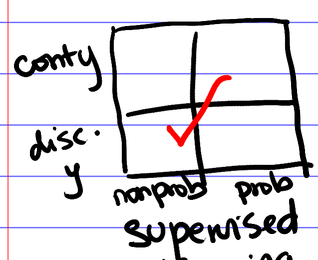
CS181 - Neural Nets, Day 1



supervised learning

and then:
- model selection
- more expressive models TODAY!
- more losses

Ch. 4 upto ~4.5 ish

also: HW2 due Friday; cc's online

So far: we talked log. reg. for classification

$p(y=1|x) = \sigma(+(\underline{w}^T\underline{x} + w_0))$ ← created a linear boundary

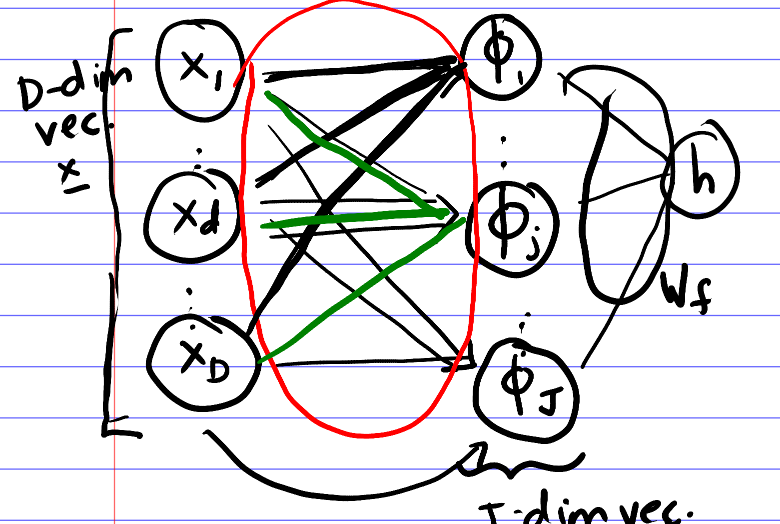
$\underline{x} \to \phi(x)$, we can apply log. reg. to $\phi(x)$: $p(y=1|x) = \sigma(+(w^T\phi(x) + w_0))$

$\phi(x)$ were hand-designed!

today: going to learn $\phi(x)$ [adaptive basis regression]

notation: $h = w^T\phi + w_0$ ← reg. output or input into a sigmoid for classification

now, let's choose a form for ϕ



$\phi_j = \sigma(\underline{w}_j^T \underline{x} + w_{0,j})$

$\underline{w}_j$: D dim vec.

there are J w_j vectors, one per ϕ_j

$\underline{\Phi}_n = \sigma(W^1 x_n + \underline{w}_0^1)$ ← vec. of length J

σ: pointwise application of σ

W^1: matrix $J \times D$ of all $\underline{w}_j$ vecs

thought exercise: what if we didn't have the σ?

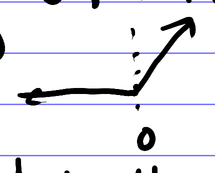
$$\Phi = W^1 x_n + \underline{w}_0^1$$

$$h = \underline{w}_f^T \phi + w_0^f$$

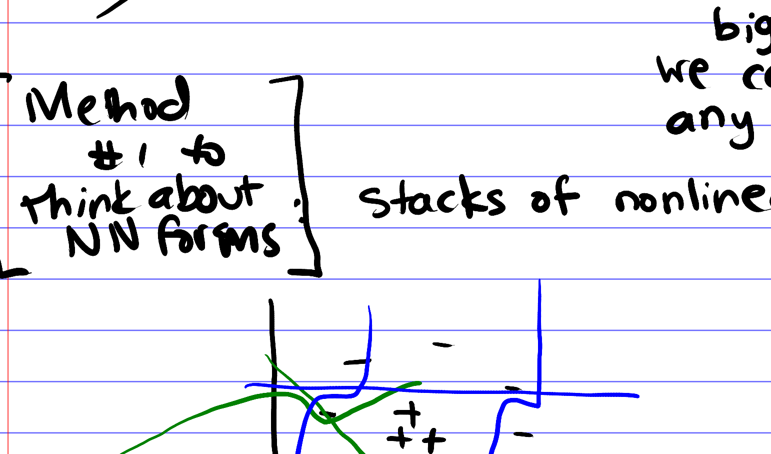
$$= \underbrace{\underline{w}_f^T W^1}_{\text{D-dim vec}\ w^{T\prime}} x_n + \underbrace{\underbrace{\underline{w}_f^T \underline{w}_0^1}_{\text{scalar}} + w_0^f}_{\text{scalar}\ w_0'}$$

without the nonlinearity, we just get a linear form!

