Tuesday, May 25, 2021 9:00 AM

Deep learning Applications to LHC

Recent years: Deep learning revolution

- Image recognition
- Image generation
- Natural language processing (text)
- Speech recognition
- Games.

⋮

Revolution is coming to HEP & LHC & many fields (Big Data)

Tuesday, May 25, 2021 9:09 AM

Theme of the course: "Applied AI" (not to much "theory" - statistics - proofs - ML theory)

- Introduction to modern DL algorithms
  - DNNs
  - CNNs
  - RNNs
  - GANs
  - Autoencoders
  - Normalizing flows
  - ⋮
- Core ML concepts
  - supervision levels
  - tasks (classification, regression ...)
  - statistics (max likelihood, Neyman Pearson lemma)
  - NN training

Tuesday, May 25, 2021 9:10 AM

- LHC applications
  - jet tagging ← Lecture II: Classifiers
  - calorimeter simulation ← Lecture III: Generative Modeling
  - searches for new physics ← Lecture IV: Anomaly Detection.

PLEASE INTERRUPT ME OFTEN WITH QUESTIONS!!!

Tuesday, May 25, 2021 9:16 AM

What is machine learning?

- "Glorified Curve Fitting"
  - Fitting functions with millions (even billions) of parameters to enormous data sets

many data points + many features/data point

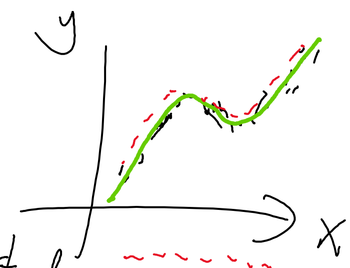
e.g. millions of images with 1000x1000 pixels each

data pts

features

How is this possible?

with fitted function, can do many things: interpolation



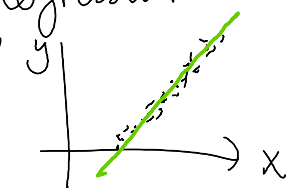
(extrapolation is much harder & requires a deeper understanding than just fitting to the data)

Tuesday, May 25, 2021 9:27 AM

Categories of ML tasks:

- Regression

"target"
"output"
"labels"



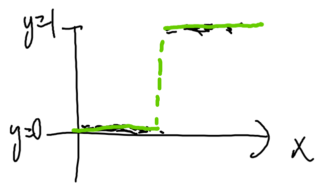
input data

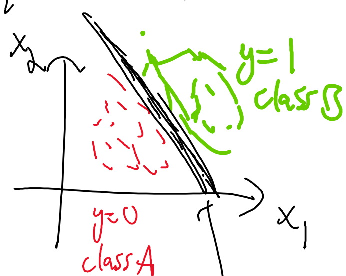
y are continuous

fit fn $f(x; w)$ ← fit parameters "weights"

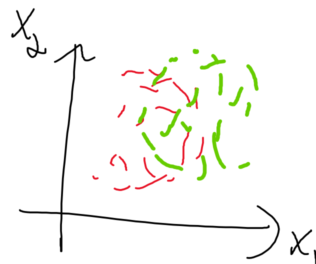
- Classification: regression w/ discrete labels

Ex





"decision boundary"



fit fn: $f_a(x; w)$ = "prob x has label a"

"supervised ML"

training with labeled data

↓ LHC

simulating QCD events
ttbar
W+jets
new physics...

Tuesday, May 25, 2021 9:41 AM

$\boxed{\text{density estimation}}$

learning underlying probability density P(x) that data was drawn from.

- $\int f(x;w)\, d^n x = 1$
- $f(x;w) \geq 0$

unsupervised ML.

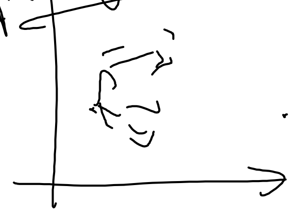
training without labels.

