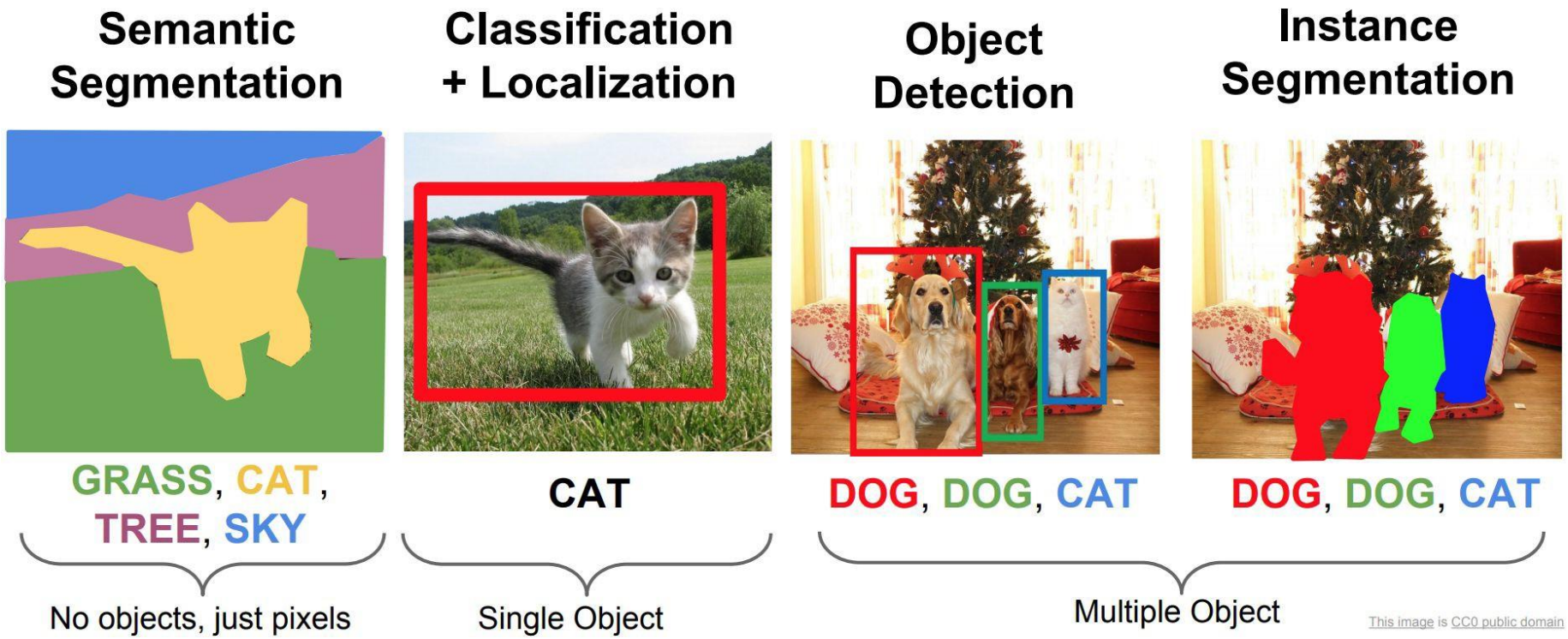
Neural Networks

Ata Kaban

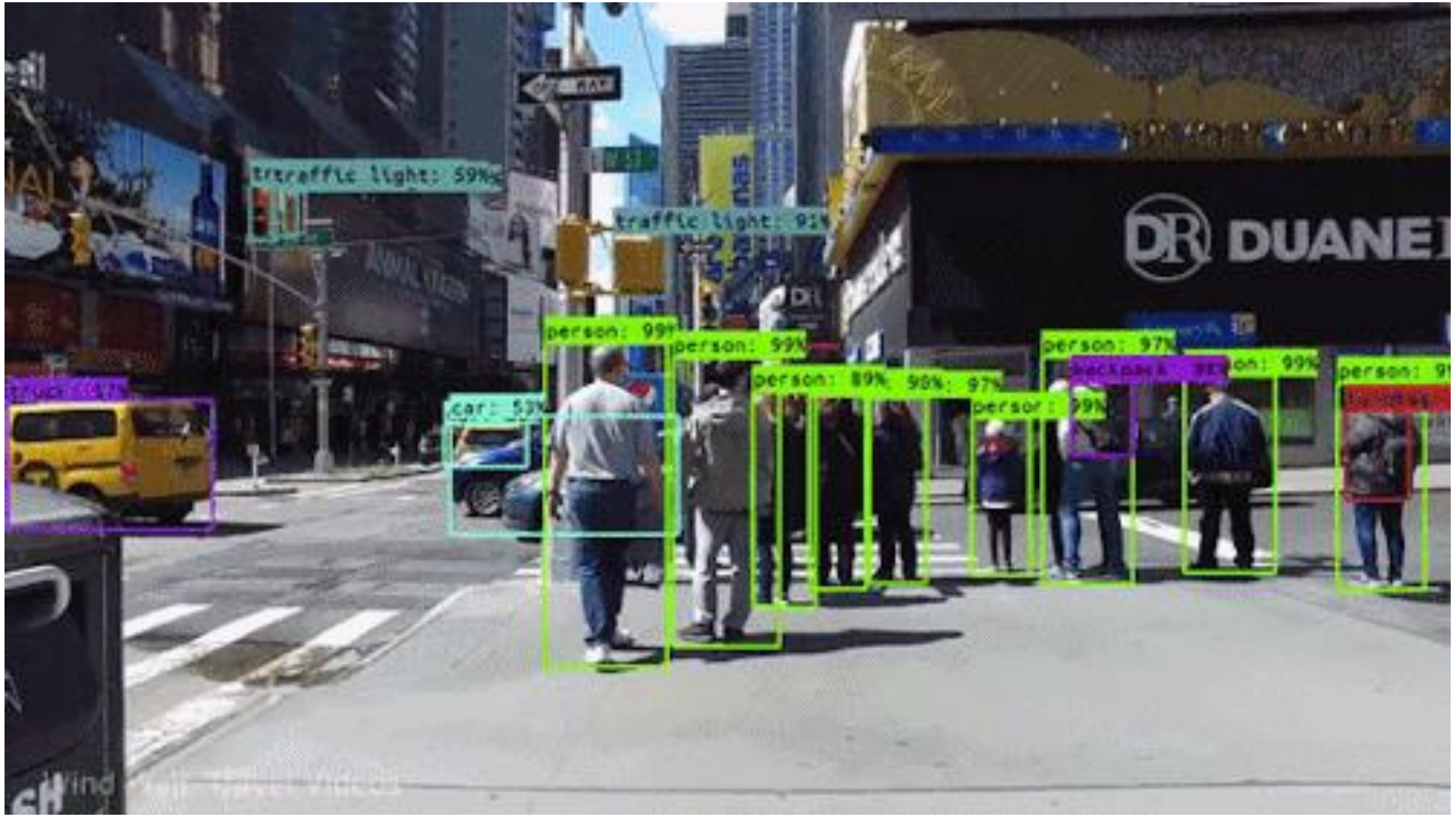
What are neural networks?

