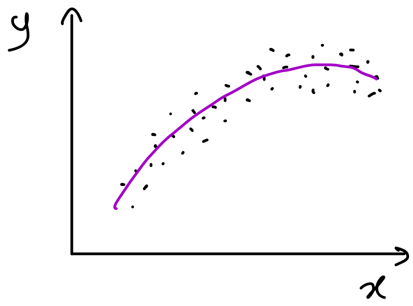
Problem : Understanding the input output relationship
f(x) is unknown

Given : Data from the past



$\hat{f}(x)$ fits the data

estimate of the unknown function
f(x)

Simple solution: linear regression

to regress = moving backward
reasoning backward
→ learning from the past

Fitting the data : $y = \underbrace{\alpha + \beta x}_{\text{linear combination}} + \underset{\text{noise}}{\varepsilon}$

Simplified ⟶ $\varepsilon = 0$

$y = \alpha + \beta x$

$E(y|x) = \alpha + \beta X = \overbrace{W_0 + W_1 X}^{\text{weights}}$

model ← → model free

Intelligence is to find w_0, w_1 !

$$E(w_0,w_1|x) = \frac{1}{2}\sum_{i=1}^{N}\Big[\underset{\text{Desired output}}{\underset{\downarrow}{y_i}} - \underbrace{(w_1 x_i + w_0)}_{\text{estimate by guessing } w_0, w_1}\Big]^2$$

Build partial derivative wrt w_0, w_1 and simplify.

$$\frac{\partial E}{\partial w_0} = \frac{1}{2}\sum_{i=1}^{N} 2[y_i - (w_1 x_i + w_0)]\,(0-(0+1)) = 0$$

$$\sum_{i=1}^{N} y_i = \sum_{i=1}^{N} w_1 x_i + w_0 = N w_0 + w_1 \sum_{i=1}^{N} x_i$$

$$\frac{\partial E}{\partial w_1} = \frac{1}{2}\sum_{i=1}^{N} 2[y_i - (w_1 x_i + w_0)]\,(0 - x_i - 0) = 0$$

$$\sum_{i=1}^{N} y_i x_i = w_0 \sum_{i=1}^{N} x_i + w_1 \sum_{i=1}^{N} x_i^2$$

Matrix form : $AW = y$

$$A = \begin{bmatrix} N & \Sigma x_i \\ \Sigma x_i & \Sigma x_i^2 \end{bmatrix} \qquad W = \begin{bmatrix} w_0 \\ w_1 \end{bmatrix}$$

$$y = \begin{bmatrix} \Sigma y_i \\ \Sigma y_i x_i \end{bmatrix} \qquad W = A^{-1} \cdot y$$

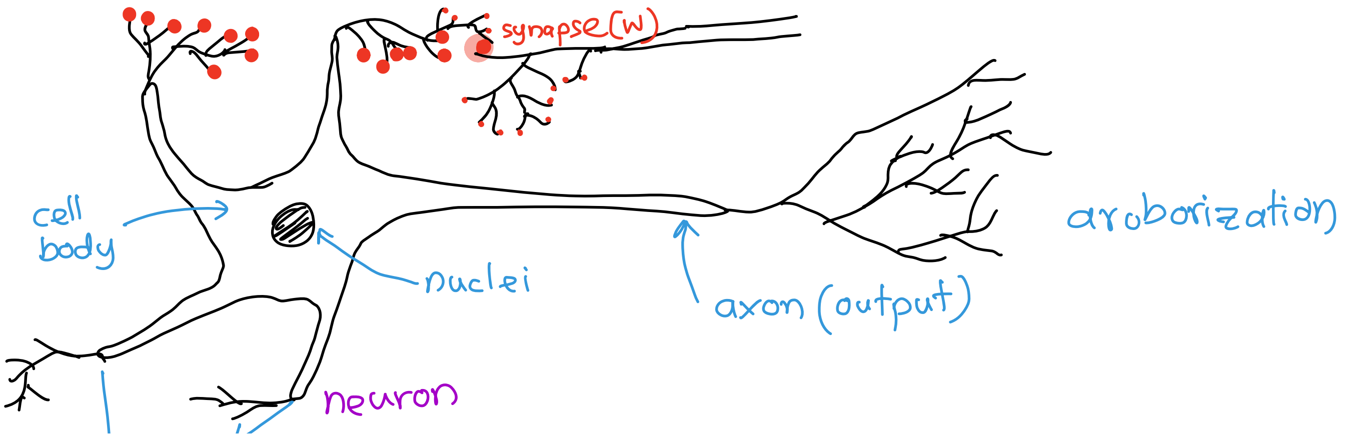
What about $y(x) = w_0 + w_1 x + w_2 x^2$

this is linear; linear combination of weights

forget about x^2

If this is still a linear regression, then how do we do non linear regression?

