LING 506 - TOPICS IN COMPUTATIONAL LINGUISTICS

Introductory Machine Learning

Yan Tang

Department of Linguistics, UIUC

Week 14

Last week…

- Parametric vs non-parametric model

- SVM regression
  - Opposite objective to SVM classification
- Decision Tree regression
  - Prone to overfitting
- Gaussian Process regression
  - Allows standard deviation and confidence intervals

Last week…

- Stepwise linear regression
  - Regresses multiple variables while removing less important features

- Data preparation for linear regression
  - Encode categorical data
  - Keep in mind the linear assumption
  - Remove outliers
  - Remove collinearity
  - Rescale features

Intro to artificial neural network

- The rudimentary logic behind artificial networks (ANN)
  - Use inspiration from brain's architecture to build an intelligent machine
  - An ANN is a ML model inspired by the networks of biological neuron in the brain
- ANNs are at the very heart of Deep learning
  - Versatile, powerful and scalable
  - e.g. Google Images, Apple Siri, DeepMind's Alpha Go

Development of ANN

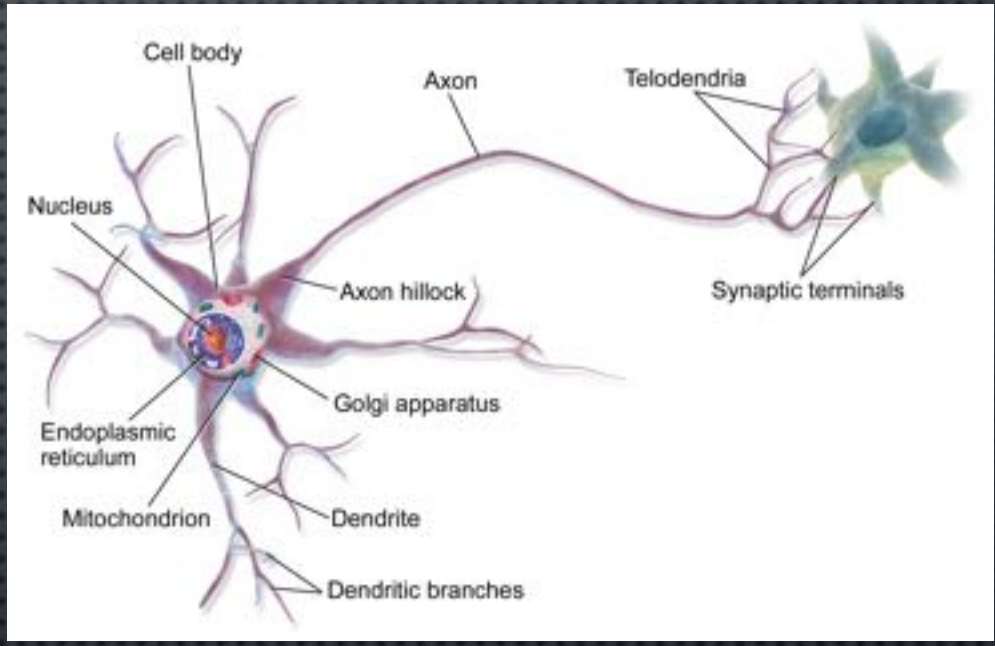
- 1943: The first neural network architecture was introduced
  - Warren McCulloch (neurophysiologist) and Walter Pitts (mathematician)
  - A computational model on the mechanism of biological neurons working together in the brain to perform complex tasks using propositional logic
- 1980s: new architectures were invented; improved training method were developed
- 1990s: SMV was invented
  - Better results and stronger theoretical foundations