- Highly nonlinear models having many free parameters
- Can be used for either regression and classification depending on the choice of loss function.
- Can replace nonlinear regression and nonlinear logistic regression which are less practical

Motivation



Source: https://towardsdatascience.com/object-detection-using-deep-learning-approaches-an-end-to-end-theoretical-perspective-4ca27eee8a9a



https://towardsdatascience.com/object-detection-using-deep-learning-approaches-an-end-to-end-theoretical-perspective-4ca27eee8a9a

Classification

255, 255, 255	255, 255, 255	255, 255, 255	255, 255, 255	255, 255, 255
255, 255, 255	255, 255, 255	255, 255, 255	255, 255, 255	255, 255, 255
255, 255, 255	255, 255, 255	255, 255, 255	255, 255, 255	255, 255, 255

Image

Computer Input

255 255 255 for *each* pixel

Source: https://towardsdatascience.com/object-detection-using-deep-learning-approaches-an-end-to-end-theoretical-perspective-4ca27eee8a9a

Neural networks - outline

Using neural networks requires the same conceptual journey as before:

1) Model formulation
2) Cost function
3) Learning algorithm by gradient descent

Neural networks - outline

Using neural networks requires the same conceptual journey as before:

1) Model formulation
2) Cost function -- nothing new here
   - for regression: Mean square error between predictions and observed targets
   - for classification: Logistic loss (also called cross-entropy)
3) Learning algorithm by gradient descent

Neural networks - outline

Using neural networks requires the same conceptual journey as before:

1) Model formulation
2) Cost function
3) Learning algorithm by gradient descent
   - The update rules are non-trivial, because the models are much more complex
   - It is performed by an algorithm called “Backpropagation”
   - Conceptually, each iteration of Backprop takes a gradient descent step
   - Implementations exist that are able to compute the grandient automatically

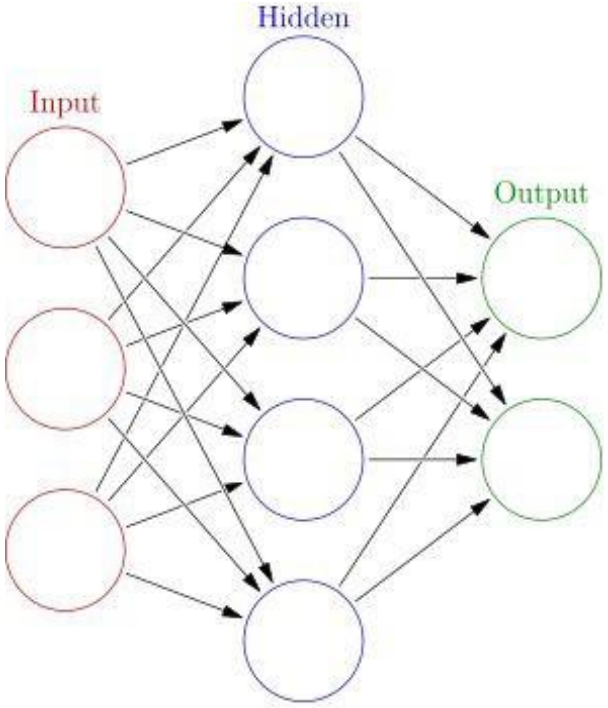
Neural networks - outline

Using neural networks requires the same conceptual journey as before:

1) Model formulation
   - Sometimes called “architecture”
   - Designing this for the problem at hand is the main challenge
2) Cost function
3) Learning algorithm by gradient descent

Neural network model

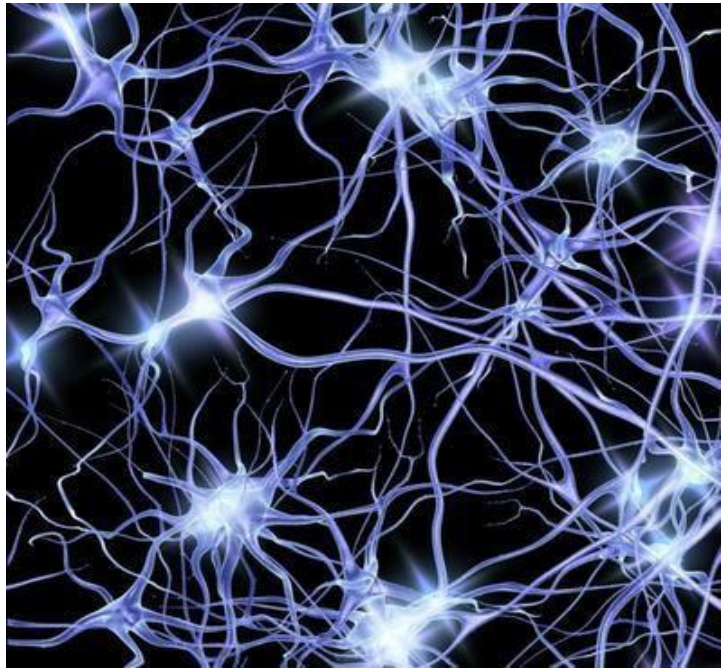
- Roughly, we can think of a neural network model as being composed of several logistic regression units

