Using Machine Learning Tools

Deep Neural Networks

University of Adelaide

Previously …

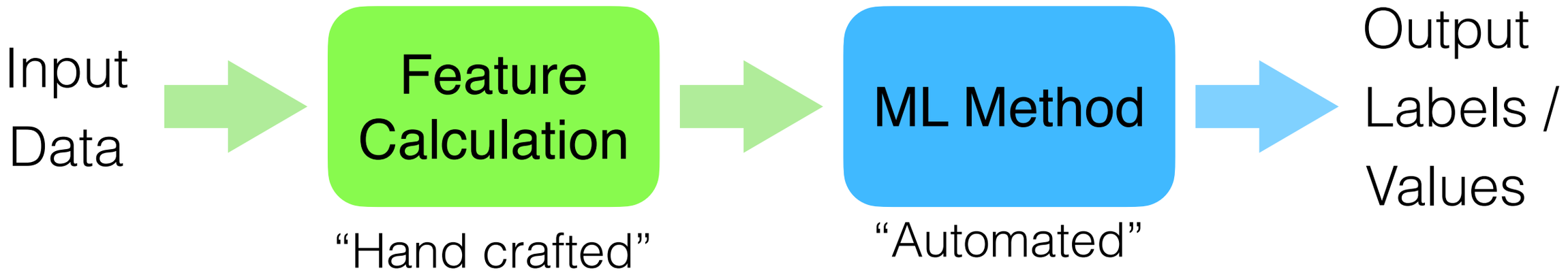
- Last time we discussed clustering, data visualisation and dimensionality reduction methods
- Prior to that we considered regression and classification ML
  - Defined loss (or cost) functions (e.g. MSE)
  - Discussed gradient-based optimisation methods (e.g. SGD)
  - Importance of learning rate and initialisation

Today

- Introduction to Neural Networks
- Network architecture: layers, connectivity and features
- Neurons: nonlinearities and activation functions
- Parameters: connections, weights and biases
- Number of parameters
- Loss functions, epochs, batches, optimisers and training
- Classification and regression variants

