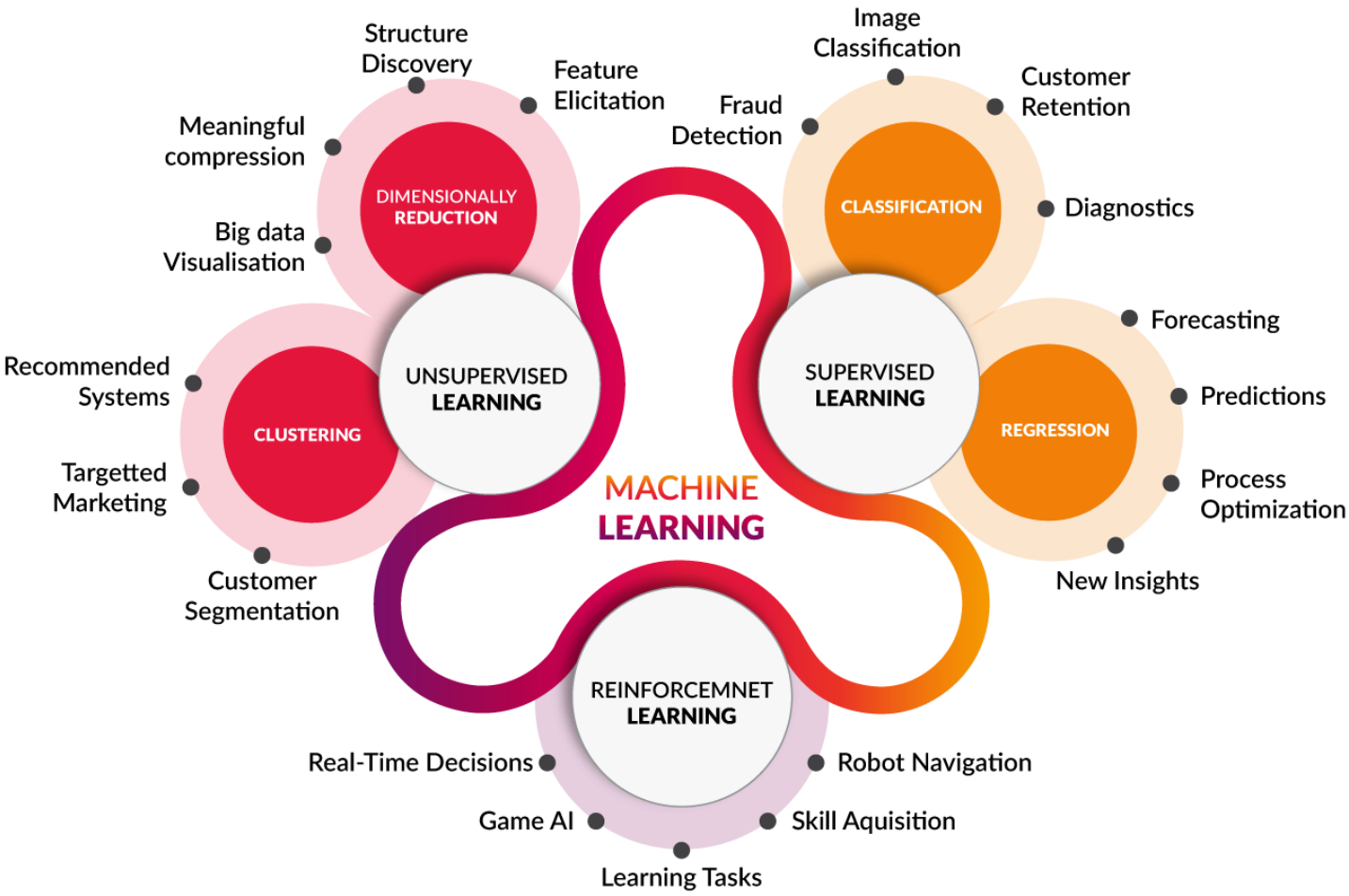
Tutorial on Deep Learning

CS - 590.21 Analysis and Modeling of Brain Networks

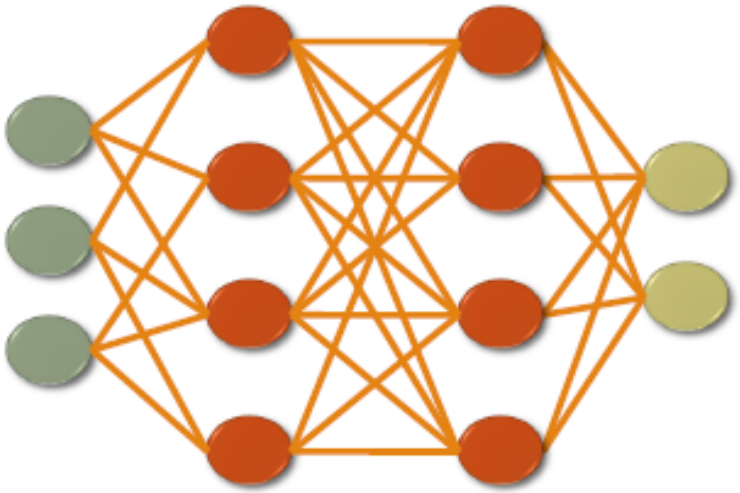
Evripidis Tzamousis

Types of Machine Learning