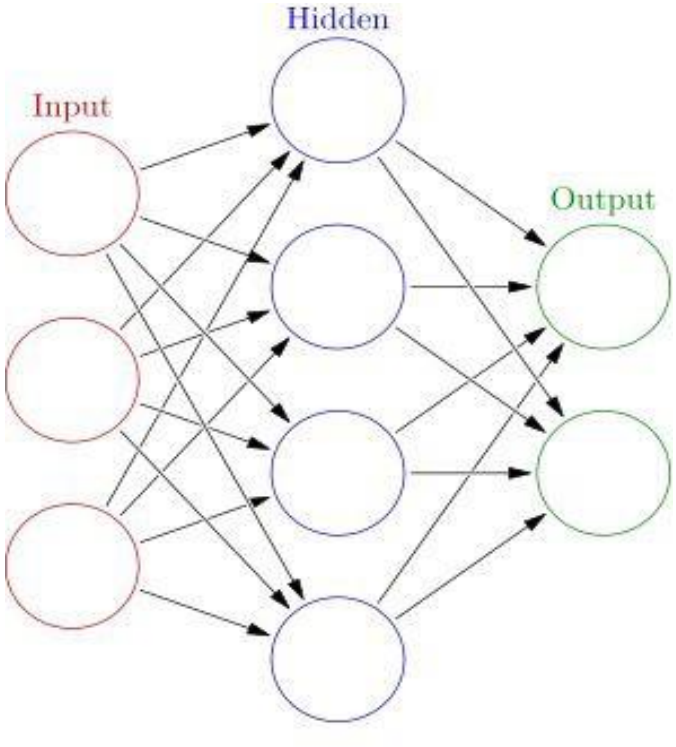


Source: wikimedia.org

Neural networks – the name

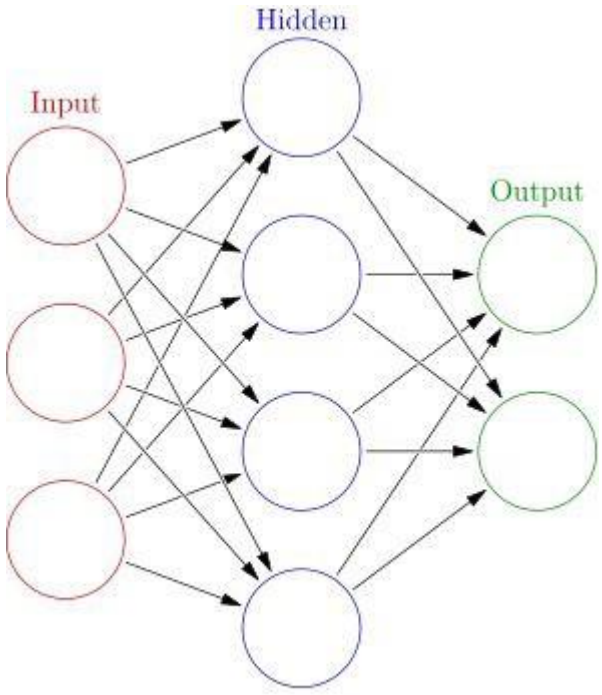
Inspiration from networks of neurons in the brain – each unit is analogous to a neuron.





Source: wikimedia.org, www.reference.com

Building blocks of a feedforward neural net

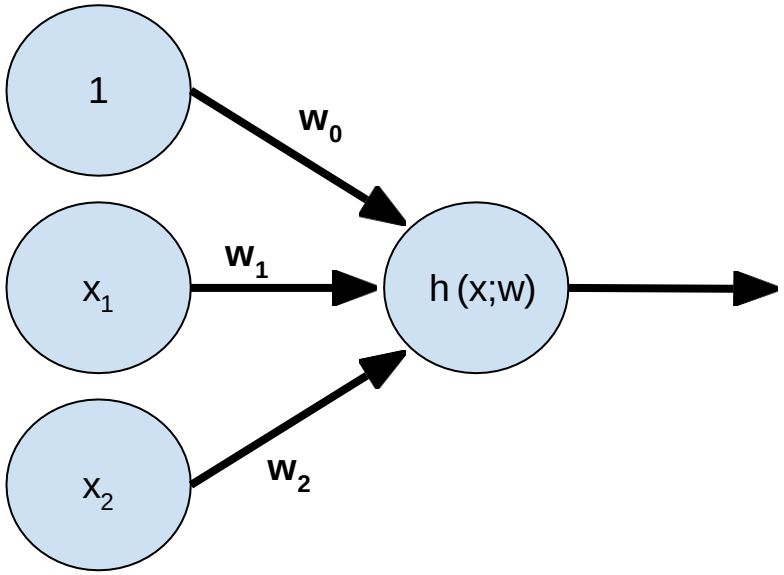


- Each node is one unit or neuron
- Each arrow is a connection with a weight
- Nodes are arranged in layers
  - One input layer
  - One output layer
  - Any number of hidden layers (0,1,2,…)
- Hidden & output nodes typically apply a sigmoid, or other activation function

Simplest neural net

- A neural net with 0 hidden layers is called a perceptron
- If the activation function is the sigmoid, then this model is equivalent to a logistic regression

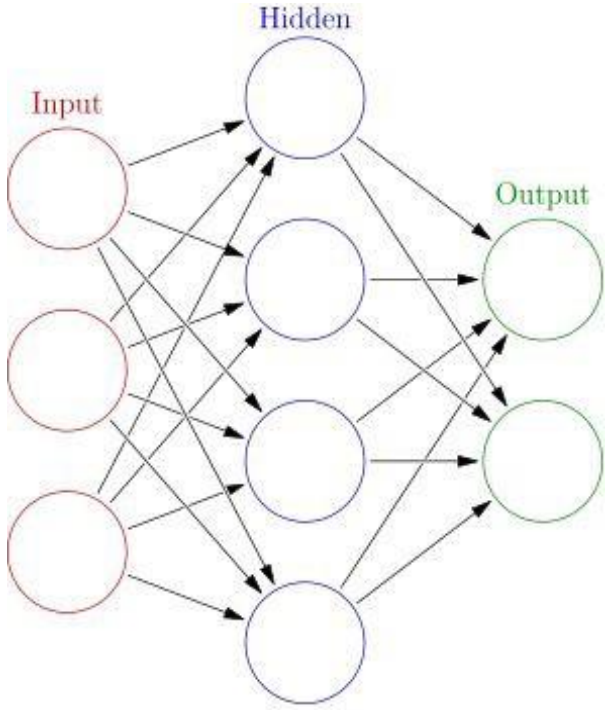
$$h(x, w) = \sigma(w_0 + w_1 x_1 + w_2 x_2)$$



- The type of computation performed by each non-input node is the same in multi-layer networks too.
- The choice of activation function can be different