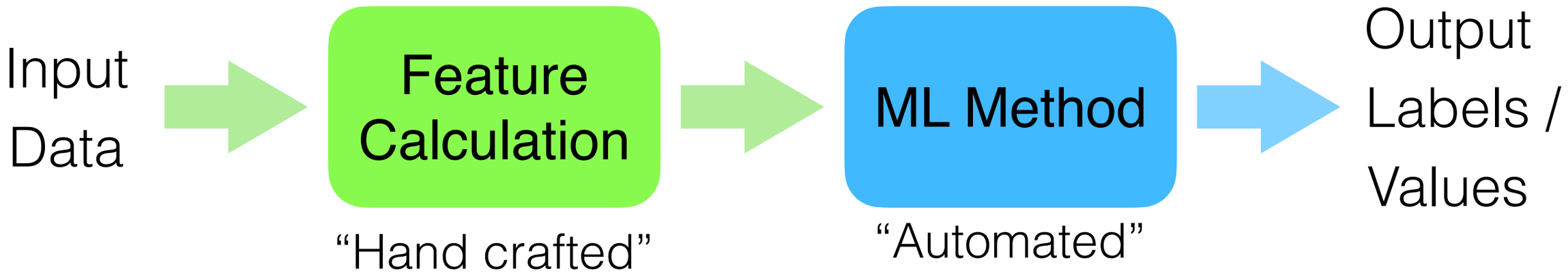
Introduction to Neural Networks

Traditional Machine Learning

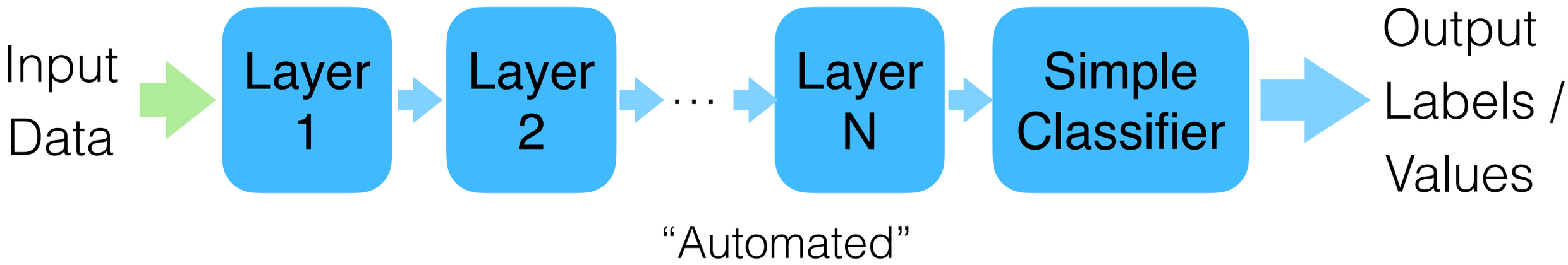


Introduction to Neural Networks

Classical Machine Learning

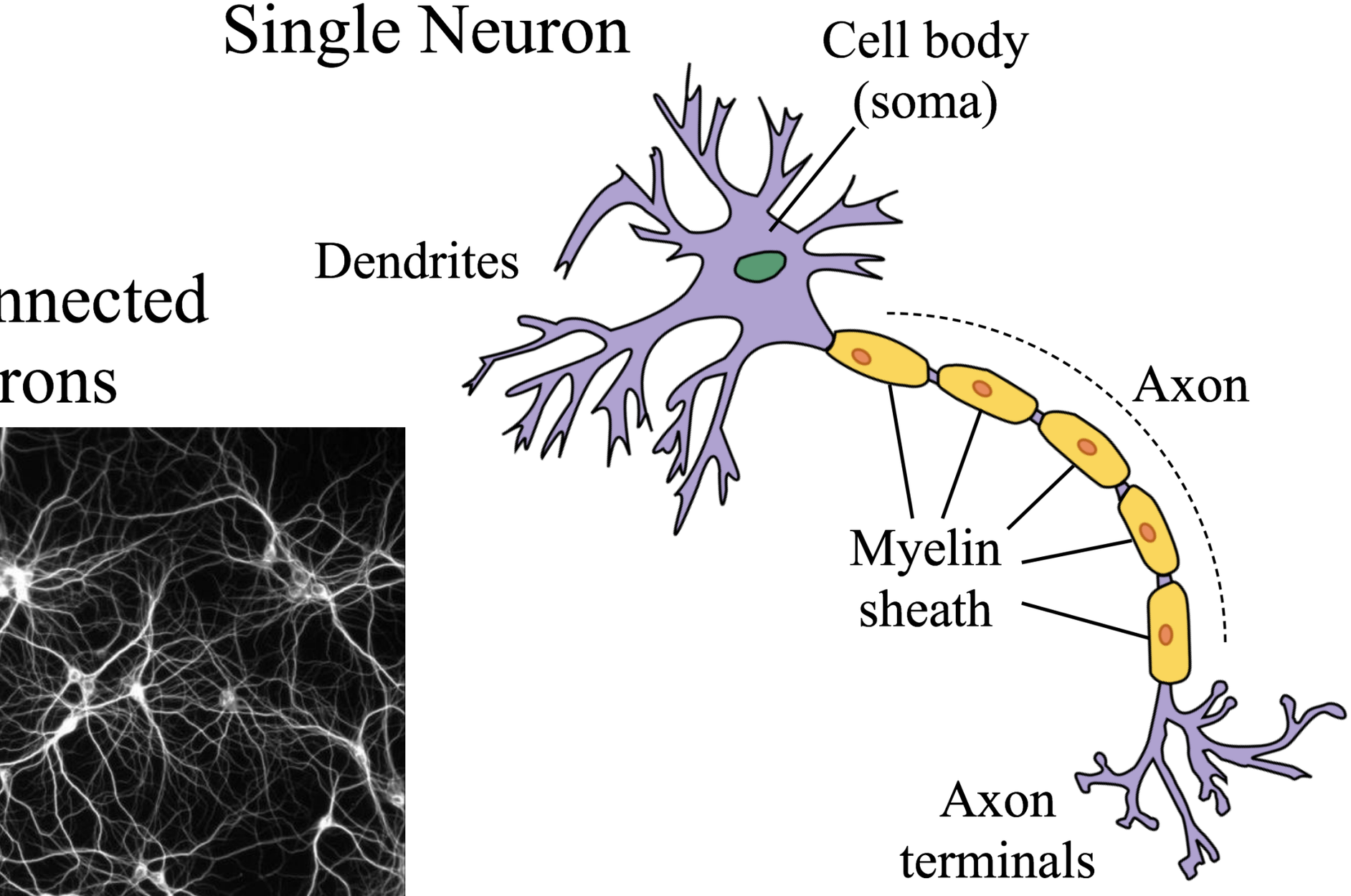


Deep Learning

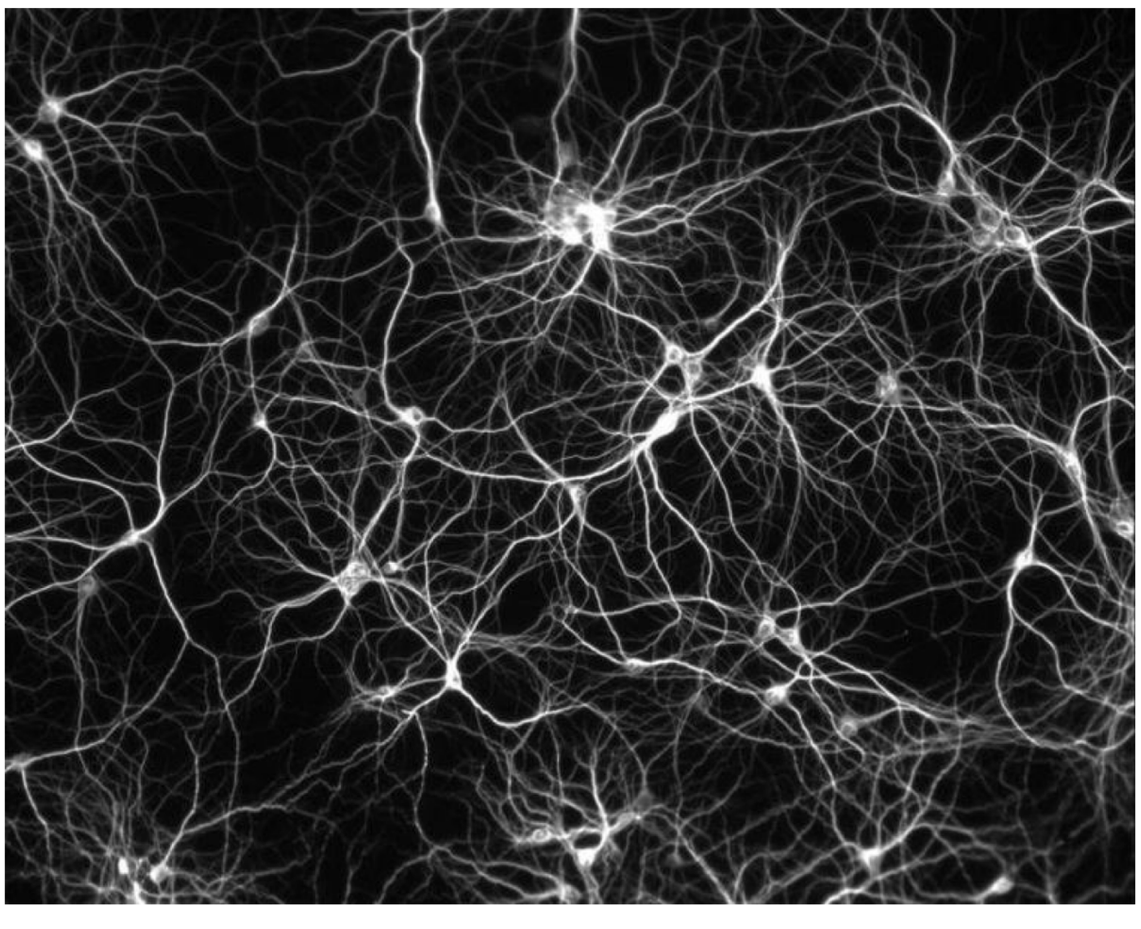


Neurons

Single Neuron

