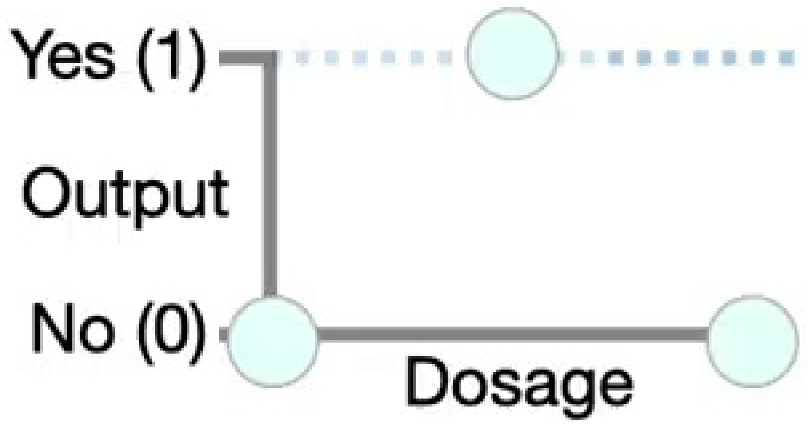
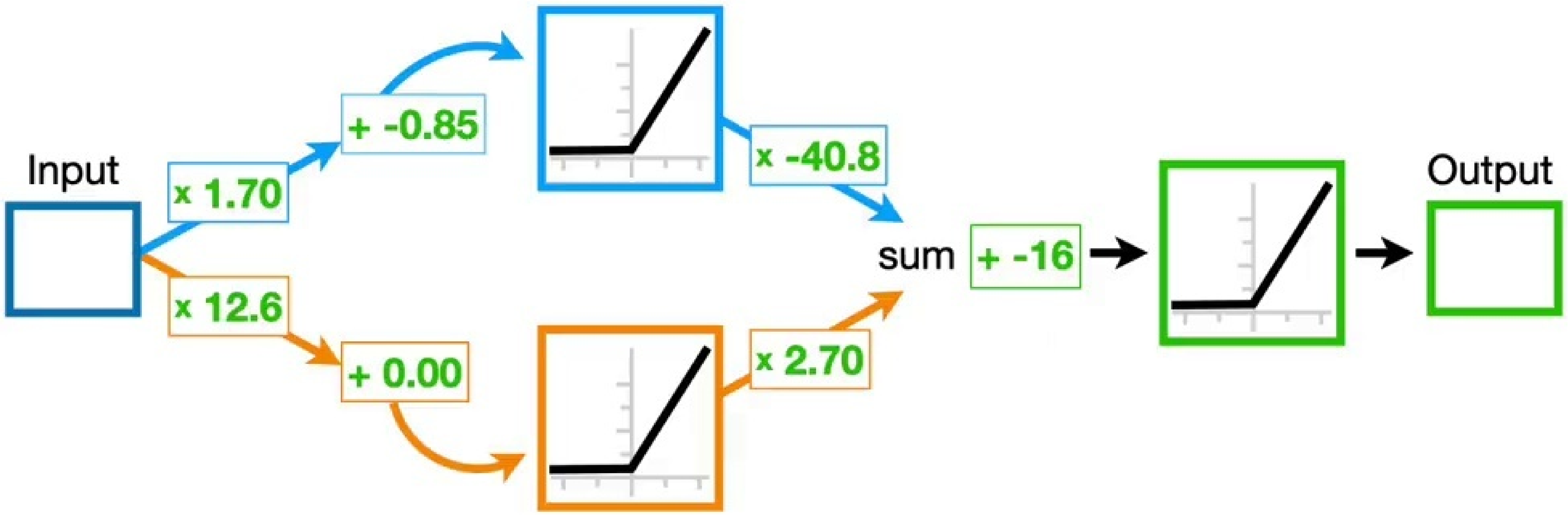
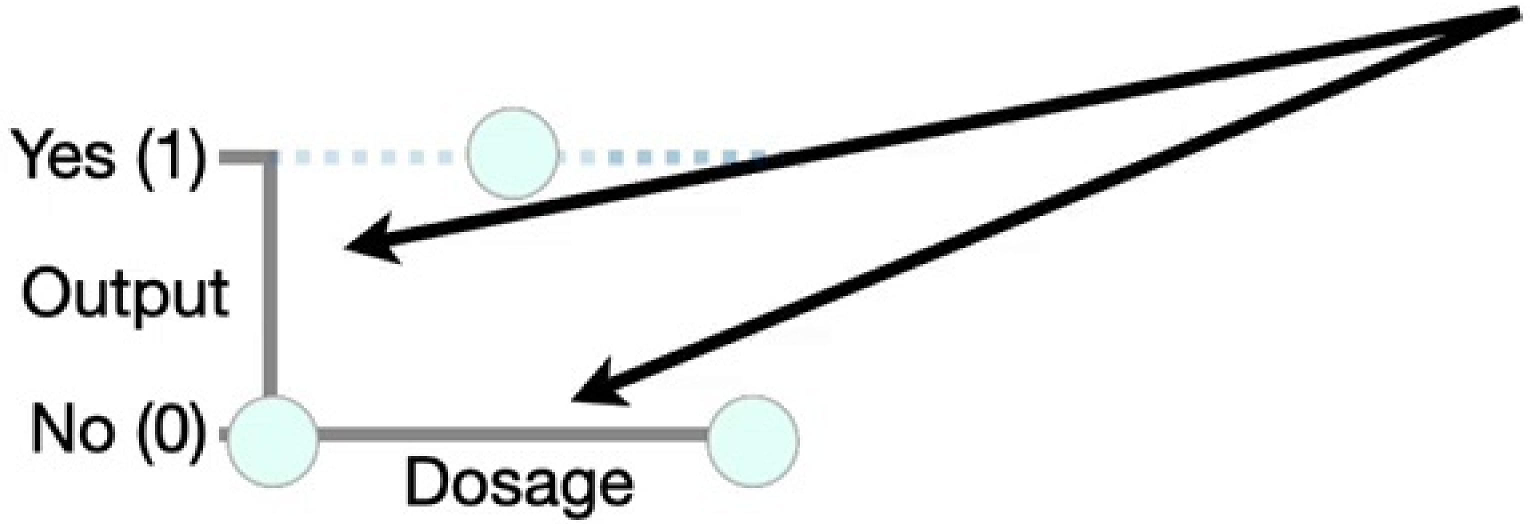
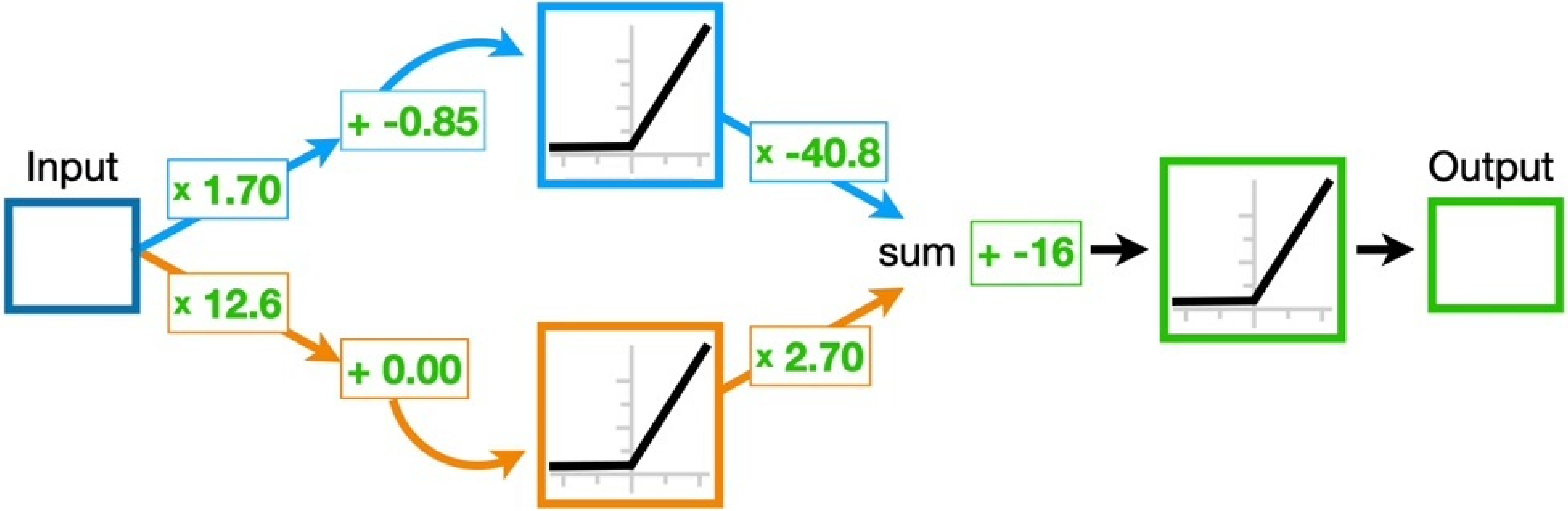
So far the **Neural Networks** that we've looked at have been super simple…



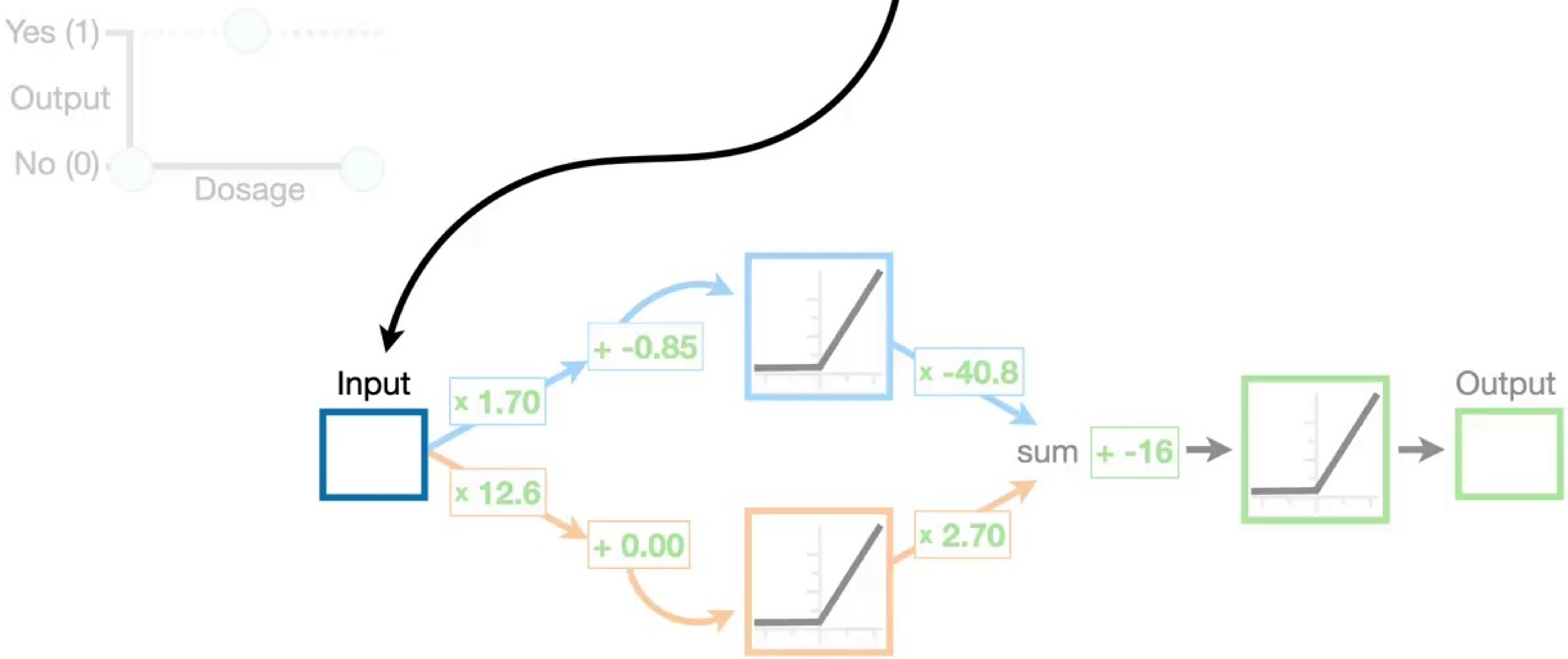


…and only predict whether or not the **Dosage** of a drug will be **Effective**.

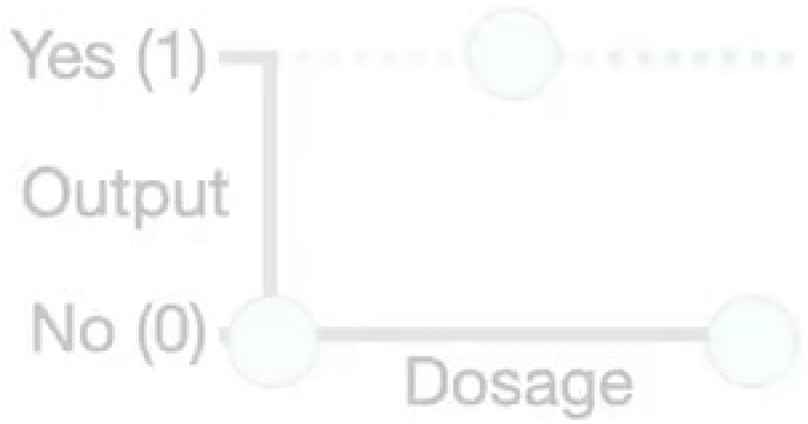


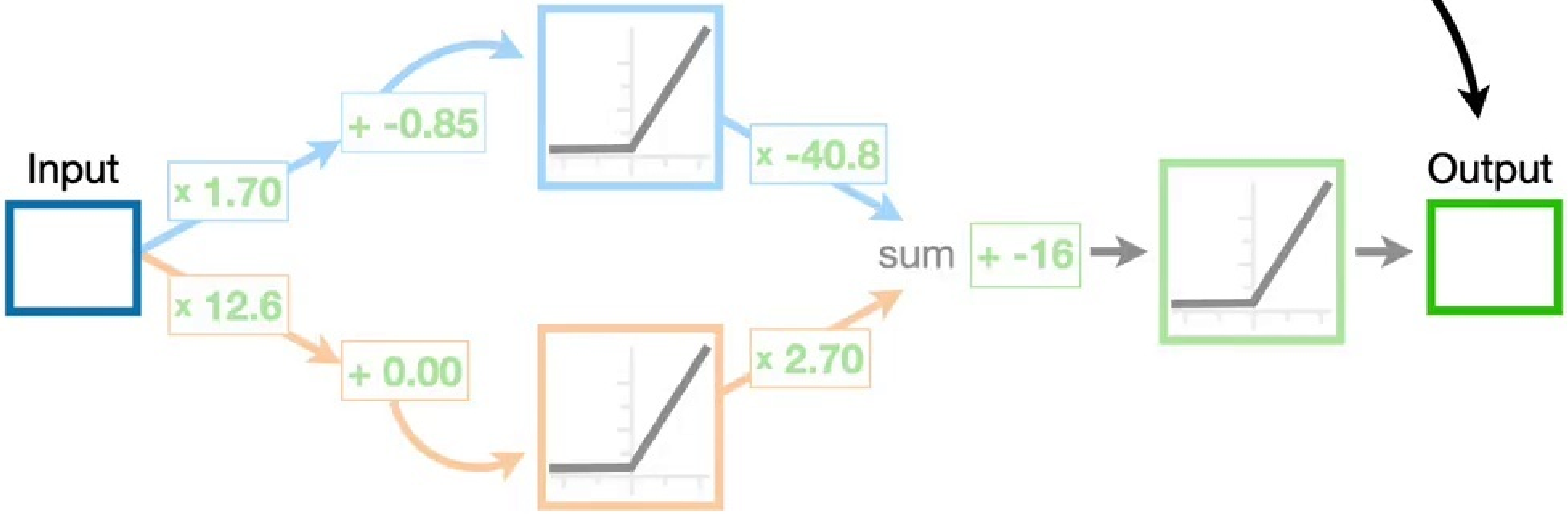


The **Neural Networks** just have one **Input Node**…

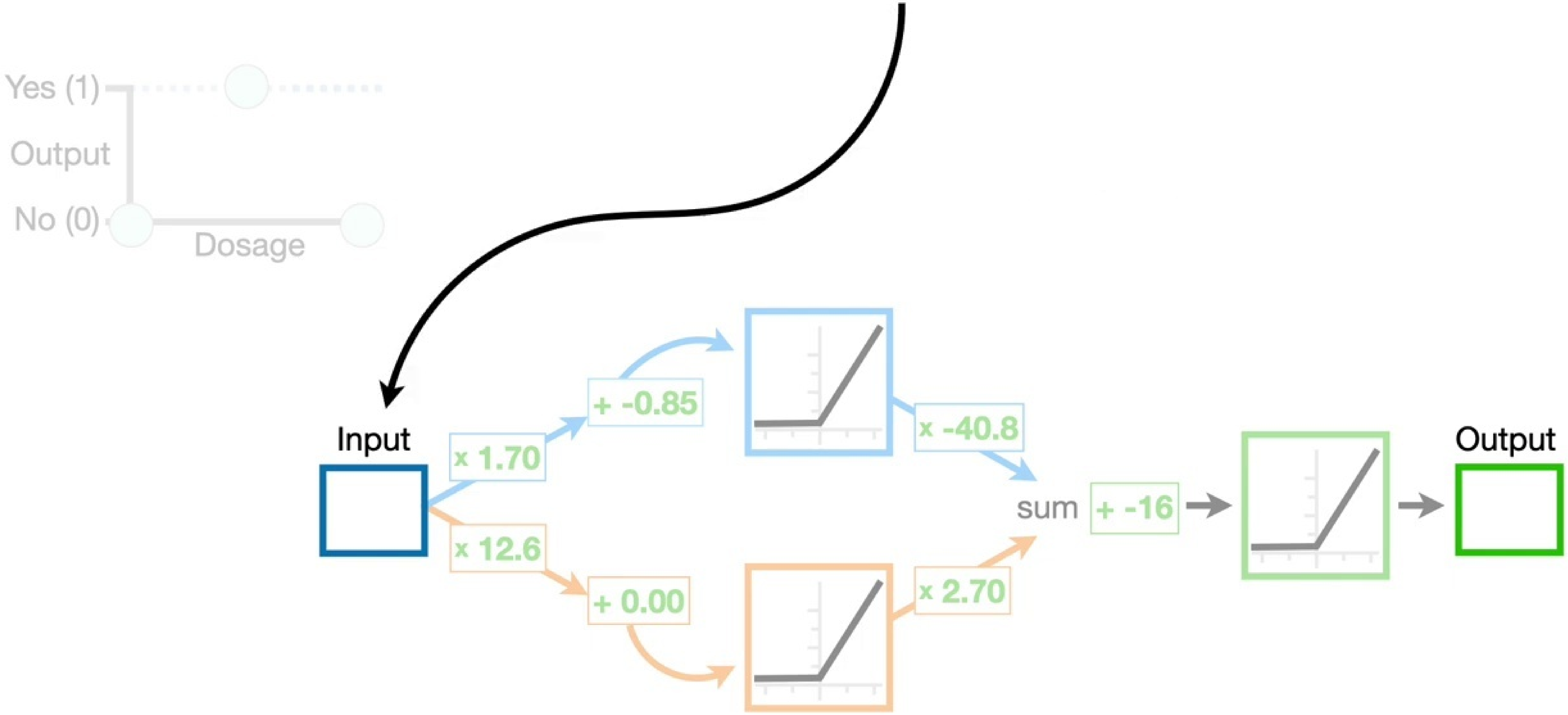


…and one **Output Node.**

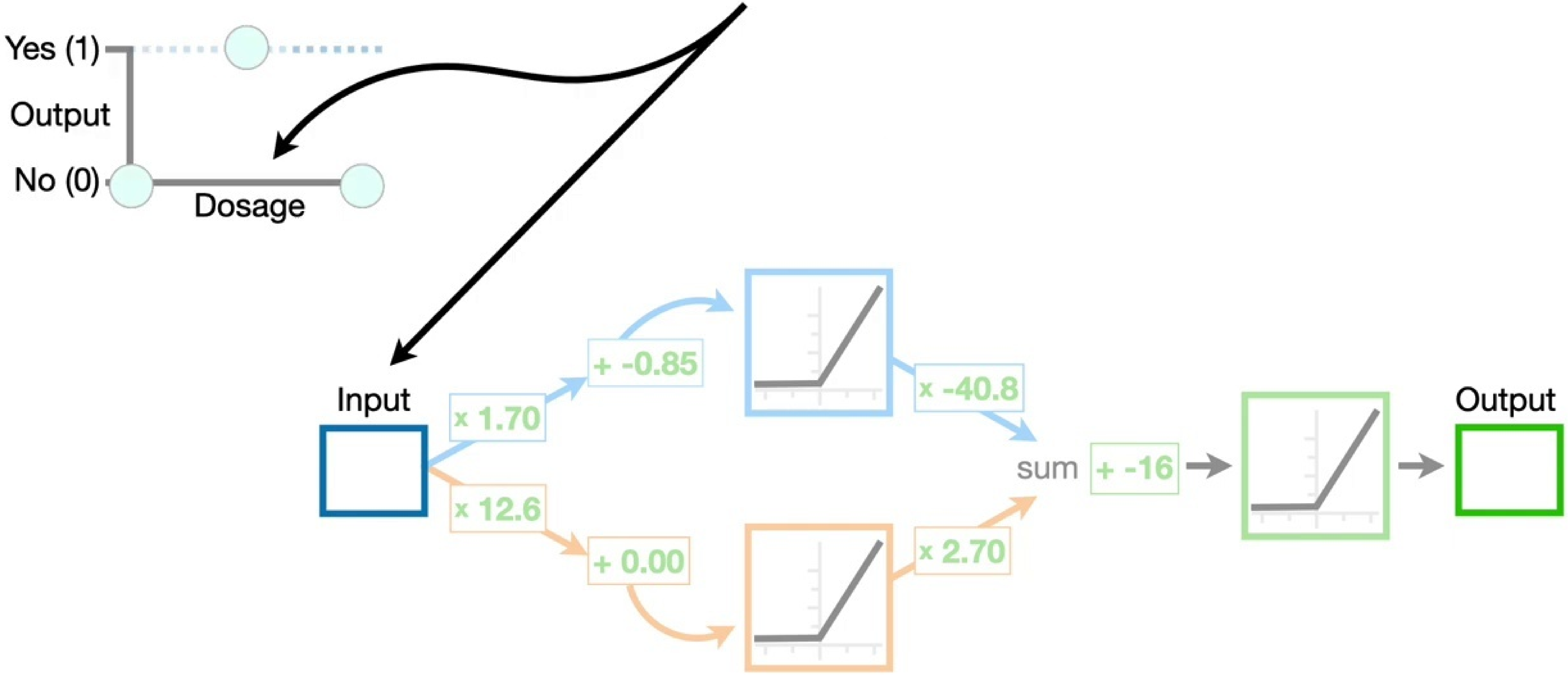




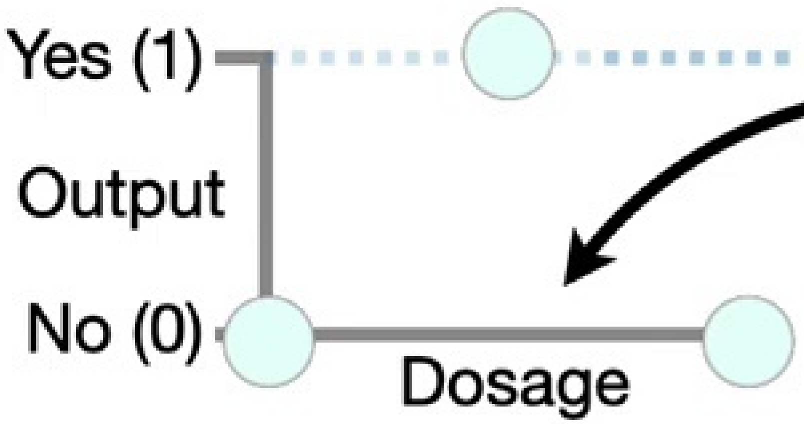
When there is only one input node…

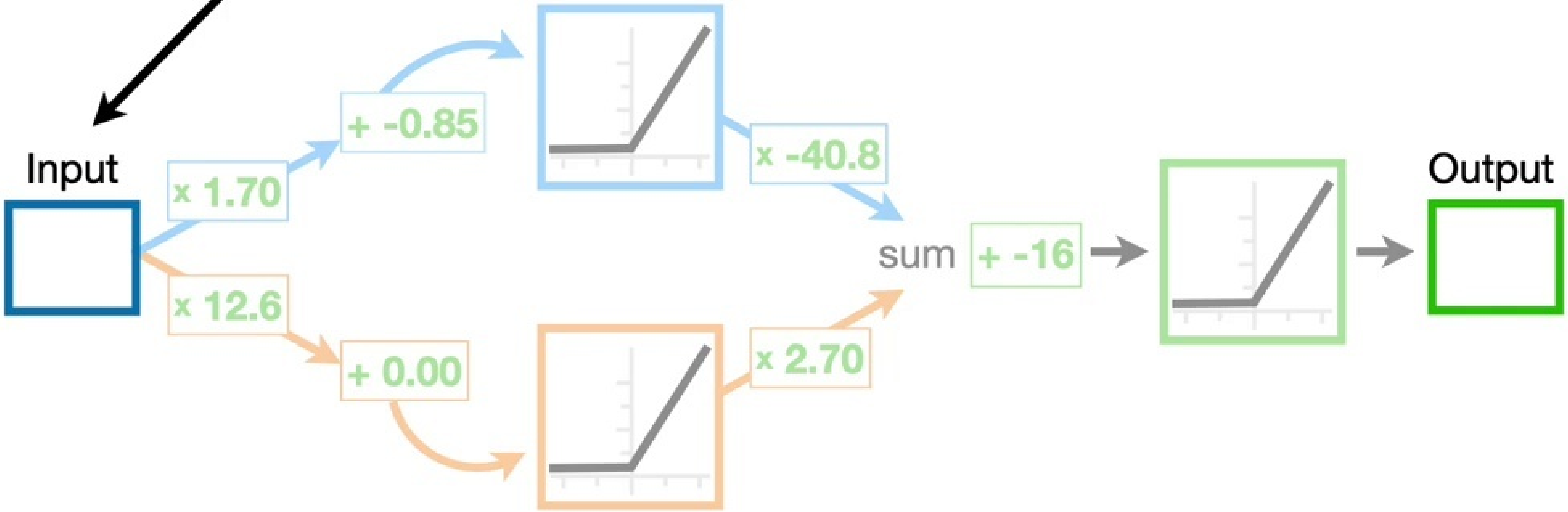


…then the data we are using to make predictions, in this case, **Dosages**, can all fit on the x-axis of this graph.

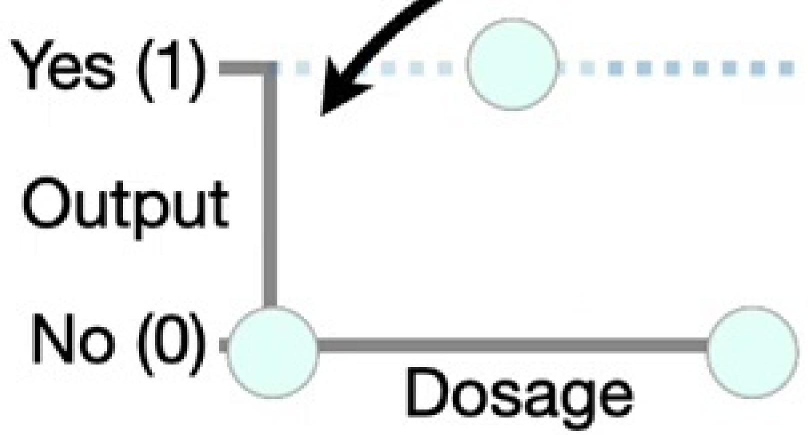


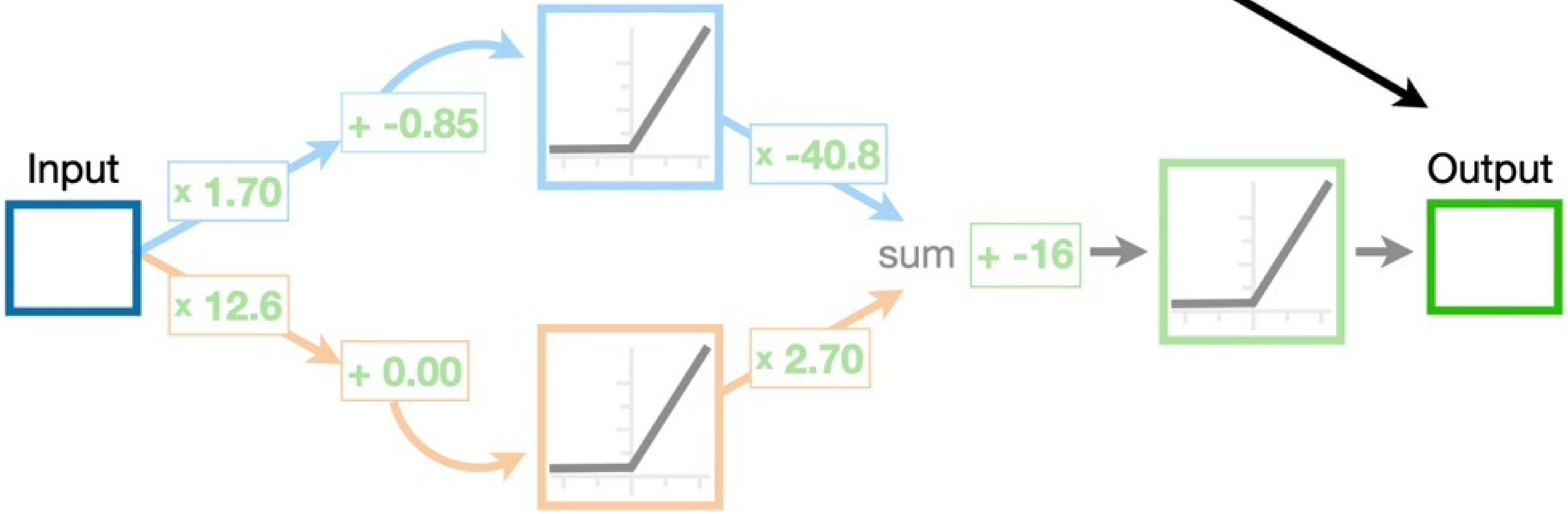
In other words, the input is **1-Dimensional**, since it only needs one axis in the graph.



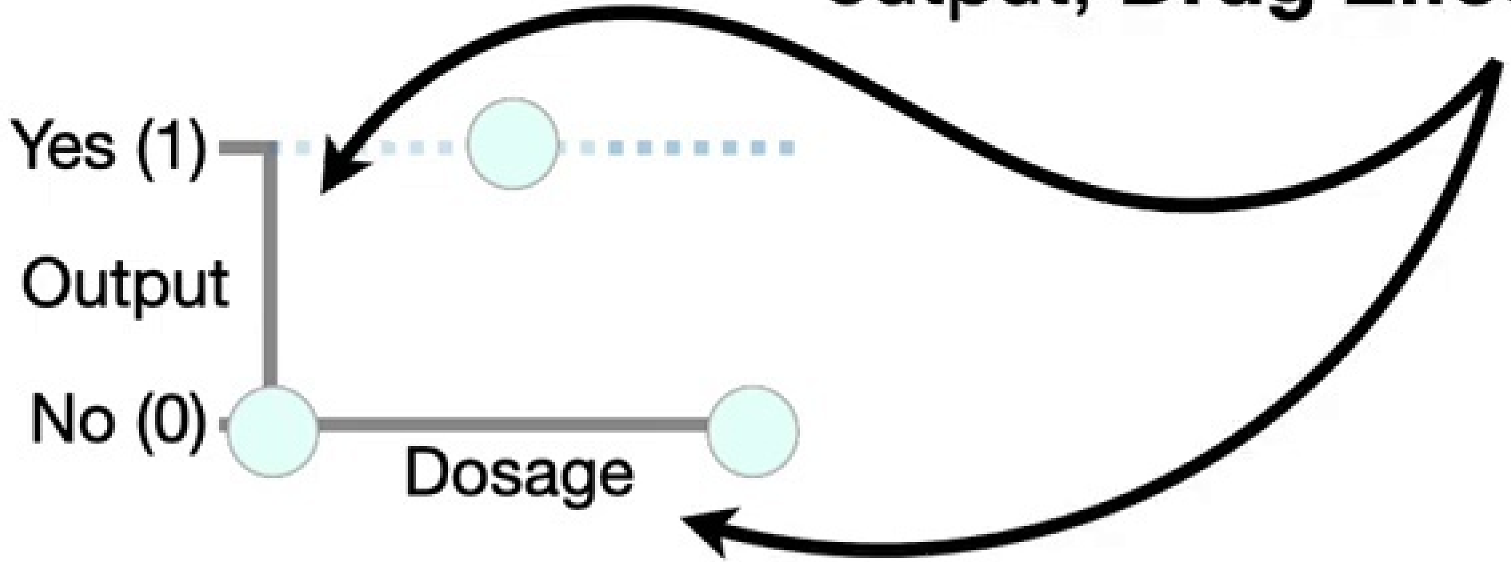


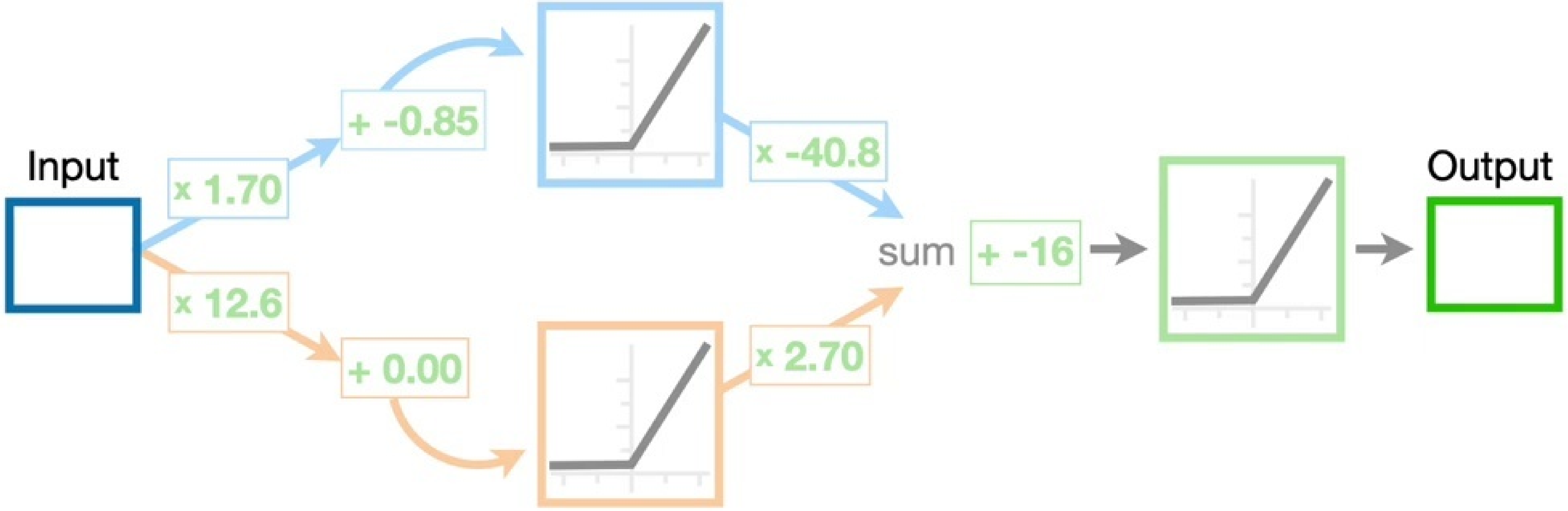
Likewise, a single dimension, the y-axis, represents the output values.



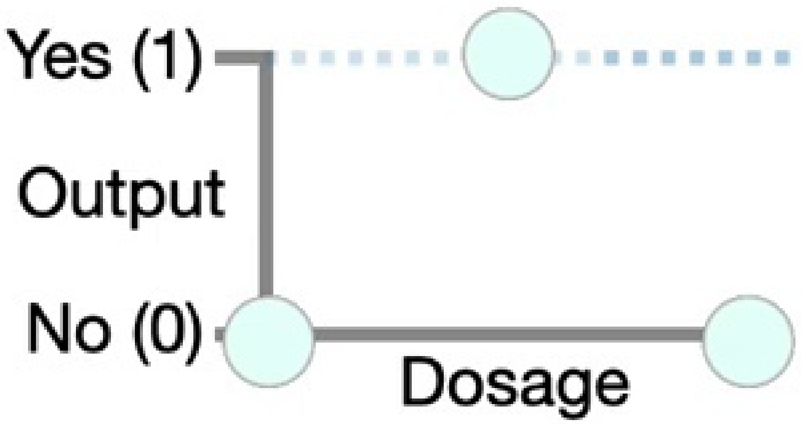


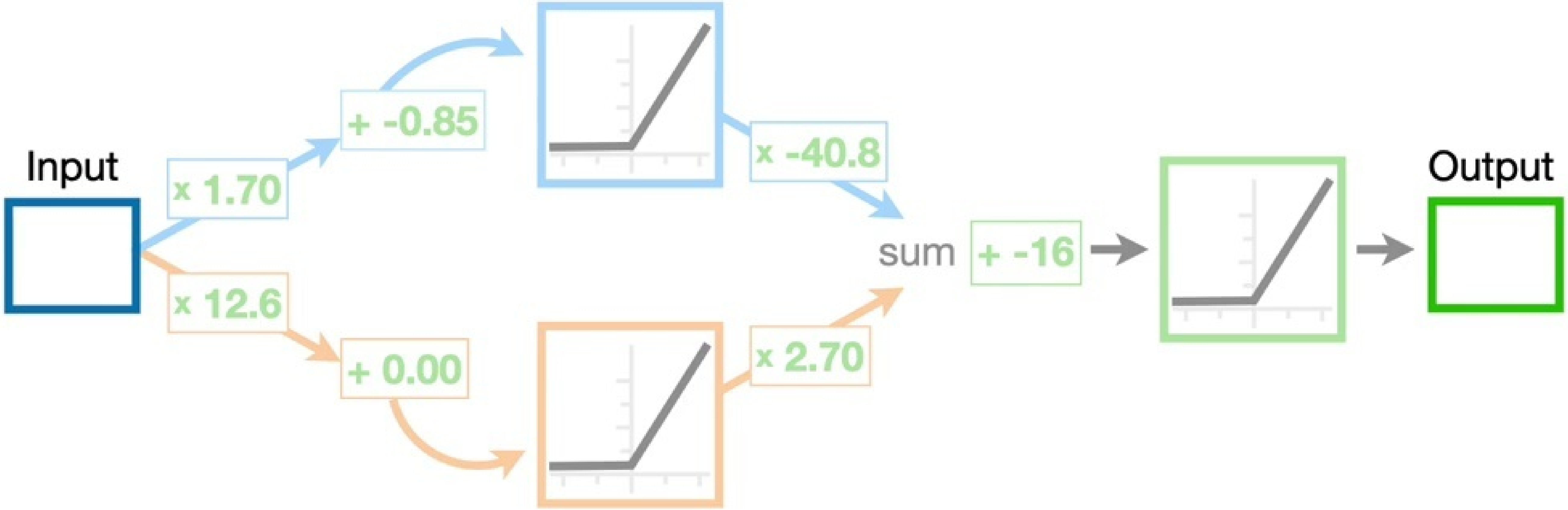
Combined, we get a **2-Dimensional** graph, with the input, **Dosage**, on the x-axis and the output, **Drug Effectiveness**, on the y-axis.