- Suppose I've learned P(x) (either implicitly or explicitly)

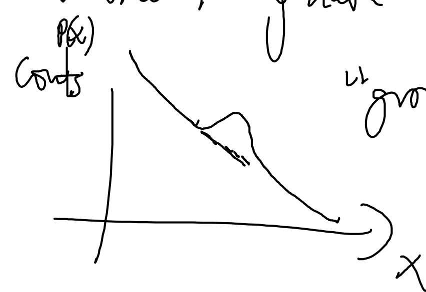
→ sample from it → $\boxed{\text{generative modeling}}$

produce more "realistic" examples of the training data

$\boxed{\text{Anomaly Detection}}$.



anomaly detection ← "outlier detection"

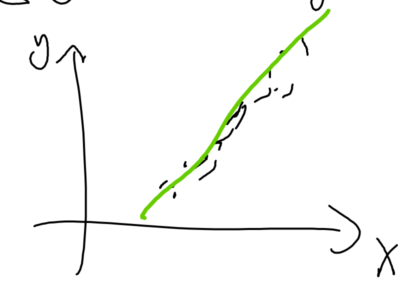


"group anomaly detection"

looking for overdensities.

Tuesday, May 25, 2021 9:41 AM

Ex: linear regression



weights $w = a, b$

$f(x; w) = ax + b$

How to fit to data?

1st thought: least squares fitting!

$$L(w) = \sum_{i=1}^{N} \left(y_i - f(x_i; w)\right)^2$$ ← "loss function" "objective function"

minimize L wrt w: $0 = \frac{\partial L}{\partial a} = \frac{\partial L}{\partial b}$ → solve for a & b.

Why this form of loss fn?

→ principle of Maximum Likelihood Estimation → from statistics

$L = P(\text{data} \mid \text{model})$

max model L → "best" estimator for model parameters

for least-squares, assume model

$$P(y \mid a, b, \{x\}) \sim e^{-\frac{(y - (ax+b))^2}{2}}$$

Tuesday, May 25, 2021 10:21 AM

$$L = \prod_{i=1}^{N} p(y_i \mid a, b, x_i)$$

$$-\log L = \frac{1}{2} \sum_{i=1}^{N} (y_i - a x_i - b)^2$$

in practice, more practical to minimize negative log likelihood (NLL).

Ex: Binary Classification — loss fn under MLE

model: $f(x; w)$ = prob x is class $A = p(A \mid x)$

$$L = P(\text{data} \mid \text{model}) = \prod_{i \in A} f(x_i; w) \prod_{j \in B} (1 - f(x_j; w))$$

"binary cross entropy"

$$\boxed{\begin{aligned} -\log L &= -\sum_{i \in A} \log f(x_i; w) - \sum_{j \in B} \log(1 - f(x_j; w)) \\ &= -\sum_{i \in \text{data}} \left[y_i \log f(x_i; w) + (1 - y_i) \log(1 - f(x_i; w)) \right] \end{aligned}}$$

$y_i = 0$ for B
$y_i = 1$ for A

Tuesday, May 25, 2021 10:26 AM

Ex: multiclass classification

N_C classes $\rightarrow$ N_C fitting fns $f_a(x;w)$ = prob x is class a

$a = 1, \ldots N_C$

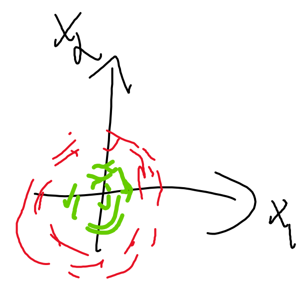
$$\sum_{a=1}^{N_C} f_a(x;w) = 1$$

loss fn $$\boxed{-\log L = -\sum_{a=1}^{N_C} \sum_{i=1}^{N} y_a \log f_a(x_i; w)}$$ "categorical cross entropy"

What should $f(x;w)$ be?

1. "High level features" — examine the data & come up with something "by hand"

e.g.