→ Neural networks



dendrites (inputs)

Intelligence is changing synapse

How many neurons?

zebra fish ≈ 250k neurons

Adult human ≈ $10^{10} - 10^{12}$ neurons

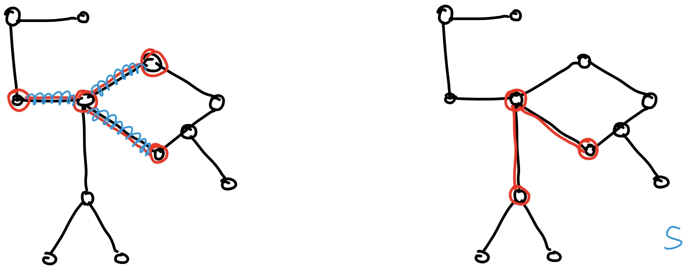
≈ 10^{14} synapses

Synapses that increase the potential are inthe exitatory mode.

Synapses that decreases the potential are in the inhibitory mode.

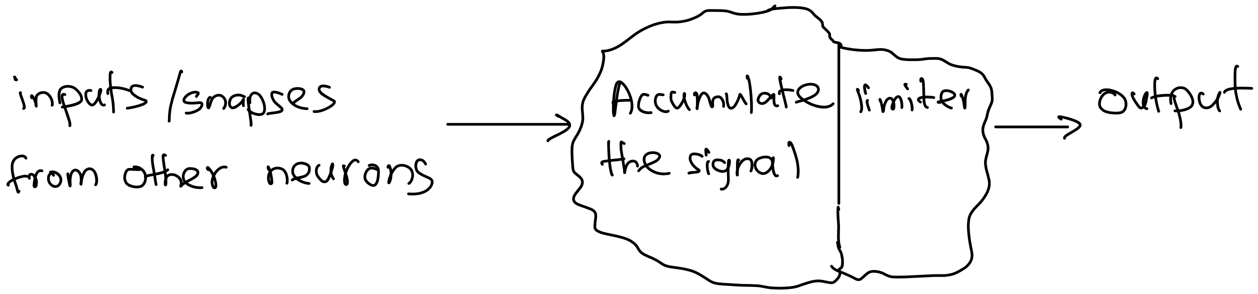
The synaptic networks are plastic.

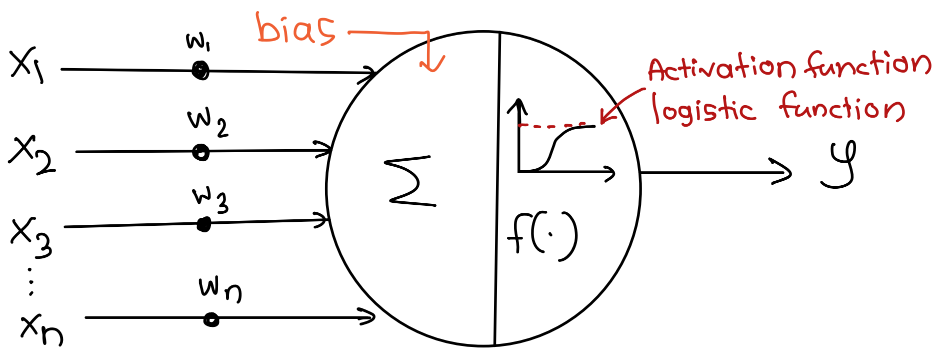
The plasticity is the most obvious manifestation of intelligence.



same network can exhibit different sub-graphs of connections

Abstraction of Neurons



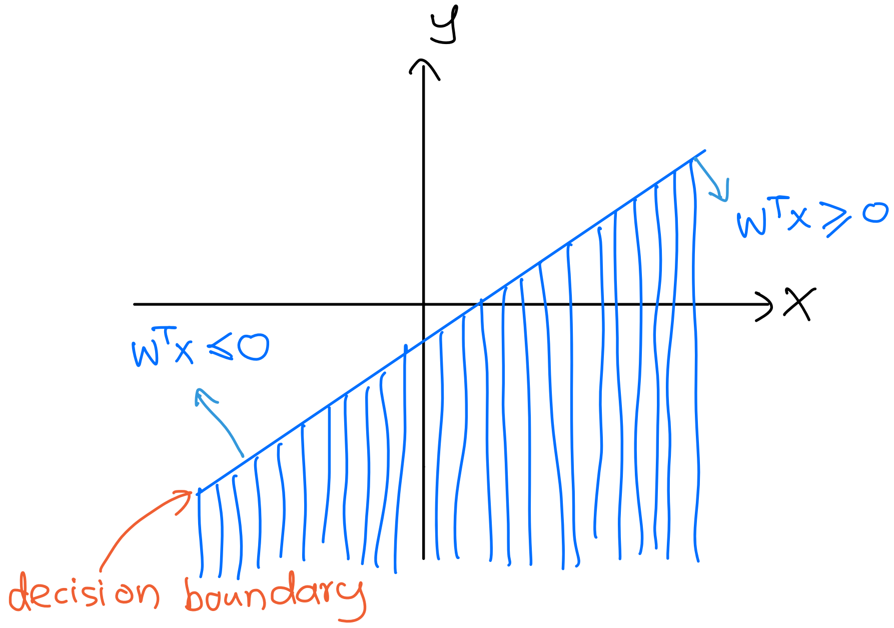


Perceptron

In perceptron, the logistic function is a "hard limiter" ie (thershold)

$$S = \sum_{i=1}^{n} w_i x_i + bias$$

This is a line (i.e hyperplane). Hence the perceptron can seperate 2 classes.