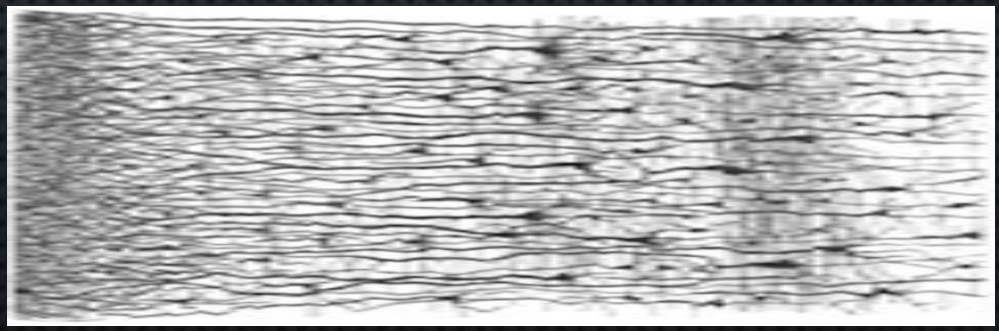
Development of ANN

- Now:
  - Massive amount of available data for training; better performance on large and complex problems
  - Progress in computing hardware makes fast and complex training possible
  - Improved training algorithms
  - Tremendous investment on development and advancing

The inspirations from biological neurons

- Cell body: nucleus and other important components
  - Processes signal, makes decisions, gives order
- Axon
  - The bond between one neuron and others
  - Signal passage
- Synaptic terminals
  - The (neuro)signal transmitter between neurons

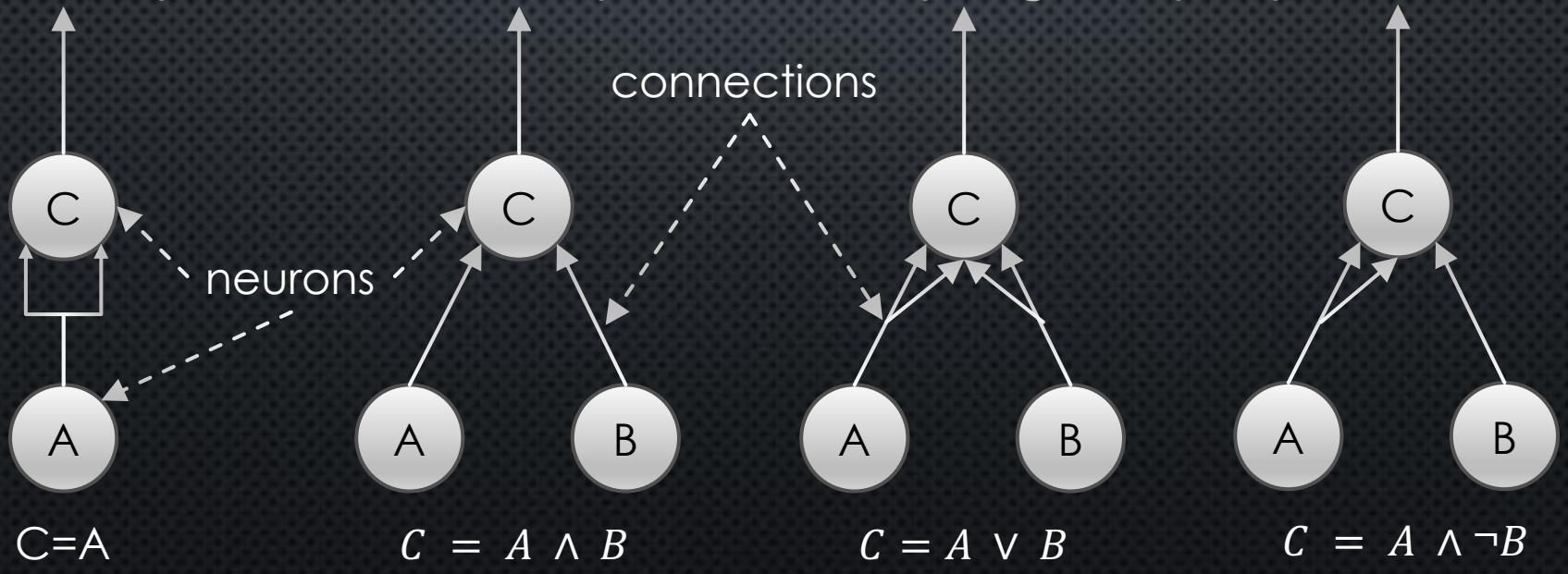




https://en.wikipedia.org/wiki/Neuron
https://en.wikipedia.org/wiki/Cerebral_cortex

Computational logics in AN

- An artificial neuron (AN) has ***one*** or ***more*** binary inputs, and one output
- An AN only activates its output when its ***active*** inputs more than a certain number
- A simple model can perform any logical proposition



