Representation Power of Multilayer Network of Sigmoid Neurons

Representation power of a multilayer network of perceptrons

Representation power of a multilayer network of sigmoid neurons

Representation power of a multilayer network of perceptrons

A multilayer network of perceptrons with a single hidden layer can be used to represent any boolean function precisely (no errors)

Representation power of a multilayer network of sigmoid neurons

Representation power of a multilayer network of perceptrons A multilayer network of perceptrons with a single hidden layer can be used to represent any boolean function precisely (no errors)

Representation power of a multilayer network of sigmoid neurons A multilayer network of neurons with a single hidden layer can be used to approximate any continuous function to any desired precision

Representation power of a multilayer network of perceptrons A multilayer network of perceptrons with a single hidden layer can be used to represent any boolean function precisely (no errors)

Representation power of a multilayer network of sigmoid neurons A multilayer network of neurons with a single hidden layer can be used to approximate any continuous function to any desired precision In other words, there is a guarantee that for any function $f(x) : \mathbb{R}^n \rightarrow \mathbb{R}^m$, we can always find a neural network (with 1 hidden layer containing enough neurons) whose output g(x) satisfies $|g(x) - f(x)| < \epsilon$!!

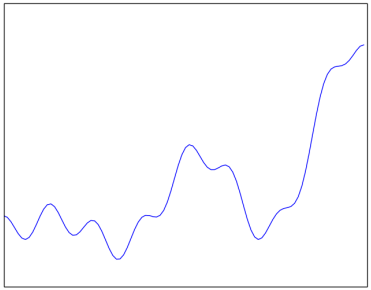
Representation power of a multilayer network of perceptrons A multilayer network of perceptrons with a single hidden layer can be used to represent any boolean function precisely (no errors)

Representation power of a multilayer network of sigmoid neurons A multilayer network of neurons with a single hidden layer can be used to approximate any continuous function to any desired precision In other words, there is a guarantee that for any function $f(x) : \mathbb{R}^n \rightarrow \mathbb{R}^m$, we can always find a neural network (with 1 hidden layer containing enough neurons) whose output g(x) satisfies $|g(x) - f(x)| < \epsilon$!!

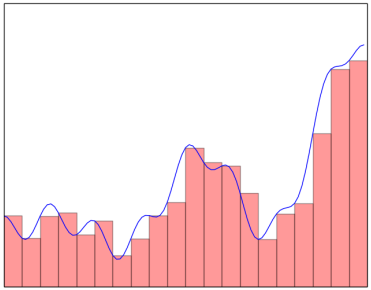
Proof: We will see an illustrative proof of this... [Cybenko, 1989], [Hornik, 1991]

- See this link* for an excellent illustration of this proof
- The discussion in the next few slides is based on the ideas presented at the above link

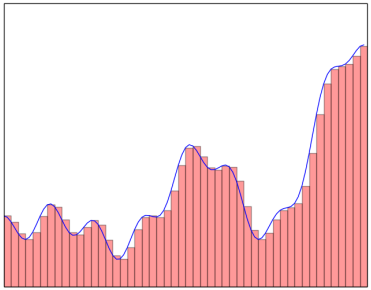
*http://neuralnetworksanddeeplearning.com/chap4.html



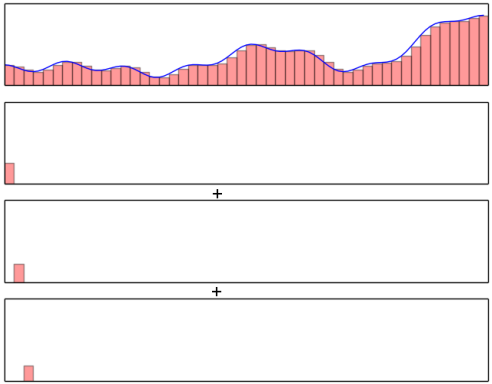
- We are interested in knowing whether a network of neurons can be used to represent an arbitrary function (like the one shown in the figure)



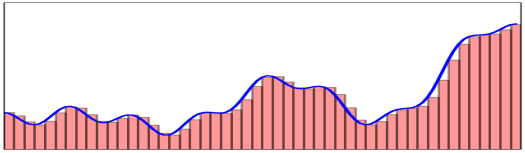
- We are interested in knowing whether a network of neurons can be used to represent an arbitrary function (like the one shown in the figure)
- We observe that such an arbitrary function can be approximated by several "tower" functions



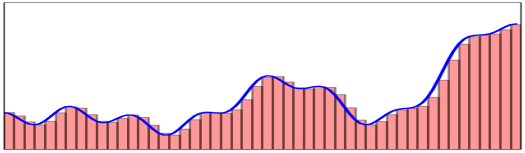
- We are interested in knowing whether a network of neurons can be used to represent an arbitrary function (like the one shown in the figure)
- We observe that such an arbitrary function can be approximated by several “tower” functions
- More the number of such “tower” functions, better the approximation



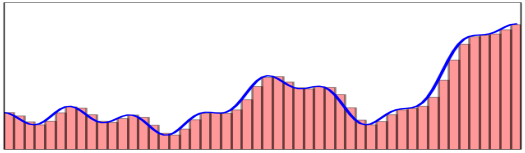
- We are interested in knowing whether a network of neurons can be used to represent an arbitrary function (like the one shown in the figure)
- We observe that such an arbitrary function can be approximated by several "tower" functions
- More the number of such "tower" functions, better the approximation
- To be more precise, we can approximate any arbitrary function by a sum of such "tower" functions



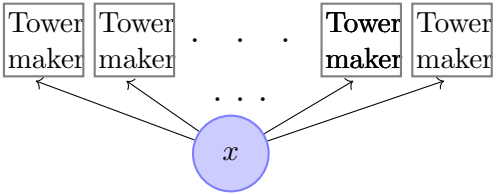
- We make a few observations

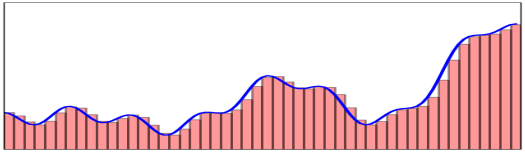


- We make a few observations
- All these "tower" functions are similar and only differ in their heights and positions on the x-axis

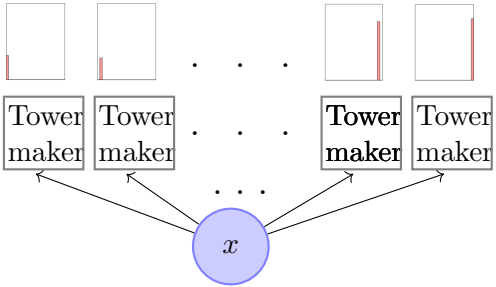


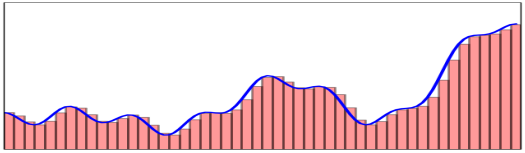
- We make a few observations
- All these "tower" functions are similar and only differ in their heights and positions on the x-axis
- Suppose there is a black box which takes the original input (x) and constructs these tower functions

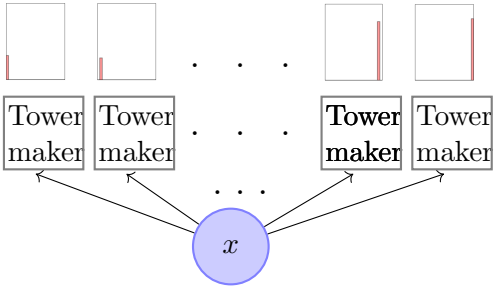




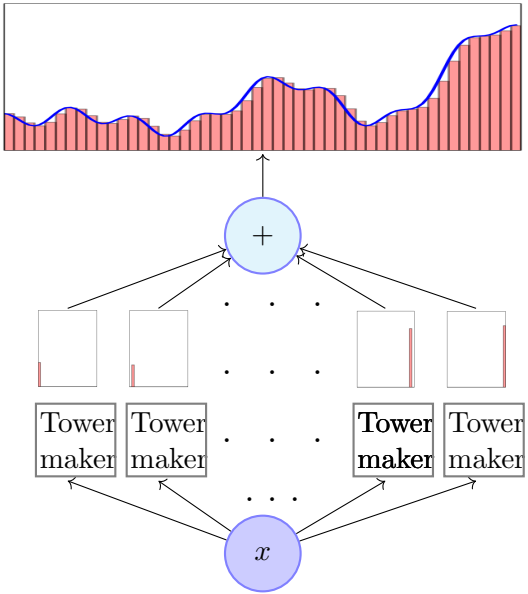
- We make a few observations
- All these "tower" functions are similar and only differ in their heights and positions on the x-axis
- Suppose there is a black box which takes the original input (x) and constructs these tower functions



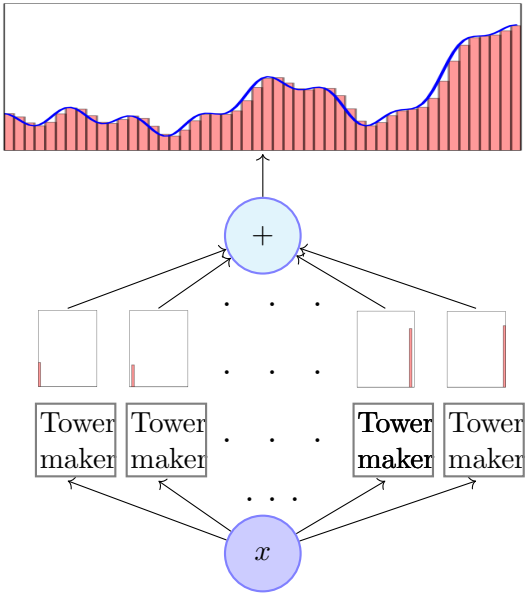




- We make a few observations
- All these "tower" functions are similar and only differ in their heights and positions on the x-axis
- Suppose there is a black box which takes the original input (x) and constructs these tower functions
- We can then have a simple network which can just add them up to approximate the function

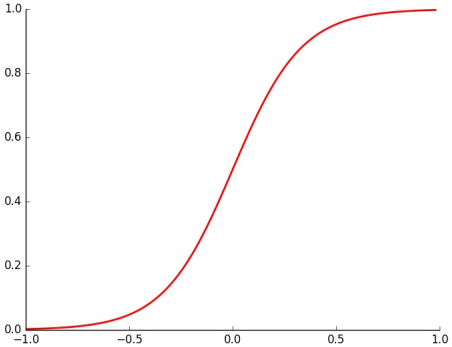


- We make a few observations
- All these "tower" functions are similar and only differ in their heights and positions on the x-axis
- Suppose there is a black box which takes the original input (x) and constructs these tower functions
- We can then have a simple network which can just add them up to approximate the function

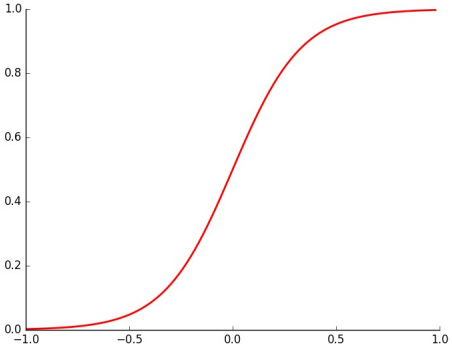


- We make a few observations
- All these "tower" functions are similar and only differ in their heights and positions on the x-axis
- Suppose there is a black box which takes the original input (x) and constructs these tower functions
- We can then have a simple network which can just add them up to approximate the function
- Our job now is to figure out what is inside this blackbox

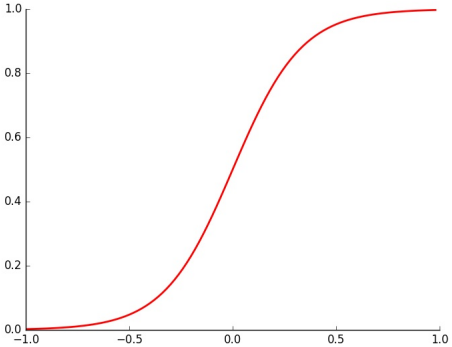
We will figure this out over the next few slides ...



- If we take the logistic function and set w to a very high value we will recover the step function

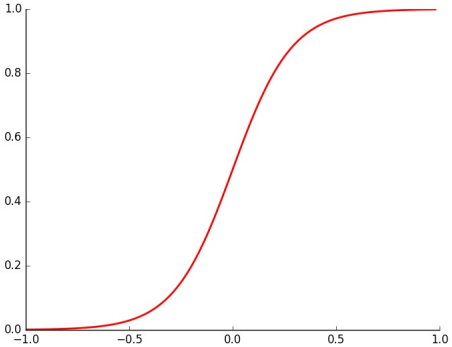


- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w



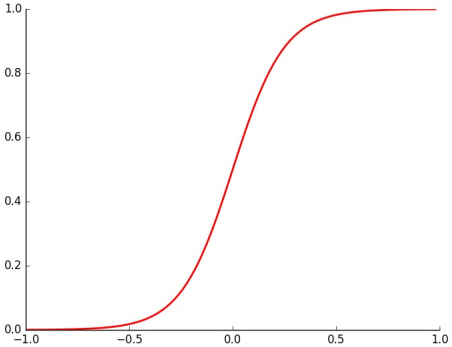
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w

$w = 6, b = 0$



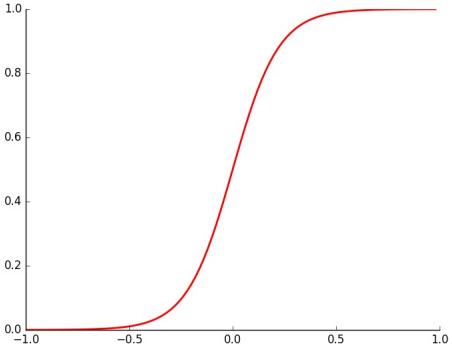
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w

$w = 7, b = 0$



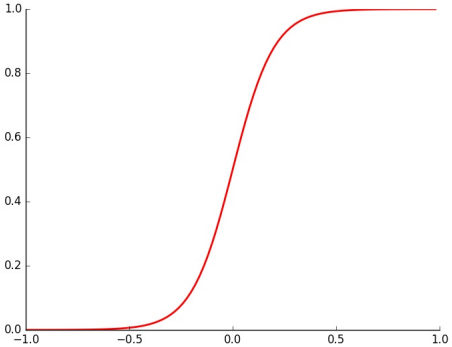
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w

$w = 8, b = 0$



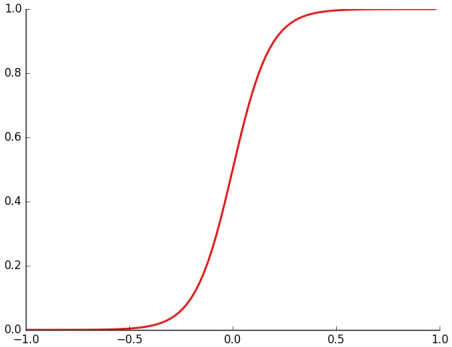
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w

$w = 9, b = 0$



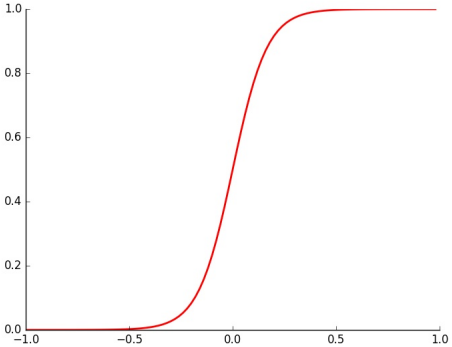
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w

