supervised

input data $x \rightsquigarrow$ $\boxed{\text{Function } F}$ $\rightsquigarrow$ "desired output" T

Start with lots of examples

Function has unknown parameters

Use examples to adjust parameters to minimize ~~t~~ these "error"

$$F(x) - T$$

Neural net is a function:

x input $\qquad$ y desired output

$$x \longrightarrow \sigma(A, x)$$

A matrix

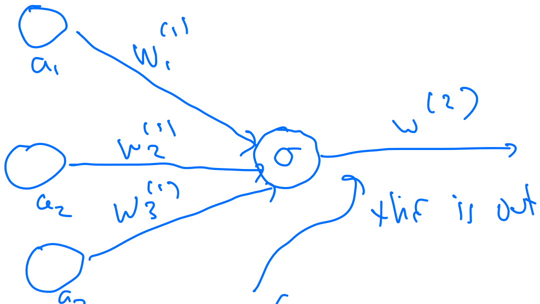
σ simple function applied to each entry

$\left(\sigma = \frac{1}{1+e^{-x}}\right.$, this is logistic regression$\left.\right)$

$$\sigma_3(A_3\,\sigma_2(A_2\,\sigma_1(A_1 x)))$$

$$\sigma(x) = \begin{cases} x & \text{if } x \geq 0 \\ 0 & \text{if } x < 0 \end{cases} \qquad \text{Relu}$$

$$\sigma(x) = \frac{1}{1+e^{-x}} \qquad \sigma(x) = \tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$



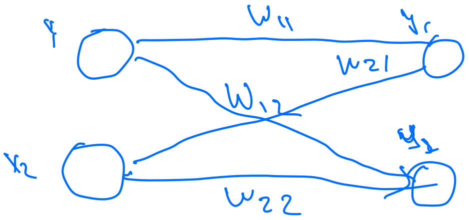
this is output

$\sigma(w_1 a_1 + w_2 a_2 + w_3 a_3)$

---

input vector

$$[x_1 \quad x_2] \begin{bmatrix} w_{11} & w_{12} \\ w_{21} & w_{22} \end{bmatrix} = [y_1 \quad y_2]$$



output layer

dense layer

Input layer

$y_2 = x_1 w_{12} + x_2 w_{22}$

Simple Logistic Regression.

$y \sim \frac{1}{1+e^{-ax-b}}$

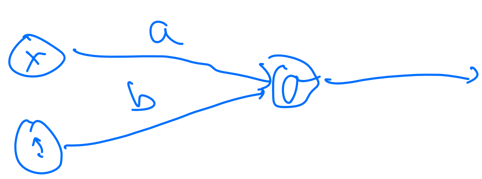
$$X \longrightarrow X \begin{bmatrix} a \\ b \end{bmatrix} \xrightarrow{\sigma} \frac{1}{1+e^{-ax_i-b}}$$

