MA5832: Data Mining & Machine Learning

Collaborate Week 5: Neural Networks

Martha Cooper, PhD

JCU Masters of Data Science

2021-09-28

Housekeeping

- Collaborates = Thursdays 6-7:30pm

For my Collaborate Sessions, you can get the slides & R code for each week on Github:

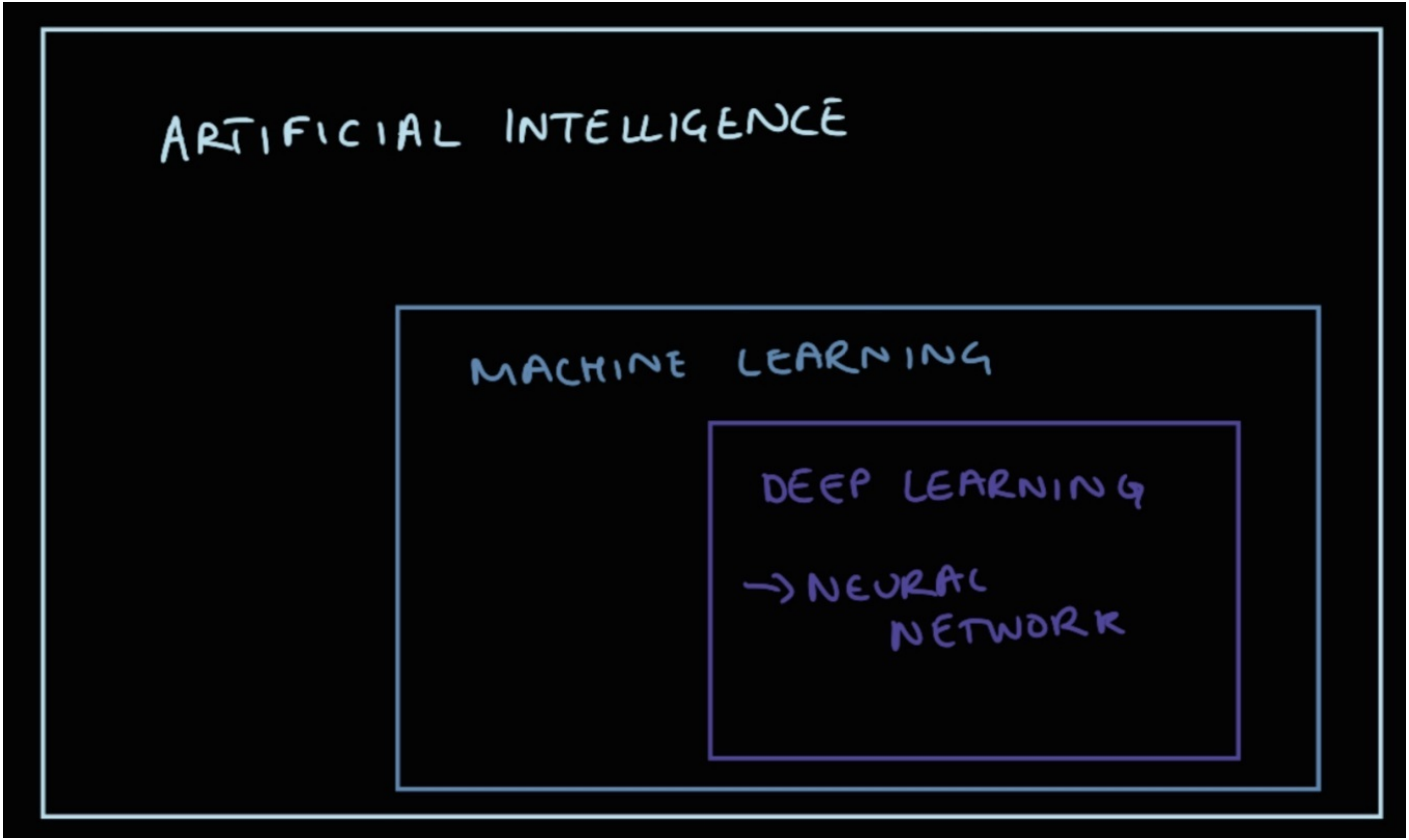
https://github.com/MarthaCooper/MA8532

Today's Goals

- Structure of Neural Network (NN)
- Estimation of NN
- Training a neural network using Keras