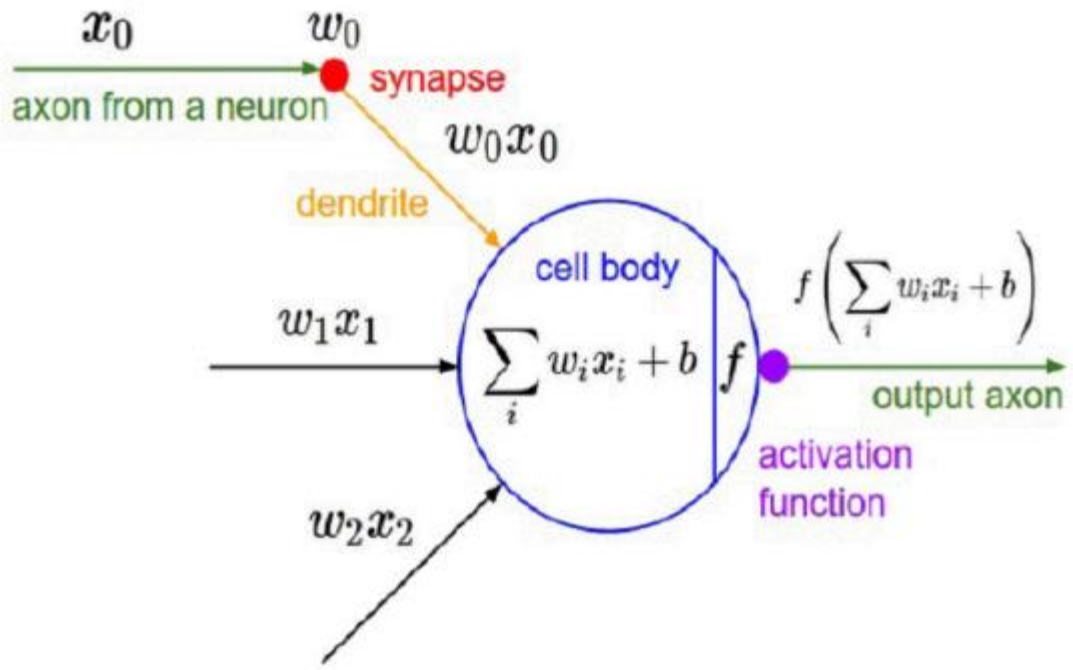
Artificial Neural Networks (ANN)

- Architecture (input/hidden/output layers)



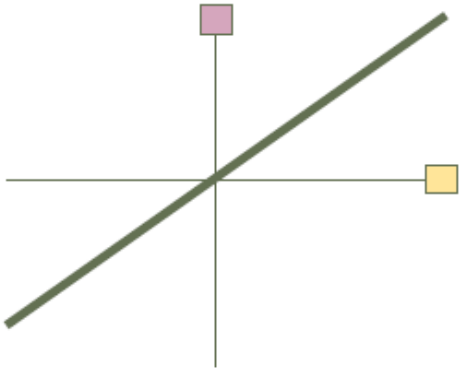
Artificial Neural Networks (ANN)

