Applied Machine Learning with Big Data
“EE 6973”

Topic:
Deep Learning

Paul Rad, Ph.D.

Chief Research Officer
UTSA Open Cloud Institute(OCI)
University of Texas at San Antonio

Outline

Neural Network Model

Forward Propagation (Prediction)

Backpropagation (Learning)

- Parameters (Weights)
- Training Data
- Cost Function and Error
- Learning Rate

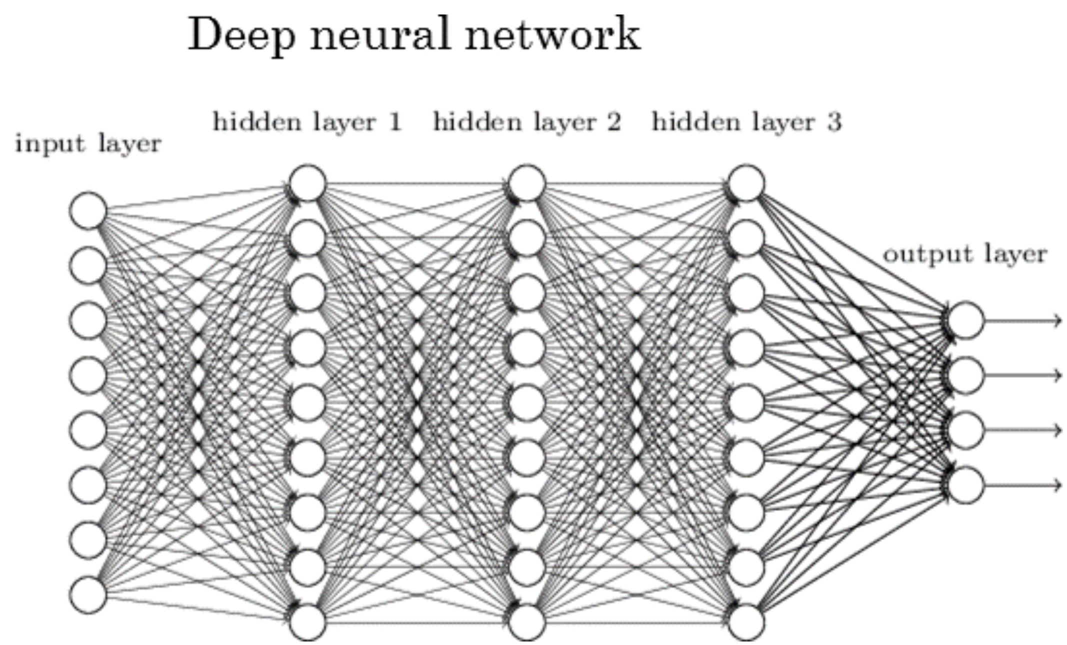
Deep Learning Architecture

An Artificial Neural Network with one or more hidden layers = Deep Learning

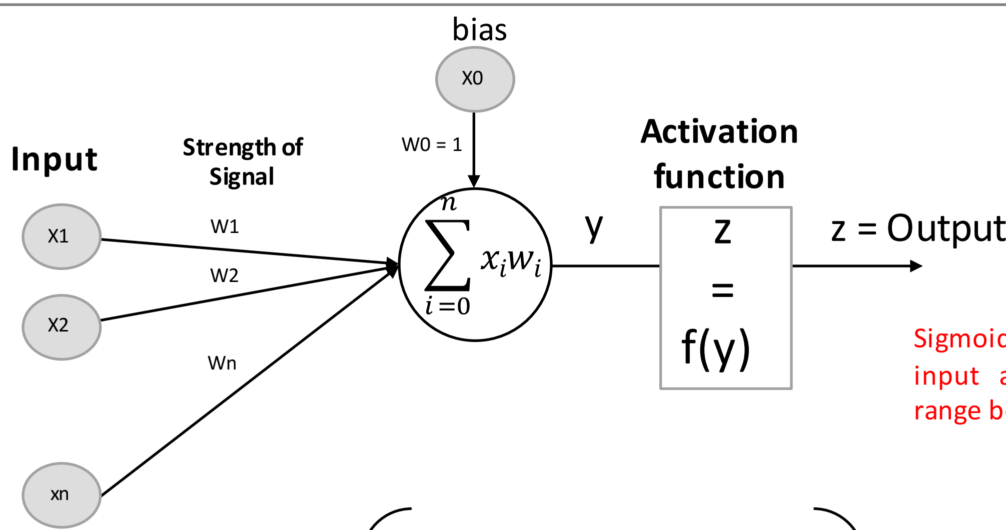
They typically consist of many hundreds of simple processing units which are wired together in a complex communication network.