Neural Networks

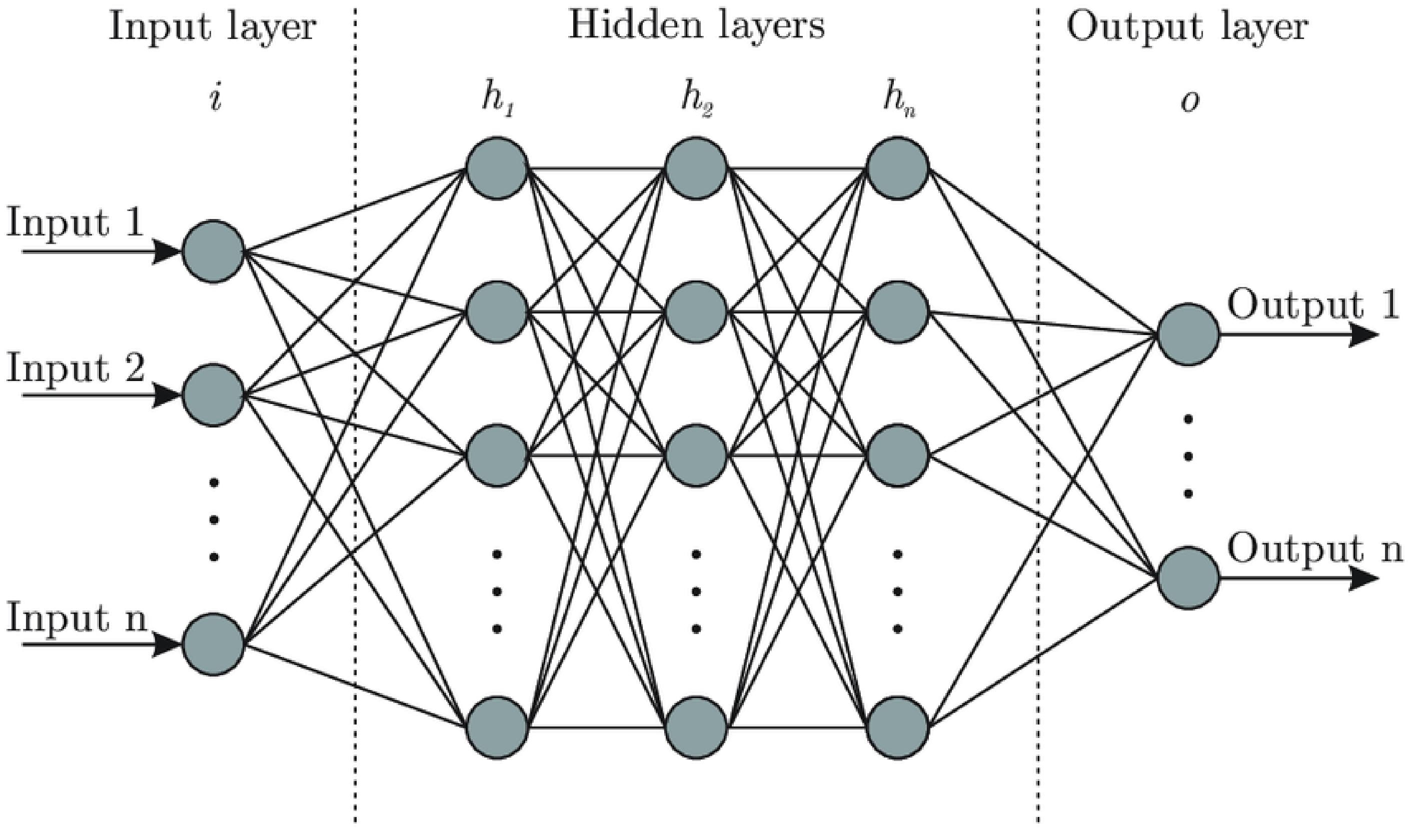
Deep Learning and Neural Networks



Neural Networks

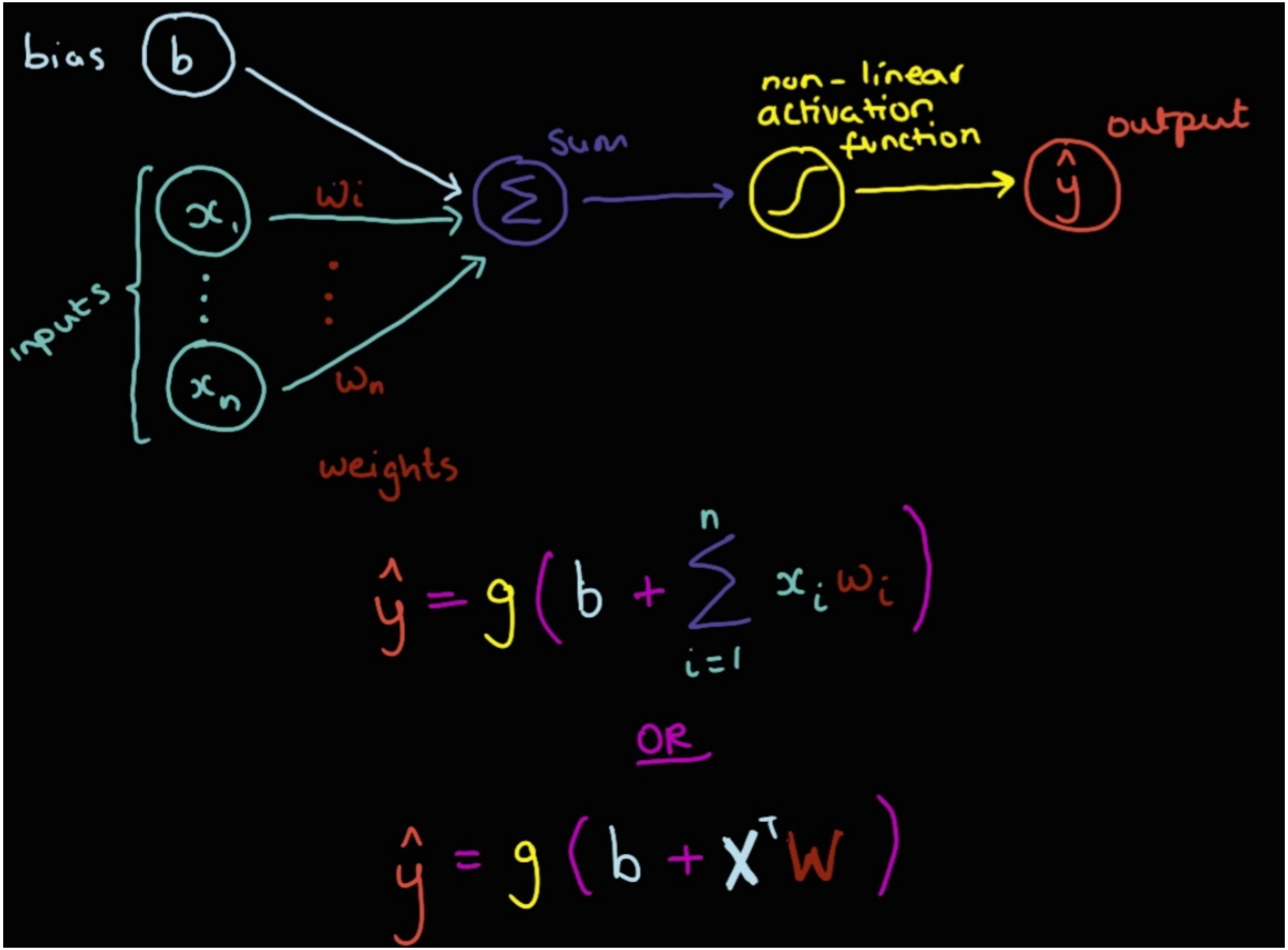
- Recognise and predict relationships in data using algorithms that are inspired by the way the human brain works.
- Many different types of NN:
  - Vanilla Neural Network
  - Recurrent Neural Network *e.g.* Long short term memory network
    Natural language processing, time series
  - Convolution Neural Network - *Computer vision*

