Neural Networks and Deep Learning, Chapter 4

The Universal Approximation Theorem

11/3/19

Jeffrey Yoo

Universal Approximation Theorem

- Neural networks with a single hidden layer can "compute" any functions.
- More precisely,

  Let our desired function be f(x) [1] and output of neural network g(x) [2].

  Then, for any desired ϵ, we can guarantee

$$|g(x) - f(x)| < \epsilon$$

Caveats

1. f(x) must be a continuous function
2. With sufficient number of hidden neurons

Universal Approximation Theorem

https://en.wikipedia.org/wiki/Universal_approximation_theorem

Let $\varphi : \mathbb{R} \to \mathbb{R}$ be a nonconstant, bounded, and continuous function (called the *activation function*). Let I_m denote the *m*-dimensional unit hypercube $[0,1]^m$. The space of real-valued continuous functions on I_m is denoted by $C(I_m)$. Then, given any $\varepsilon > 0$ and any function $f \in C(I_m)$, there exist an integer N, real constants $v_i, b_i \in \mathbb{R}$ and real vectors $w_i \in \mathbb{R}^m$ for $i = 1, \ldots, N$, such that we may define:

$$F(x) = \sum_{i=1}^{N} v_i \varphi\left(w_i^T x + b_i\right)$$

as an approximate realization of the function f; that is,

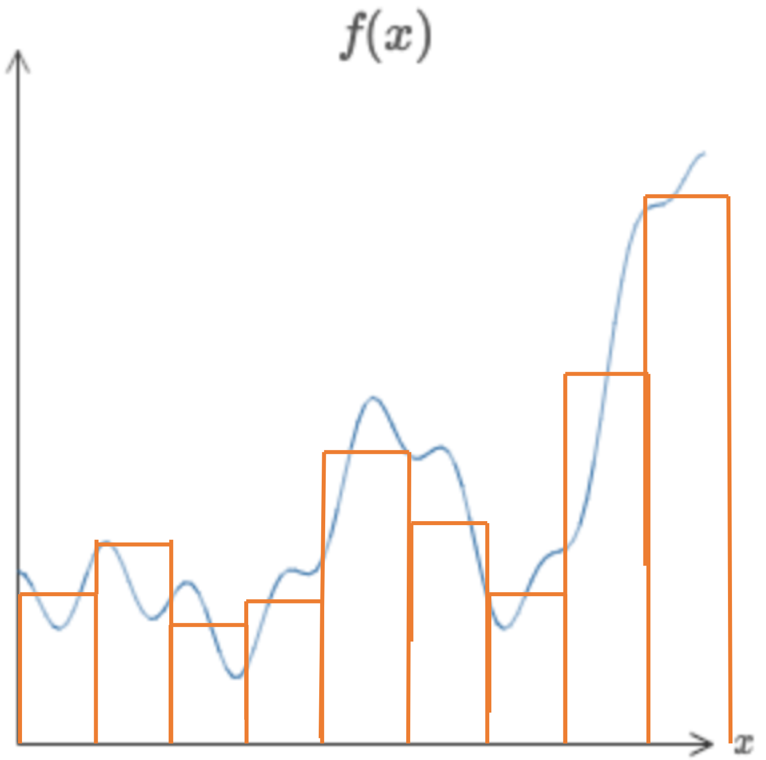
$$|F(x) - f(x)| < \varepsilon$$

for all $x \in I_m$. In other words, functions of the form $F(x)$ are dense in $C(I_m)$.