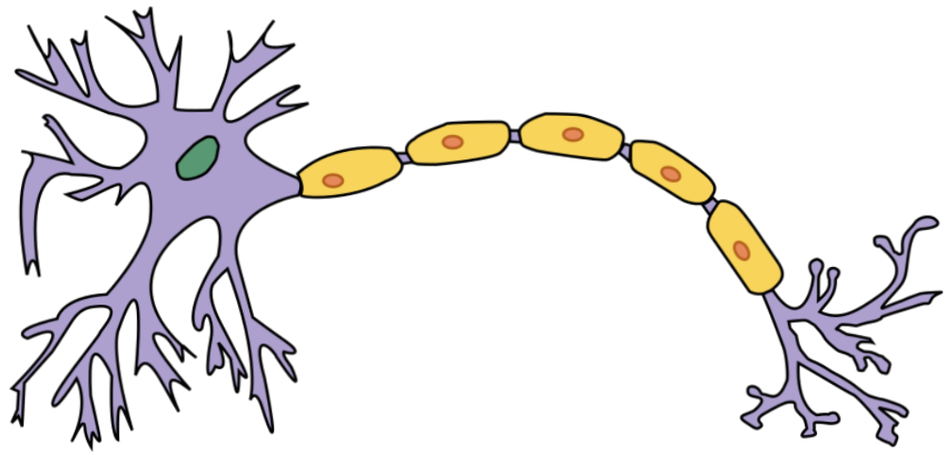


Interconnected Neurons



Jenkinson and Chappell; Introduction to Neuroimaging Analysis; OUP

Neurons



Inputs

Score function

$f = w.x + b$

x_1

x_2

x_3

x_n

w_1

w_2

w_3

w_n

b

f(...)

+1

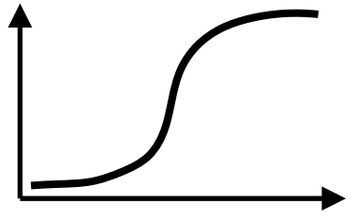
Bias term

Activation function

σ(f)

