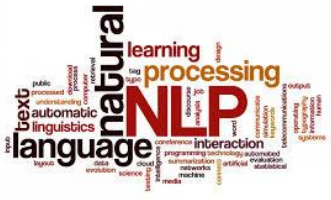
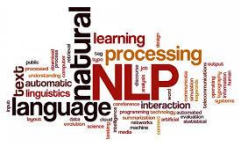
Keras


natural
learning
processing
NLP
text
automatic
linguistics
language
interaction

Agenda

- Keras



Keras

- The initial building block of Keras is a model, and the simplest model is called sequential.
- A sequential Keras model is a linear pipeline (a stack) of neural networks layers.

The artificial neuron

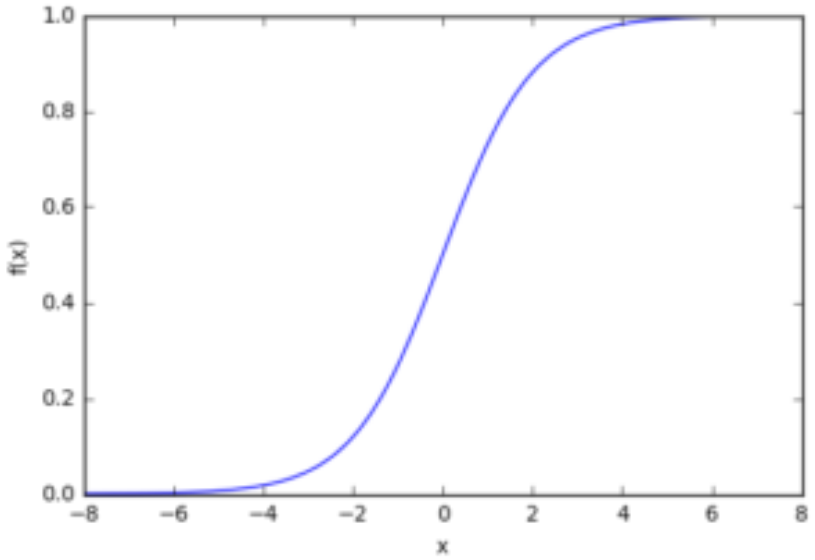
- The biological neuron is simulated in an ANN by an activation function.
- In classification tasks (e.g. identifying spam e-mails) this activation function has to have a “switch on” characteristic – in other words, once the input is greater than a certain value, the output should change state i.e. from 0 to 1, from -1 to 1 or from 0 to >0.
- This simulates the “turning on” of a biological neuron.

Activation function - sigmoid

- A common activation function that is used is the sigmoid function

- function is “activated” i.e. it moves from 0 to 1 when the input *x* is greater than a certain value.
- The edge is “soft”, and the output doesn’t change instantaneously

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



Activation function - ReLU

- A simple function called Rectified Linear Unit (ReLU) became very popular because it generates very good experimental results.
- A ReLU is simply defined as , and the nonlinear function
- *f(x) = max(0,x);*
- function is zero for negative values, and it grows linearly for positive values:

Activation function - sigmoid

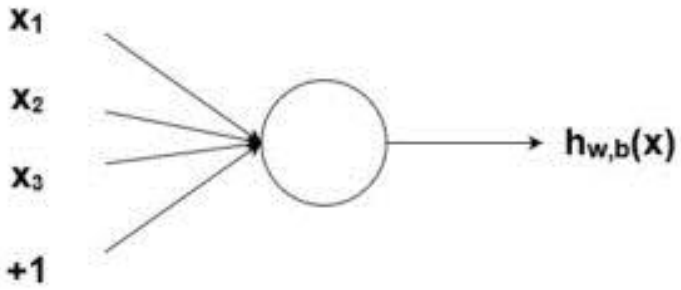
- Keras supports a number of activation functions, and a full list is available at

- https://keras.io/activations

```
# create model
model = Sequential()
model.add(Dense(12, input_dim=8, activation='relu'))
model.add(Dense(8, activation='relu'))
model.add(Dense(1, activation='sigmoid'))
```

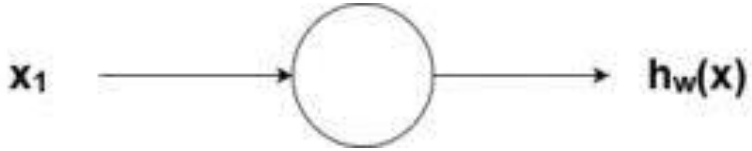
Nodes

- also called a perceptron
- Biological neurons are connected hierarchical networks, with the outputs of some neurons being the inputs to others.
- We can represent these networks as connected layers of *nodes.*
- Each node takes multiple weighted inputs, applies the *activation function* to the summation of these inputs, and in doing so generates an output.

