Lecture 8

Artificial Neural Network

Dr. Le Huu Ton

Hanoi, 2017

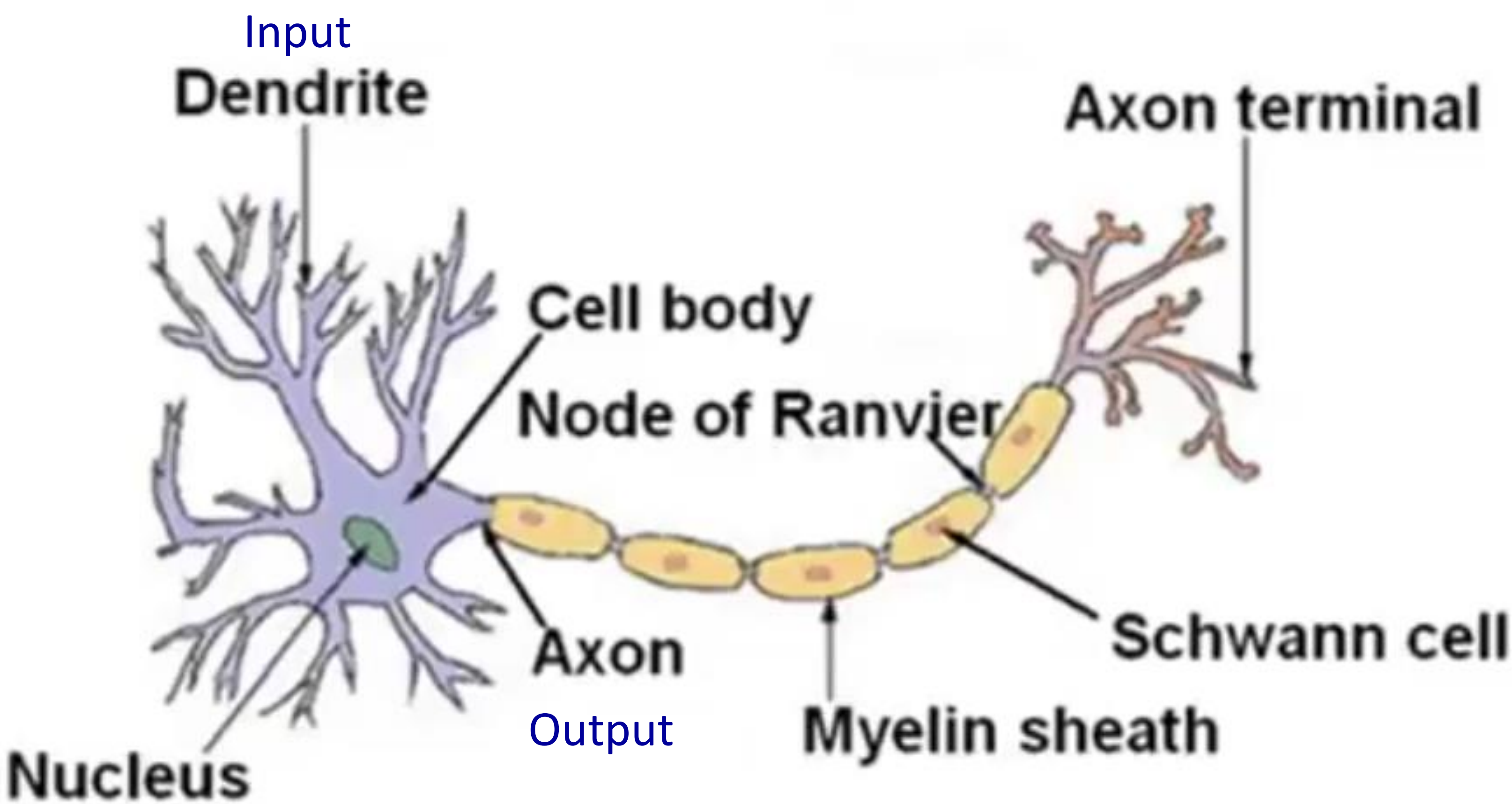
Outline

- Artificial Neural Network
- Back Propagation Gradient Descent
- Model Evaluation

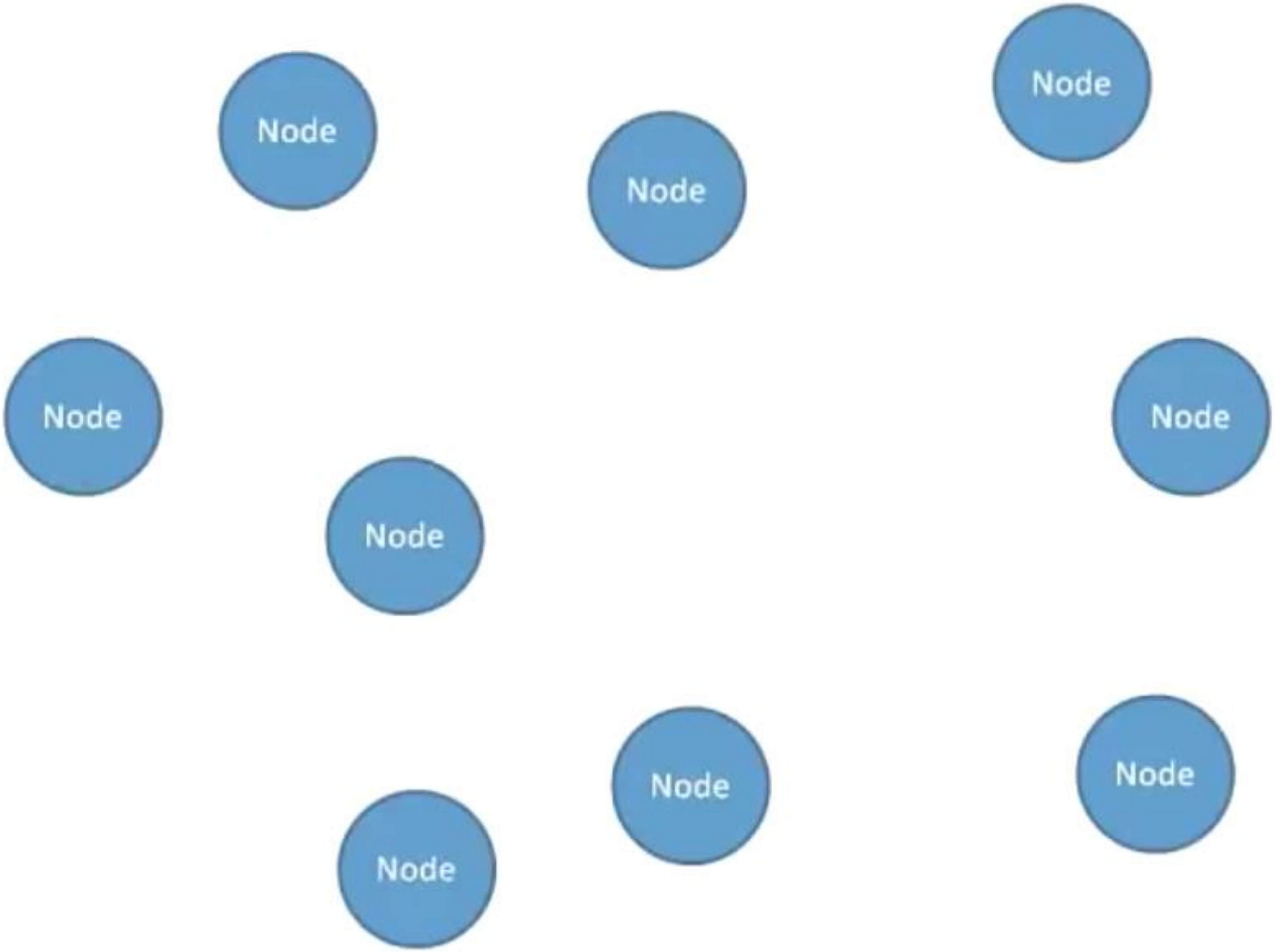
Outline

- Artificial Neural Network
- Back Propagation Gradient Descent
- Model Evaluation

