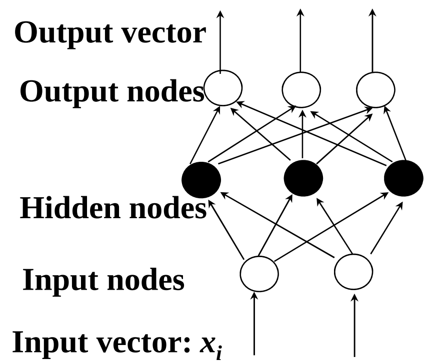
# Artificial neural networks

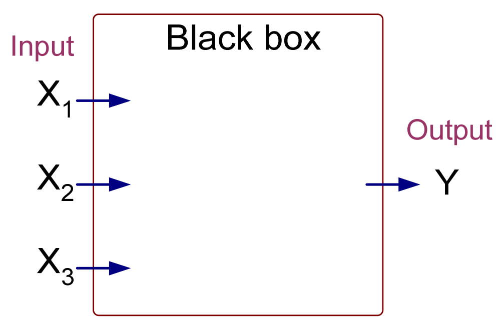
- *Artificial* neural networks
  - Highly abstracted from some properties of real neurons
  - Unrestricted by properties of real neurons
- A general nonlinear modeling method
  - Arbitrarily accurate modeling of arbitrarily complex functions
  - But you get what you pay for






# Artificial Neural Networks (ANN)

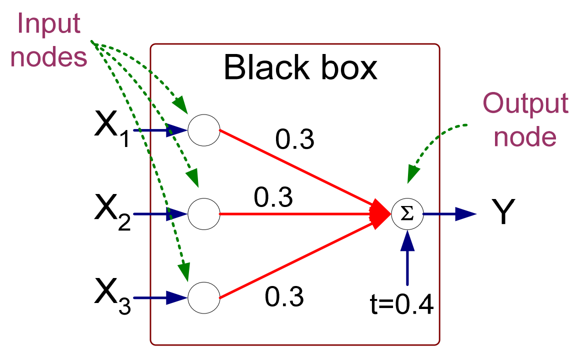
| $X_1$ | $X_2$ | $X_3$ | Y |
|---|---|---|---|
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 |
| 0 | 0 | 0 | 0 |




Output Y is 1 if at least two of the three inputs are equal to 1.

# Artificial Neural Networks (ANN)

| $X_1$ | $X_2$ | $X_3$ | Y |
|---|---|---|---|
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 |
| 0 | 0 | 0 | 0 |



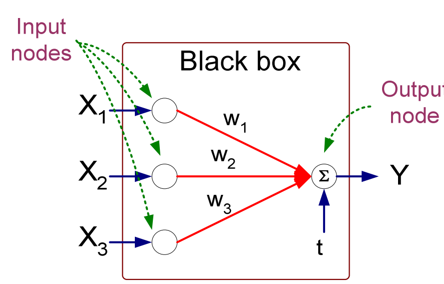

$$Y = I(0.3X_1 + 0.3X_2 + 0.3X_3 - 0.4 > 0)$$

$$\text{where} \quad I(z) = \begin{cases} 1 & \text{if } z \text{ is true} \\ 0 & \text{otherwise} \end{cases}$$



# Artificial Neural Networks (ANN)

- Model is an assembly of inter-connected nodes and weighted links
- Output node sums up each of its input value according to the weights of its links
- Compare output node against some threshold t




**Perceptron Model**

$$Y = I(\sum_i w_i X_i - t) \quad \text{or}$$

$$Y = sign(\sum_i w_i X_i - t)$$

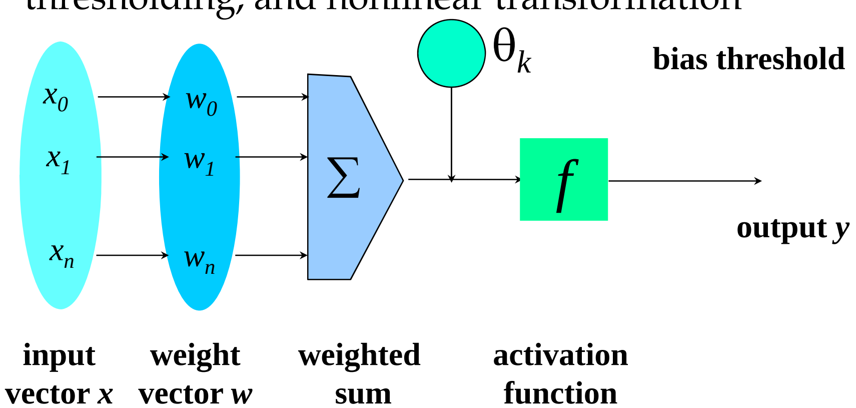
## Historical problems

- Early work focused on linear threshold recognizers: single-layer neural nets called perceptrons
- Lots of terrible work and false claims
  - The tank recognizer
- Minsky & Papert's discouraging news
  - Many important patterns not linearly recognizable
  - Astronomical training times
  - Astronomical recognition parameters
- Multilayer nonlinear networks improve the outlook



## Artificial neurons

- Nonlinear functions mapping n-dimensional input vectors $x$ into outputs $y$ by linear weighting, thresholding, and nonlinear transformation