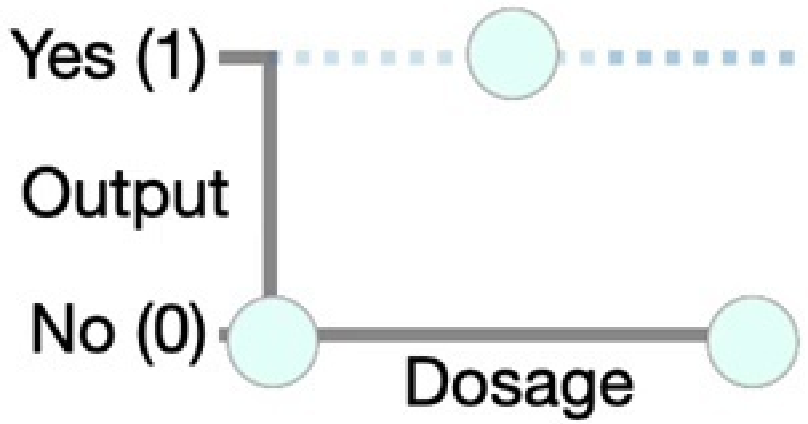


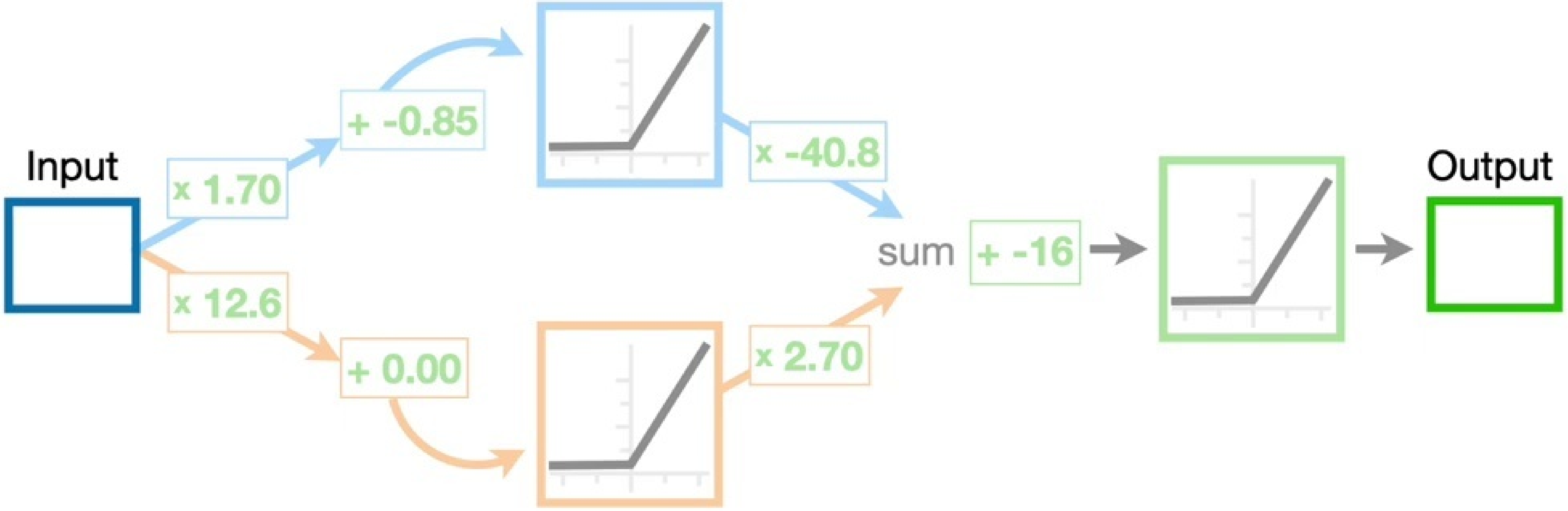
Because the input and the output combine
to form a **2-Dimensional** graph…



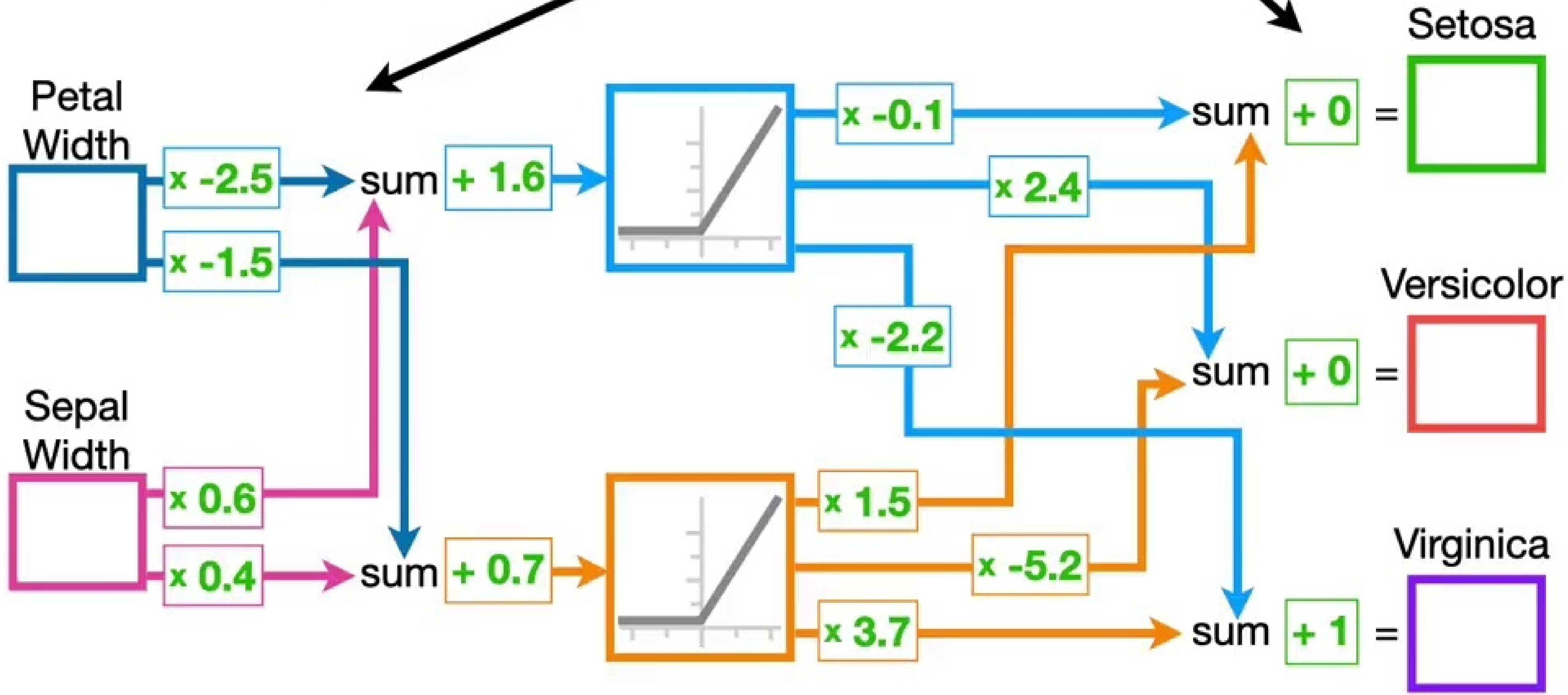


…we can see how the **Weights** and **Biases** in this **Neural Network** slice, flip and stretch the curved or bent **Activation Functions** into new shapes…

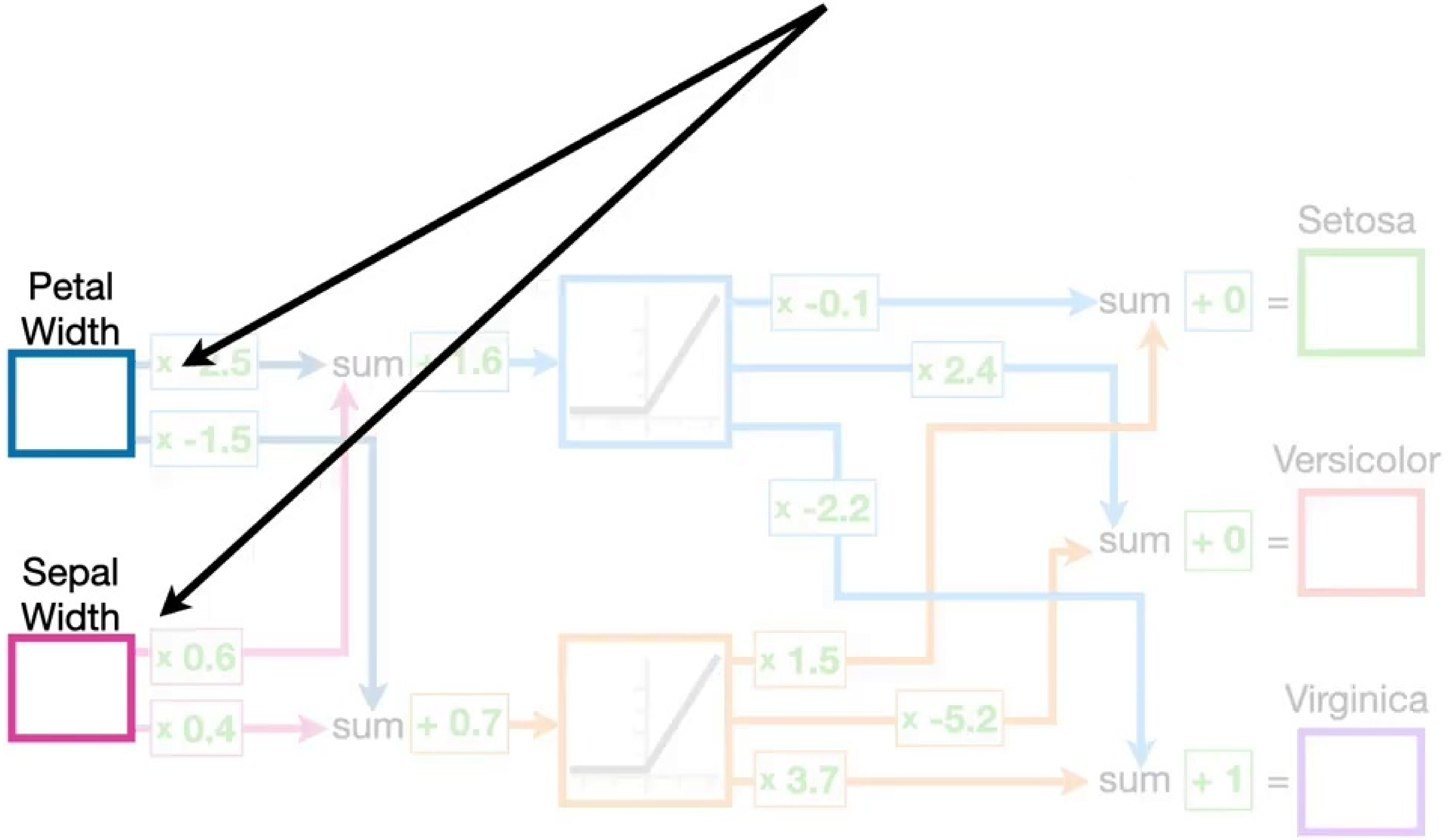




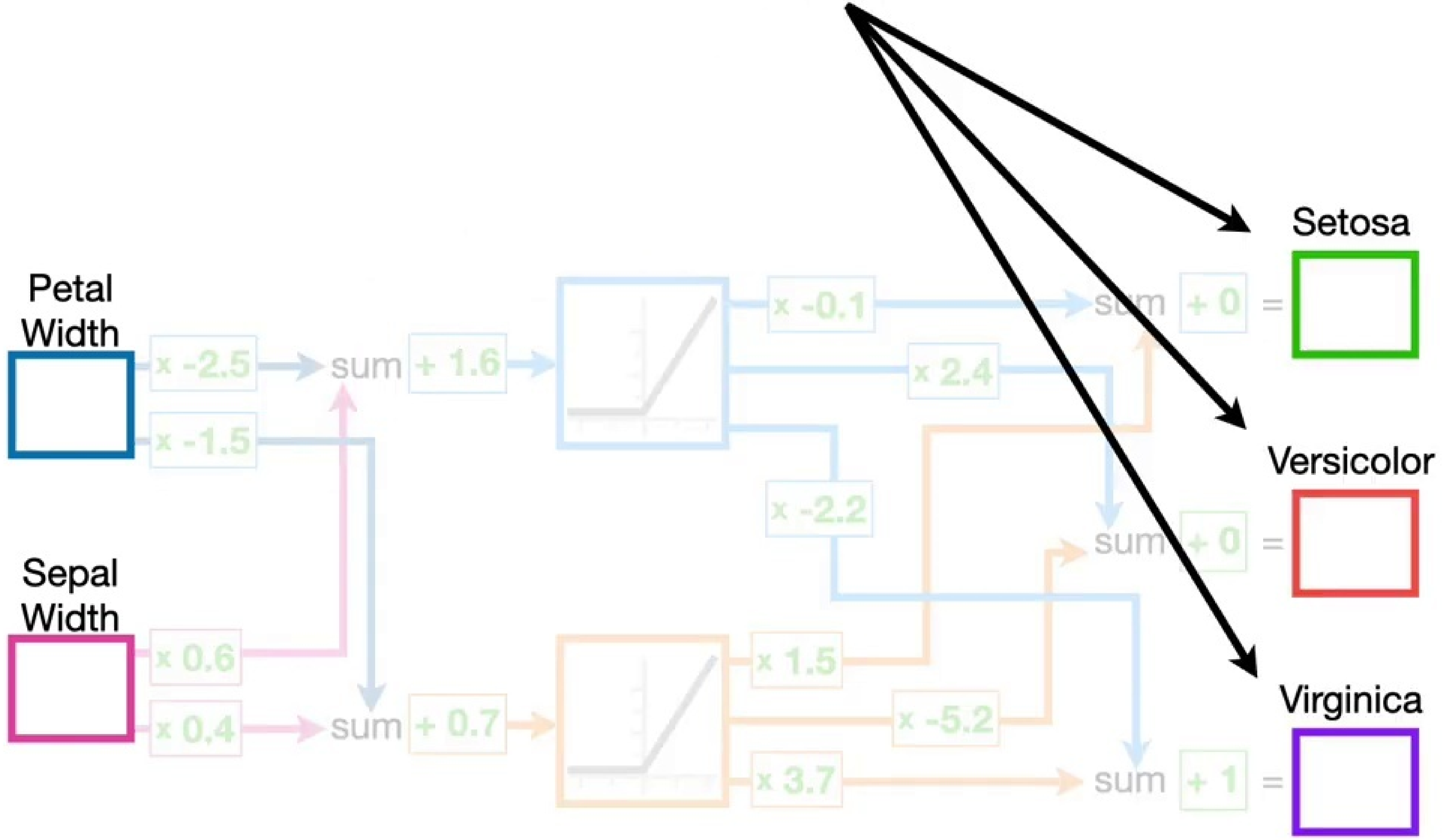
Now let's look at a more complicated **Neural Network**…



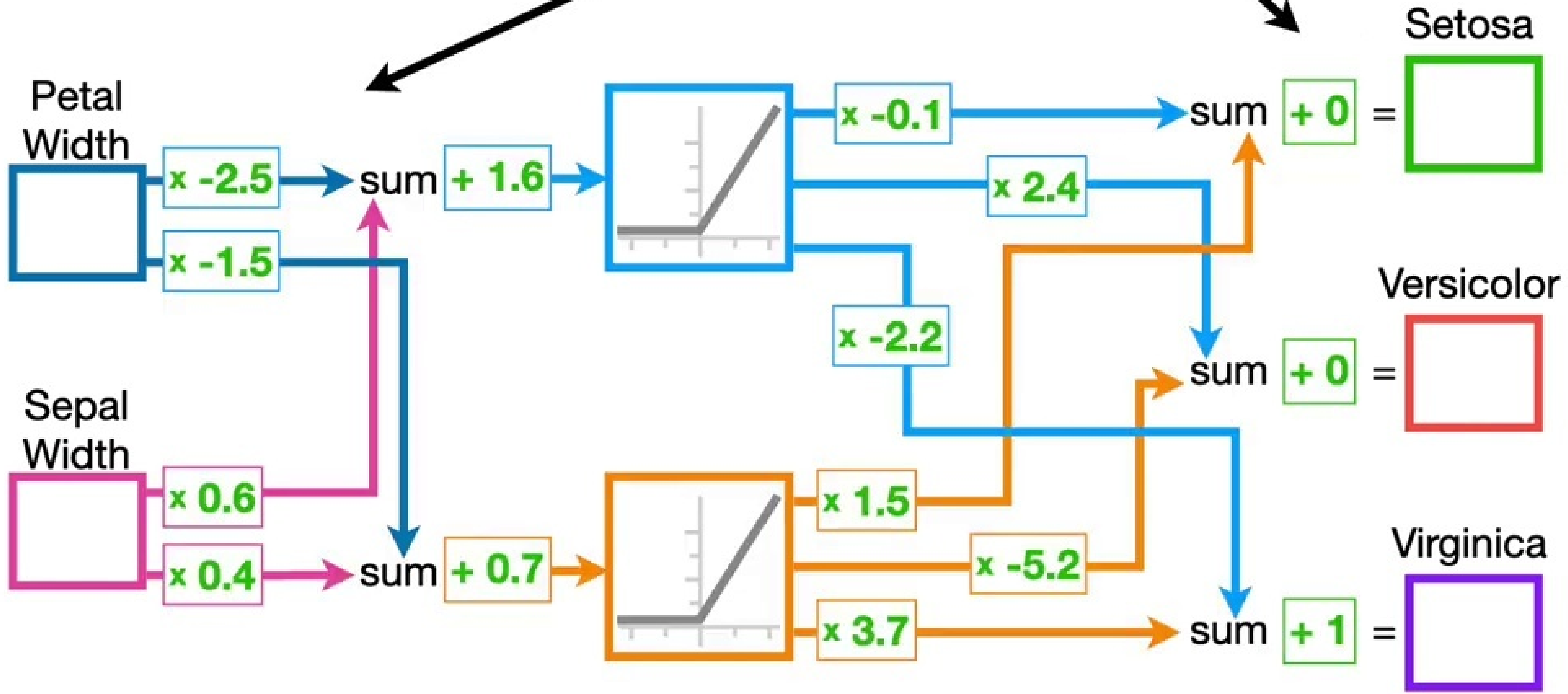
…that has more than one **Input Node**…



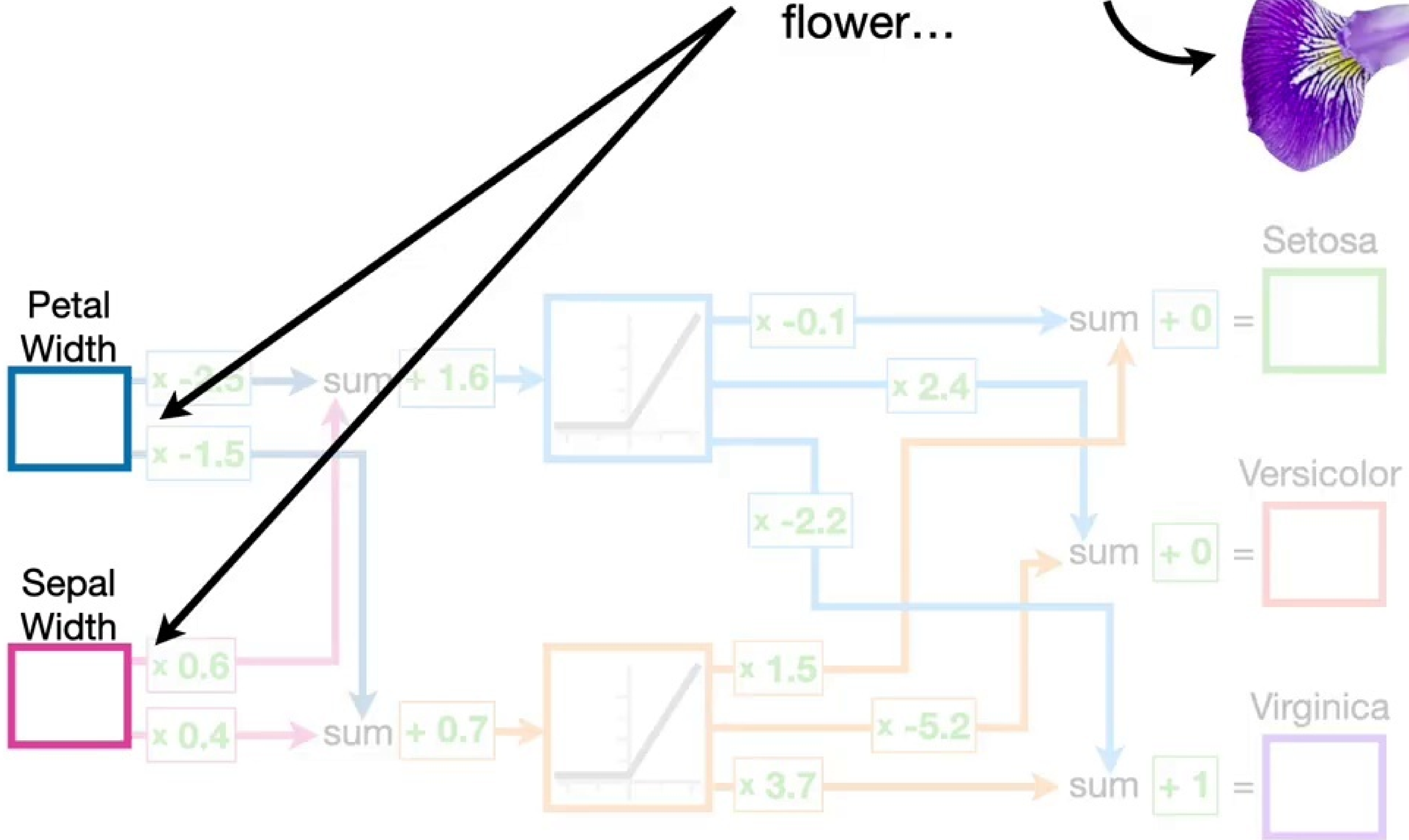
…and more than one
Output Node.

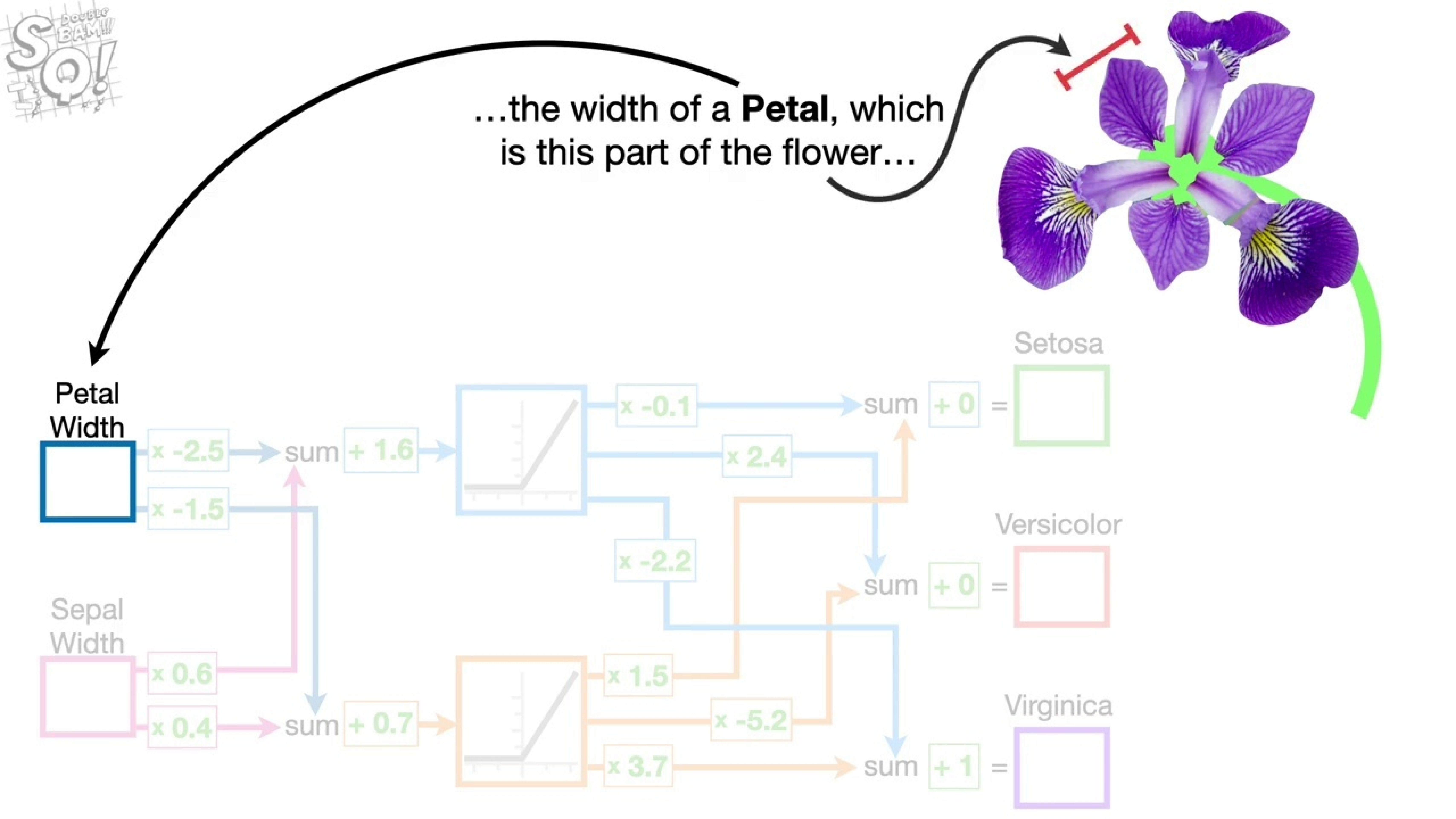


Now, this **Neural Network** may look really fancy…

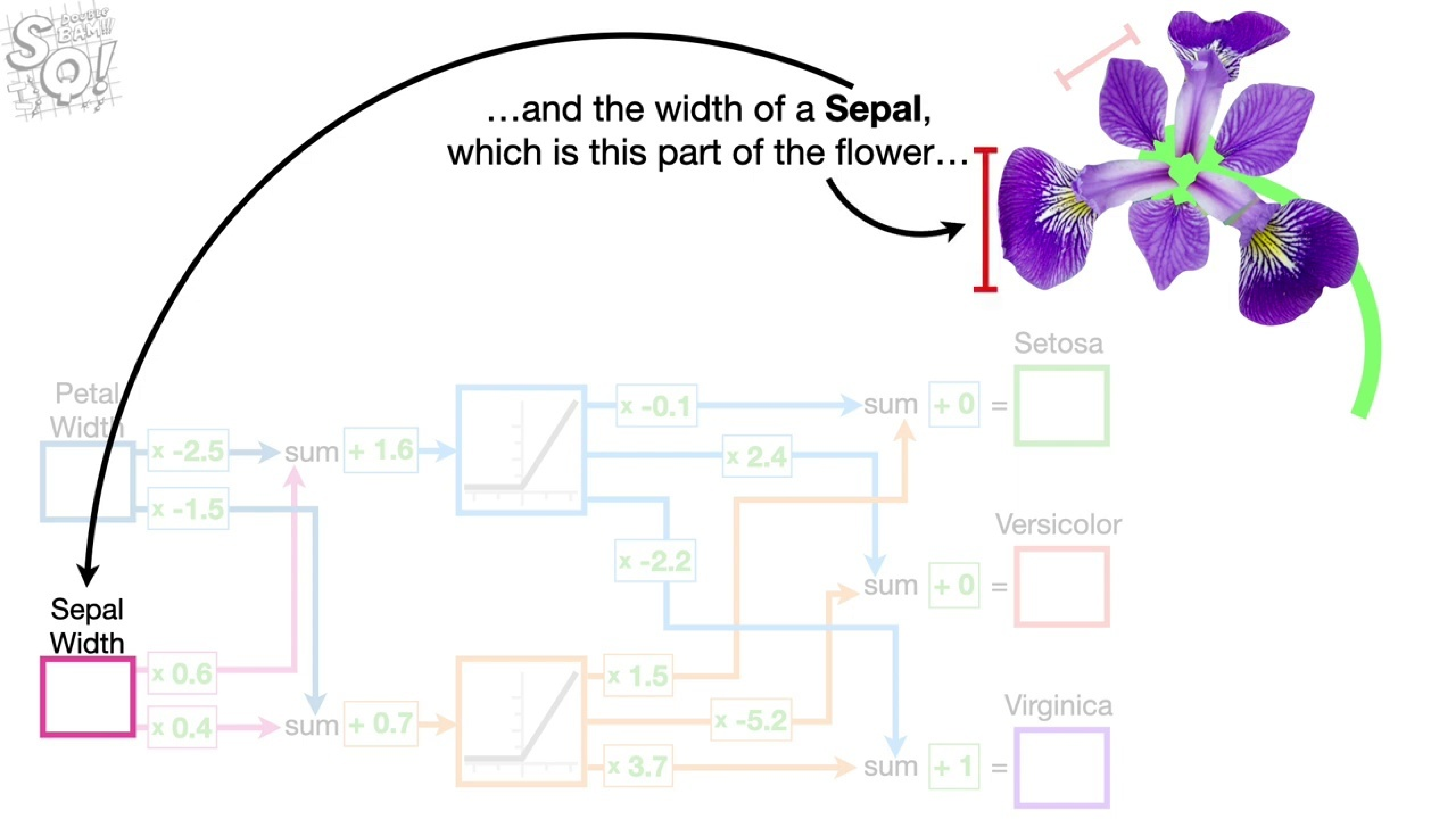


...but all it does it take two measurements from an **Iris** flower...



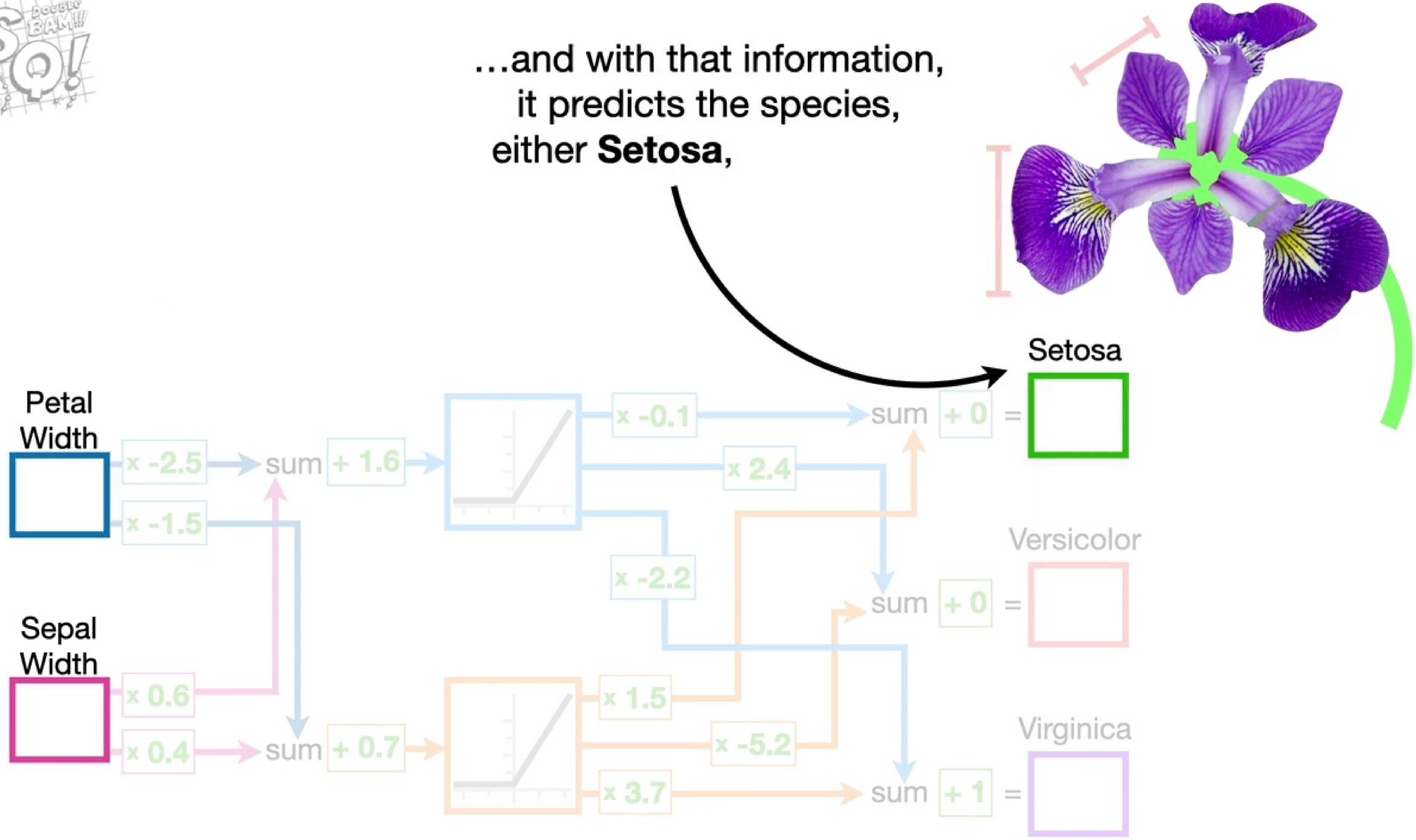


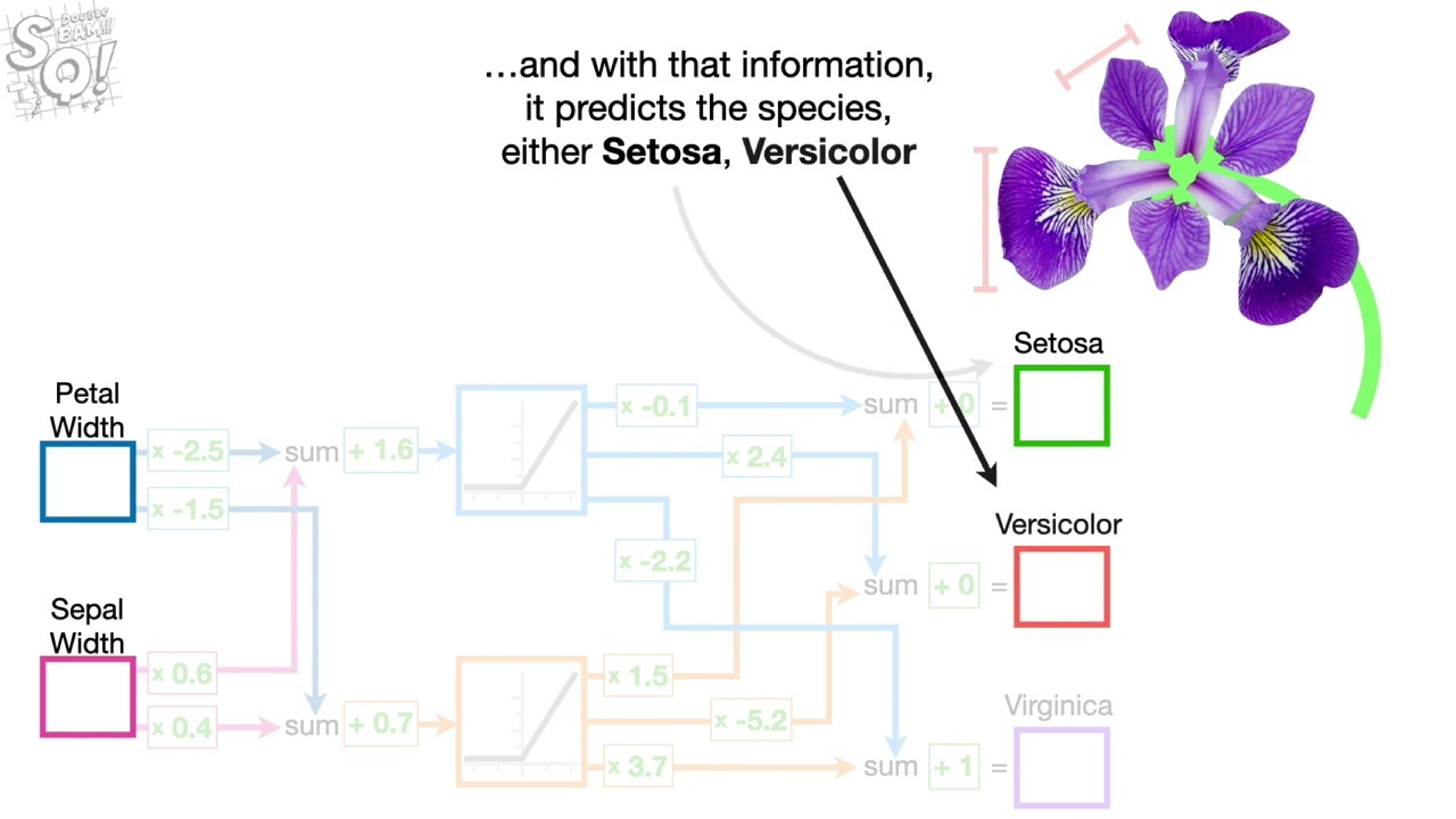
…the width of a **Petal**, which
is this part of the flower…
Petal
Width
x -2.5
x -1.5
sum
+ 1.6
Sepal
Width
x 0.6
x 0.4
sum
+ 0.7
x -0.1
x 2.4
x -2.2
x 1.5
x -5.2
x 3.7
sum
+ 0
=
Setosa
sum
+ 0
=
Versicolor
sum
+ 1
=
Virginica



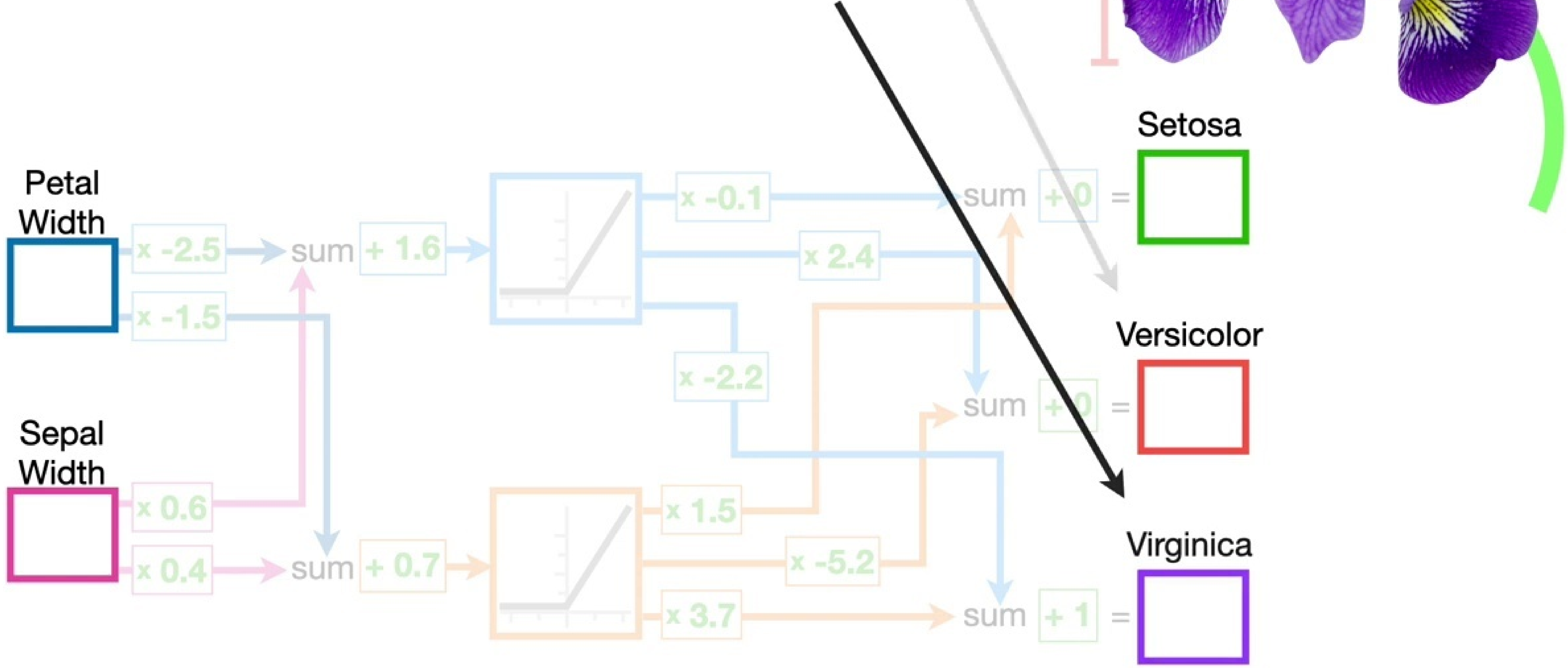
…and the width of a **Sepal**,
which is this part of the flower…
Petal Width
x -2.5
x -1.5
sum
+ 1.6
Sepal Width
x 0.6
x 0.4
sum
+ 0.7
x -0.1
x 2.4
x -2.2
x 1.5
x -5.2
x 3.7
Setosa
sum + 0 =
Versicolor
sum + 0 =
Virginica
sum + 1 =

