SVM:

loss $f(x^{(i)})$ → nm convex

↓

loss $f(x^{(i)})$ → convex

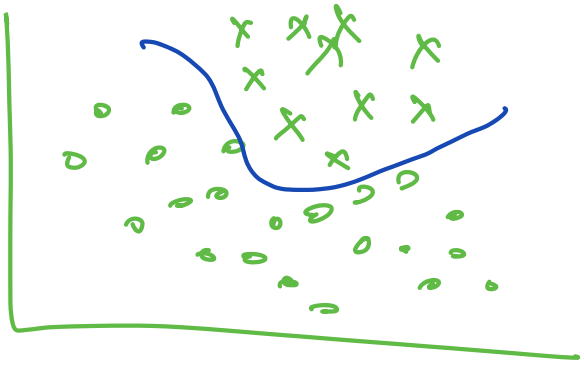
↓

Errors

↓

Remove Linear Constraints } Pegasus
Lagrangian Equality → CS 229 Andrew Ng

Nm Linearly Separable Data:



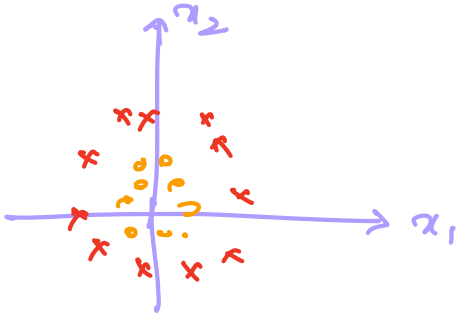
Loss $f(x^{(i)})$:

$$\min \frac{1}{2} w \cdot w^T + C \sum_{i=1}^{m} \varepsilon^{(i)}$$

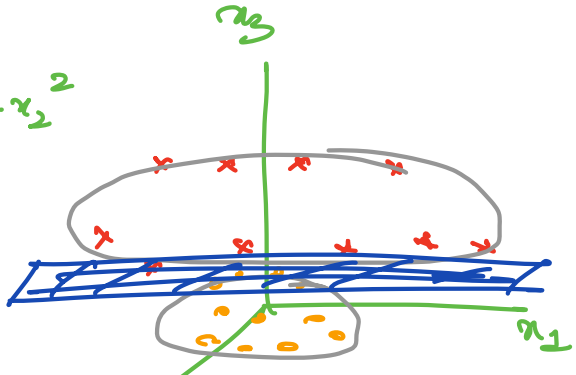
$$y^{(i)} (w^T x^{(i)} + b) \geq 1 - \varepsilon^{(i)}$$

nm Linearly Separable data:

$$x_i \longrightarrow \phi(x_i)$$



$x_3 = x_1^2 + x_2^2$



$$\max \begin{cases} \frac{1}{2} w^T w + C \sum_{i=1}^{m} \xi^{(i)} \\ y_i (w^T \phi(x^i) + b) \geq 1 - \xi^{(i)} \end{cases}$$

Langragian:

$$\max \left(\sum \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \; \phi(x_i^T)\, \phi(x_j) \right)$$

Project higher dim. } costly
Dot Product }

$$K(x_i, x_j) = \phi(x_i)^T \phi(x_j)$$

Eg:

$x = (x_1, x_2, x_3)$ → 3D $\qquad y = (y_1, y_2, y_3)$

$$\phi(x) = (x_1x_1, x_1x_2, x_1x_3, x_2x_1, x_2x_2, x_2x_3, x_3x_1, x_3x_2, x_3x_3)$$

→ 9D

$$K(x,y) = (x \cdot y)^2 \quad \rightarrow \text{Already provided}$$

$x = (1,2,3) \qquad y = (4,5,6)$

$\phi(x) = (1, 2, 3, 2, 4, 6, 3, 6, 9)$

$\phi(y) = (16, 20, 24, 20, 25, 30, 24, 30, 36)$

$\phi(x) \cdot \phi(y) = 16 + 40 + 72 + 40 + 100 + 180 + 72 + 180 + 324$
$= 1024$