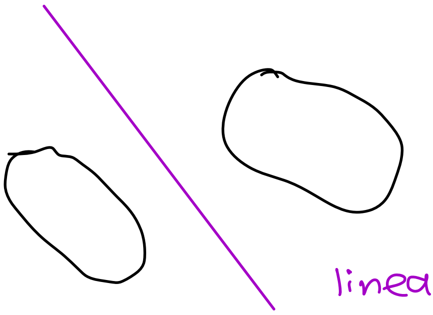
$x_0 = +1$, $w_0 = b$

$$S = \sum_{i=0}^{m} x_i w_i$$

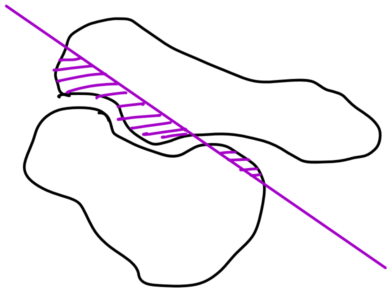
Now we have to iterate over data

$$S(n) = \sum_{i=0}^{m \,\leftarrow\, \text{number of instances}} W_i(n)\cdot x_i(n) \quad \text{number of iterations}$$

$= \quad w^T(n) \cdot x(n)$



linearly separable



non linearly separable

underfitting

Weight adjustment

x(n) correctly classified by w(n)

boundary decision

$w(n+1) = w(n)$ if $w^T \cdot x(n) \underset{<}{\geq} 0$ $\quad x(n) \in c_{i,j}$

no change since classification is correct

If misclassified,

$$w(n+1) = w(n) - \eta(n) \cdot \underline{x(n)}$$

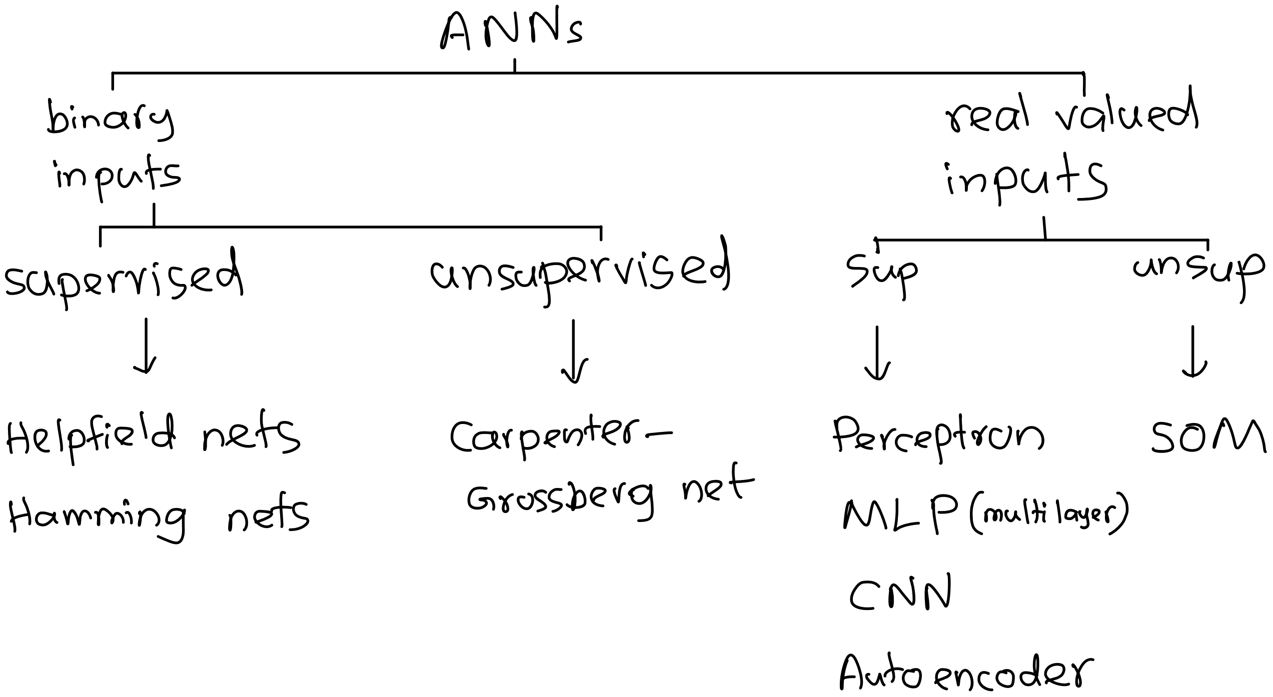
contribution of input

If $\eta(n) = \eta > 0$ fixed increment adaptation

How η is not very important as long as positive

(Its just scales the contribution)

Perceptron can be proven to converge $\eta = 1$


ANNs
binary inputs
real valued inputs
supervised
unsupervised
Sup
unsup
Helpfield nets
Hamming nets
Carpenter-Grossberg net
Perceptron
MLP (multi layer)
CNN
Auto encoder
SOM