$w = 10, b = 0$



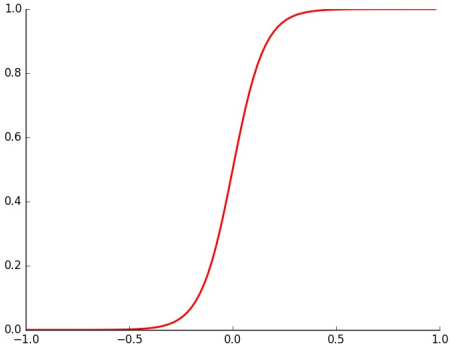
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w

$w = 11, b = 0$



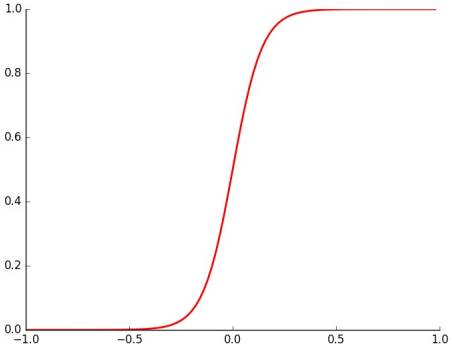
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w

$w = 12, b = 0$



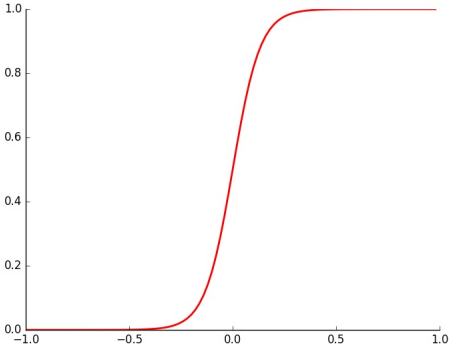
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w

$w = 13, b = 0$



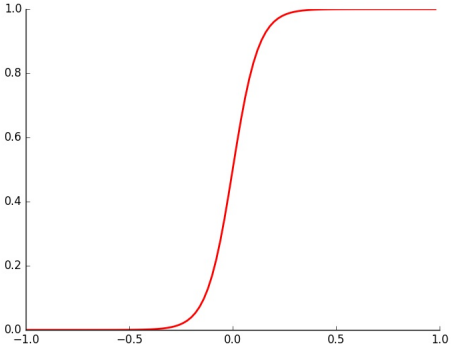
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w

$w = 14, b = 0$



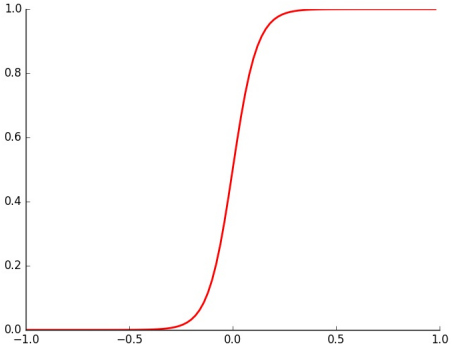
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w

$w = 15, b = 0$



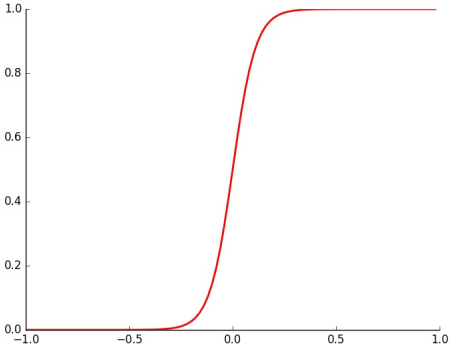
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w

$w = 16, b = 0$



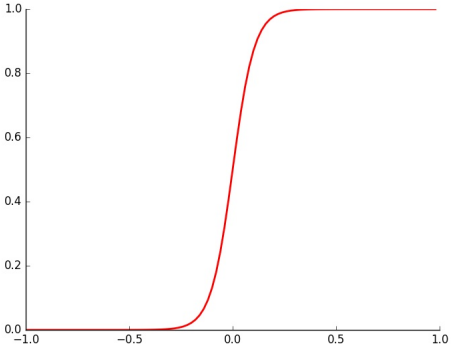
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w

$w = 17, b = 0$



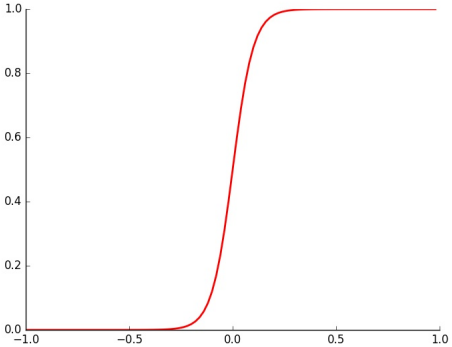
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w

$w = 18, b = 0$



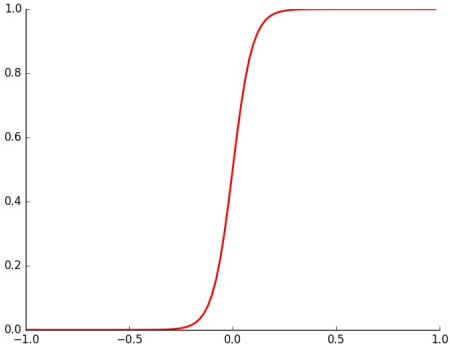
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w

$w = 19, b = 0$



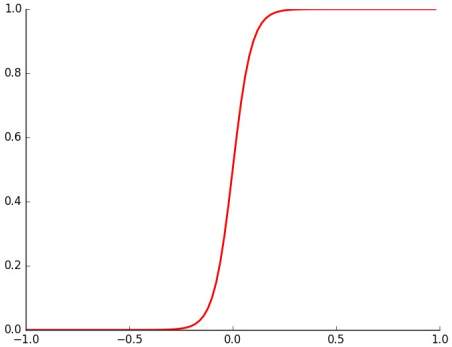
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w

$w = 20, b = 0$



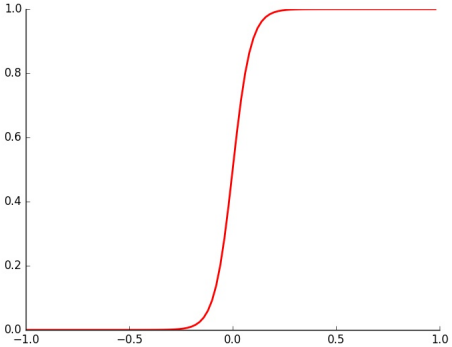
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w

$w = 21, b = 0$



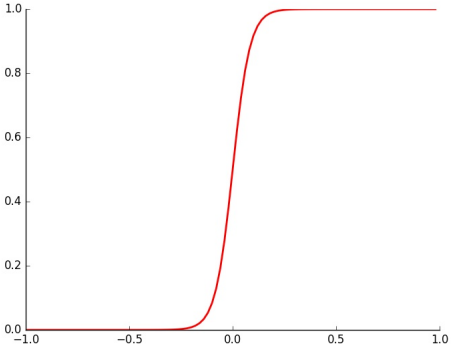
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w

$w = 22, b = 0$



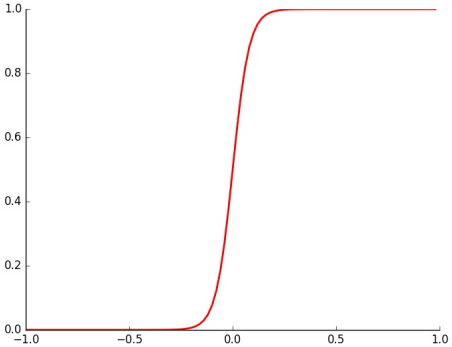
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w

$w = 23, b = 0$



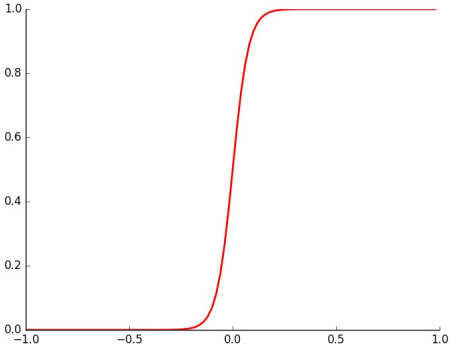
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w

$w = 24, b = 0$



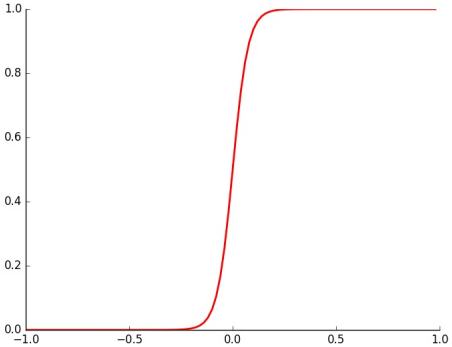
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w

$w = 25, b = 0$



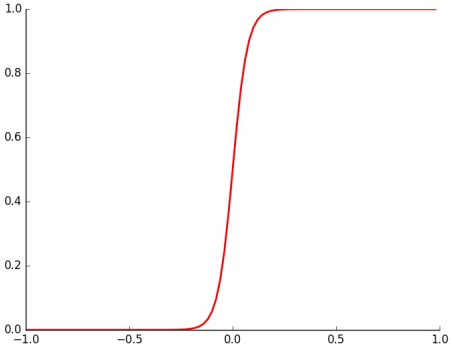
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w

$w = 26, b = 0$



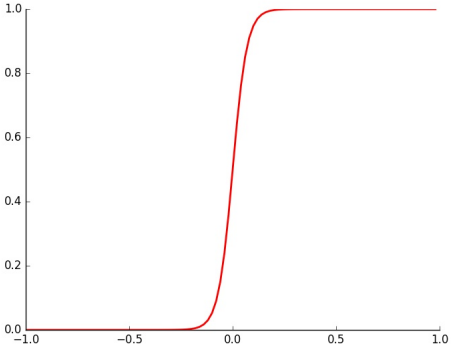
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w

$w = 27, b = 0$



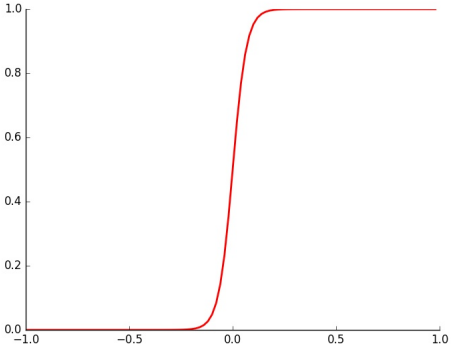
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w

$w = 28, b = 0$



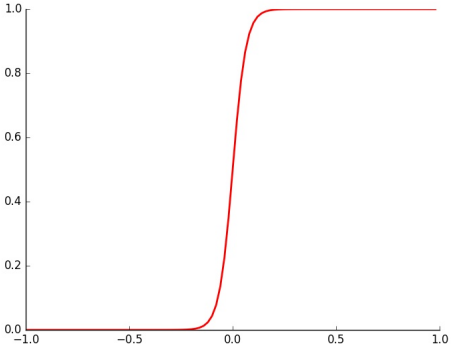
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w

$w = 29, b = 0$



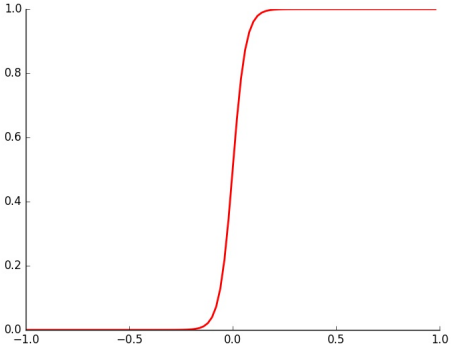
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w

$w = 30, b = 0$



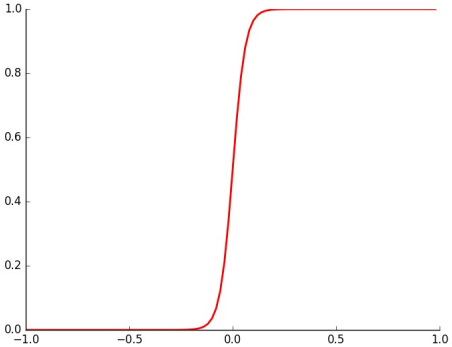
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w

$w = 31, b = 0$



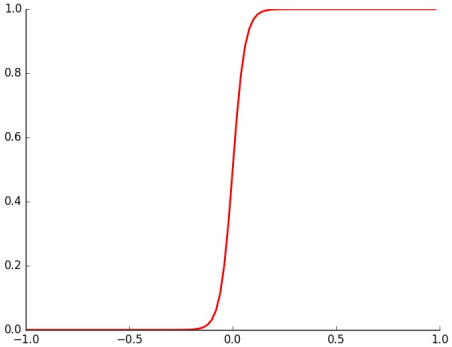
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w

$w = 32, b = 0$



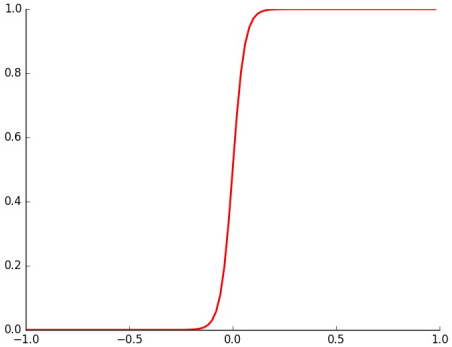
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w

$w = 33, b = 0$



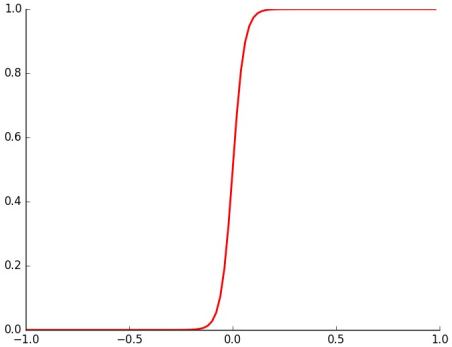
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w

$w = 34, b = 0$



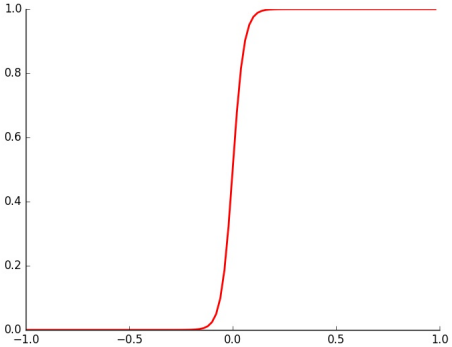
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w

$w = 35, b = 0$



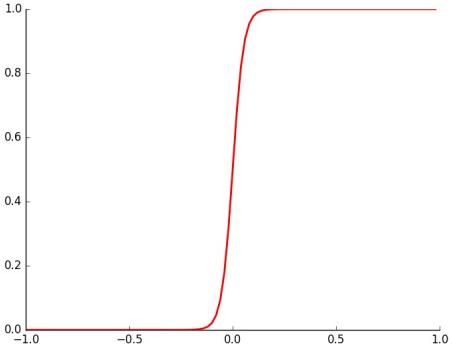
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w

$w = 36, b = 0$



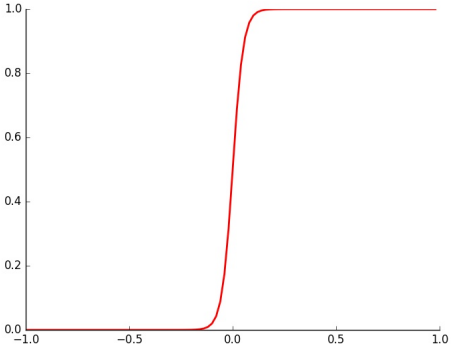
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w