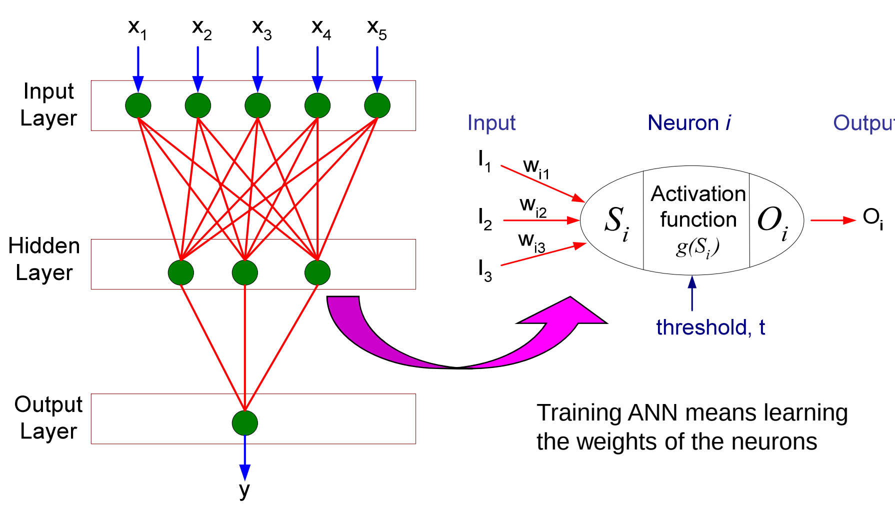


# Activation functions

- Linear thresholds
- Sigmoid function
- Radial basis functions
- Other function bases



# General Structure of ANN




Training ANN means learning the weights of the neurons

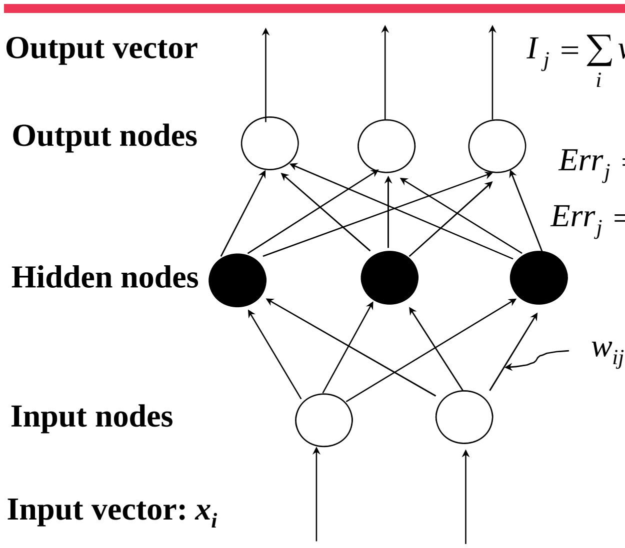
## Algorithm for learning ANN

- Initialize the weights ($w_0$, $w_1$, …, $w_k$)

- Adjust the weights in such a way that the output of ANN is consistent with class labels of training examples
  - Objective function: $E = \sum_i \left[ Y_i - f(w_i, X_i) \right]^2$
  - Find the weights $w_i$'s that minimize the above objective function
    - e.g., backpropagation algorithm



## Backpropagation training




$$I_j = \sum_i w_{ij} O_i + \theta_j$$

$$O_j = \frac{1}{1 + e^{-I_j}}$$

$$Err_j = O_j (1 - O_j)(T_j - O_j)$$

$$Err_j = O_j (1 - O_j) \sum_k Err_k w_{jk}$$

$$\theta_j = \theta_j + (l) Err_j$$

$$w_{ij} = w_{ij} + (l) Err_j O_i$$

## Advantages

- High prediction accuracy possible
  - With choice of suitable network topology
  - With choice of suitable learning method
- Discrete, continuous, or vector outputs
- Fast classification once trained



## Disadvantages

- Long training time for each specific network
- Empirical search for suitable networks
- Overfitting still a possibility
- No explicit knowledge
- Incorporating domain knowledge is awkward

## Network construction

- Network topology selection
  - Number of hidden layers
  - Number of nodes in hidden layers
  - Interconnections of nodes
- Network training
  - Find weights that yield the best classification given topology
- Network topology editing
  - Find topology that yields best classification
  - Find topology that yields clearest interpretation



## Network topology selection

- Knowledge-free networks
  - Start with one fully-connected hidden layer
  - Add hidden layers if performance improves
- Knowledge-based networks
  - Identify nodes representing significant propositions
  - Add connections representing inferential relations
  - Can resemble Bayesian network, but different interpretation and training procedure

## Network pruning

- A fully connected network is big
  - *H(M+N)* weights from *N* input nodes, *H* hidden nodes and *M* output nodes
- Reduce size by removing insignificant links
  - Try removing links with tiny weights compared to siblings
  - Check effect on classification accuracy



## A grain of salt

- High accuracy with sufficient training is no big deal
- Many sets of functions provide universal representations
  - Polynomials, Fourier series, wavelets, etc.
  - Rubel's universal differential equation
  - Rubel's universal entire function
- Look for clear expression and rapid approximation methods
- Vapnik-Chervonenkis (VC) dimension

## Network rule extraction

- A fully connected network is unintelligible
  - What does it mean?
  - Can you explain its content in rules?
- Transform classification to discrete propositions
  - Identify discrete activation values as cluster averages
  - Check that classification accuracy preserved
- Identify implicit classification rules
  - Enumerate outputs produced by discrete input values
  - Recast as classification rules or decision tree



## The upshot

- Neural nets popular, and sometimes work
  - Fancy mathematics, but in the end, largely trial-and-error with few guarantees
  - Latest incarnation of black-box training
  - Still slow or slower
- Trained networks call for further analysis
  - Understand structure of trained network
  - Prune and solidify
  - Extract intelligible knowledge
  - Assess the remainder