MAP : maximize w/ respect to $\vec{w}$ $\quad \mathcal{P}(\vec{w}|\mathcal{D}) = \dfrac{\mathcal{P}(\mathcal{D}|\vec{w})\,\mathcal{P}(\vec{w})}{\mathcal{P}(\mathcal{D})}$

($\vec{w}$ = model params)

$\Rightarrow$ find "best" set of params $\vec{w}^*$

example: machine learning classification

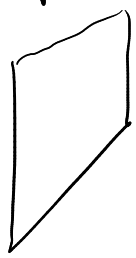


image of cat or dog (100 x 100 px x 3 colors)

$\Rightarrow$ input vector $\vec{x}$ (length 3×10^4) $\rightarrow$ [function (neural net)] $\rightarrow$ prob. vector $\vec{p} = \begin{pmatrix} p_1 \\ p_2 \end{pmatrix}$

input

2 categories

p_i = prob. that input is in category i

What does the neural net func. look like?

one example: dense neural net (DNN)

or multilayer perceptron

to find output $\vec{p}$ we do a series of operations:

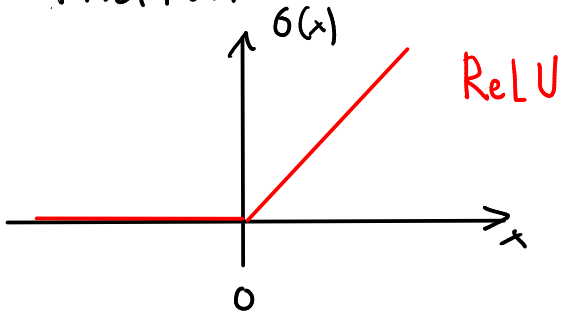
$$\vec{z}^{(1)} = \sigma\left(A^{(1)} \vec{x} + \vec{b}^{(1)} \right)$$

$\vec{z}^{(1)}$: vector; $A^{(1)}$: matrix (could be rectangular); $\vec{b}^{(1)}$: vector

one "layer" of net

$\sigma(x)$ = nonlinear "threshold" function

example: $\sigma(x) = \max(0, x)$



$$\sigma(\vec{x}) = \left(\sigma(x_1), \sigma(x_2), \ldots \right)$$

next layer: $\vec{z}^{(2)} = \sigma\left(A^{(2)}\vec{z}^{(1)} + \vec{b}^{(2)}\right)$

$\circ \circ \circ$

$\vec{z}^{(n)} = \sigma\left(A^{(n)}\vec{z}^{(n-1)} + \vec{b}^{(n)}\right)$ n th layer

$\downarrow$

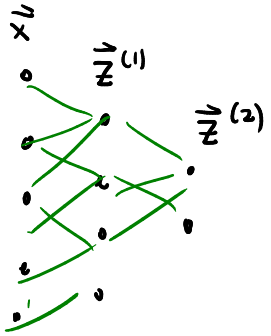
same dim. as # categories

overall func. depends all the components of $A^{(i)}, \vec{b}^{(i)}$ $i = 1, \ldots, n$

$\Rightarrow$ vector of params $\vec{w}$

final step: convert $\vec{z}^{(n)}$ into prob. $\vec{p}$

"softmax": $p_i = \dfrac{e^{-z_i^{(n)}}}{e^{-z_1^{(n)}} + e^{-z_2^{(n)}} + \cdots}$ $\sum_i p_i = 1$, $p_i \geq 0$

$\vec{x} \Rightarrow$  $\Rightarrow$ soft max $\Rightarrow \vec{p}(\vec{x}; \vec{w})$

universal approx. theorems:

in the case of DNNs if # layers + size of layers is large enough

$\Rightarrow$ you can approx. any func. that takes input from finite domain

Bayesian interpretation:

dataset $\mathcal{D}$: "training" data for net

"image"	label:
$\vec{x}^{(1)}$	$\ell_1 = \begin{cases} 1 \\ 2 \end{cases}$ category
$\vec{x}^{(2)}$	ℓ_2
⋮	⋮
$\vec{x}^{(N)}$	ℓ_N

$$\mathcal{D} = \left\{ (\vec{x}^{(i)}, \ell_i) \mid i = 1, \ldots, N \right\}$$

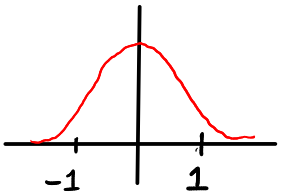
interested in $\mathcal{P}(\vec{w} \mid \mathcal{D})$ ⇒ first step is calculate likelihood $\mathcal{P}(\mathcal{D} \mid \vec{w})$

$$\mathcal{P}(\mathcal{D} \mid \vec{w}) = \prod_{i=1}^{N} p_{\ell_i}(\vec{x}^{(i)}; \vec{w}) \leq 1$$

$= 1$ when perfect classifier

want to maximize $\mathcal{P}(\vec{w} \mid \mathcal{D}) = \dfrac{\mathcal{P}(\mathcal{D} \mid \vec{w})\,\mathcal{P}(\vec{w})}{\mathcal{P}(\mathcal{D})}$ → "prior"?

"regularization": assume net params are not too big (to prevent blow-ups)



$$\mathcal{P}(w_i) = \frac{1}{\sqrt{2\pi}} e^{-w_i^2/2}$$ Gaussian of width 1

$$\mathcal{P}(\vec{w}) = \prod_j \mathcal{P}(w_j)$$

trick: to maximize $\mathcal{P}(\vec{w} \mid \mathcal{D})$ we take $-\log$ of it & minimize

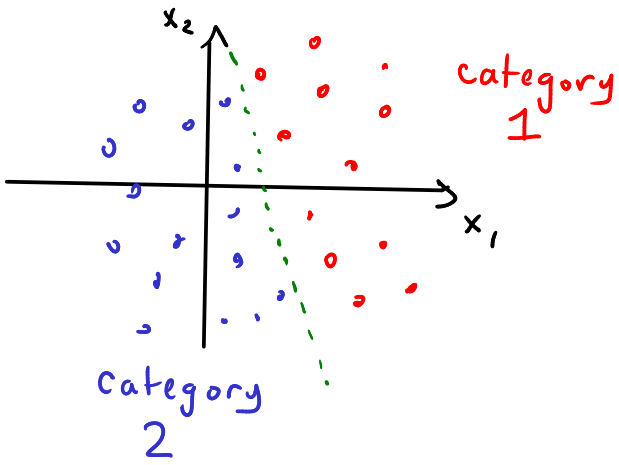
$$\mathcal{L}(\vec{w}) = -\log \mathcal{P}(\vec{w}|\mathcal{D})$$

$$= -\log \mathcal{P}(\mathcal{D}|\vec{w}) - \log \mathcal{P}(\vec{w}) + \underbrace{\log \mathcal{P}(\mathcal{D})}_{\text{const.}}$$

$\Rightarrow$ minimize $\mathcal{L}(\vec{w})$ w/ respect to $\vec{w}$ (i.e. gradient descent)

ignore

prob. 2 of PS #1



$\vec{x} = (x_1, x_2)$

$\rightsquigarrow$

$p_1(\vec{x}; \vec{w})$

= prob. that $\vec{x}$ is in category 1

$p_2(\vec{x}; \vec{w}) = 1 - p_1(\vec{x}; \vec{w})$