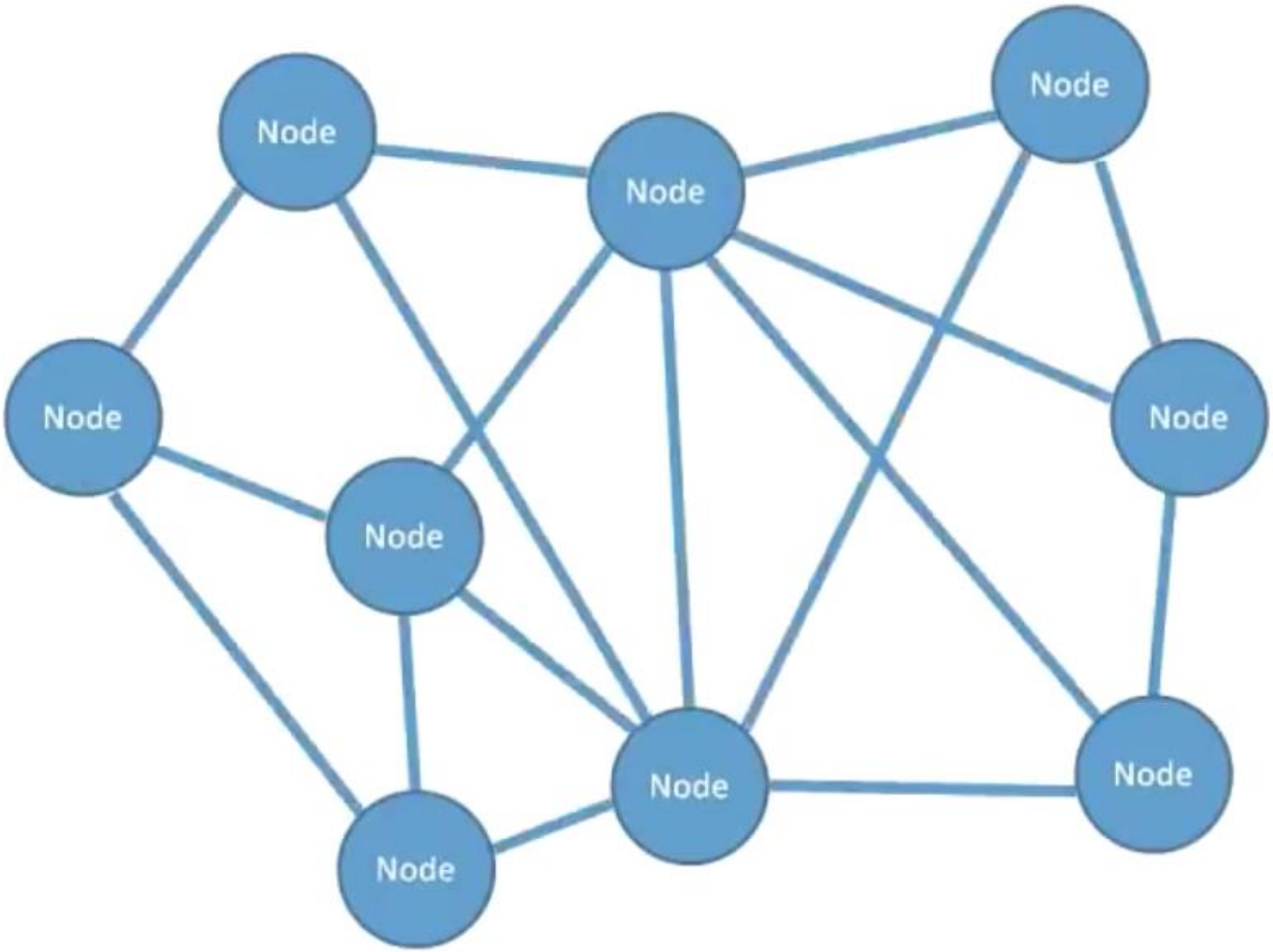
Artificial Neural Network



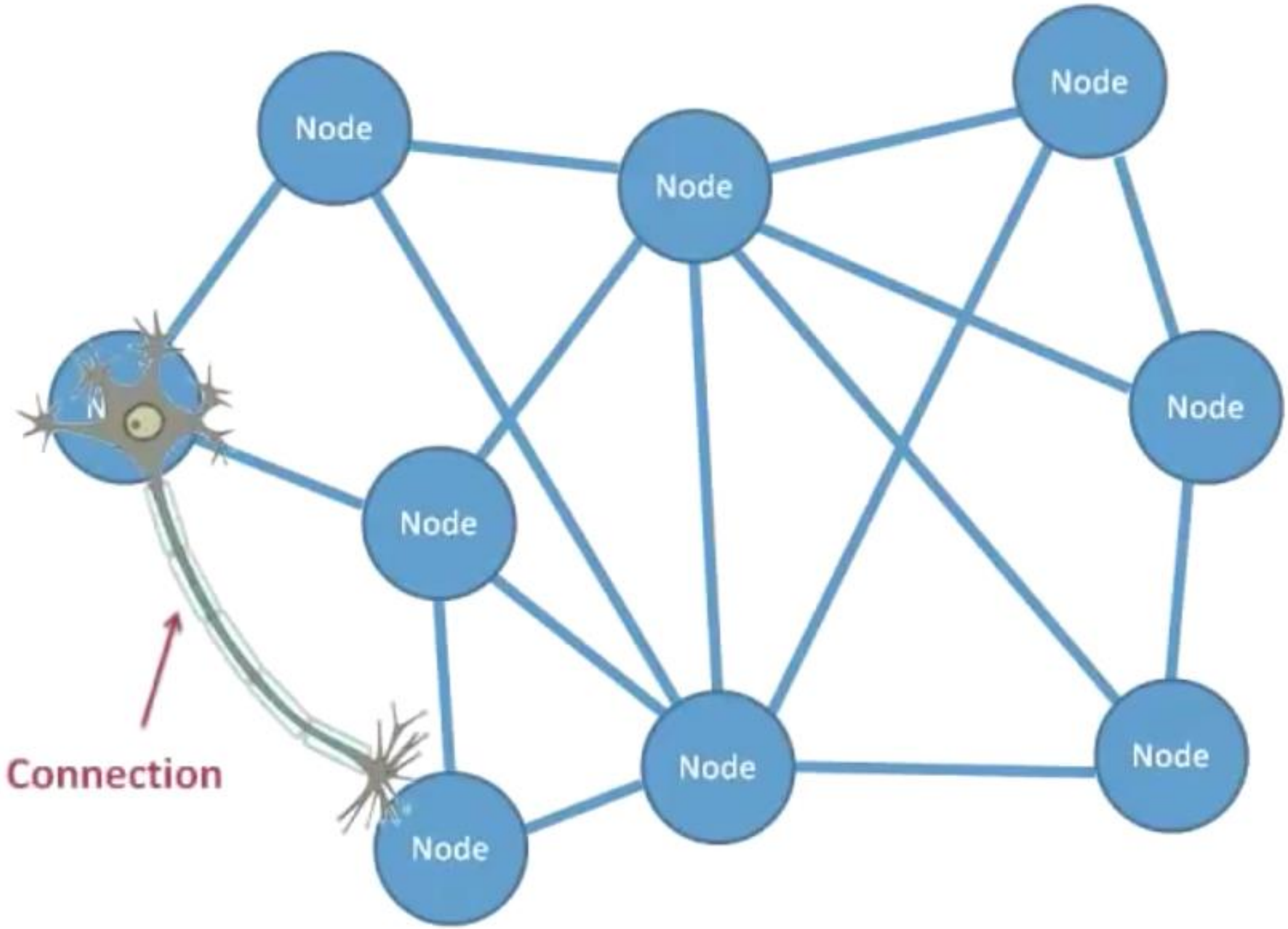
Artificial Neural Network



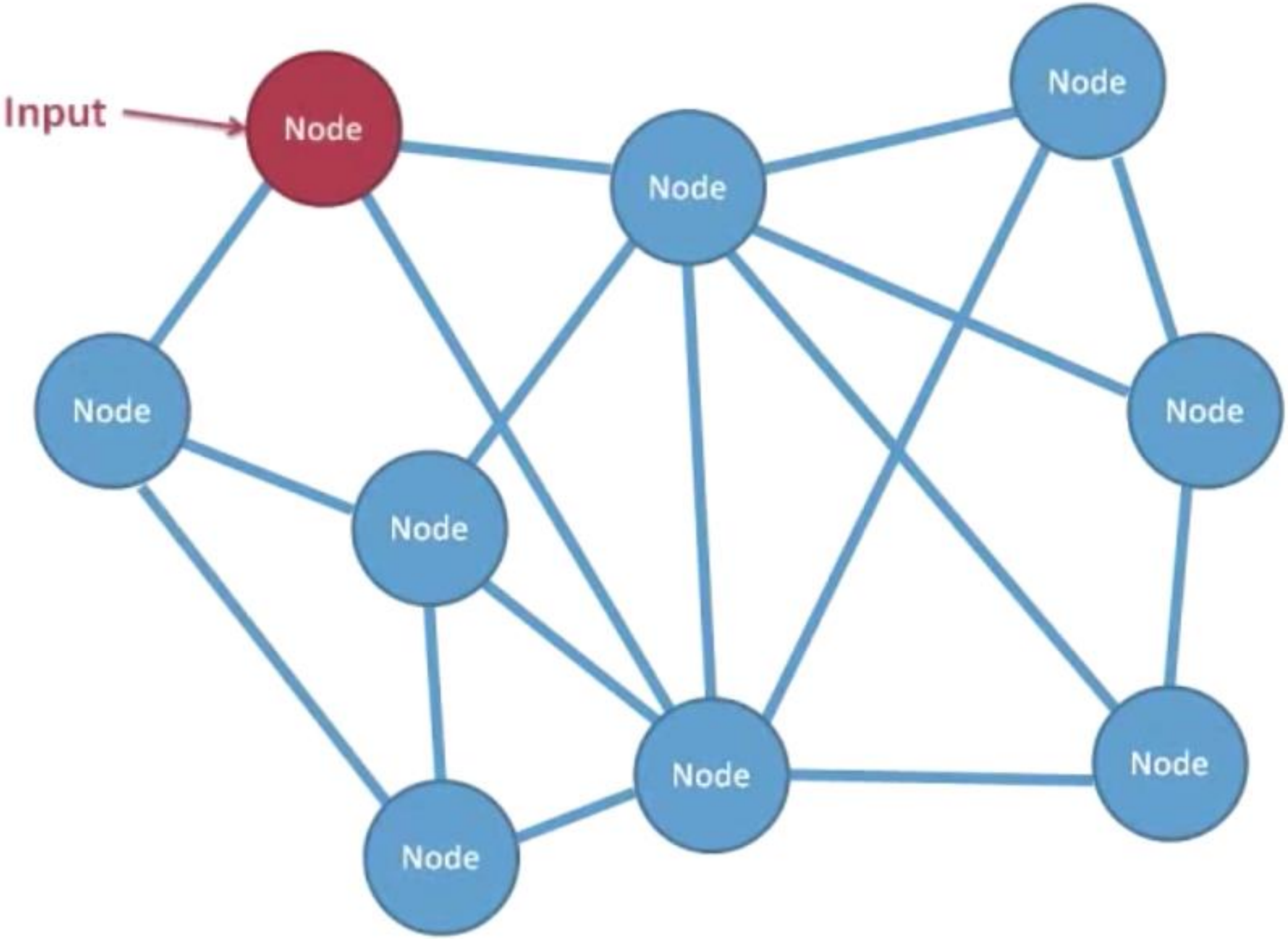
Artificial Neural Network



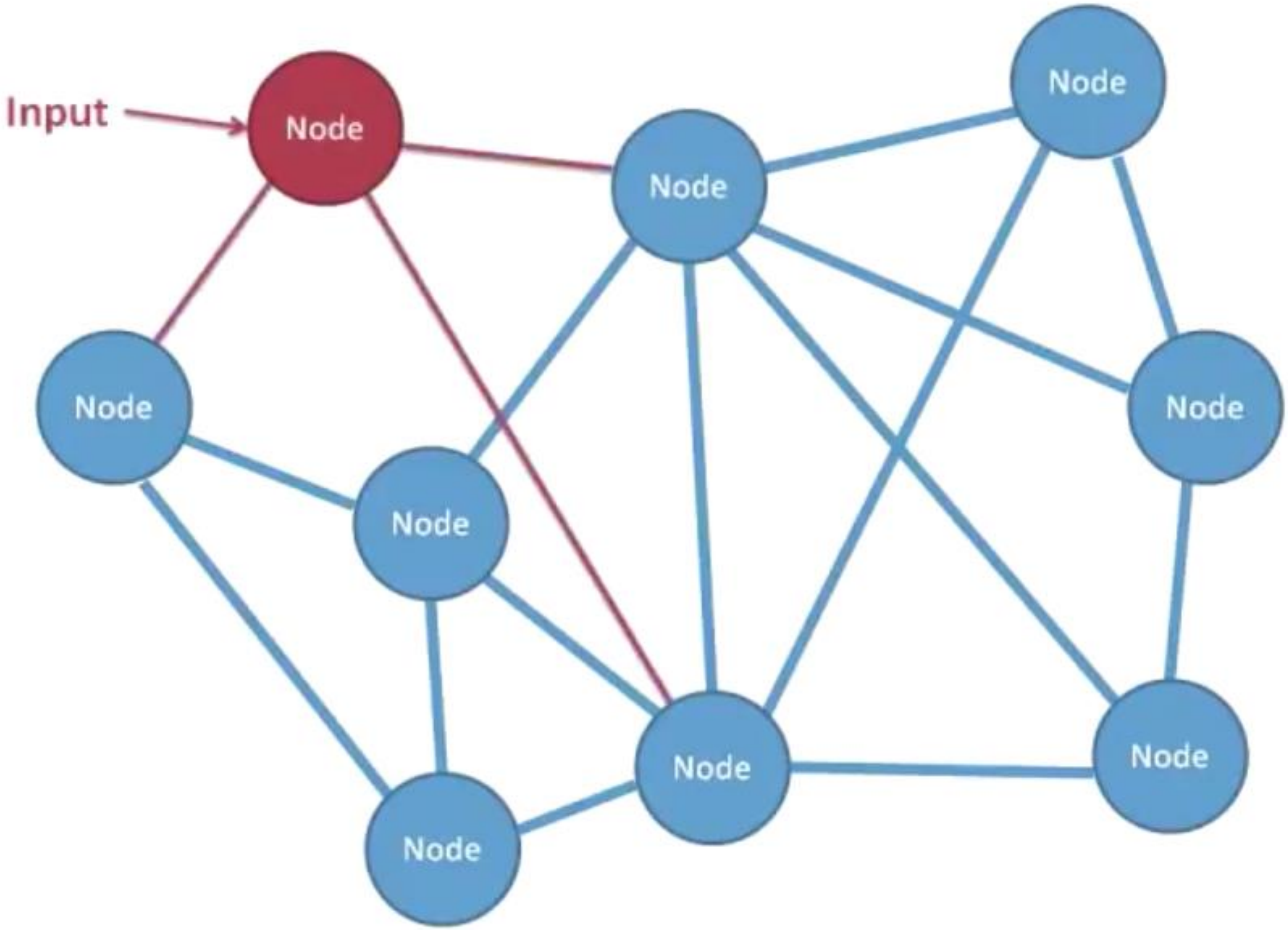
Artificial Neural Network



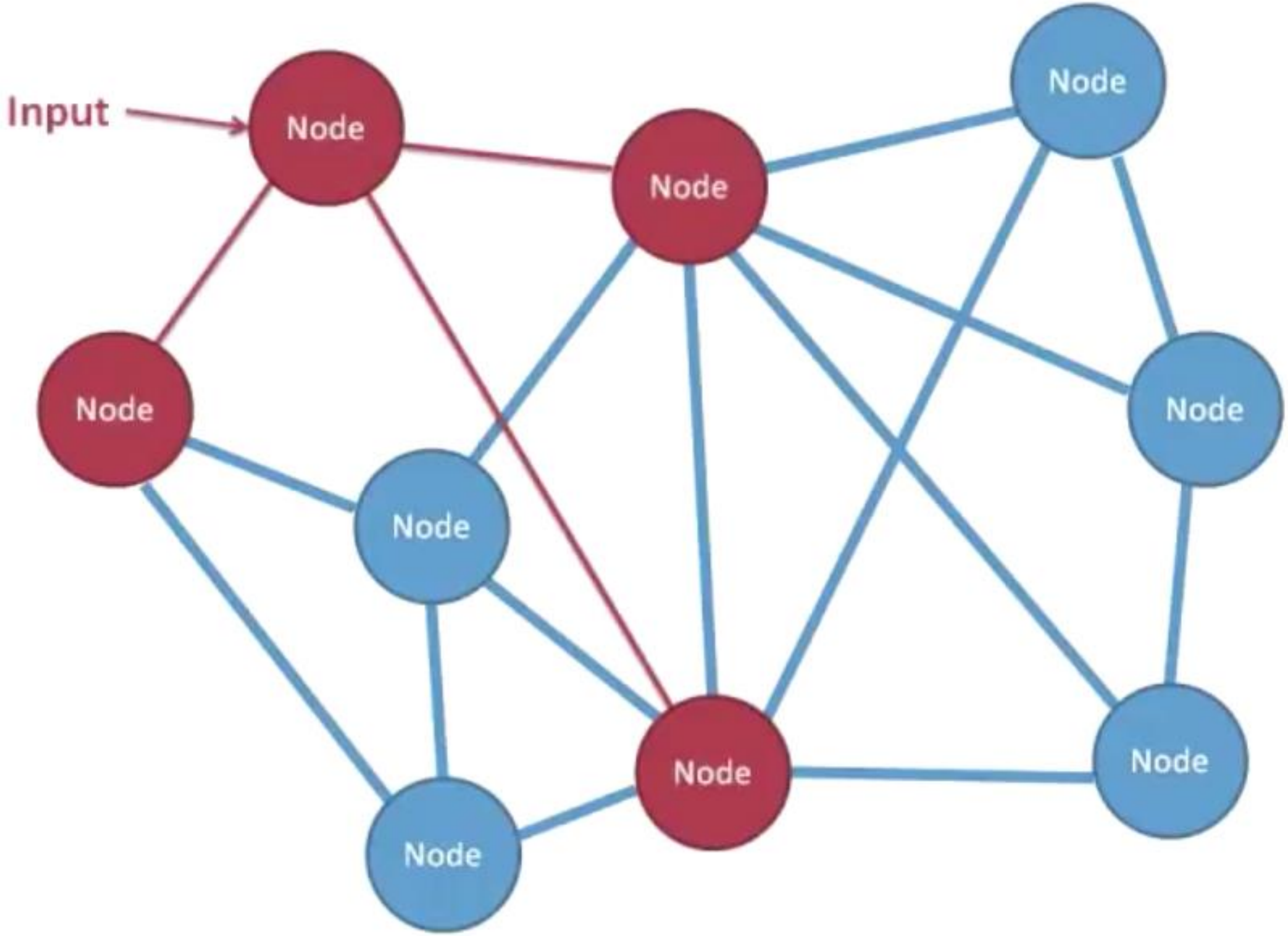
Artificial Neural Network



Artificial Neural Network

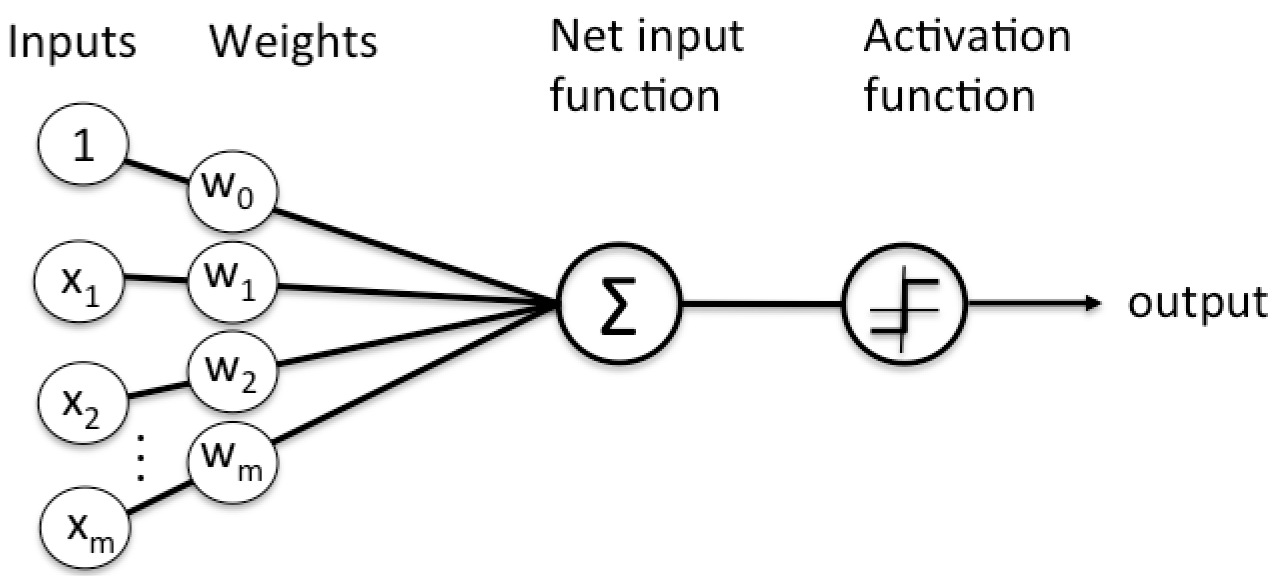