…and with that information,
it predicts the species,
either **Setosa**,

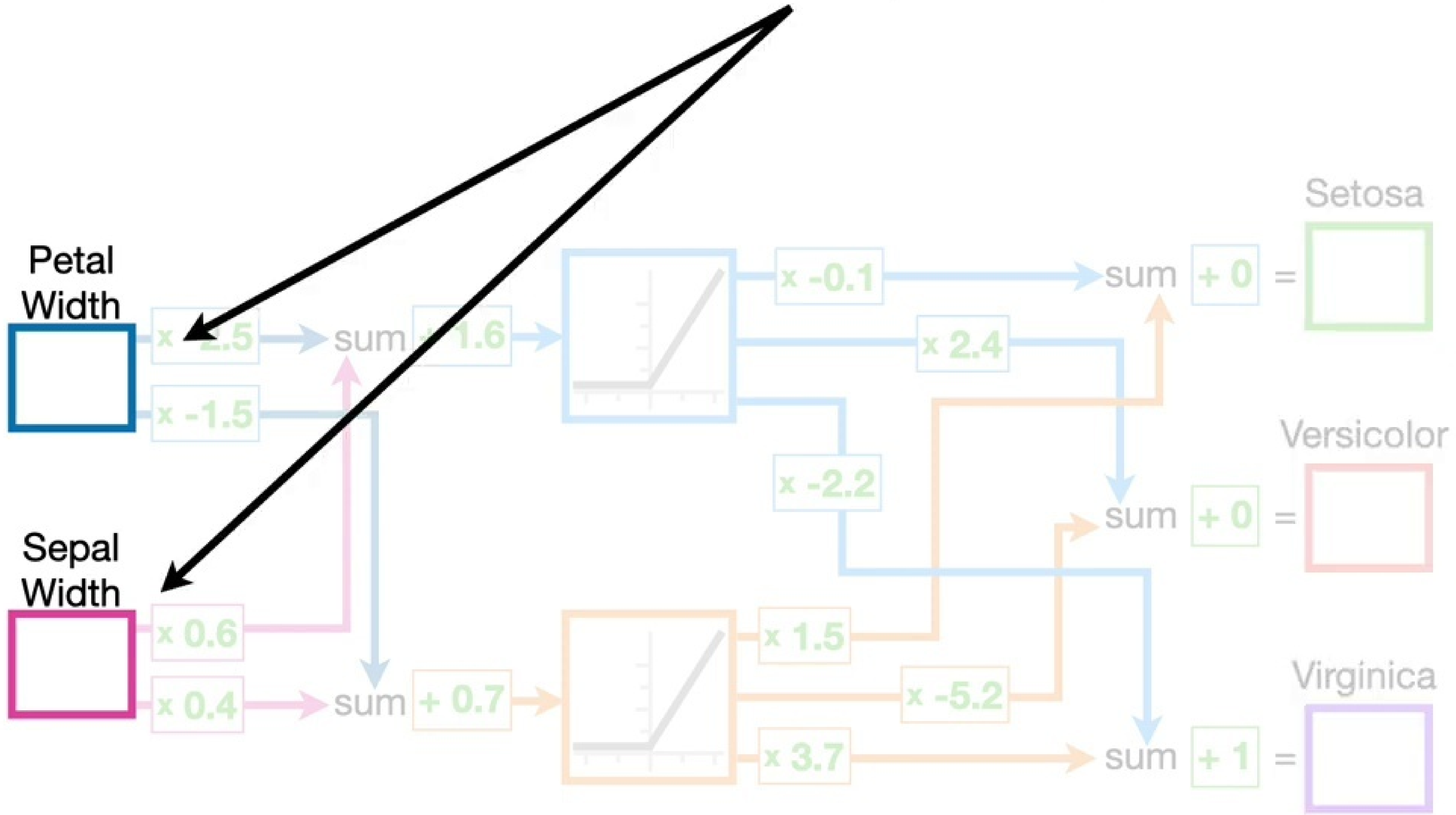



…and with that information, it predicts the species, either **Setosa**, **Versicolor**
Petal Width
Sepal Width
x -2.5
x -1.5
x 0.6
x 0.4
sum
+ 1.6
sum
+ 0.7
x -0.1
x 2.4
x -2.2
x 1.5
x -5.2
x 3.7
sum
+ 0
=
sum
+ 0
=
sum
+ 1
=
Setosa
Versicolor
Virginica

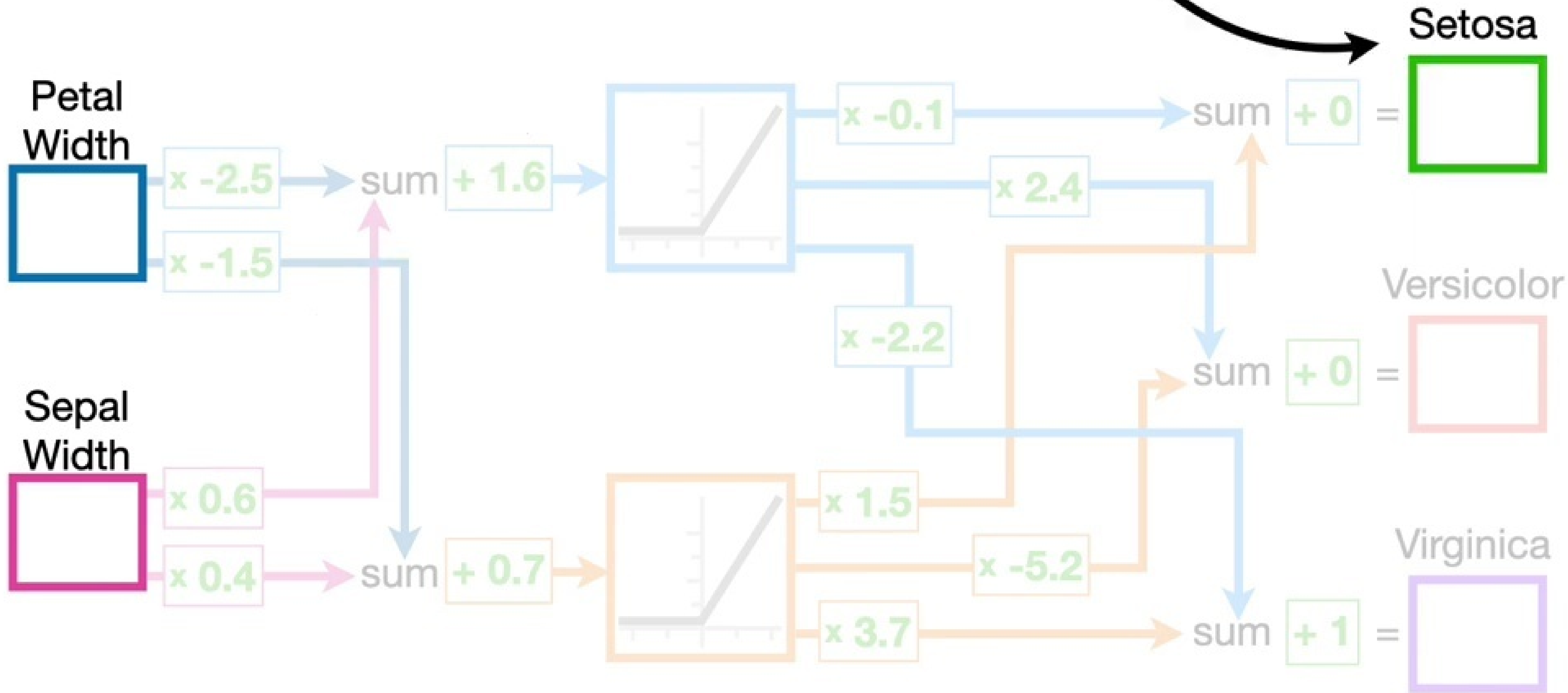
…and with that information, it predicts the species, either **Setosa**, **Versicolor** or **Virginica**.



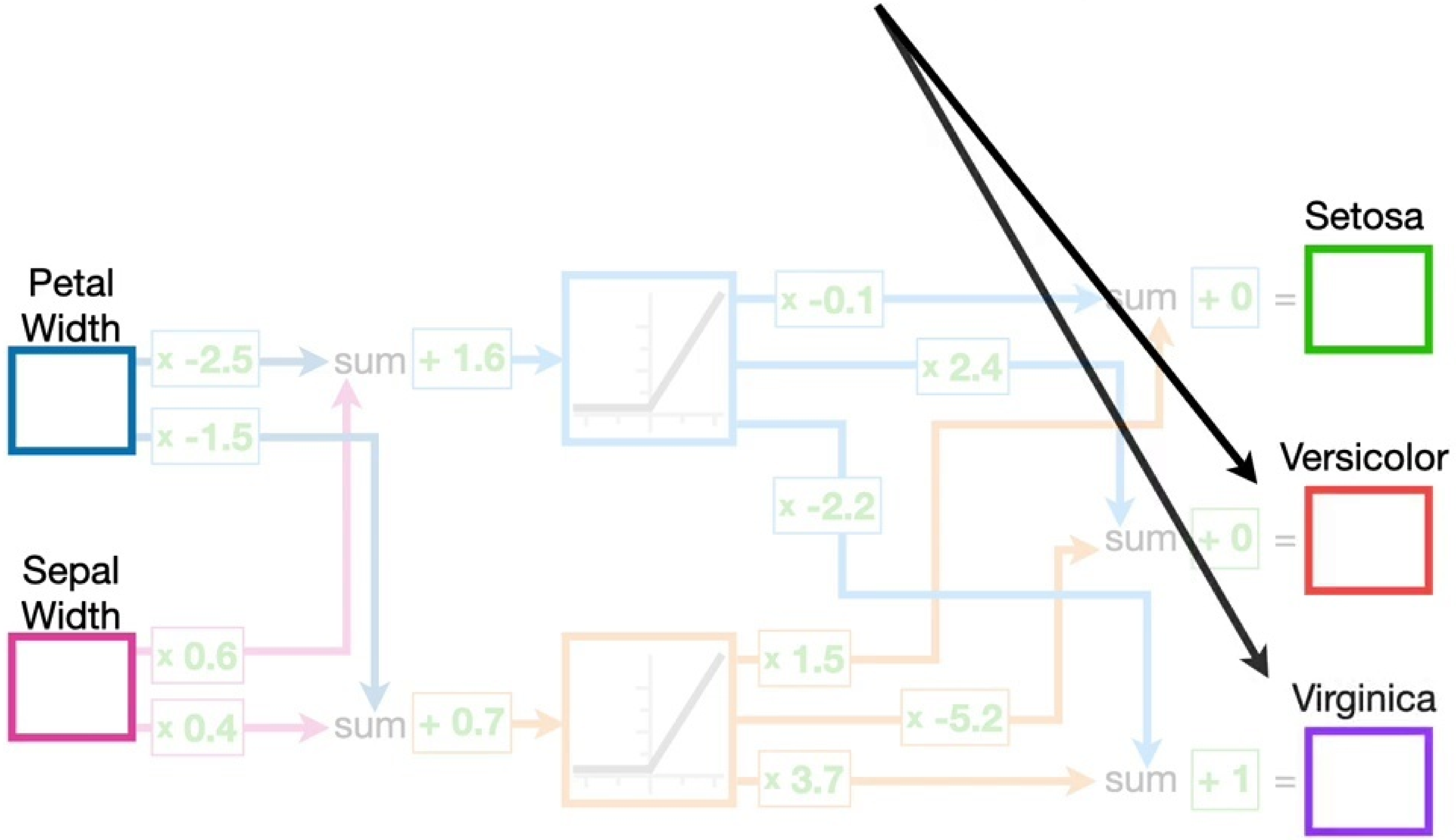
Anyway, to keep things simple at the start, let's begin with both input nodes…



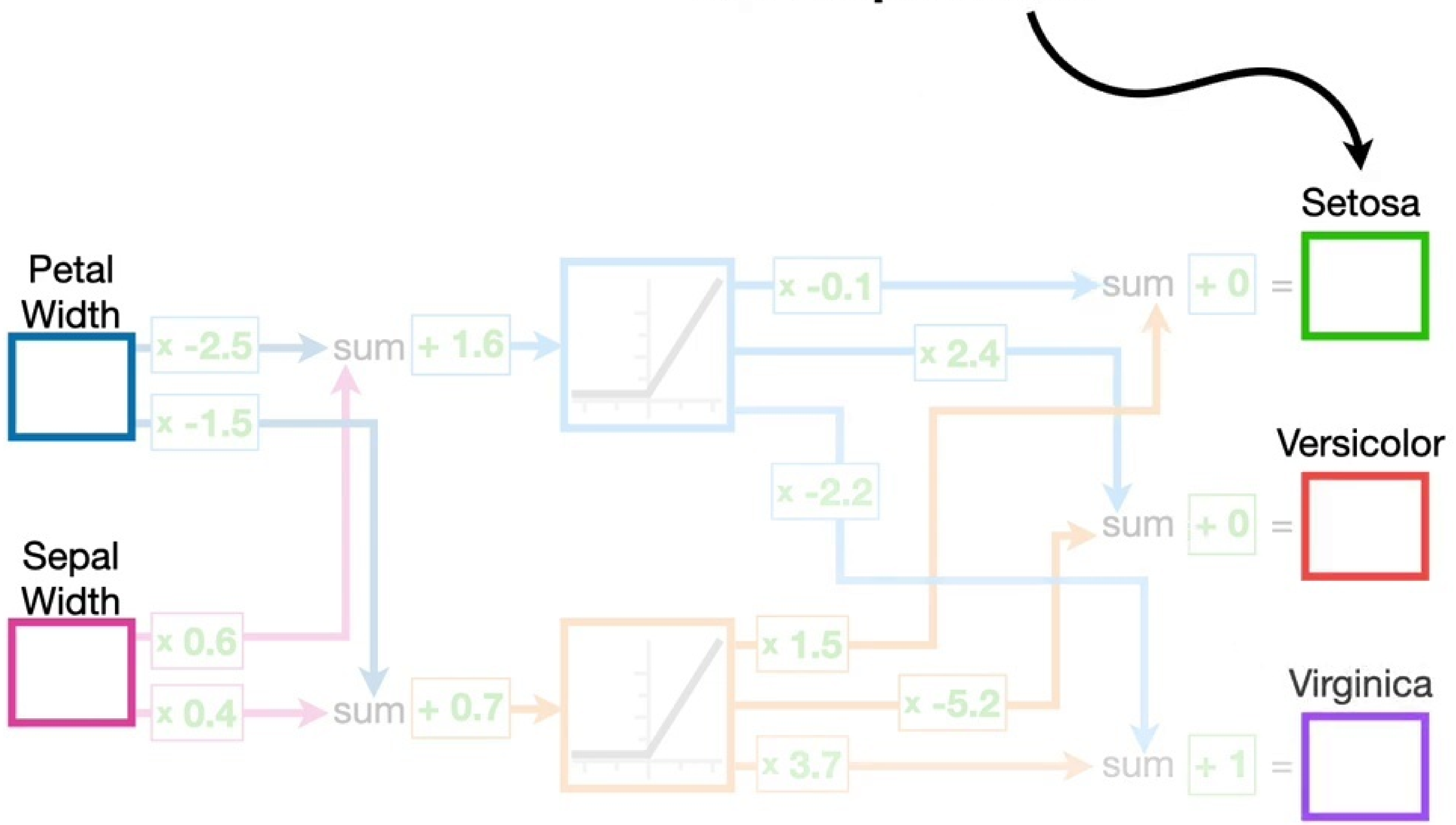
...but just one output node for **Setosa**.



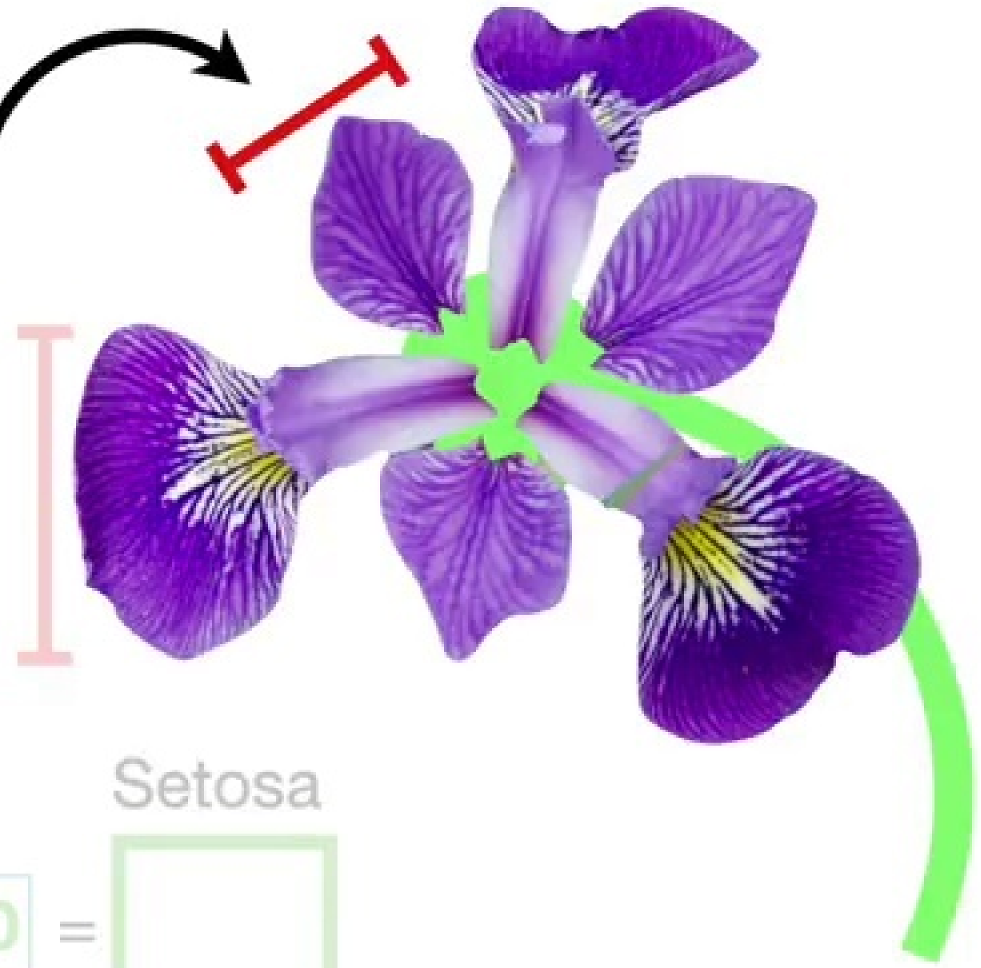
Later on we'll add the other
two output nodes...

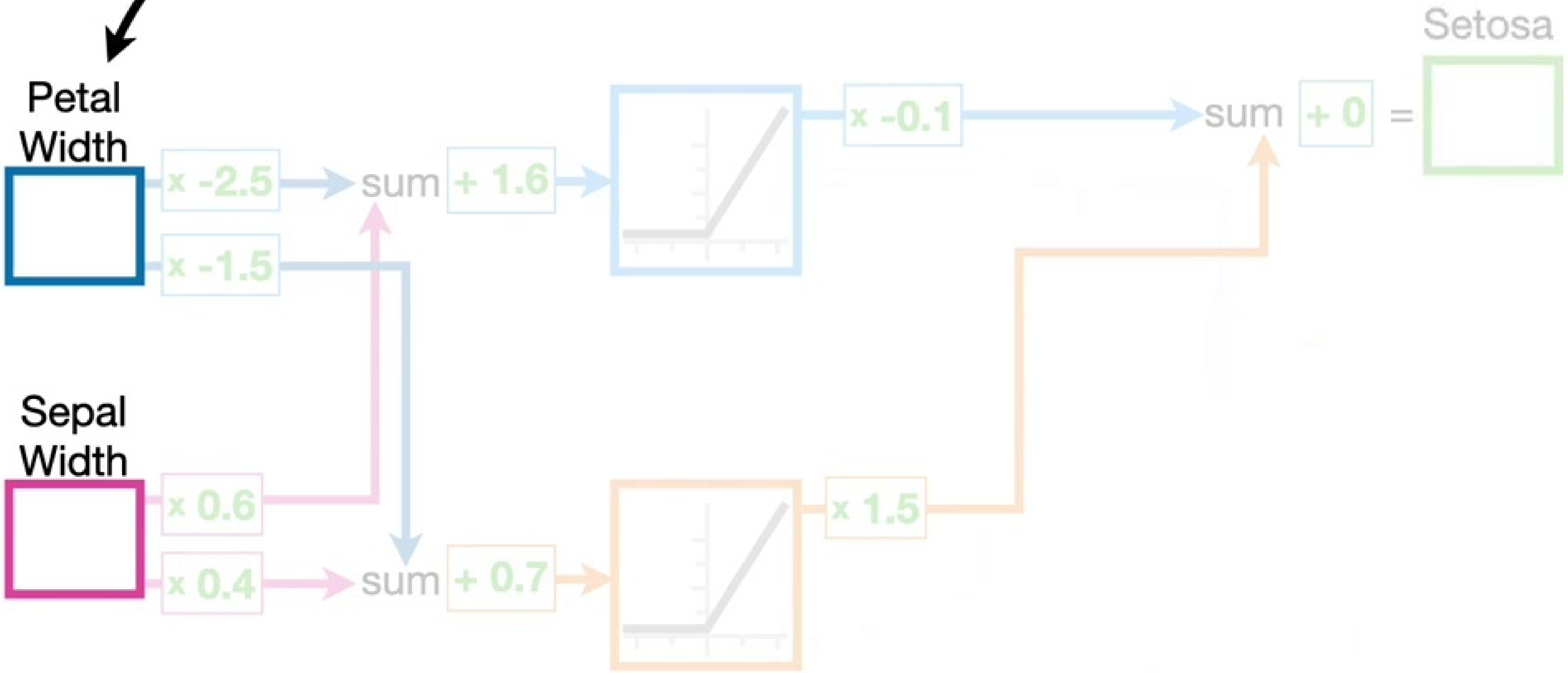


…but for now, let's keep things simple and just use one **Output Node**.

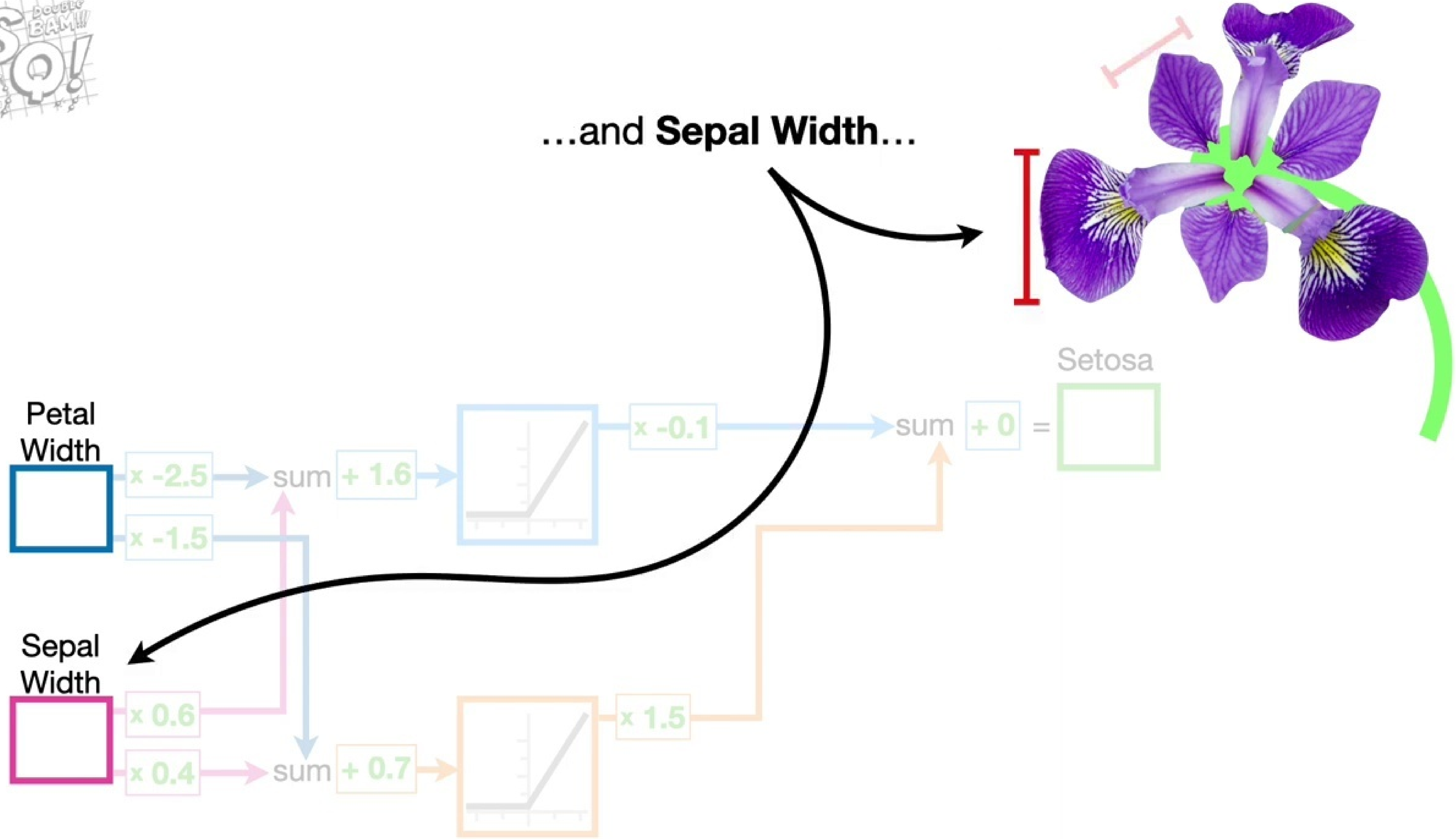


Now let's see what happens when we plug values for **Petal Width**…

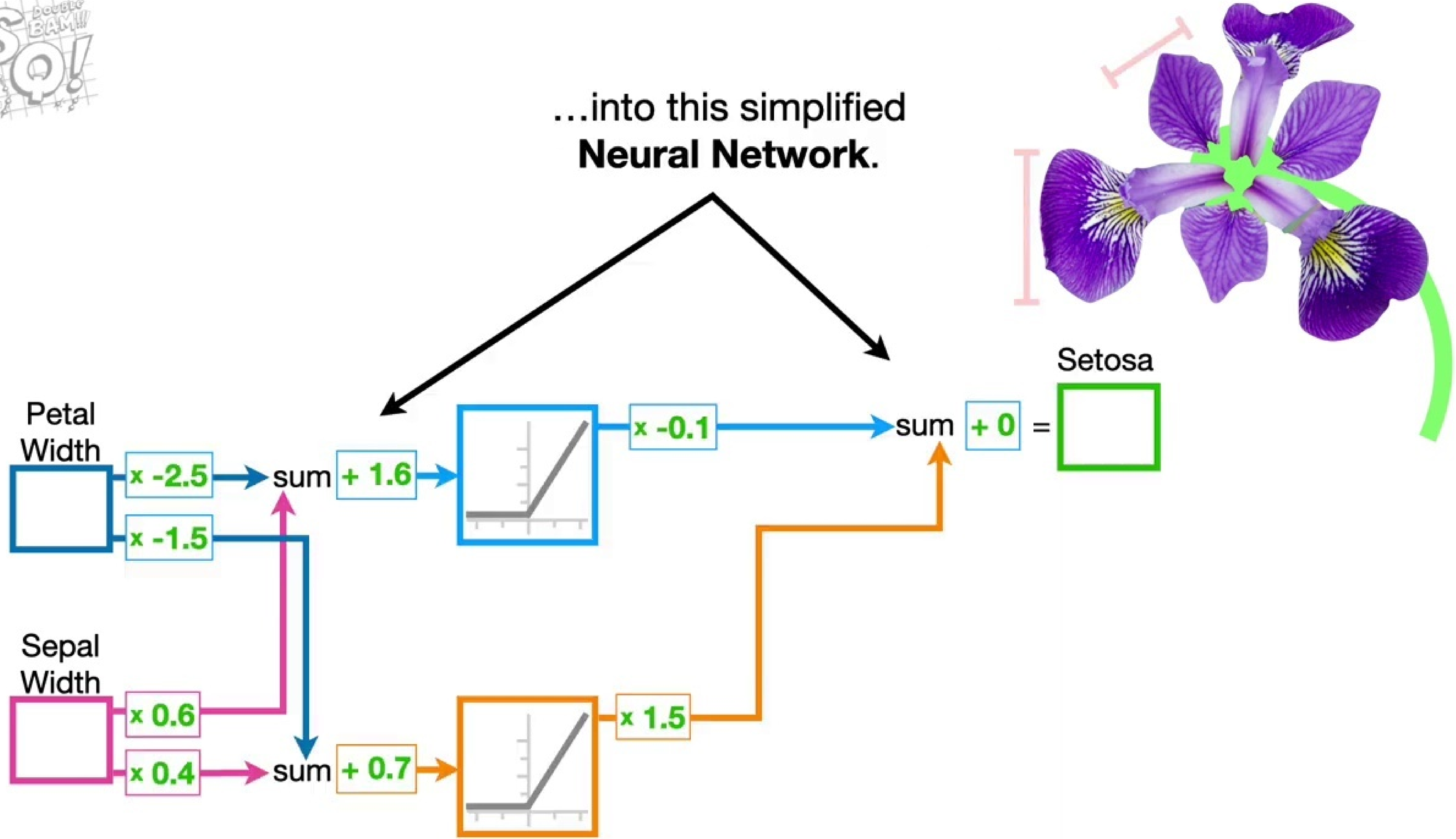


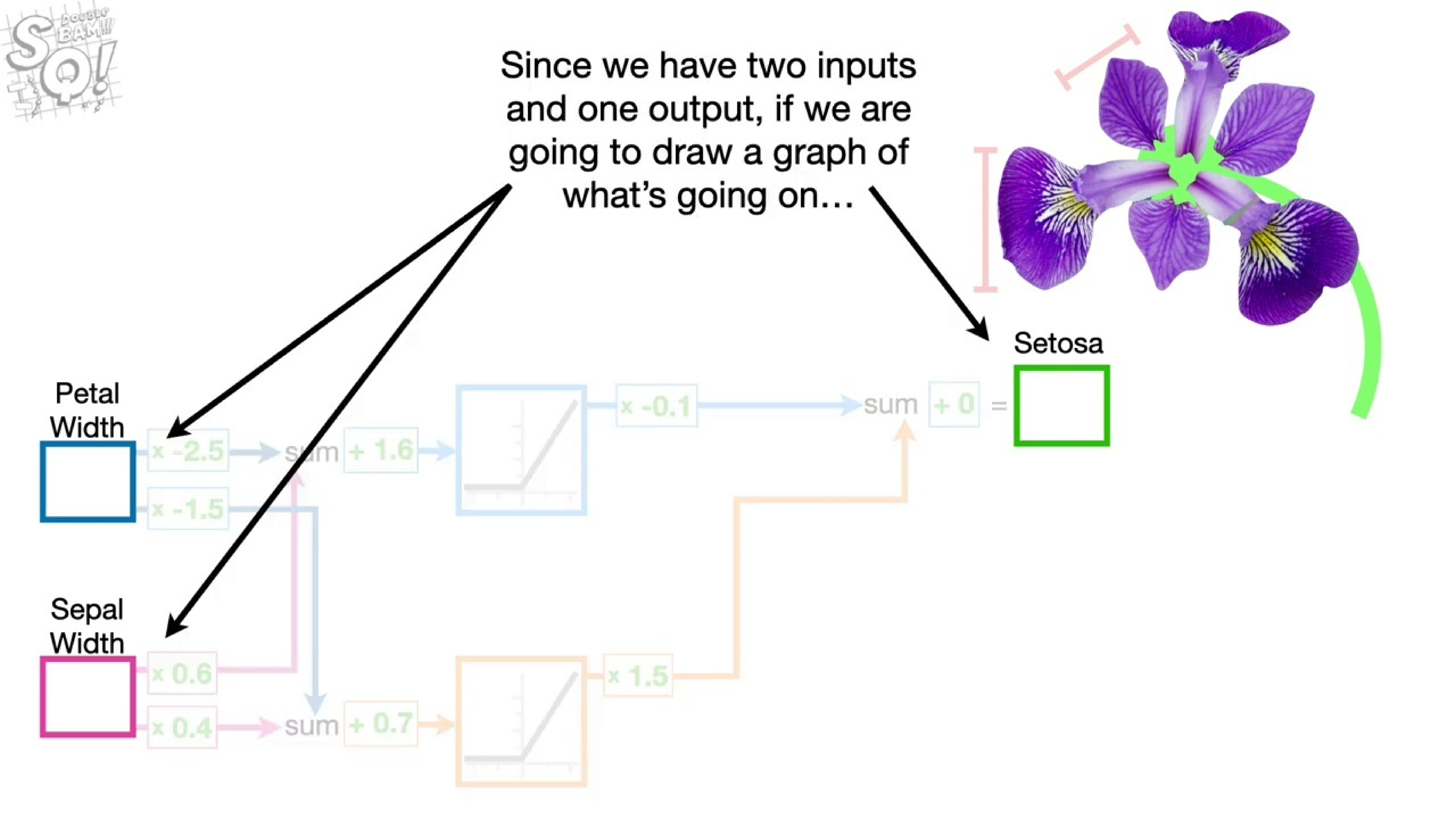


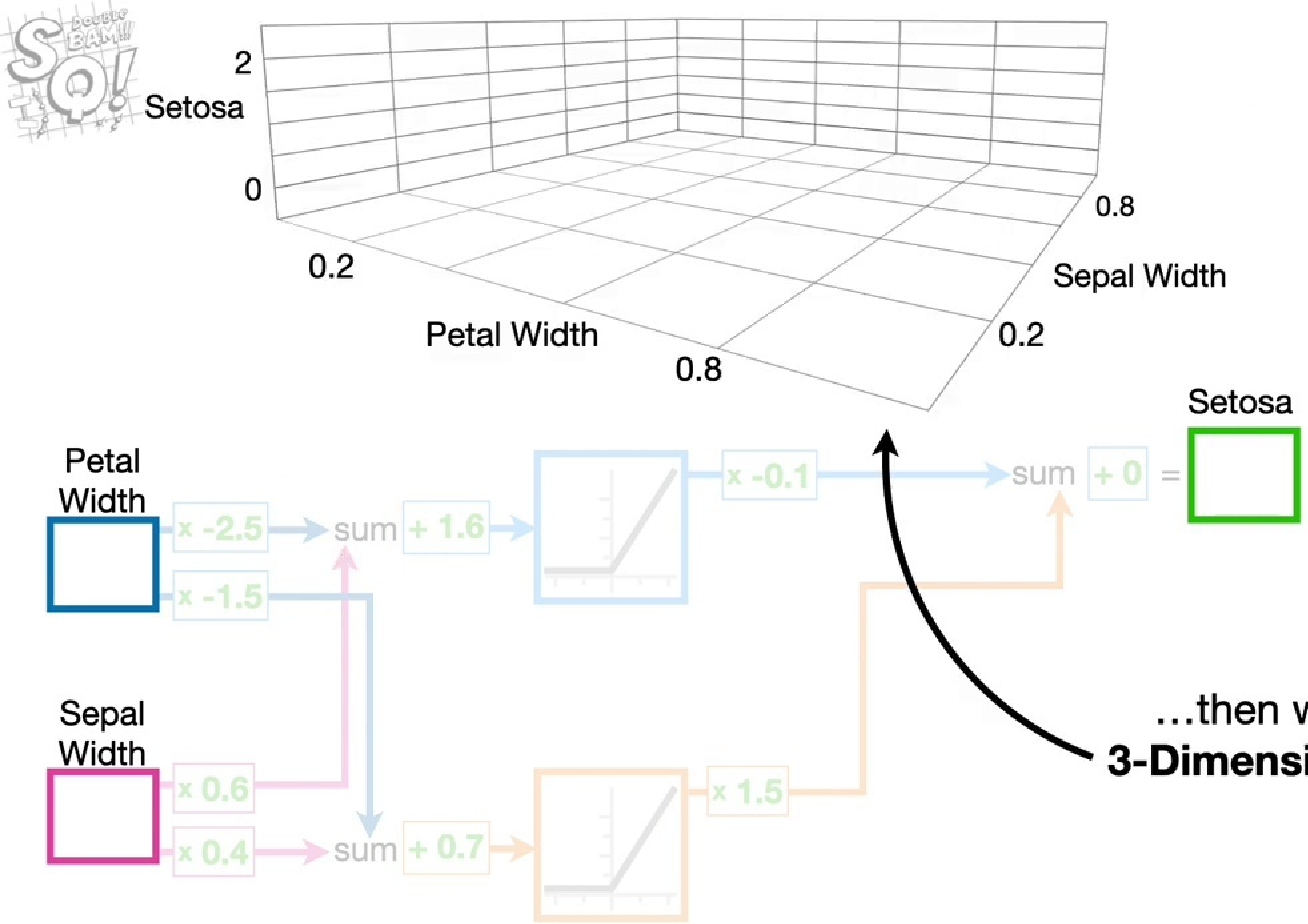
...and **Sepal Width**...