Output

y



$y = \sigma(w.x + b)$

$= \sigma(\Sigma_j w_j.x_j + b)$

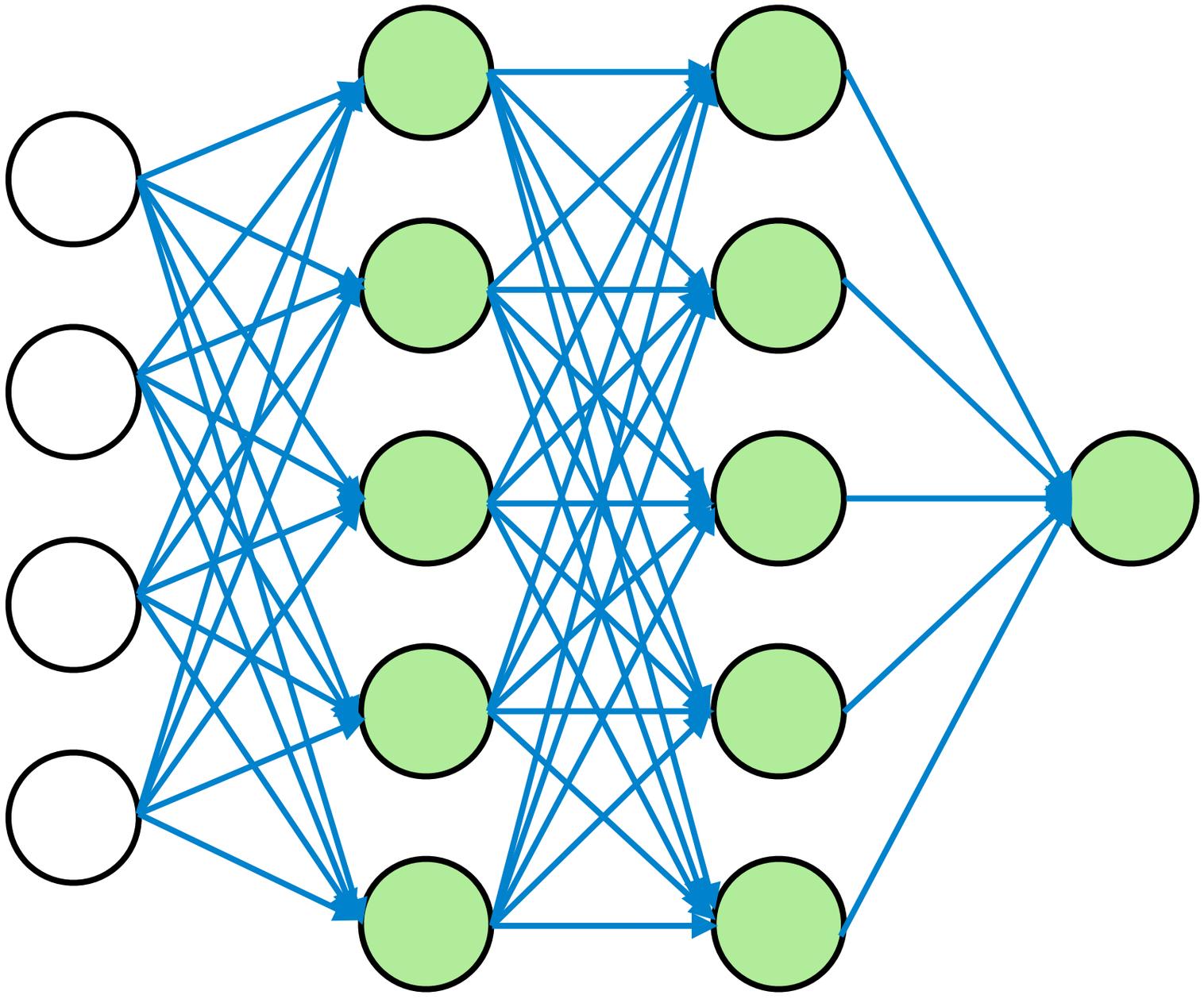
Neural Network

Input Layer

Hidden Layer 1

Hidden Layer 2

Output Layer



Fully Connected Layout

Neural Network

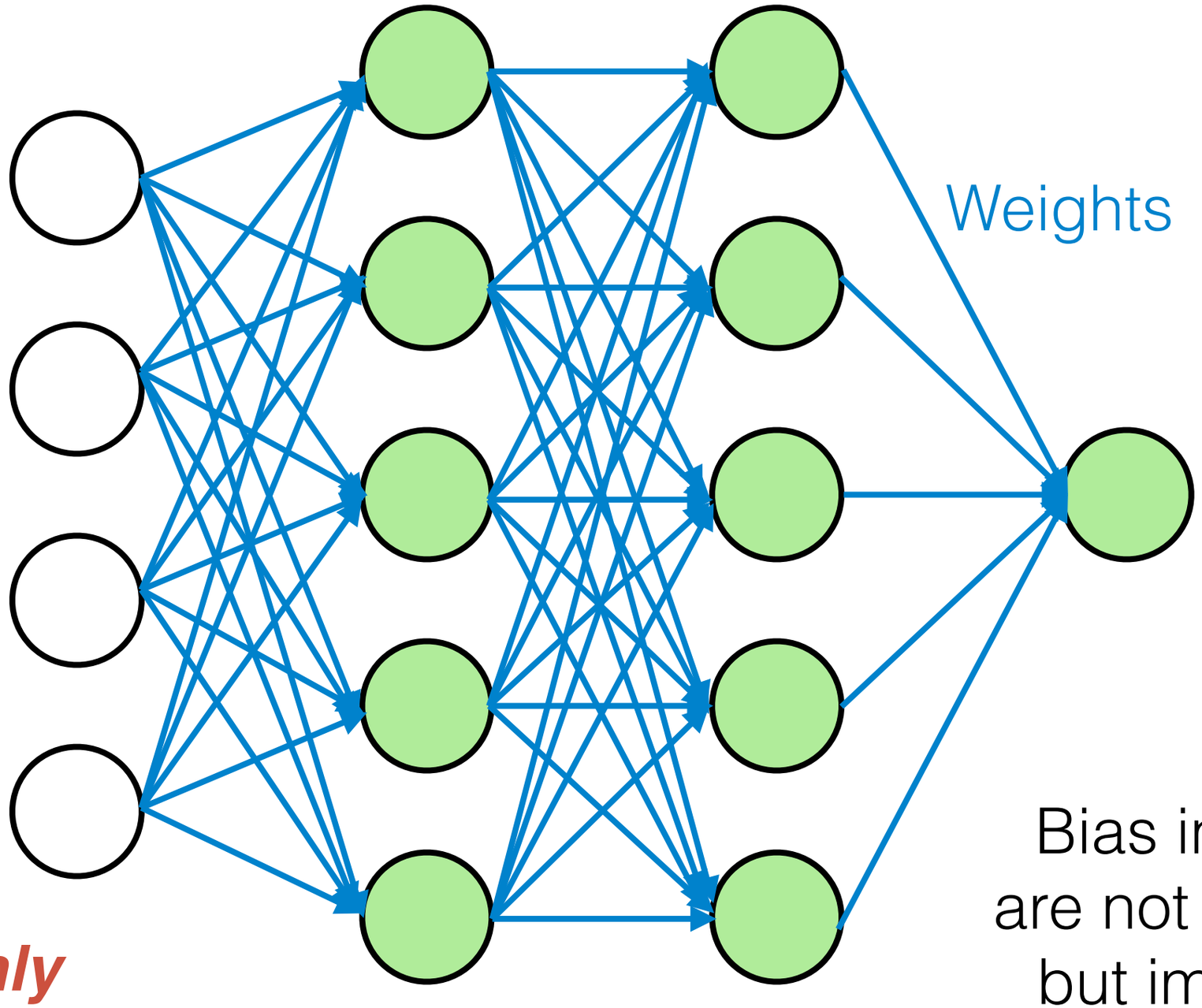
Input Layer

Hidden Layer 1

Hidden Layer 2

Output Layer

Input values as 1D vector



Weights

Fully Connected Layout

Each layer ***only*** connects to next layer

Bias inputs are not shown but implicit