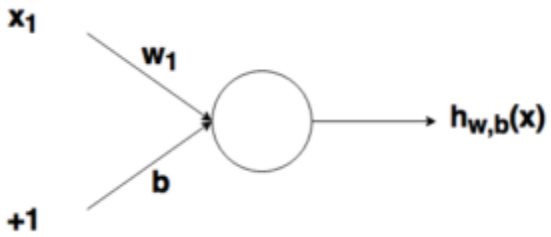


$$x_1 w_1 + x_2 w_2 + x_3 w_3 + b$$

Bias



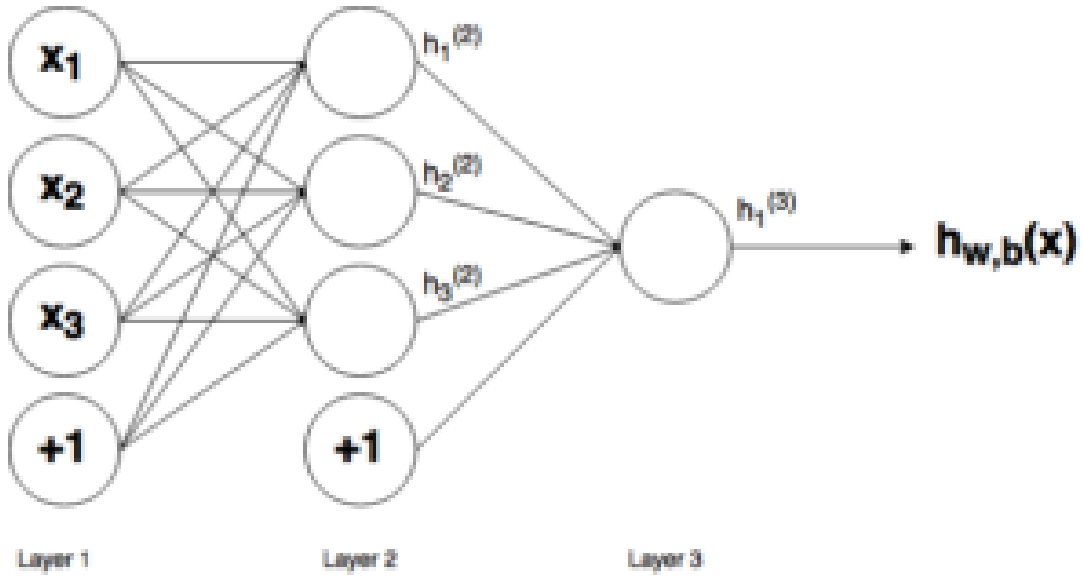
- the w_i values are weights.
- They are the variables that are changed during the learning process, and, along with the input, determine the output of the node.
- The b is the weight of the +1 bias element – the inclusion of this bias enhances the flexibility of the node



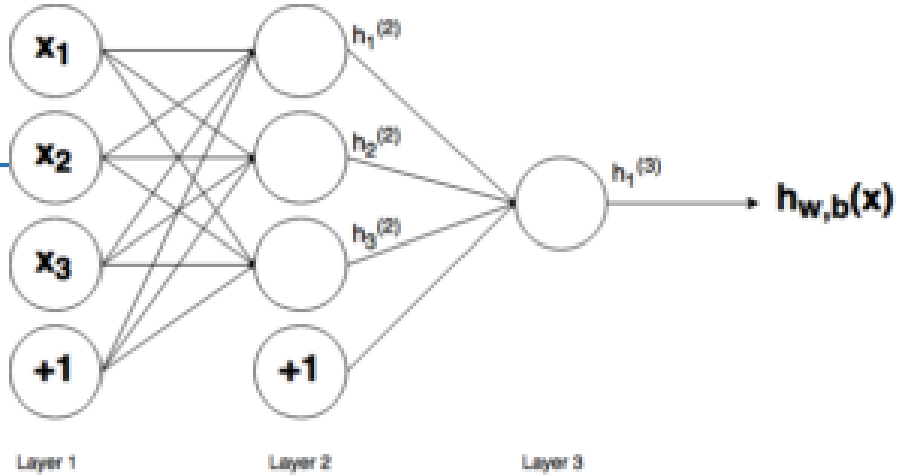
The structure

Many such interconnected nodes in a fully fledged neural network.