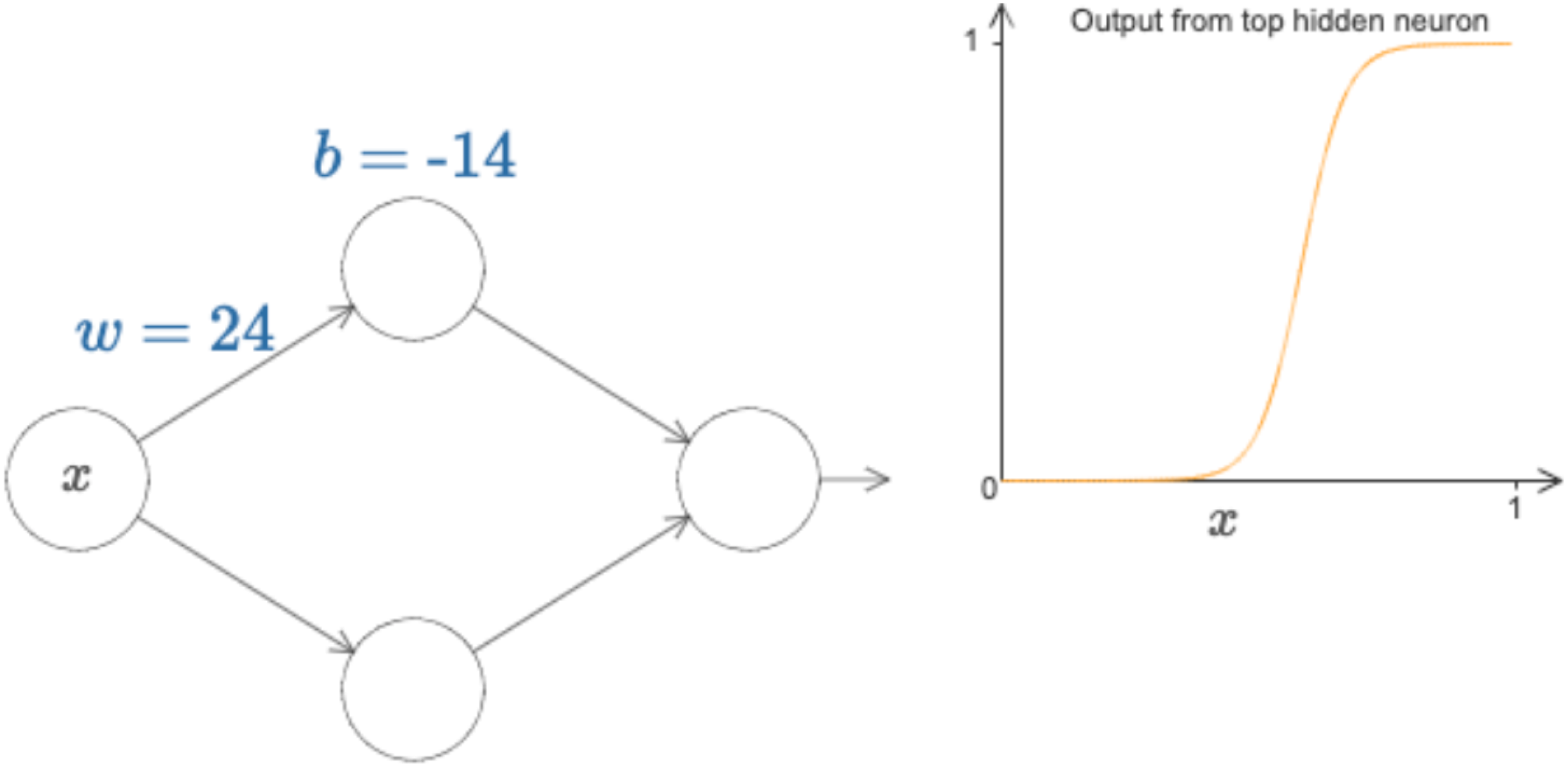
This still holds when replacing I_m with any compact subset of $\mathbb{R}^m$.