Artificial Neural Network



Artificial Neural Network

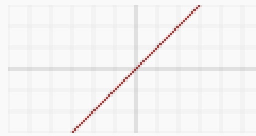
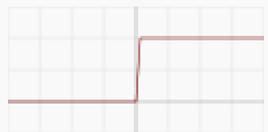
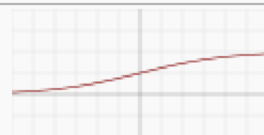
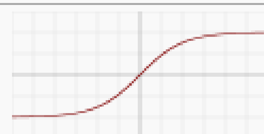
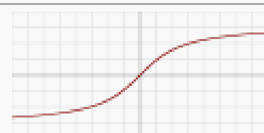
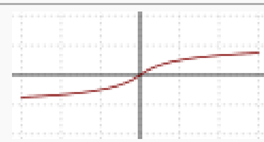
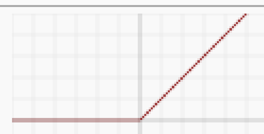
Perceptron: mimic the operation of neuron



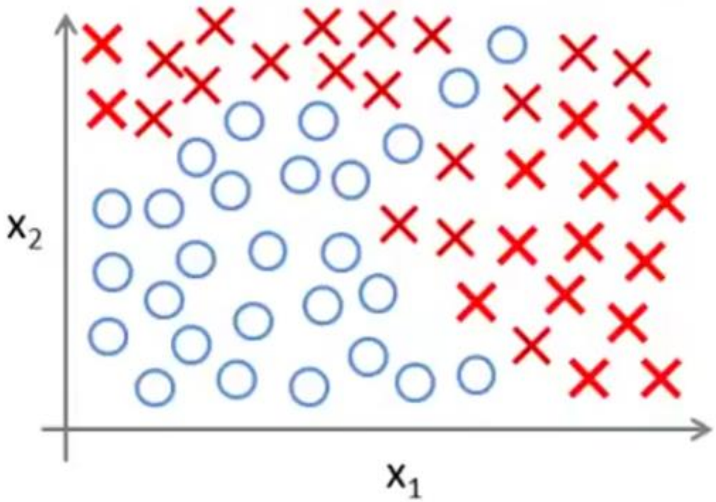