These structures can come in a myriad of different forms, but the most common simple neural network structure consists of an input layer, a hidden layer and an output layer.



Layers

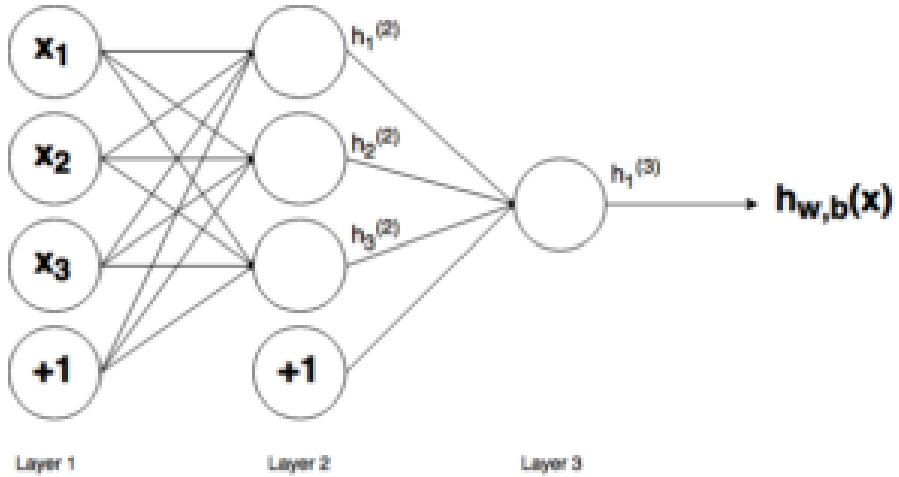


- 3 layers of the network
- Layer 1 represents the ***input layer*** –
  - the external input data enters the network.
- Layer 2 is called the ***hidden layer*** as this layer is not part of the input or output.
- Neural networks can have many hidden layers, - here one.
- Layer 3 is the ***output layer***.
- Many connections between the layers, in particular between Layer 1 (L1) and Layer 2 (L2).
- As can be seen, each node in L1 has a connection to all the nodes in L2.
- Likewise for the nodes in L2 to the single output node L3.
- Each of these connections will have an associated weight.

Layers

- Fully connected layers are defined using the Dense class.
- Specify the number of neurons in the layer as the first argument,
- specify the activation function using the **activation** argument.

```
# create model
model = Sequential()
model.add(Dense(12, input_dim=8, activation='relu'))
model.add(Dense(8, activation='relu'))
model.add(Dense(1, activation='sigmoid'))
```