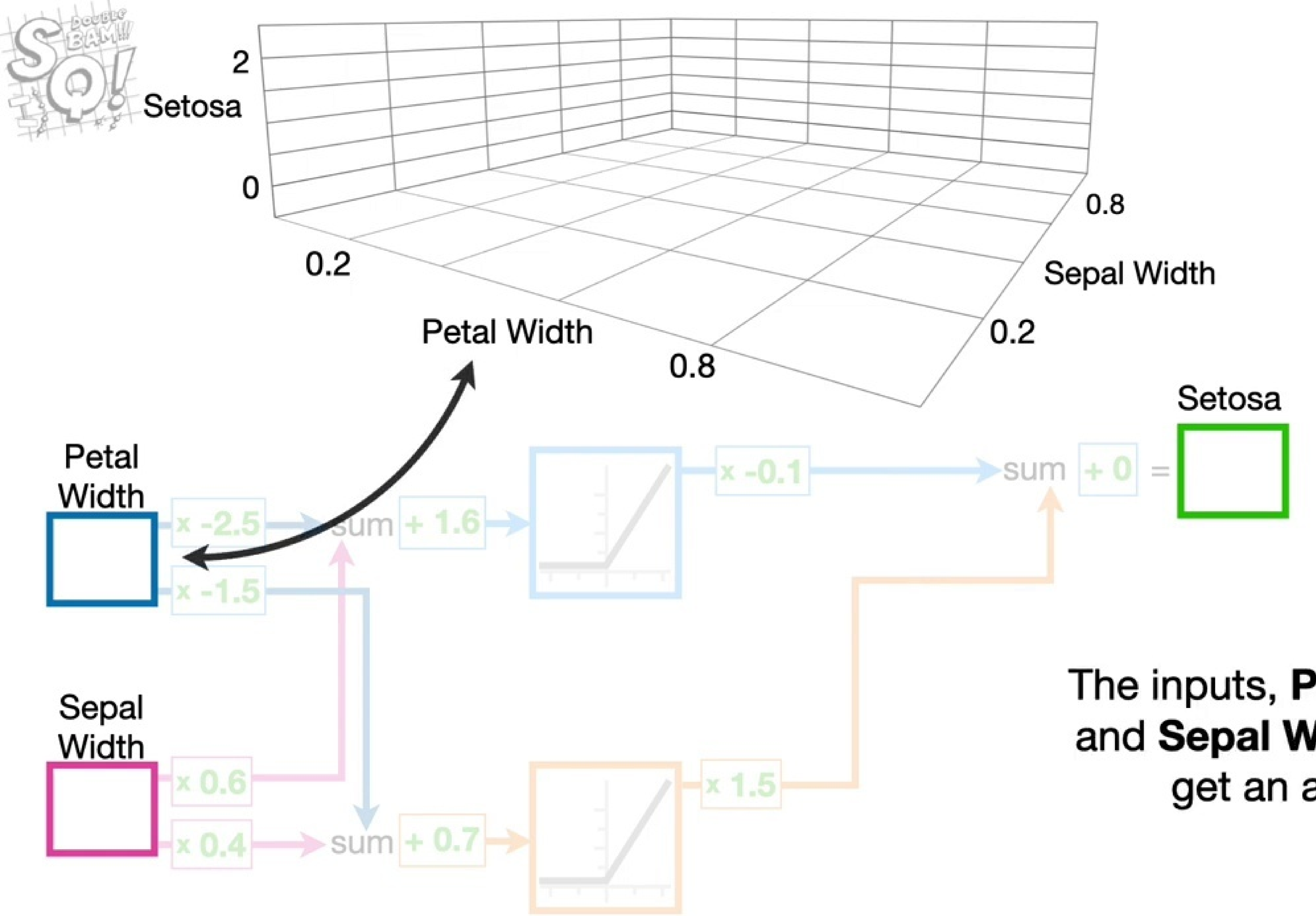
…into this simplified **Neural Network.**



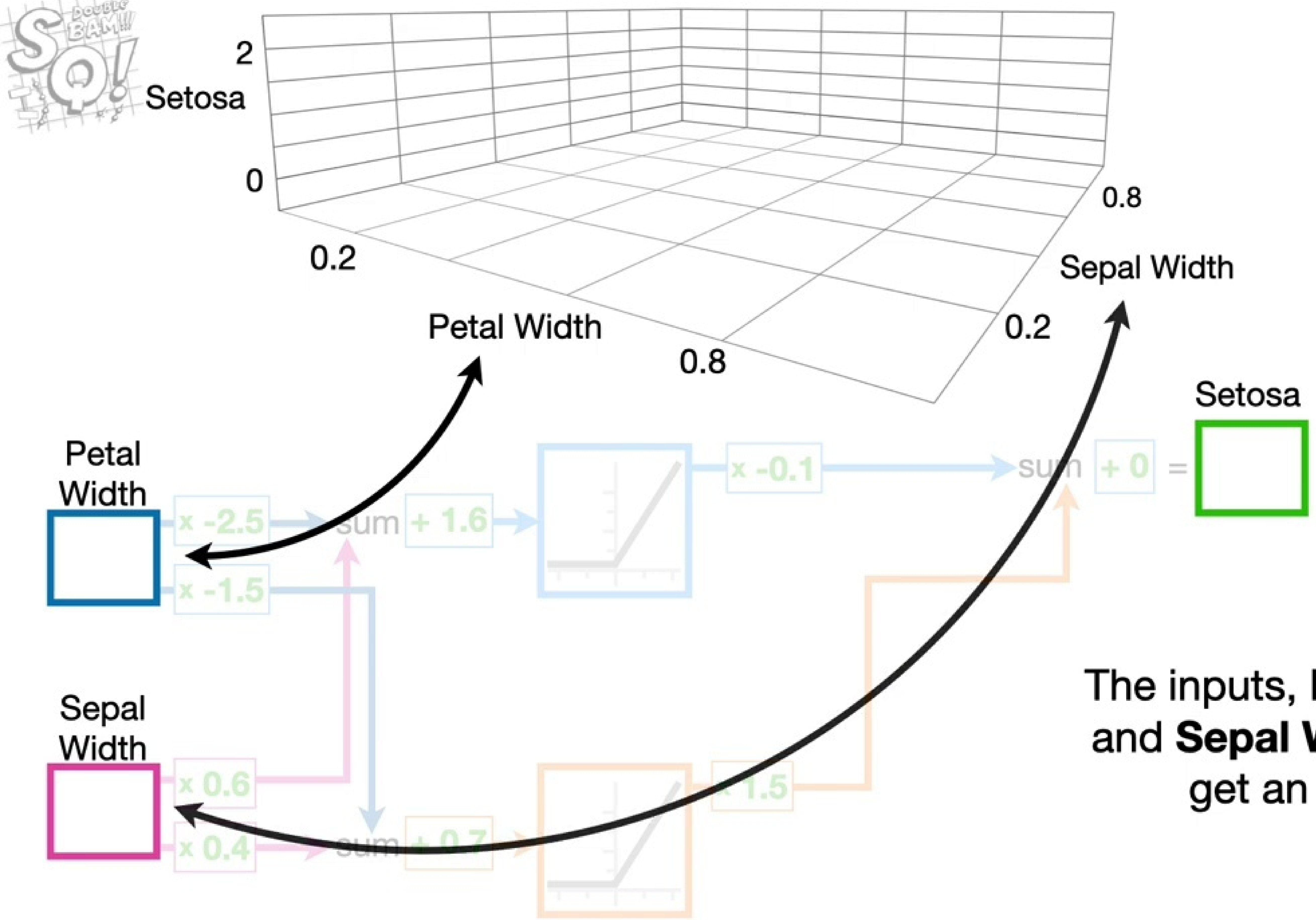

Since we have two inputs and one output, if we are going to draw a graph of what's going on…
Petal Width
Sepal Width
x -2.5
x -1.5
x 0.6
x 0.4
sum
+ 1.6
sum
+ 0.7
x -0.1
x 1.5
sum
+ 0
=
Setosa



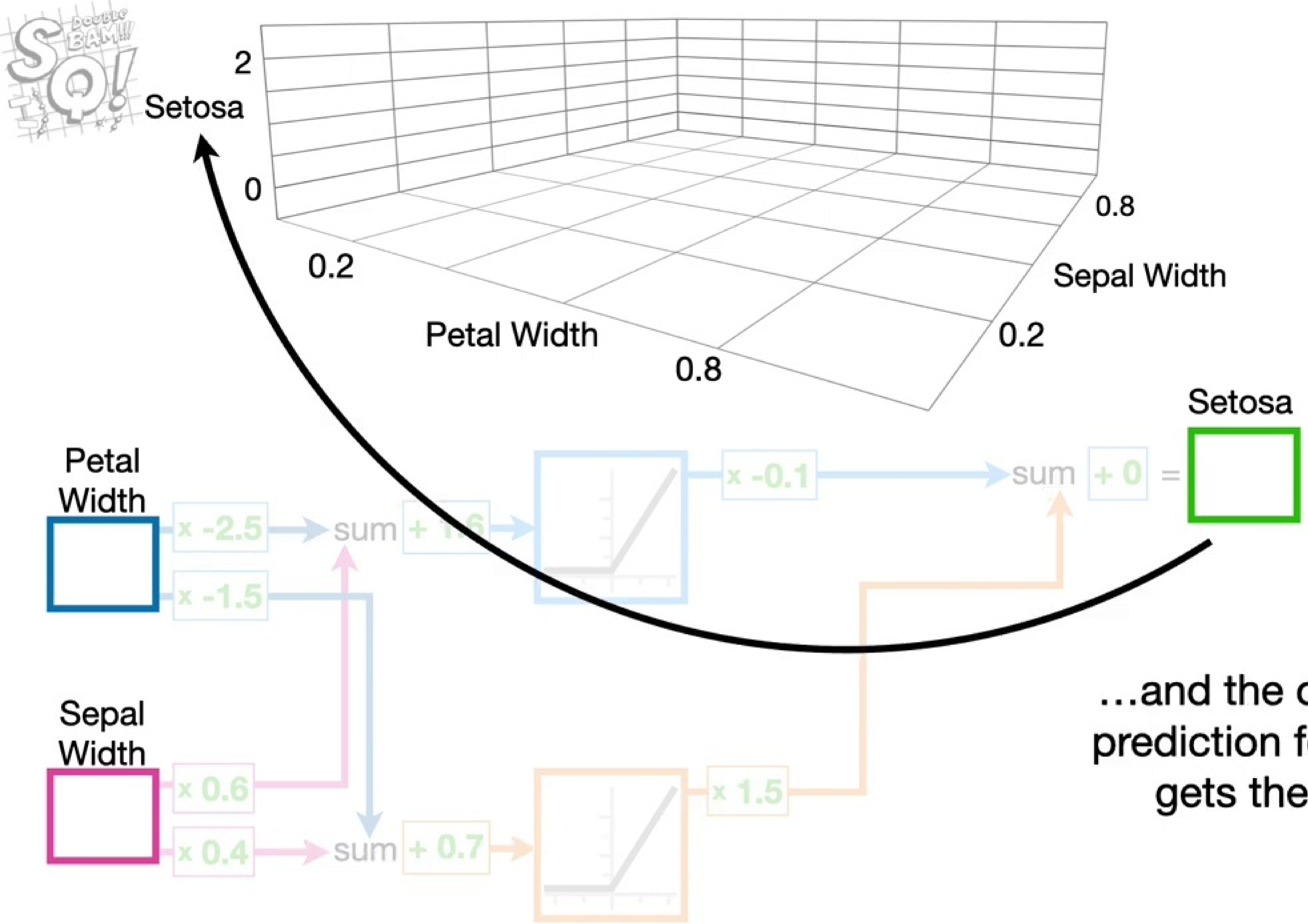
…then we need a **3-Dimensional** graph.



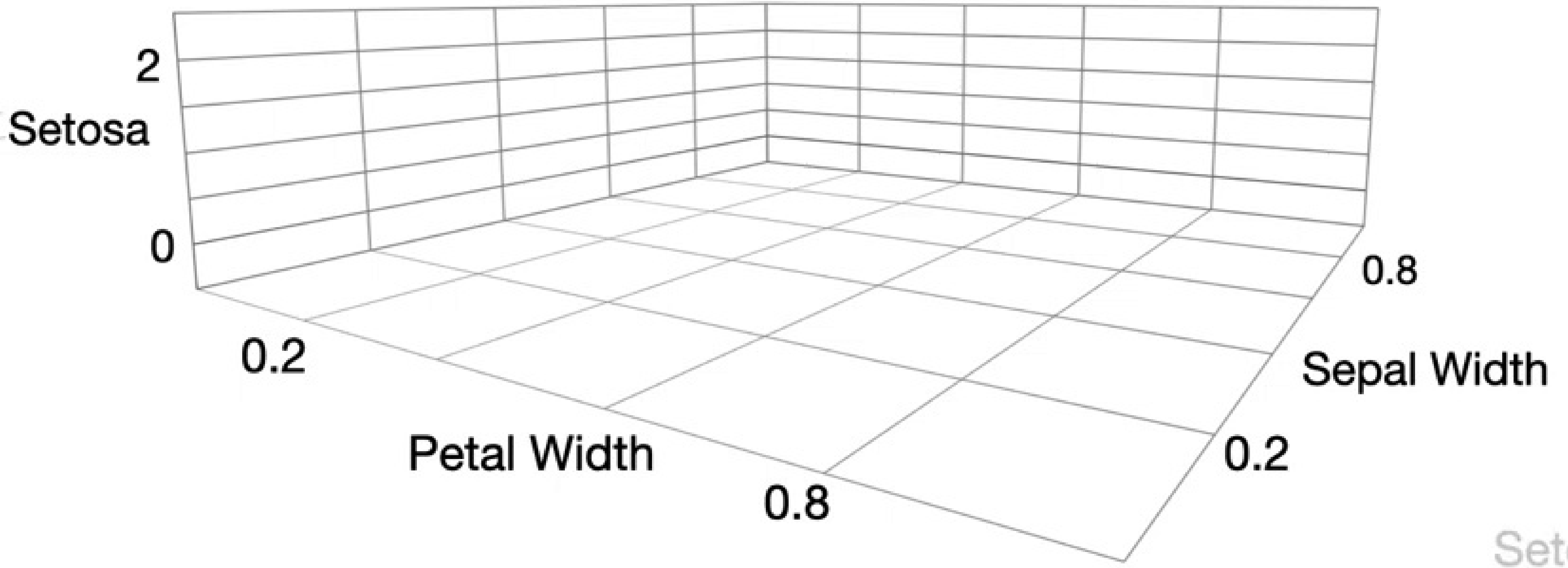
The inputs, **Petal Width** and **Sepal Width**, each get an axis…

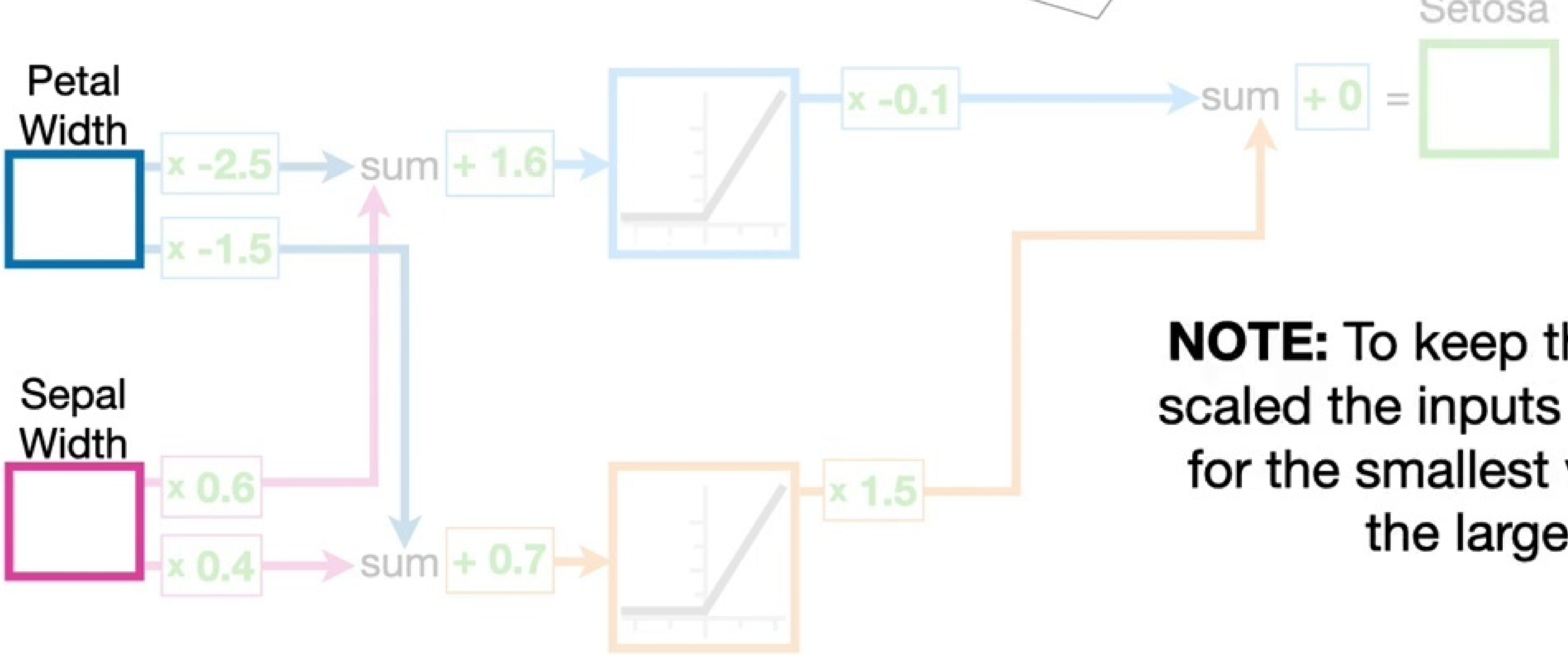


The inputs, **Petal Width** and **Sepal Width**, each get an axis…

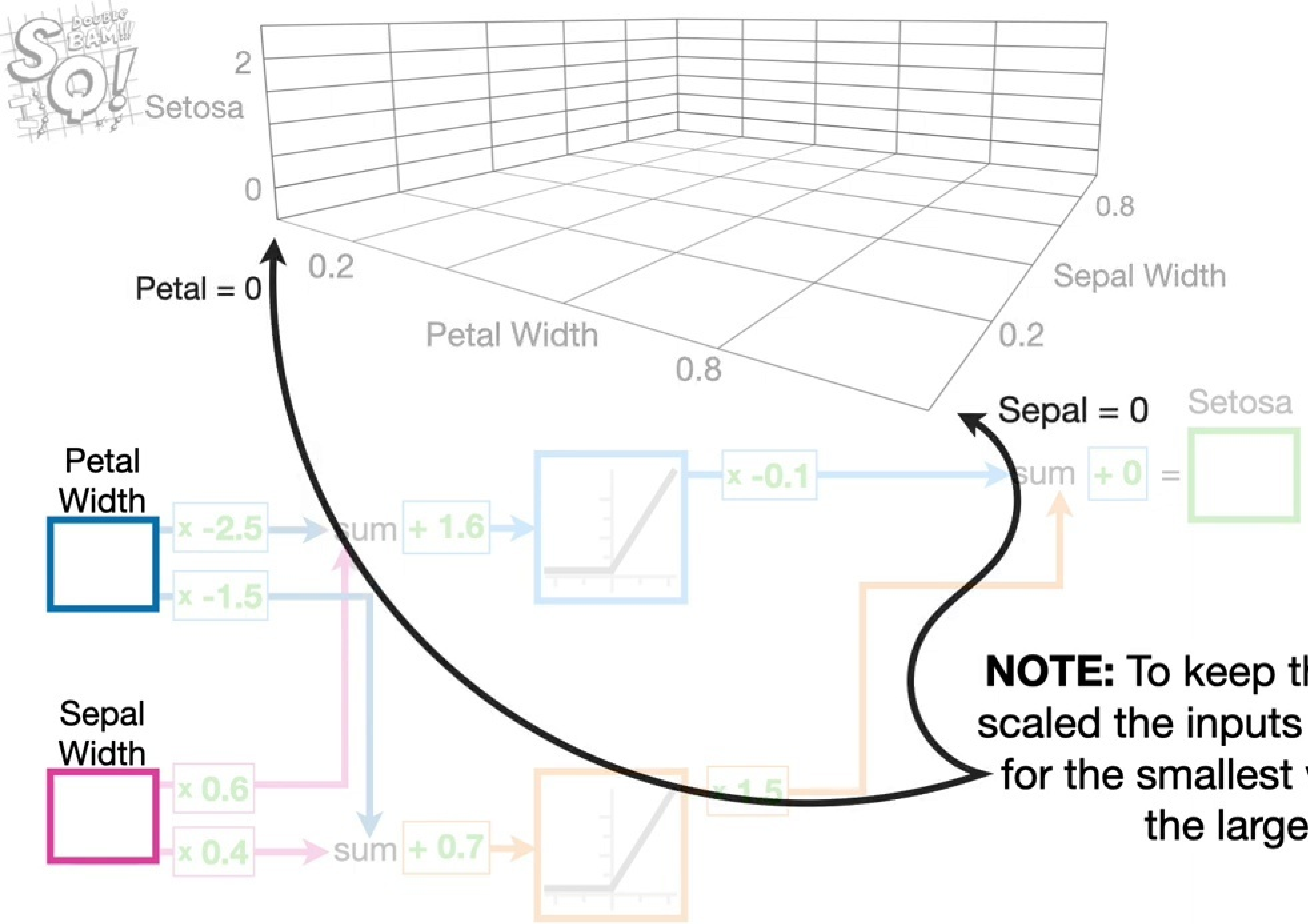


...and the output, the prediction for **Setosa**, gets the y-axis.

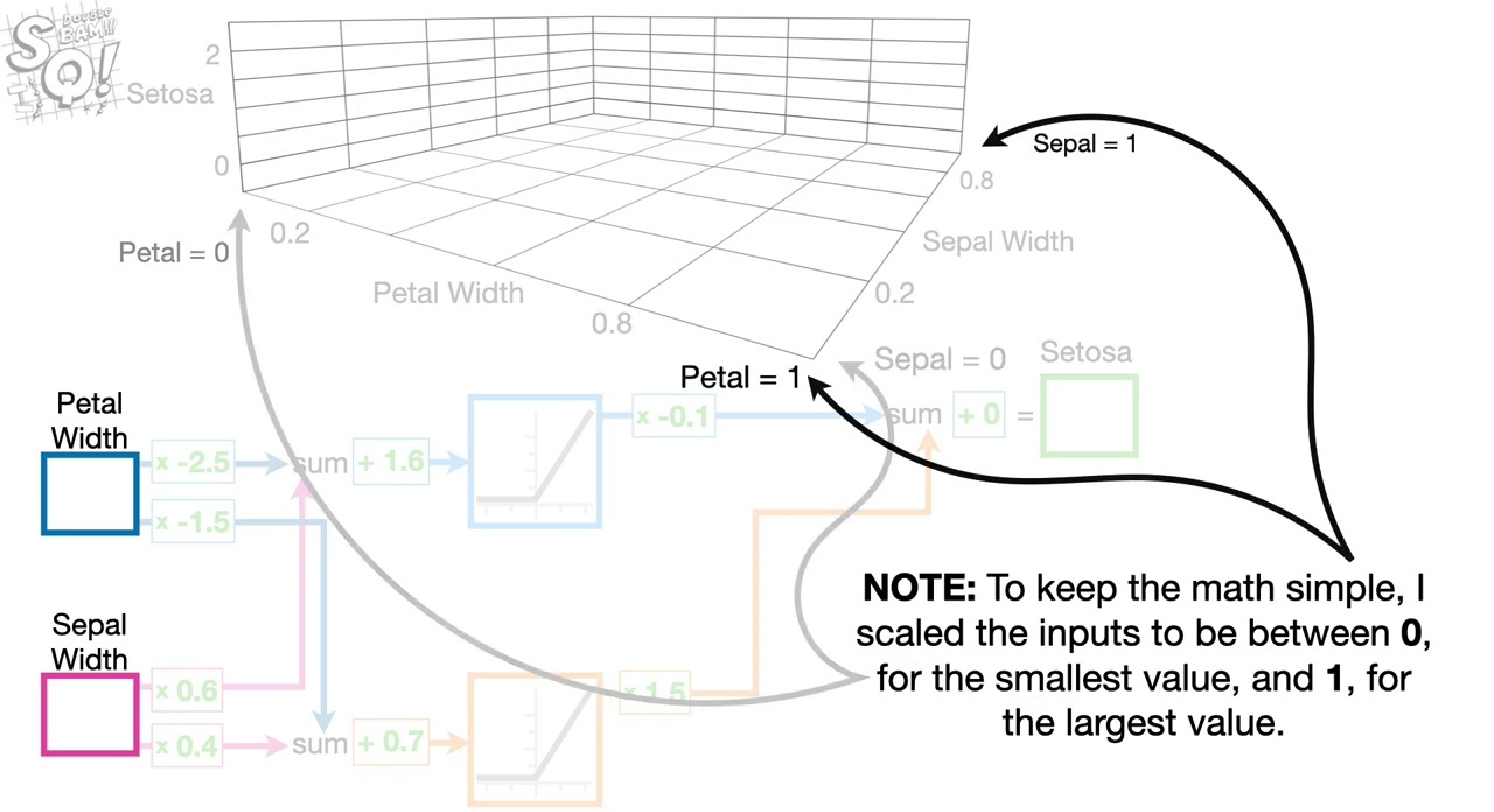




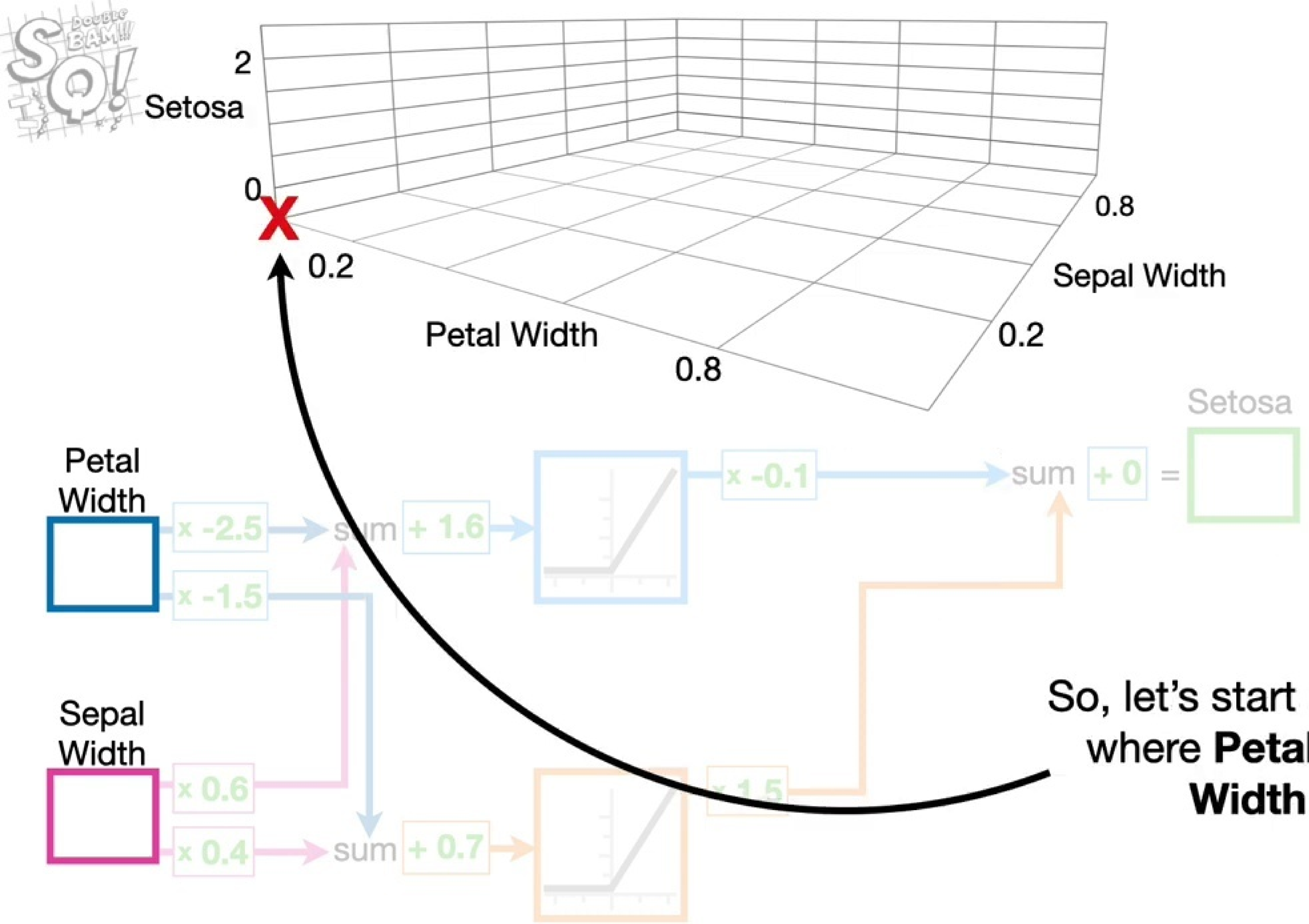
NOTE: To keep the math simple, I scaled the inputs to be between **0**, for the smallest value, and **1**, for the largest value.



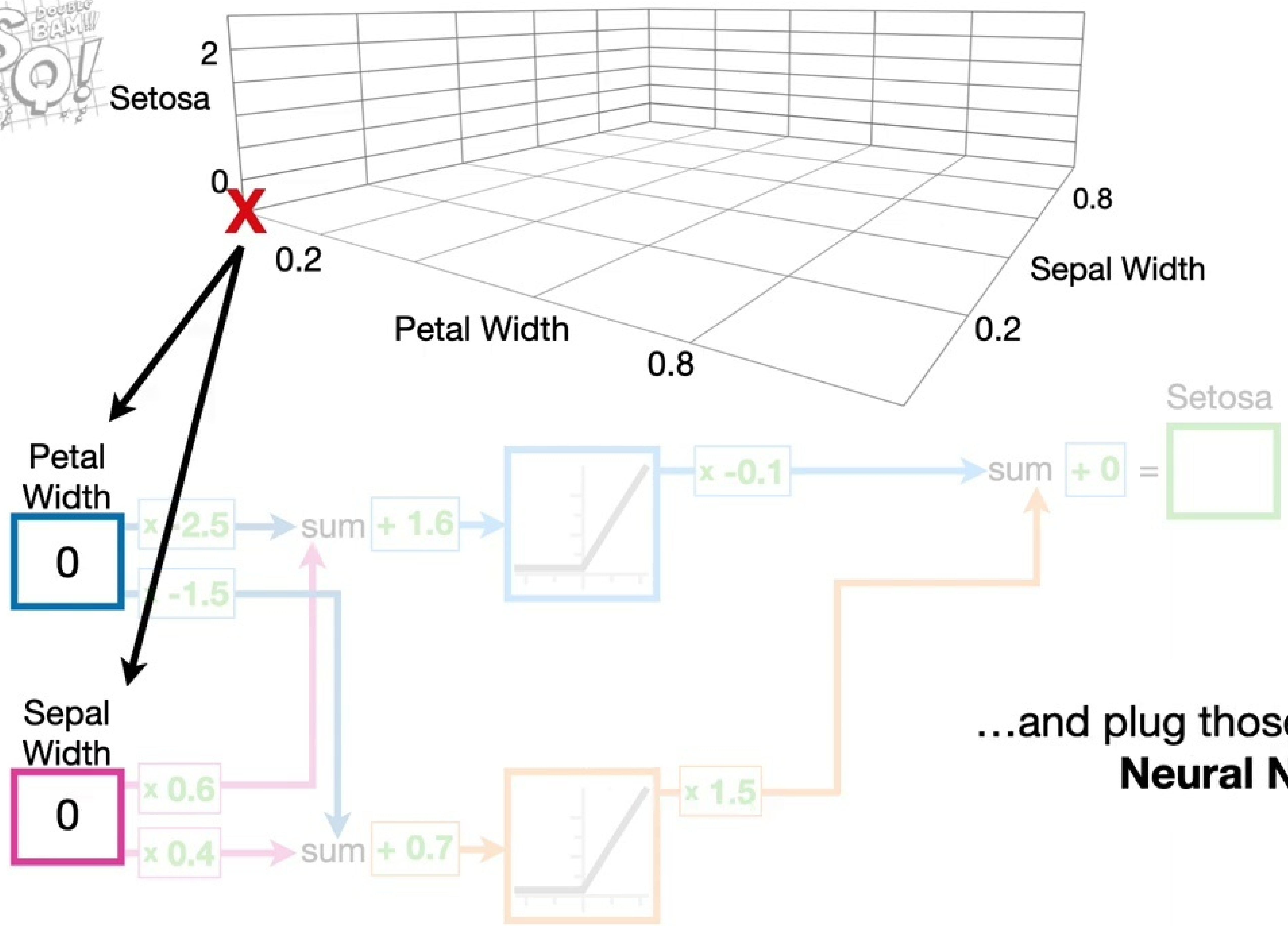
NOTE: To keep the math simple, I scaled the inputs to be between **0**, for the smallest value, and **1**, for the largest value.



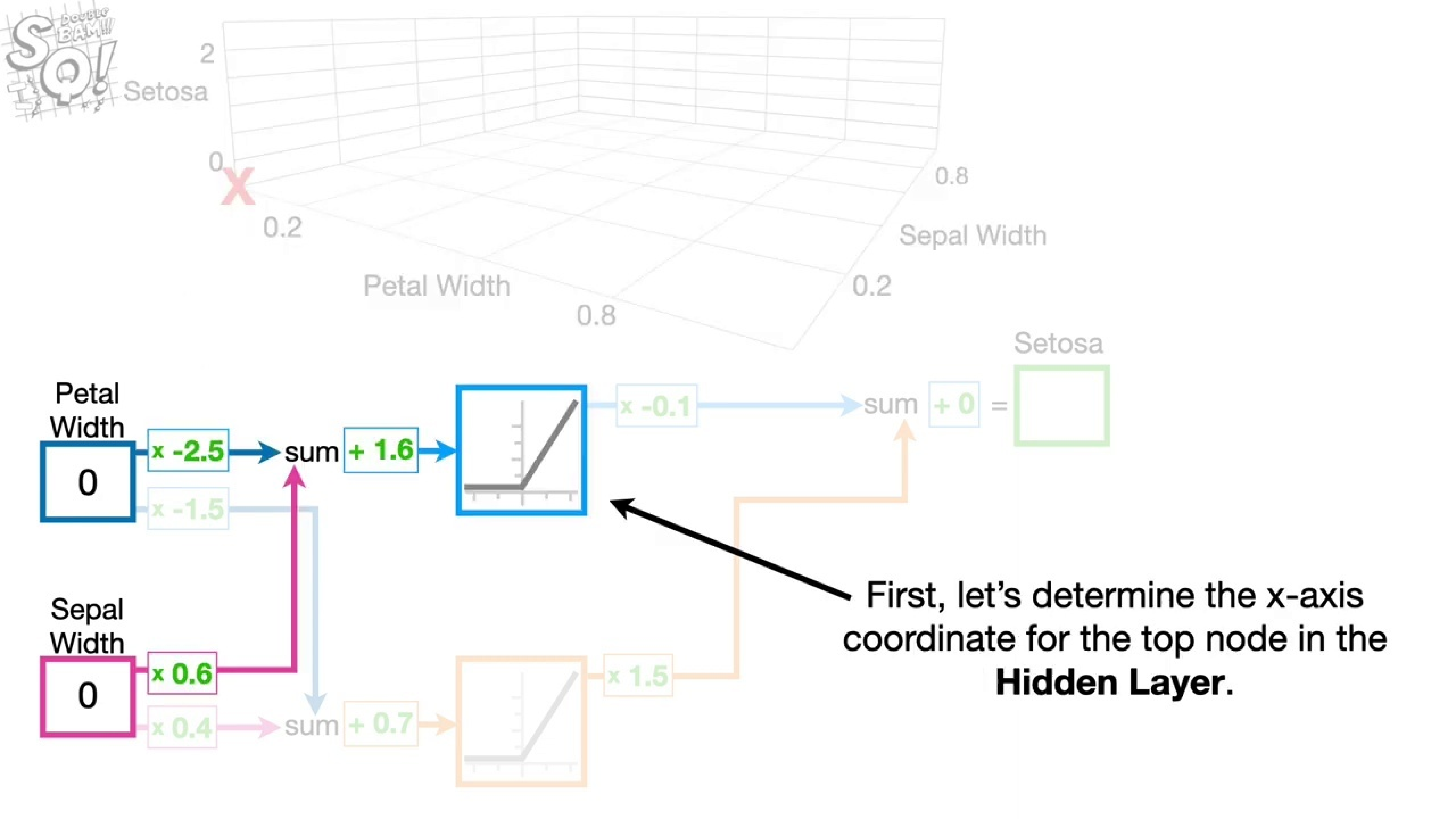
2
Setosa
0
Sepal = 1
0.8
0.2
Petal = 0
Sepal Width
Petal Width
0.2
0.8
Sepal = 0
Setosa
Petal = 1
Petal
Width
x -0.1
sum
+ 0
=
x -2.5
sum
+ 1.6
x -1.5
Sepal
Width
x 0.6
x 1.5
x 0.4
sum
+ 0.7
NOTE: To keep the math simple, I scaled the inputs to be between **0**, for the smallest value, and **1**, for the largest value.



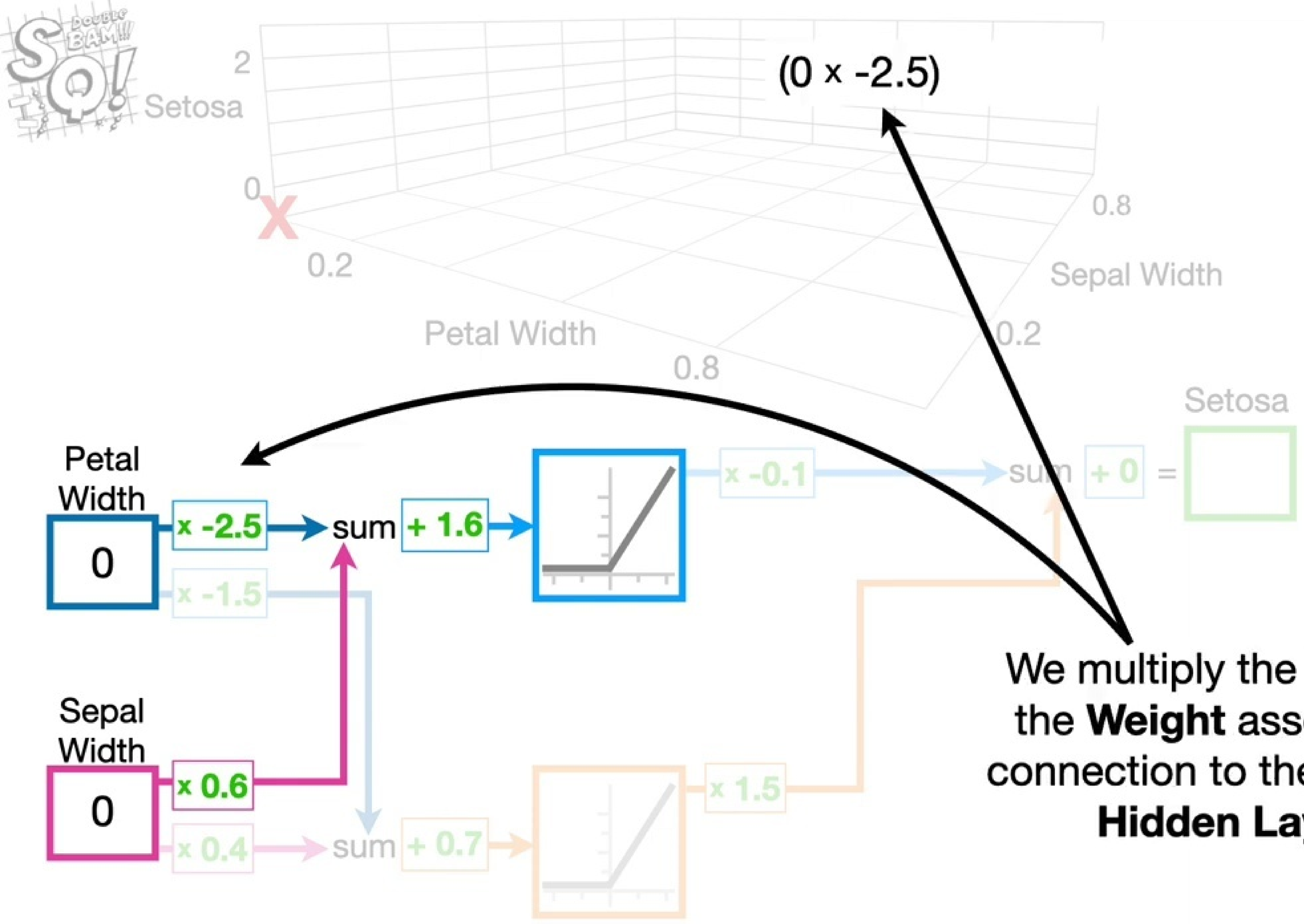
So, let's start in this corner, where **Petal** and **Sepal Width = 0**…



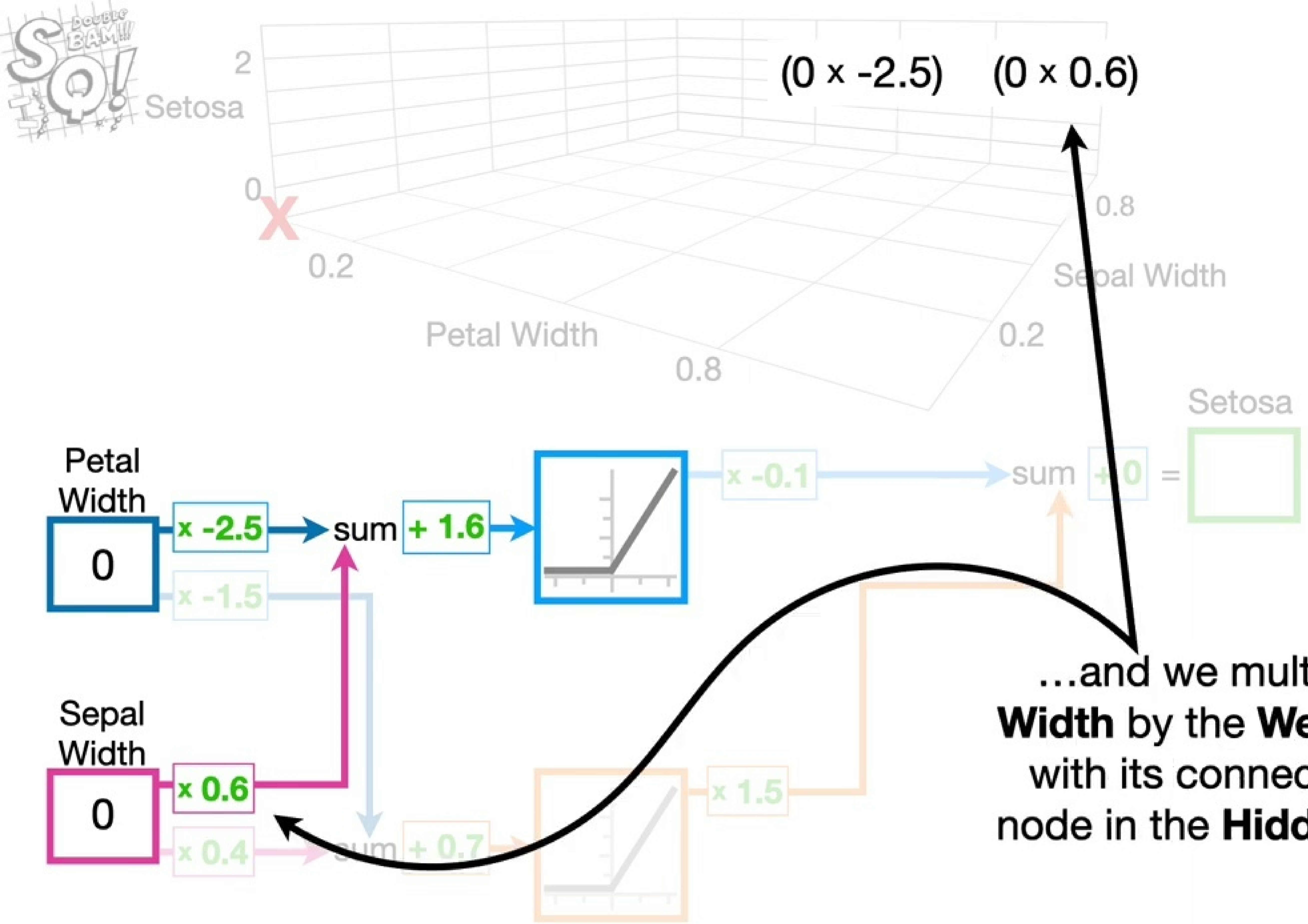
…and plug those values into the **Neural Network**.