$w = 37, b = 0$



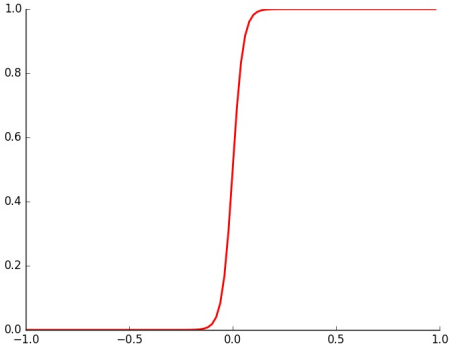
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w

$w = 38, b = 0$



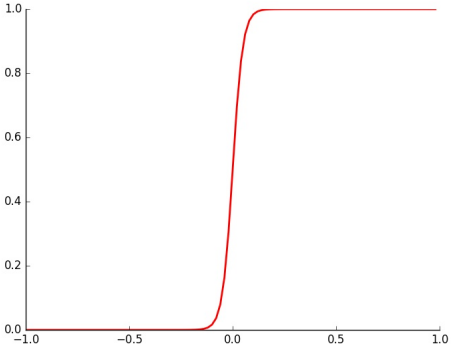
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w

$w = 39, b = 0$



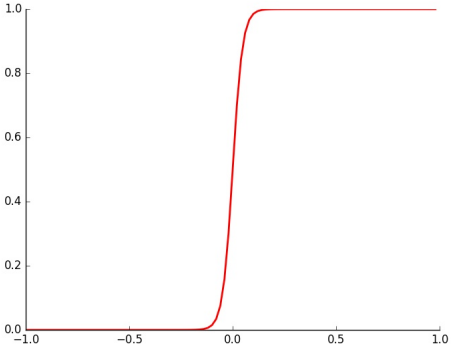
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w

$w = 40, b = 0$



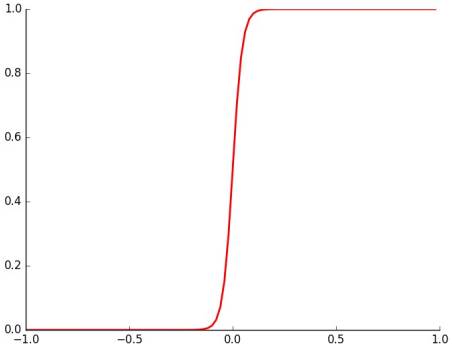
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w

$w = 41, b = 0$



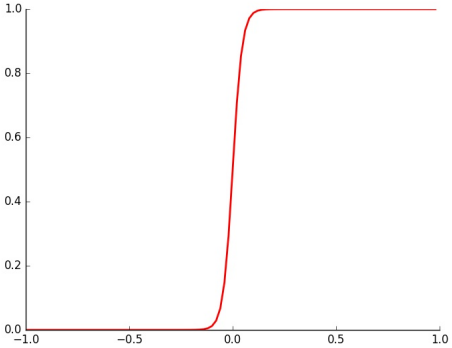
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w

$w = 42, b = 0$



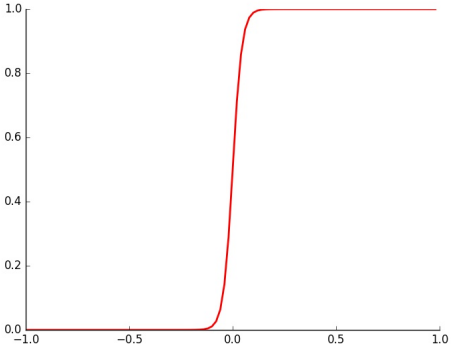
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w

$w = 43, b = 0$



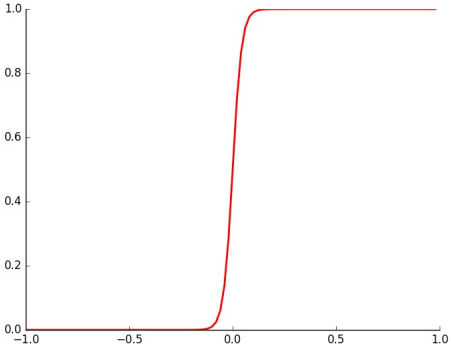
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w

$w = 44, b = 0$



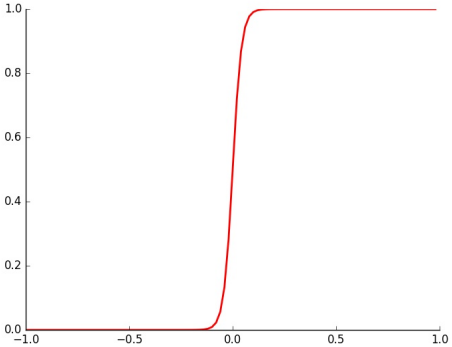
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w

$w = 45, b = 0$



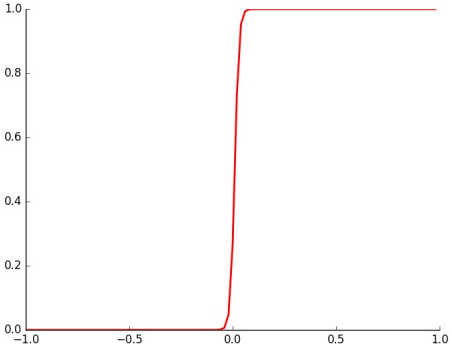
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w

$w = 46, b = 0$



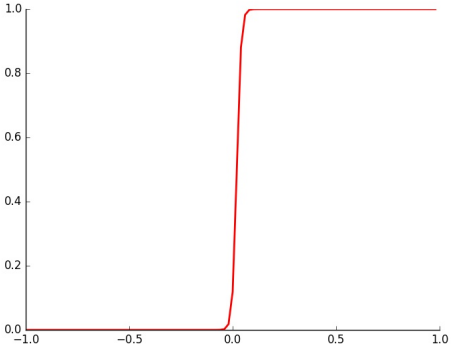
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w

$w = 47, b = 0$



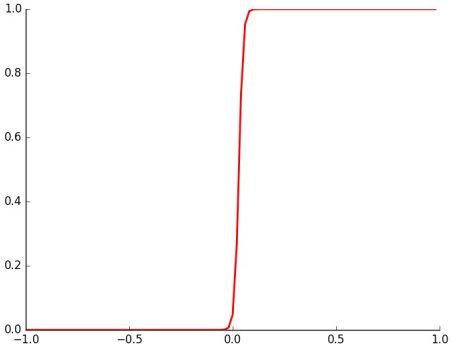
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w
- Further we can adjust the value of b to control the position on the x-axis at which the function transitions from 0 to 1

$w = 50, b = 1$



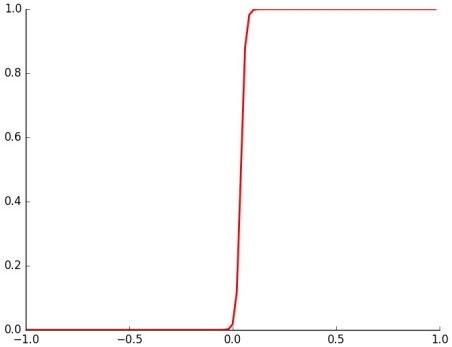
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w
- Further we can adjust the value of b to control the position on the x-axis at which the function transitions from 0 to 1

$w = 50, b = 2$



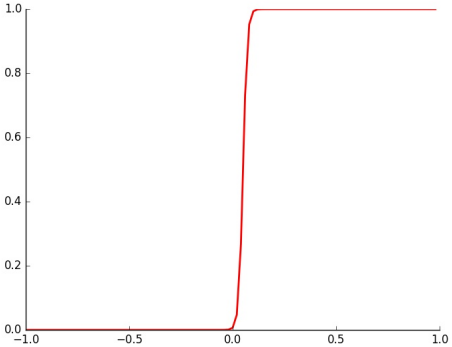
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w
- Further we can adjust the value of b to control the position on the x-axis at which the function transitions from 0 to 1

$w = 50, b = 3$



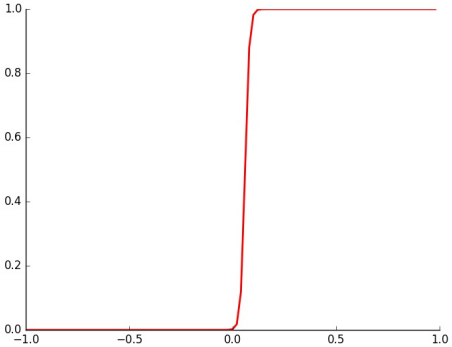
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w
- Further we can adjust the value of b to control the position on the x-axis at which the function transitions from 0 to 1

$w = 50, b = 4$



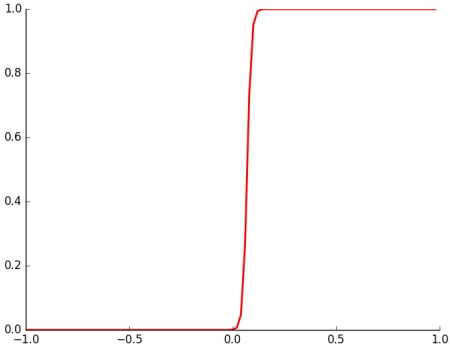
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w
- Further we can adjust the value of b to control the position on the x-axis at which the function transitions from 0 to 1

$w = 50, b = 5$



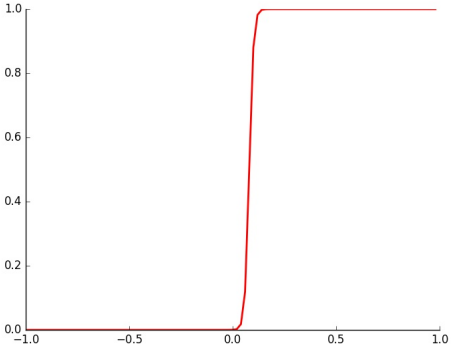
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w
- Further we can adjust the value of b to control the position on the x-axis at which the function transitions from 0 to 1

$w = 50, b = 6$



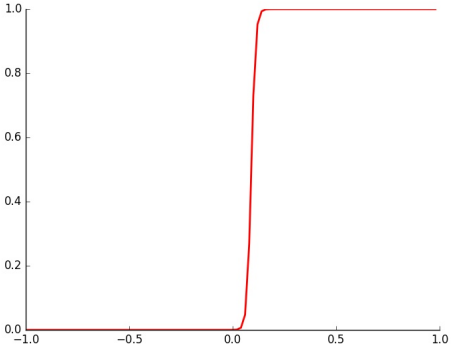
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w
- Further we can adjust the value of b to control the position on the x-axis at which the function transitions from 0 to 1

$w = 50, b = 7$



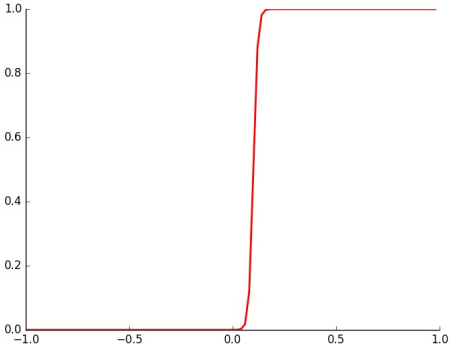
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w
- Further we can adjust the value of b to control the position on the x-axis at which the function transitions from 0 to 1

$w = 50, b = 8$



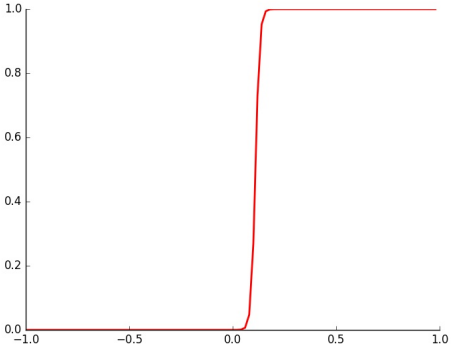
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w
- Further we can adjust the value of b to control the position on the x-axis at which the function transitions from 0 to 1

$w = 50, b = 9$



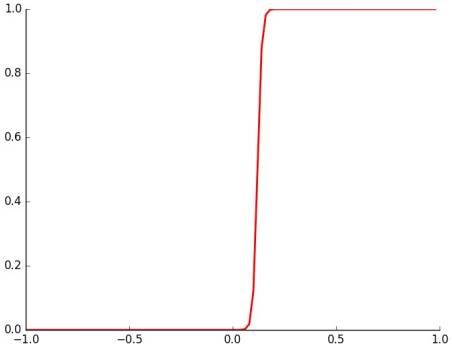
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w
- Further we can adjust the value of b to control the position on the x-axis at which the function transitions from 0 to 1

$w = 50, b = 10$



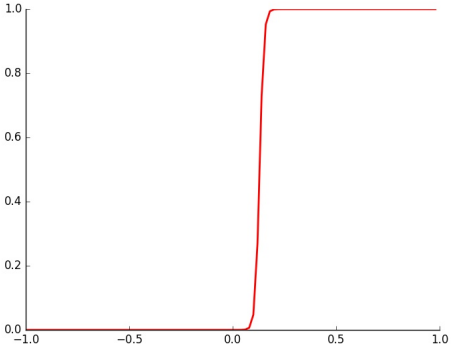
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w
- Further we can adjust the value of b to control the position on the x-axis at which the function transitions from 0 to 1

$w = 50, b = 11$



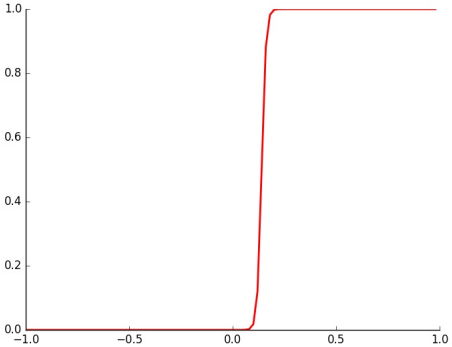
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w
- Further we can adjust the value of b to control the position on the x-axis at which the function transitions from 0 to 1

$w = 50, b = 12$



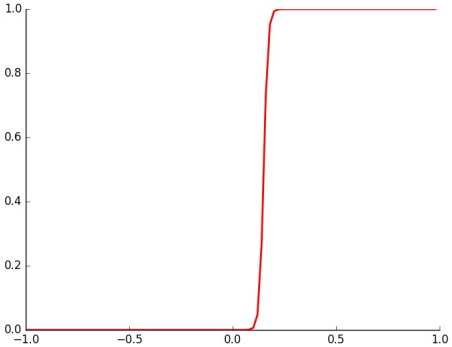
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w
- Further we can adjust the value of b to control the position on the x-axis at which the function transitions from 0 to 1

$w = 50, b = 13$



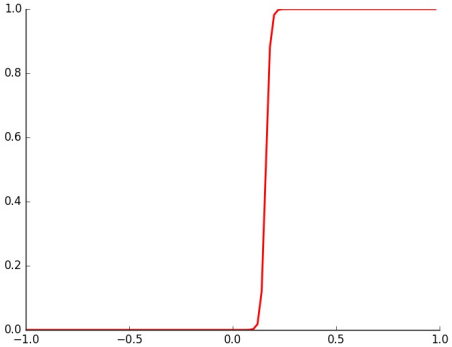
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w
- Further we can adjust the value of b to control the position on the x-axis at which the function transitions from 0 to 1