The perceptron

- The perceptron (Rosenblatt, 1957)
  - One of the simplest ANN architectures
  - Based on threshold logic unit (TLU)

- Inputs and outputs are numbers
  - Inputs are weighted
  - Output is a sum of the weighed inputs, further applied to a *step function* – the activation function

The perceptron : TLU

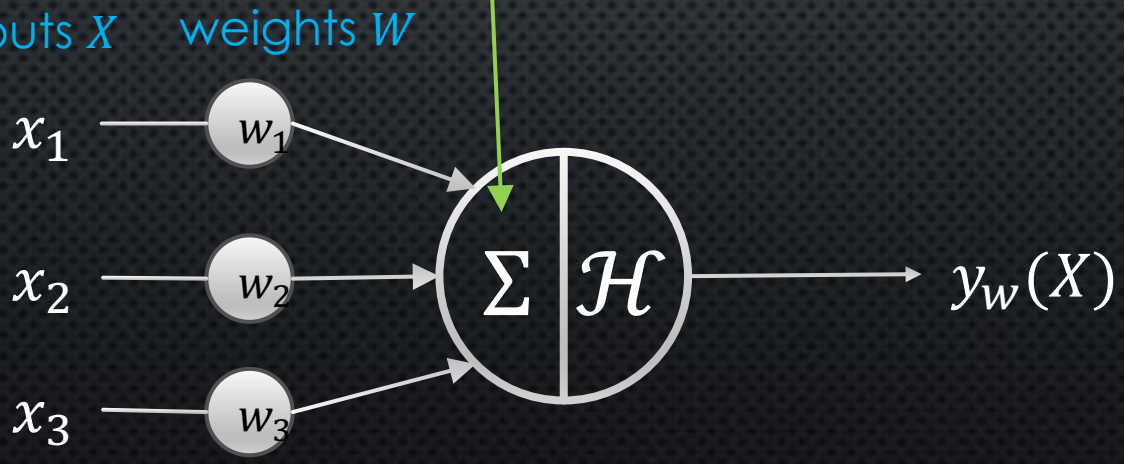
- Heaviside step function:

$$\mathcal{H}(z) = \begin{cases} 0 \; if \; z < 0 \\ 1 \; if \; z \geq 0 \end{cases}$$