input

$N \times 2$

second column is all ones

$$[x \quad 1]$$



$\frac{1}{1+e^{-ax-b}}$

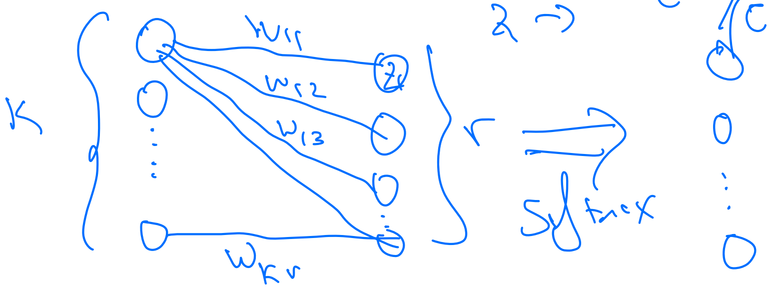
multiclass

$XW \longrightarrow N\times r$

$N\times K \quad K\times r$

$\sigma(XW)$ $\qquad \sigma$ softmax

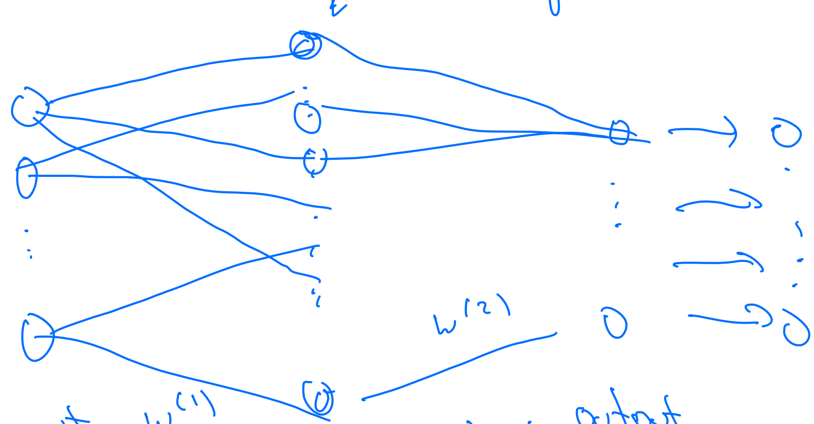
$z \to e^{z_1}/(e^{z_1}+\dots+e^{z_r})$



---

perceptron

hidden layer



input layer $d^{(0)}$ nodes

$W^{(1)}$

$d^{(0)} \times d^{(1)}$ matrix $W^{(1)}$

$d^{(1)}$

$d^{(1)} \times d^{(2)}$ matrix $W^{(2)}$

output layer $d^{(2)}$ nodes

easy by [illegible]

$\sigma_2(\sigma_1(XW^{(1)})W^{(2)})$

O.L.S. as a neural net

$y \sim x_1w_1 + \dots + x_Kw_K$

$\sigma = id.$

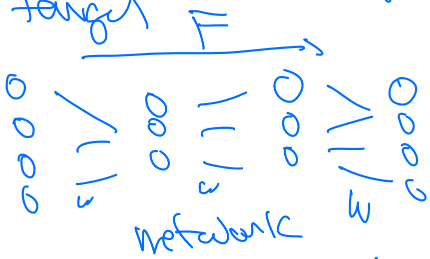
$[x_1 \dots\dots x_K]\, W$

$1\times K \qquad K\times 1$

$\sigma = id$

$\|y - y_{true}\|^2 = \text{Loss}$

Training data, target (x^i, y^i) ← features, target



"Loss" ~ error between target

L compute $\sum_i L(F(x^i), y^i) = L$

Compute ∇L w.r.t weights in F

update weights using gradient

Repeat

1 time through = training epoch

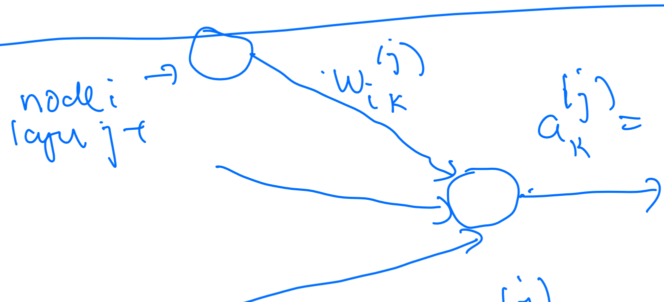
Weights $\longleftrightarrow$ $\underline{W_{ij}^{(k)}}$

k layer
connects node i
in Layer k-1
to node j in layer k

∇

$$w_{ij}^{(k)} \leadsto w_{ij}^{(k)} - \lambda \frac{\partial L}{\partial w_{ij}^{(k)}}$$

λ ↑ learning rate



$a_k^{(j)} = \sigma(z_k^{(j)})$

$z_k^{(j)}$ jth layer
k^{th} node

$$z_k^{(j)} = \sum a_i^{(j-1)} w_{ik}$$

$$a_i^{(j-1)} = \sigma(z_i^{(j-1)})$$

Find

$$\delta_k^{(j)} = \frac{\partial L}{\partial z_k^{(j)}}$$

What is relationship between $\delta_k^{(j-1)}$ and $\delta_k^{(j)}$?

$$\delta_k^{(j-1)} = \frac{\partial L}{\partial z_k^{(j-1)}} = \sum_i \frac{\partial L}{\partial z_i^{(j)}} \frac{\partial z_i^{(j)}}{\partial z_k^{(j-1)}}$$

Side remark

$F(x_1, x_2)$

$x_1 = x_1(t)$

$y_1 = y_1(t)$

$$\boxed{\frac{\partial F}{\partial t} = \frac{\partial F}{\partial x}\frac{\partial x}{\partial t} + \frac{\partial F}{\partial y}\frac{\partial y}{\partial t}}$$

$$z_i^{(j)} = \sum_t a_t^{(j-1)} W_{ti}^{(j)}$$

$$a_t^{(j-1)} = \sigma(z_t^{(j-1)})$$

$$z_i^{(j)} = \sum_t \sigma(z_t^{(j-1)}) W_{ti}^{(j)}$$

$$\delta = \frac{\partial z_i^{(j)}}{\partial z_k^{(j-1)}} = \sigma'(z_k^{(j-1)}) W_{ki}^{(j)}$$

$$\delta_k^{(j-1)} = \sum_i \delta_i^{(j)} \sigma'(z_k^{(j-1)}) W_{ki}^{(j)}$$

$$\delta^{(j-1)} = \sigma'(z^{(j-1)}) W^{(j)} \delta^{(j)}$$