- Architecture (input/hidden/output layers)
- Weights

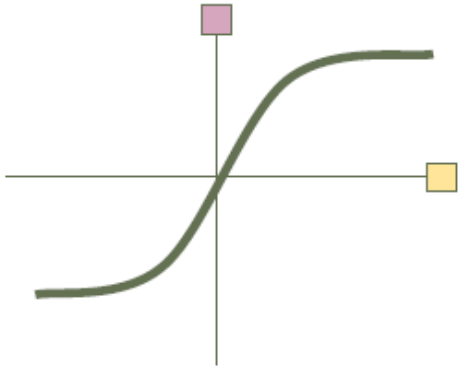


Artificial Neural Networks (ANN)

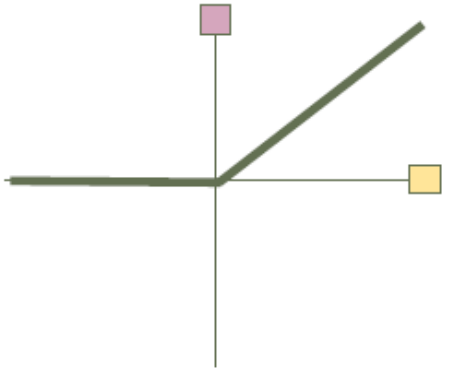
- Architecture (input/hidden/output layers)
- Weights
- Activation functions



LINEAR

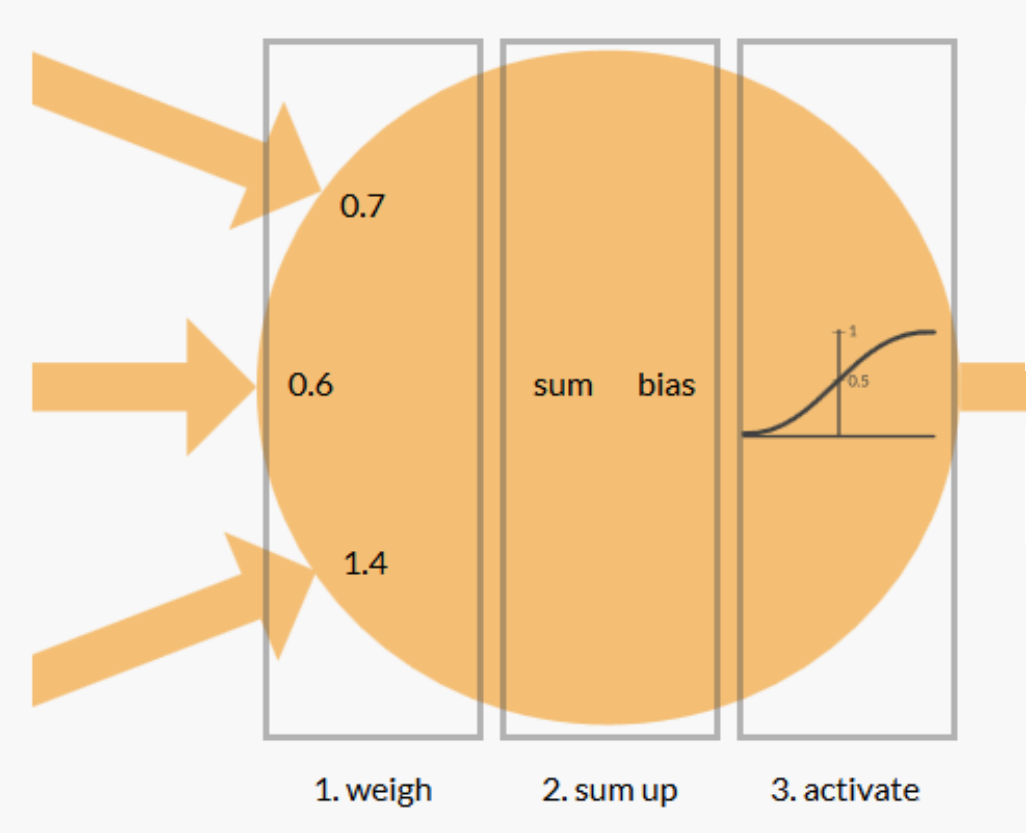


LOGISTIC / SIGMOIDAL / TANH



RECTIFIED LINEAR (ReLU)