$$z = w_1 x_1 + w_1 x_1 + \ldots + w_n x_n = X^T \cdot W$$

X: input feature vector

W: the weight vector associated with features in X

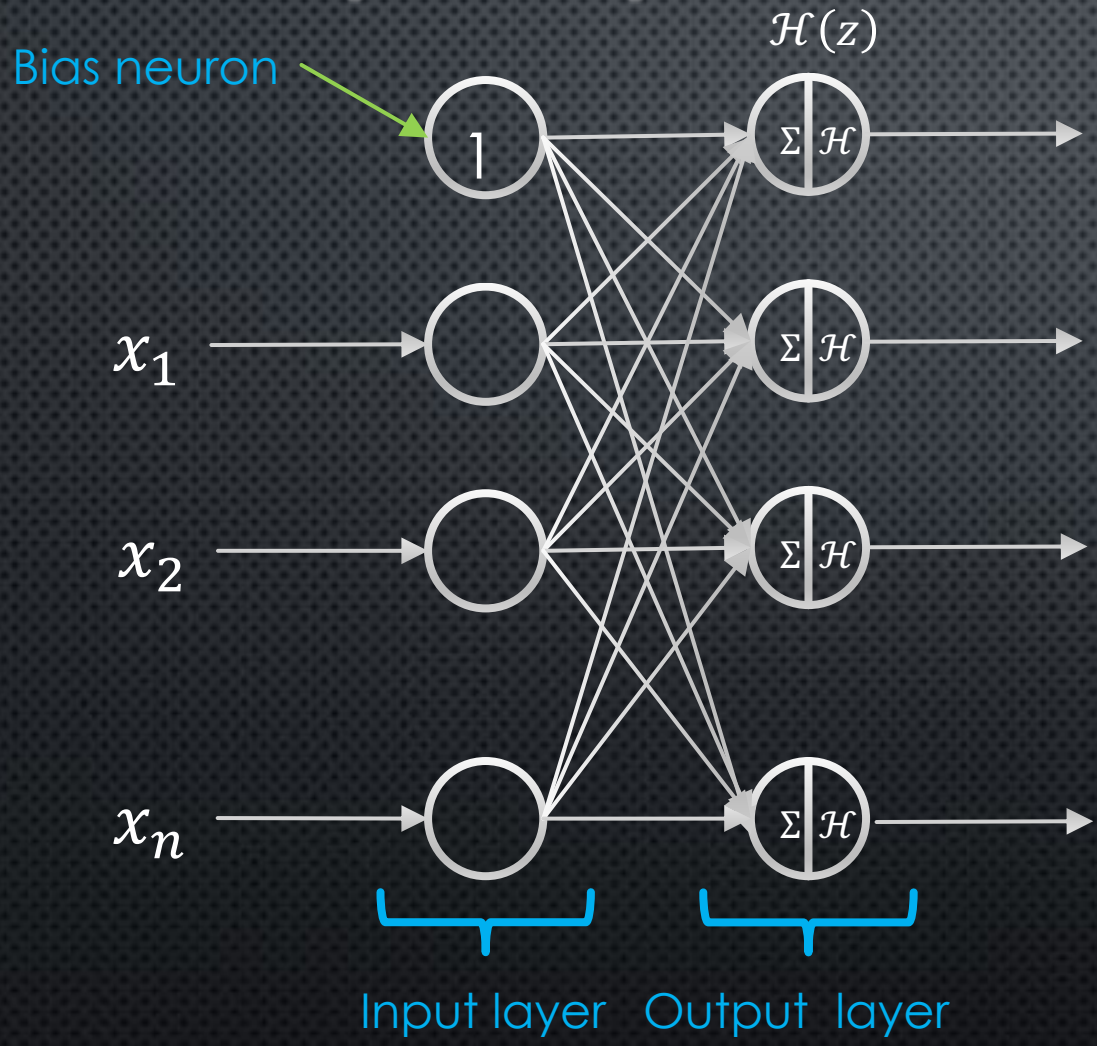


TLU illustration

The perceptron: structure

- The structure of perceptron
  - Consists of a *single* layer of TLUs
  - Each TLU connects all the inputs
- Input layer:
  - Each input is passed to an input neuron
  - Input neurons passthrough all inputs without processing
  - Bias neuron: always outputs 1
- Fully connected layer (dense layer)
  - All neurons (TLU) in a layer connected to every neuron in the previous layer

The perceptron: structure



$$y_w(X) = \emptyset(XW + b)$$

$\boldsymbol{b}$: bias vector – all the connection weights between the bias neuro and the ANs

$\emptyset$: the *activation function*. $\mathcal{H}(z)$ is for TLUs

The perceptron: training

- Perceptron learning rule: it increases the weights of connections from the inputs that contribute to the correct prediction:

$$w'_{i,j} = w_{i,j} + \eta(y - \hat{y}_j)x_i$$

$w_{i,j}$: the connection weight between the i^{th} input neuron and the j^{th} output neuron