Topology of a (vanilla) Neural Network

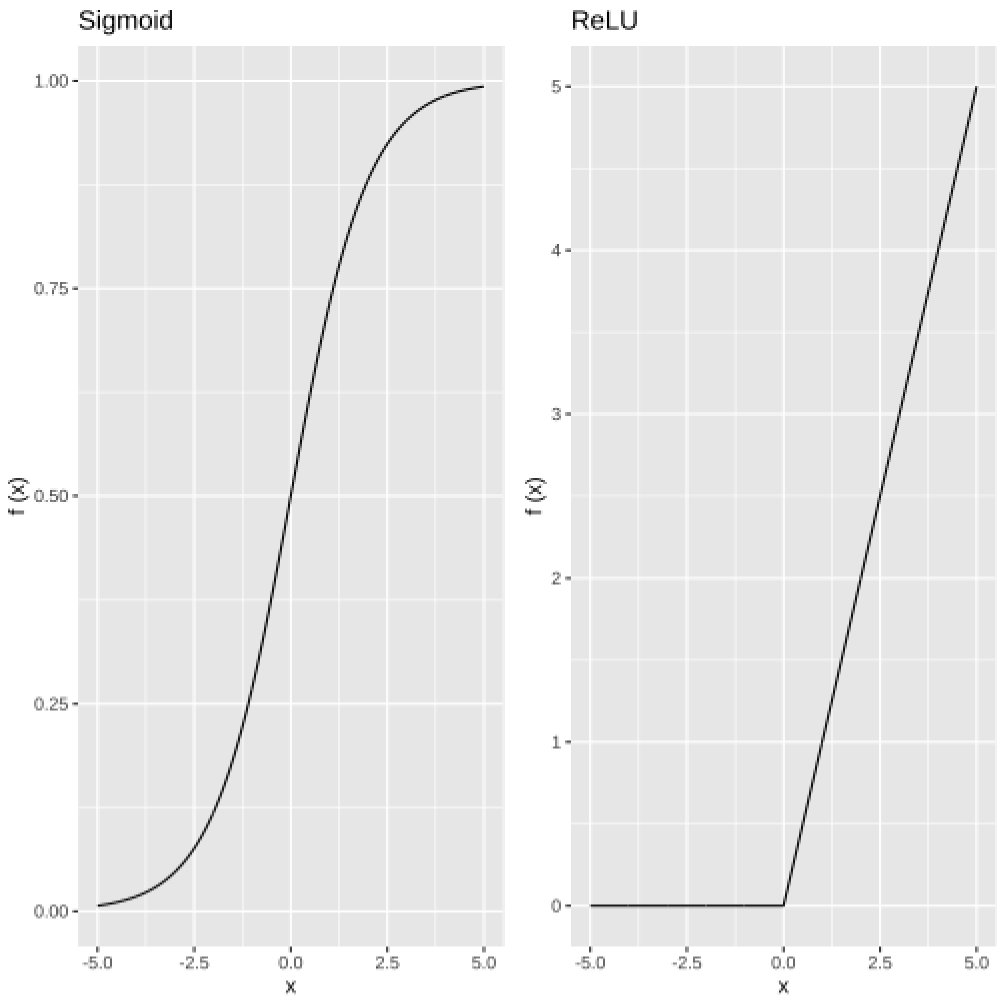


kdnuggets

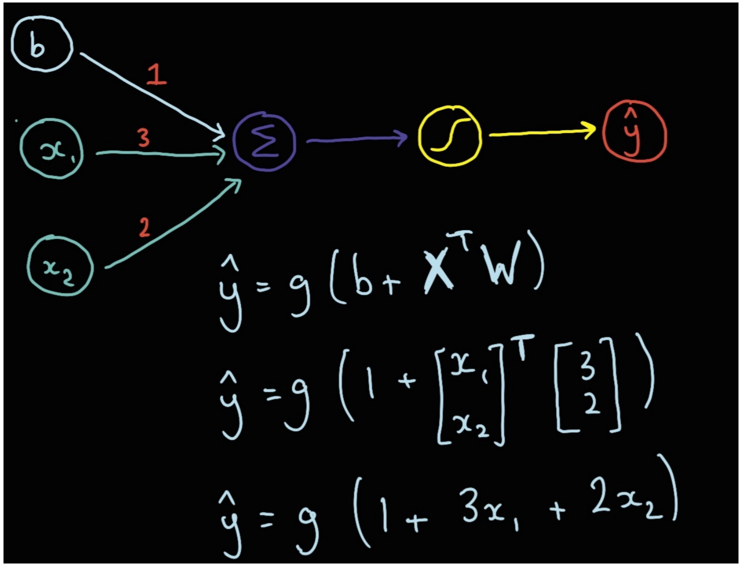
A single NN unit - the perceptron

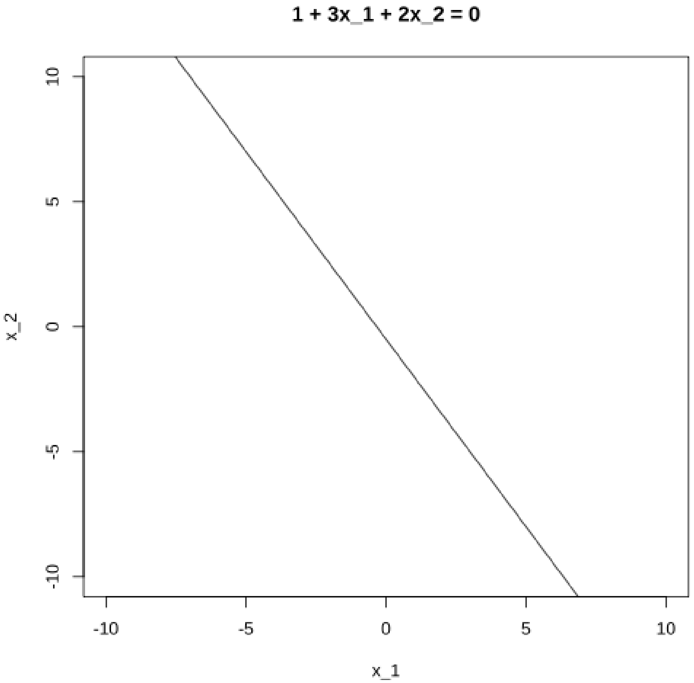