$$net = 1 * w_0 + x_1 * w_1 + x_2 * w_2 + x_3 * w_3 = w^T x$$

$$y = f(z) : activation\ \ function$$

Popular activation function

Identity		$f(x) = x$
Binary step		$f(x) = \begin{cases} 0 & \text{for} & x < 0 \\ 1 & \text{for} & x \geq 0 \end{cases}$
Logistic (a.k.a Soft step)		$f(x) = \frac{1}{1 + e^{-x}}$
TanH		$f(x) = \tanh(x) = \frac{2}{1 + e^{-2x}} - 1$
ArcTan		$f(x) = \tan^{-1}(x)$
Softsign [7][8]		$f(x) = \frac{x}{1 + \lvert x \rvert}$
Rectifier (ReLU)[9]		$f(x) = \begin{cases} 0 & \text{for} & x < 0 \\ x & \text{for} & x \geq 0 \end{cases}$

Neural Network and logistic regression



$$g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1 x_2 + \theta_4 x_1^2 x_2 + \theta_5 x_1^3 x_2 + \theta_6 x_1 x_2^2 + \ldots)$$

Many features, and their relation is complex

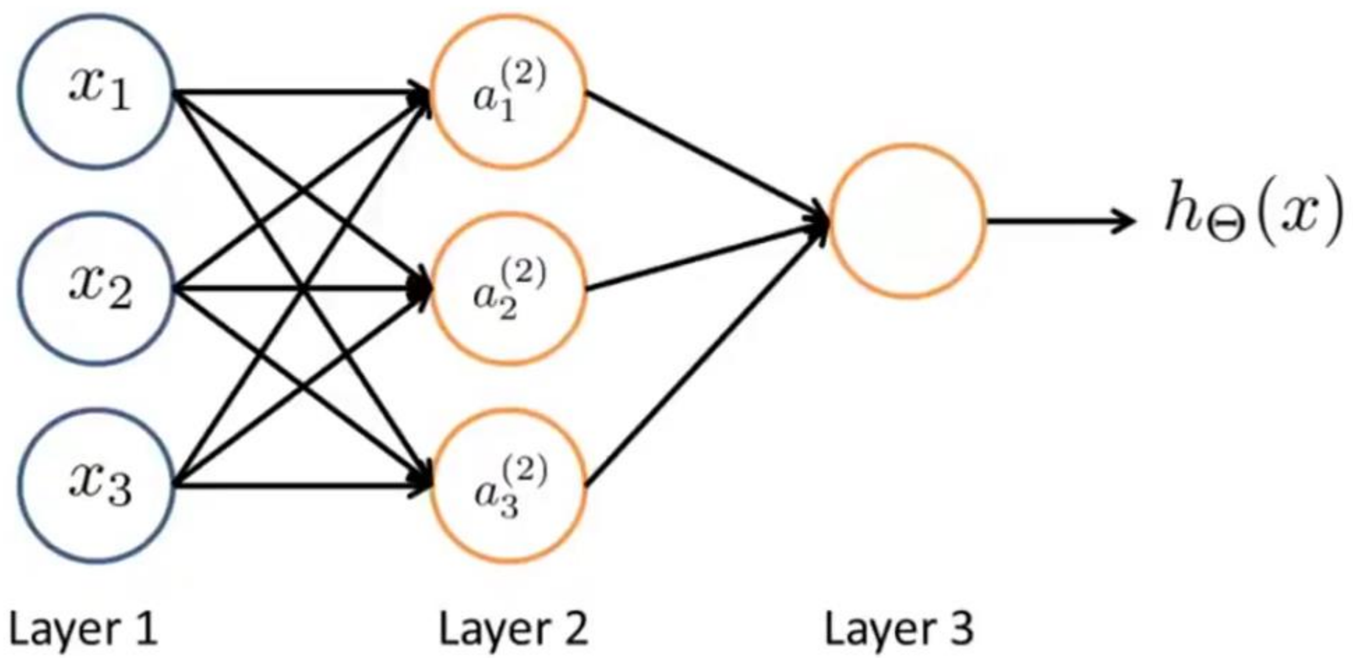
100 original features => 5000 second order features