$w = 50, b = 14$



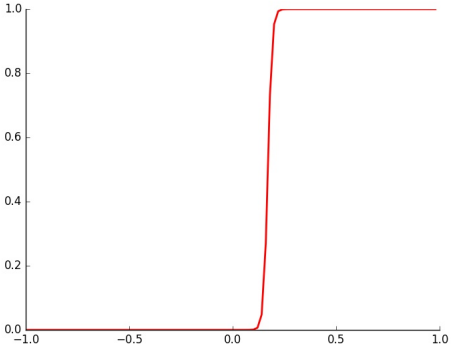
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w
- Further we can adjust the value of b to control the position on the x-axis at which the function transitions from 0 to 1

$w = 50, b = 15$



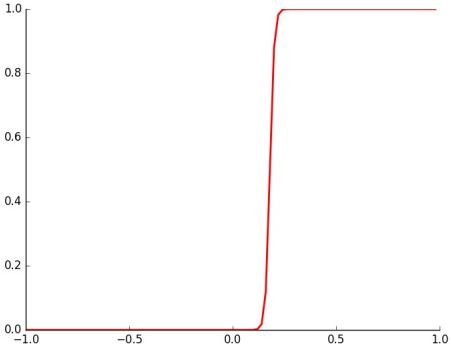
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w
- Further we can adjust the value of b to control the position on the x-axis at which the function transitions from 0 to 1

$w = 50, b = 16$



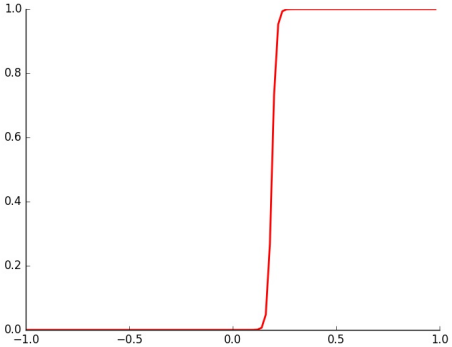
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w
- Further we can adjust the value of b to control the position on the x-axis at which the function transitions from 0 to 1

$w = 50, b = 17$



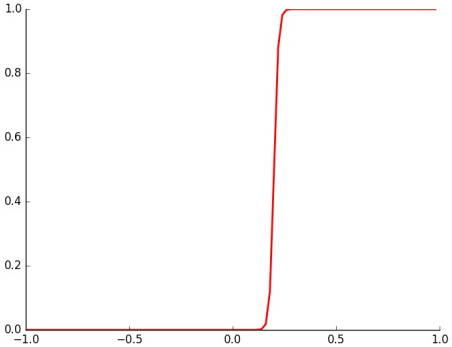
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w
- Further we can adjust the value of b to control the position on the x-axis at which the function transitions from 0 to 1

$w = 50, b = 18$



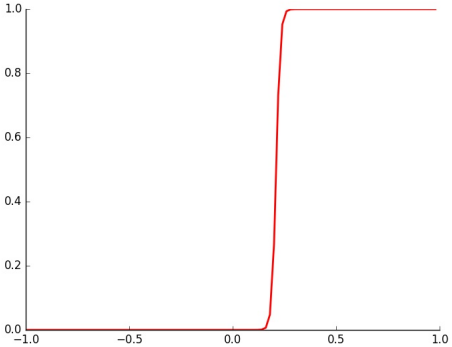
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w
- Further we can adjust the value of b to control the position on the x-axis at which the function transitions from 0 to 1

$w = 50, b = 19$



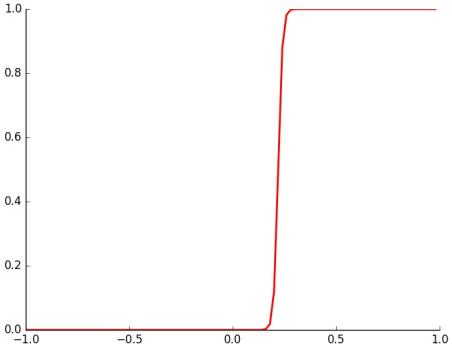
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w
- Further we can adjust the value of b to control the position on the x-axis at which the function transitions from 0 to 1

$w = 50, b = 20$



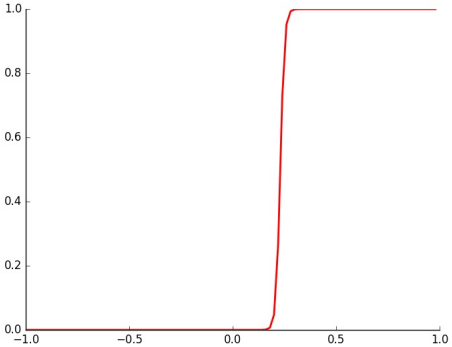
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w
- Further we can adjust the value of b to control the position on the x-axis at which the function transitions from 0 to 1

$w = 50, b = 21$



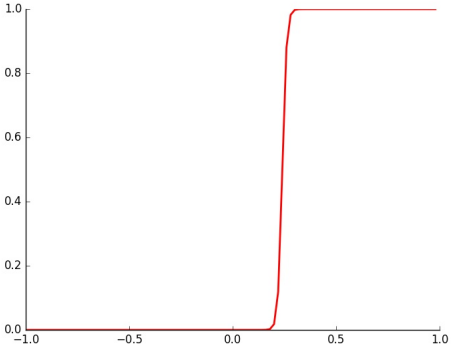
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w
- Further we can adjust the value of b to control the position on the x-axis at which the function transitions from 0 to 1

$w = 50, b = 22$



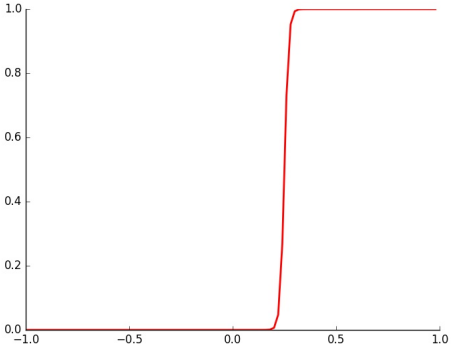
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w
- Further we can adjust the value of b to control the position on the x-axis at which the function transitions from 0 to 1

$w = 50, b = 23$



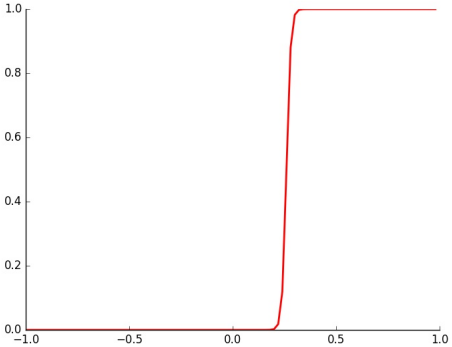
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w
- Further we can adjust the value of b to control the position on the x-axis at which the function transitions from 0 to 1

$w = 50, b = 24$



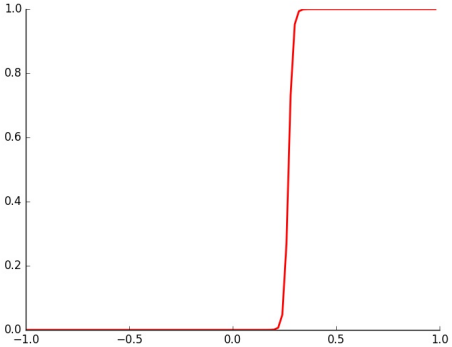
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w
- Further we can adjust the value of b to control the position on the x-axis at which the function transitions from 0 to 1

$w = 50, b = 25$



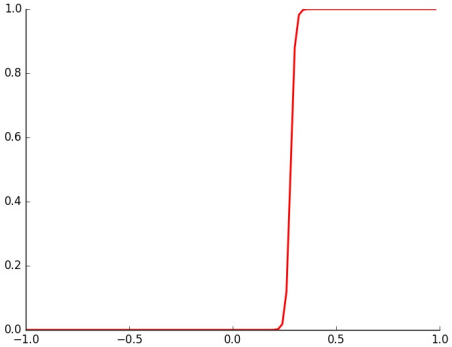
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w
- Further we can adjust the value of b to control the position on the x-axis at which the function transitions from 0 to 1

$w = 50, b = 26$



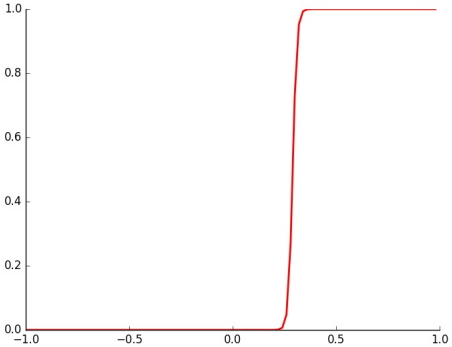
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w
- Further we can adjust the value of b to control the position on the x-axis at which the function transitions from 0 to 1

$w = 50, b = 27$



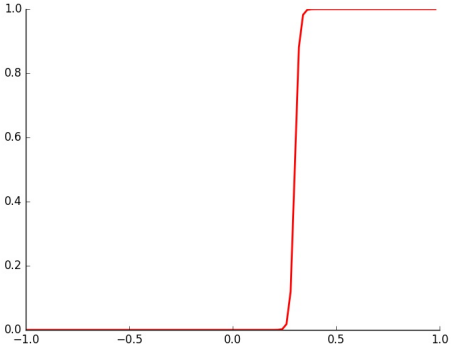
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w
- Further we can adjust the value of b to control the position on the x-axis at which the function transitions from 0 to 1

$w = 50, b = 28$



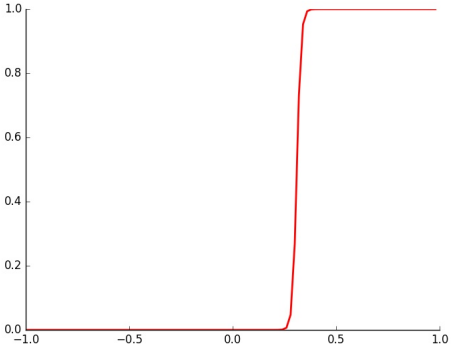
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w
- Further we can adjust the value of b to control the position on the x-axis at which the function transitions from 0 to 1

$w = 50, b = 29$



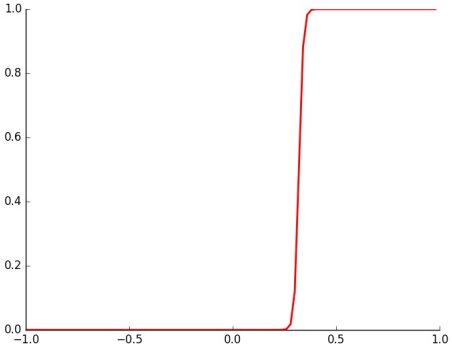
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w
- Further we can adjust the value of b to control the position on the x-axis at which the function transitions from 0 to 1

$w = 50, b = 30$



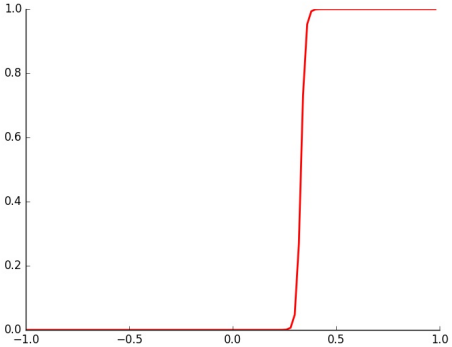
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w
- Further we can adjust the value of b to control the position on the x-axis at which the function transitions from 0 to 1

$w = 50, b = 31$



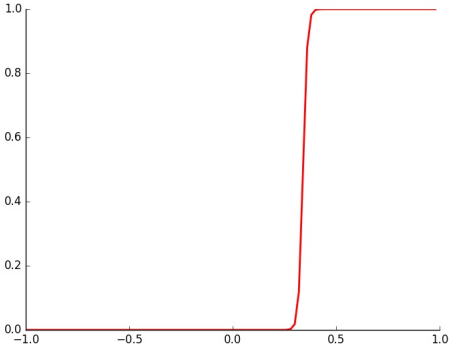
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w
- Further we can adjust the value of b to control the position on the x-axis at which the function transitions from 0 to 1

$w = 50, b = 32$



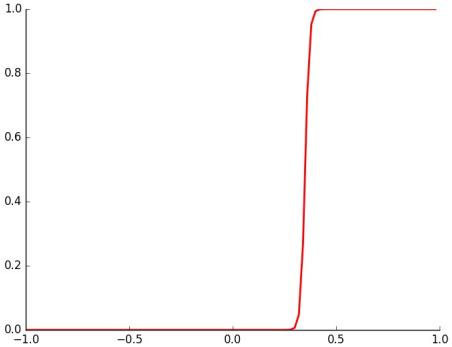
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w
- Further we can adjust the value of b to control the position on the x-axis at which the function transitions from 0 to 1

$w = 50, b = 33$



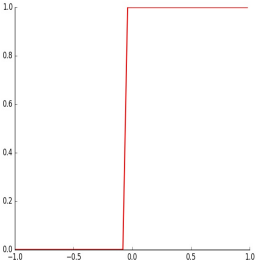
- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w
- Further we can adjust the value of b to control the position on the x-axis at which the function transitions from 0 to 1

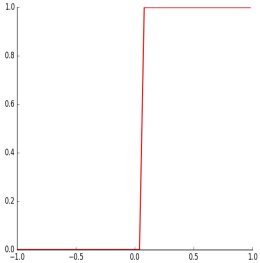
$w = 50, b = 34$



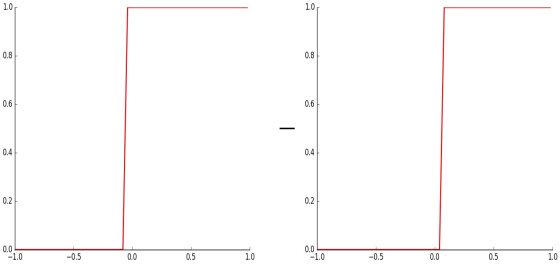
$w = 50, b = 35$

- If we take the logistic function and set w to a very high value we will recover the step function
- Let us see what happens as we change the value of w
- Further we can adjust the value of b to control the position on the x-axis at which the function transitions from 0 to 1

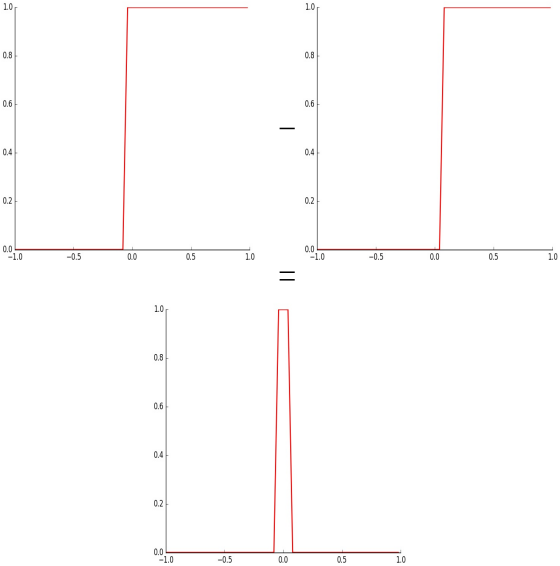




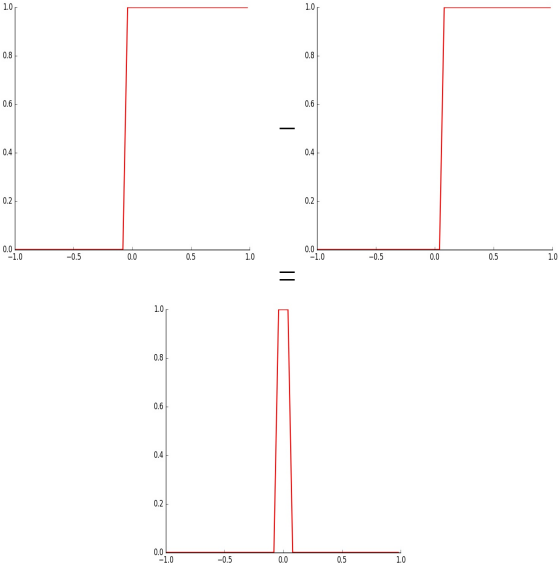
- Now let us see what we get by taking two such sigmoid functions (with different bs) and subtracting one from the other



- Now let us see what we get by taking two such sigmoid functions (with different bs) and subtracting one from the other

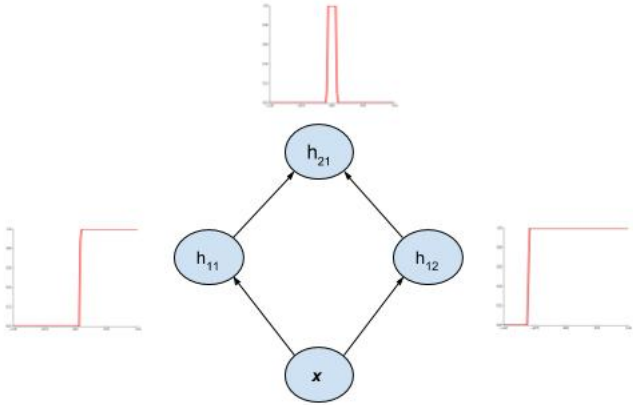


- Now let us see what we get by taking two such sigmoid functions (with different bs) and subtracting one from the other

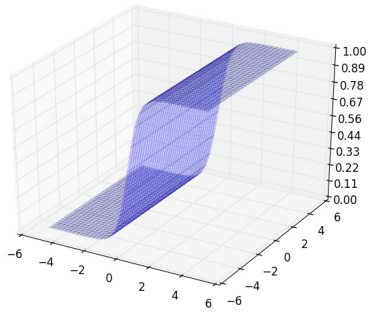


- Now let us see what we get by taking two such sigmoid functions (with different bs) and subtracting one from the other
- Voila! We have our tower function !!