Inside an artificial neuron



Activation of the ***j***-th neuron in the ***l***-th layer

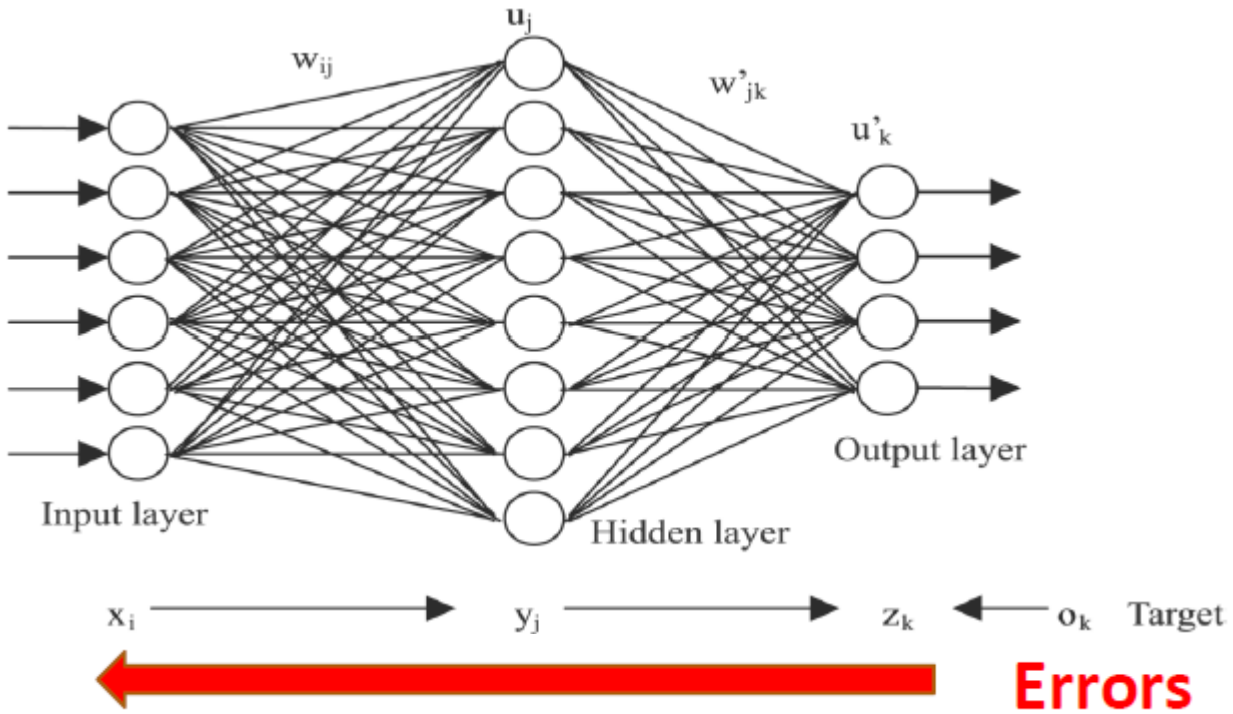
$$a_j^l = \sigma\left(\sum_k w_{jk}^l a_k^{l-1} + b_j^l\right)$$

w_{jk}^l weight from the ***k***-th neuron in the ***(l-1)***-th layer to the ***j***-th neuron in the ***l***-th layer

b_j^l weight from the ***k***-th neuron in the ***(l-1)***-th layer to the ***j***-th neuron in the ***l***-th layer

Training an ANN

1. Get data
2. Forward through the network → compute error
3. **Backpropagate** error
4. Update weights based on gradient



