Lecture II

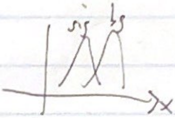
- email list
- slack

- pitches for applications

Neyman-Pearson Lemma

$$L = -\left(\sum_{i \in S} \log f(x_i) + \sum_{i \in B} \log(1 - f(x_i)) \right)$$

what f minimizes L in ideal case?



continuum limit (∞ data

prob density of sg & bg

$$L = -\int dx \left[p_S(x) \log f(x) + p_B(x) \log(1 - f(x)) \right]$$

(assume $N_S = N_B$ WLOG)

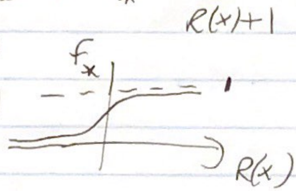
$\int dx\, p(x) \approx \sum_{x \sim p(x)}$

$$f \to f + \delta f, \quad \left. \frac{\partial L}{\partial \delta f} \right|_{f = f_*} = 0 \quad \leftarrow \text{optimal } f$$

$$0 = \frac{p_S(x)}{f_*} - \frac{p_B(x)}{1 - f_*} \implies \boxed{f_* = \frac{p_S(x)}{p_S(x) + p_B(x)}}$$

So optimal classifier: $f_* = \dfrac{R(x)}{R(x) + 1}$

$$R(x) = \frac{p_S(x)}{p_B(x)}$$



f_* is monotonic w/ $R(x)$

$$\implies \boxed{f_* > f_c \iff R(x) > R_c}$$

cut on f equiv to cut on R

So optimal classifier is likelihood ratio

"Neyman-Pearson Lemma"

- powerful result! fundamental result in statistics

(LR uniformly most powerful test for simple hypothesis)

later will see:

- Can turn around → learn ML classifier from samples → approach LR.

"likelihood ratio trick"

- Another perspective: Bayes Thm

$$f_* = \frac{p_S(x)}{p_S(x)+p_B(x)} = \frac{p(x|S)}{p(x|S)+p(x|B)} = \frac{p(x|S)\,\overset{\frac{1}{2}}{p(S)}}{\underbrace{p(x)}_{=\frac{p(x|S)+p(x|B)}{2}}}$$

$$= p(S|x)$$

So f_* is the prob of signal given x which is where we started!

i.e. BCE is minimized when our model for this prob = true prob. ✓

Back to NP lemma: what does "most powerful" test mean?

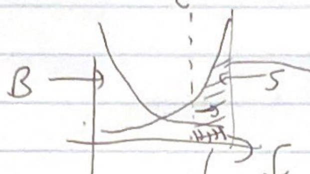
→ <u>metrics for binary classification</u>

How do we use binary classifier?

Usually: cut $f(x) > f_c$ — all x satisfy cut are "signal" (classified as)

all x fail cut are "bg"

"working pt"

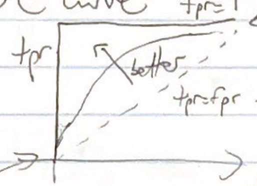


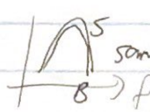
$tpr = \dfrac{N_S(f > f_c)}{N_S}$ $\quad fnr = 1 - tpr$

$fpr = \dfrac{N_B(f > f_c)}{N_B}$ $\quad tnr = 1 - fpr$

- accuracy $= \max\limits_{f_c} \dfrac{tpr + (1 - fpr)}{2}$

- ROC curve





Vary f_c, sweep out "ROC curve" $(fpr(f_c), tpr(f_c))$

→ AUC "Area under the curve"

AUC = 1 perfect
AUC = 0.5 random

(same for ACC but AUC ≠ ACC)