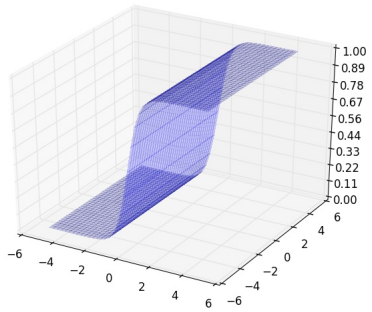
- Can we come up with a neural network to represent this operation of subtracting one sigmoid function from another ?


h21
h11
h12
x

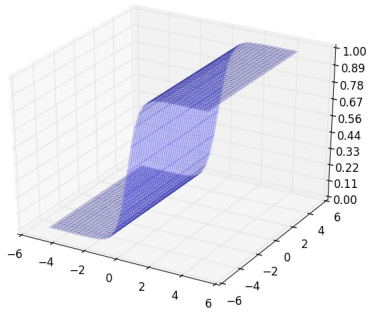
What if we have more than 1 input ?



- This is what a 2-dimensional sigmoid looks like

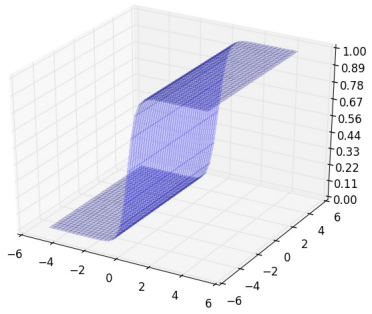


- This is what a 2-dimensional sigmoid looks like
- We need to figure out how to get a 3-dimensional tower



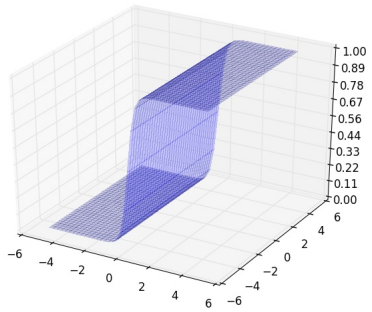
- This is what a 2-dimensional sigmoid looks like
- We need to figure out how to get a 3-dimensional tower
- First, let us set w_2 to 0 and see if we can get a two dimensional step function

$w_1 = 2, w_2 = 0, b = 0$



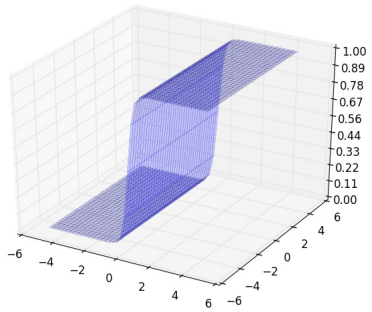
- This is what a 2-dimensional sigmoid looks like
- We need to figure out how to get a 3-dimensional tower
- First, let us set w_2 to 0 and see if we can get a two dimensional step function

$w_1 = 3, w_2 = 0, b = 0$



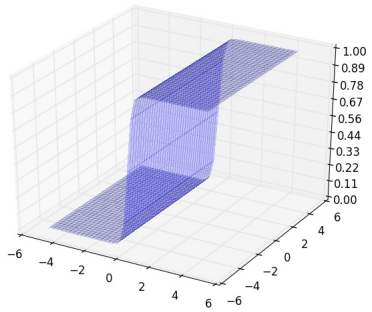
- This is what a 2-dimensional sigmoid looks like
- We need to figure out how to get a 3-dimensional tower
- First, let us set w_2 to 0 and see if we can get a two dimensional step function

$w_1 = 4, w_2 = 0, b = 0$



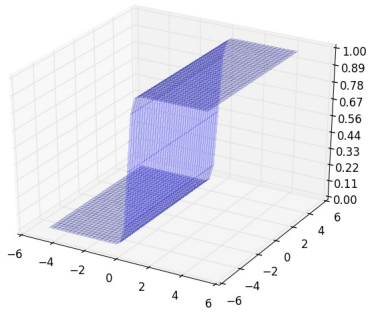
- This is what a 2-dimensional sigmoid looks like
- We need to figure out how to get a 3-dimensional tower
- First, let us set w_2 to 0 and see if we can get a two dimensional step function

$w_1 = 5, w_2 = 0, b = 0$



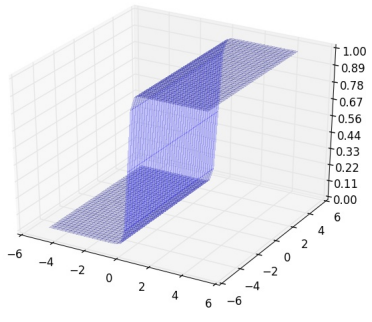
- This is what a 2-dimensional sigmoid looks like
- We need to figure out how to get a 3-dimensional tower
- First, let us set w_2 to 0 and see if we can get a two dimensional step function

$w_1 = 6, w_2 = 0, b = 0$



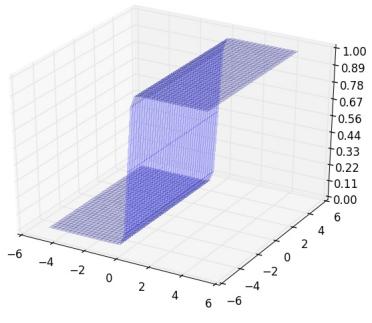
- This is what a 2-dimensional sigmoid looks like
- We need to figure out how to get a 3-dimensional tower
- First, let us set w_2 to 0 and see if we can get a two dimensional step function

$w_1 = 7, w_2 = 0, b = 0$



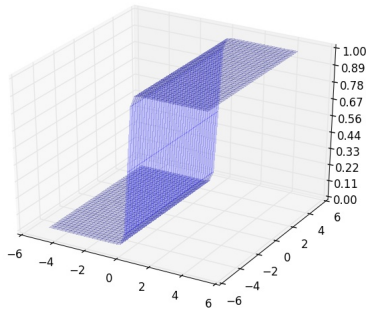
- This is what a 2-dimensional sigmoid looks like
- We need to figure out how to get a 3-dimensional tower
- First, let us set w_2 to 0 and see if we can get a two dimensional step function

$w_1 = 8, w_2 = 0, b = 0$



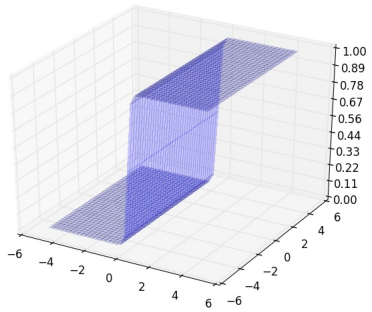
- This is what a 2-dimensional sigmoid looks like
- We need to figure out how to get a 3-dimensional tower
- First, let us set w_2 to 0 and see if we can get a two dimensional step function

$w_1 = 9, w_2 = 0, b = 0$



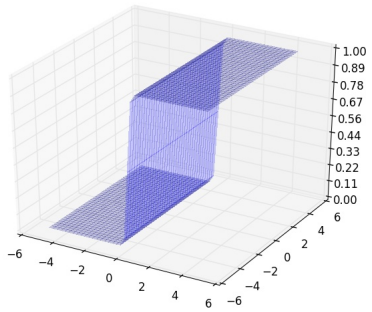
- This is what a 2-dimensional sigmoid looks like
- We need to figure out how to get a 3-dimensional tower
- First, let us set w_2 to 0 and see if we can get a two dimensional step function

$w_1 = 10, w_2 = 0, b = 0$



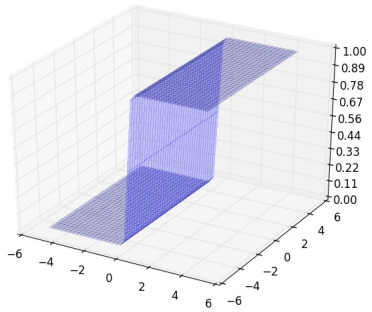
- This is what a 2-dimensional sigmoid looks like
- We need to figure out how to get a 3-dimensional tower
- First, let us set w_2 to 0 and see if we can get a two dimensional step function

$w_1 = 11, w_2 = 0, b = 0$



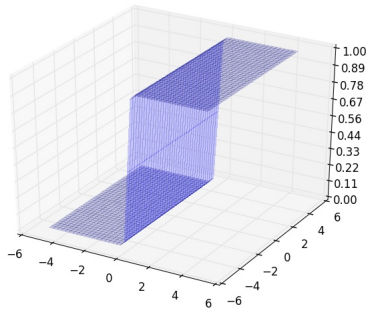
- This is what a 2-dimensional sigmoid looks like
- We need to figure out how to get a 3-dimensional tower
- First, let us set w_2 to 0 and see if we can get a two dimensional step function

$w_1 = 12, w_2 = 0, b = 0$



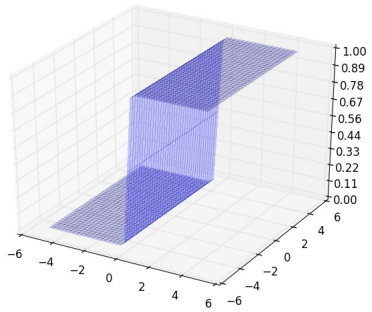
- This is what a 2-dimensional sigmoid looks like
- We need to figure out how to get a 3-dimensional tower
- First, let us set w_2 to 0 and see if we can get a two dimensional step function

$w_1 = 13, w_2 = 0, b = 0$



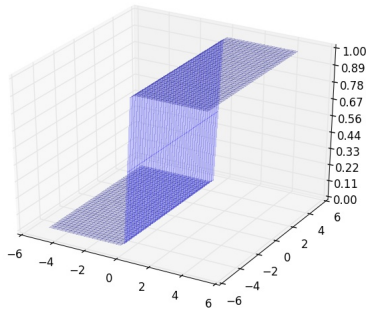
- This is what a 2-dimensional sigmoid looks like
- We need to figure out how to get a 3-dimensional tower
- First, let us set w_2 to 0 and see if we can get a two dimensional step function

$w_1 = 14, w_2 = 0, b = 0$



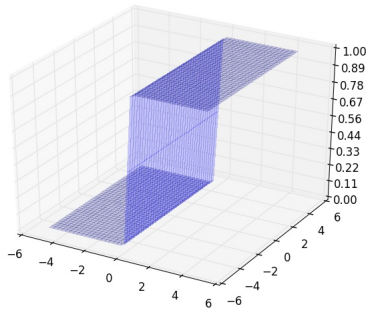
- This is what a 2-dimensional sigmoid looks like
- We need to figure out how to get a 3-dimensional tower
- First, let us set w_2 to 0 and see if we can get a two dimensional step function

$w_1 = 15, w_2 = 0, b = 0$



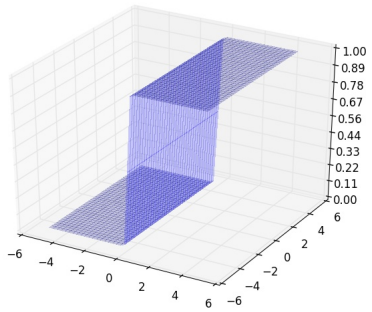
- This is what a 2-dimensional sigmoid looks like
- We need to figure out how to get a 3-dimensional tower
- First, let us set w_2 to 0 and see if we can get a two dimensional step function

$w_1 = 16, w_2 = 0, b = 0$



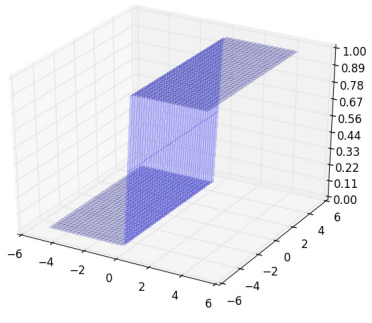
- This is what a 2-dimensional sigmoid looks like
- We need to figure out how to get a 3-dimensional tower
- First, let us set w_2 to 0 and see if we can get a two dimensional step function

$w_1 = 17, w_2 = 0, b = 0$



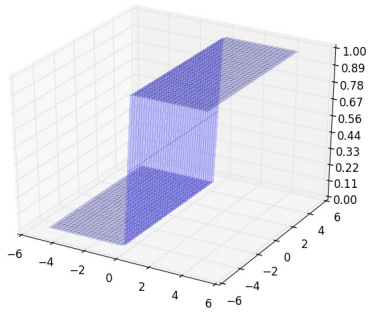
- This is what a 2-dimensional sigmoid looks like
- We need to figure out how to get a 3-dimensional tower
- First, let us set w_2 to 0 and see if we can get a two dimensional step function

$w_1 = 18, w_2 = 0, b = 0$



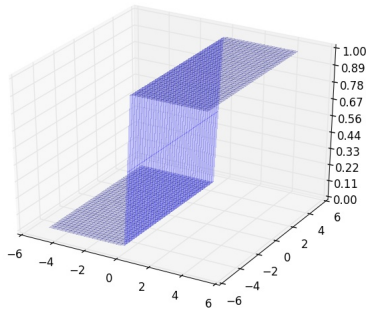
- This is what a 2-dimensional sigmoid looks like
- We need to figure out how to get a 3-dimensional tower
- First, let us set w_2 to 0 and see if we can get a two dimensional step function

$w_1 = 19, w_2 = 0, b = 0$



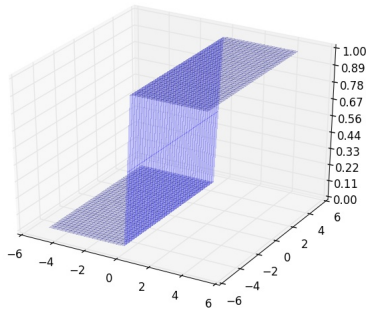
- This is what a 2-dimensional sigmoid looks like
- We need to figure out how to get a 3-dimensional tower
- First, let us set w_2 to 0 and see if we can get a two dimensional step function