Backpropagation

Define a loss/cost function

$$J(x, y; \theta) = \frac{1}{2} \sum (y - f(x; \theta))^2$$

$$f(x; \theta) = w^T x + b \quad , \quad \theta = \{w, b\}$$

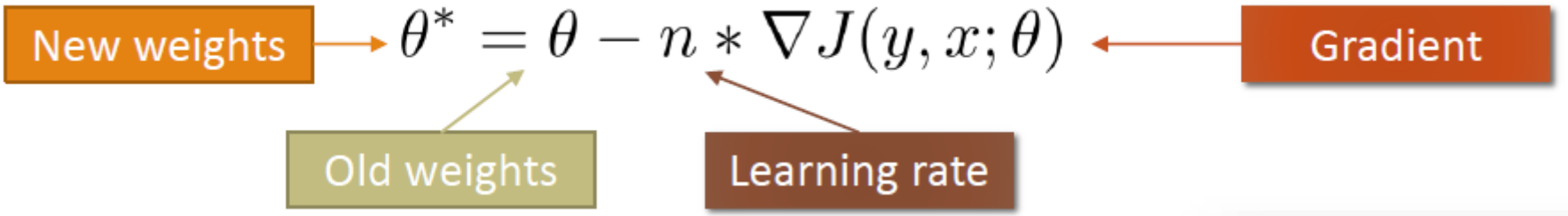
Backpropagation

Define a loss/cost function

$$J(x, y; \theta) = \frac{1}{2} \sum (y - f(x; \theta))^2$$

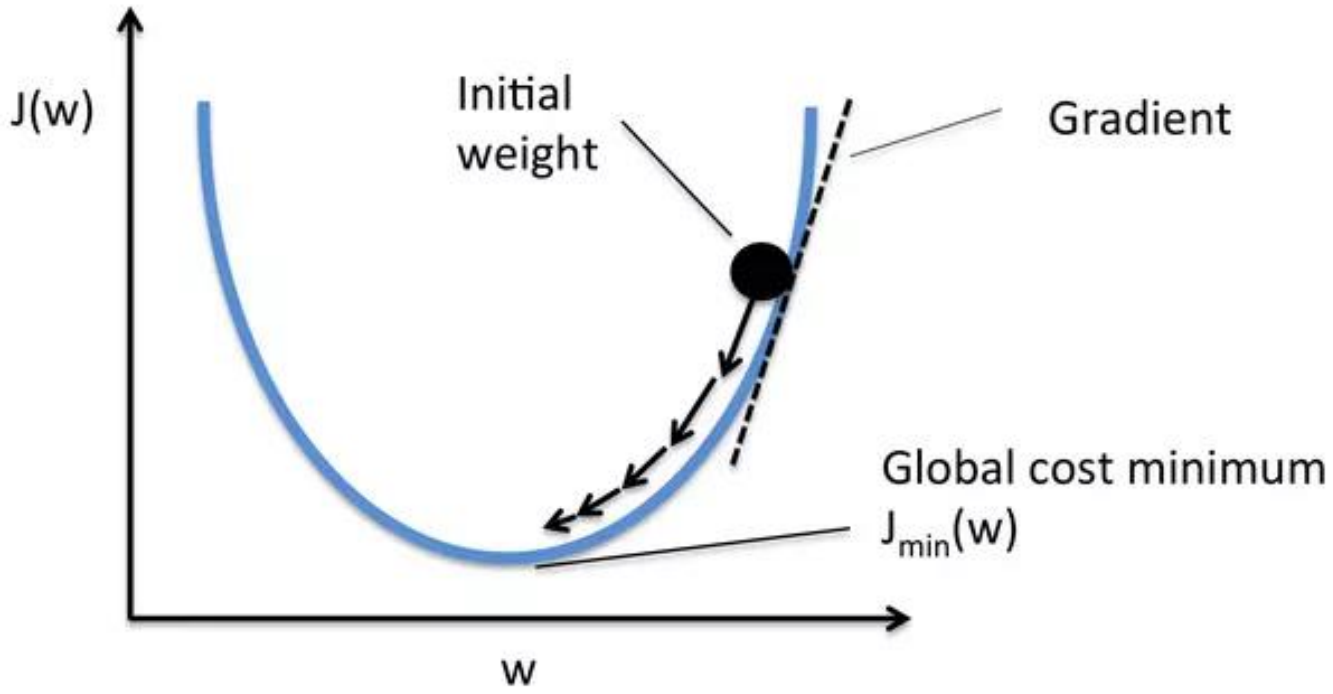
$$f(x; \theta) = w^T x + b \quad , \quad \theta = \{w, b\}$$