General Idea

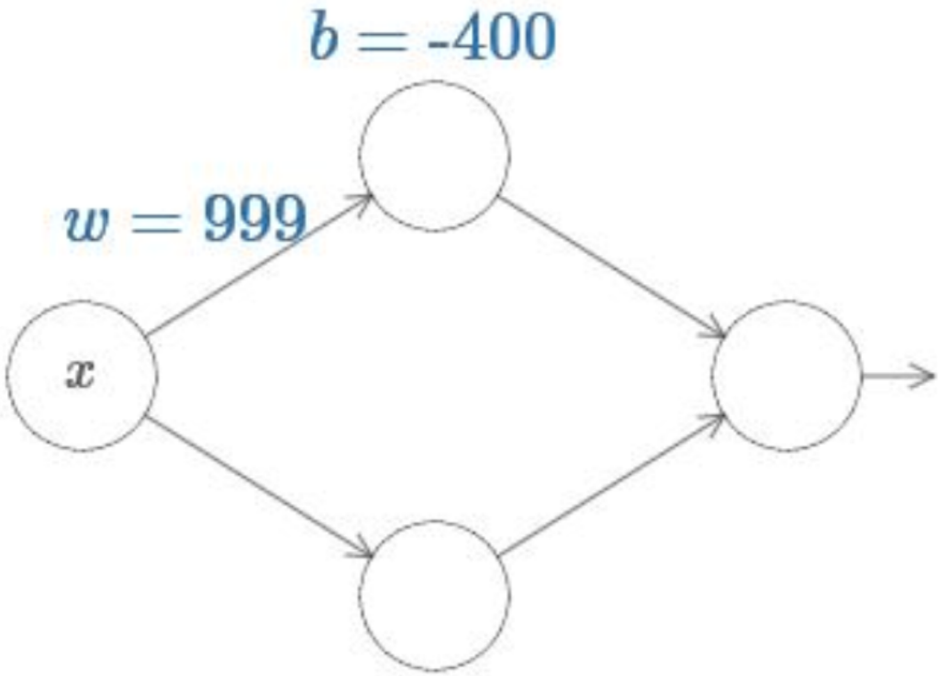
- How to approximate following function?
- Idea:
  - Use many step functions
  - Can do this using neural network with one layer of hidden neurons + sigmoid activation.