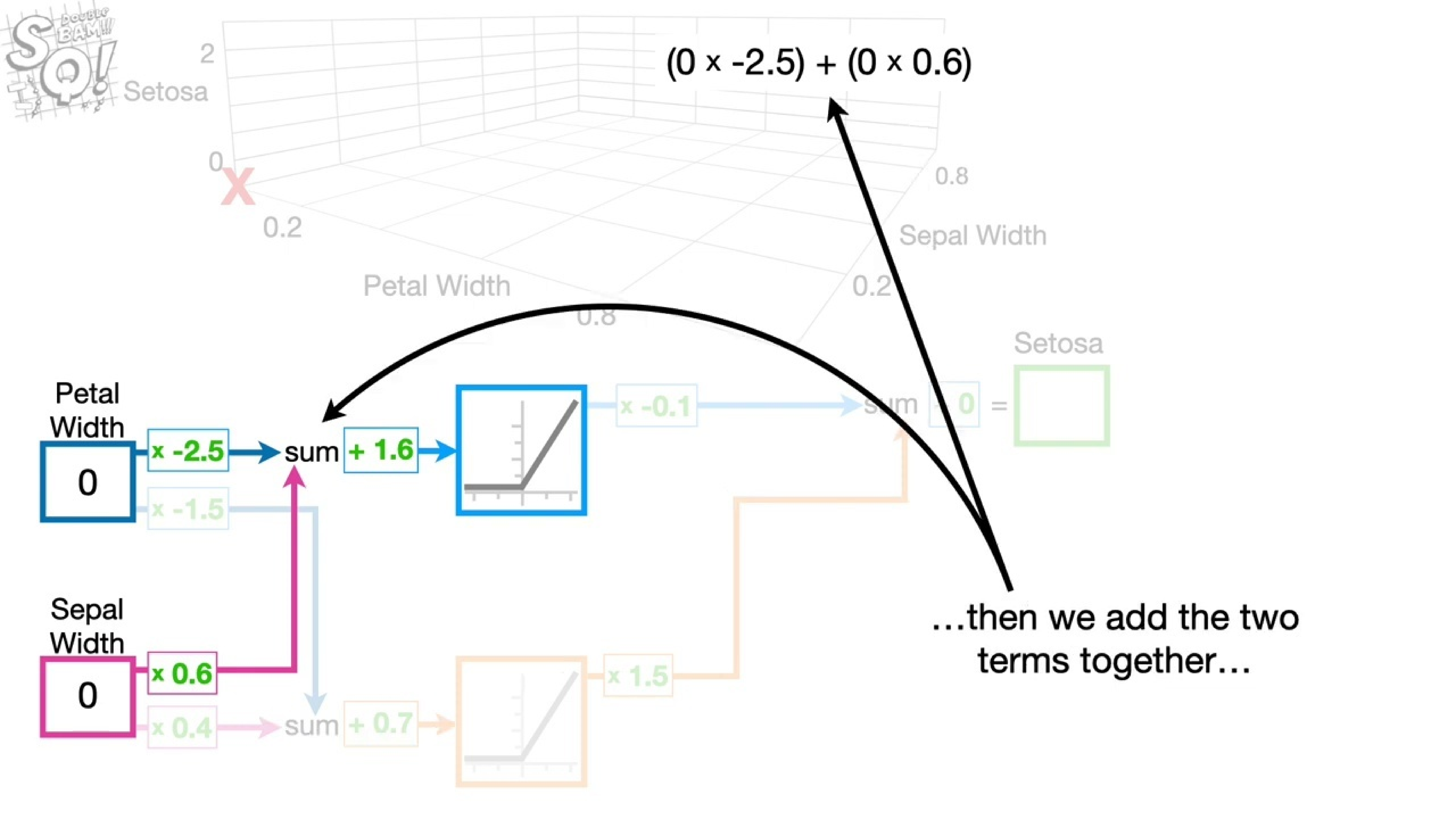
2
Setosa
0
0.2
Petal Width
0.8
0.8
Sepal Width
0.2
Setosa
Petal Width
0
x -2.5
sum
+ 1.6
x -1.5
x -0.1
sum
+ 0
=
Sepal Width
0
x 0.6
x 0.4
sum
+ 0.7
x 1.5
First, let's determine the x-axis coordinate for the top node in the **Hidden Layer**.



We multiply the **Petal Width** by the **Weight** associated with its connection to the top node in the **Hidden Layer**, **-2.5**…

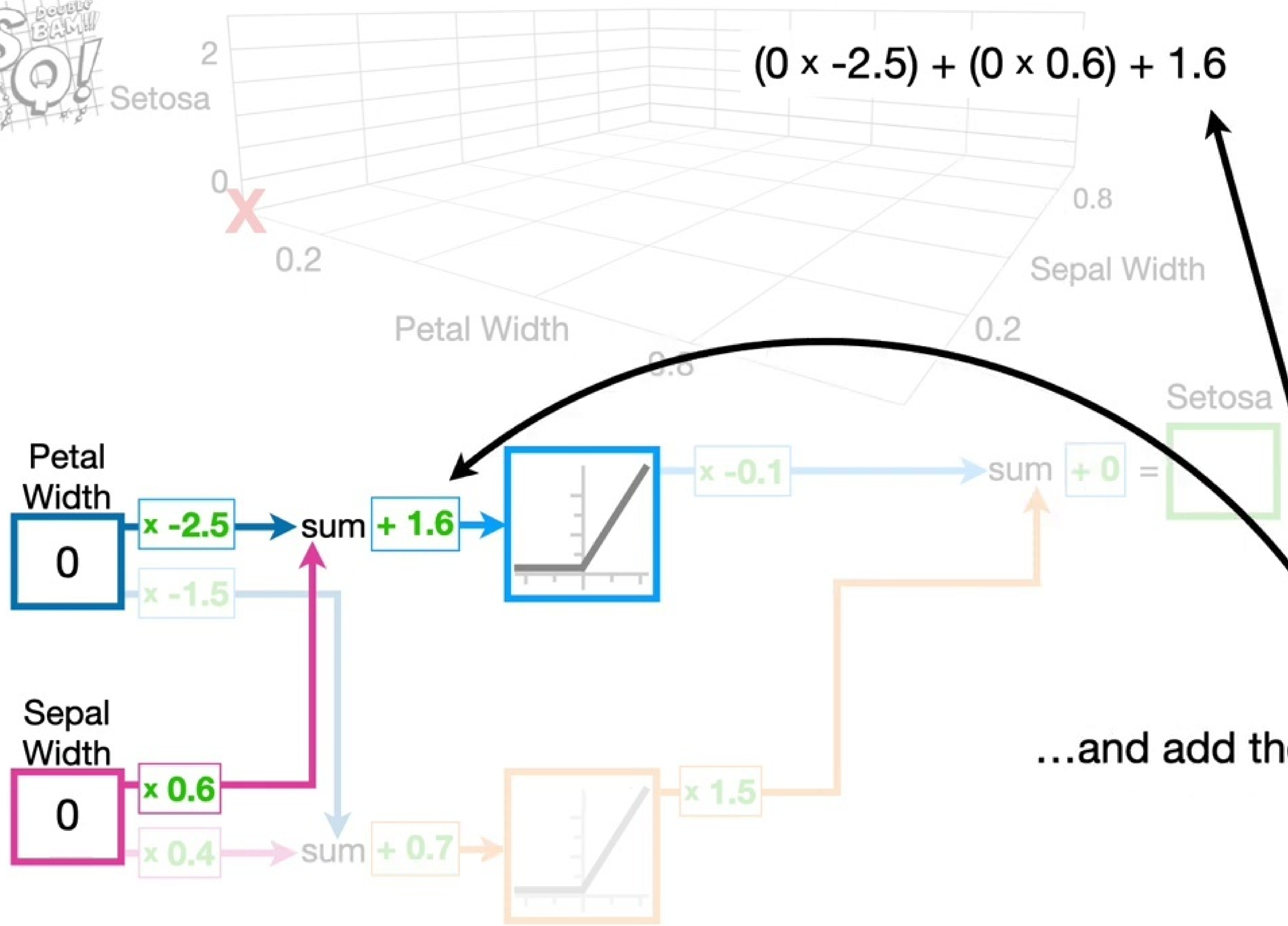


…and we multiply the **Sepal Width** by the **Weight** associated with its connection to the top node in the **Hidden Layer**, **0.6**…

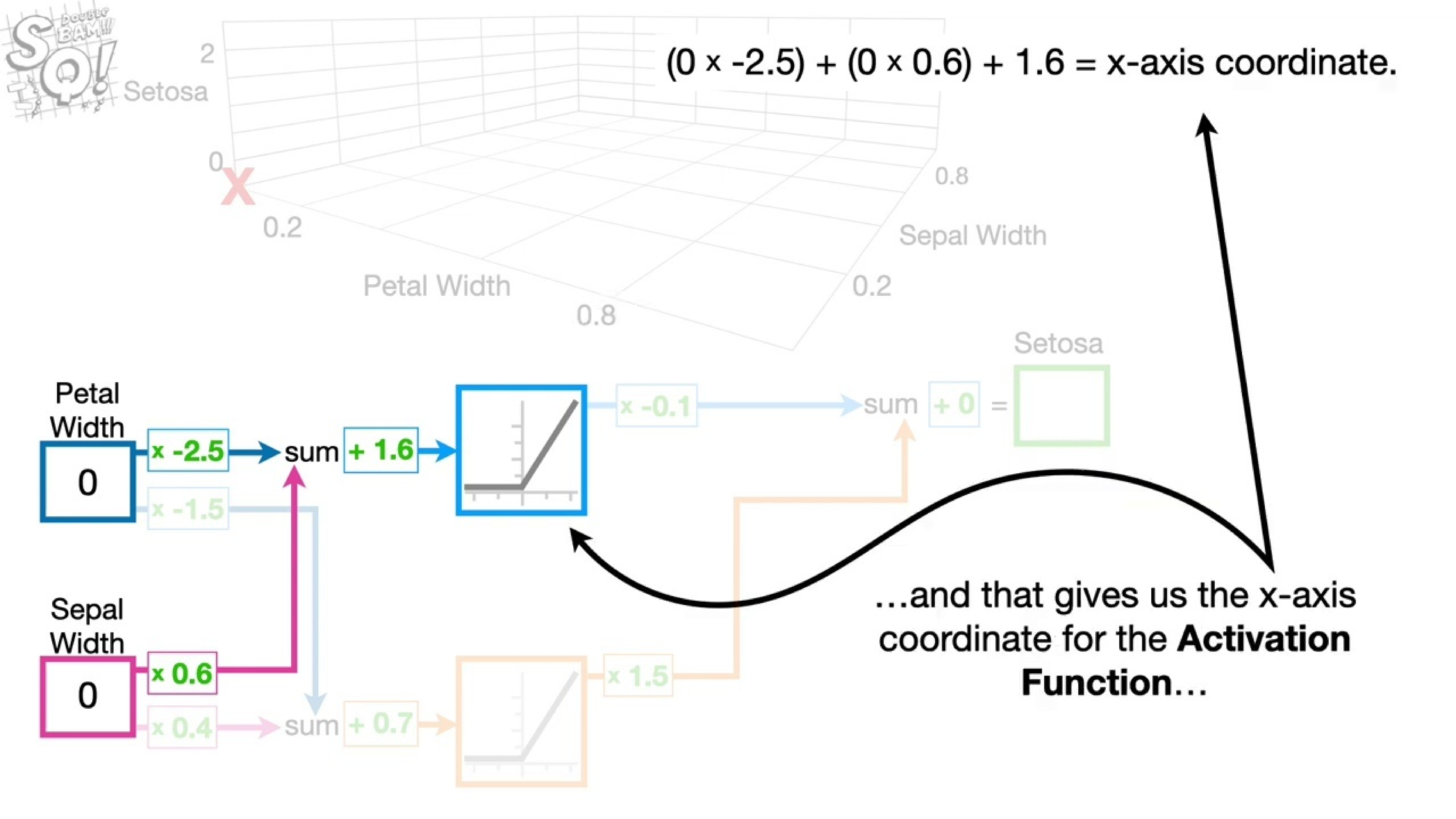


(0 x -2.5) + (0 x 0.6)
Setosa
Petal Width
Sepal Width
Petal Width
0
x -2.5
sum
+ 1.6
x -1.5
x -0.1
sum
0
=
Setosa
Sepal Width
0
x 0.6
x 0.4
sum
+ 0.7
x 1.5
…then we add the two terms together…

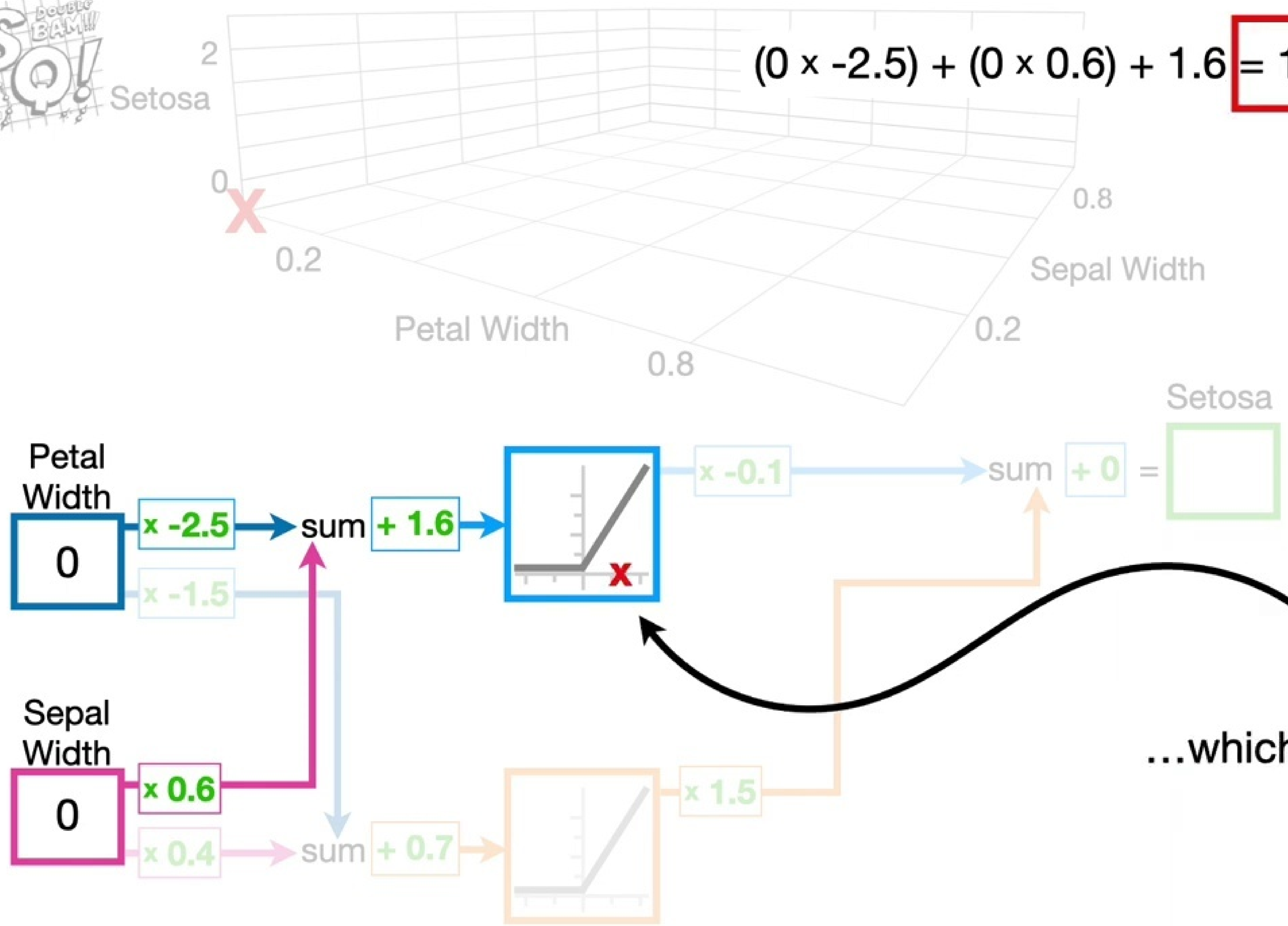
$$(0 \times -2.5) + (0 \times 0.6) + 1.6$$



…and add the **Bias, 1.6**…


(0 x -2.5) + (0 x 0.6) + 1.6 = x-axis coordinate.
Setosa
2
0
0.2
Petal Width
0.8
0.8
Sepal Width
0.2
Petal Width
0
x -2.5
sum
+ 1.6
x -1.5
Sepal Width
0
x 0.6
x 0.4
sum
+ 0.7
x -0.1
sum
+ 0
=
Setosa
x 1.5
…and that gives us the x-axis coordinate for the **Activation Function**…

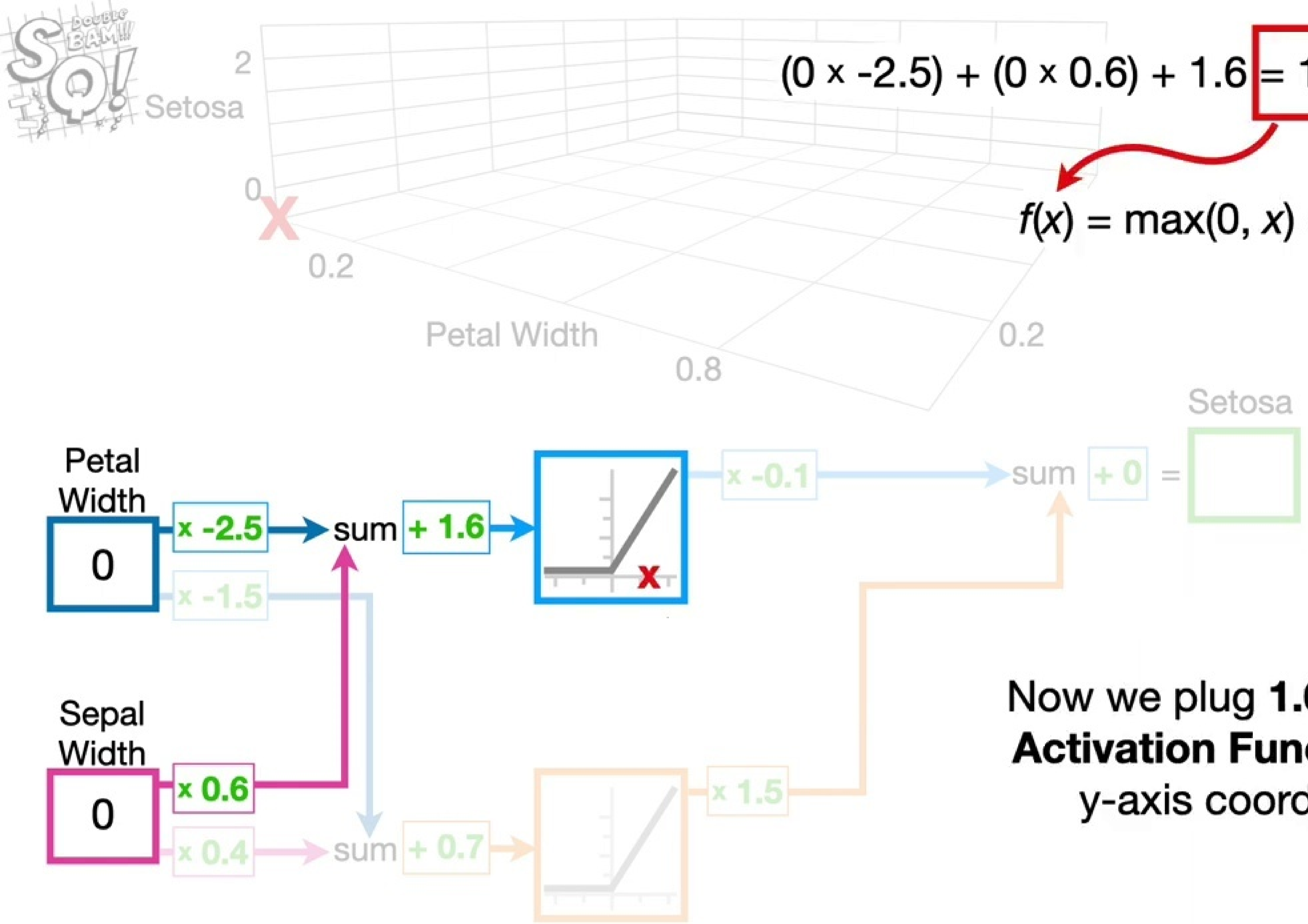
$(0 \times -2.5) + (0 \times 0.6) + 1.6 \boxed{= 1.6}$



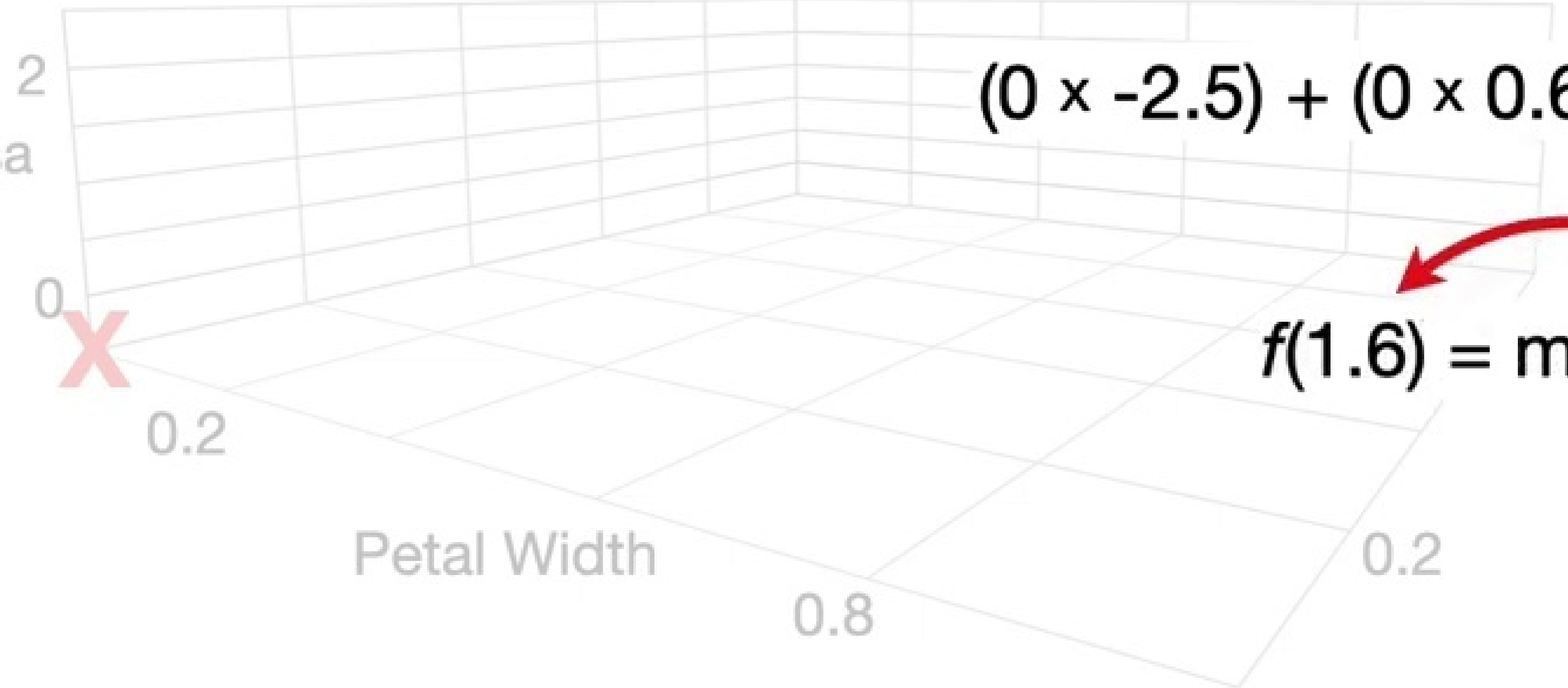
…which is **1.6**.

$$(0 \times -2.5) + (0 \times 0.6) + 1.6 \boxed{= 1.6}$$

$$f(x) = \max(0, x) = \text{y-axis coordinate}$$

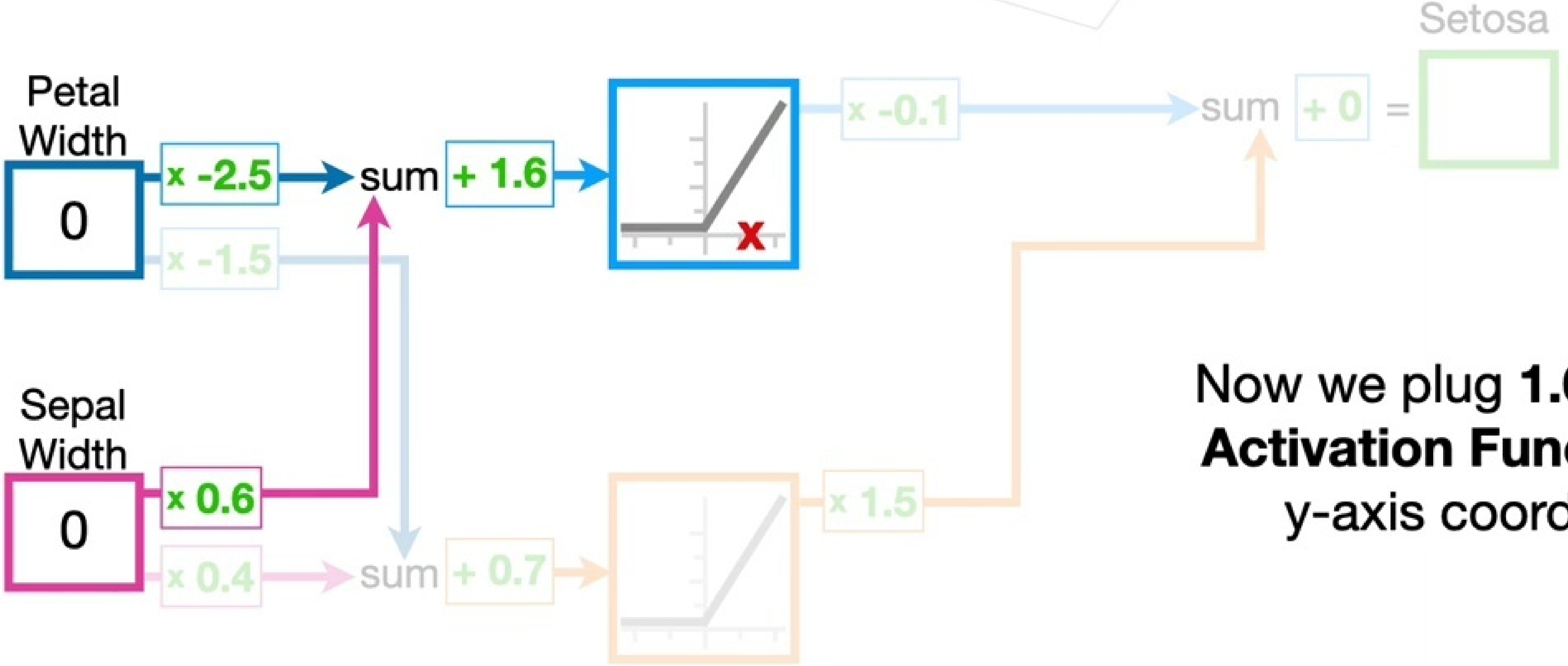


Now we plug **1.6** into the **ReLU Activation Function** to get the y-axis coordinate, **1.6**…

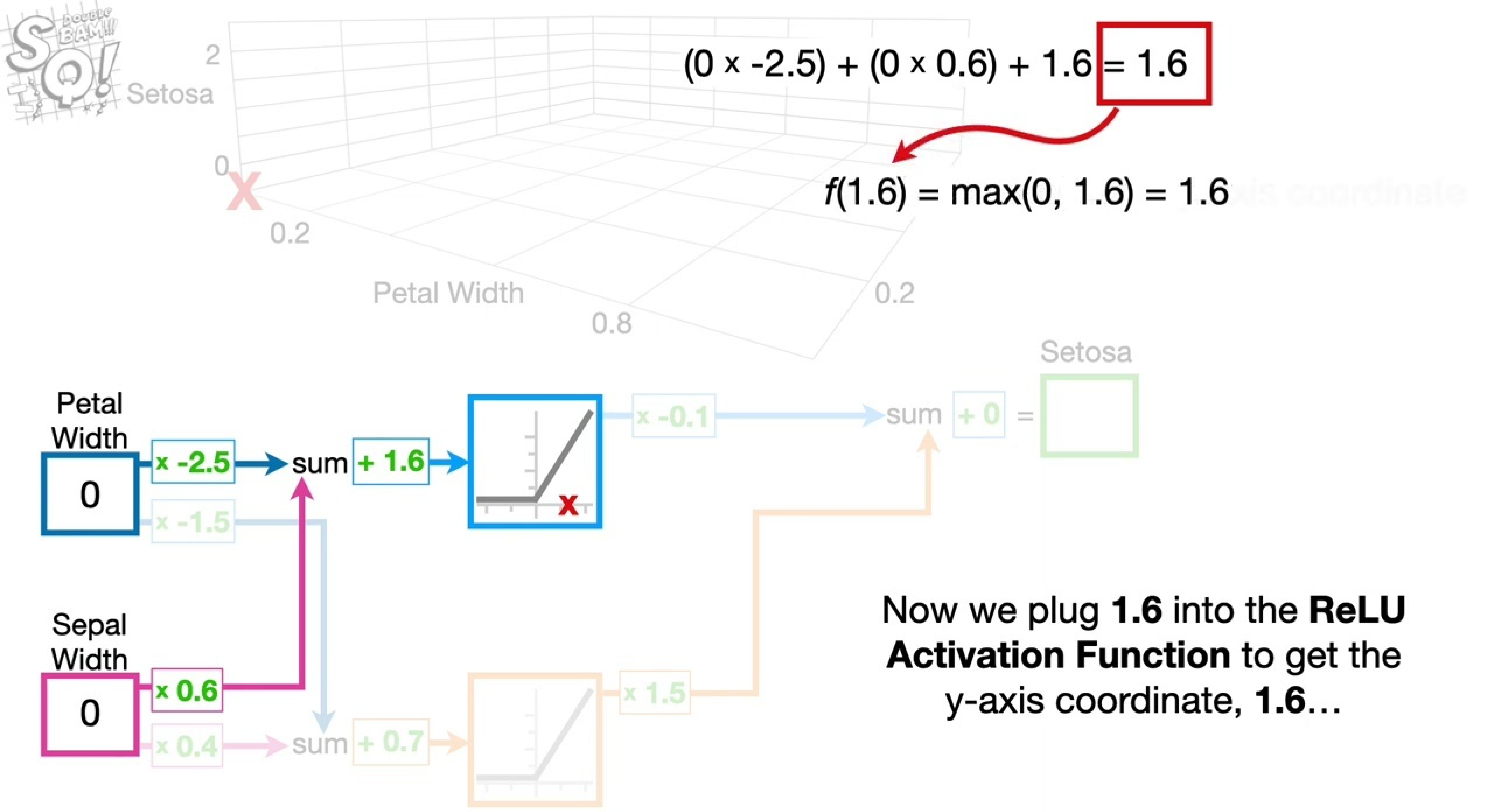


$(0 \times -2.5) + (0 \times 0.6) + 1.6 \boxed{= 1.6}$

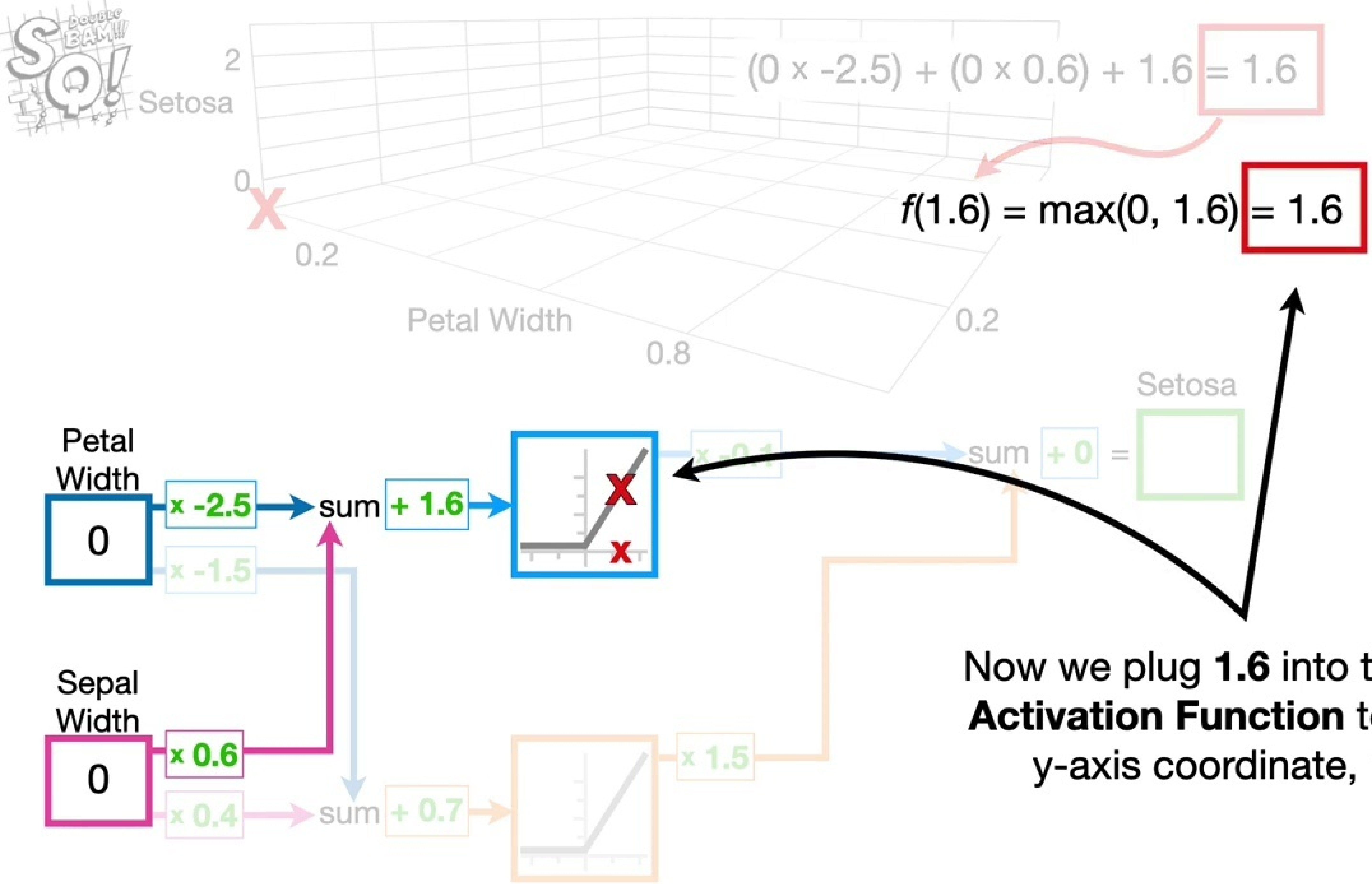
$f(1.6) = \max(0, 1.6) =$ y-axis coordinate



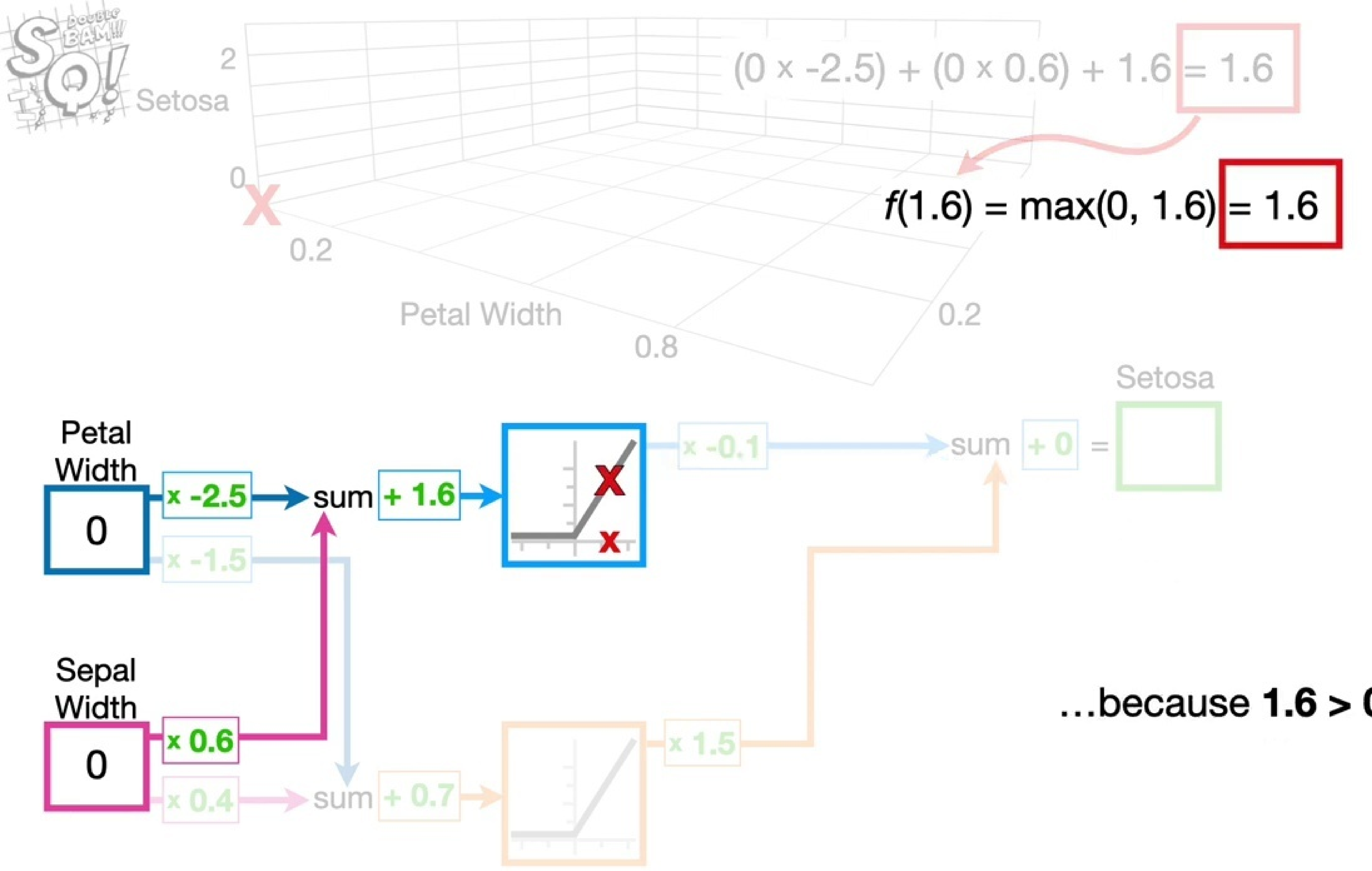
Now we plug **1.6** into the **ReLU Activation Function** to get the y-axis coordinate, **1.6**…



Now we plug **1.6** into the **ReLU Activation Function** to get the y-axis coordinate, **1.6**…

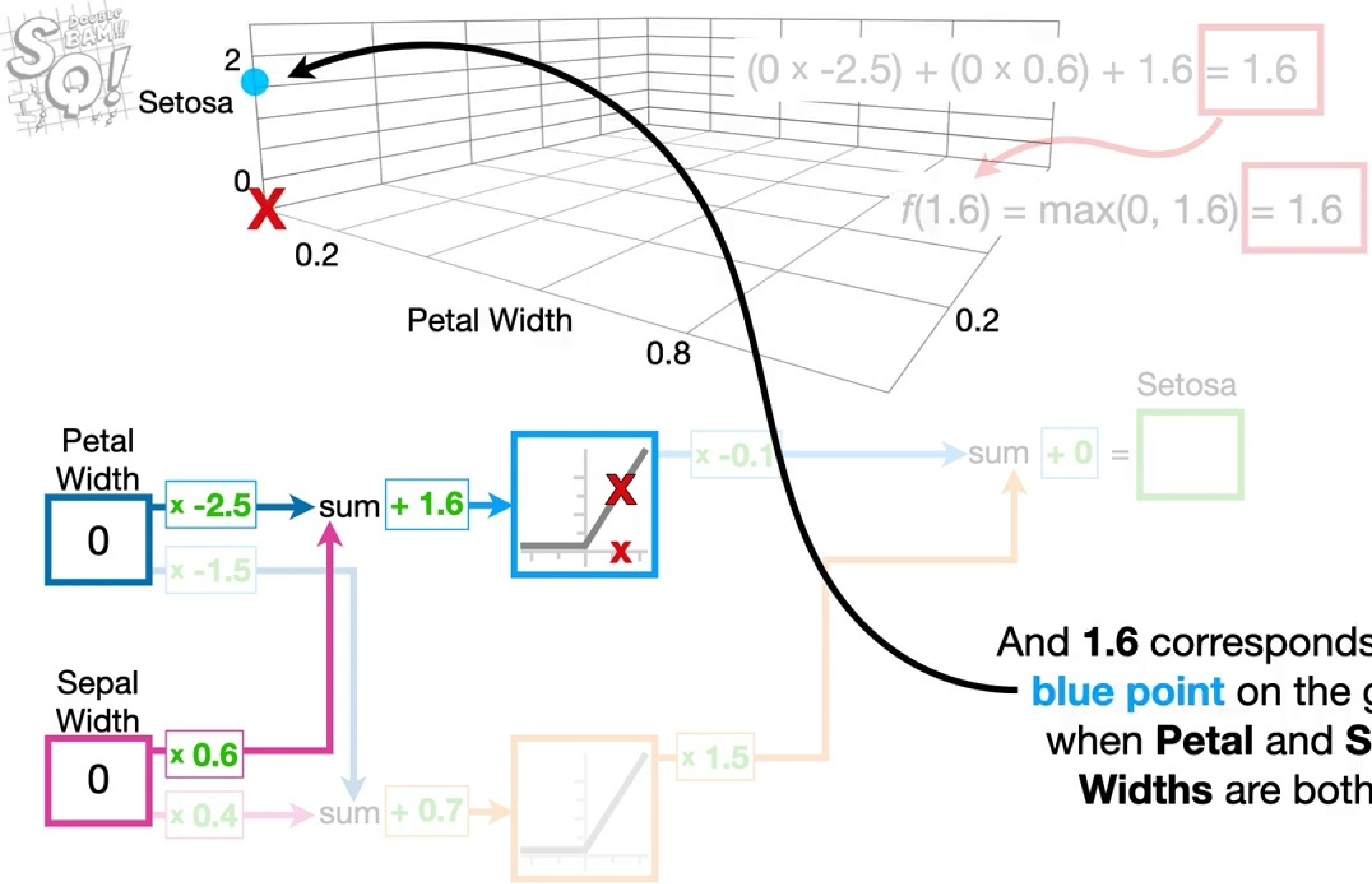


Now we plug **1.6** into the **ReLU Activation Function** to get the y-axis coordinate, **1.6**…