Activation Functions

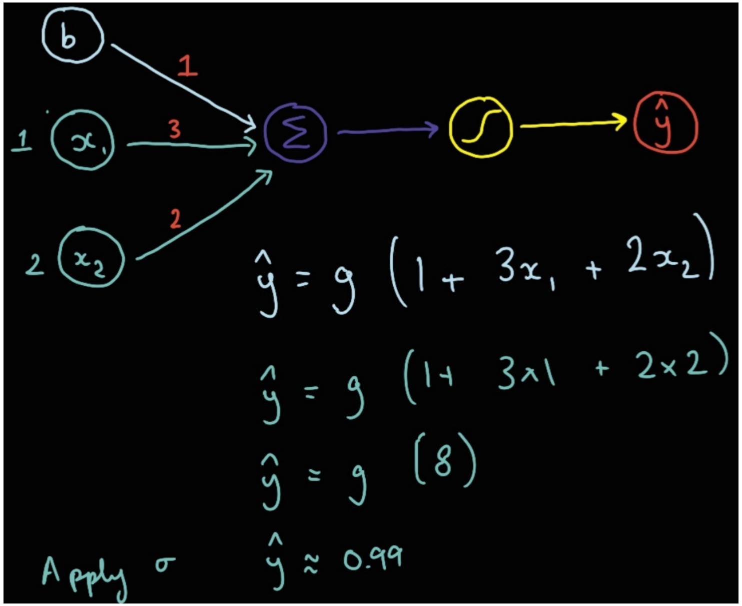


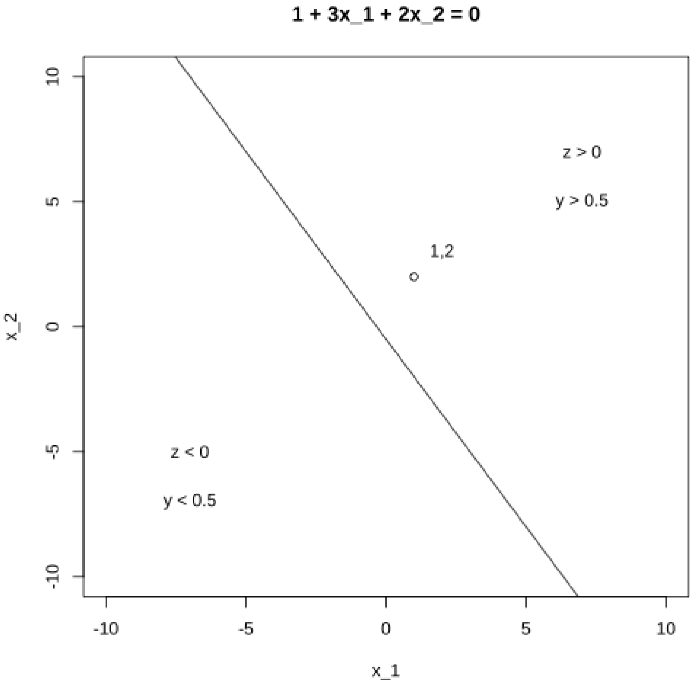
Classification using a perceptron



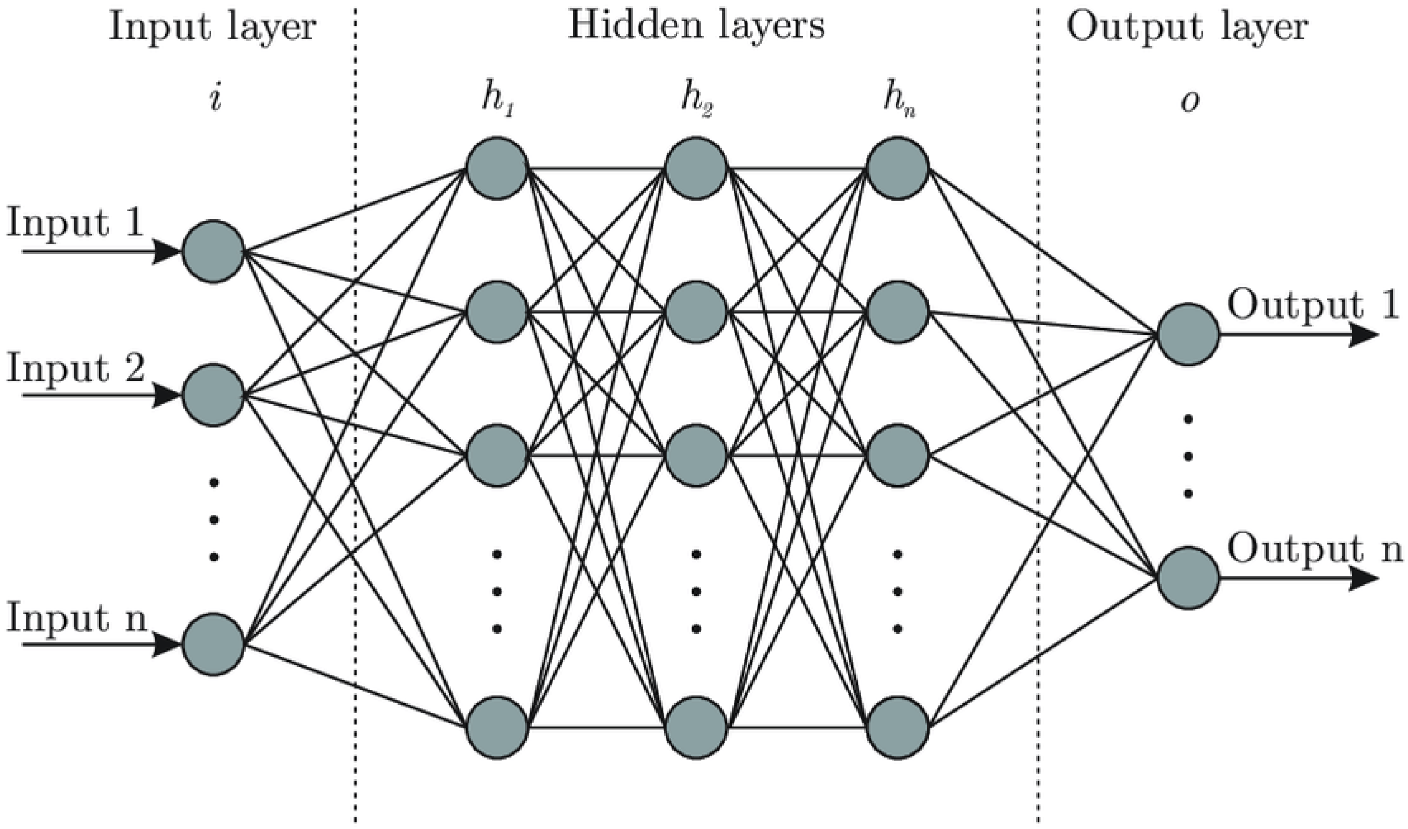


Classification using a perceptron





Building Neural Networks from Perceptrons



kdnuggets

Training Neural Networks

What do we need to estimate to train a neural network?

How do we estimate those parameters?