$w_1 = 20, w_2 = 0, b = 0$



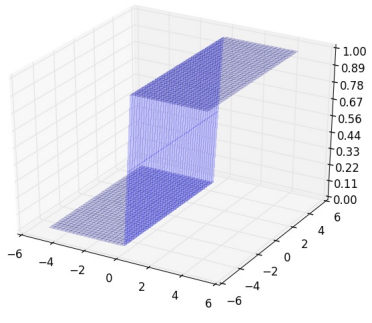
- This is what a 2-dimensional sigmoid looks like
- We need to figure out how to get a 3-dimensional tower
- First, let us set w_2 to 0 and see if we can get a two dimensional step function

$w_1 = 21, w_2 = 0, b = 0$



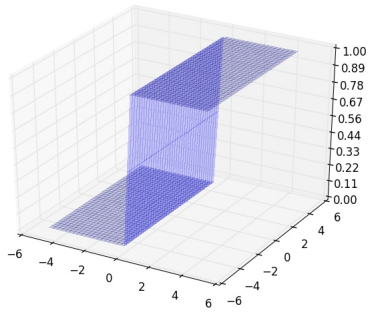
- This is what a 2-dimensional sigmoid looks like
- We need to figure out how to get a 3-dimensional tower
- First, let us set w_2 to 0 and see if we can get a two dimensional step function

$w_1 = 22, w_2 = 0, b = 0$



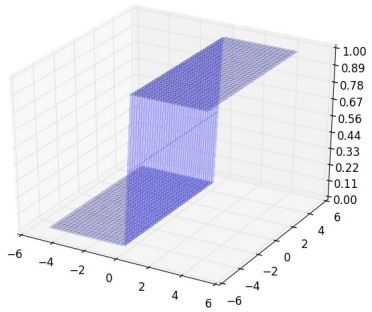
- This is what a 2-dimensional sigmoid looks like
- We need to figure out how to get a 3-dimensional tower
- First, let us set w_2 to 0 and see if we can get a two dimensional step function

$w_1 = 23, w_2 = 0, b = 0$

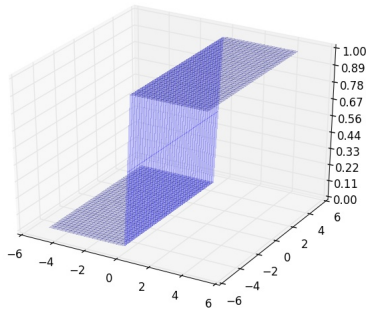


$w_1 = 24, w_2 = 0, b = 0$

- This is what a 2-dimensional sigmoid looks like
- We need to figure out how to get a 3-dimensional tower
- First, let us set w_2 to 0 and see if we can get a two dimensional step function

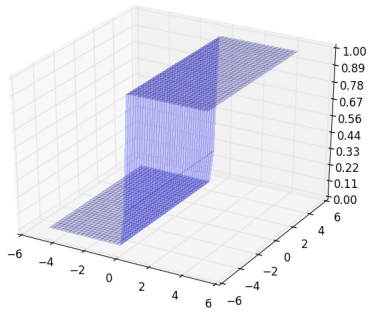


- This is what a 2-dimensional sigmoid looks like
- We need to figure out how to get a 3-dimensional tower
- First, let us set w_2 to 0 and see if we can get a two dimensional step function
- What would happen if we change b ?



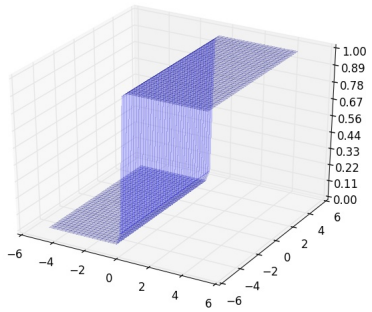
- This is what a 2-dimensional sigmoid looks like
- We need to figure out how to get a 3-dimensional tower
- First, let us set w_2 to 0 and see if we can get a two dimensional step function
- What would happen if we change b ?

$w_1 = 25, w_2 = 0, b = 0$



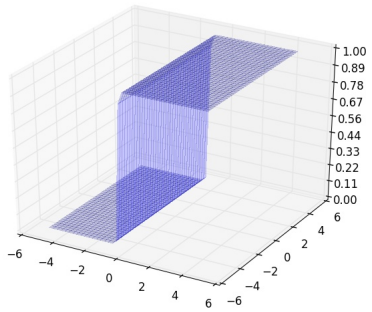
- This is what a 2-dimensional sigmoid looks like
- We need to figure out how to get a 3-dimensional tower
- First, let us set w_2 to 0 and see if we can get a two dimensional step function
- What would happen if we change b ?

$w_1 = 25, w_2 = 0, b = 5$



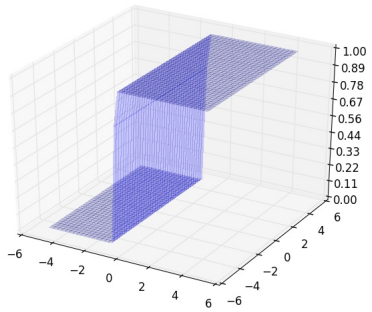
- This is what a 2-dimensional sigmoid looks like
- We need to figure out how to get a 3-dimensional tower
- First, let us set w_2 to 0 and see if we can get a two dimensional step function
- What would happen if we change b ?

$w_1 = 25, w_2 = 0, b = 10$



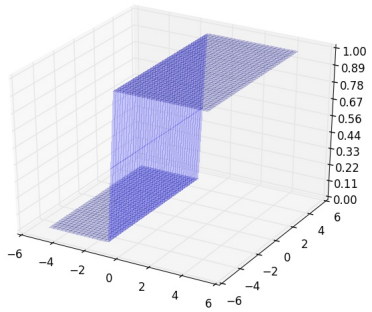
- This is what a 2-dimensional sigmoid looks like
- We need to figure out how to get a 3-dimensional tower
- First, let us set w_2 to 0 and see if we can get a two dimensional step function
- What would happen if we change b ?

$w_1 = 25, w_2 = 0, b = 15$



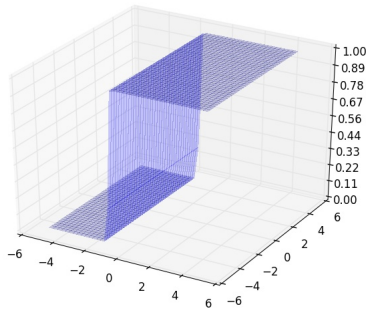
- This is what a 2-dimensional sigmoid looks like
- We need to figure out how to get a 3-dimensional tower
- First, let us set w_2 to 0 and see if we can get a two dimensional step function
- What would happen if we change b ?

$w_1 = 25, w_2 = 0, b = 20$



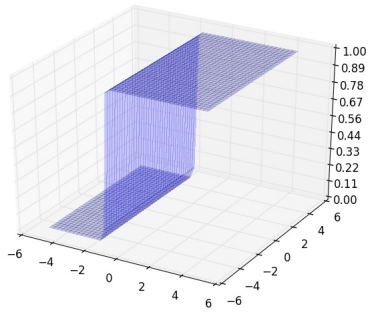
- This is what a 2-dimensional sigmoid looks like
- We need to figure out how to get a 3-dimensional tower
- First, let us set w_2 to 0 and see if we can get a two dimensional step function
- What would happen if we change b ?

$w_1 = 25, w_2 = 0, b = 25$



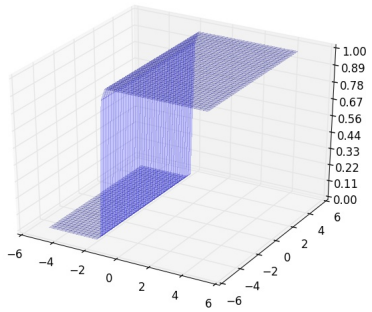
- This is what a 2-dimensional sigmoid looks like
- We need to figure out how to get a 3-dimensional tower
- First, let us set w_2 to 0 and see if we can get a two dimensional step function
- What would happen if we change b ?

$w_1 = 25, w_2 = 0, b = 30$



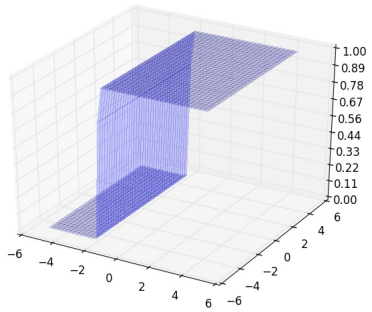
- This is what a 2-dimensional sigmoid looks like
- We need to figure out how to get a 3-dimensional tower
- First, let us set w_2 to 0 and see if we can get a two dimensional step function
- What would happen if we change b ?

$w_1 = 25, w_2 = 0, b = 35$



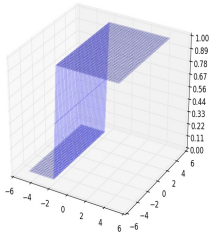
- This is what a 2-dimensional sigmoid looks like
- We need to figure out how to get a 3-dimensional tower
- First, let us set w_2 to 0 and see if we can get a two dimensional step function
- What would happen if we change b ?

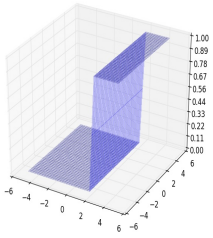
$w_1 = 25, w_2 = 0, b = 40$



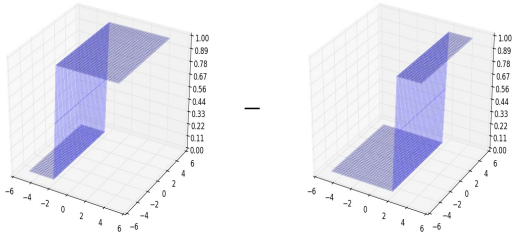
$w_1 = 25, w_2 = 0, b = 45$

- This is what a 2-dimensional sigmoid looks like
- We need to figure out how to get a 3-dimensional tower
- First, let us set w_2 to 0 and see if we can get a two dimensional step function
- What would happen if we change b ?

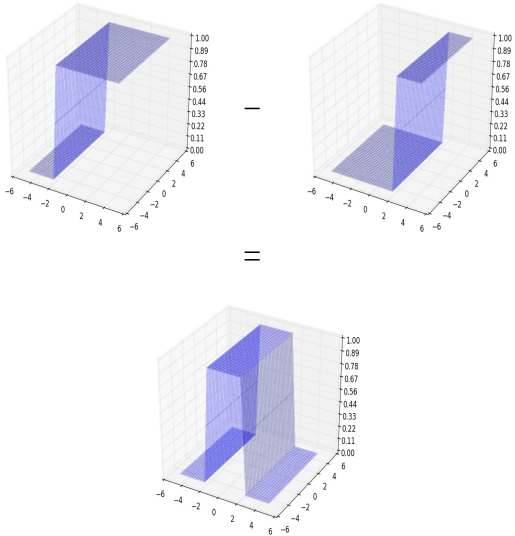




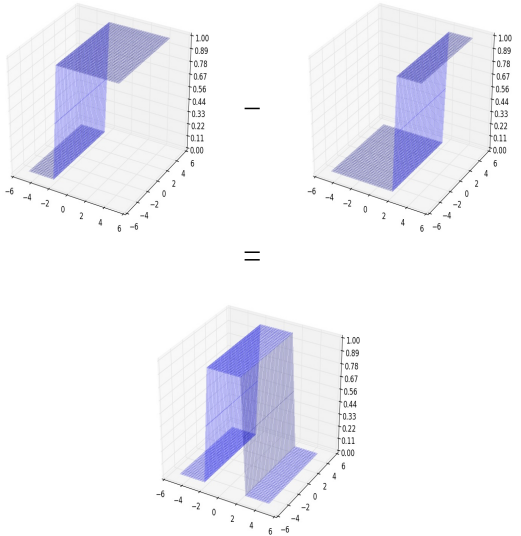
- What if we take two such step functions (with different b values) and subtract one from the other



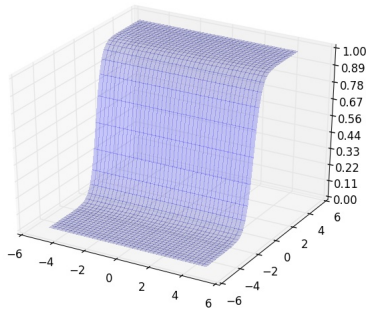
- What if we take two such step functions (with different b values) and subtract one from the other



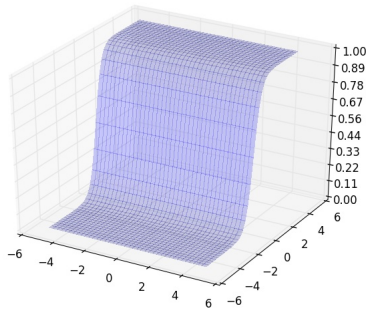
- What if we take two such step functions (with different b values) and subtract one from the other



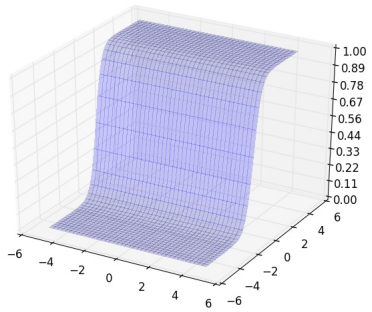
- What if we take two such step functions (with different b values) and subtract one from the other
- We still don't get a tower (or we get a tower which is open from two sides)



- Now let us set w_1 to 0 and adjust w_2 to get a 3-dimensional step function with a different orientation

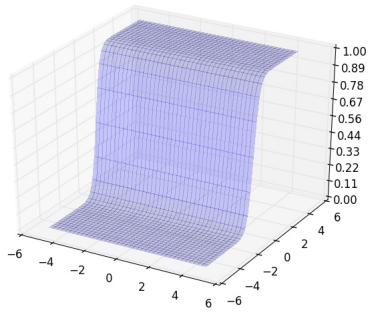


- Now let us set w_1 to 0 and adjust w_2 to get a 3-dimensional step function with a different orientation



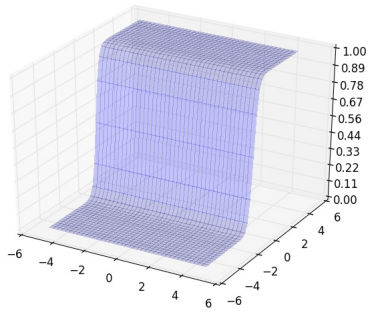
- Now let us set w_1 to 0 and adjust w_2 to get a 3-dimensional step function with a different orientation

$w_1 = 0, w_2 = 2, b = 0$



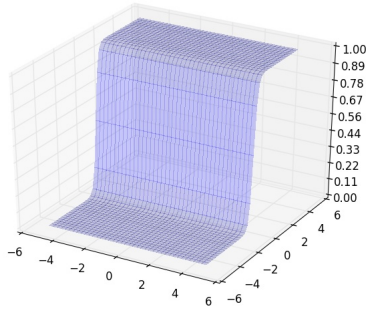
- Now let us set w_1 to 0 and adjust w_2 to get a 3-dimensional step function with a different orientation

$w_1 = 0, w_2 = 3, b = 0$



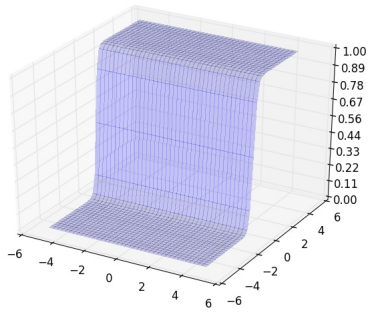
- Now let us set w_1 to 0 and adjust w_2 to get a 3-dimensional step function with a different orientation