Each unit or node is a simplified model of a real neuron which fires (sends off a new signal) if it receives a sufficiently strong input signal from the other nodes to which it is connected.

The output is aiming for a target.



Single Layer Neural Networks with Nonlinear Math Model

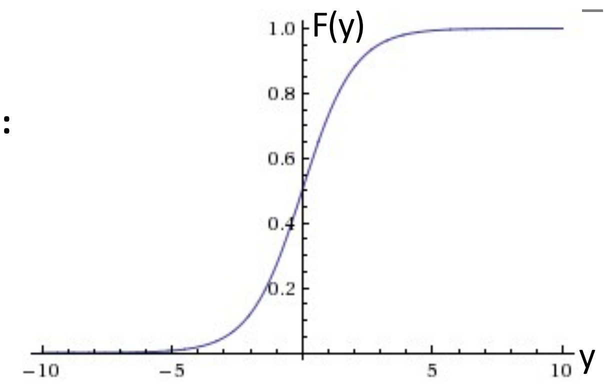


Sigmoid function:

$$f(y) = \frac{1}{1+e^{-y}}$$

$$\left(0 , 1 \right)$$

Sigmoid takes a real-valued input and squashes it to range between 0 and 1

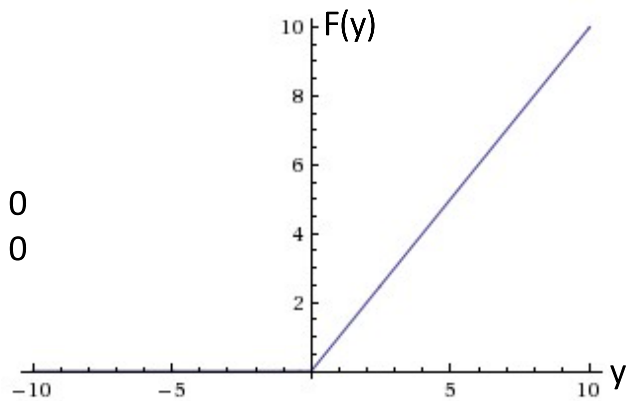


$$\textbf{Output} = f\left(\begin{bmatrix} x0 & x1 & x2 & \dots & xn \end{bmatrix} \times \begin{bmatrix} 1 \\ w1 \\ w2 \\ \dots \\ wn \end{bmatrix} \right)$$

ReLU function:

f(y) =

$$\begin{cases} y & y \Rightarrow 0 \\ 0 & y < 0 \end{cases}$$

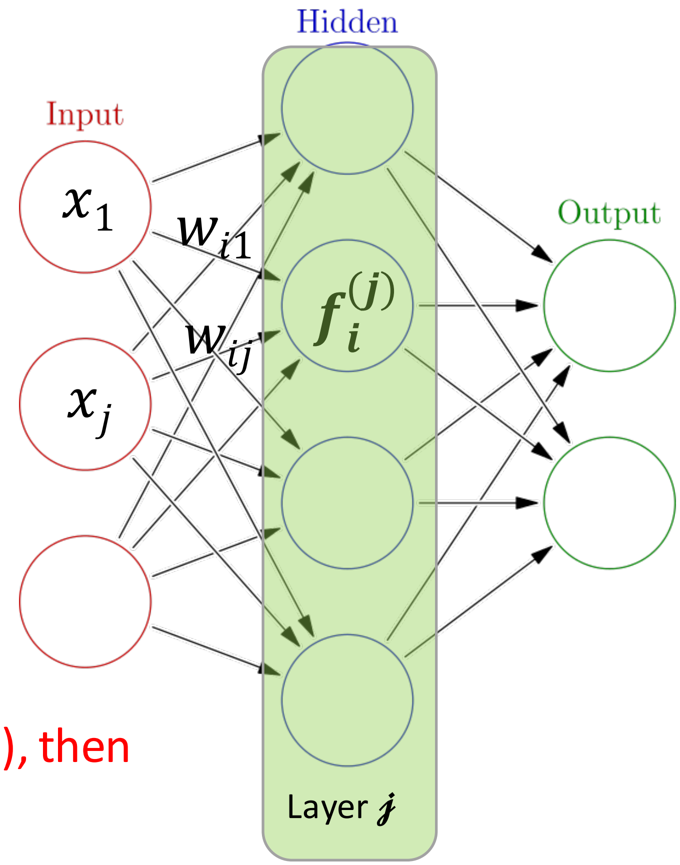


Multi Layer NN Nonlinear Math Model

$f_i^{(j)}$ = "activation function" of unit $\boldsymbol{i}$ in layer $\boldsymbol{j}$

$W^{(j)}$ = matrix of weights controlling function mapping from layer $\boldsymbol{j}$ to layer $\boldsymbol{j}$+1

$$y_i = fi(\sum_{j=0}^{n} w_{ij}\, x_j)$$



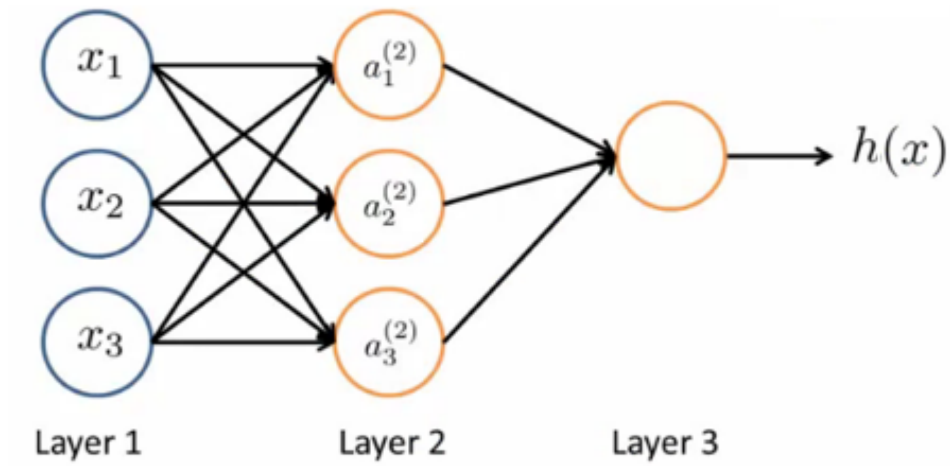
If network has N units in layer $\boldsymbol{j}$, and M units in layer ($\boldsymbol{j}$-1), then $W^{(j)}$ will be of dimension of (M+1) x N

Forward Propagation to Calculate h(x)

$$a_1{}^{(2)} = f\left(\sum_{j=0}^{n} w_{1j}\, x_j\right) =$$

$$a_2{}^{(2)} = f\left(\sum_{j=0}^{n} w_{2j}\, x_j\right) =$$

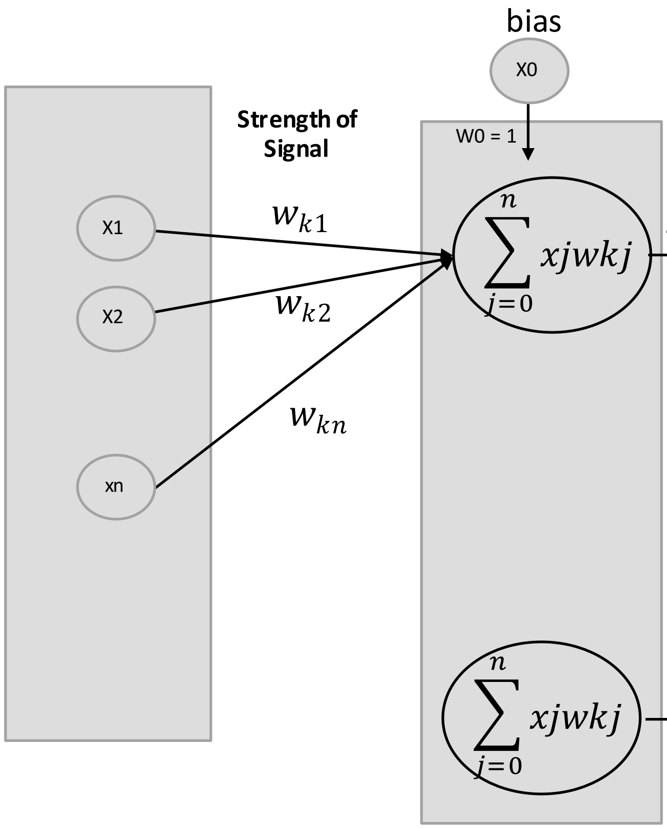
$$a_3{}^{(2)} = f\left(\sum_{j=0}^{n} w_{3j}\, x_j\right) =$$