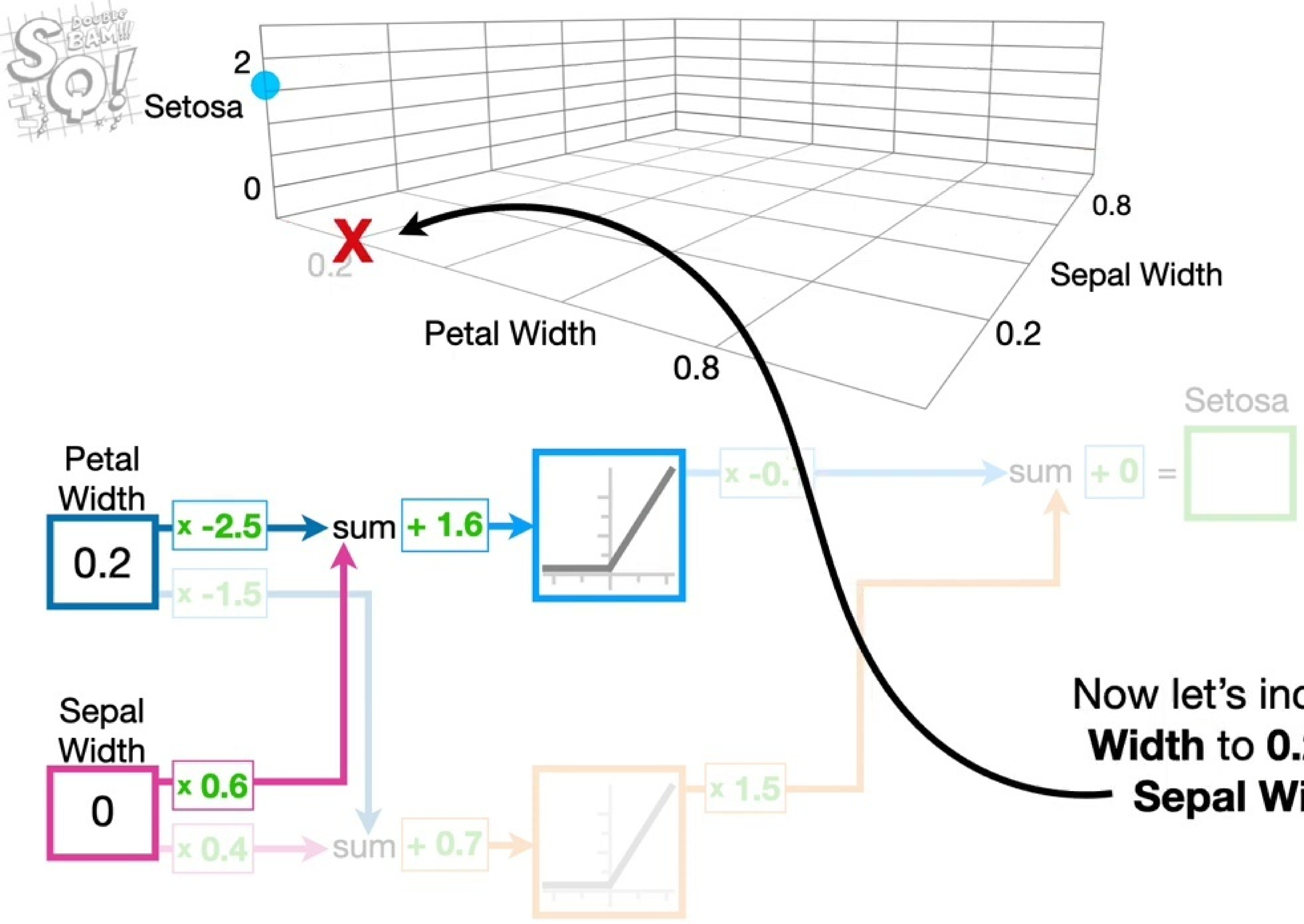


2
Setosa
0
0.2
Petal Width
0.8
0.2
(0 x -2.5) + (0 x 0.6) + 1.6 = 1.6
f(1.6) = max(0, 1.6) = 1.6
Setosa
Petal Width
0
Sepal Width
0
x -2.5
x -1.5
x 0.6
x 0.4
sum
+ 1.6
sum
+ 0.7
x -0.1
x 1.5
sum
+ 0
=

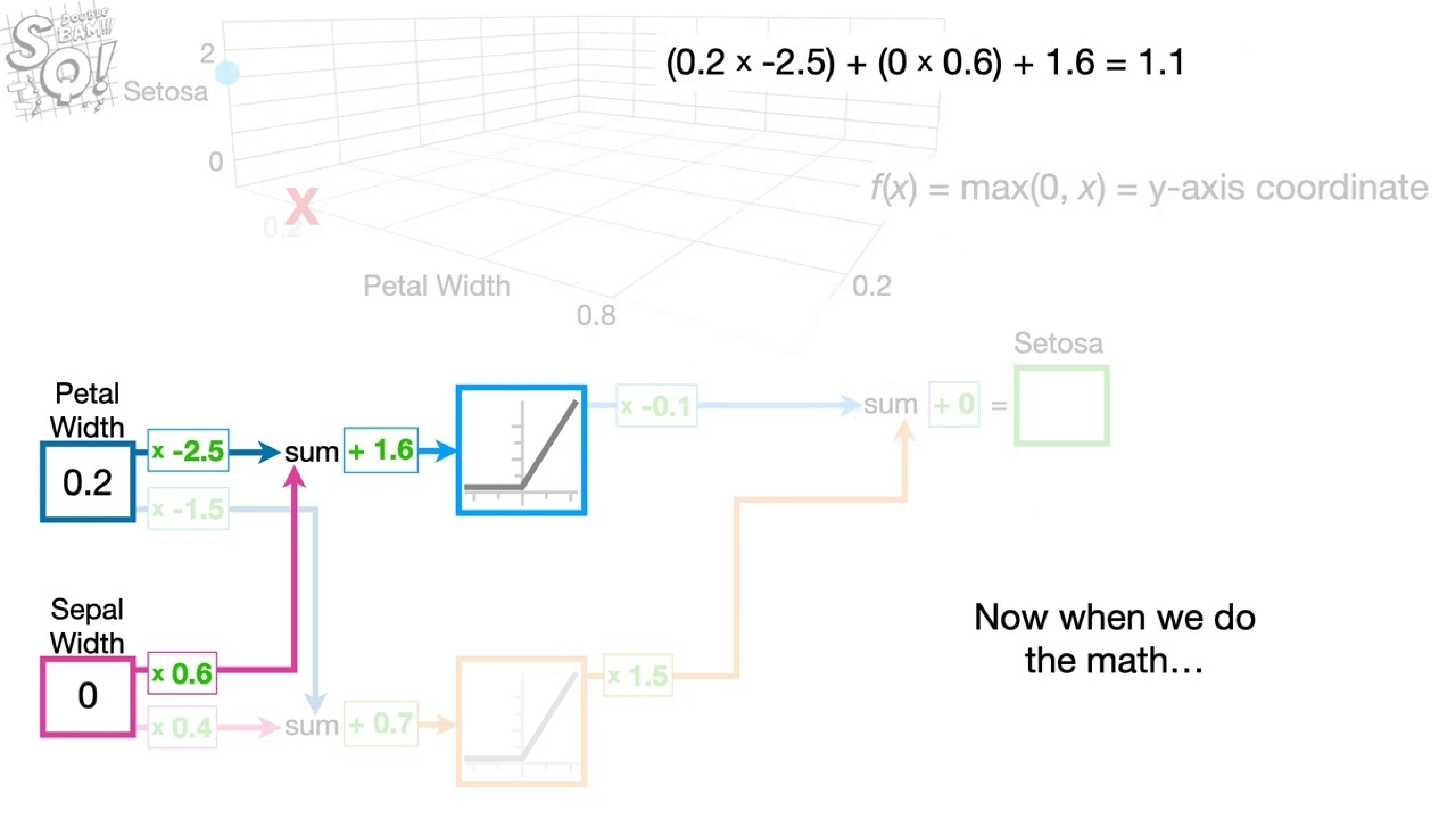
…because **1.6 > 0**.



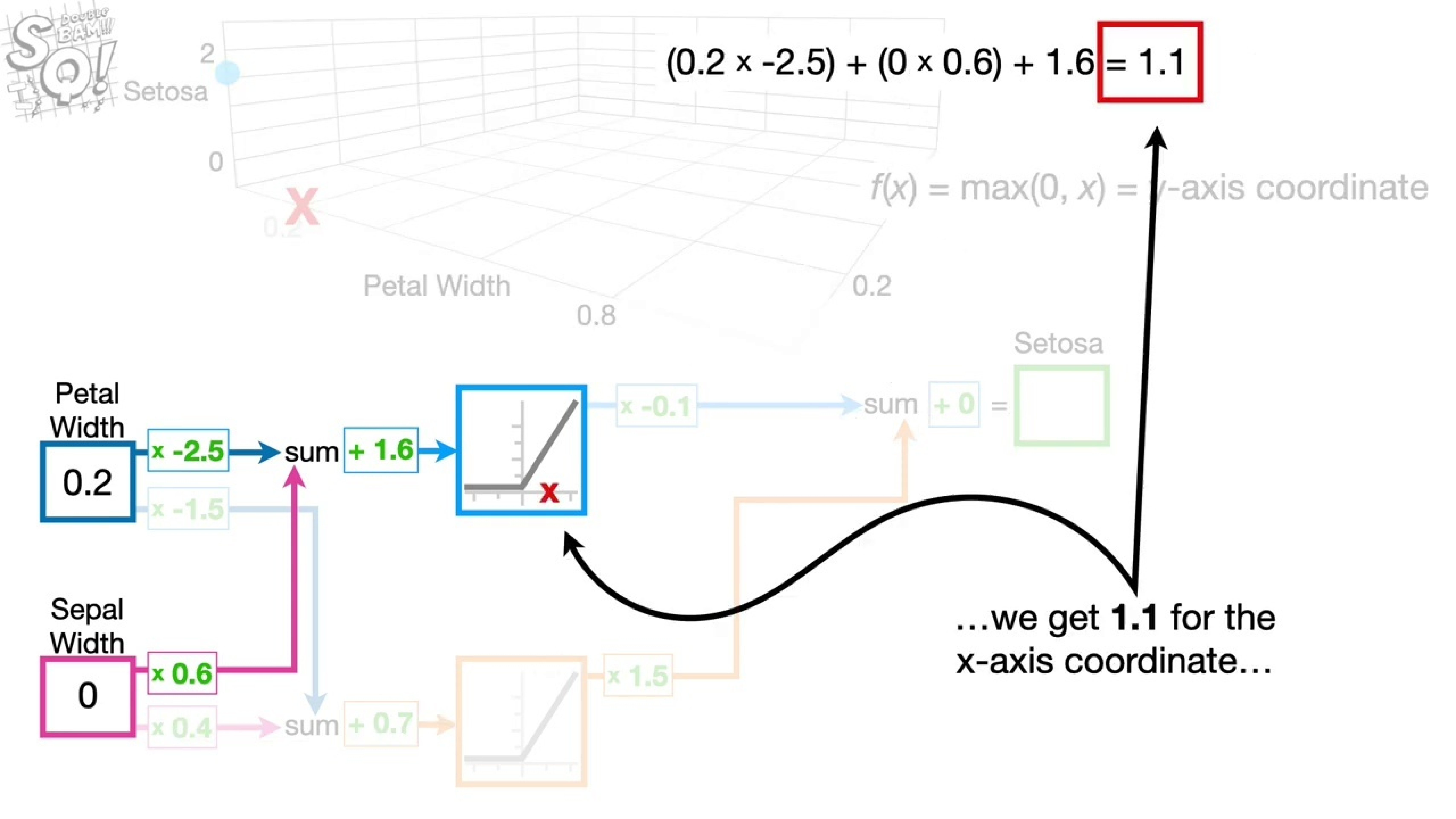
And **1.6** corresponds to this **blue point** on the graph when **Petal** and **Sepal Widths** are both **0**.

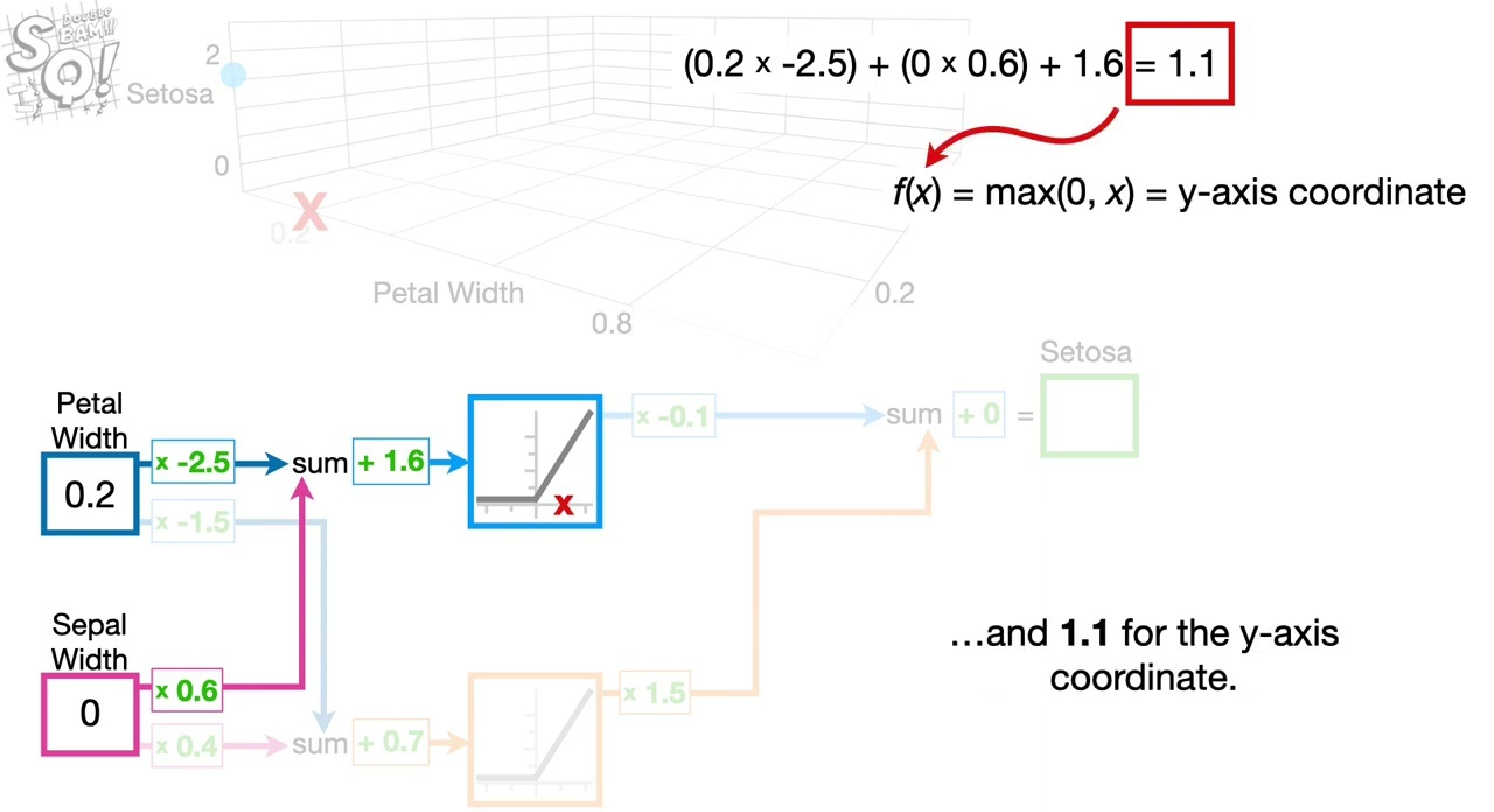


Now let's increase **Petal Width** to **0.2**, but keep **Sepal Width** at **0**.

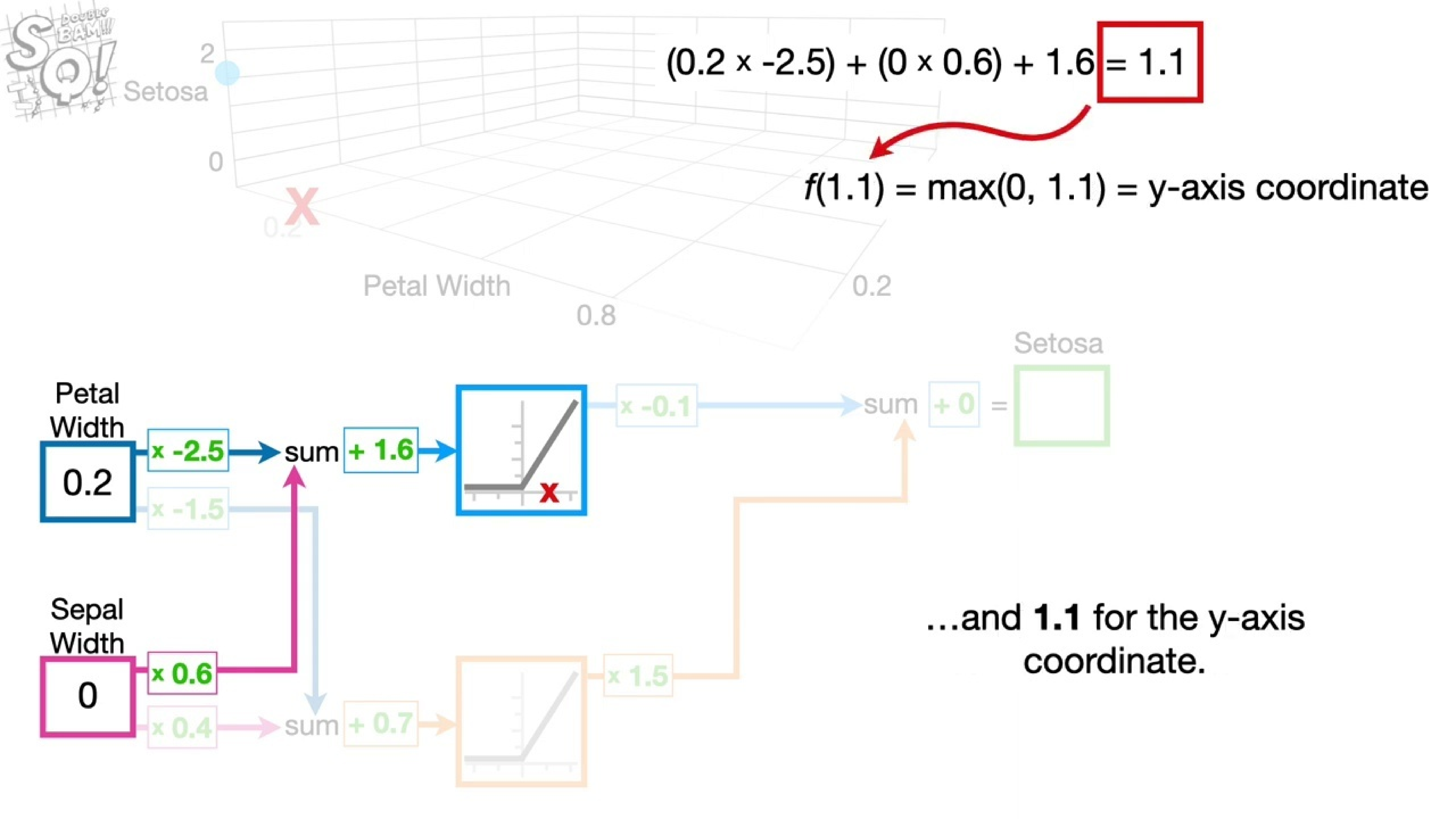


(0.2 x -2.5) + (0 x 0.6) + 1.6 = 1.1
2
0
Setosa
0.2
Petal Width
0.8
0.2
f(x) = max(0, x) = y-axis coordinate
Setosa
Petal Width
0.2
x -2.5
sum
+ 1.6
x -0.1
sum
+ 0
=
x -1.5
Sepal Width
0
x 0.6
x 0.4
sum
+ 0.7
x 1.5
Now when we do the math…


(0.2 x -2.5) + (0 x 0.6) + 1.6 = 1.1
f(x) = max(0, x) = y-axis coordinate
2
Setosa
0
0.2
Petal Width
0.8
0.2
Setosa
Petal Width
0.2
x -2.5
sum
+ 1.6
x -1.5
x -0.1
sum
+ 0
=
Sepal Width
0
x 0.6
x 0.4
sum
+ 0.7
x 1.5
…we get 1.1 for the x-axis coordinate…



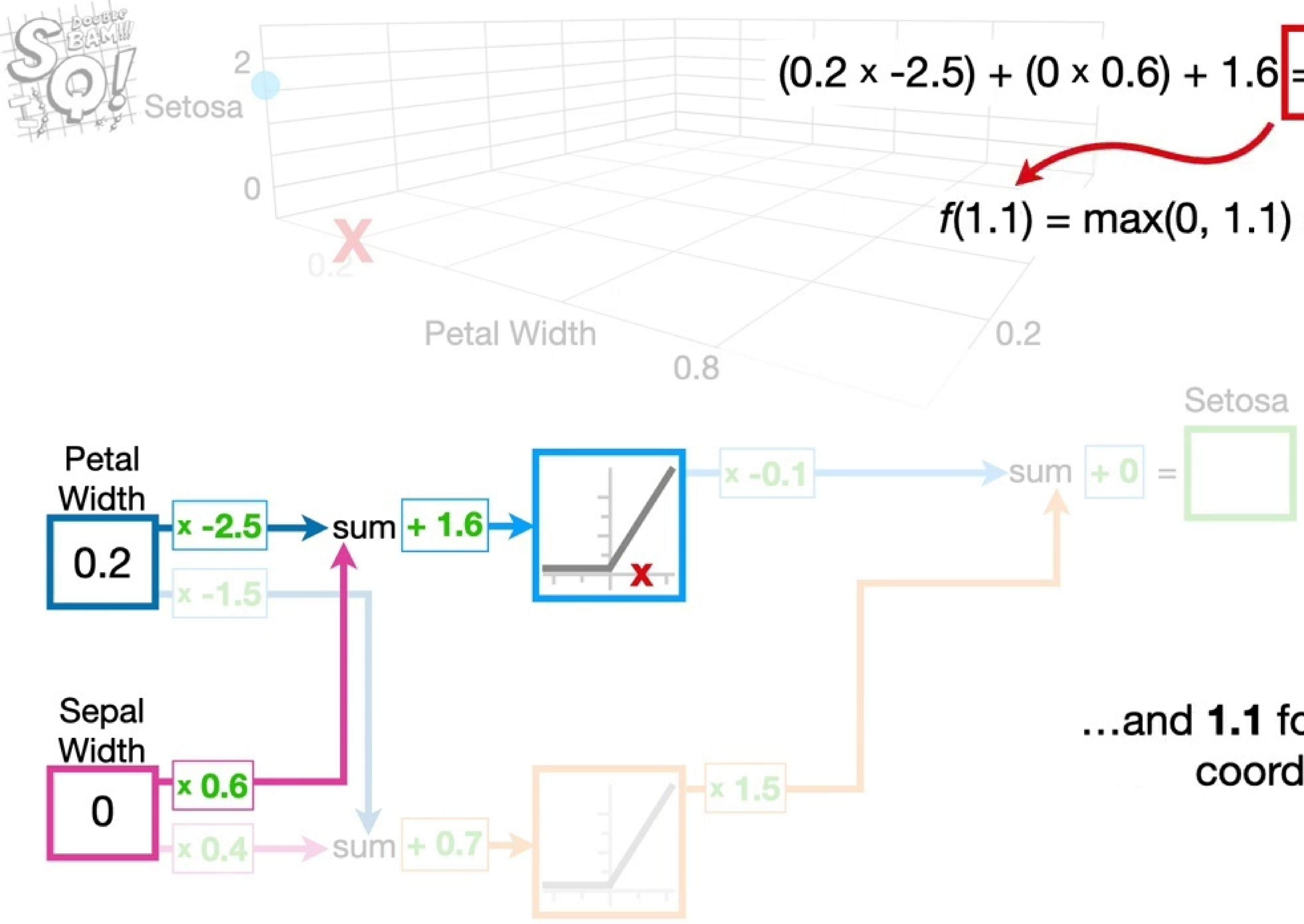
(0.2 x -2.5) + (0 x 0.6) + 1.6 = 1.1
f(x) = max(0, x) = y-axis coordinate
Setosa
2
0
0.2
Petal Width
0.8
0.2
Setosa
Petal Width
0.2
x -2.5
sum
+ 1.6
x -0.1
sum
+ 0
=
x -1.5
Sepal Width
0
x 0.6
x 0.4
sum
+ 0.7
x 1.5
…and 1.1 for the y-axis coordinate.



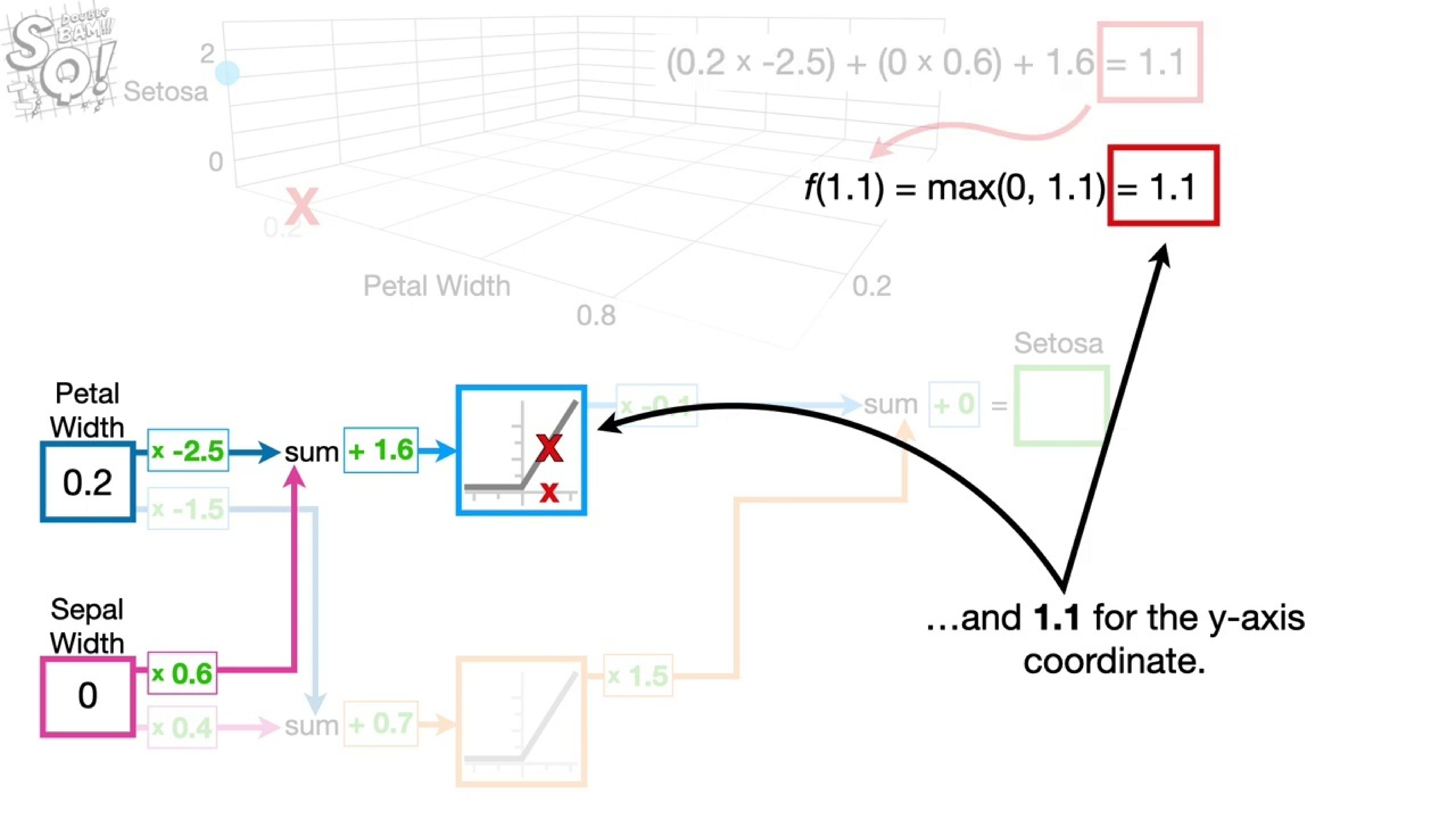
(0.2 x -2.5) + (0 x 0.6) + 1.6 = 1.1
f(1.1) = max(0, 1.1) = y-axis coordinate
2
Setosa
0
0.2
Petal Width
0.8
0.2
Setosa
Petal Width
0.2
x -2.5
sum
+ 1.6
x -0.1
sum
+ 0
=
x -1.5
Sepal Width
0
x 0.6
x 0.4
sum
+ 0.7
x 1.5
…and 1.1 for the y-axis coordinate.

$$(0.2 \times -2.5) + (0 \times 0.6) + 1.6 \boxed{= 1.1}$$

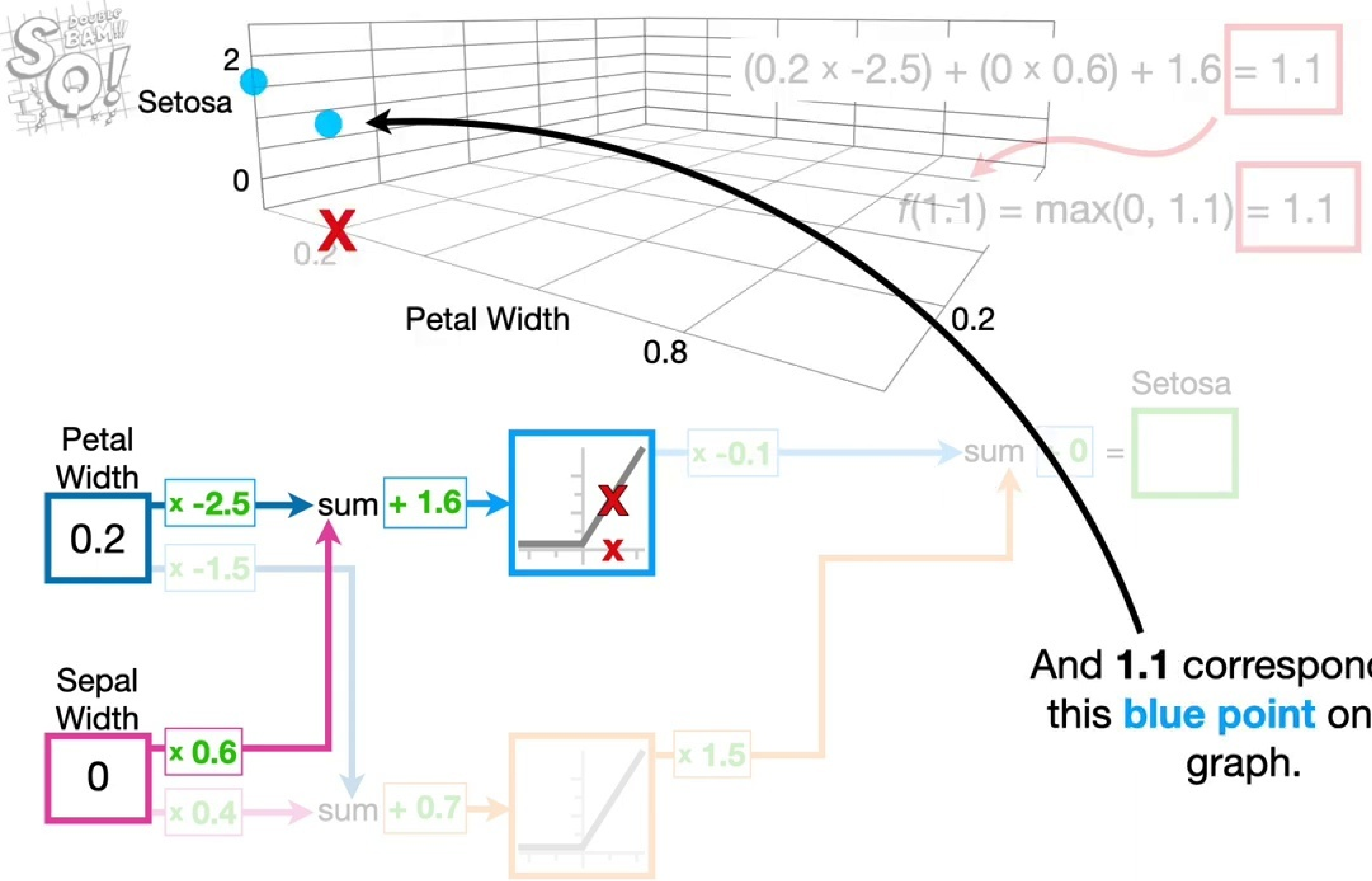
$$f(1.1) = \max(0, 1.1) = 1.1$$



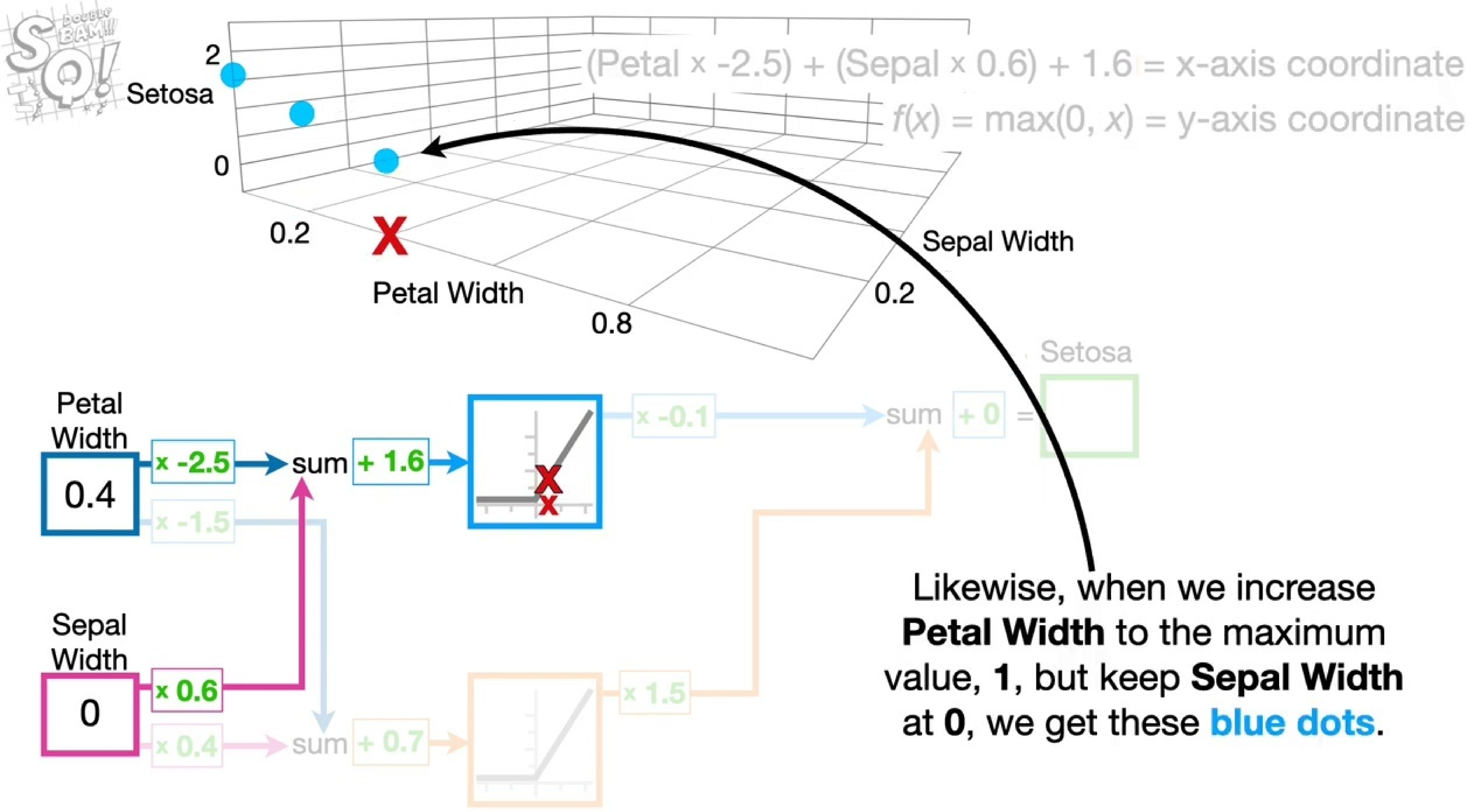
…and **1.1** for the y-axis coordinate.



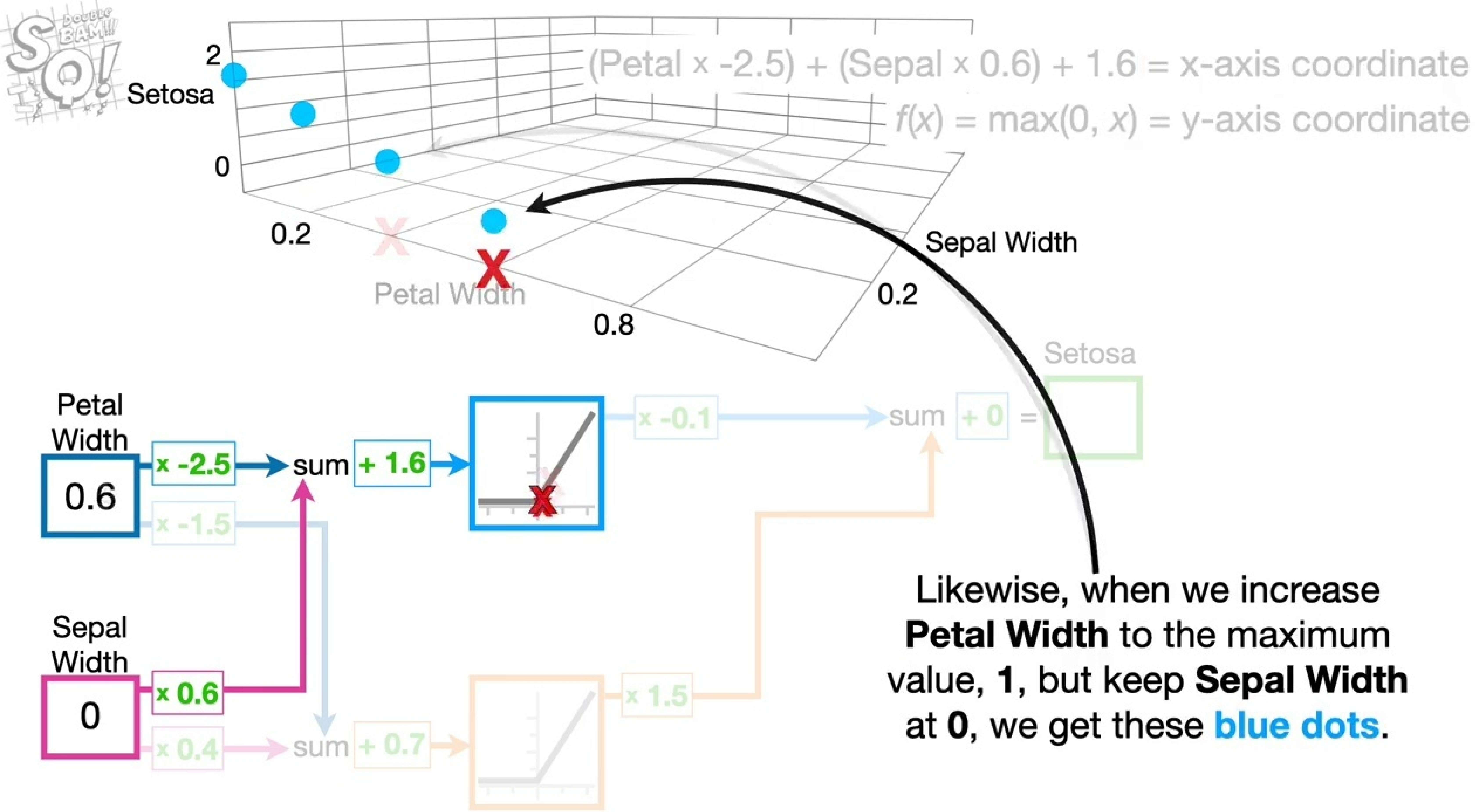
2
0
Setosa
(0.2 x -2.5) + (0 x 0.6) + 1.6 = 1.1
f(1.1) = max(0, 1.1) = 1.1
0.2
Petal Width
0.8
0.2
Setosa
Petal Width
0.2
x -2.5
x -1.5
sum
+ 1.6
x -0.1
sum
+ 0
=
Sepal Width
0
x 0.6
x 0.4
sum
+ 0.7
x 1.5
…and 1.1 for the y-axis coordinate.