Neural Network

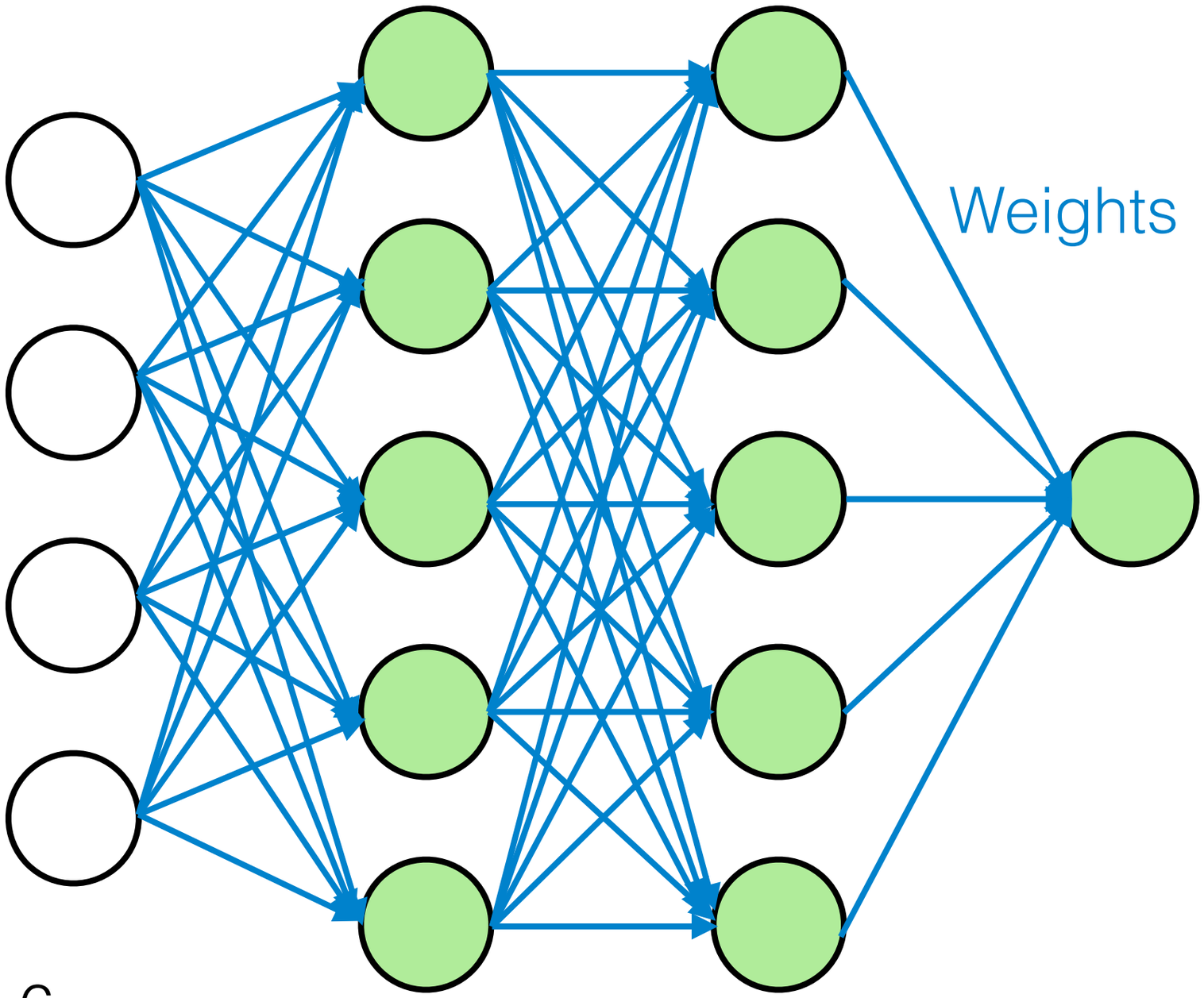
Input Layer

Hidden Layer 1

Hidden Layer 2

Output Layer

Parameters = # weights + # bias



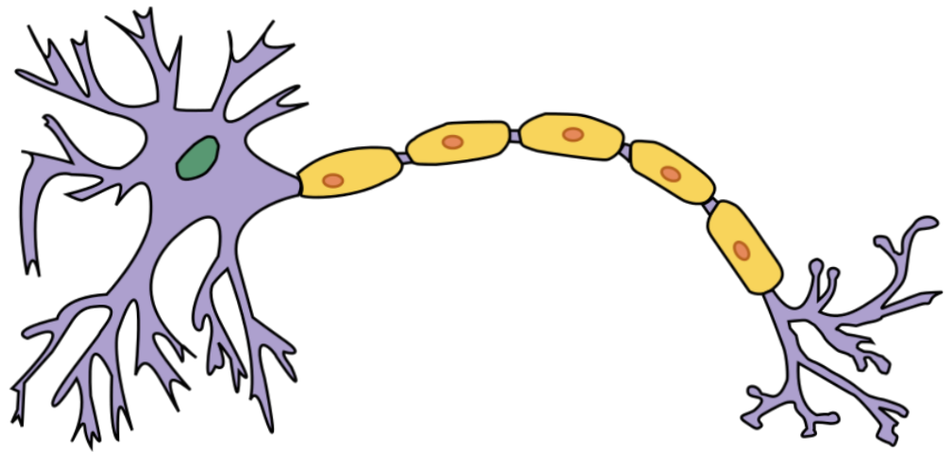
5 * 5 + 5 * 6 + 1 * 6 = 61 parameters

4 inputs + 1 bias

5 inputs + 1 bias

5 inputs + 1 bias

Importance of Nonlinearity



Inputs

x_1

x_2

x_3

x_n

w_1

w_2

w_3

w_n

Score function

$f = w.x + b$

f(...)

b

+1 Bias term

Activation function

σ(f)