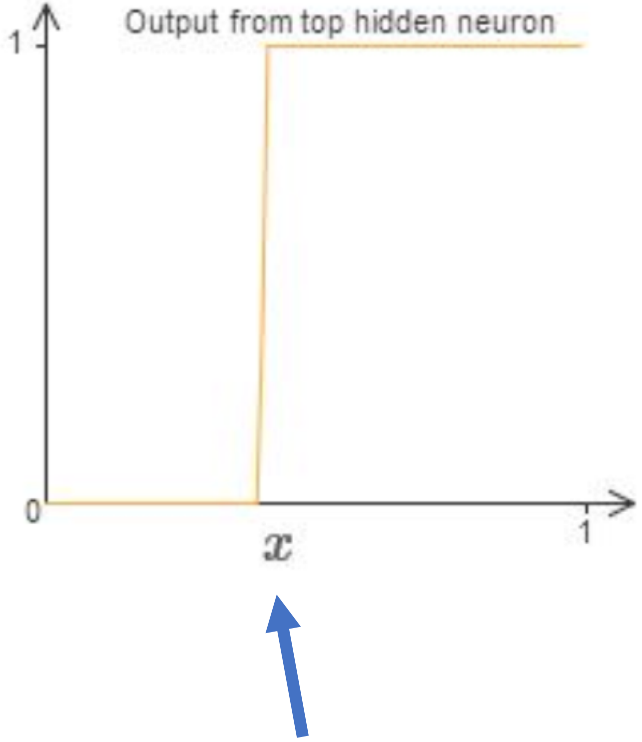


Example: one input variable



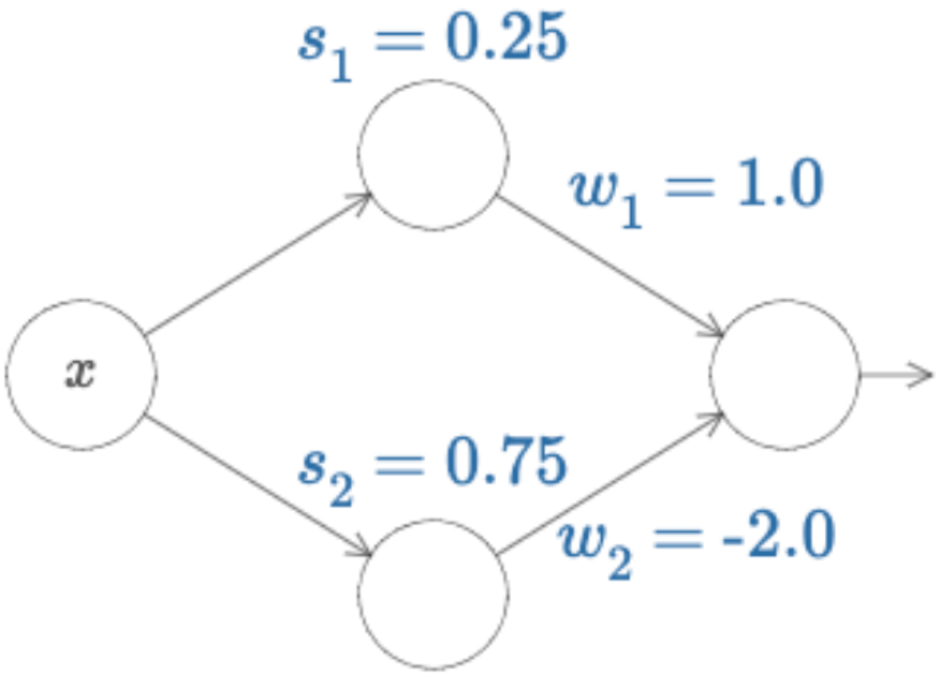
Example: one input variable