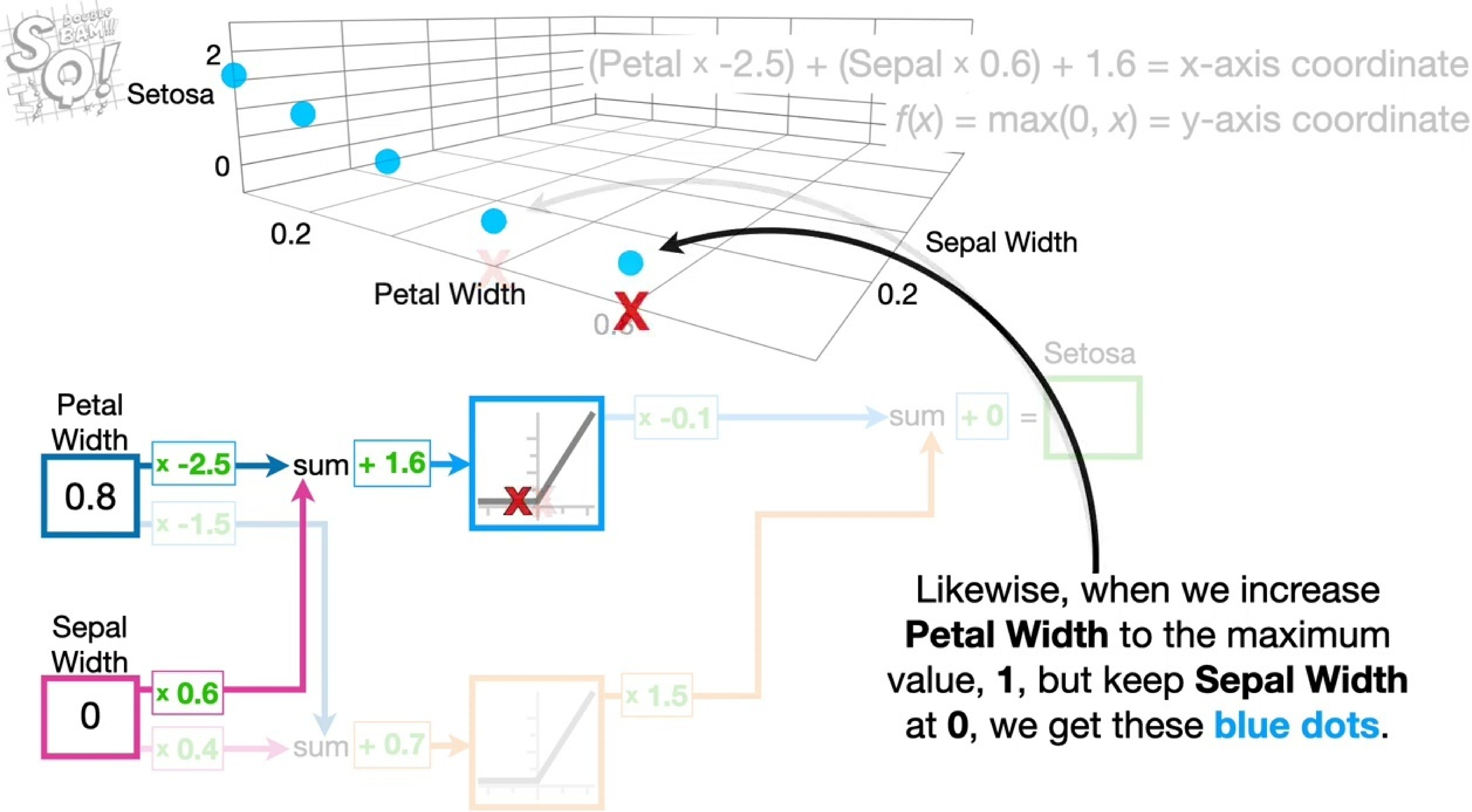
And **1.1** corresponds to this **blue point** on the graph.



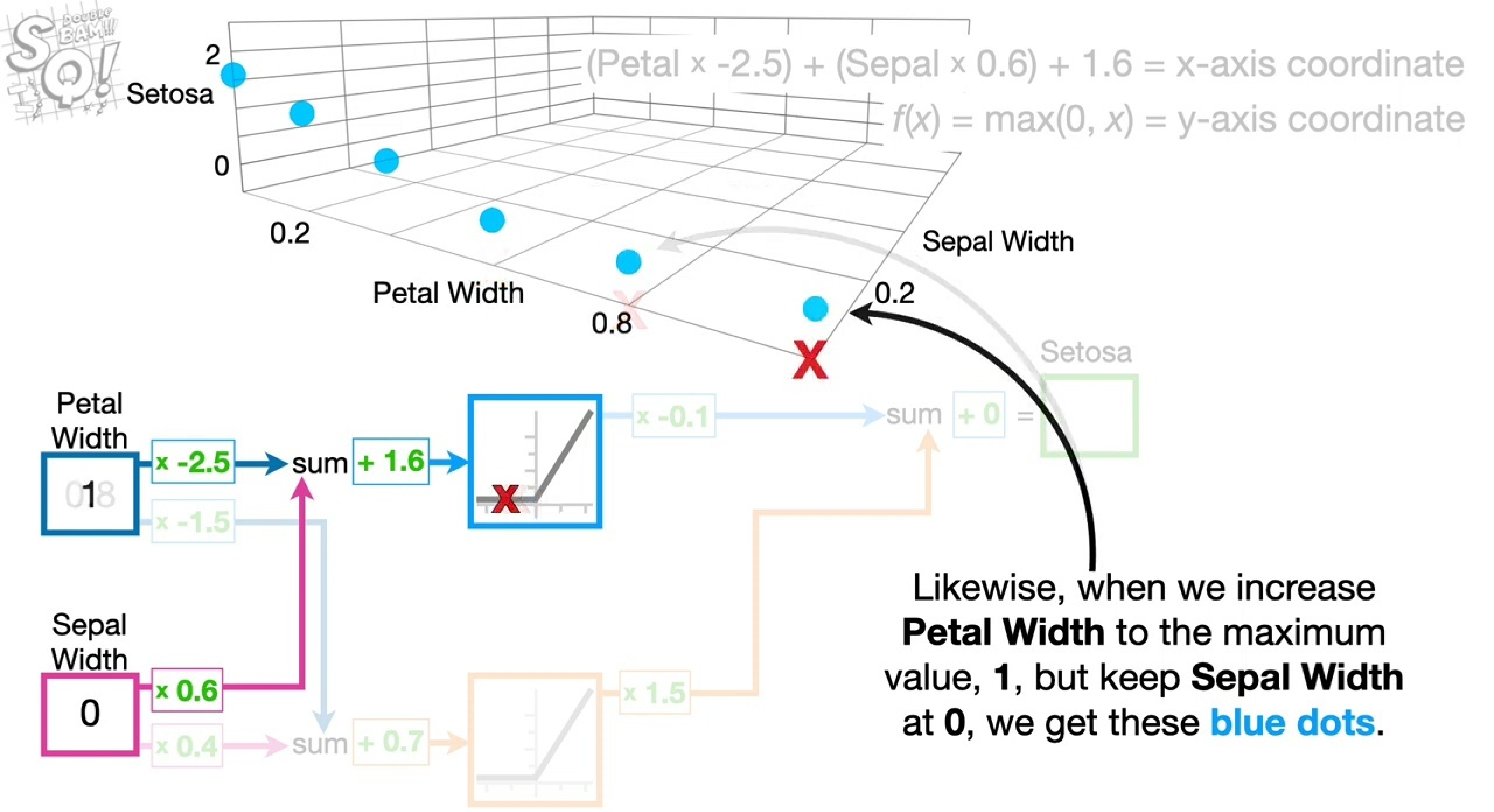
Likewise, when we increase **Petal Width** to the maximum value, **1**, but keep **Sepal Width** at **0**, we get these **blue dots**.



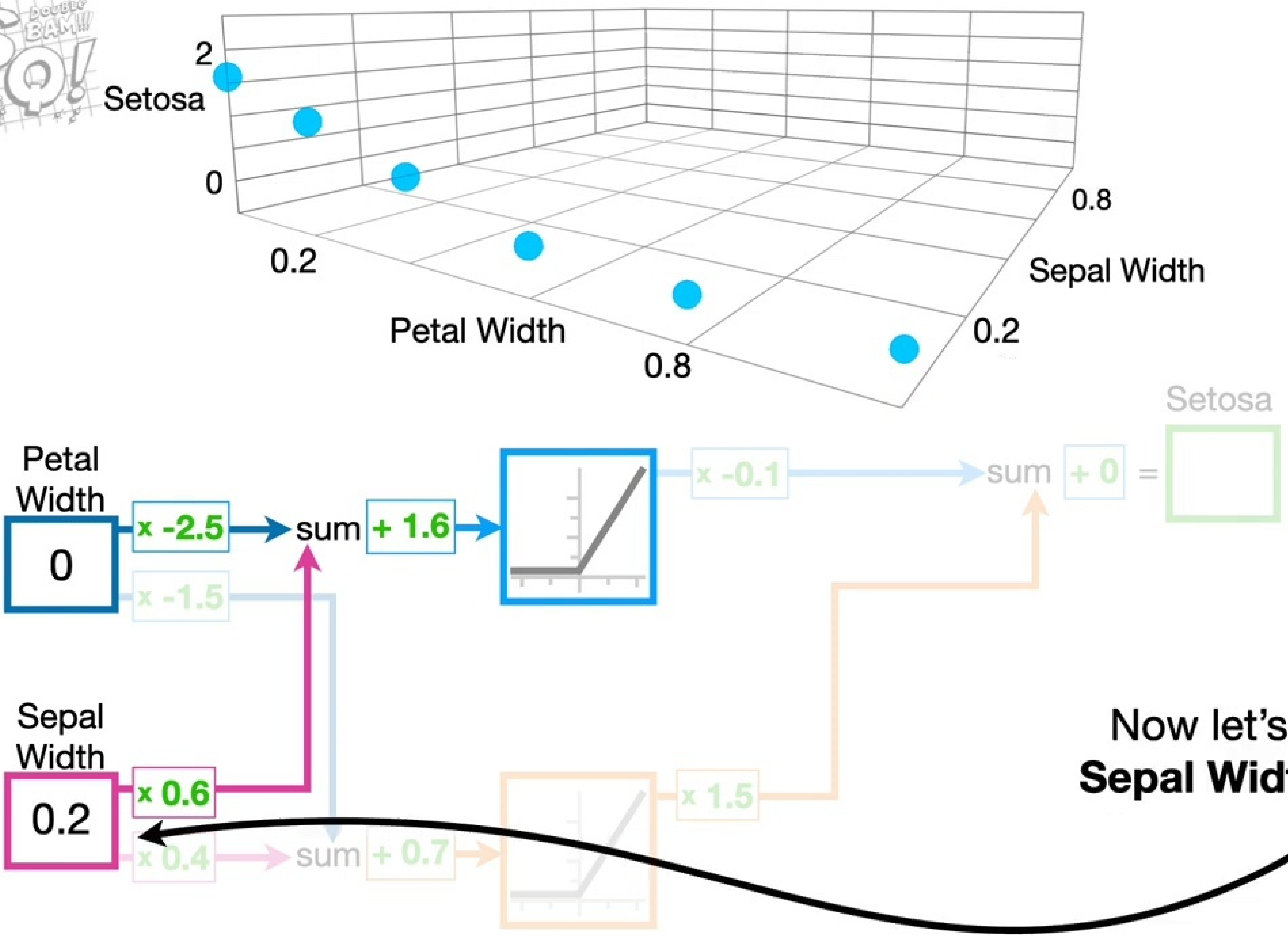
Likewise, when we increase **Petal Width** to the maximum value, **1**, but keep **Sepal Width** at **0**, we get these blue dots.



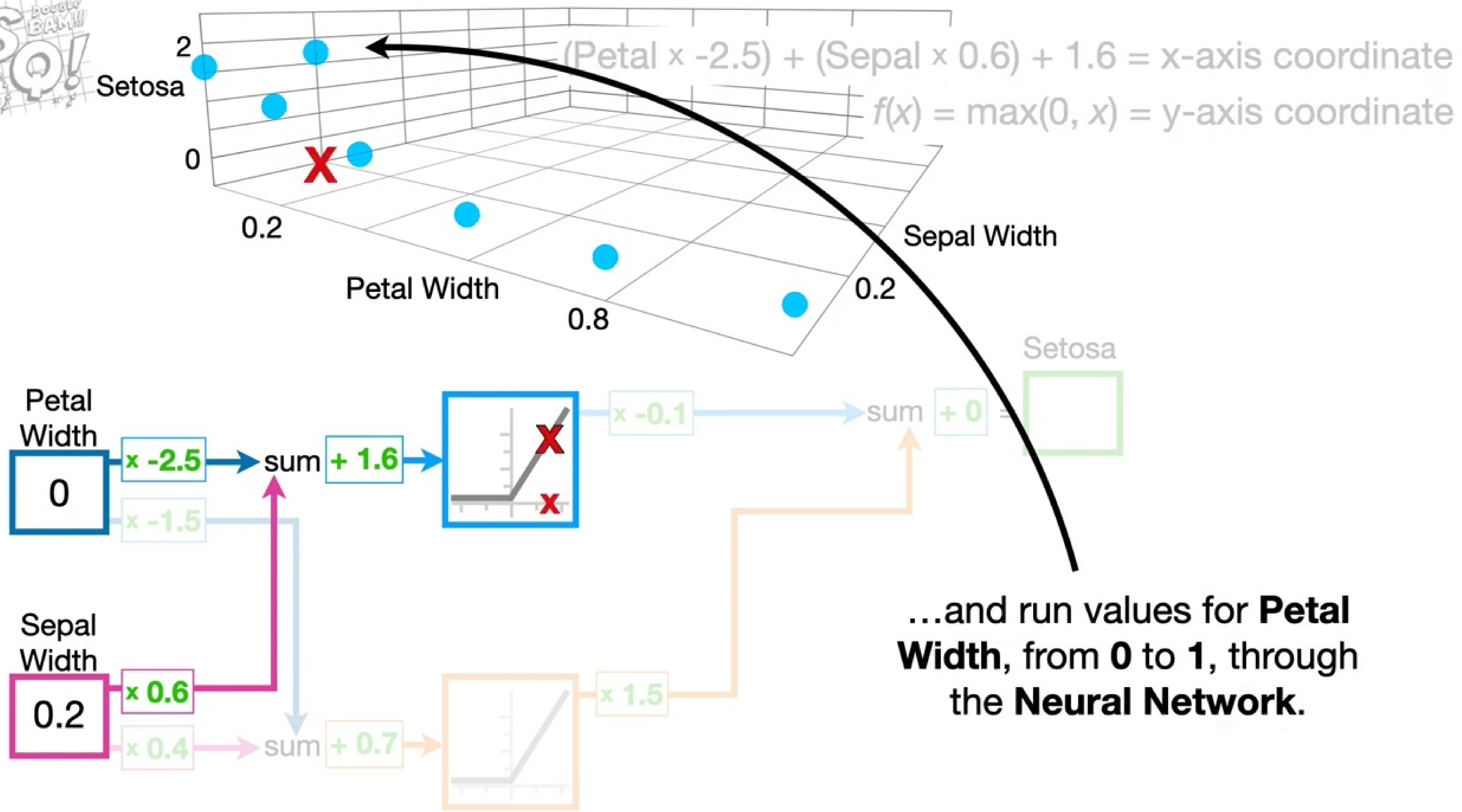
Likewise, when we increase **Petal Width** to the maximum value, **1**, but keep **Sepal Width** at **0**, we get these **blue dots**.



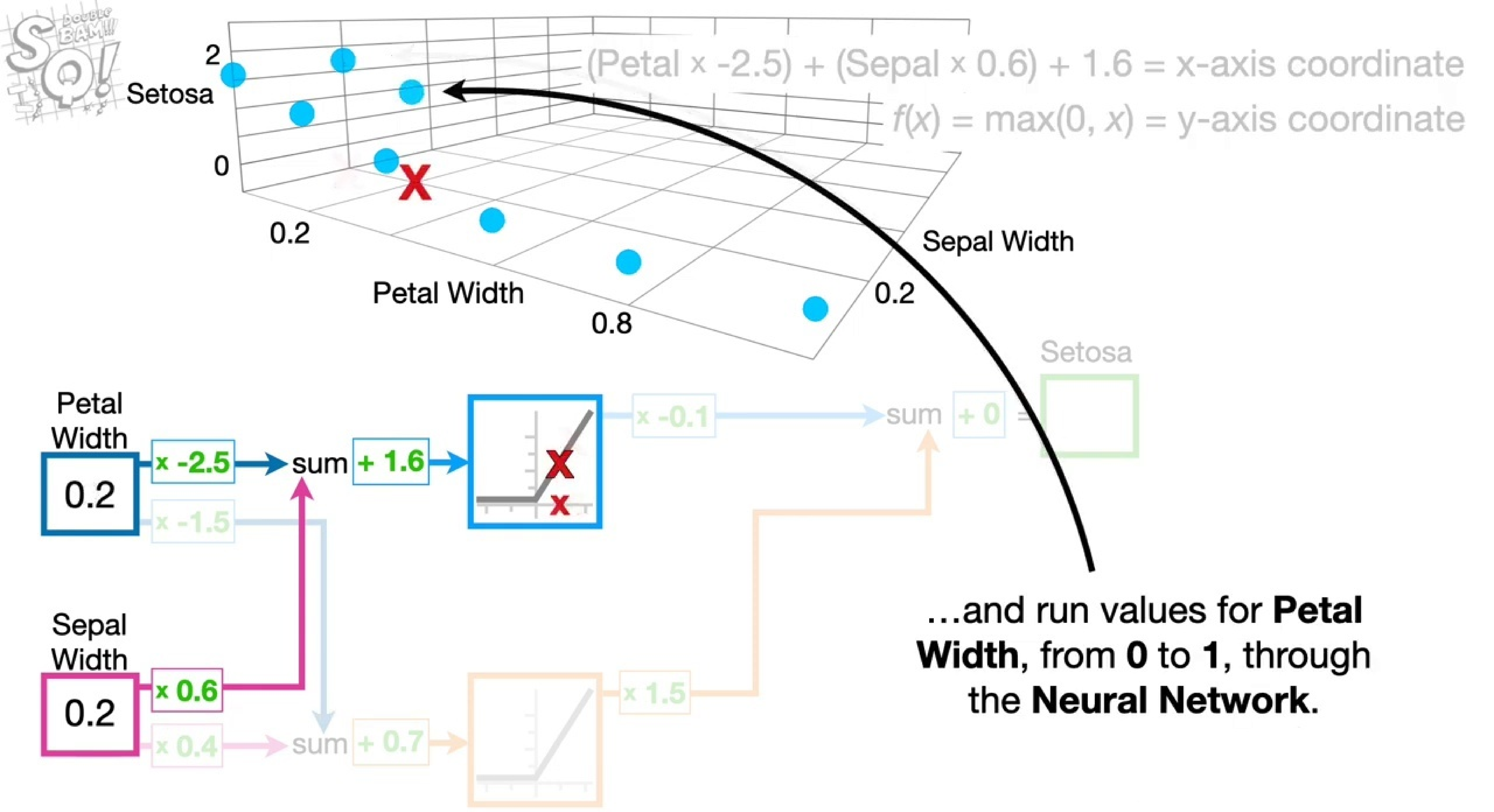
Likewise, when we increase **Petal Width** to the maximum value, **1**, but keep **Sepal Width** at **0**, we get these **blue dots**.



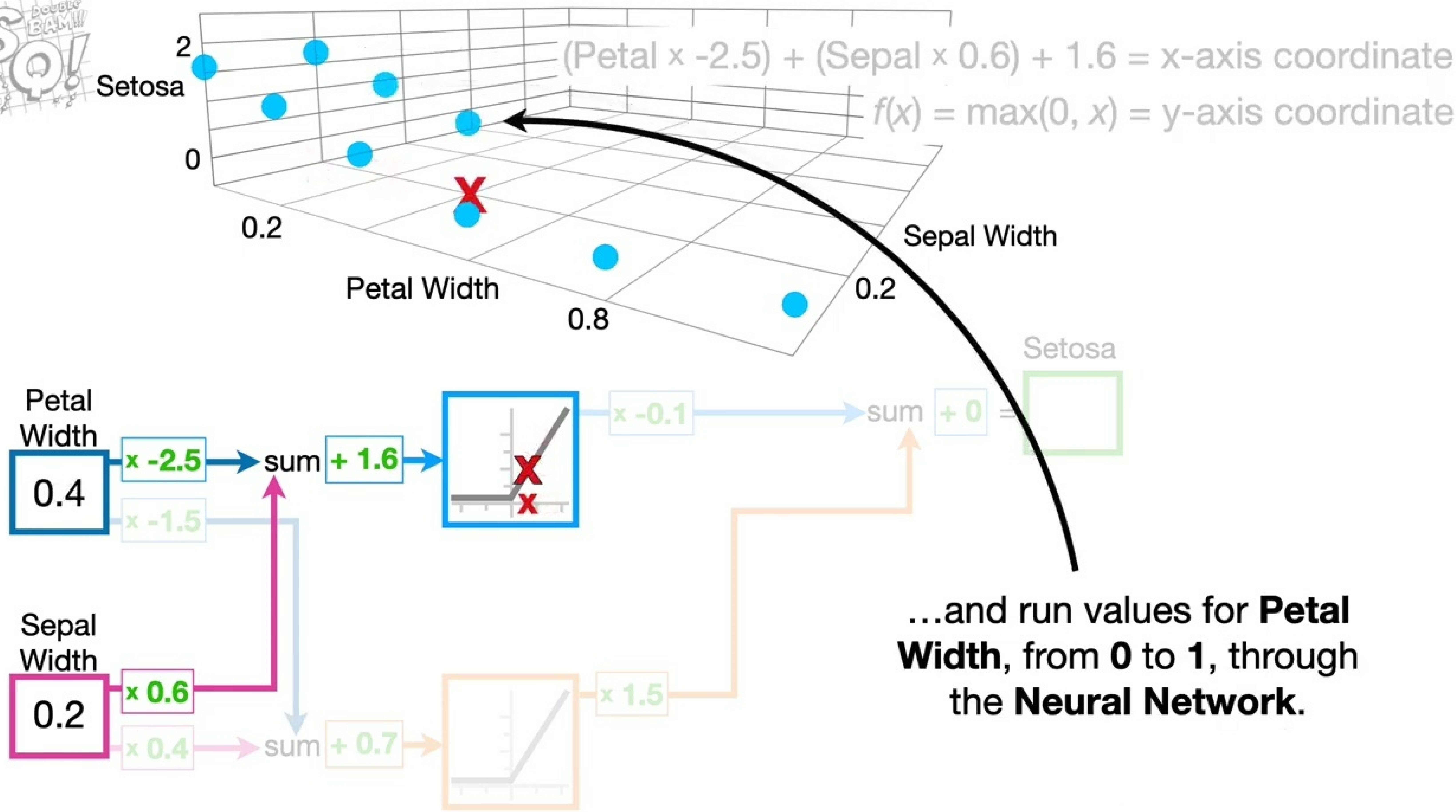
Now let's increase **Sepal Width** to **0.2**…



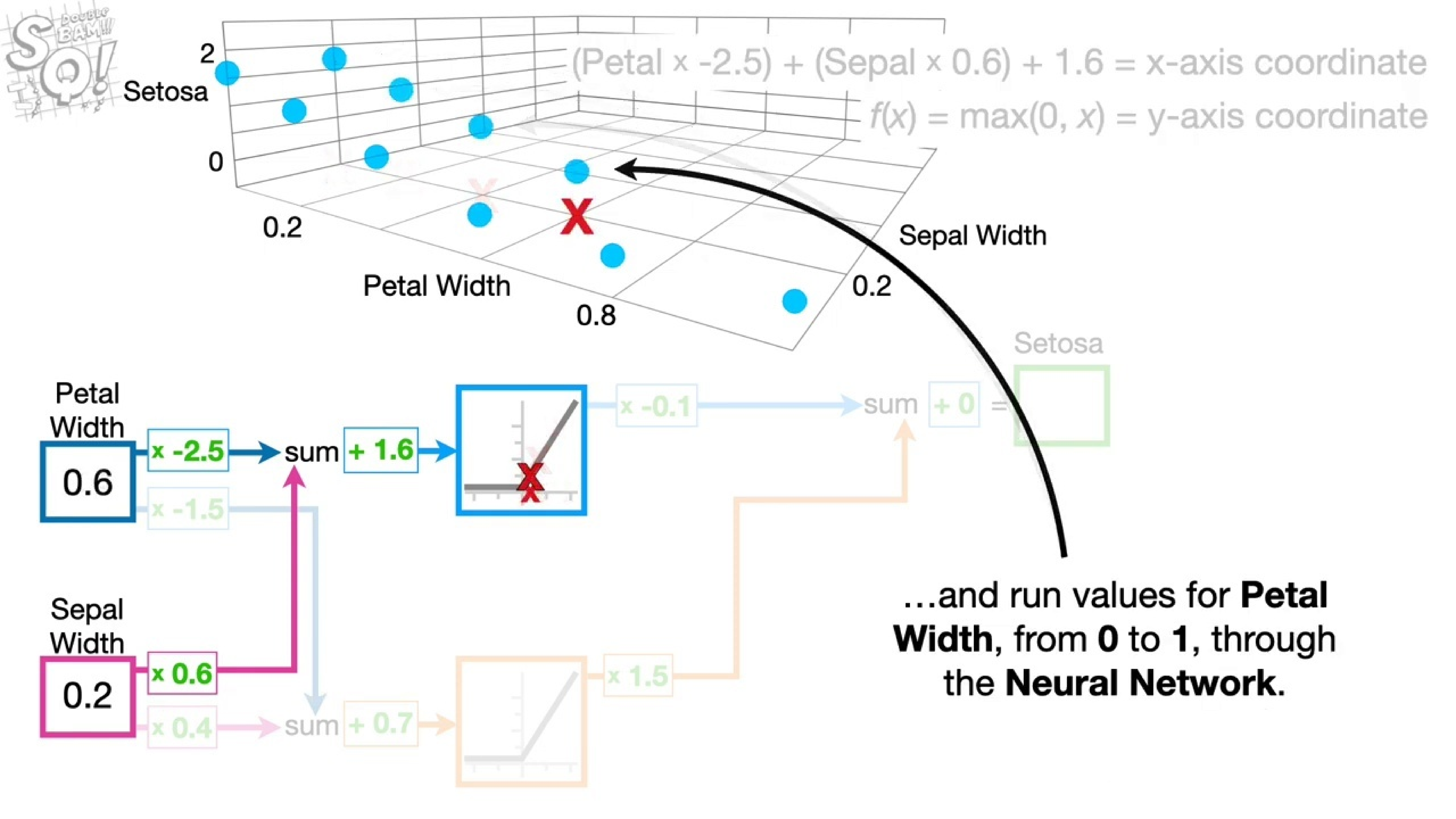
…and run values for **Petal Width**, from **0** to **1**, through the **Neural Network**.

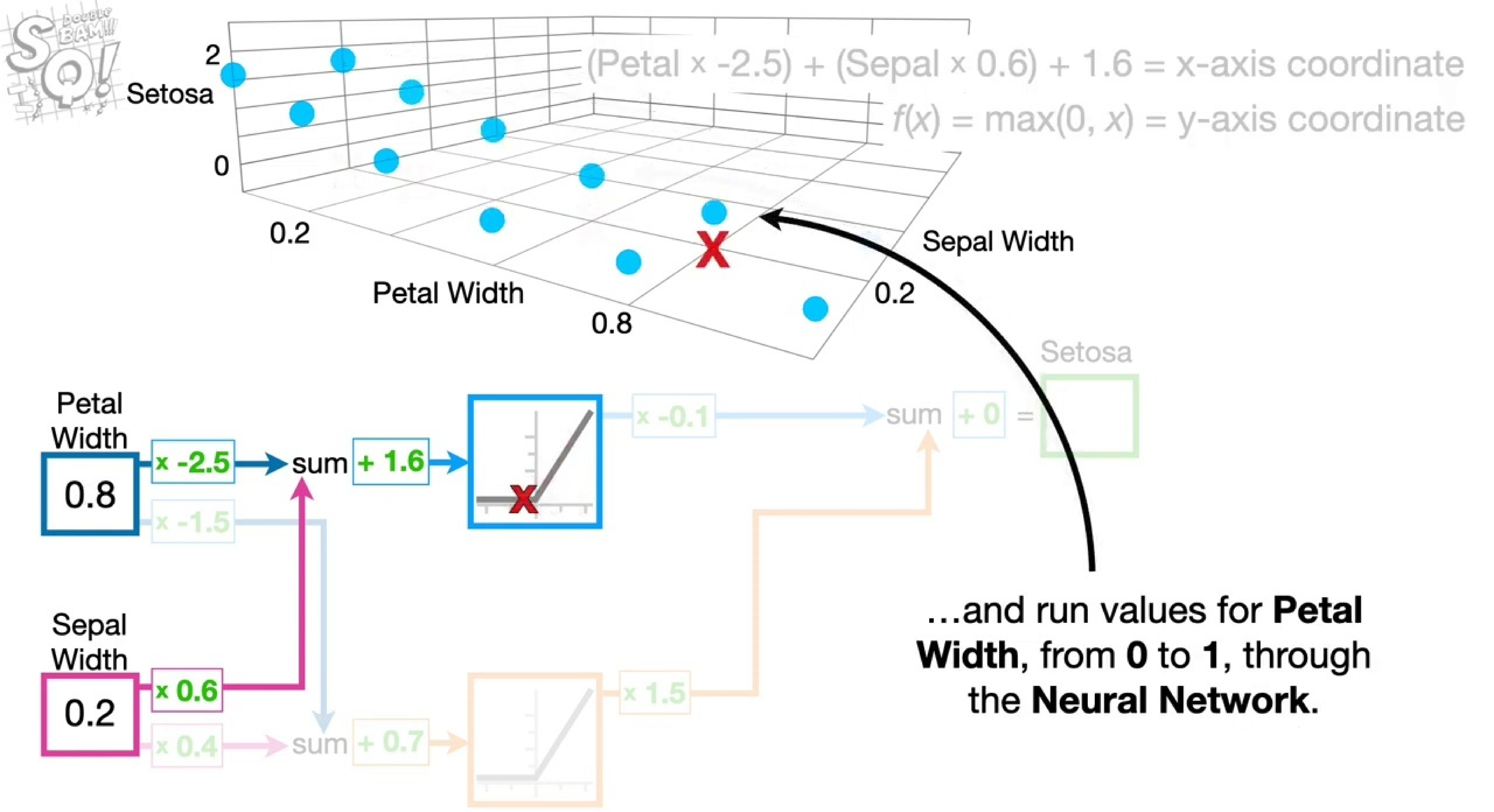


…and run values for **Petal Width**, from **0** to **1**, through the **Neural Network**.

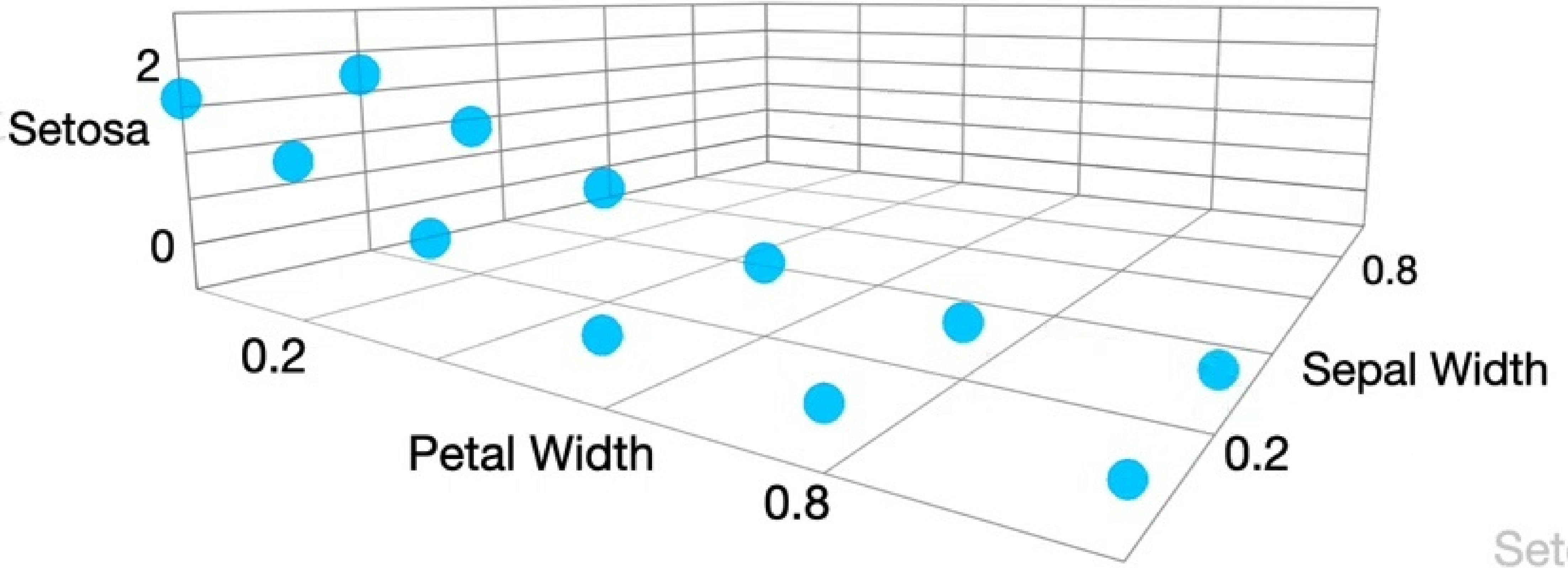


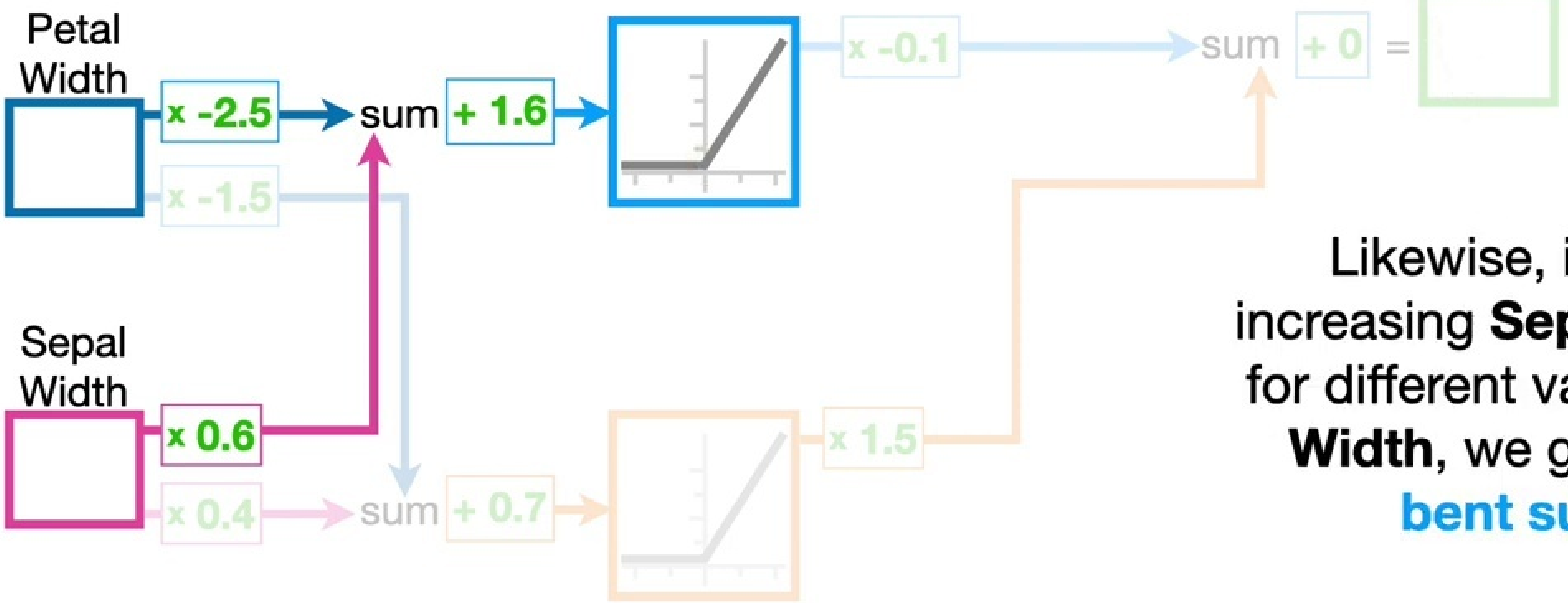
…and run values for **Petal Width**, from **0** to **1**, through the **Neural Network**.


(Petal x -2.5) + (Sepal x 0.6) + 1.6 = x-axis coordinate
f(x) = max(0, x) = y-axis coordinate
Setosa
2
0
0.2
Petal Width
0.8
Sepal Width
0.2
Setosa
Petal Width
0.6
x -2.5
sum
+ 1.6
x -1.5
x -0.1
sum
+ 0
Sepal Width
0.2
x 0.6
x 0.4
sum
+ 0.7
x 1.5
…and run values for **Petal Width**, from **0** to **1**, through the **Neural Network**.