Multi-layer perceptron

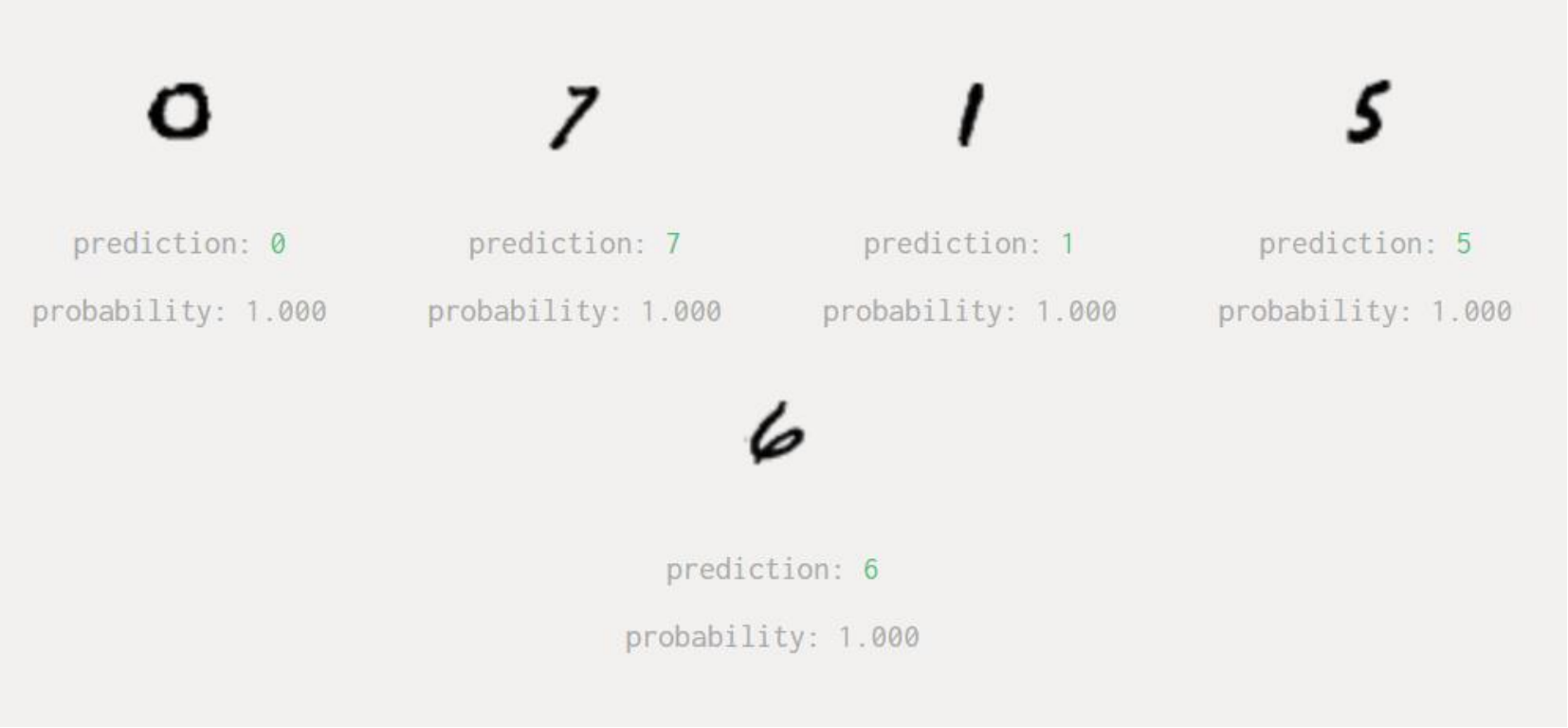
- When we have one hidden layer, the model is called multi-layer perceptron
- It is a truly non-linear model.
- How many free parameters does it have?
  - Weights = parameters
  - Number of hidden units, choice of activation function = hyperparameters
- Number of output nodes = number of targets or labels we want to predict



In theory, with sufficient number of hidden units this type of net is able to approximate any function (cf. Cybenko 1989)

Handwriting recognition

- MLP is well suited to learning from sensor data
- MLP can achieve approximately 98% accuracy in handwriting recognition on the MNIST data set

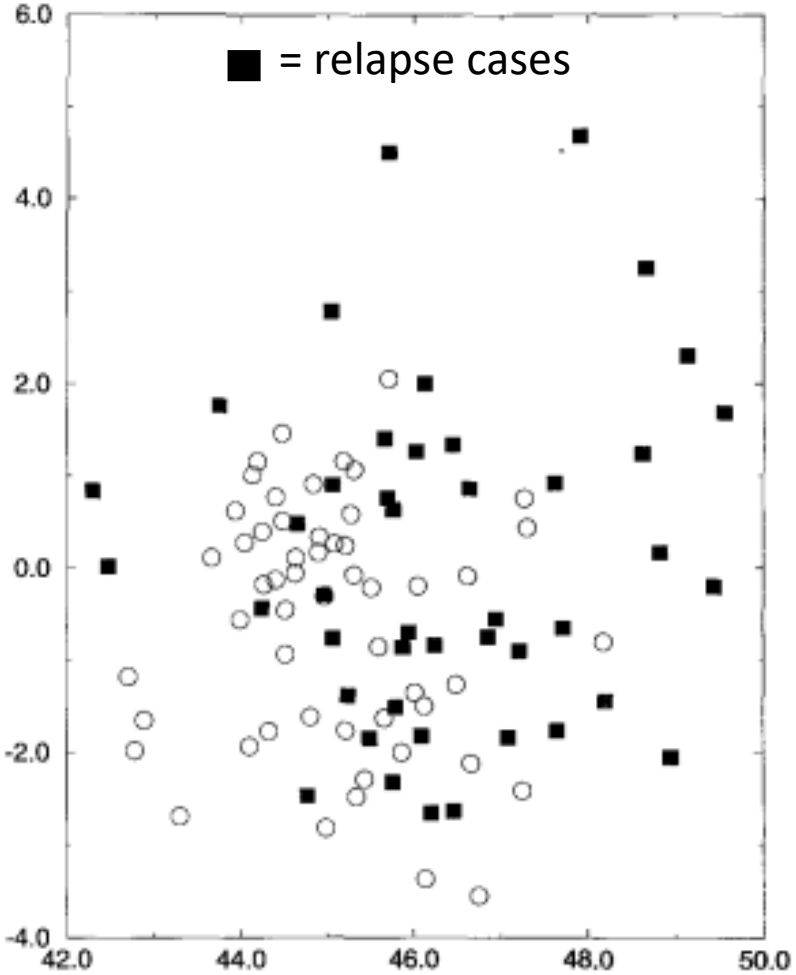


http://scienceai.github.io/neocortex/mnist_mlp/

Multilayer perceptron for clinical data

Neural network prediction of relapse in breast cancer patients (paper by Tarassenko et al., 1996)

- **Target:** relapse/no within 3 years
- **Attributes:** Age, Tumour size, nr.Nodes, Log er, Log egfr
- **Data:** 350 patients
- **Architecture:** 1 hidden layer
- **Results:** 72% classification accuracy