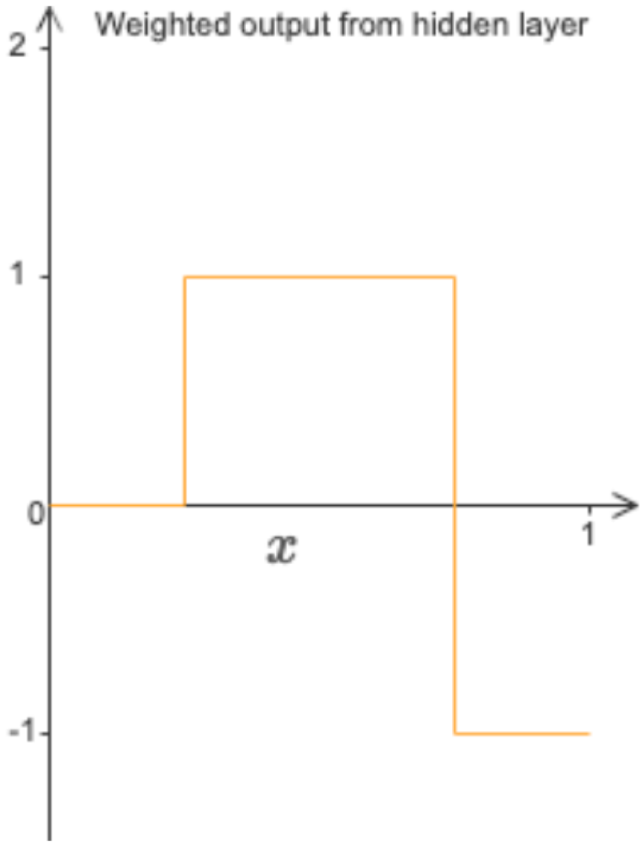




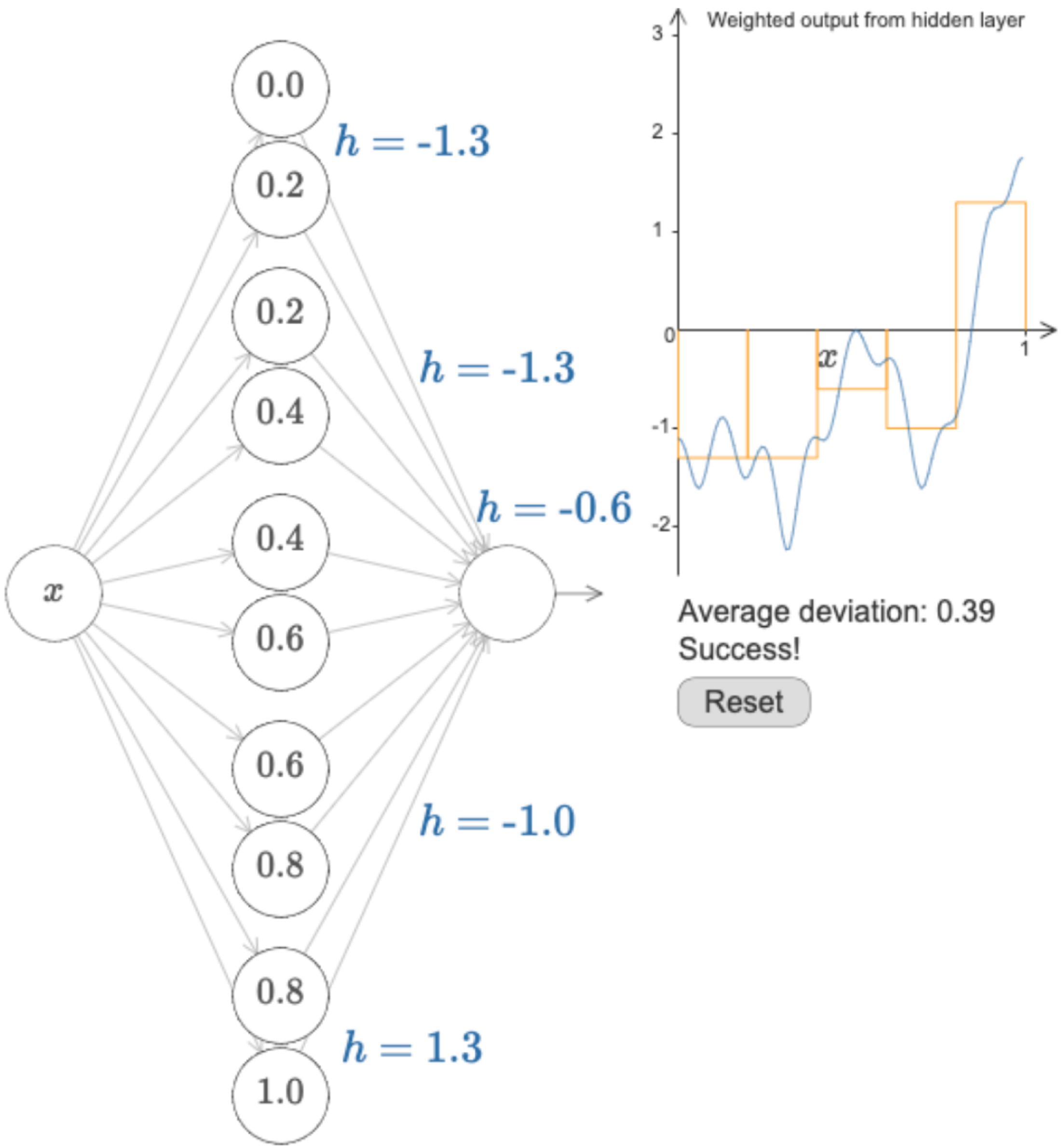
$$s = -\frac{b}{w} = -\frac{-400}{999} \approx 0.4$$