=> 170.000 third order features

=> Not really a good way to introduce too many features to build non-linear classification

Artificial Neural Network

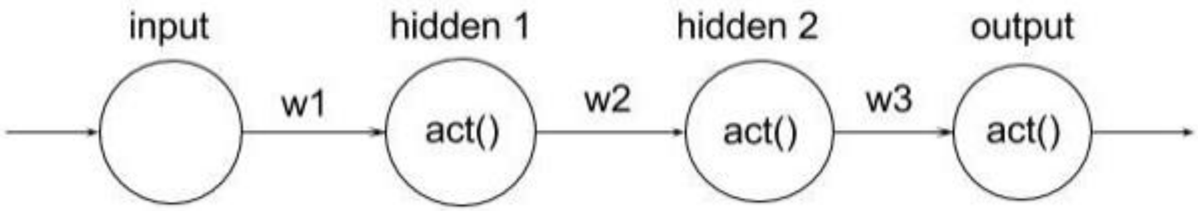
Artificial Neural Network: neurons connect to each others



Input Layer　　Hidden Layer　　Output layer

Artificial Neural Network

Neural Network: feed forward structure

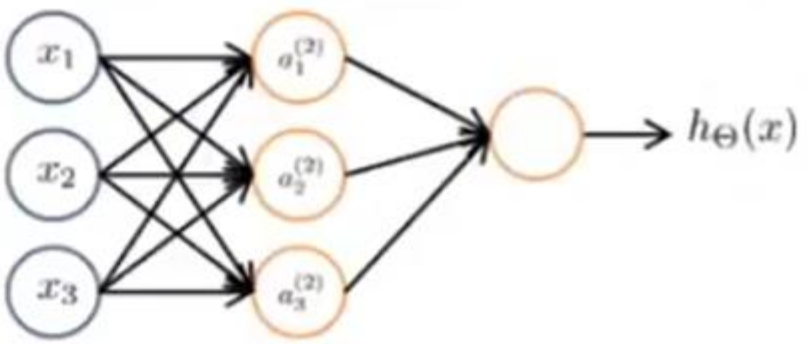


$$a^1 = g(w^1 x)$$

$$a^2 = g(w^2 a^1)$$

$$output = a^3 = g(w^3 a^2)$$

Neural Network: feed forward structure



$a_i^{(j)}$ = "activation" of unit i in layer j

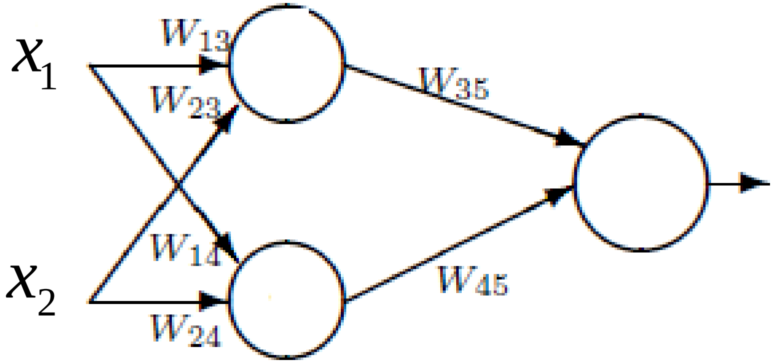
w^j = matrix of weights controlling function mapping from layer j to layer $j+1$

$$a_1^2 = g(w_{10}^1 x_0 + w_{11}^1 x_1 + w_{12}^1 x_2 + w_{13}^1 x_3)$$
$$a_2^2 = g(w_{20}^1 x_0 + w_{21}^1 x_1 + w_{22}^1 x_2 + w_{23}^1 x_3)$$
$$a_3^2 = g(w_{30}^1 x_0 + w_{31}^1 x_1 + w_{32}^1 x_2 + w_{33}^1 x_3)$$
$$h(x) = a_1^3 = g(w_{10}^2 a_0^2 + w_{11}^2 a_1^2 + w_{12}^2 a_2^2 + w_{13}^2 a_3^2)$$

Artificial Neural Network

Exercise: Calculate the output of the following network with x_1=1, x_2=0;

$w_{13} = 2$ $w_{23} = -3$	$w_{35} = 2$ $w_{45} = -1$
$w_{14} = 1$ $w_{24} = 4$	



Given the following activation function

$$f(v) = \begin{cases} 1 & \text{if } v \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

Outline

- Artificial Neural Network
- Back Propagation Gradient Descent

Artificial Neural Network

How to train the network: Gradient descents algorithm

Cost function

$$E(w) = \frac{1}{2m} \sum (y - h(x))^2$$

Where y is the expected value of the output

h(x) is the predicted value of the output

$$w := w - \alpha \frac{\partial E}{\partial_w}$$

Artificial Neural Network

Back propagation: Update the weighting w layer by layer, starting from the last layer of the network.

Apply the chain rule derivation

$$\frac{\partial z}{\partial x} = \frac{\partial z}{\partial y} \cdot \frac{\partial y}{\partial x}$$