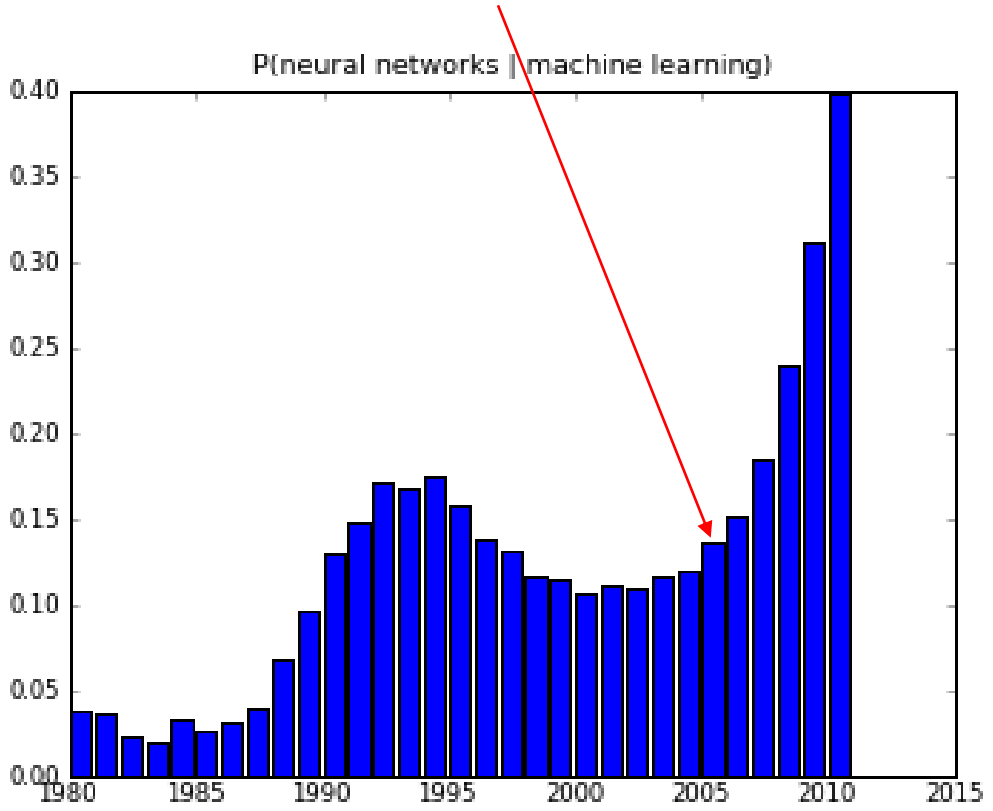


https://link.springer.com/content/pdf/10.1007/BF01413746.pdf

Deep neural networks – Deep Learning

- Proportion of machine learning papers that contain the term ‘neural networks’
- Popular in the 90s
- Dip in the 2000s
- The problem was that training fully connected neural nets is difficult
- Re-emergence of the topic of neural networks with the advent of deep learning

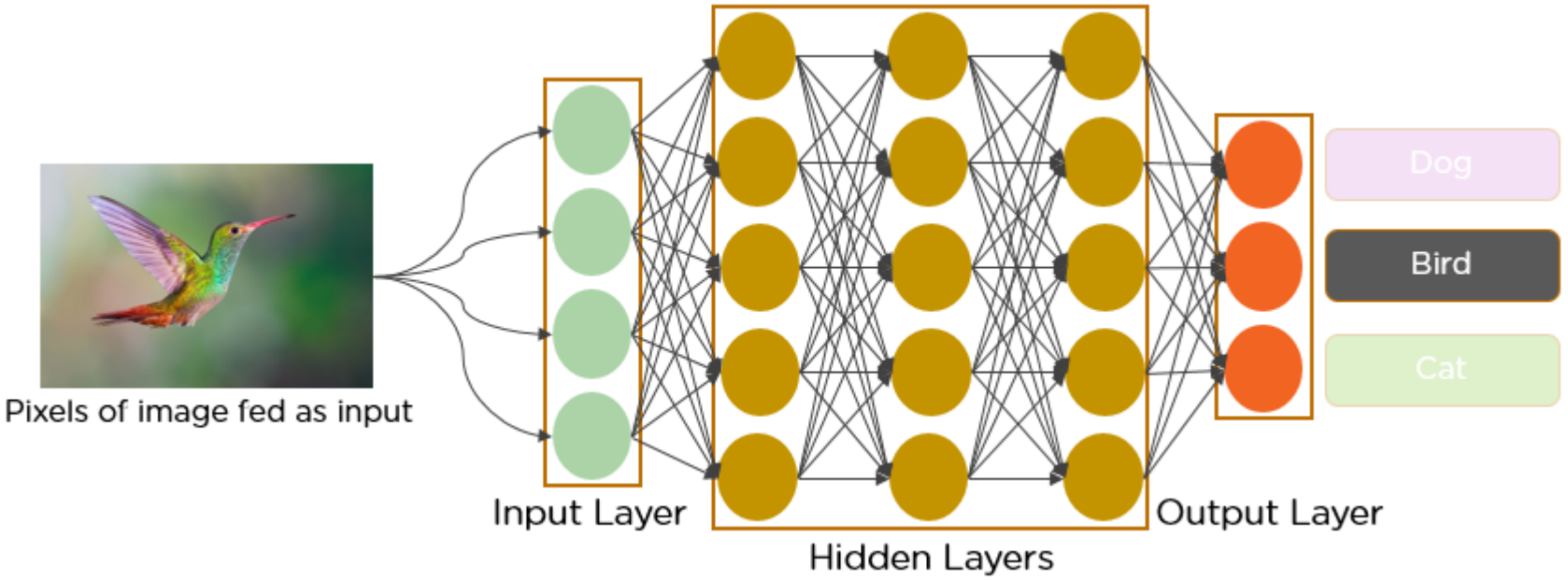
A Fast Learning Algorithm for Deep Belief Nets, Hinton, 2006



http://www.cs.toronto.edu/~fritz/absps/ncfast.pdf

Deep neural networks

- Very simply, deep learning is machine learning using neural networks that have multiple hidden layers.
- Number of hidden layers is another hyperparameter.



https://www.analyticsvidhya.com/blog/2021/05/convolutional-neural-networks-cnn/

Activation functions

- Recall the perceptron where neurons use the sigmoid function on the weighted sum of their input.
- This function used on the weighted sum of the input is called the activation function.
  - Sigmoid is classic, but it is not the only activation function in use
  - Others include: tanh, ReLu (“rectified linear unit”)
  - A full list can be found at https://en.wikipedia.org/wiki/Activation_function #Comparison_of_activation_functions

Why “Deep”?

