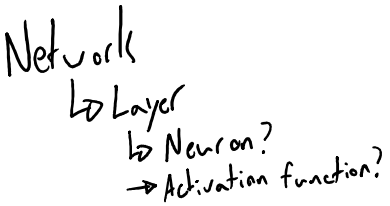
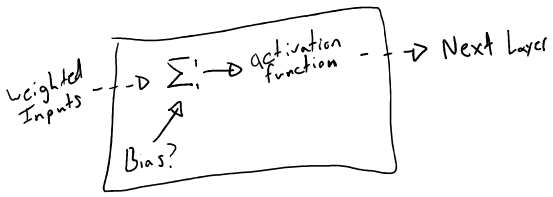
Components

January 2, 2022 3:51 PM

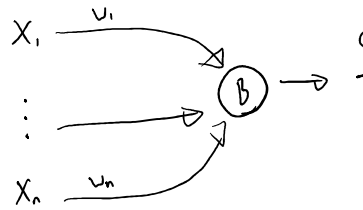


Neurons

January 2, 2022 4:22 PM



Perceptrons



$$\begin{cases} 0, & \text{if } \left(\sum_{i=1}^{n} x_i w_i\right) + B \leq 0 \\ 1, & \text{if } \left(\sum_{i=1}^{n} x_i w_i\right) + B > 0 \end{cases}$$

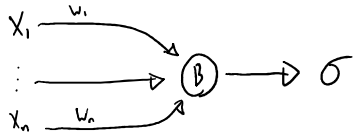
$x \in \{0, 1\}$

$w, B \in \mathbb{Z}$

→ Not ideal, since a small change in bias or weights may have a significant impact on outcome

Sigmoid Neurons

→ Ensures small change in weight/bias results in small change in outcome



$$z = \left(\sum_{i=1}^{n} x_i w_i\right) + B$$

$$\sigma = \frac{1}{1+e^{-z}} \rightarrow \sigma \in [0, 1]$$

Change in a neuron's output w.r.t. a change in the neuron's weights/bias:

$$\Delta \text{output} \doteq \sum_j \frac{\partial \text{output}}{\partial w_j} \Delta w_j + \frac{\partial \text{output}}{\partial B} \Delta B$$

} good estimate for small changes

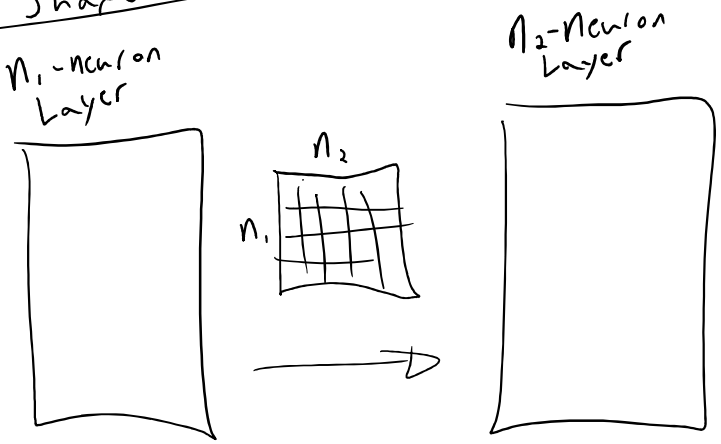
Basically, perceptrons act like a step function, while sigmoid neurons "smooth out" the function, allowing for subtle adjustments to the network.

Thinking page

January 2, 2022 6:00 PM

Network	Layer
- Layers[] - outputLayer	- Bias[] - Weight[][]
- addLayer	- Constructors - Accessors - Mutators

Data Shape



$$Value[\underset{row}{n_1}][\underset{col}{n_2}] = (n1_{out}) \times (weight[n_2][n_1]) + bias[n_2]$$

adding Layer

→ Construct data matrix b/n layers
  ↳ member of the 2nd layer in the pair
    ↳ requires 1 array from 1st (outputs)
    → needs 1 array and a 2D matrix from 2nd (biases, weights)

Should the data have its own object?
or Should it be transferred directly to each layer?

✓ own object

Own object

- Can use polymorphism to support multiple forms of data.
- Could maybe build activ. function directly into data object
- Tidier code?
- might have to make data matrix resizable.
  - or create several data objects
- allows easy organization of data sample
  - can keep input, output, expected output, etc. together for easy backpropogation, etc.