$\phi(x)\, \phi(y)$ takes a lot of time

Kernel trick

$x = (1, 2, 3)$

$y = (4, 5, 6)$

$x \cdot y = 4 + 10 + 18$

$(x - y)^2 = (4 + 10 + 18)^2$

$= 1024$

Kernels:

① RBF Kernel

$$K(x_i, x_j) = e^{-\gamma |x_i - x_j|^2}$$

γ = Amplitude

② Polynomial Kernel

$$K(x_i, x_j) = \left[\gamma x_i^T \cdot x_j + r \right]^c$$

c = Degree

③ Sigmoid Kernel

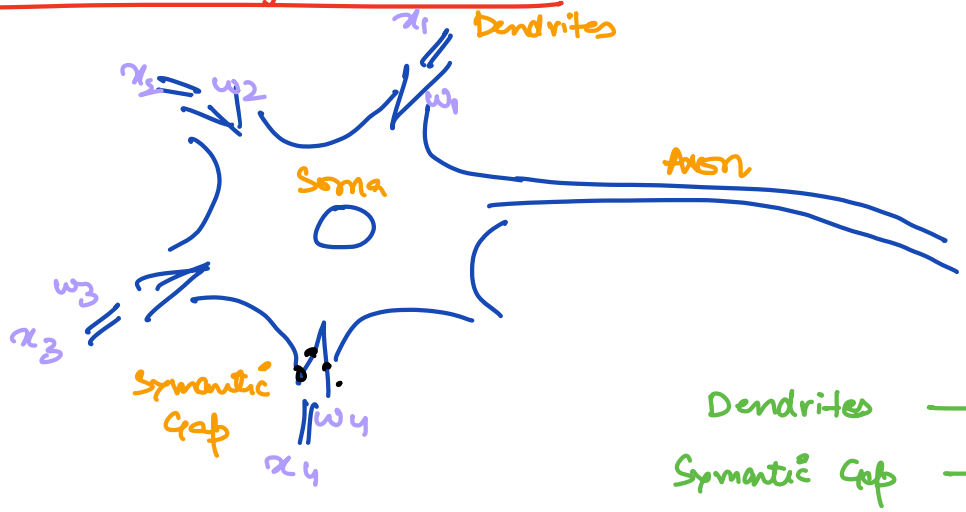
$$K(x_i, x_j) = \frac{1 - e^{-2(\gamma x_i x_j + r)}}{1 + e^{-2(\gamma x_i x_j + r)}}$$

SL. [NN

UL [KMeans GMM

RL [

Artificial & Biological Neurons



Dendrites ⟶ Inputs

Synaptic Gap ⟶ weight

Soma ⟶ Activation

Axon ⟶ output

$Z = w_1x_1 + w_2x_2 + \cdots w_4x_4$

$Z = \sum w_i x_i$

$a = g(z)$

$g(z) \rightarrow$ Sigmoid $\quad \frac{1}{1+e^{-z}}$

↳ ReLU (Rectified Linear Unit)