$$a_i{}^{(2)} = f\left(\sum_{j=0}^{n} w_{ij}\, x_j\right)$$

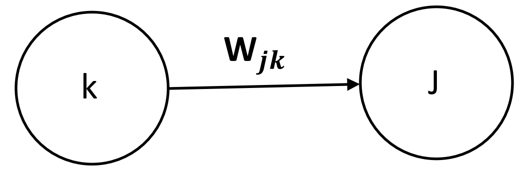
$$\boldsymbol{h\ (x) = f\ (w_3 a_3{}^{(2)} + w_2 a_2{}^{(2)} + w_1 a_1{}^{(2)} + a_0)}$$

Multi Layer NN Nonlinear Math Model



$$y_k = fk\left(\sum_{j=0}^{n} x_j w_{kj}\right)$$

Connection weight from
neuron in layer k to neuron in layer j = $\mathbf{W}_{jk}$



$$\begin{bmatrix} y_1 \\ y_2 \\ \\ y_k \end{bmatrix} = f\left(\begin{matrix} w_{k1} & w_{k2} & & w_{kj} \end{matrix}\right)\begin{bmatrix} x_1 \\ x_2 \\ \\ xj \end{bmatrix}$$

Cost Function

For logistic regression,

$$E\ (\mathbf{w})\ =\frac{1}{N}\sum_{n=1}^{N}\ln\left(1+e^{-y_n\mathbf{w}^{\intercal}\mathbf{x}_n}\right)$$

Compare to linear regression:

$$E\ (\mathbf{w})\ =\frac{1}{N}\sum_{n=1}^{N}(\mathbf{w}^{\intercal}\mathbf{x}_n-y_n)^2$$

Backpropagation Algorithm (Learning)

Initialize all weights $w_{ij}^{(l)}$ **at random**

for $t = 0, 1, 2, \ldots$ **do**

  Pick $n \in \{1, 2, \cdots, N\}$

  Forward: Compute all $x_j^{(l)}$

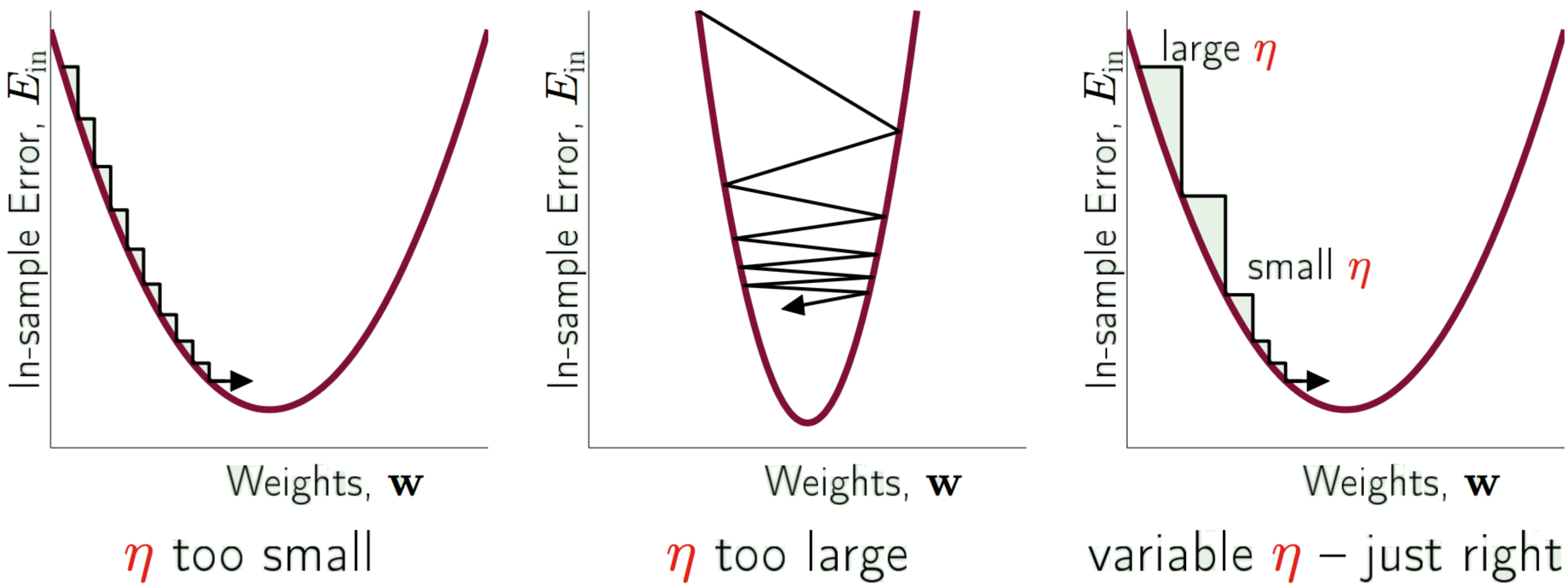
  Backward: Compute all $\nabla \mathbf{e}(\mathbf{w})$: $\dfrac{\partial\ \mathbf{e}(\mathbf{w})}{\partial\ w_{ij}^{(l)}}$ for all i, j, l

