Physics 693: Modern ML for Physics & Astronomy

ML is revolutionizing nearly every field of science & society more broadly

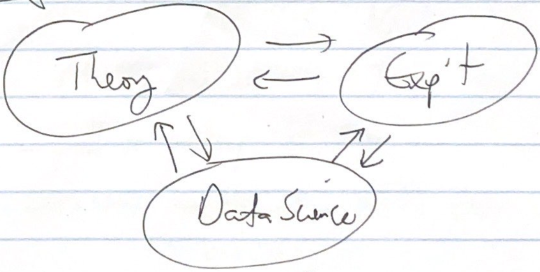
Powerful new tool
- enabling new analyses previously impossible
- enhancing sensitivity & precision
- accelerating simulation & inference
- unifying solutions to problems across domains

↓

made possible by
- Big data
- Computing (GPU)
- Sophisticated algorithms (NNs, ...)

ML like a telescope! or microscope

~~This course~~:



↓ formal mathematics

~~This course~~:

novel mix of ~~theory~~, statistics, numerics

in ML same paper can have all 3!

~~lectures M-Th 10:20~~

This course: general ML concepts

popular & state of the art architectures

(I know more ⟶ less)

applications to LHC, Astro, Cosmo, Condensed Matter

- No required HWs but will be hands on tutorials & exercises
  & no exams
- BB + demos/slides
- Will ask people to pitch applications to present in class
  (either you can present, or we can learn it together & I will present)

need input!

- No domain knowledge of any subfield required
- Prior experience w/ python, numpy, matplotlib would be good

please interrupt frequently & ask me lots of ???'s

- lectures M-Th 10:20–11:40
  - poll for makeup slot?
  - in Nov some travel – either Zoom or guest lectures (4 lectures)
- will provide refs & reading on course website; no textbook
- Will not require 568 (future will) but would help if you took it
  will try to go fast in overlapping content (SGD, backprop, NN basics ...)

What is ML?

"learning from data" "glorified curve fitting"

- Data: $\vec{x}_i \in \mathbb{R}^d$ ← d can be high! e.g. $10^2, 10^3, \dots$

  ↳ a vector of features

  features could include labels $\vec{y}_i$ e.g. cat vs dog, ...

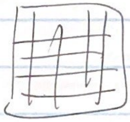
"low level" & vs "high level" features

↳ "tabular data"

e.g.

particle momenta @LHC or $x, y, z, v_x, v_y, v_z, G, B-R, \dots$ @Gaia

or



pixel intensities in image

$i = 1, \dots N$ # of instances → N could be large $10^6, 10^9, \dots$

(e.g. events @LHC
stars in Gaia
images ~~of~~ of galaxies)

- Assume in most applications: each $\vec{x}_i \sim p_{data}(x)$ drawn iid

parameters ← fit fn $f(\vec{x}_i;\theta)$ to data

- Goal: minimize some "loss" or "objective" fn

$$L(\theta) = \sum_{i=1}^{N} \mathcal{L}(f(\vec{x}_i;\theta), \vec{x}_i)$$

wrt parameters θ.

target labels
- discrete "classification"
- continuous" regression

Example: Maybe want $f(x_i;\theta) = y_i$

Could use $\boxed{\mathcal{L}(\theta) = (f(x_i;\theta) - y_i)^2}$

"mean squared error (MSE)"

- many more examples to follow (& how to choose loss fn in principled way)

- Where do labels come from?
  - if data is simulated have access to "ground truth"
  - in actual data — human labeling? (common in real world apps like Imagenet, also Astro citizen science)
  - no labels or labels derived from data itself

DNN, RNN, CNN, graph NNs transformers, ... → building blocks

Categories of ML

pretty much solved problem in ML
- fully supervised — all data labeled & used in training
  - classification, regression

This is where the active ML development is happening
- unsupervised — no labels!
  - generative modeling, density estimation, some kinds of anomaly detection
- weakly supervised — noisy labels
- semi supervised — mix of labeled & unlabeled data
- self supervised — labels generated from data e.g. contrastive loss

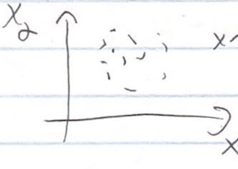
less-than-supervised

Also really important for science — strive to be as <u>data-driven</u> as possible

(simulation-free)

In more detail:

- Generative modeling (GANs, VAEs, ~~n~~ normalizing flows, diffusion, ...)



$x \sim p_{data}(x)$

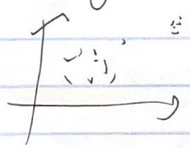
learn to sample from $p_{data}(x)$

- Density estimation — learn $p_{data}(x)$ (normalizing flows, ...)

(Gen modeling $\not\Leftrightarrow$ Density est)

one doesn't guarantee the other

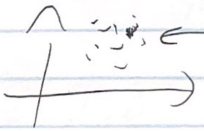
- Anomaly detection



← outlier detection

"point anomalies"

(autoencoders, ...)



← overdensity detection

"group anomalies"

General principle for loss fns: Maximum Likelihood Estimation (MLE)

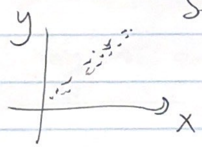
want to maximize $P(\text{data} \mid \underset{\theta}{\text{model}})$

if data iid $\quad = \prod_{i=1}^{N} P(x_i \mid \theta)$

$$\boxed{L = -\log P(\text{data} \mid \text{model}) = -\sum_{i=1}^{N} \log P(x_i \mid \theta)}$$

- MLE nice properties in lim of large N:
  - consistency (as $N \to \infty$, estimated $\theta \to$ true θ)
  - efficiency (as $N \to \infty$, minimum variance estimator of θ)

Example:



Suppose want to predict y given x (regression)

model: y is gaussian dist'ed around $f(x;\theta)$ w/ some width σ

then $\boxed{P(y \mid x, \theta) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(y - f(x;\theta))^2}{2\sigma^2}}}$

$$\boxed{L = -\log P = \sum \frac{(y_i - f(x_i;\theta))^2}{2\sigma^2} + \log\sqrt{2\pi\sigma^2}}$$

if σ known $\to$ MSE! (otherwise also fit for σ)

Ex: binary classification

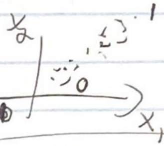
~~what dist bin~~

labels $y_i = 0$ or 1 from data x_i

model: $f(x_i;\theta) =$ prob of x_i being label 1.

data: $\vec{x}_i$ label 1, $i=1,\dots,N_1$

$\vec{x}_{0i}$ label 0, $i=N_1+1,\dots,N_1+N_0$



$$\boxed{P(\text{data}\mid\text{model}) = \prod_{i=1}^{N_1} f(x_i;\theta) \prod_{i=N_1+1}^{N_1+N_0} \left(1-f(x_i;\theta)\right)}$$

$$L = -\log P = -\sum_{i\in 1} \log f(x_i;\theta) - \sum_{i\in 0} \log\left(1-f(x_i;\theta)\right)$$

$$= -\sum_i \left(y_i \log f(x_i;\theta) + (1-y_i)\log\left(1-f(x_i;\theta)\right)\right)$$

"binary cross entropy loss" → best loss for classification

→ could use MSE

but it would be suboptimal