$$g(z) = \begin{cases} z & z \geq 0 \\ 0 & z < 0 \end{cases}$$

Neuron fires

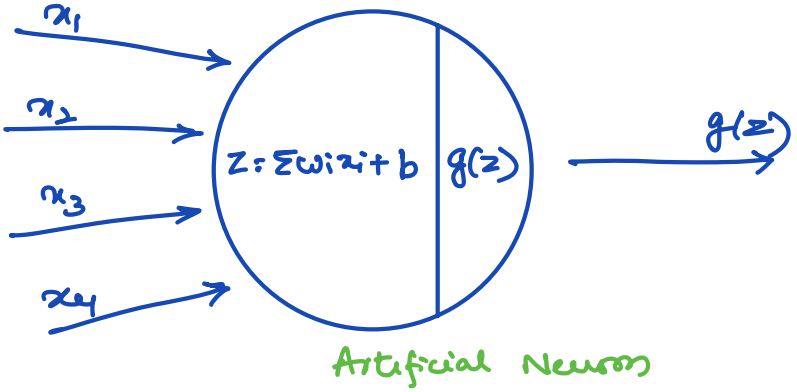
Neuron doesn't fire

Neuron should fire when value is $-10 \rightarrow$ Bias $+10$

$Z = w_i x_i + b$

$-10 \quad 10$

0

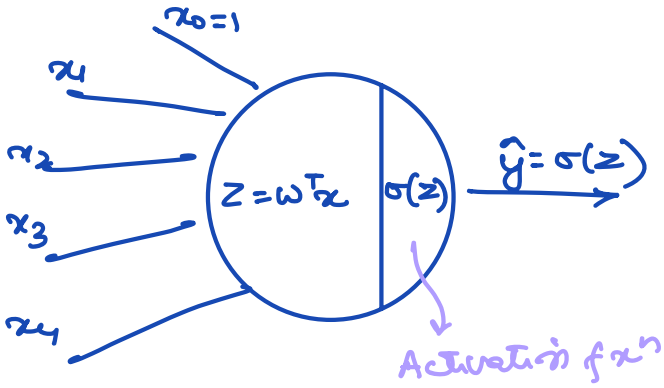


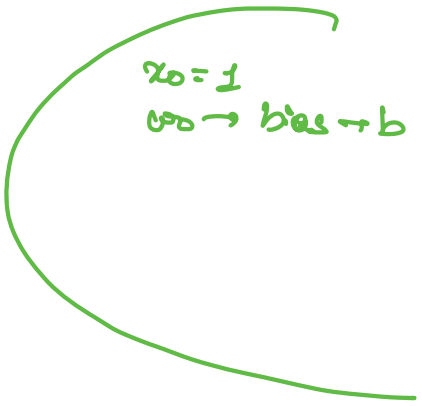
Artificial Neuron

Perceptron

↳ Single layer neural network

loss fx^n
weight update rule



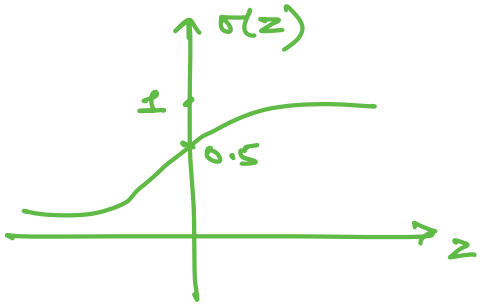


$$z = w^T x = [w_0 \; w_1 \; w_2 \; \ldots \; w_n] \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

Sigmoid fx^n $\quad \sigma(z) = \dfrac{1}{1+e^{-z}}$

↓

o/p is any no. b/w 0 & 1.



Dog Cat ?

	D	D	D	C	C
$\hat{y}$	0.7	0.8	0.8	0.3	0.4

↳ Probability that it belongs to class 1.

y	1	1	1	0	0

Loss fx^n: MSE $= \frac{1}{m}\sum_{i=1}^{m}(\hat{y}^{(i)} - y^{(i)})^2$

→ non convex fx^n

$\hat{y} = \sigma(z) = \sigma(w^T x)$

log loss / Binary Cross Entropy

$$J(w) = -\sum_{i=1}^{m}\left(y^{(i)} \log \hat{y}^{(i)} + (1-y^{(i)}) \log (1-\hat{y}^{(i)})\right)$$

→ convex fx^n

$\hat{y}^{(i)} = \sigma(z) = \sigma(w^T x^{(i)})$

$J(w)$ is a fx^n of $\hat{y}^{(i)}$

$\hat{y}^{(i)}$ is a fx^n of $w^T x^{(i)}$

GRADIENT DESCENT:

$w = w - \eta \frac{\partial J}{\partial w}$

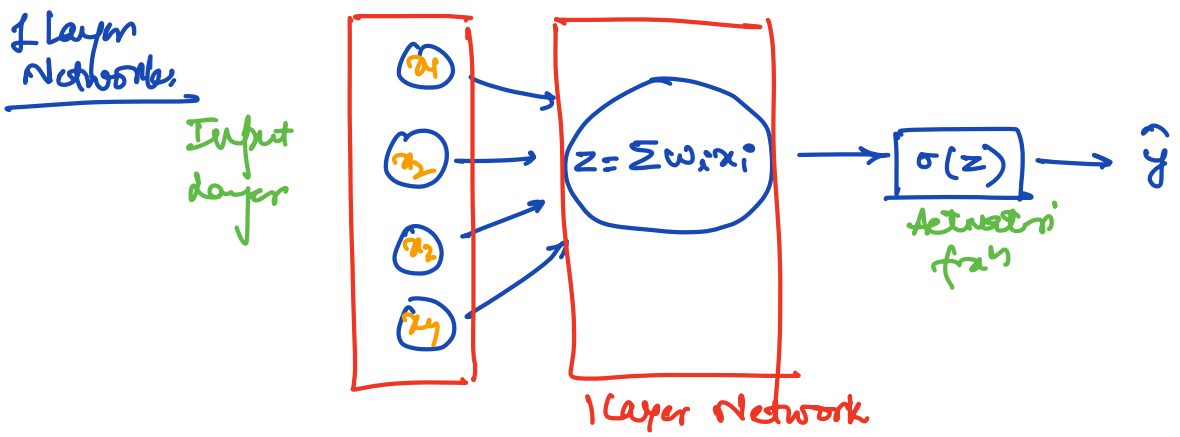
→ Goal is to learn w such that loss is minimized

↓ Exactly same that we did in Log. Regression

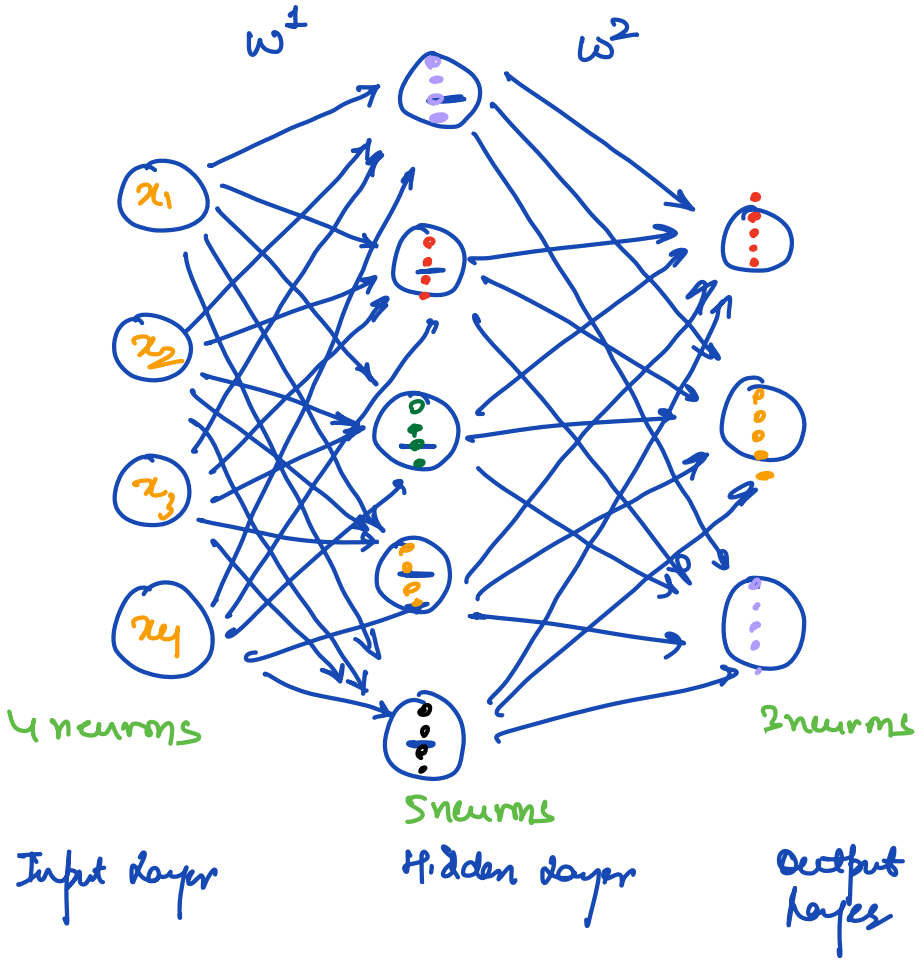
Perceptron + Sigmoid = Logistic Regression