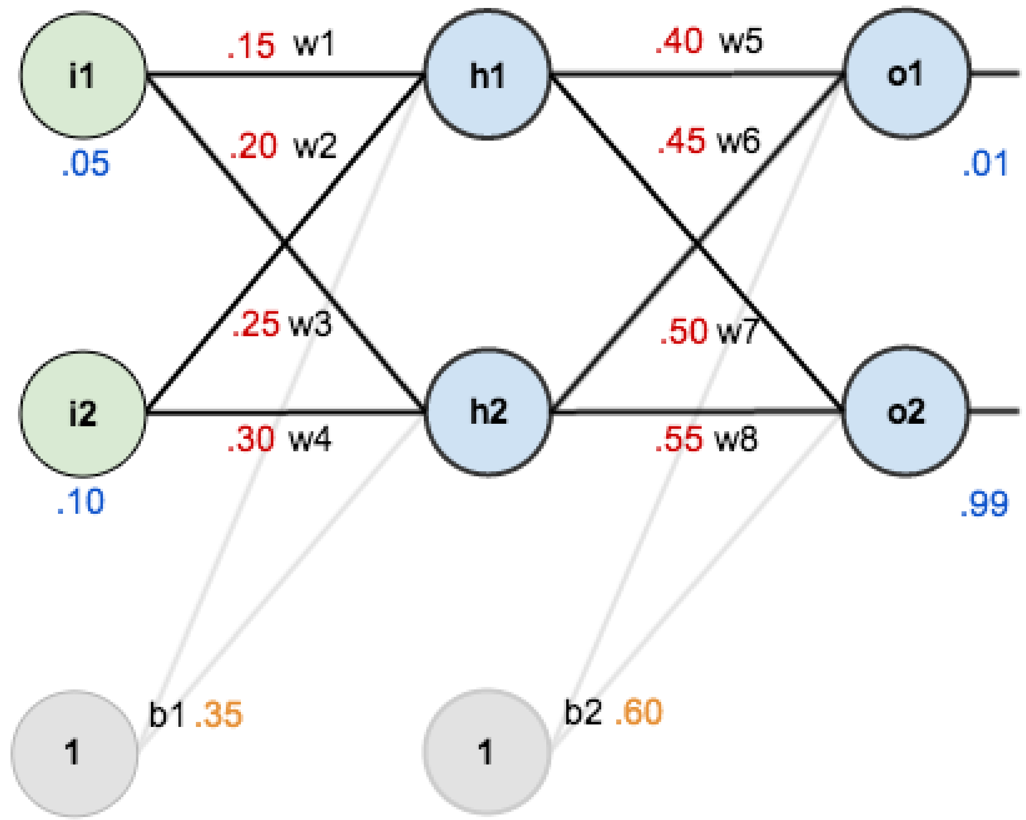
$$\frac{\partial E}{\partial w} = \frac{\partial E}{\partial a} \cdot \frac{\partial a}{\partial net} \cdot \frac{\partial net}{\partial w}$$

Artificial Neural Network

Example: Given the following network with follow with initial value and training data. Apply the back propagation to update the weighting w

The output of the network

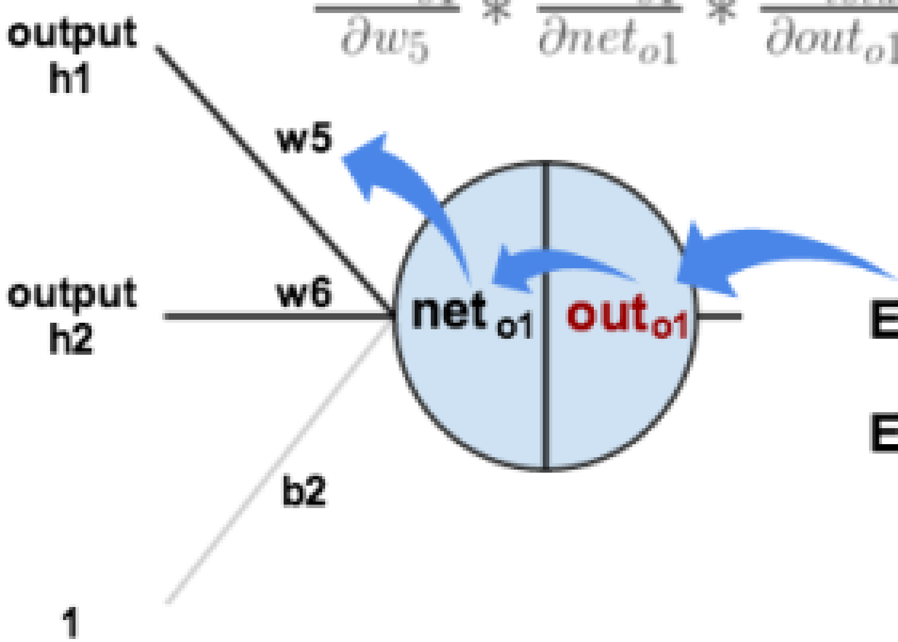
The total error



Artificial Neural Network

Using back propagation gradient descent to update w_5

$$\frac{\partial net_{o1}}{\partial w_5} * \frac{\partial out_{o1}}{\partial net_{o1}} * \frac{\partial E_{total}}{\partial out_{o1}} = \frac{\partial E_{total}}{\partial w_5}$$

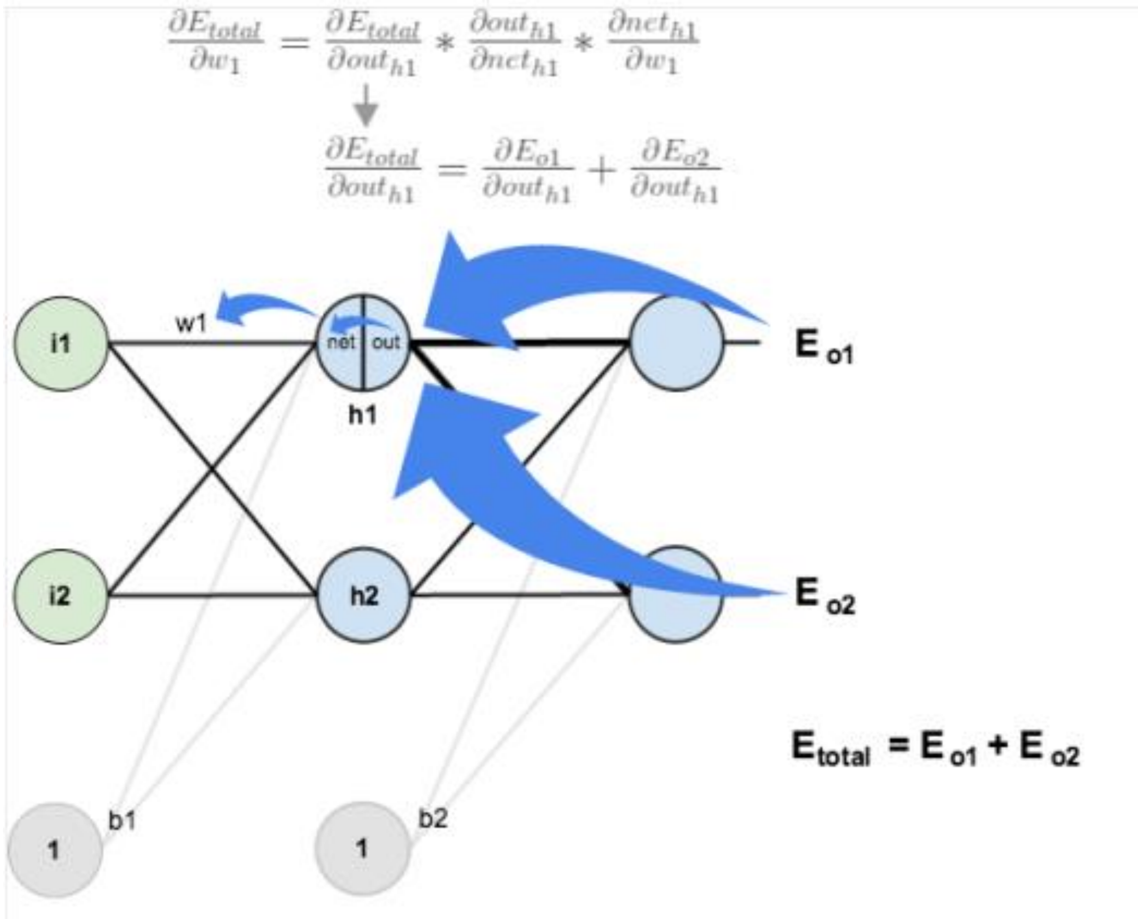


$E_{o1} = ½(target_{o1} - out_{o1})^2$

$E_{total} = E_{o1} + E_{o2}$

Artificial Neural Network

Using back propagation gradient descent to update w_1



Outline

- Artificial Neural Network
- Back Propagation Gradient Descent
- Model Evaluation