- In theory, any classification function can be learned with 1 hidden layer if enough hidden units, why do we need deep learning?
  - With just one hidden layer, you might need very large number of neurons, and training is time-consuming
  - More hidden layer work better in practice
  - Analogy with human vision - early layers correspond to primitive features (e.g. straight lines), late layers correspond to higher-level components (e.g. eye, nose, mouth).
  - No need to hand-craft the input attributes that describe the data

Deep Learning & the ImageNet Challenge

- ImageNet - a large visual database for visual object recognition
- Classify 150K images into 1,000 categories (e.g. Egyptian cat, gazelle, wok, photocopier)
- 5 guesses allowed per picture
- In 2012, AlexNet, a ‘deep’ Neural Network won by a huge margin, achieving 12% error
- By now, the best is around 3% error
- You can learn about ImageNet here: https://thephotographersgallery.org.uk/whats-on/imagenet-10th-birthday-party

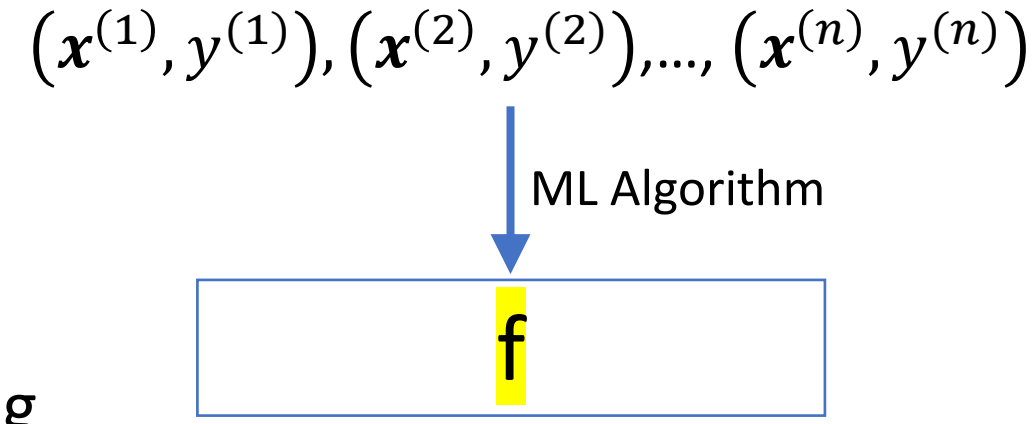
Extensions

- Many variants of neural networks – very active area of research
- Generative Adversarial Neural Networks (GAN)
  - Generator network can produce new samples (e.g. new faces)
  - Neural Network discriminator classifies face as ‘real’ or ‘fake’
  - Generator weights are updated based on how well it fools the discriminator

- See if you can do better than a GAN: http://www.whichfaceisreal.com/

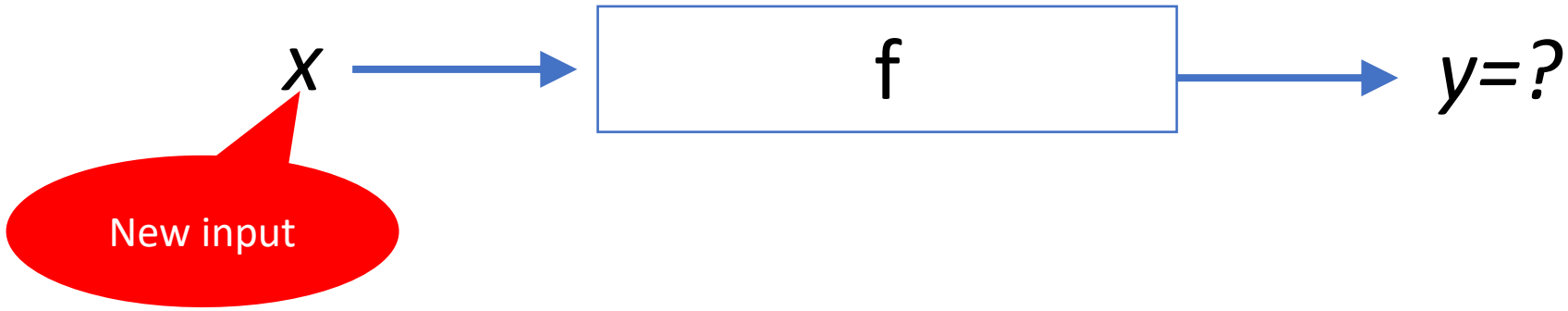
Overfitting

Training data and test data



- Recall the workflow of supervised learning
- This applies to all supervised learning, including deep learning
  - Training data (input, output) is used to train the model (the neural net)
  - Test data is new inputs to feed the obtained neural net, which then must predict appropriate outputs

Training data and test data

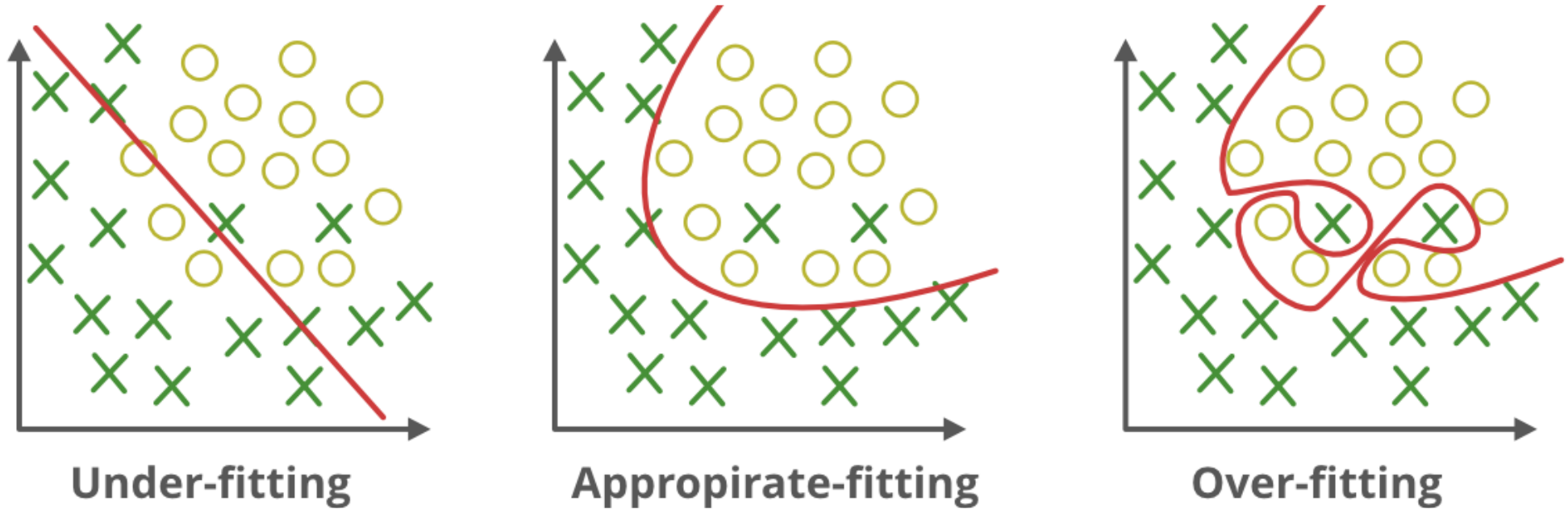


- Recall the workflow of supervised learning
- This applies to all supervised learning, including deep learning
  - Training data (input, output) is used to train the model (the neural net)
  - Test data is new inputs to feed the obtained neural net, which then must predict appropriate outputs