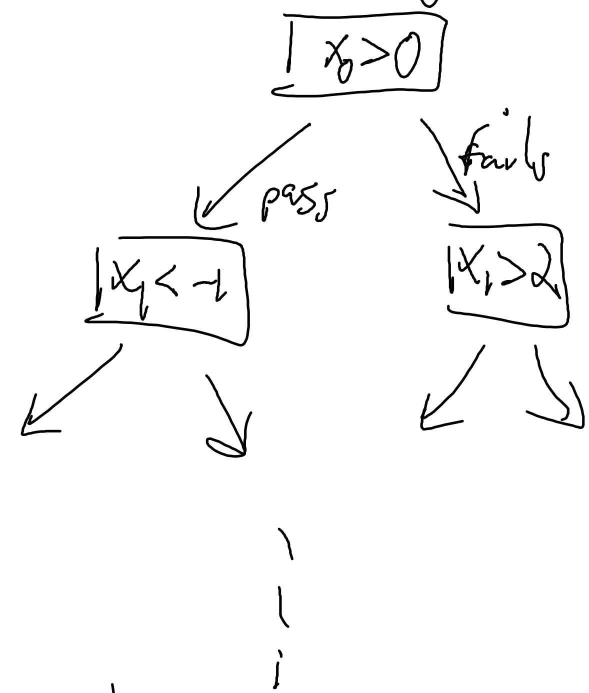


$r = \sqrt{x_1^2 + x_2^2}$

Tuesday, May 25, 2021 10:35 AM

2. (Boosted) Decision Trees (BDTs) — automating the choosing of cuts

(primarily for classification)



- Very popular choice in HEP for ML classification
  - very fast & easy to train for smallish # of inputs (e.g. # of X features ~ 10)
  - if dimensionality of data $\gtrsim 10$s then BDTs don't scale well.

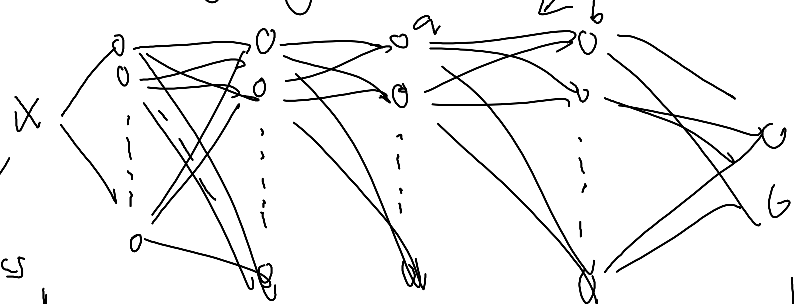
Tuesday, May 25, 2021 10:42 AM

3. Neural Networks — scale well to higher-dimensional data

Biologically Inspired

weights w_{cb}

input



represents a big vector

hidden layers

output

"Dense NN" (DNN) or "Fully connected NN"

"feed forward NN"

each layer $\vec{x}^{(j)} = \begin{pmatrix} x_1^{(j)} \\ \vdots \\ x_{n_j}^{(j)} \end{pmatrix}$

$j=0$ input

$j=1$ 1st hidden layer n_j

$\vdots$

layer $j \to j+1$

$W^{(j)} \vec{x}^{(j)} + \vec{b}^{(j)}$

matrix mult. $n_j \times n_{j+1}$ dim'l matrix

"biases"

Tuesday, May 25, 2021 10:48 AM

- Nonlinear Activation

$j \to j+1$

$$\boxed{x^{(j+1)} = A_j\left(W^{(j)} x^{(j)} + b^{(j)}\right)}$$

↓

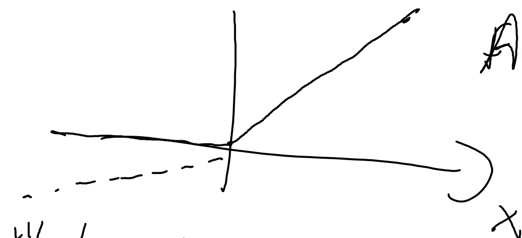
acts <u>elementwise</u> on vector input

sigmoid $A(x) = \dfrac{e^x}{1+e^x}$

vanishing gradient problem

tanh

ReLU



$$A(x) = \begin{cases} x & \text{if } x > 0 \\ 0 & \text{if } x < 0 \end{cases}$$

"leaky ReLU"

"best practices"

- Fully connected NN w/ nonlinear activation fn

↓

"Universal approximation theorem"

→ "an infinitely wide, single layer NN can approx "any" function to arbitrary accuracy"

Tuesday, May 25, 2021 10:58 AM

Idea of proof

show that NN can approx this fn. → build "any" fn out of these

Univ. approx. thm is part of "why NN's "work" — they are an incredibly expressive family of fitting functions.