  Update the weights: $w_{ij}^{(l)} \leftarrow w_{ij}^{(l)} - \eta\, \nabla \mathbf{e}(\mathbf{w})$: $\dfrac{\partial\ \mathbf{e}(\mathbf{w})}{\partial\ w_{ij}^{(l)}}$ for all i, j, l

  Iterate to the next step until it is time to stop

Return the final weights $w_{ij}^{(l)}$

Learning Rate: Steps

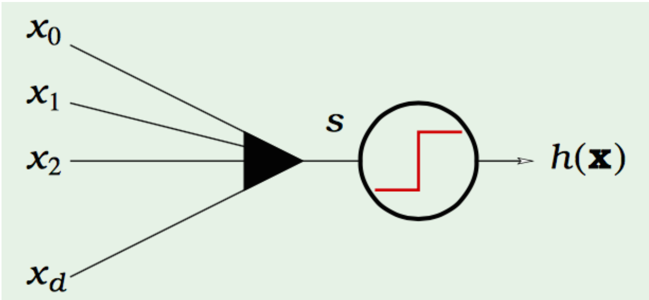


Learning Rate should increase with the slope

Review Models

Linear Classification

h(x) = Sign ($\sum_{i=0}^{n} w_i \, x_i$)



Sign function:

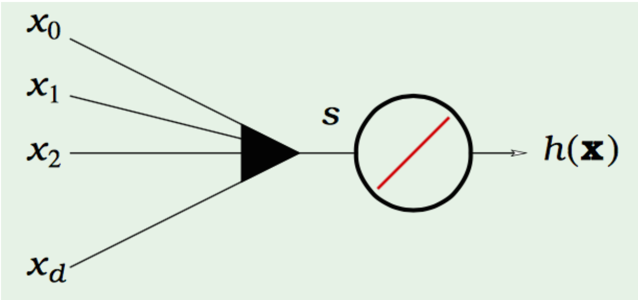
h(s) = 1 s >=0

h(s) = 0 s < 0

Hard Threshold : Certainty

Linear Regression

h(x) = $\sum_{i=0}^{n} w_i \, x_i$

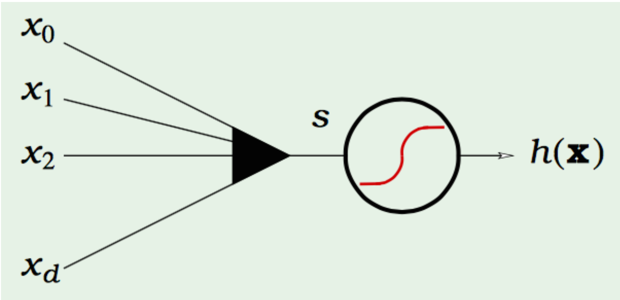


identity function:

h(s) = s

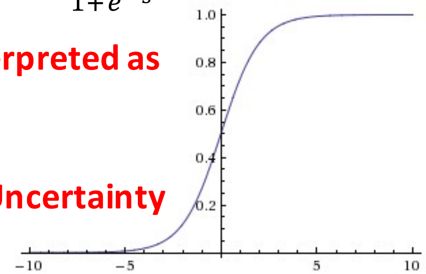
Logistic Regression

h(x) = Sigmoid ($\sum_{i=0}^{n} w_i \, x_i$)