Training Neural Networks

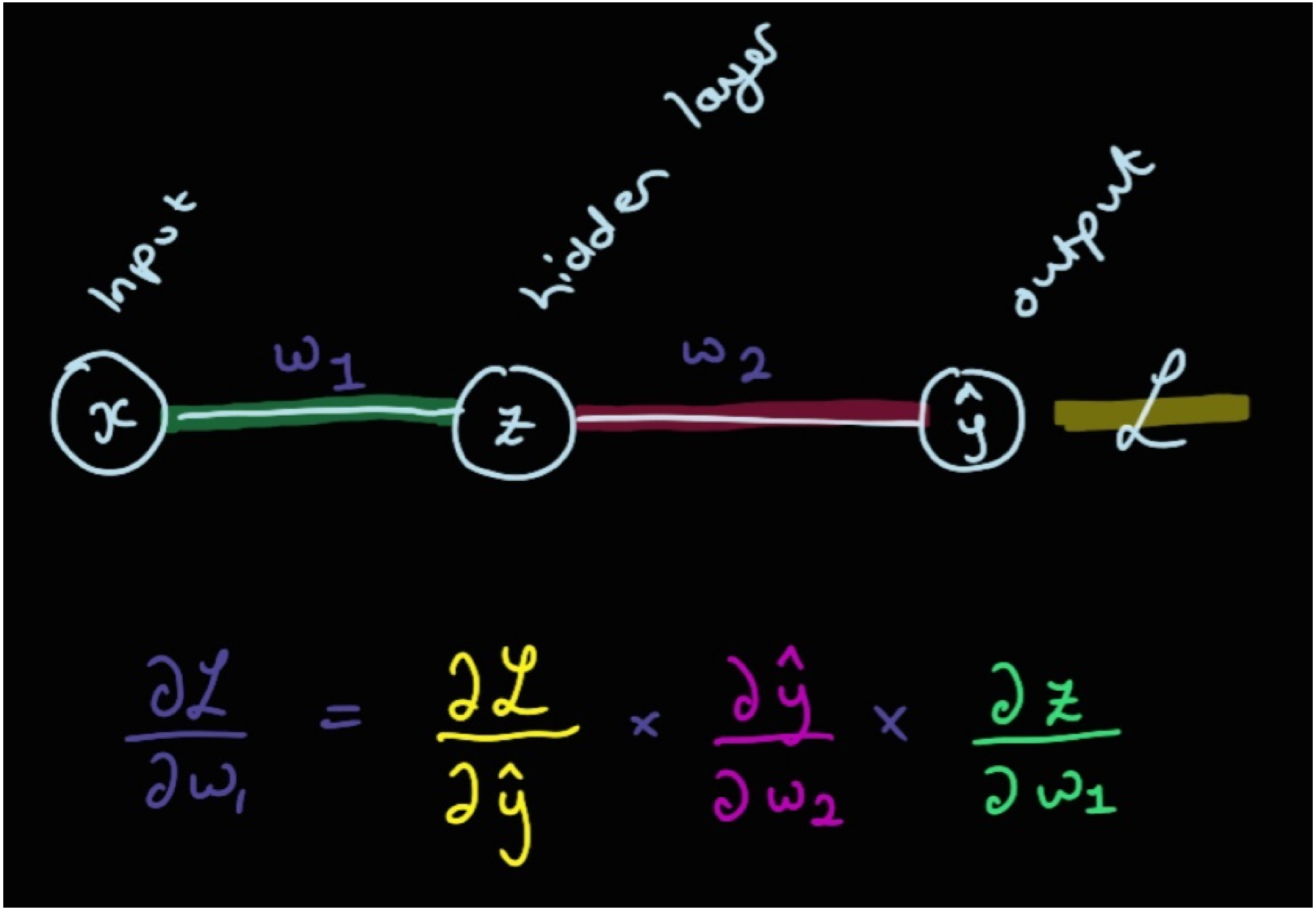
What do we need to estimate to train a neural network?

- All weights and biases for all neurons in the network

How do we estimate those parameters?

1. Loss function to quantify error *e.g.* Regression = MSE = $\mathcal{L} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$
2. Solve using optimisation *e.g.* Stochastic gradient descent

