data

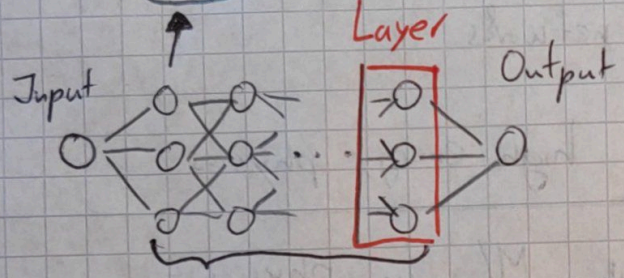
vs. ($\Leftrightarrow$) handcrafted algorithms

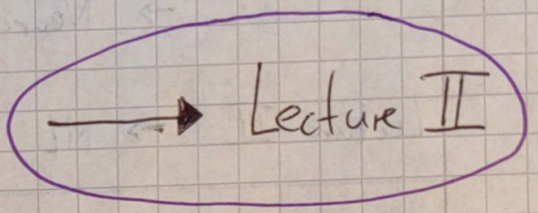
What is Deep Learning?

↳ subfield of ML

↳ use of (deep) neural networks







deep $\hat{=}$ many layers $\sim$ many parameters / trainable weights

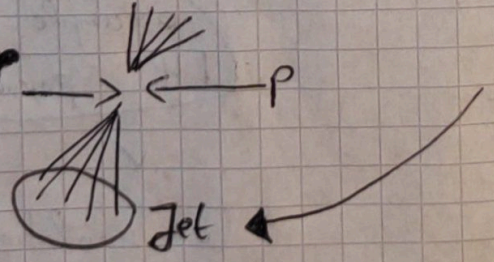
What is the advantage of deep networks?

→ allows to learn features of high dimensional data

$\Leftrightarrow$

→ "non deep" learning:
- require more human work
- reduce dimensionality by extracting features by hand!

Example: LHC physics



Jet image

high dimensional data

Deep network

vs

features

$(p_T^J, \varphi^J, \eta^J,$ N-subjettiness$)$

BDT

Categories of Machine Learning

- **Supervised:** labeled data!

  Example:

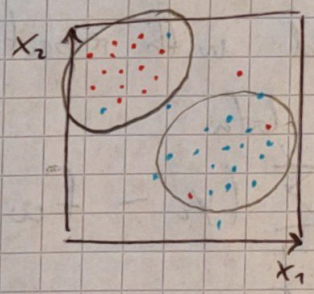
  (i) Classify images of cats and dogs. Each image is either labeled "cat" or "dog".

  (ii) top tagging: hadronically decaying top vs. QCD jet. For each jet of the training data set we know that it originated from a top or light quark/gluon!
  → label: "top", "QCD"

- **unsupervised:** no labels! Learn patterns in data, detect anomilies

  Example:

  (i) clustering



  (ii) data set with mostly dogs but a few cats.
  However, no labels! We don't know which image is a cat or a dog!
  ↑ anomaly

- **reinforcement learning:** Similar to supervised but no simple data set. Instead an "agent" interacts with an enviroment and has to adopt behaviour accordingly.
  ↳ Sequential decisions: trial and error

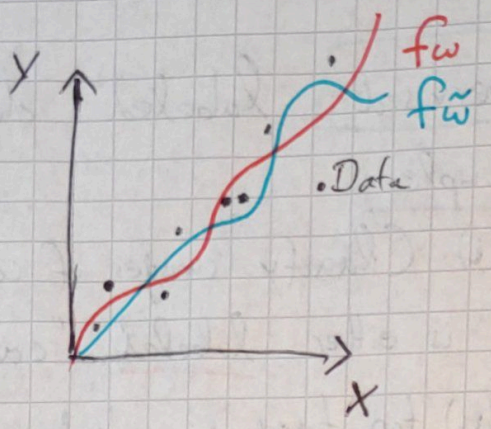
  Example:

  (i) games where agent has to adopt constantly to new enviroment

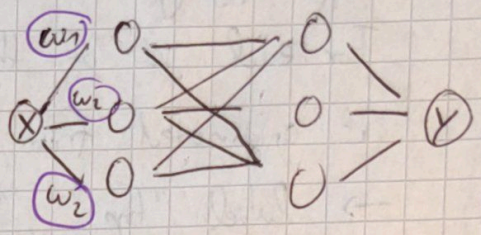
  (ii) robot which has to adopt to enviroment

Basic ingridients of ML (supervised)

→ **Dataset** $\mathcal{D} = (X, Y)$
Input ↑ ↑ Label



→ **Model**: $f_w(x)$
↑ w: trainable parameters



→ **Loss function**: Defines the task!
- Object which is minimized with respect to w
  to best describe the data

Example: $MSE = \sum_{i=1}^{N} (y_i - f_w(x_i))$, N: Number of data points

~ Error
(neg log of Gaussian)