$w_1 = 0, w_2 = 4, b = 0$



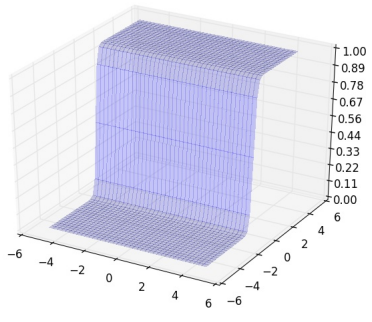
- Now let us set w_1 to 0 and adjust w_2 to get a 3-dimensional step function with a different orientation

$w_1 = 0, w_2 = 5, b = 0$



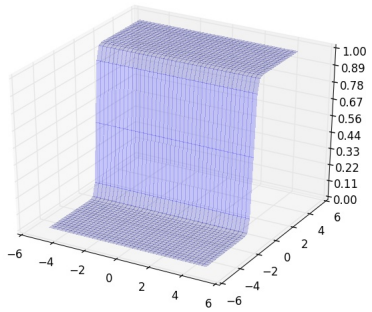
- Now let us set w_1 to 0 and adjust w_2 to get a 3-dimensional step function with a different orientation

$w_1 = 0, w_2 = 6, b = 0$



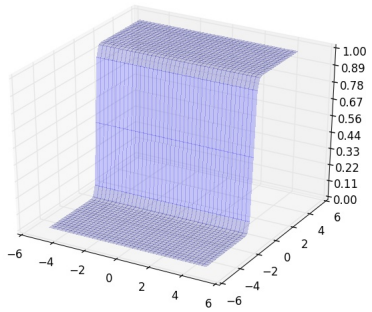
- Now let us set w_1 to 0 and adjust w_2 to get a 3-dimensional step function with a different orientation

$w_1 = 0, w_2 = 7, b = 0$



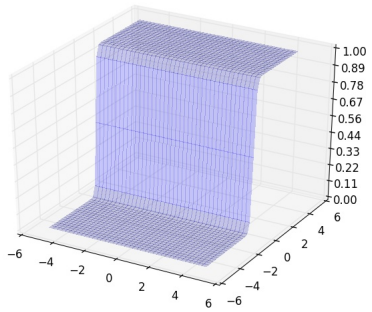
- Now let us set w_1 to 0 and adjust w_2 to get a 3-dimensional step function with a different orientation

$w_1 = 0, w_2 = 8, b = 0$



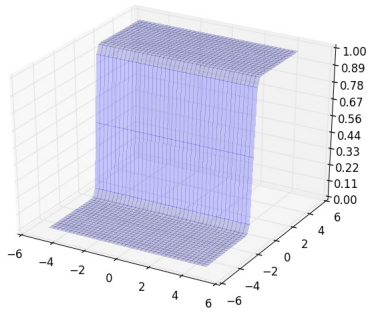
- Now let us set w_1 to 0 and adjust w_2 to get a 3-dimensional step function with a different orientation

$w_1 = 0, w_2 = 9, b = 0$



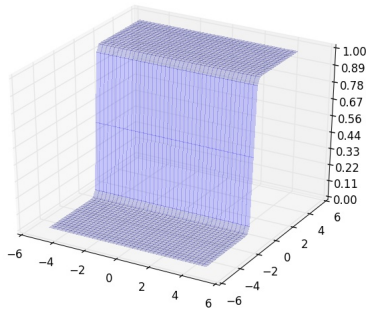
- Now let us set w_1 to 0 and adjust w_2 to get a 3-dimensional step function with a different orientation

$w_1 = 0, w_2 = 10, b = 0$



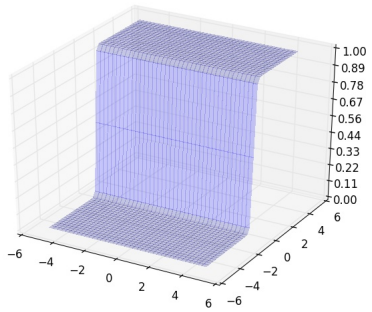
- Now let us set w_1 to 0 and adjust w_2 to get a 3-dimensional step function with a different orientation

$w_1 = 0, w_2 = 11, b = 0$



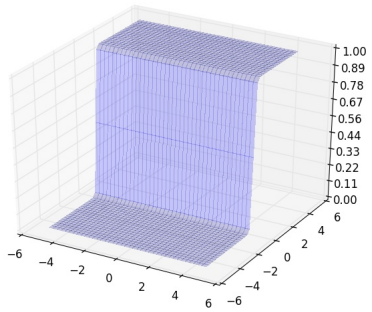
- Now let us set w_1 to 0 and adjust w_2 to get a 3-dimensional step function with a different orientation

$w_1 = 0, w_2 = 12, b = 0$



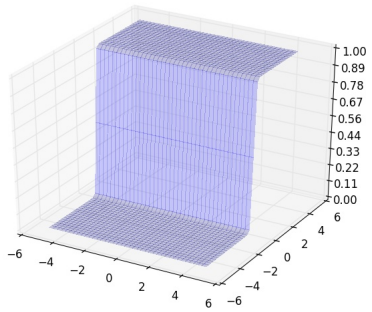
- Now let us set w_1 to 0 and adjust w_2 to get a 3-dimensional step function with a different orientation

$w_1 = 0, w_2 = 13, b = 0$



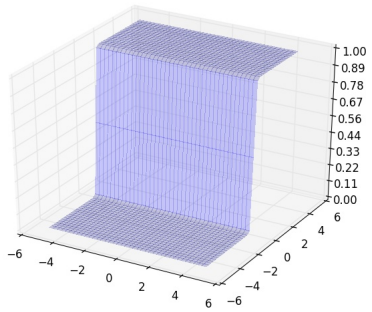
- Now let us set w_1 to 0 and adjust w_2 to get a 3-dimensional step function with a different orientation

$w_1 = 0, w_2 = 14, b = 0$



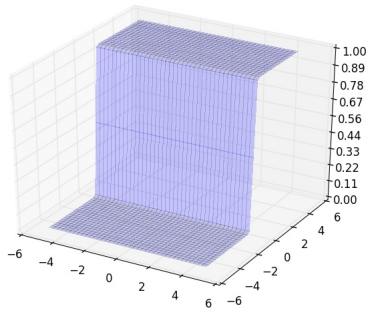
- Now let us set w_1 to 0 and adjust w_2 to get a 3-dimensional step function with a different orientation

$w_1 = 0, w_2 = 15, b = 0$



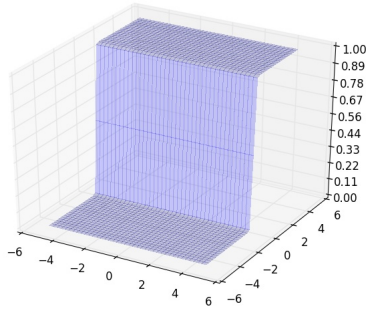
- Now let us set w_1 to 0 and adjust w_2 to get a 3-dimensional step function with a different orientation

$w_1 = 0, w_2 = 16, b = 0$



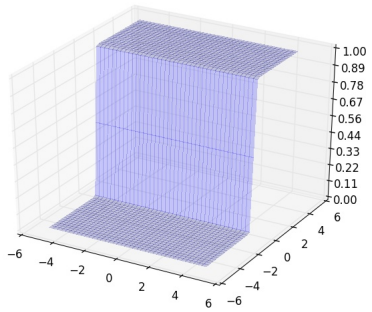
- Now let us set w_1 to 0 and adjust w_2 to get a 3-dimensional step function with a different orientation

$w_1 = 0, w_2 = 17, b = 0$



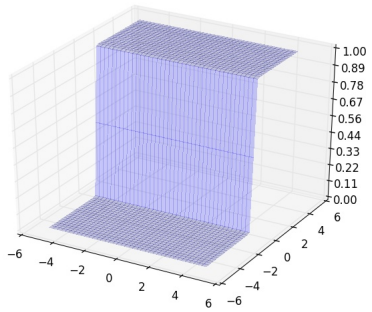
- Now let us set w_1 to 0 and adjust w_2 to get a 3-dimensional step function with a different orientation

$w_1 = 0, w_2 = 18, b = 0$



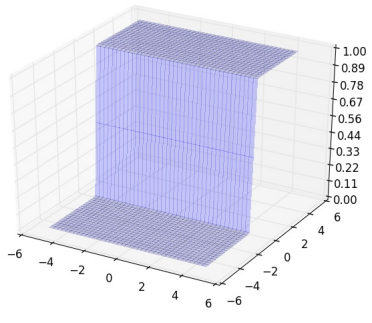
- Now let us set w_1 to 0 and adjust w_2 to get a 3-dimensional step function with a different orientation

$w_1 = 0, w_2 = 19, b = 0$



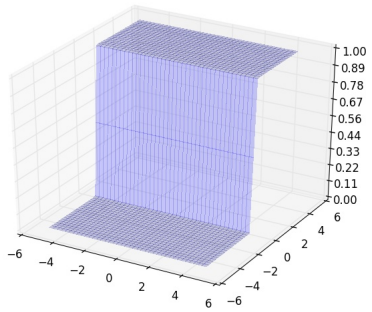
- Now let us set w_1 to 0 and adjust w_2 to get a 3-dimensional step function with a different orientation

$w_1 = 0, w_2 = 20, b = 0$



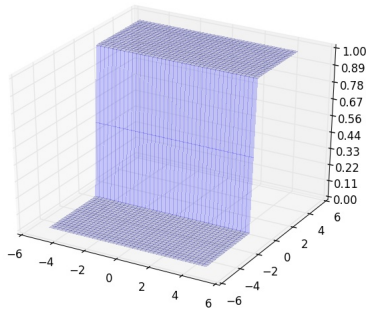
- Now let us set w_1 to 0 and adjust w_2 to get a 3-dimensional step function with a different orientation

$w_1 = 0, w_2 = 21, b = 0$



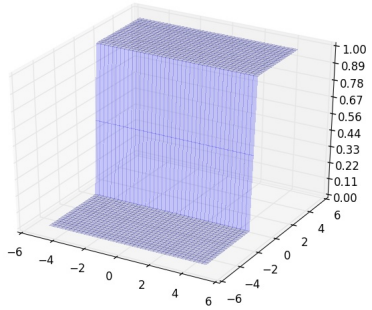
- Now let us set w_1 to 0 and adjust w_2 to get a 3-dimensional step function with a different orientation

$w_1 = 0, w_2 = 22, b = 0$



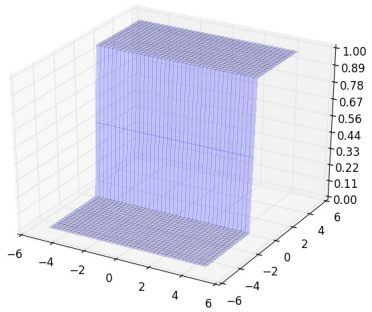
- Now let us set w_1 to 0 and adjust w_2 to get a 3-dimensional step function with a different orientation

$w_1 = 0, w_2 = 23, b = 0$

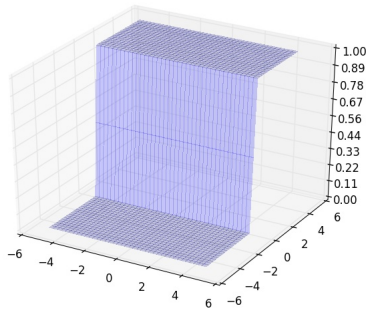


$w_1 = 0, w_2 = 24, b = 0$

- Now let us set w_1 to 0 and adjust w_2 to get a 3-dimensional step function with a different orientation

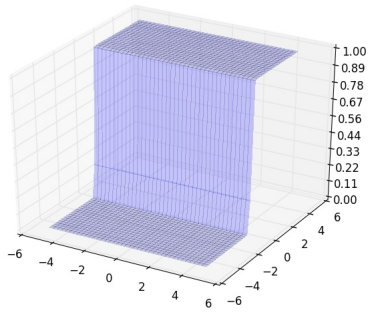


- Now let us set w_1 to 0 and adjust w_2 to get a 3-dimensional step function with a different orientation
- And now we change b



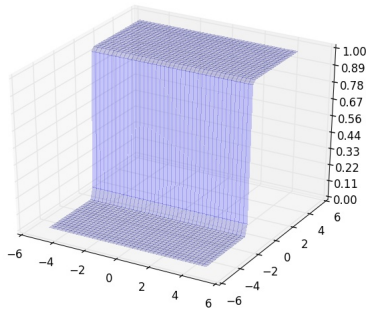
- Now let us set w_1 to 0 and adjust w_2 to get a 3-dimensional step function with a different orientation
- And now we change b

$w_1 = 0, w_2 = 25, b = 0$



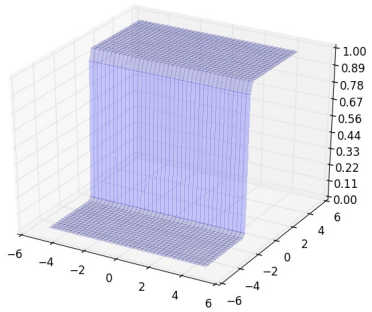
- Now let us set w_1 to 0 and adjust w_2 to get a 3-dimensional step function with a different orientation
- And now we change b

$w_1 = 0, w_2 = 25, b = 5$



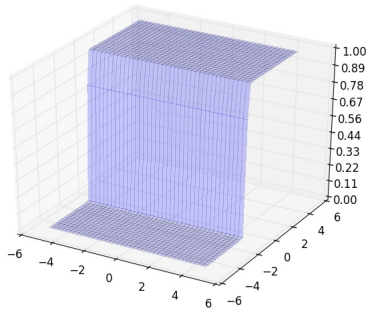
- Now let us set w_1 to 0 and adjust w_2 to get a 3-dimensional step function with a different orientation
- And now we change b

$w_1 = 0, w_2 = 25, b = 10$



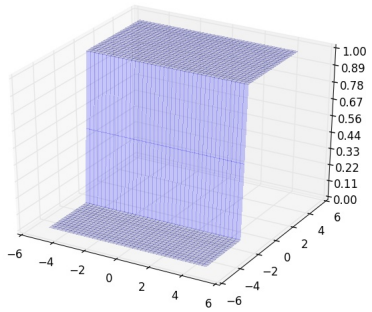
- Now let us set w_1 to 0 and adjust w_2 to get a 3-dimensional step function with a different orientation
- And now we change b

$w_1 = 0, w_2 = 25, b = 15$



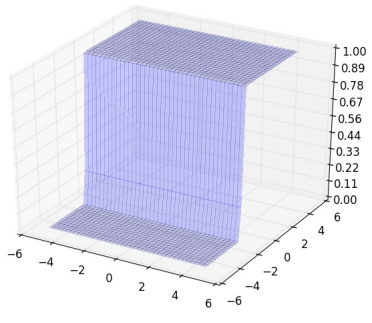
- Now let us set w_1 to 0 and adjust w_2 to get a 3-dimensional step function with a different orientation
- And now we change b

$w_1 = 0, w_2 = 25, b = 20$



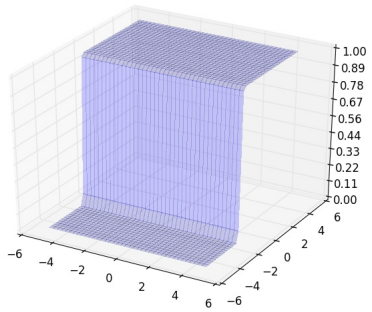
- Now let us set w_1 to 0 and adjust w_2 to get a 3-dimensional step function with a different orientation
- And now we change b