- ~~often~~ AUC, ACC mostly sensitive to $tpr, fpr \sim O(1)$ part of ROC curve when $N_{bg} \gg N_{sig}$ (as in many cases, eg LHC)

care more about $fpr \ll 1$ part of ROC curve

→ often report fpr @ fixed tpr eg tpr = 50%

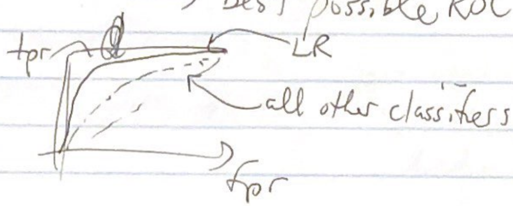
or 30%

or rejection factor

$$\begin{cases} R50 = \dfrac{1}{fpr@tpr=50\%} \\ R30 \text{ etc} \end{cases}$$

- Neyman–Pearson LR classifier is optimal

  → best possible ROC curve



- In practice cannot achieve NP optimality
  - exact likelihoods unknown
  - finite training data → era of Big data
  - limited model capacity (expressivity)

    ↳ NNs very ~~powerful~~ expressive

    ⇒ can maybe get very close to optimal!

Neural Networks

- So far have not specified family of fit fns $f(x;\theta)$
- Many choices (BDTs, SVMs, Boltzmann machines, ...)

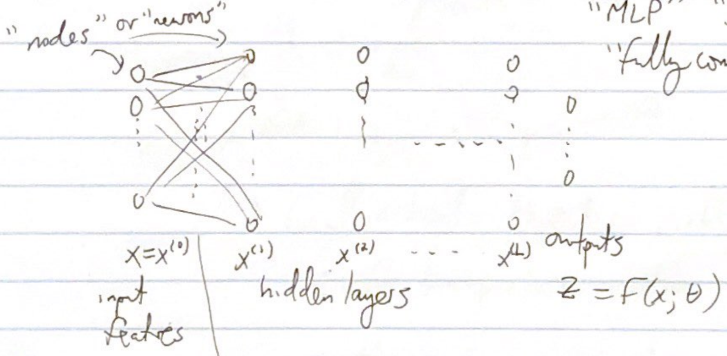
→ deep learning / "modern ML" : $f(x;\theta) = NN$

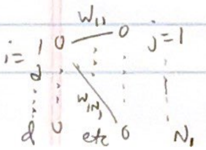
why? — universal approx thm : roughly, "NN can approximate any fn"

— differentiable (extremely large models can be trained on a lot of data using backprop & SGD)

— not well understood, but surprisingly resistant to overfitting "generalization puzzle" "inductive bias"

- Basic structure of NN "feed-forward" "MLP" "DNN" "fully connected"





each connection: "weight"
all together: weight matrix W_{ij}

"biases"

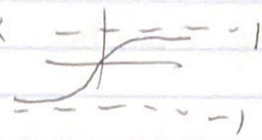
$$x^{(1)} = A^{(1)}\left(W^{(1)}x^{(0)} + b^{(1)}\right)$$

↳ "activation"

• activation: only source of nonlinearity.

examples: $A(x) = \frac{e^x}{1+e^x}$ "sigmoid"

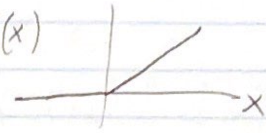
$= \tanh x$



best choice!

others lead to "vanishing gradient problem" → more later

$= ReLU(x)$



$= \begin{cases} x, & \text{if } x>0 \\ 0, & \text{if } x<0 \end{cases}$

• Note: activation acts element wise $A^{(1)}\begin{pmatrix}x_1\\x_2\\ \vdots \\ x_n\end{pmatrix} = \begin{pmatrix}A^{(1)}(x_1)\\ \vdots \\ A^{(1)}(x_n)\end{pmatrix}$

• Structure of NN is recursive

$$x^{(n)} = A^{(n)}\left(W^{(n)}x^{(n-1)} + b^{(n)}\right)$$

$$z = x^{(L+1)} = A^{(L+1)}\left(W^{(L+1)}x^{(L)} + b^{(L+1)}\right)$$

↳ final activation depends on problem

eg for binary classification $z = \begin{pmatrix} p(S|x) \\ p(B|x) = 1-p(S|x)\end{pmatrix}$

want outputs to sum to 1

popular choice: softmax $\left(\frac{e^{x_1}}{\sum_{j=1} e^{x_j}}, \frac{e^{x_2}}{\sum e^{x_j}}\right)$