→ **Training**: Minimization of loss function → Lecture 2

Example of polynomial regression

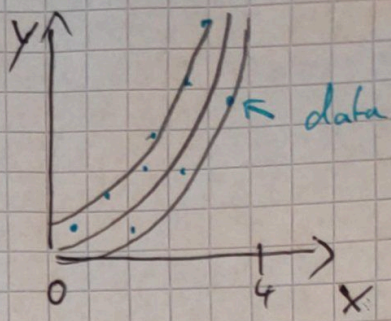
Motivation:
- get some intuition about ML
- key points extrapolate to Deep Learning

(This is stolen from the great review: arXiv: 1803.08823)

Data set: $x \in [0,4]$, $y = x^2$ + Noise

→ true dependence!

→ we don't know!

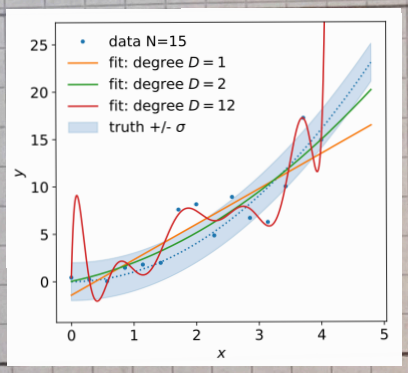


Model: $f_w(x) = w_0 + w_1 x + w_2 x^2 + \dots w_D x^D$

polynomial of degree D

$\equiv f_D(x)$

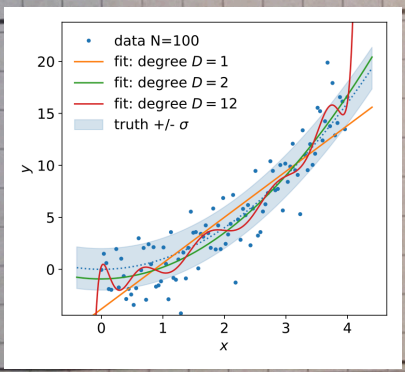
Loss function: MSE

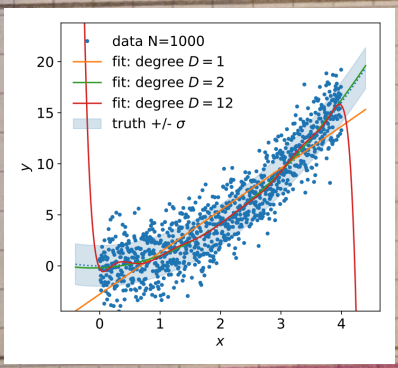


→ f_1: not "complex" enough

→ f_{12}: fits all statistical fluctuations! ≙ Overfitting

How to fix this?





→ Complex models need large data sets!

Deep Learning requires large data sets!

(Example for state of the art deep networks: GPT-3 ~ 172 B trainable weights)

How to quantify performance?

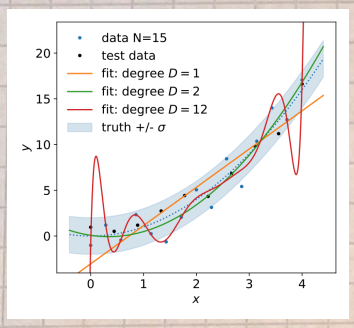
MSE ~ Error

But MSE is smallest for $f_{12}(x)$!. Even for N = 15!

⇒ Have to split data set before training!

D → D_{train} ~80% → for training

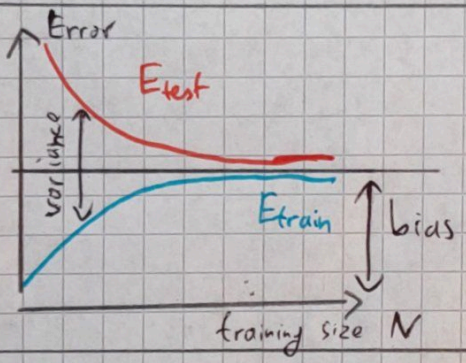
D → D_{test} ~20% → for evaluation

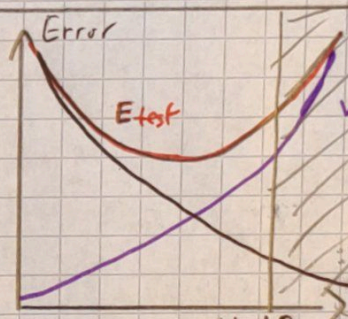


Compare MSE =: E

$E^{train} = MSE(D^{train})$

$E^{test} = MSE(D^{test})$ ⇄ vs.





← Deep Learning

- Need of seperate data set for evaluation!
- For Deep Learning: need of ~~regularisation methods~~ ~~to~~ large data sets and potentially regularisation methods

↳ lecture 2 or 3?