Overfitting

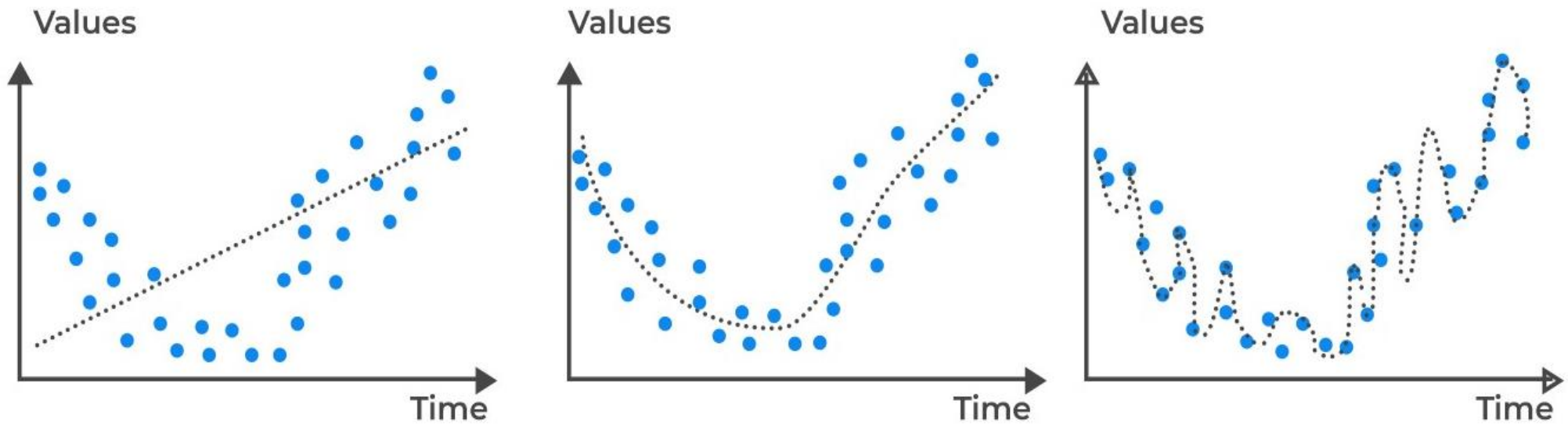
- Fitting the training data too well is BAD! Why?
- Remember the data you actually want to classify, or predict for, is not the same as the training data – so learning every irrelevant detail (noise) in a training data set will not help
- Overfitting happens when the model is more complex than required

Classification



https://www.geeksforgeeks.org/regularization-in-machine-learning/

Regression



https://analystprep.com/study-notes/wp-content/uploads/2021/03/Img_13.jpg

Regularisation

- Neural networks, like all nonlinear models, can overfit the data
- One way to guard against overfitting is regularisation
- Sometimes we need to regularise even linear models (e.g. if there is too much noise in the data)

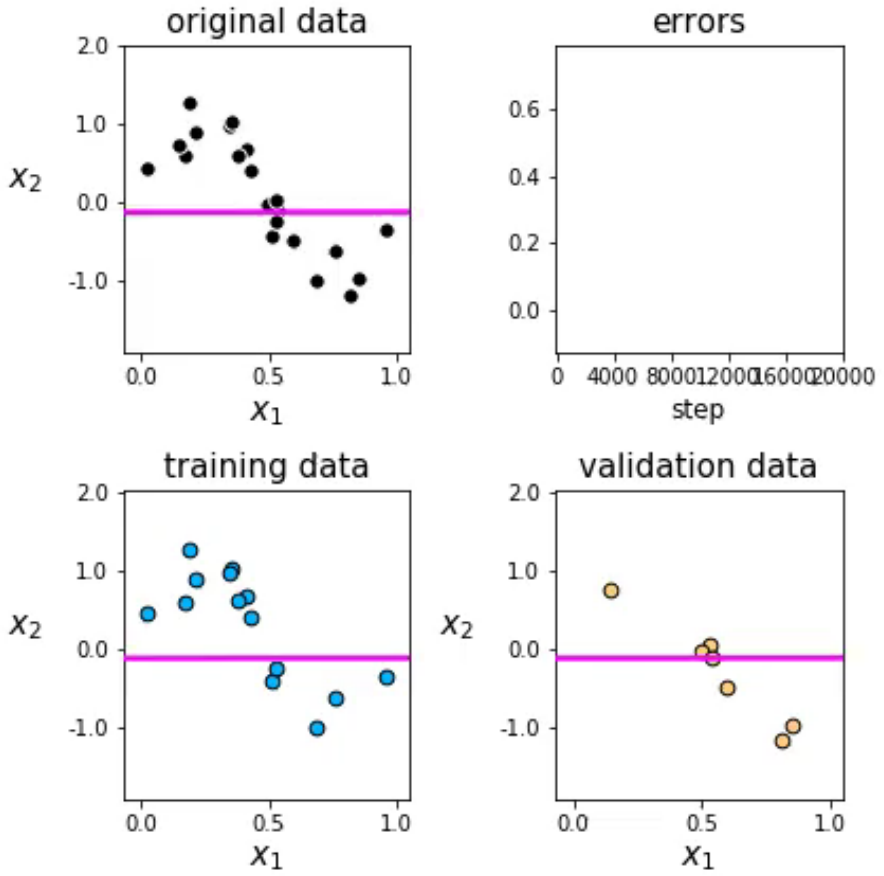
Ways to regularise:

- Add a penalty to the cost function to penalise more complex models
  - e.g. the number of free parameters, or the magnitude of weights
- Prune the model
  - "Dropout" is the process of "leaving out" a proportion of nodes (typically half) when training a deep network.