$w_1 = 0, w_2 = 25, b = 25$



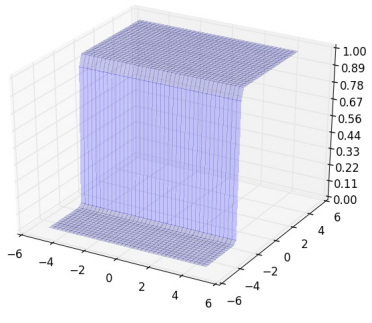
- Now let us set w_1 to 0 and adjust w_2 to get a 3-dimensional step function with a different orientation
- And now we change b

$w_1 = 0, w_2 = 25, b = 30$



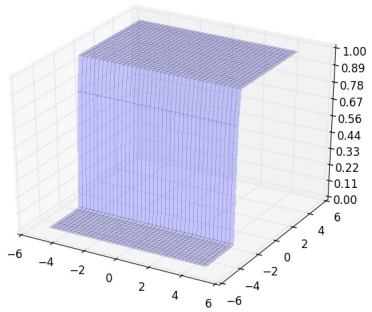
- Now let us set w_1 to 0 and adjust w_2 to get a 3-dimensional step function with a different orientation
- And now we change b

$w_1 = 0, w_2 = 25, b = 35$



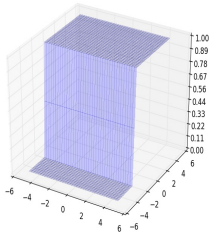
- Now let us set w_1 to 0 and adjust w_2 to get a 3-dimensional step function with a different orientation
- And now we change b

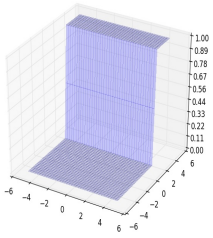
$w_1 = 0, w_2 = 25, b = 40$



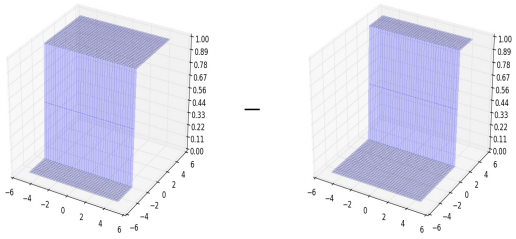
$w_1 = 0, w_2 = 25, b = 45$

- Now let us set w_1 to 0 and adjust w_2 to get a 3-dimensional step function with a different orientation
- And now we change b

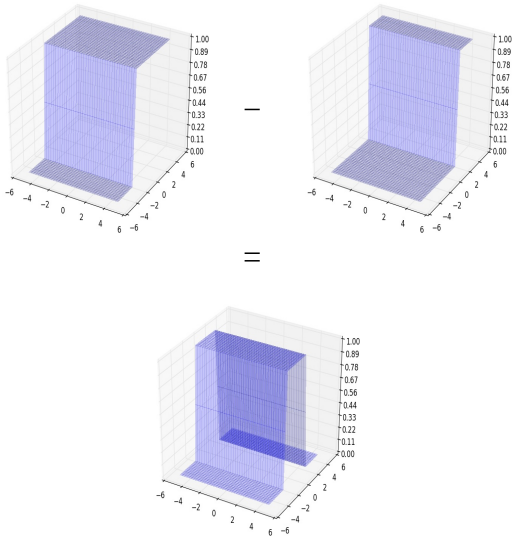




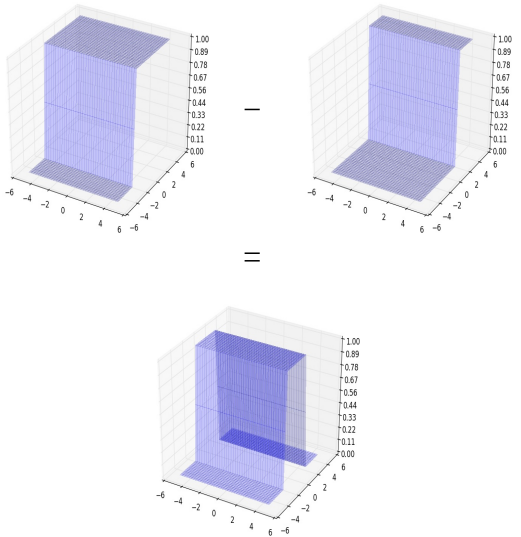
- Again, what if we take two such step functions (with different b values) and subtract one from the other



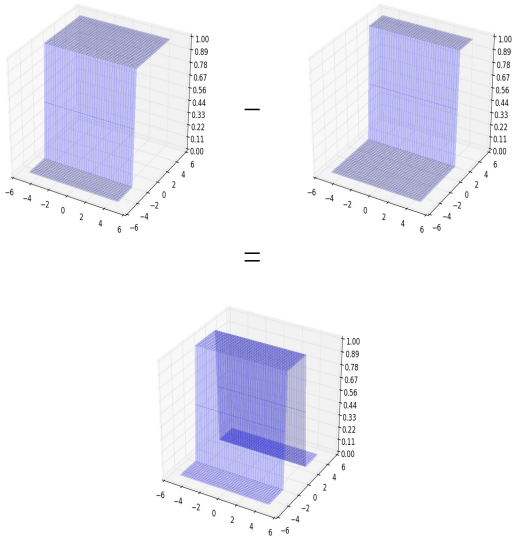
- Again, what if we take two such step functions (with different b values) and subtract one from the other



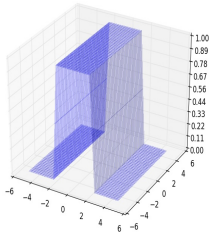
- Again, what if we take two such step functions (with different b values) and subtract one from the other

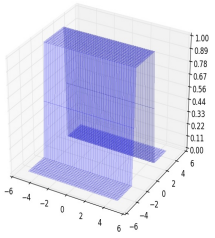


- Again, what if we take two such step functions (with different b values) and subtract one from the other
- We still don't get a tower (or we get a tower which is open from two sides)

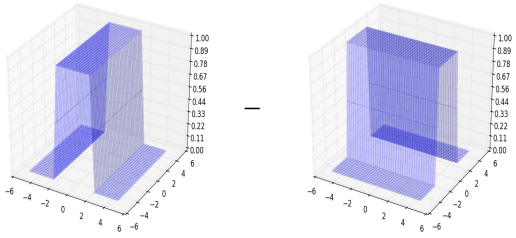


- Again, what if we take two such step functions (with different b values) and subtract one from the other
- We still don't get a tower (or we get a tower which is open from two sides)
- Notice that this open tower has a different orientation from the previous one

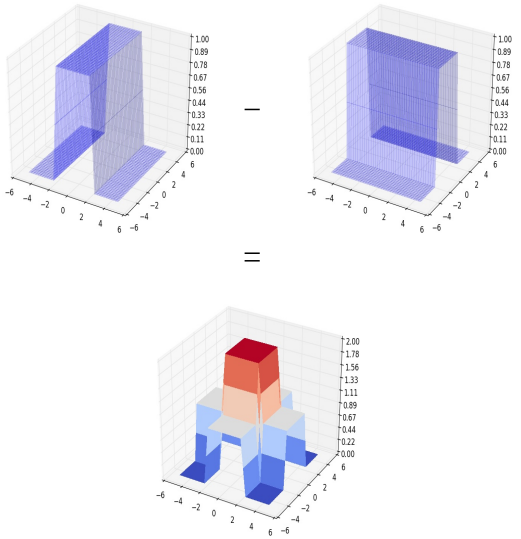




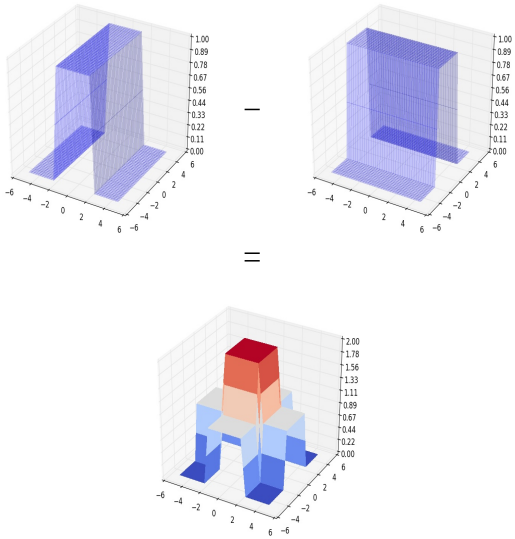
- Now what will we get by adding two such open towers ?



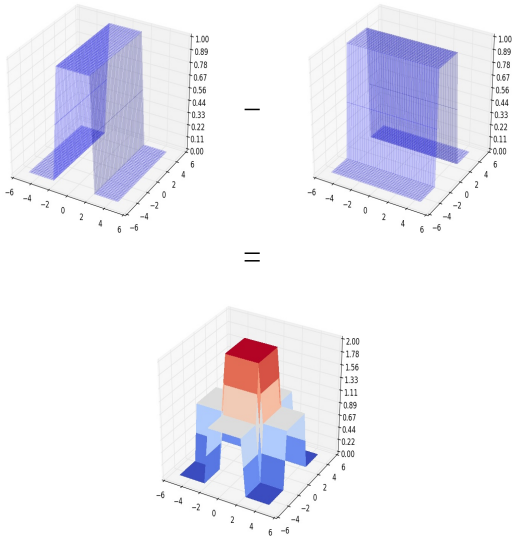
- Now what will we get by adding two such open towers ?



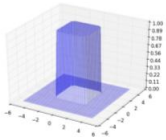
- Now what will we get by adding two such open towers ?

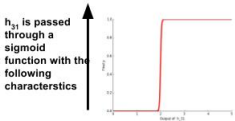


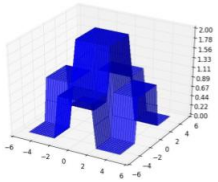
- Now what will we get by adding two such open towers ?
- We get a tower standing on an elevated base



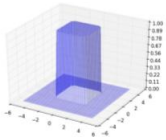
- Now what will we get by adding two such open towers ?
- We get a tower standing on an elevated base
- We can now pass this output through another sigmoid neuron to get the desired tower !

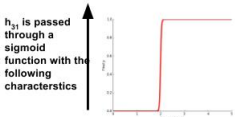


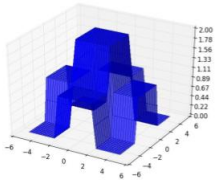




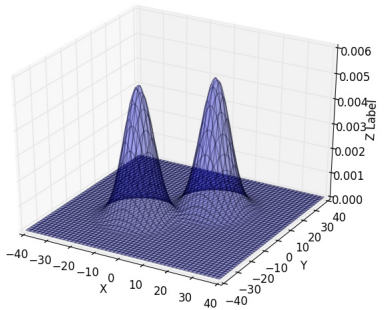
- Now what will we get by adding two such open towers ?
- We get a tower standing on an elevated base
- We can now pass this output through another sigmoid neuron to get the desired tower !





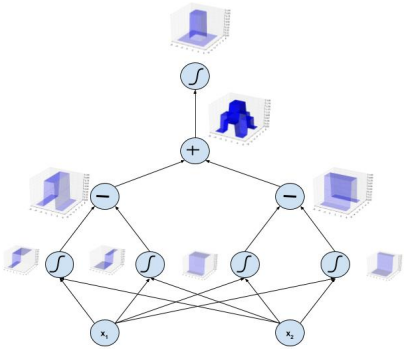


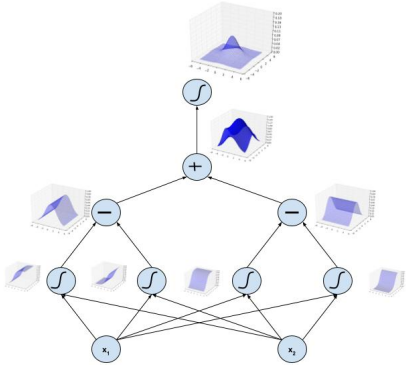
- Now what will we get by adding two such open towers ?
- We get a tower standing on an elevated base
- We can now pass this output through another sigmoid neuron to get the desired tower !
- We can now approximate any function by summing up many such towers



- For example, we could approximate the following function using a sum of several towers

- Can we come up with a neural network to represent this entire procedure of constructing a 3 dimensional tower ?


x1
x2

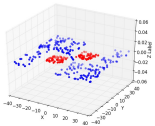


Think

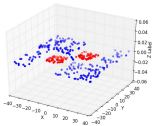
- For 1 dimensional input we needed 2 neurons to construct a tower
- For 2 dimensional input we needed 4 neurons to construct a tower
- How many neurons will you need to construct a tower in n dimensions ?

Time to retrospect

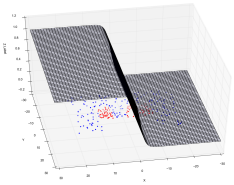
- Why do we care about approximating any arbitrary function ?
- Can we tie all this back to the classification problem that we have been dealing with ?



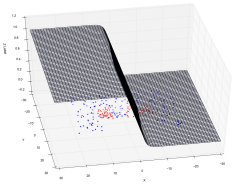
- We are interested in separating the blue points from the red points



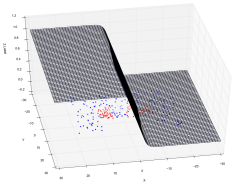
- We are interested in separating the blue points from the red points
- Suppose we use a single sigmoidal neuron to approximate the relation between $x = [x_1, x_2]$ and y



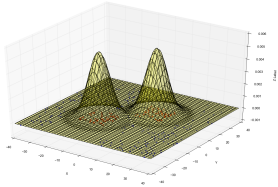
- We are interested in separating the blue points from the red points
- Suppose we use a single sigmoidal neuron to approximate the relation between $x = [x_1, x_2]$ and y



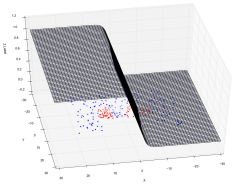
- We are interested in separating the blue points from the red points
- Suppose we use a single sigmoidal neuron to approximate the relation between $x = [x_1, x_2]$ and y
- Obviously, there will be errors (some blue points get classified as 1 and some red points get classified as 0)



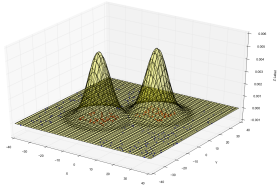
- We are interested in separating the blue points from the red points
- Suppose we use a single sigmoidal neuron to approximate the relation between $x = [x_1, x_2]$ and y
- Obviously, there will be errors (some blue points get classified as 1 and some red points get classified as 0)



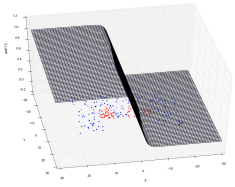
- This is what we actually want



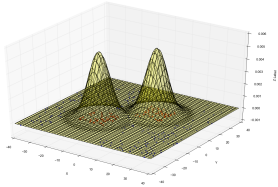
- We are interested in separating the blue points from the red points
- Suppose we use a single sigmoidal neuron to approximate the relation between $x = [x_1, x_2]$ and y
- Obviously, there will be errors (some blue points get classified as 1 and some red points get classified as 0)



- This is what we actually want
- The illustrative proof that we just saw tells us that we can have a neural network with two hidden layers which can approximate the above function by a sum of towers



- We are interested in separating the blue points from the red points
- Suppose we use a single sigmoidal neuron to approximate the relation between $x = [x_1, x_2]$ and y
- Obviously, there will be errors (some blue points get classified as 1 and some red points get classified as 0)



- This is what we actually want
- The illustrative proof that we just saw tells us that we can have a neural network with two hidden layers which can approximate the above function by a sum of towers
- Which means we can have a neural network which can exactly separate the blue points from the red points !!