Sigmoid function:

h(s) = $\frac{1}{1+e^{-s}}$



The output is interpreted as probability

Soft Threshold : Uncertainty

Probability Interpretation

h(x) = Sigmoid ($\sum_{i=0}^{n} w_i x_i$) = θ (s) is interpreted as a probability

Logistic Regression

h(x) = Sigmoid ($\sum_{i=0}^{n} w_i x_i$)

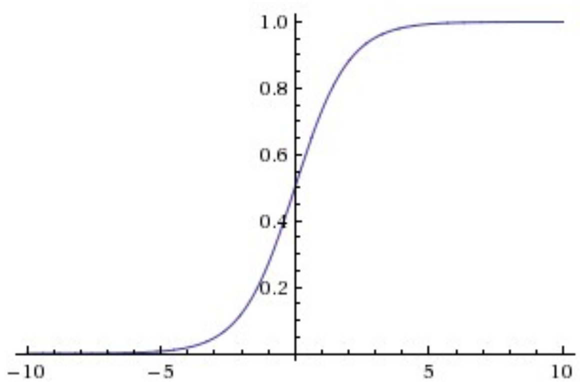
Example: Prediction of heart attacks

Input **X**: x1 =cholesterol level, x2 =patient age, x3 =patient weight, etc.

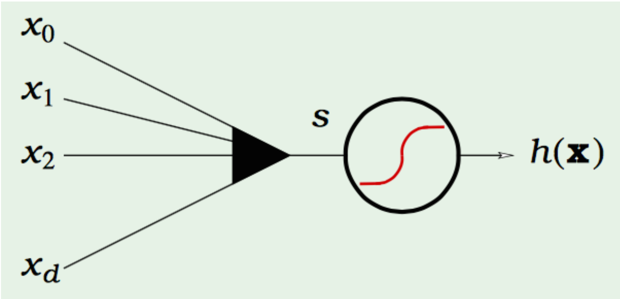
θ (s): probability of a heart attack

S= $\sum_{i=0}^{n} w_i x_i$ **"risk score"**

θ(s) = $\frac{1}{1+e^{-s}}$



S



Sigmoid function

Derivative of Sigmoid Function

$$\frac{ds(x)}{dx} = \frac{1}{1+e^{-x}}$$

$$= \left(\frac{1}{1+e^{-x}}\right)^2 \frac{d}{dx}(1+e^{-x})$$

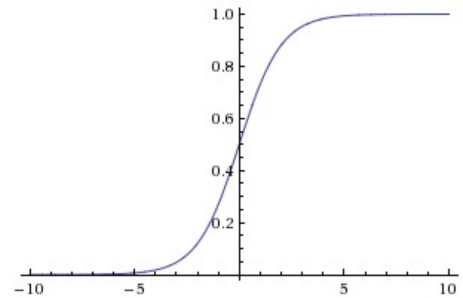
$$= \left(\frac{1}{1+e^{-x}}\right)^2 e^{-x}(-1)$$

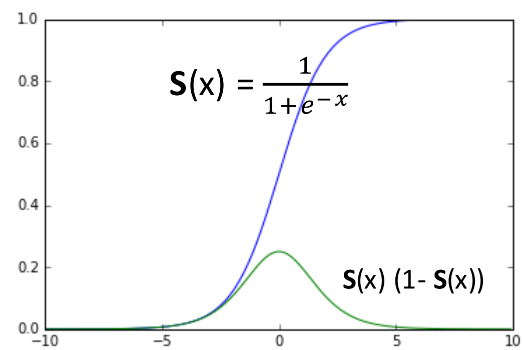
$$= \left(\frac{1}{1+e^{-x}}\right)\left(\frac{1}{1+e^{-x}}\right)(-e^{-x})$$