Early stopping

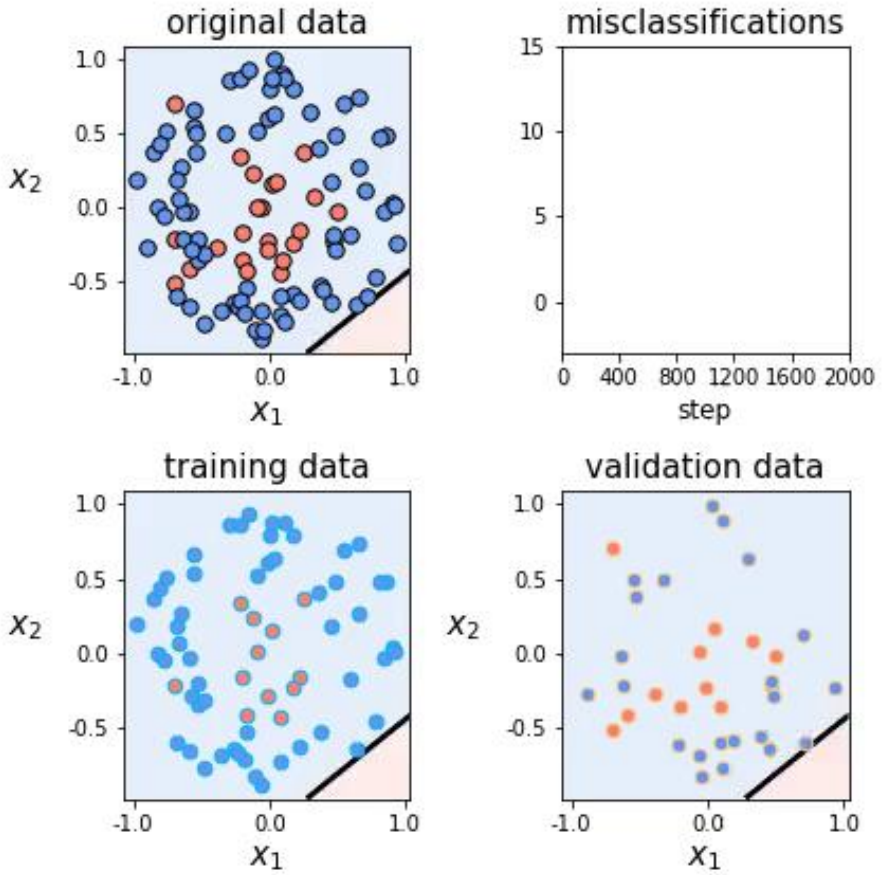
- Stopping the training early is another effective way to guard against overfitting
- After each gradient update (or Backprop cycle), the training cost will decrease until it reaches 0.
- Set aside a subset of the data (called hold-out set) to use only for monitoring the cost on previously unseen data.
- The error on hold-out set will decrease at first, but as training continues, it can start increasing
- Stop training when the error on hold-out set starts increasing.

Regression example



https://github.com/jermwatt/machine_learning_refined/blob/gh-pages/notes/13_Multilayer_perceptrons/videos/animation_5.mp4

Classification example



https://github.com/jermwatt/machine_learning_refined/blob/gh-pages/notes/13_Multilayer_perceptrons/videos/animation_6.mp4

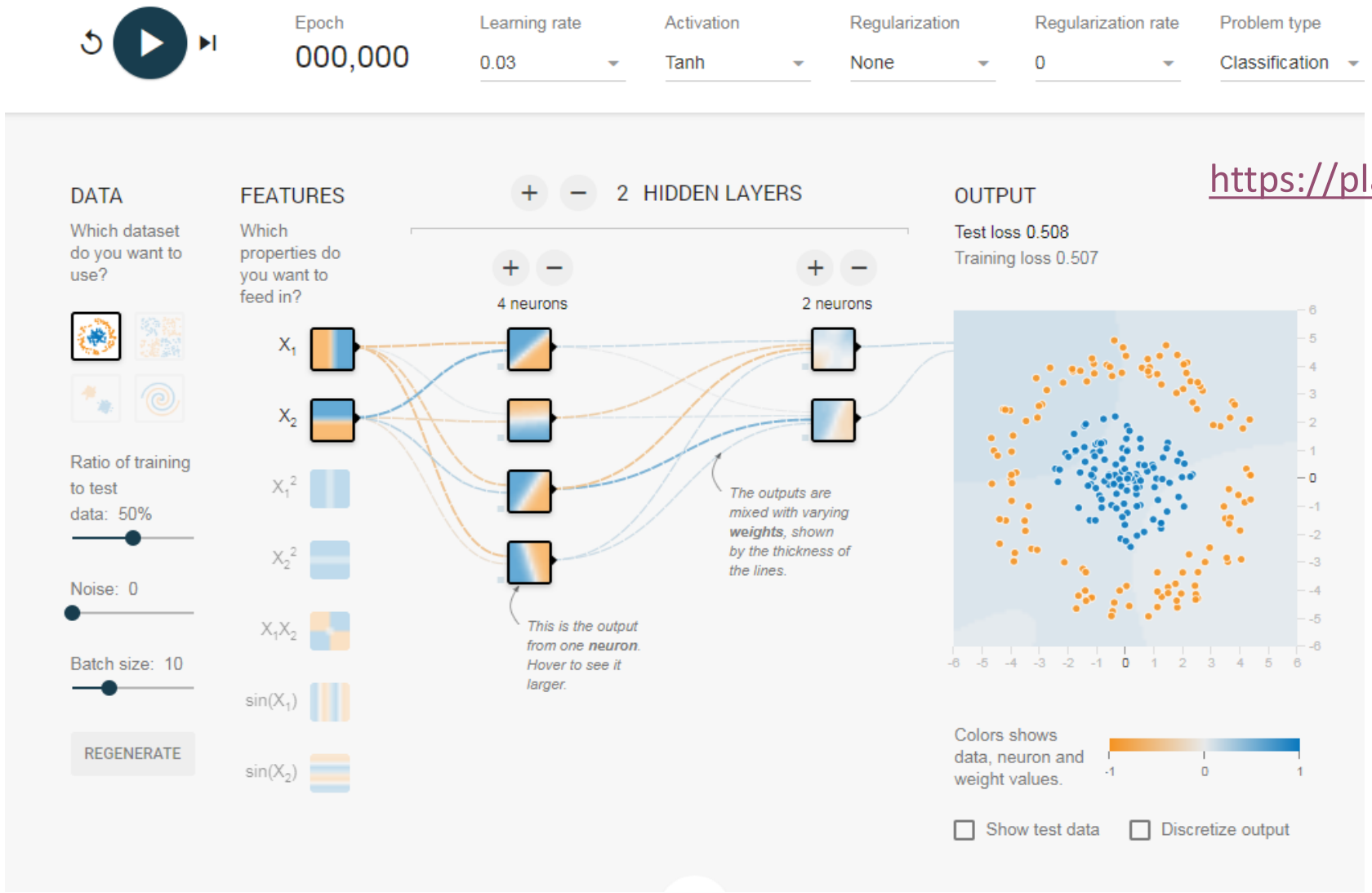
Next

As we finish our study of supervised learning, one last video lecture next week will cover:

- How to tune hyperparameters
- How to evaluate / validate supervised learning models

Then you will be ready to have a go at running supervised learning methods in practice!

Playground: Try out neural networks!



https://playground.tensorflow.org