Model Evaluation

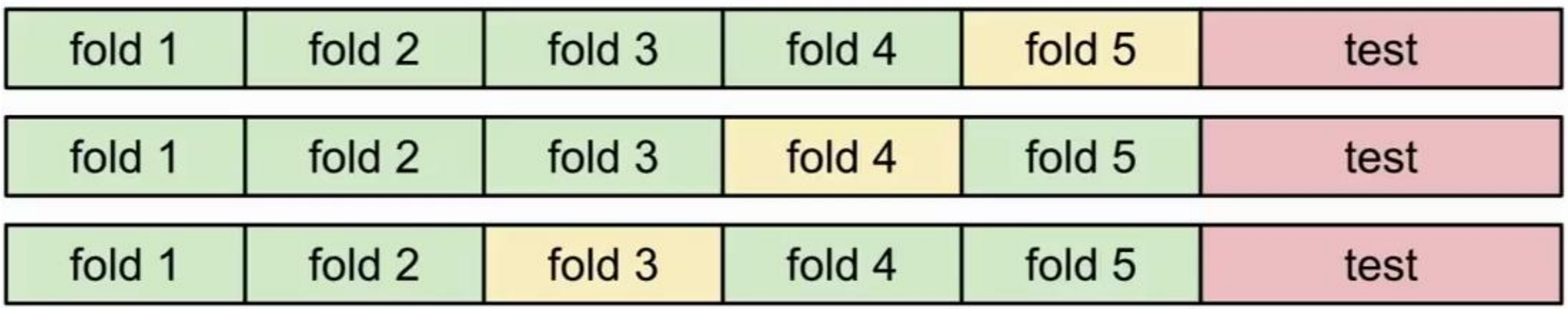
Idea #1: Choose hyperparameters
that work best on the data

Idea #2: Split data into **train** and **test**, choose
hyperparameters that work best on test data

Idea #3: Split data into **train**, **val**, and **test**; choose
hyperparameters on val and evaluate on test

Model Evaluation

Idea #4: **Cross-Validation**: Split data into **folds**,
try each fold as validation and average the results



Useful for small datasets, but not used too frequently in deep learning

Model Evaluation

- **Training Dataset**: The sample of data used to fit the model.
- **Validation Dataset**: The sample of data used to provide an unbiased evaluation of a model fit on the training dataset while tuning model hyperparameters. The evaluation becomes more biased as skill on the validation dataset is incorporated into the model configuration.
- **Test Dataset**: The sample of data used to provide an unbiased evaluation of a final model fit on the training dataset.

Evaluation of Regression Model

- Mean Absolute Error.
- Mean Squared Error.
- R Square (R^2)

Evaluation of Regression Model

Mean Absolute Error: is the sum of the absolute differences between predictions and actual values

$$MAE = \frac{1}{m}\sum_{i=1}^{m} |y^i - h(x^i)|$$

Evaluation of Regression Model

Mean Square Error: measures the average of the squares of the errors- the difference between the real value and the predicted one

$$MSE = \frac{1}{m}\sum_{i=1}^{m}\left(y^i - h(x^i)\right)^2$$

Evaluation of Regression Model

R Square: provides an indication of the goodness of fit of a set of predictions to the actual values. In statistical literature, this measure is called the coefficient of determination.

The total sum of squares (proportional to the variance of the data):

$$SS_{tot} = \sum_{i=1}^{m} \left(y^i - \bar{y}\right)^2$$

The sum of squares of residuals, also called the residual sum of squares

$$SS_{res} = \sum_{i=1}^{m} \left(y^i - h(x^i)\right)^2$$

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}}$$

Evaluation of Classification Model

- Confusion matrix
- Recall
- Precision
- Specificity
- Accuracy
- F-score
- Root mean squared error (RMSE)

Evaluation of Classification Model

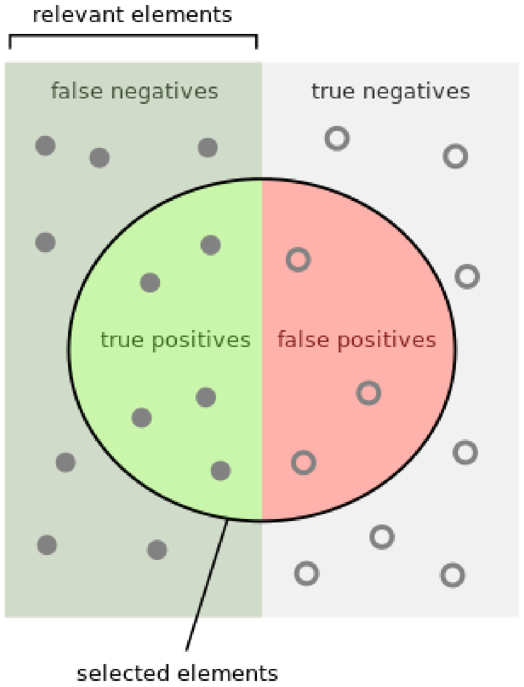
- Confusion matrix: a **confusion matrix**, also known as an error matrix, is a specific table layout that allows visualization of the performance of an algorithm

		Actual class		
		Cat	**Dog**	**Rabbit**
Predicted class	**Cat**	5	2	0
	Dog	3	3	2
	Rabbit	0	1	11

Evaluation of Classification Model

- Precision, Recall, Accuracy and Specificity

		Predicted Label	
		Positive	**Negative**
Known Label	**Positive**	True Positive (TP)	False Negative (FN)
	Negative	False Positive (FP)	True Negative (TN)



Measure	Formula	Intuitive Meaning
Precision	TP / (TP + FP)	The percentage of positive predictions that are correct.
Recall / Sensitivity	TP / (TP + FN)	The percentage of positive labeled instances that were predicted as positive.
Specificity	TN / (TN + FP)	The percentage of negative labeled instances that were predicted as negative.
Accuracy	(TP + TN) / (TP + TN + FP + FN)	The percentage of predictions that are correct.

Evaluation of Classification Model

- F-score:

$$F = 2 * \frac{precision * recall}{precision + recall}$$

References

http://openclassroom.stanford.edu/MainFolder/CoursePage.php?course=MachineLearning

https://en.wikipedia.org/wiki/Activation_function

https://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example/