Output

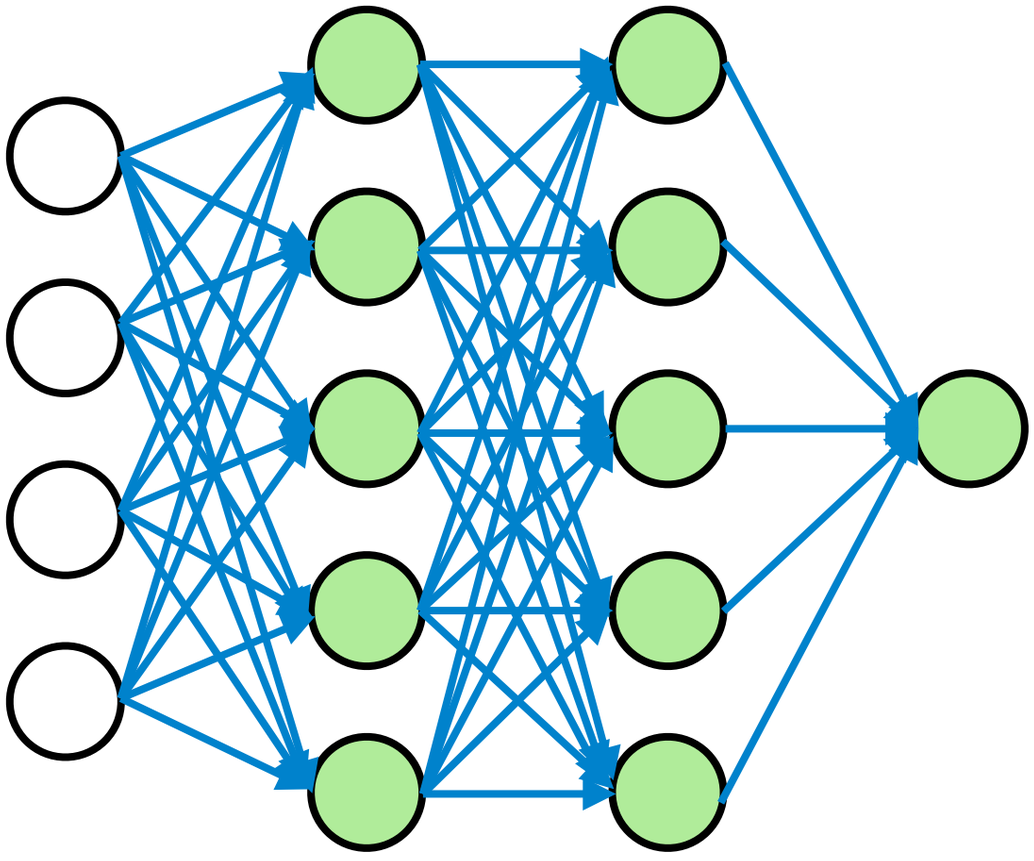
y

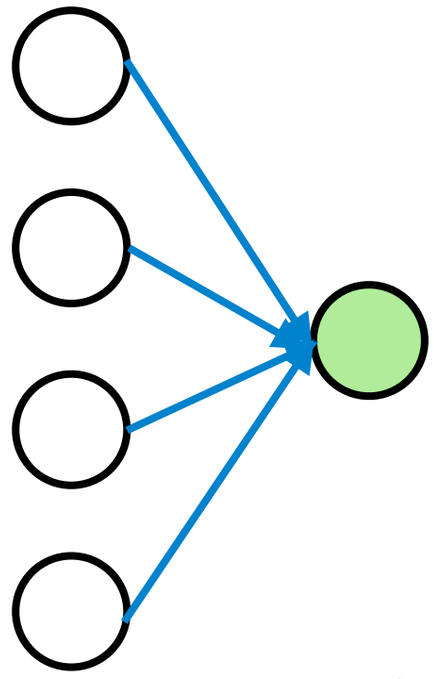
$y = \sigma(w.x + b)$

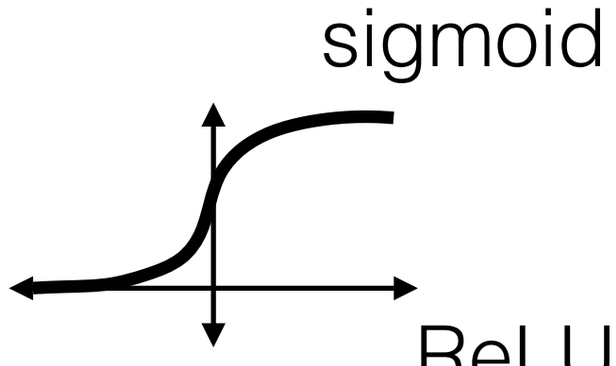
$= \sigma(\Sigma_j w_j.x_j + b)$

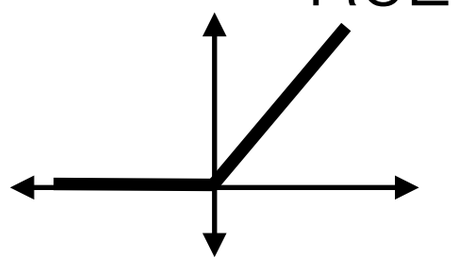
Importance of Nonlinearity

Without the nonlinearity the whole network just computes a linearly weighted sum and these are equivalent.





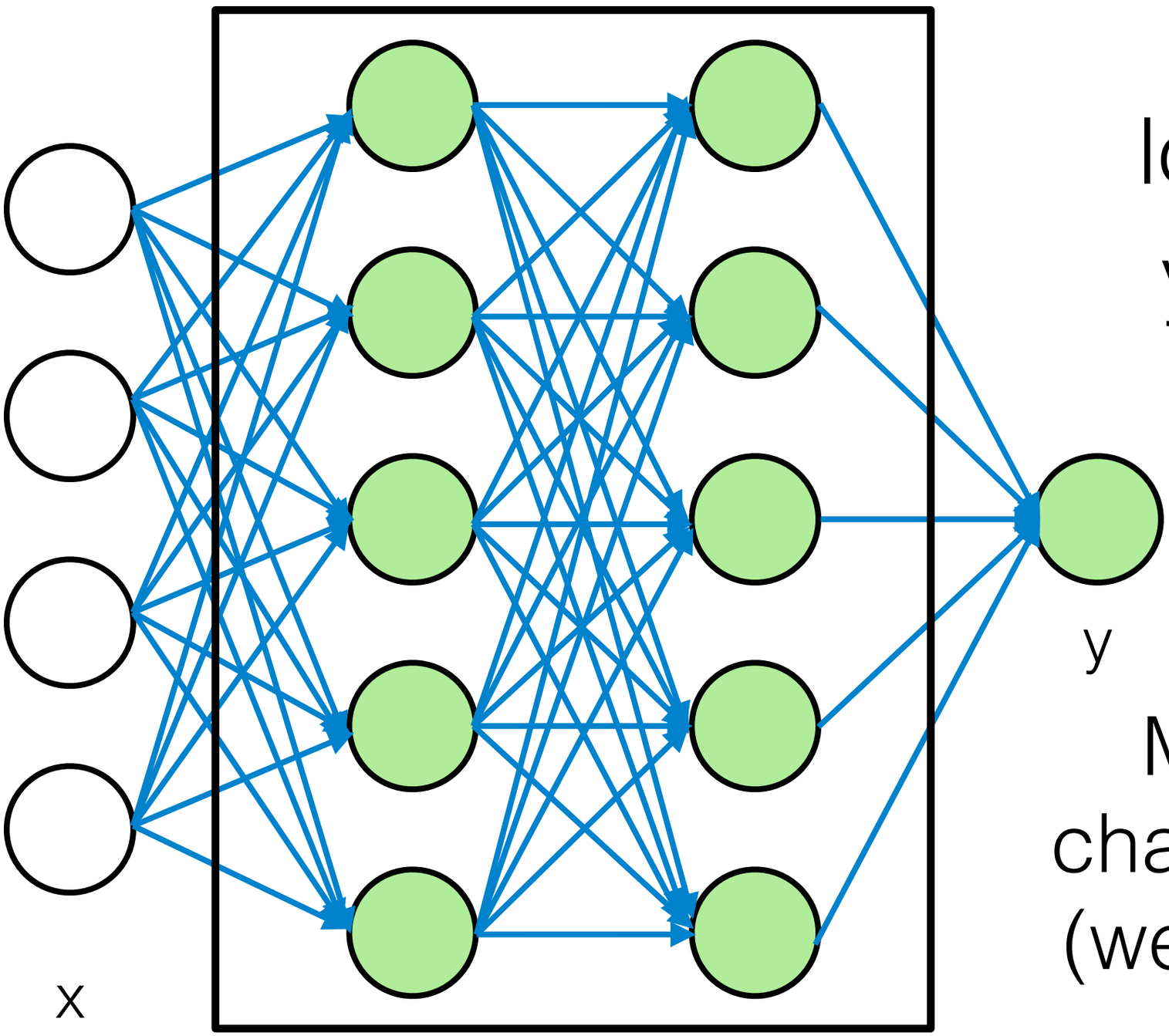




The nonlinearity is crucial, but a number of options exist, all similar to sigmoids.

ReLU (Rectified Linear Unit) is the most popular.

Loss Function



$$y = g(x; w, b)$$

$$loss(y_{pred}, y_{train})$$

$$y_{pred} = g(x_{train})$$

Minimise loss by changing parameters (weights and biases)