x_i : the i^{th} input value of the current training instance

y: the groundtruth for the current training instance

$\hat{y}_j$: the output of the j^{th} output neuron

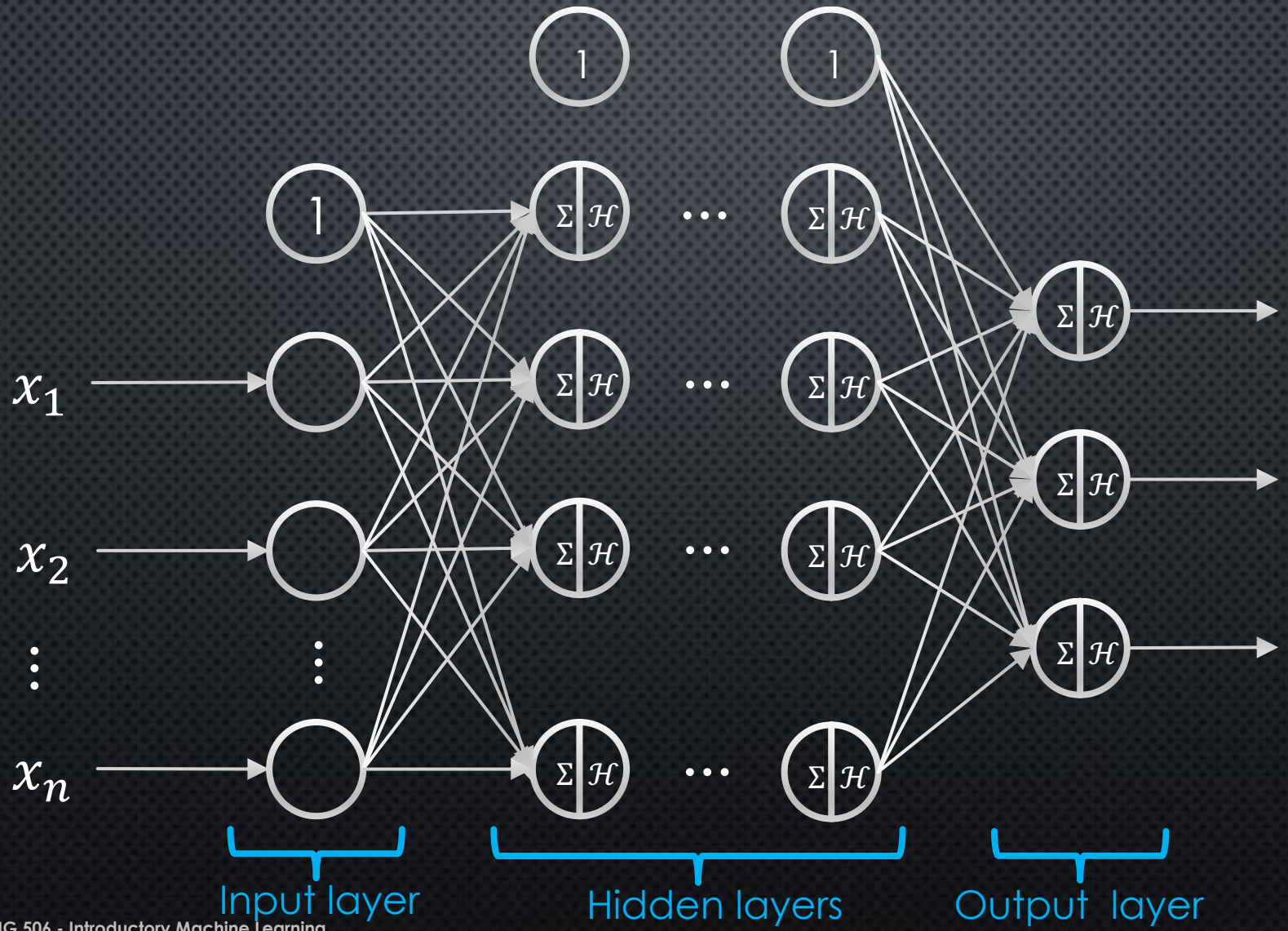
η: the learning rate

- Perceptron convergence theorem:
  - Perceptron would converge to a solution if the training samples are linearly separable

The perceptron: limitation

- For classification problem, perceptrons do not produce a class probability; predictions are made based on a hard threshold (e.g. 0)
- They cannot solve trivial problems, such as XOR
- However:
  - Such the "XOR" issue can be solved by concatenating or stacking perceptrons – Multilayer Perceptron (MLP)

The Multilayer Perceptron



Feedforward neural network and deep neural network

- Feedforward Neural Network (FNN)
  - An arrangement of interconnected neurons that map inputs to response
  - Data is passed in one direction
  - Model complexity: number of hidden layers and the number of neurons in a hidden layer
- Deep neural network (DNN)
  - An ANN contains a big number of hidden layers

Backpropagation

- Issue of MLP: how to efficiently adjust W by learning from the model errors to improve the model accuracy??