Example: one input variable





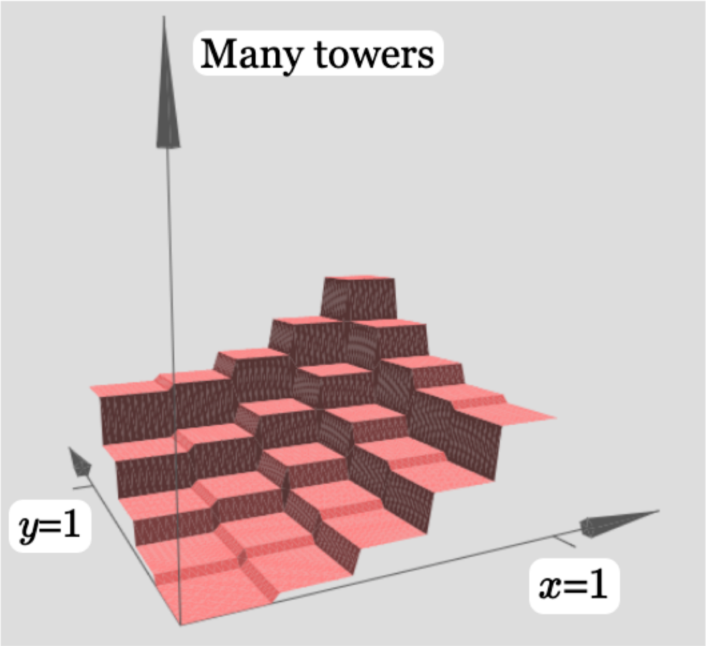
Example



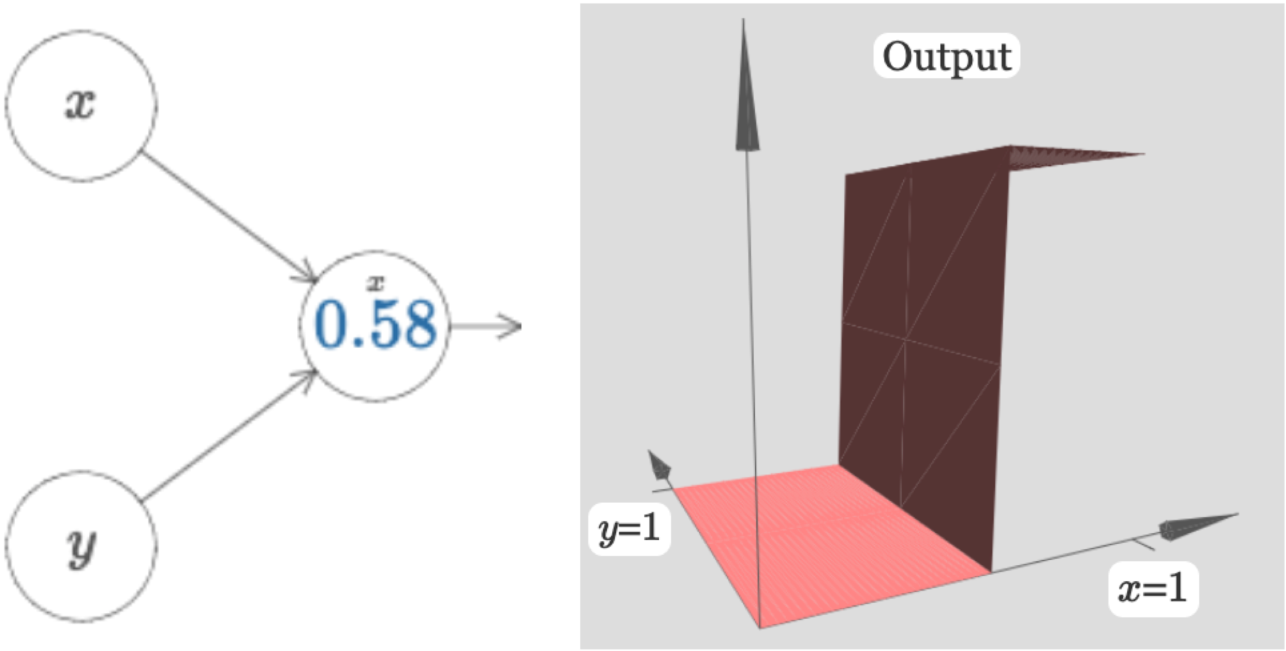
0.0
$h = -1.3$
0.2
0.2
$h = -1.3$
0.4
0.4
$h = -0.6$
x
0.6
0.6
$h = -1.0$
0.8
0.8
$h = 1.3$
1.0
Weighted output from hidden layer
3
2
1
0
-1
-2
x
1
Average deviation: 0.39
Success!
Reset