$$= \left(\frac{1}{1+e^{-x}}\right)\left(\frac{-e^{-x}}{1+e^{-x}}\right)$$

$$= s(x)(1-s(x))$$

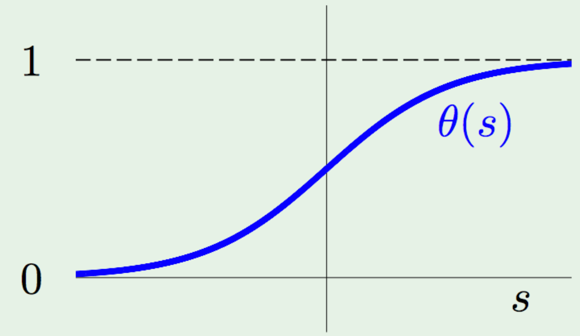
$$\mathbf{S}(x) = \frac{1}{1+e^{-x}}$$





Formula for likelihood

$$P(y \mid \mathbf{x}) = \begin{cases} h(\mathbf{x}) & \text{for } y = +1; \\ 1 - h(\mathbf{x}) & \text{for } y = -1. \end{cases}$$



Substitute $h(\mathbf{x}) = \theta(\mathbf{w}^{\intercal}\mathbf{x})$, noting $\theta(-s) = 1 - \theta(s)$

$$P(y \mid \mathbf{x}) = \theta(y\ \mathbf{w}^{\intercal}\mathbf{x})$$

Likelihood of $\mathcal{D} = (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)$ is

$$\prod_{n=1}^{N} P(y_n \mid \mathbf{x}_n) = \prod_{n=1}^{N} \theta(y_n \mathbf{w}^{\intercal}\mathbf{x}_n)$$

Minimize $$-\frac{1}{N}\ln\left(\prod_{n=1}^{N}\theta(y_n\ \mathbf{w}^{\mathsf{T}}\ \mathbf{x}_n)\right)$$

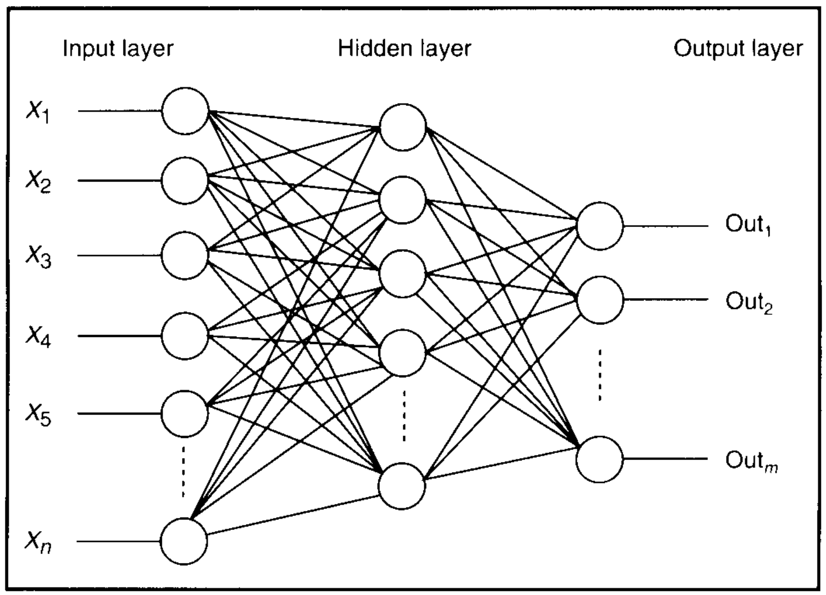
$$=\frac{1}{N}\sum_{n=1}^{N}\ln\left(\frac{1}{\theta(y_n\ \mathbf{w}^{\mathsf{T}}\ \mathbf{x}_n)}\right)$$

$$\left[\theta(s)=\frac{1}{1+e^{-s}}\right]$$

$$E_{\text{in}}(\mathbf{w})=\frac{1}{N}\sum_{n=1}^{N}\underbrace{\ln\left(1+e^{-y_n\mathbf{w}^{\mathsf{T}}\mathbf{x}_n}\right)}_{\mathrm{e}\left(h(\mathbf{x}_n),y_n\right)}$$

"cross-entropy" error

Example: Multi output units



0

1

…

9

$$h(x) = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

$$h(x) = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

$$h(x) = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$