Minimize cost function w.r.t. parameters θ



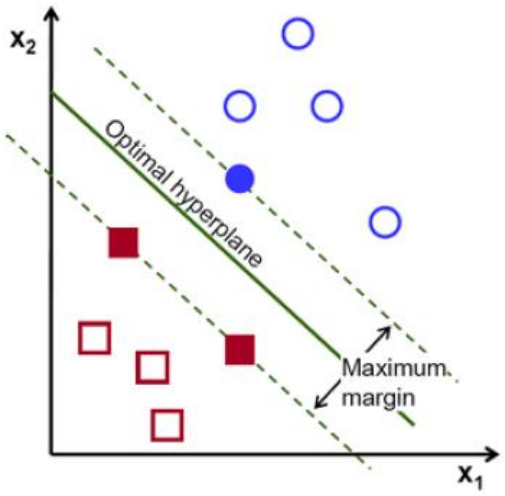
Gradient descent

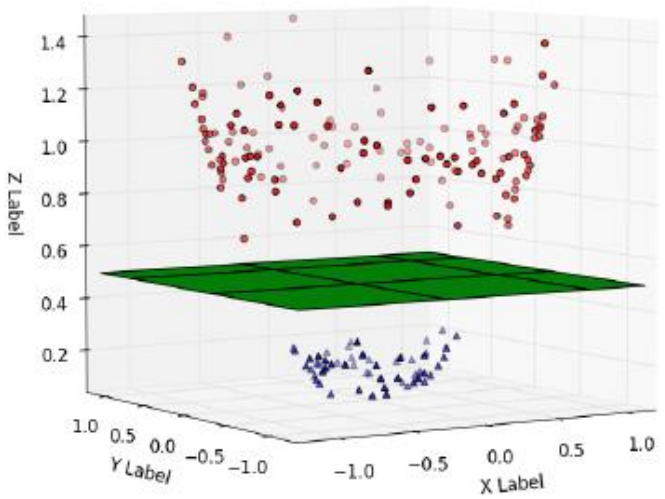
Gradient: $$\nabla J(x) = (\frac{\partial J(x)}{\partial x_1}, \frac{\partial J(x)}{\partial x_2}, ..., \frac{\partial J(x)}{\partial x_n})$$