Neural Architecture

1 Layer Networks:



2 layer Network.



$$W^{[1]} = \begin{bmatrix} & & & & \\ & & & & \\ & & & & \\ & & & & \end{bmatrix}$$

4 x 5 → neurons in l-th layers

no. of neurons in $(l-1)$ layer

$$W^{[1]} = \begin{bmatrix} | & | & | & | & | \\ w_1^{[1]} & w_2^{[1]} & w_3^{[1]} & w_4^{[1]} & w_5^{[1]} \\ | & | & | & | & | \end{bmatrix}$$

Weights associated with 5th neuron in 1st layer

$W_5^{[1]}$ → Weights for 5th neuron in 1st layer

$W^{[1]} =$  5×3

o/p layer

In o/p layer which activation fn to apply?

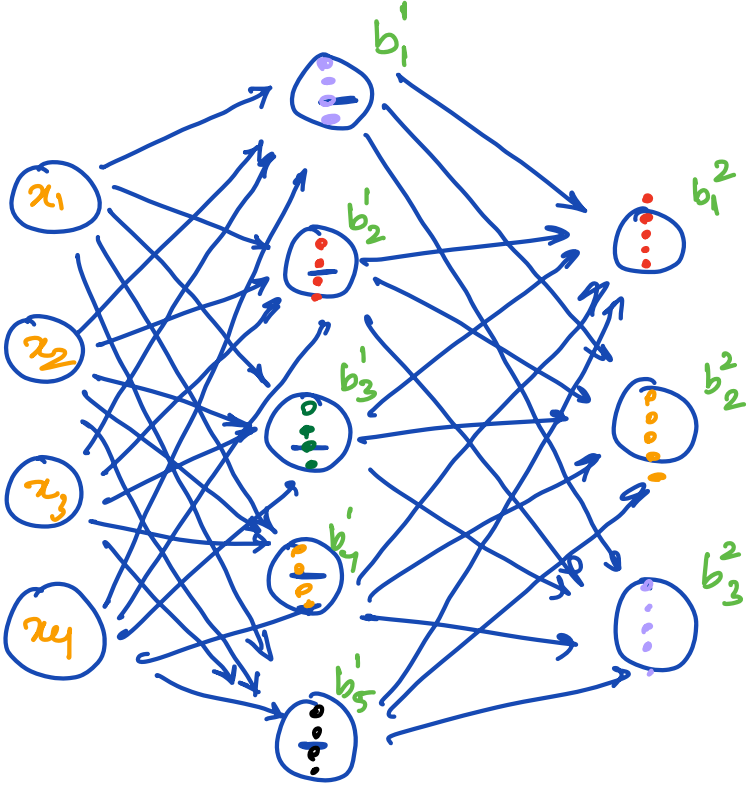
Softmax

[10, 10, 20]

↓ softmax : Prob

$\left[\frac{10}{40}, \frac{10}{40}, \frac{20}{40}\right]$

[0.25, 0.25, 0.5]



$b^{[1]}$ = layer

$b_{[2]}$ = neuron no

$$b^{[1]} = \begin{bmatrix} b_1^1 \\ b_2^1 \\ b_3^1 \\ b_4^1 \\ b_5^1 \end{bmatrix} \qquad b^{[2]} = \begin{bmatrix} b_1^2 \\ b_2^2 \\ b_3^2 \end{bmatrix}$$

No. of parameters for model and learn

$$= w^{[1]} + b^{[1]} + w^{[2]} + b^{[2]}$$

$$= (4 \times 5) + 5 + (5 \times 3) + 3$$

$$= 20 + 5 + 15 + 3$$

$$= 43$$

LLM

- GPT
- Llama
- Google Palm

} Billions parameters