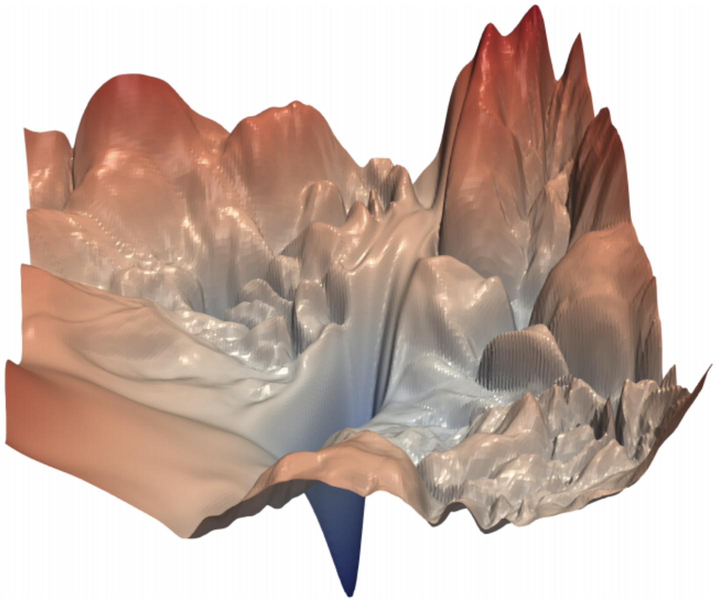
Calculating Gradients using Backpropogation



Optimisation methods

- Stocastic gradient descent
- Adam
- AdaDelta

.... and more ...



Li et al., 2018

Questions

- Which activation function?
- How many neurons should I choose in a hidden layer?
- How many hidden layers?
- Which loss function?

Which activation function?

Sigmoid

Advantages

- Output values bound between 0 and 1, normalizing the output of each neuron
- Clear predictions

Disadvantages

- Vanishing gradient
- Computationally expensive

ReLU

Advantages

- Computationally efficient (Sparsity)
- No vanishing gradient

Disadvantages

- The Dying ReLU problem: when inputs approach zero, or are negative, the output becomes zero. The weight and bias cannot be updated

How many neurons should I choose in a hidden layer?

- Trials and error
- Cross-validation approach
- Rule of thumb:
  - in the range between the number of input and output ;
  - The number of hidden neurons should be less than twice the number of inputs;
  - the number of hidden neurons should be 2/3 of the number of inputs, plus the number of outputs.
- setting nodes equal to $ns/(c*(n_i+n_o))$ where n_i is the number of inputs, n_o is the number of outputs. n_s : the number observations of training sample. c is between 2 and 10.

How many hidden layers?

- A single-layer neural network can only be used to represent linearly separable functions. It can be used for a simple issues such as classifying two classes where they can neatly separated by a line.
- A multi-layer NN can be used to represent with more complex issues, and high-dimensional space.

Which Loss function?

Problem Type	Output Type	Final Activation Function	Loss Function
Regression	Numerical value	Linear	Mean Squared Error (MSE)
Classification	Binary outcome	Sigmoid	Binary Cross Entropy
Classification	Single label, multiple classes	Softmax	Cross Entropy
Classification	Multiple labels, multiple classes	Sigmoid	Binary Cross Entropy

Towards Data Science

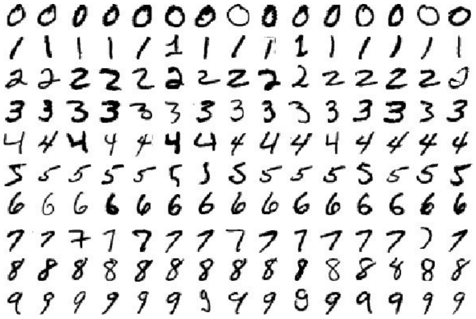
NN in R

Possible packages:

- neuralnet
- Keras
- Tensorflow
- Pytorch

Comparison of deep learning frameworks

NN in R



MNIST database

- We use the MNIST dataset which contains 60,000 training images and 10,000 test images.
- The goal is to classify greyscale images of handwritten digits (28 x 28 pixels) into their 10 categories (0 through to 9)

NN in R

See Rmarkdown as code doesn't fit on slide...

```
library(keras)
mnist <- dataset_mnist()
head(mnist)

train_images <- mnist$train$x
train_labels <- mnist$train$y
test_images <- mnist$test$x
test_labels <- mnist$test$y

# Coverting 28x28 pixel into Tensorflow format
train_images <- array_reshape(train_images, c(60000, 28 * 28))
train_images <- train_images / 255 #normalise input to between 0 & 1

test_images <- array_reshape(test_images, c(10000, 28 * 28))
test_images <- test_images / 255 #normalise input to between 0 & 1


# Encode y variable into 0 and 1
train_y_mtx <- to_categorical(train_labels)
test_y_mtx <- to_categorical(test_labels)

# NN structure
```

Homework

- Install Keras and Miniconda (especially if you have windows!)

References & Extra reading

- Deep Learning

Extra watching

- MIT 6.S191: Introduction to Deep Learning
- 3b1b Deep Learning