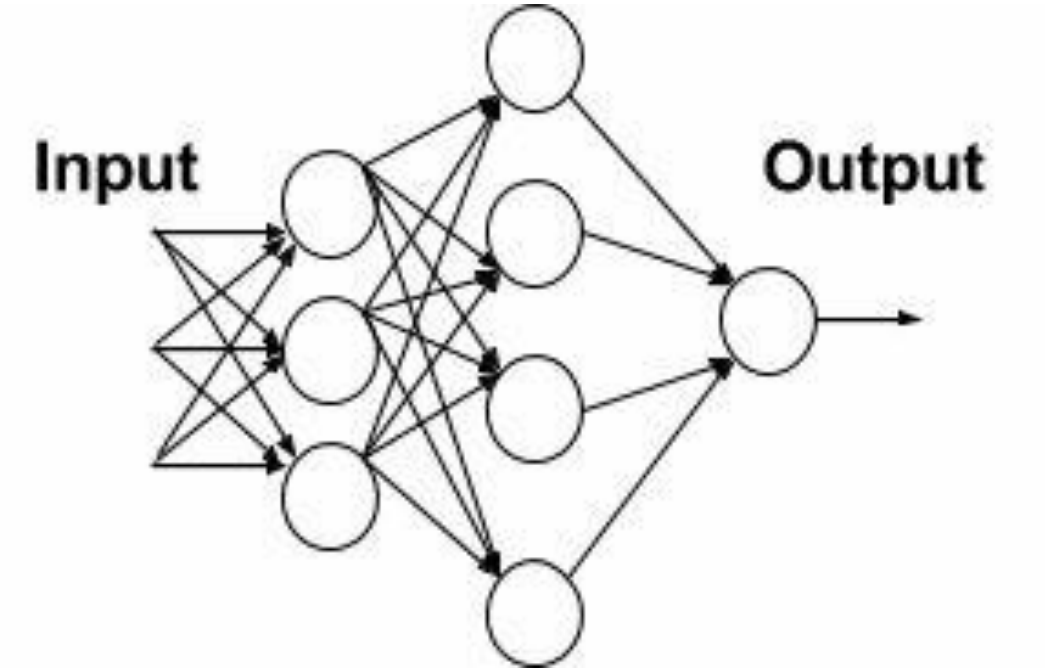


Keras

- defines a single layer with
- 12 artificial neurons, and
- it expects 8 input variables
  - also known as features
- Each neuron can be initialized with specific weights.
- Keras provides a few choices, the most common :
  - random_uniform: Weights are initialized to uniformly random small values in (*-0.05*, *0.05*).
    - Any value within the given interval is equally likely to be drawn.
  - random_normal: Weights are initialized according to a Gaussian, with a zero mean and small standard deviation of *0.05*.
  - zero: All weights are initialized to zero.
- Full - https://keras.io/initializations/.

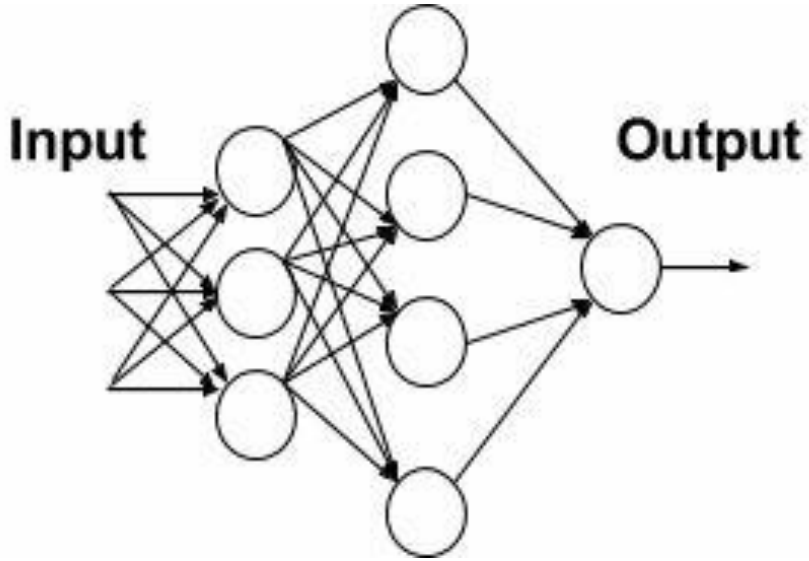
Multilayer perceptron - MLP

- Historically, Perceptron was the name given to a model having one single linear layer
- If it has multiple layers, it is called multilayer perceptron (MLP).



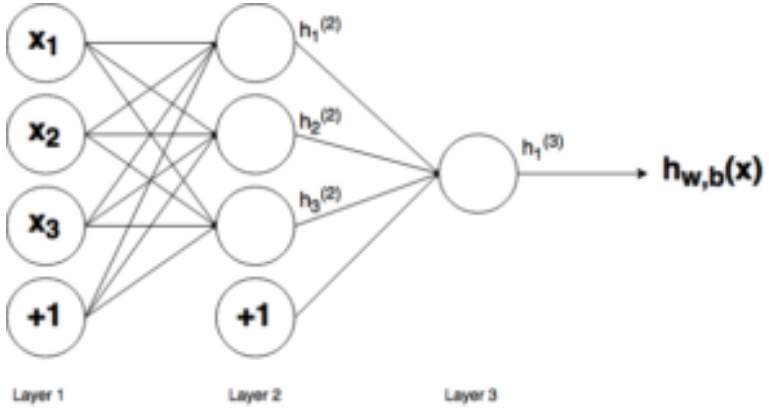
Multilayer perceptron - MLP

- Each node in the first layer receives an input and fires according to the predefined local decision boundaries.
- Output of the first layer is passed to the second layer, the results of which are passed to the final output layer consisting of one single neuron.
- Net is dense,
- each neuron in a layer is connected to
  - all neurons located in the previous layer and to
  - all the neurons in the following layer



Problems training the perceptron -Solution

- Choices for the weight *w* and the bias *b*?
- Ideally, we would like to provide a set of training examples
- and let the computer adjust the weight and the bias
- in such a way that the errors produced in the output are minimized.



Composing models in Keras

- There are two ways of composing models in Keras :
  - Sequential composition
  - Functional composition

Sequential composition

- Different predefined models are stacked together in a linear pipeline of layers similar to a stack or a queue.

Predefined neural network layers

- Keras has a number of prebuilt layers.
- Regular dense
  - A dense model is a fully connected neural network layer.