Multiple input variables

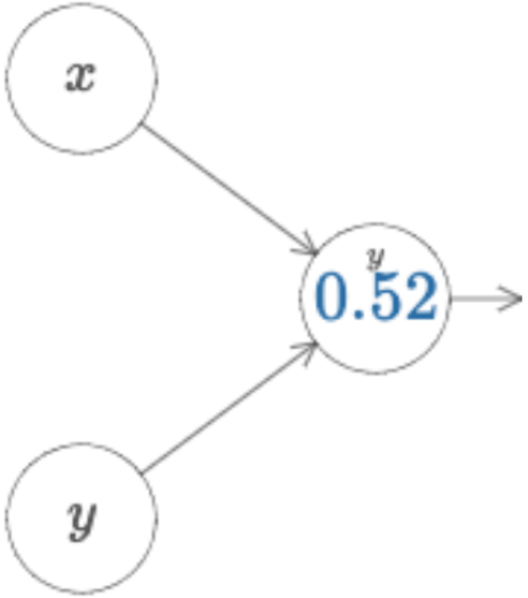
- The same idea can be generalized for multiple inputs
- Ex: If we have x and y as inputs, use “towers” to simulate f(x,y).

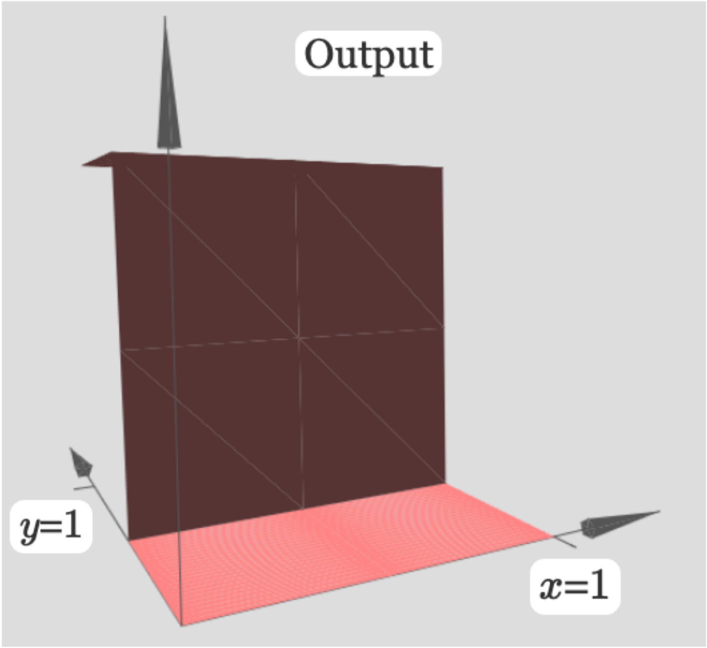


Example: Two inputs

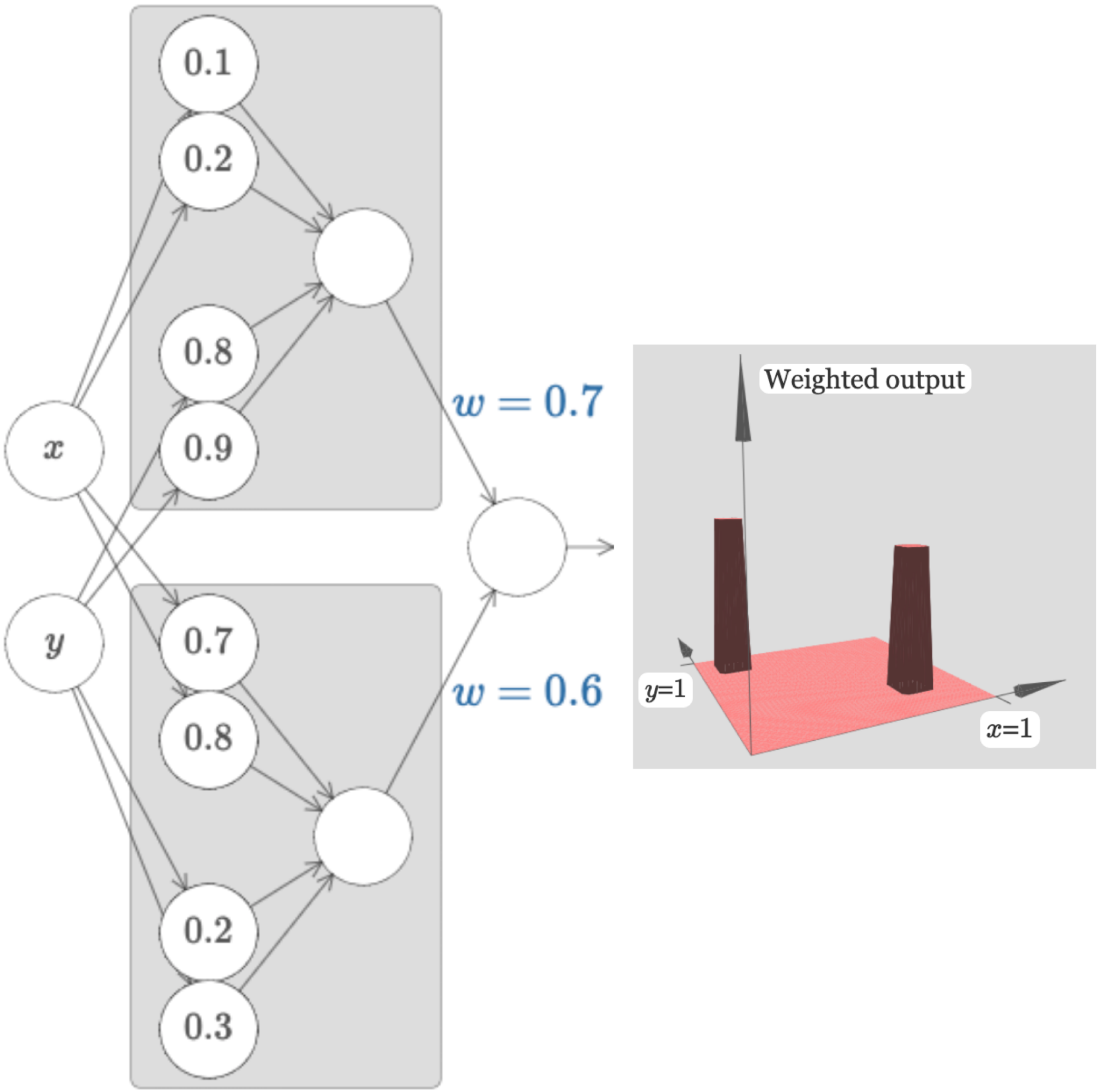


Example: Two inputs





Example



Other activation functions

- Recall we need our activation function to be a nonconstant, bounded, and continuous function.
- Would ReLU work?
- What about linear function $\phi(x) = x$?
- But…neural network with ReLU activation can be a universal approximator if its width is of n+4 (where n is input dimension)
- Paper: https://papers.nips.cc/paper/7203-the-expressive-power-of-neural-networks-a-view-from-the-width.pdf