Note: we have many choices for this nonlinearity: σ, tanh, rectified linear unit (ReLU)

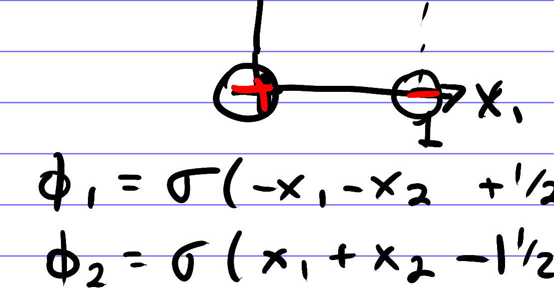


↳ good in the sense that they can be universal function approx.

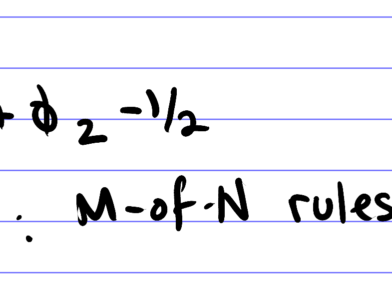


if we had enough sigmoids (wide enough network, big enough J) we can approx any function.

[Method #1 to think about NN forms]: stacks of nonlinearities



Ex #2:



$\phi_1 = \sigma(-x_1 - x_2 + 1/2)$ ← positive only for (0,0)

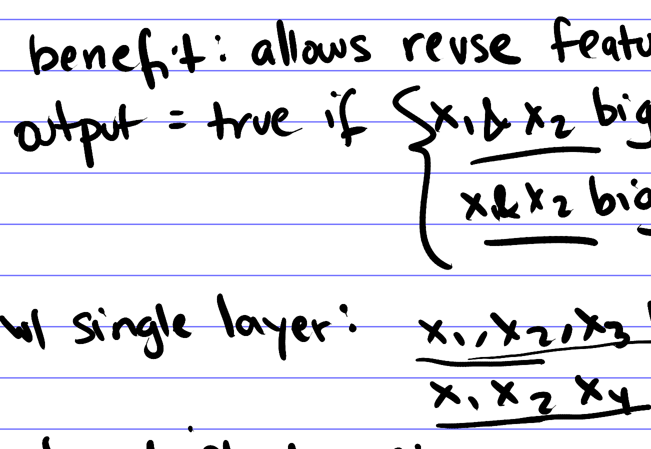
$\phi_2 = \sigma(x_1 + x_2 - 1\tfrac{1}{2})$ ← positive only for (1,1)

$h = \phi_1 + \phi_2 - 1/2$

[Method #2 think about NNs]: M-of-N rules.

single layer ↑

1. Deep, fully connected network



[feed-forward neural net]

... (could have many layers!)