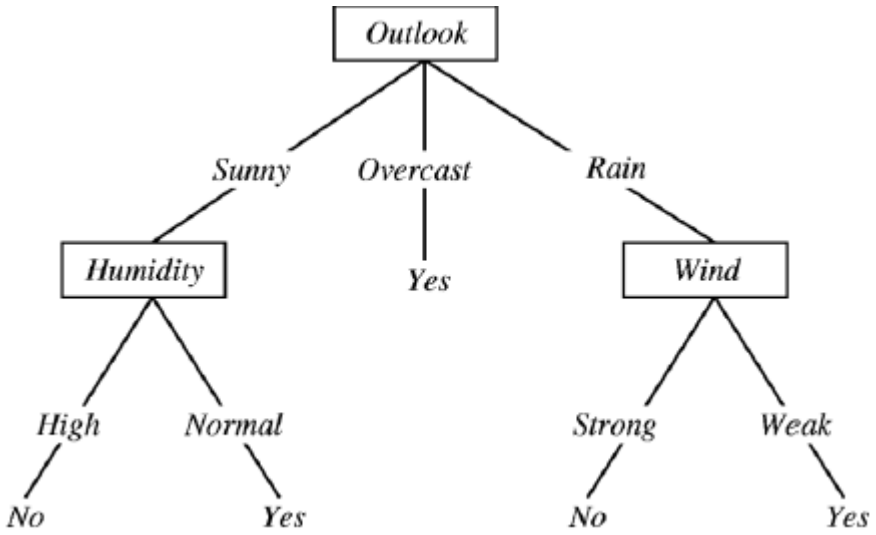
Before Deep Learning

Support Vector Machines

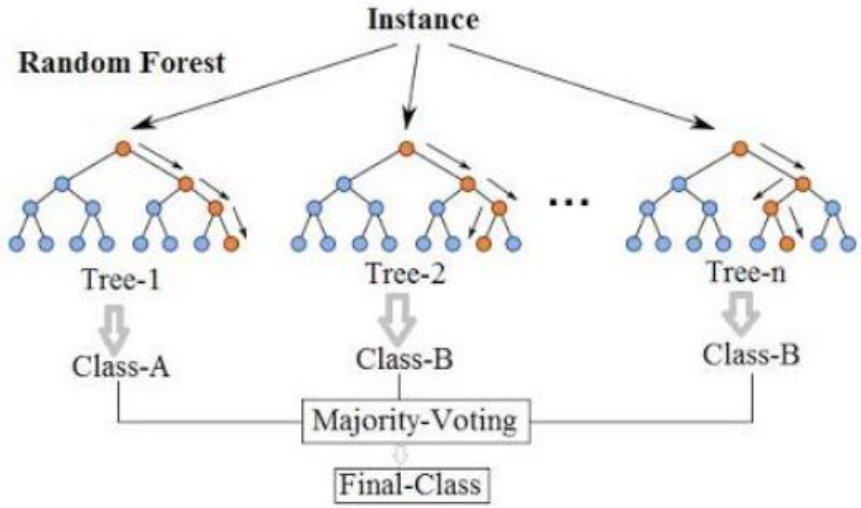




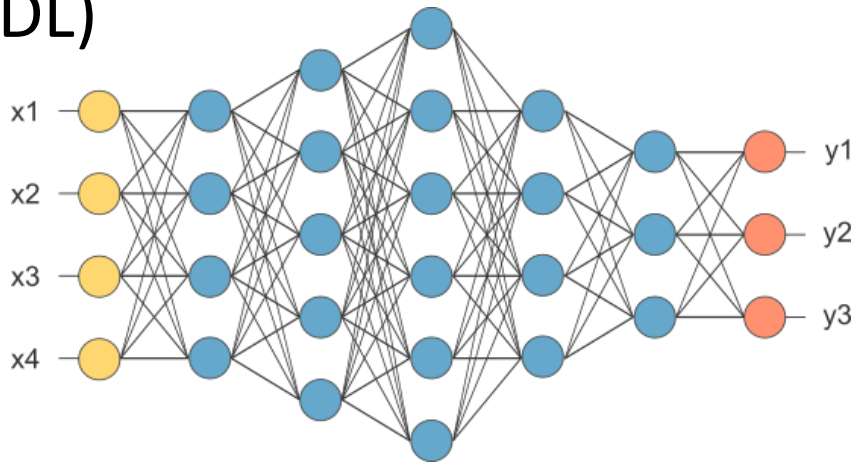
Decision Tree



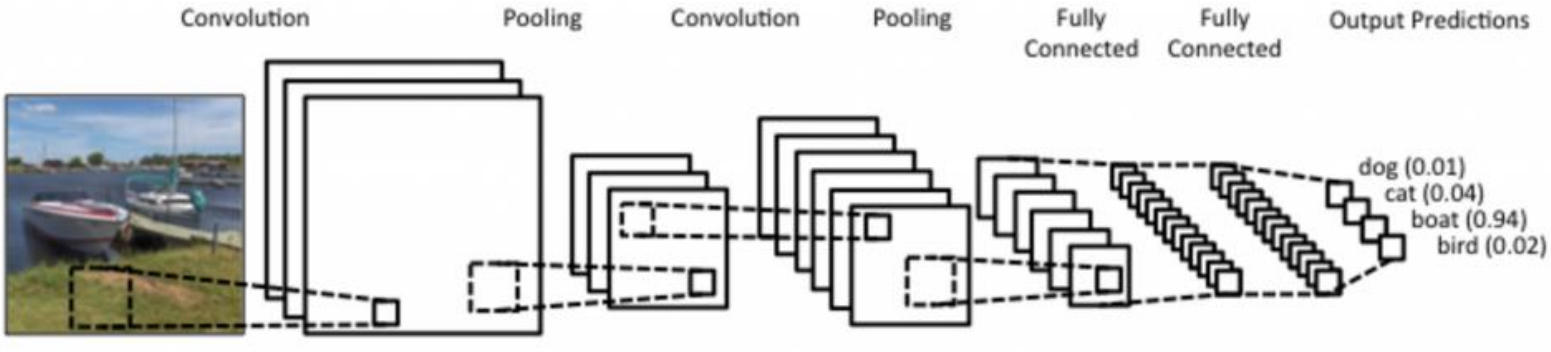
Random Forest Simplified



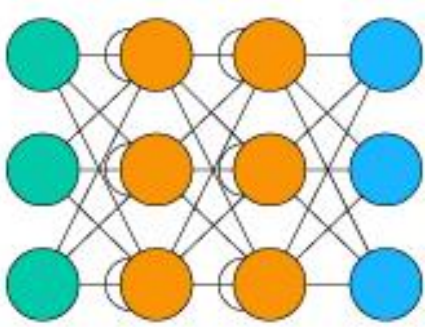
Deep learning (DL)



Convolutional Neural Networks (CNN)



Recurrent Neural Networks (RNN)



Matlab examples