- Backpropagation:
  - An algorithm to compute the gradients of the network's error for given model parameters
  - Reverse-mode automatic differentiation
  - Enables regular Gradient Descent for MLP

Backpropagation - principles

Two passes through the network:

- **The *forward* pass**: Sends a small set of samples through all the layers; record all the intermediate results from each layer for further use
- Computes the loss of the network
- ***The reverse pass:*** uses the chain rule to calculate how much output connection from the previous layer contributed to the error, going backwards until the input layer
- Performs Gradient Descent to adjust W for entire network, using the error gradients computed during the reverse pass

Activation functions

- An important change to original activation (i.e. the step) function in TLU for backpropagation
  - The logistic (sigmoid) function
  - Gradient cannot be calculated on the step function

- The true purpose of activation function - nonlinear transformation
  - Multiple linear functions in cascade is still linear
  - i.e. even a deep network with linear activation is equivalent to a shallow network
  - Various nonlinearities is the power of ANN

Activation functions

- The sigmoid (logistic) function:

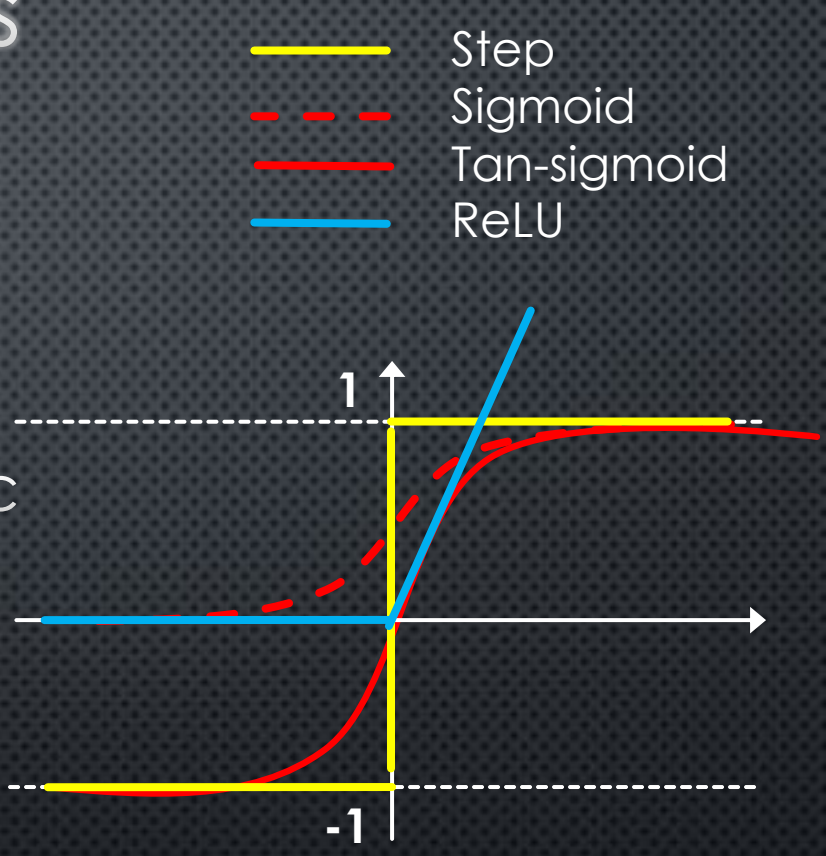
$$sig(z) = \frac{1}{1 + \exp(-z)}$$

- The tangent sigmoid (hyperbolic tangent) function:

$$tansig(z) = \frac{2}{(1 + \exp(-2z)) - 1}$$

- The Rectified Linear Unit function:

$$ReLU(z) = \max(0, z)$$



Four common activation functions

Regression MLPs

Hyperparameter	Typical value
No. of input neurons	1 / input feature
No. of hidden layers	Problem specific, typically 1 to 5
No. of neurons/hidden layer	Problem specific, typically 10 to 100
No. of output neurons	1 / prediction dimension
Activation Fn. for hidden layer	ReLU or SELU
Activation Fn. for output layer	None, or ReLU(+outputs) or sig/tansig (bounded outputs)
Loss function	MSE or MAE (if outliers)

Chapter 10, *Aurélien Géron, 2019*

Classification MLPs

Hyperparameter	Binary classfi.	Multiclass classfi.
No. of input neurons	1 / input feature	1 / input feature
No. of hidden layers	Problem specific, typically 1 to 5	Problem specific, typically 1 to 5
No. of neurons/hidden layer	Problem specific, typically 10 to 100	Problem specific, typically 10 to 100
No. of output neruons	1	1 / class
Act. Fn. for hidden layer	ReLU or SELU	ReLU or SELU
Activation Fn. for output layer	sig	softmax
Loss function	Cross entropy	Cross entropy

Chapter 10, *Aurélien Géron, 2019*