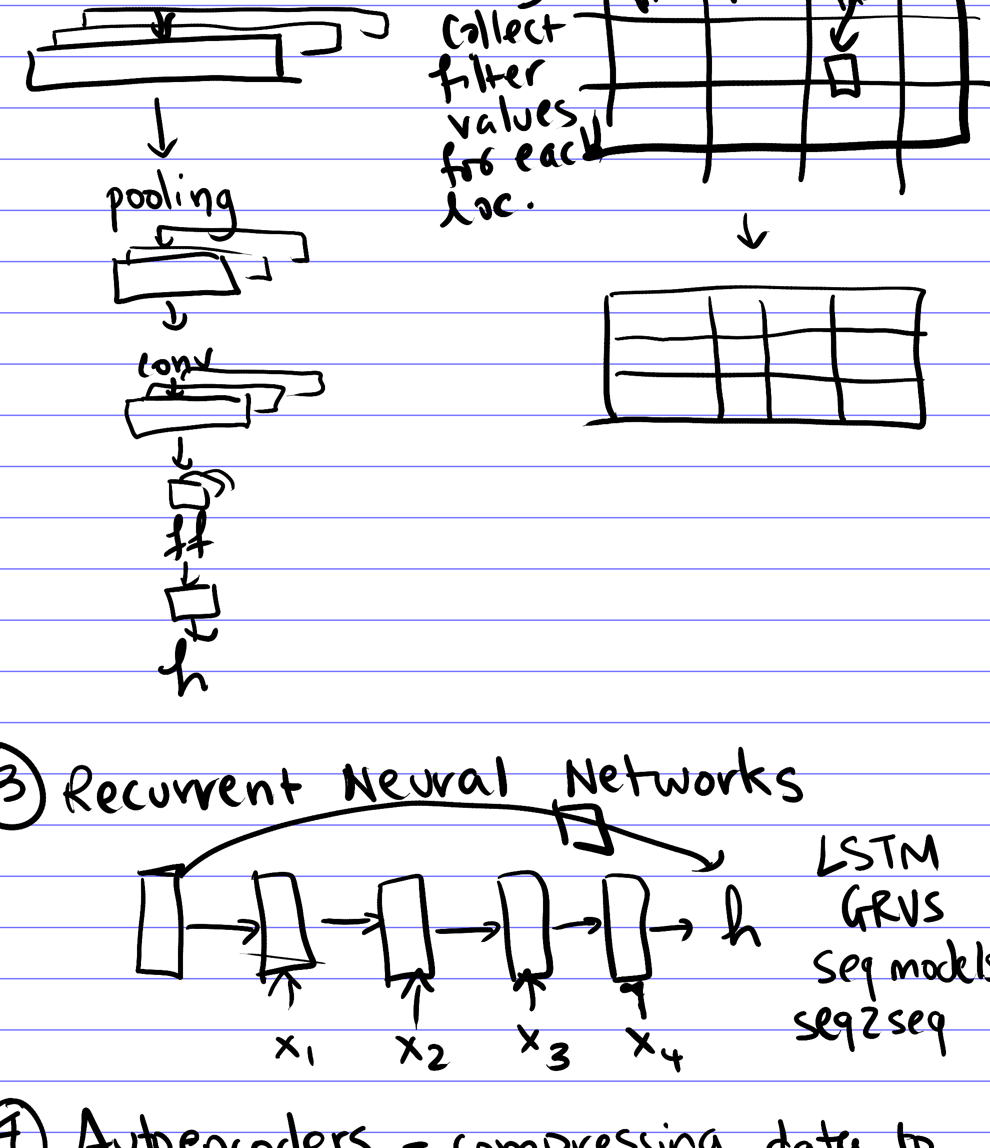
benefit: allows reuse features

output = true if $\begin{cases} x_1 \,\&\, x_2 \text{ big AND } x_3 \text{ big} \\ x_1 \,\&\, x_2 \text{ big AND } x_4 \text{ big} \end{cases}$

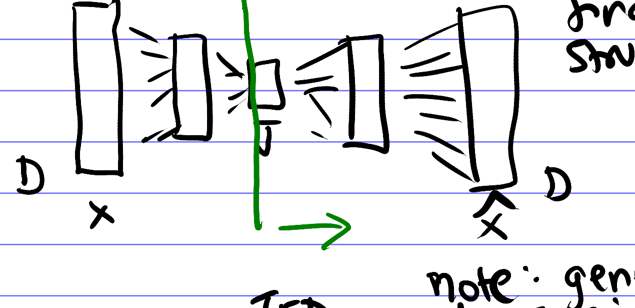
w/ single layer: x_1, x_2, x_3 big; x_1, x_2, x_4 big

w/ multiple layers: $\phi_1' = x_1, x_2$ big

2. Convolutional Network

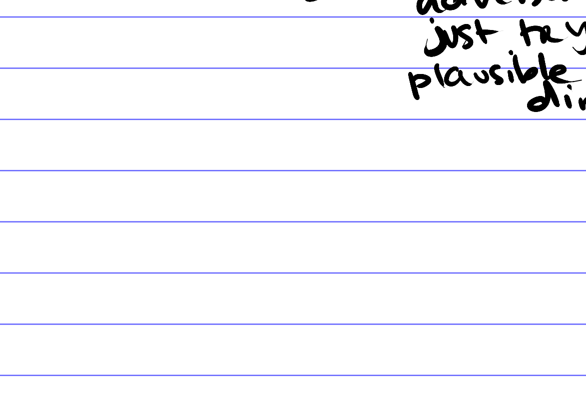


3. Recurrent Neural Networks



LSTM
GRUs
seq models
seq2seq

4. Autoencoders - compressing data to find structure



note: generative adversarial networks just try to produce plausible x's from low dim J → D