Sum loss over batches/epochs

Optimisation

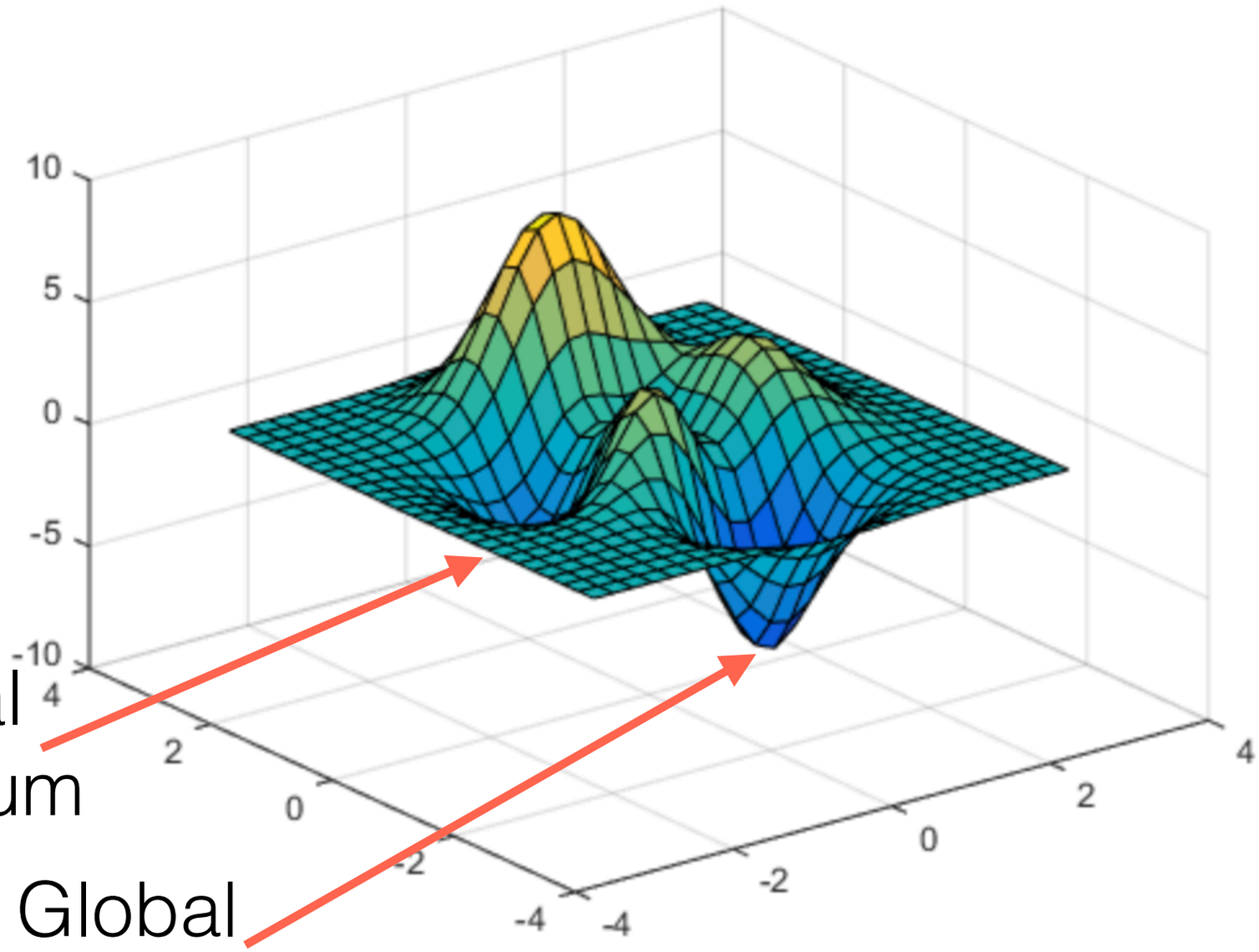
Optimisation method used to find minimum loss value

Typically use methods based on gradient descent: backpropagation

Important to have right:

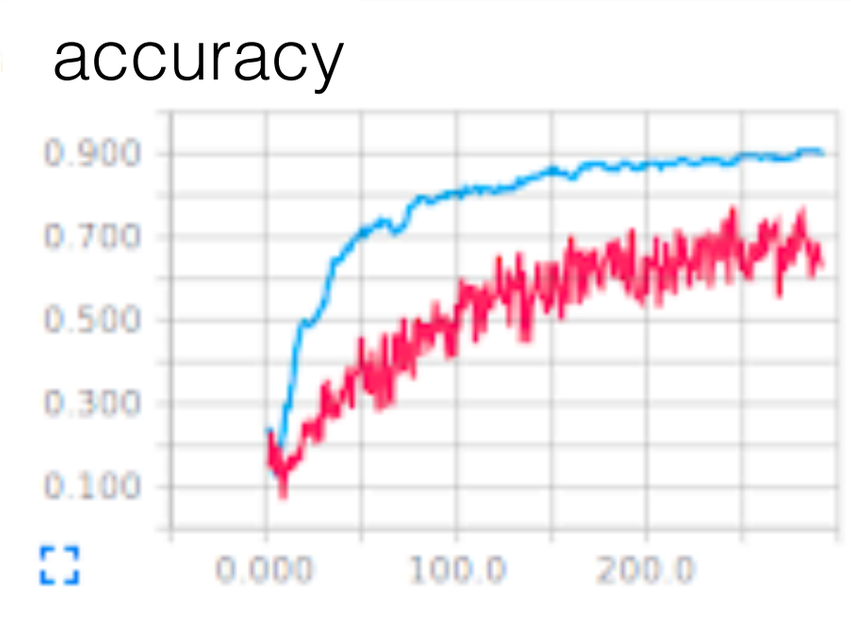
- initialisation
- learning rate

Loss function



Optimisation: Batches and Epochs

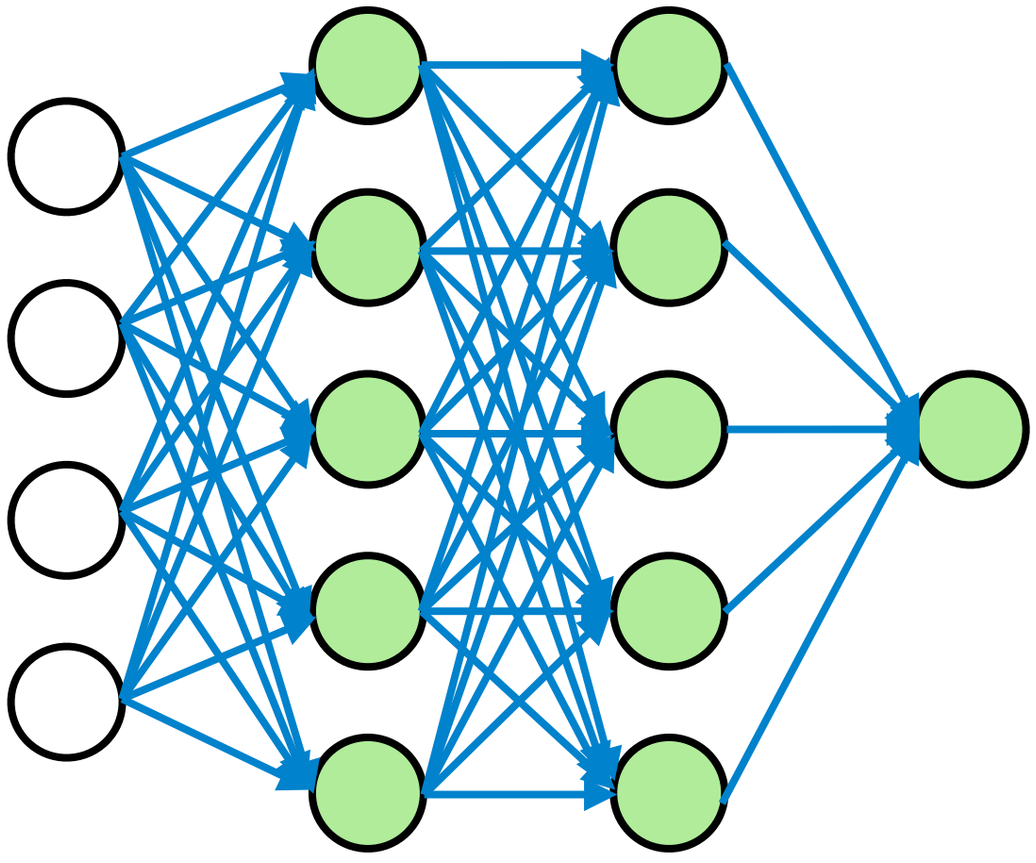
- Optimiser calculates gradients and updates once per batch
    - Batch size = number of samples
        - impacts memory requirements
        - impacts execution speed
- Repeat batches for whole training set
    - One epoch = all training samples
- Repeat epochs until convergence



Loss and Activation Functions: Regression

Loss function options:

- mean_squared_error
- mean_absolute_error



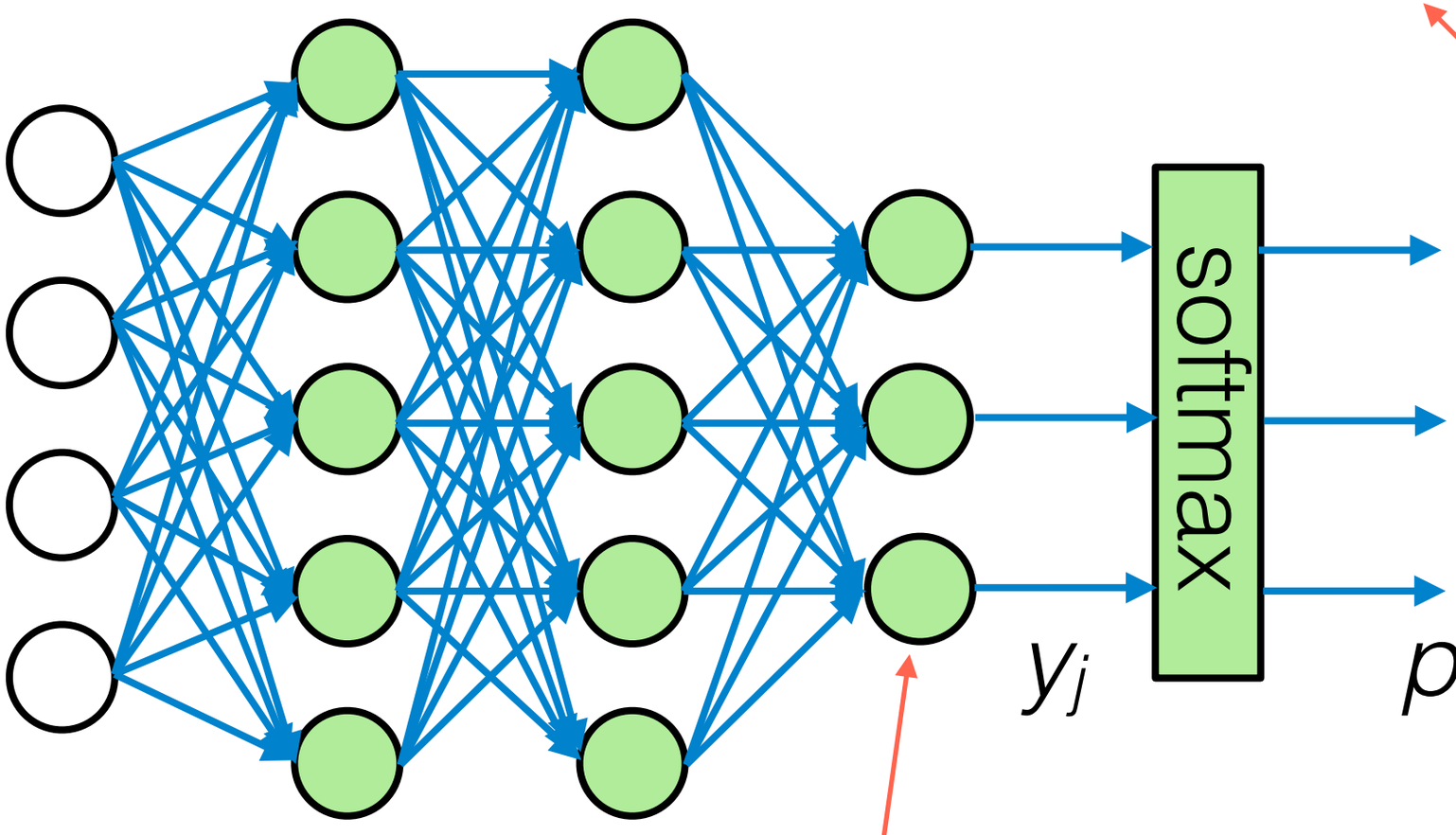
Activation function for final node:

- none (unconstrained)
- ReLU (positive range)
- sigmoid (limited range)

Loss and Activation Functions: Classification

Loss function options:

- binary_crossentropy
- categorical_crossentropy
- sparse_categorical_crossentropy



Each prefers extreme values (near 0 or 1)

Activation function for final node:

One-hot representation

softmax: $p_j = \frac{\exp(y_j)}{\sum_k \exp(y_k)}$

Summary

- Network architecture: fully connected layers, one to next
- Neurons: importance of nonlinear activation functions (ReLU)
- Parameters: weights and biases, number of parameters
- Loss functions, epochs, batches, optimisers
- Regression:
  - loss = mean squared error ; mean absolute error
  - activation = None, ReLU, sigmoid
- Classification:
  - loss = cross entropy variants
  - activation = softmax (with one-hot representation)