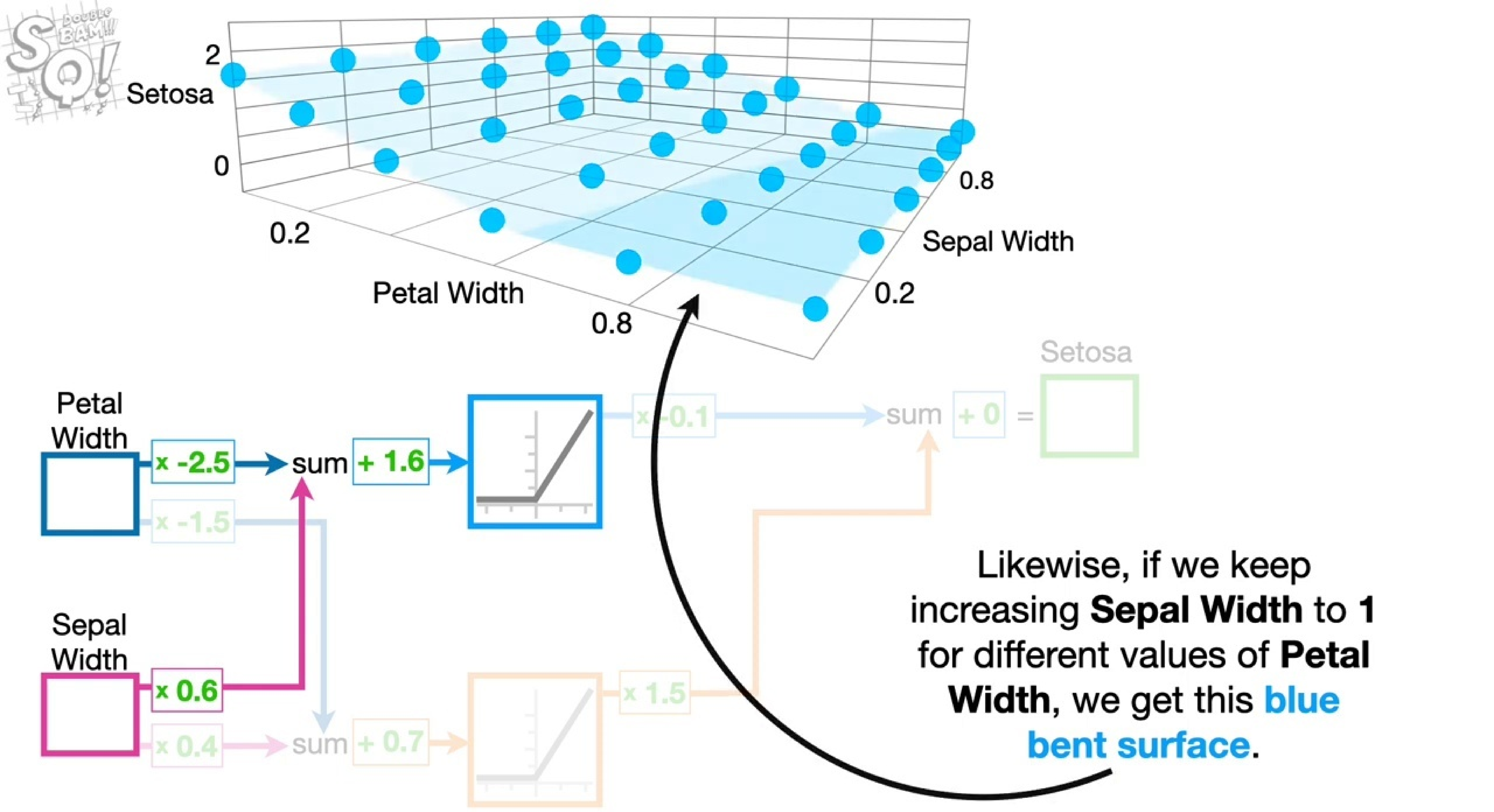


…and run values for **Petal Width**, from **0** to **1**, through the **Neural Network**.

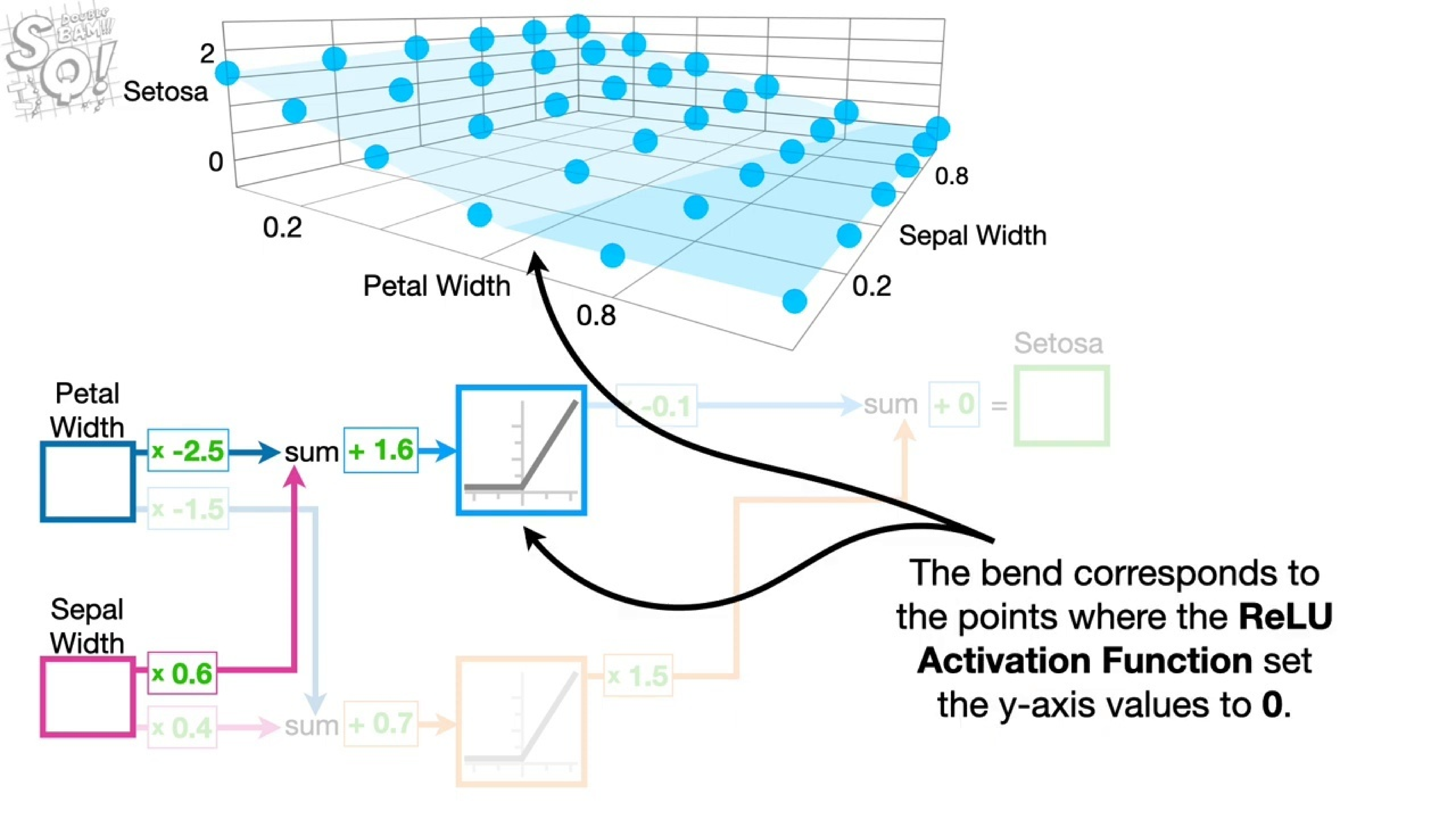




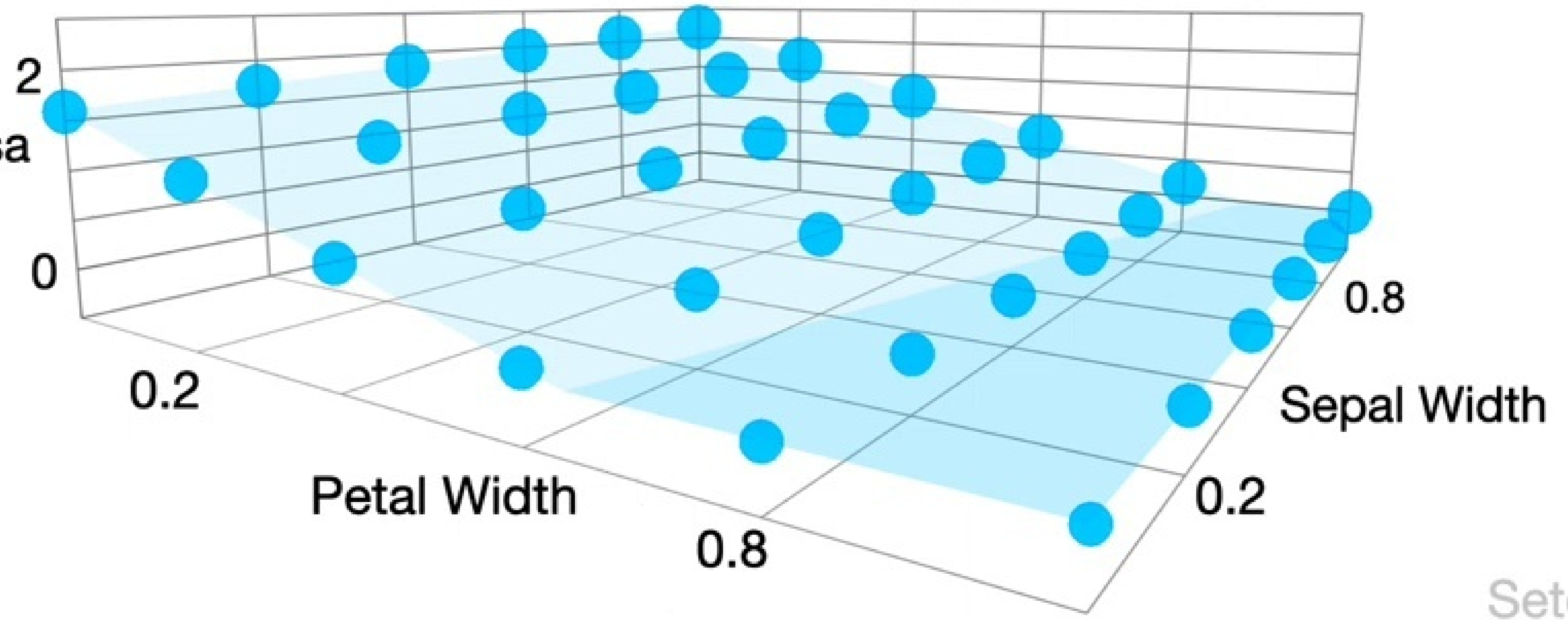
Likewise, if we keep increasing **Sepal Width** to **1** for different values of **Petal Width**, we get this **blue bent surface**.

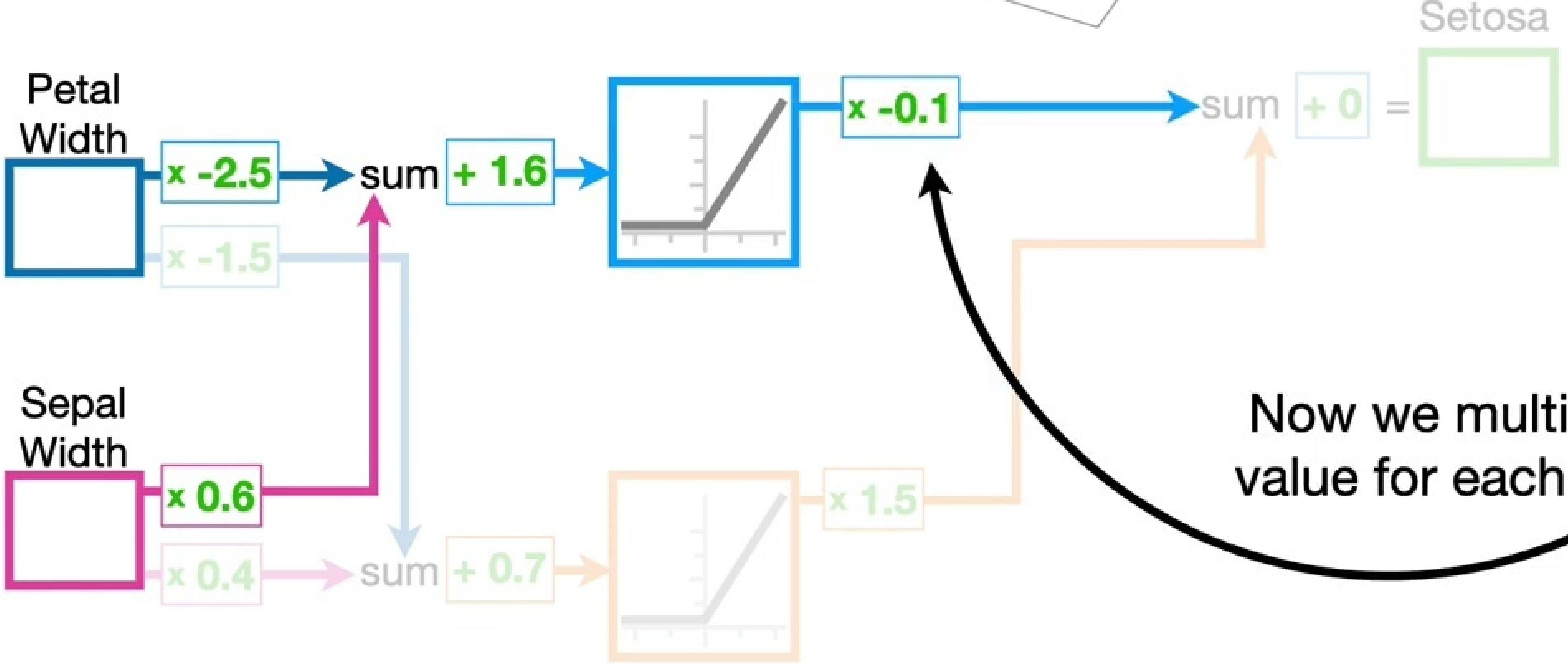


Setosa
2
0
0.2
Petal Width
0.8
0.8
Sepal Width
0.2
Setosa
Petal Width
x -2.5
sum
+ 1.6
x -1.5
x -0.1
sum
+ 0
=
Sepal Width
x 0.6
x 0.4
sum
+ 0.7
x 1.5
Likewise, if we keep increasing **Sepal Width** to **1** for different values of **Petal Width**, we get this **blue bent surface**.

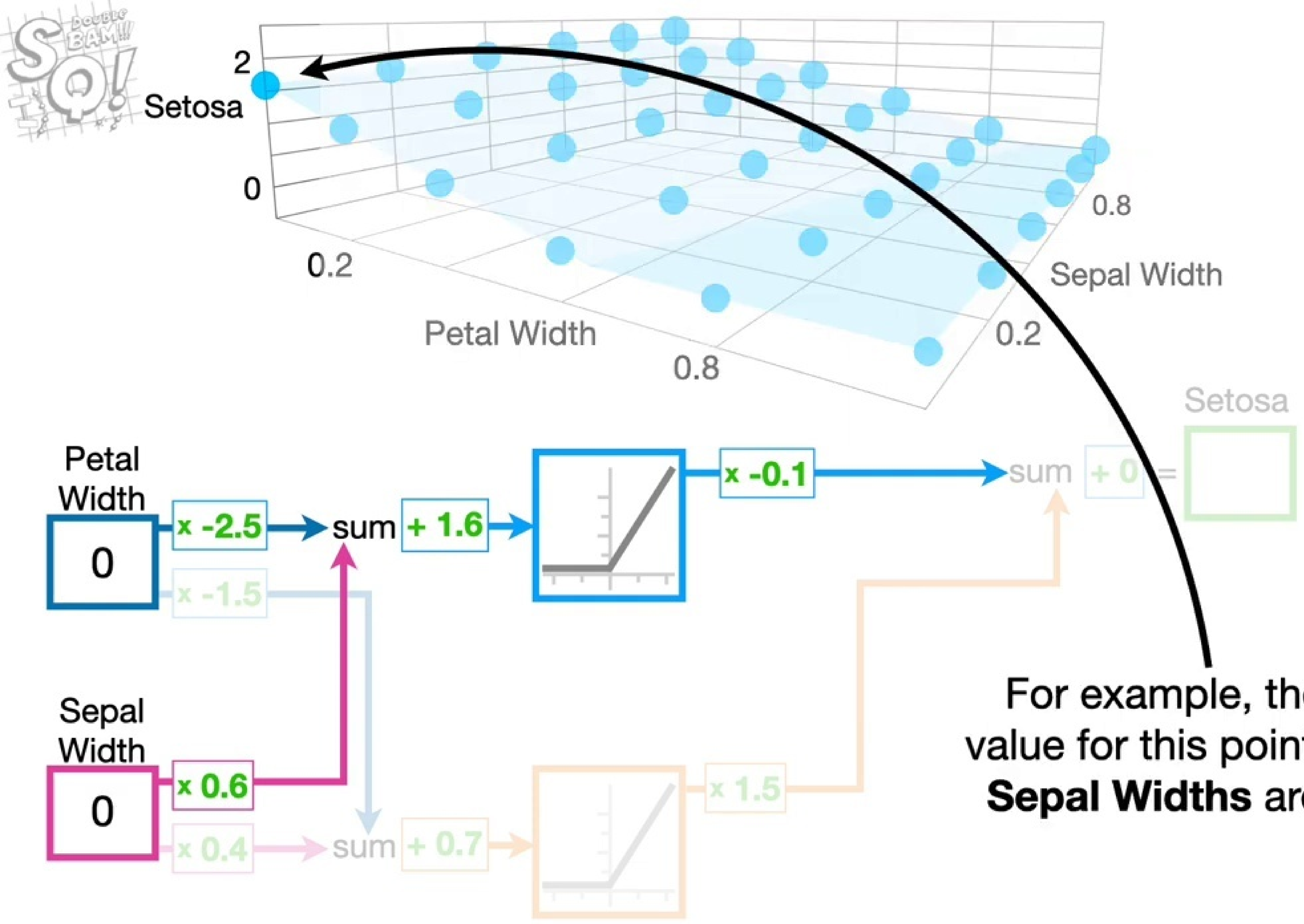


2
0
Setosa
0.2
Petal Width
0.8
0.8
Sepal Width
0.2
Petal Width
Sepal Width
x -2.5
x -1.5
x 0.6
x 0.4
sum
+ 1.6
sum
+ 0.7
x -0.1
x 1.5
sum
+ 0
=
Setosa
The bend corresponds to the points where the **ReLU Activation Function** set the y-axis values to **0**.

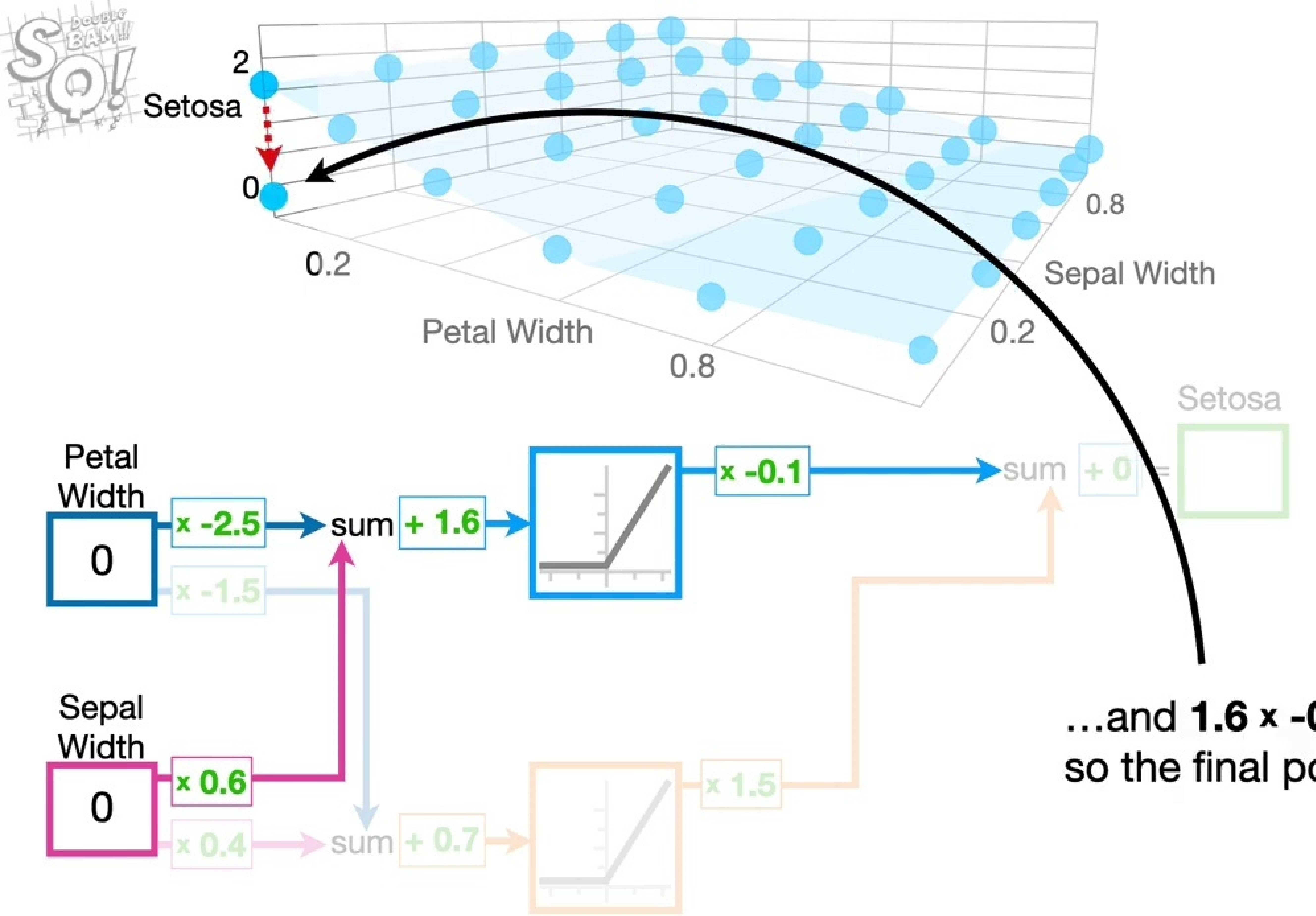




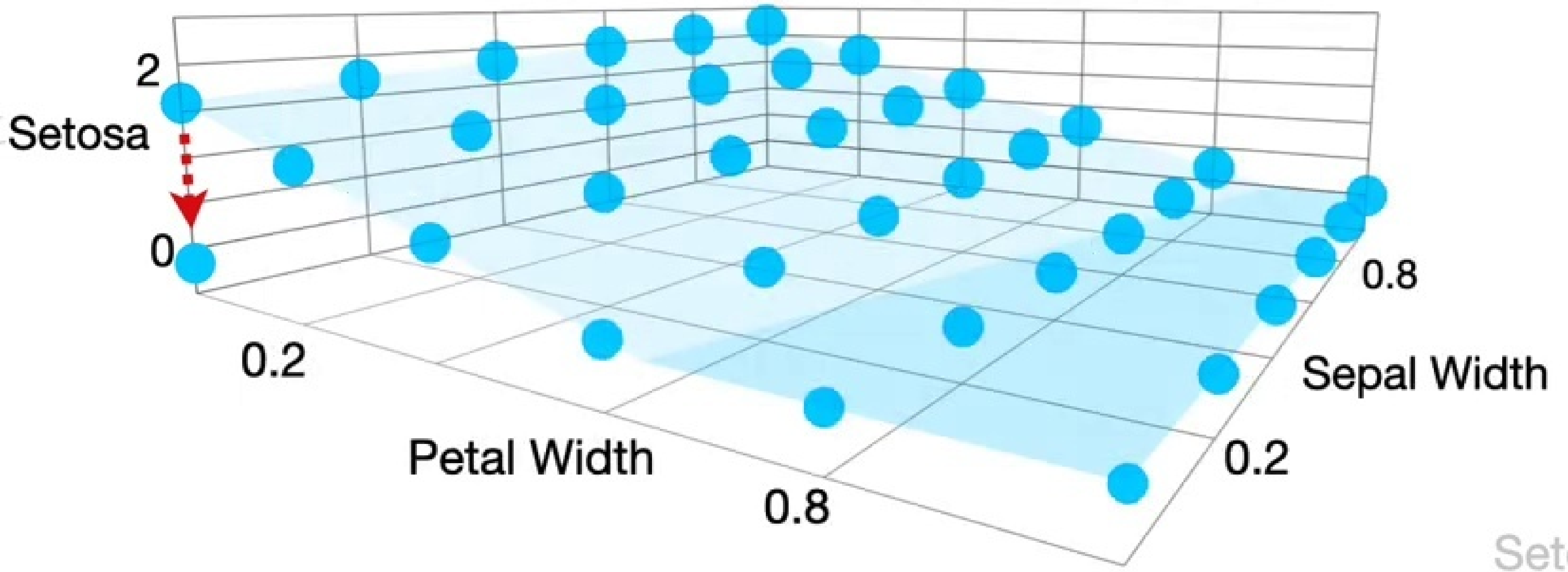
Now we multiply the y-axis value for each point by **-0.1**.

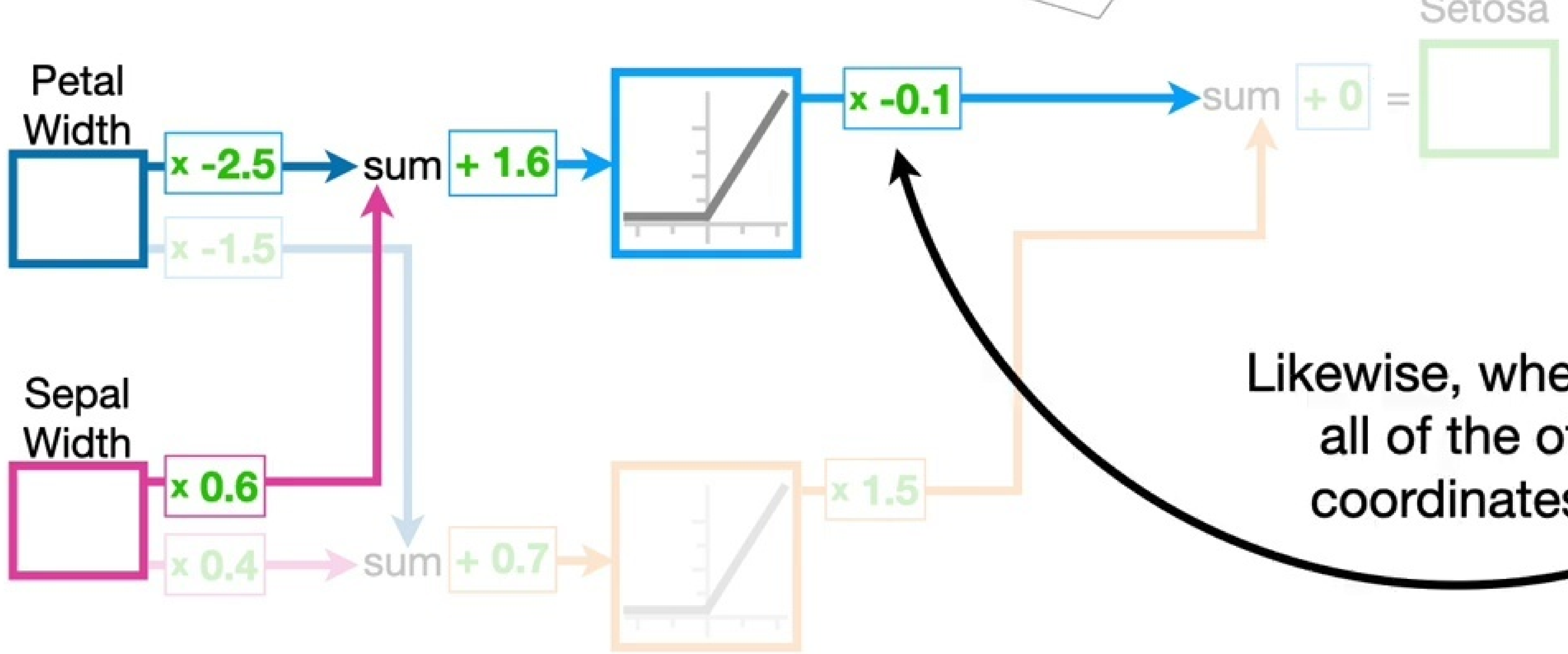


For example, the original y-axis value for this point, when **Petal** and **Sepal Widths** are both **0**, is **1.6**...

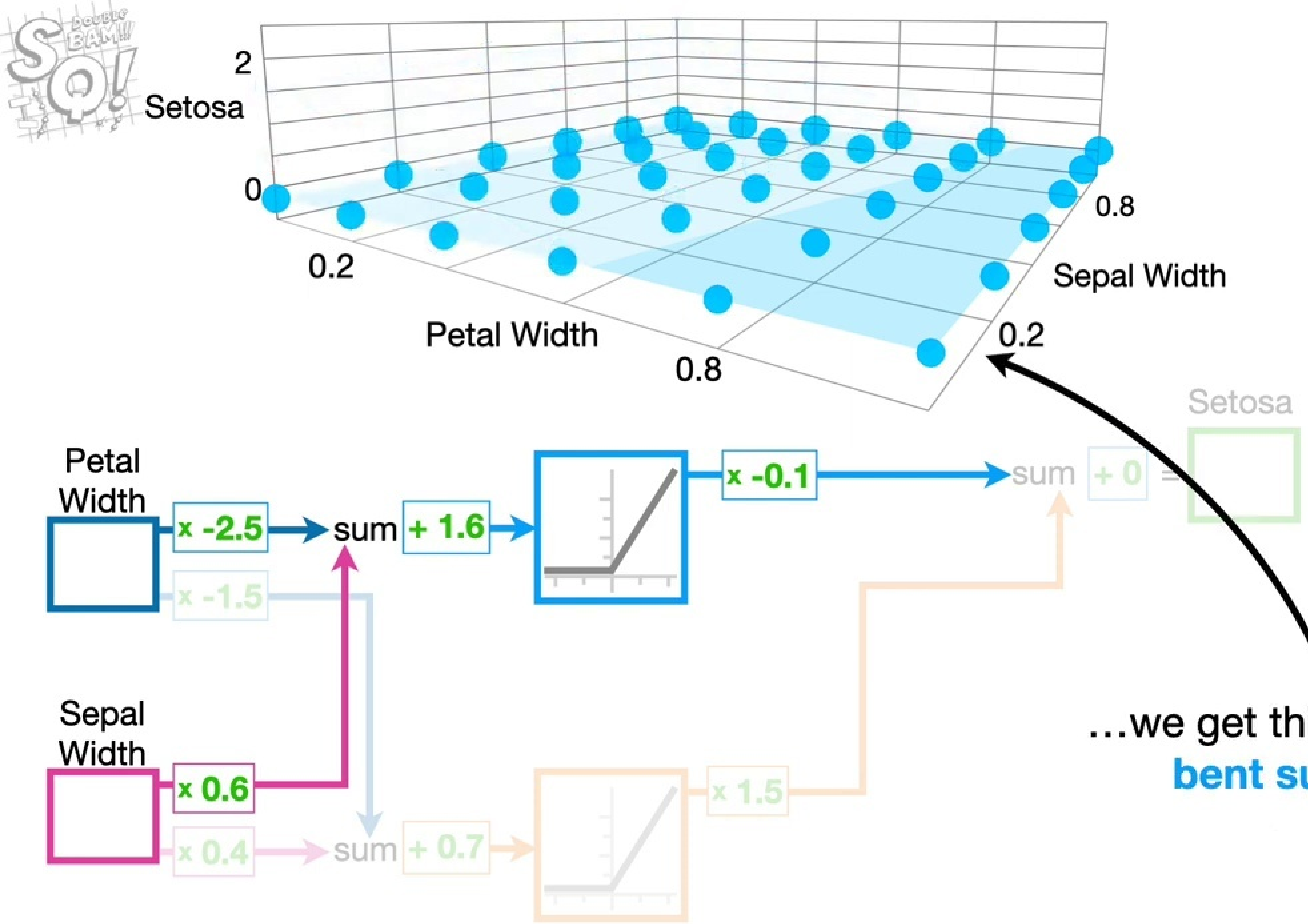


…and **1.6 x -0.1 = -0.16**, so the final point is here.

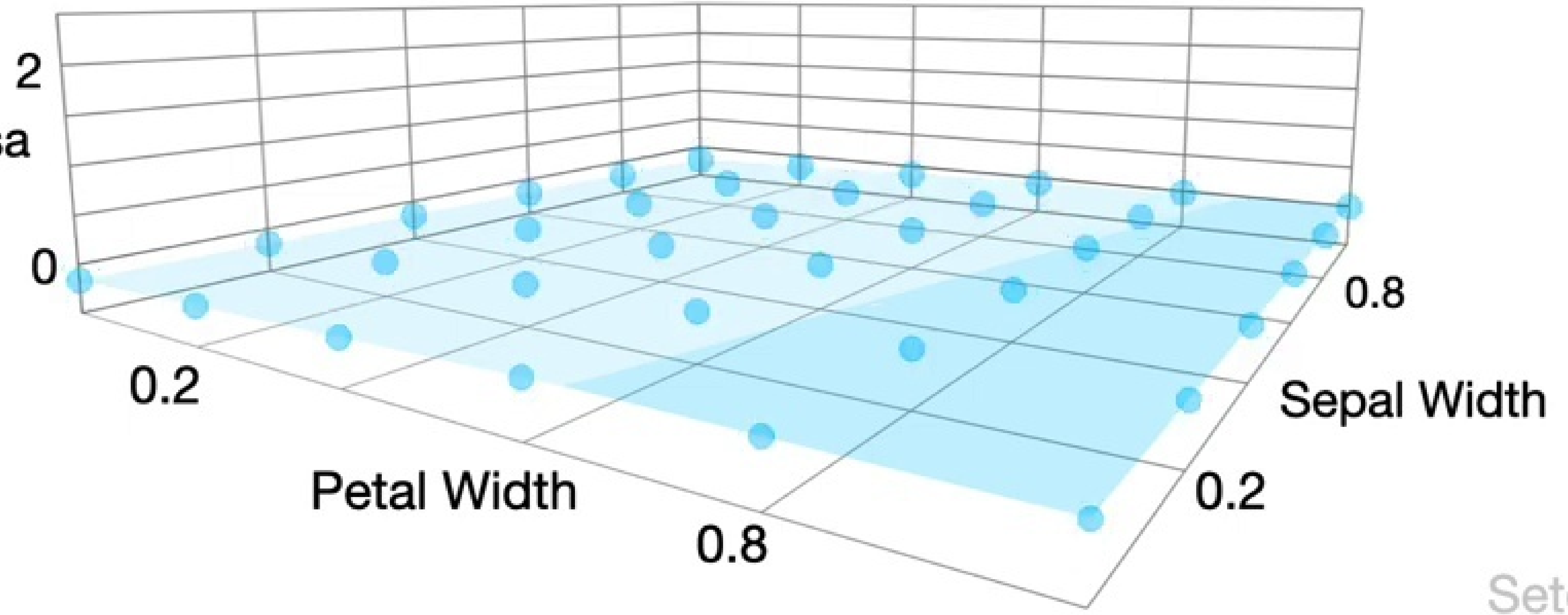


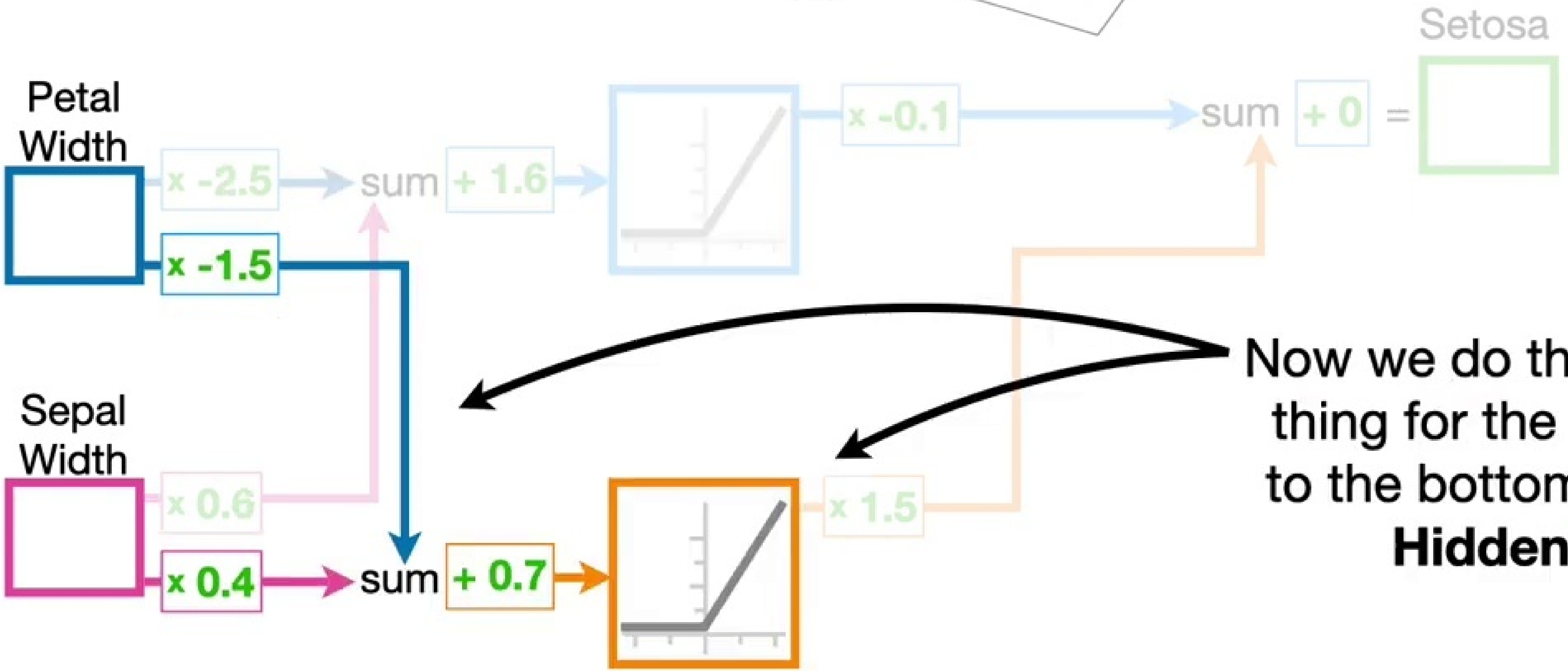


Likewise, when we multiply all of the other y-axis coordinates by **-0.1**…

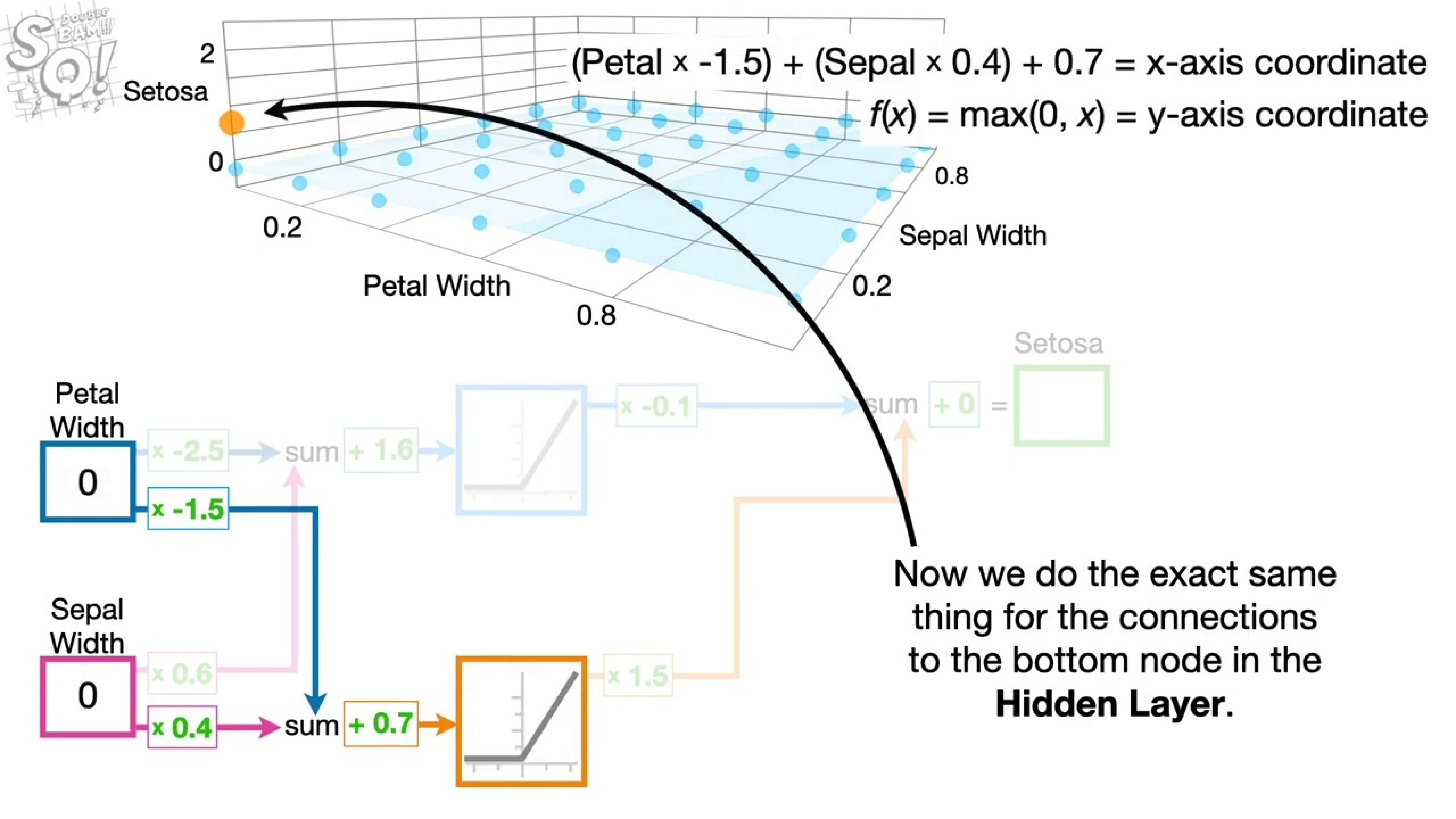


…we get this final **blue bent surface**.

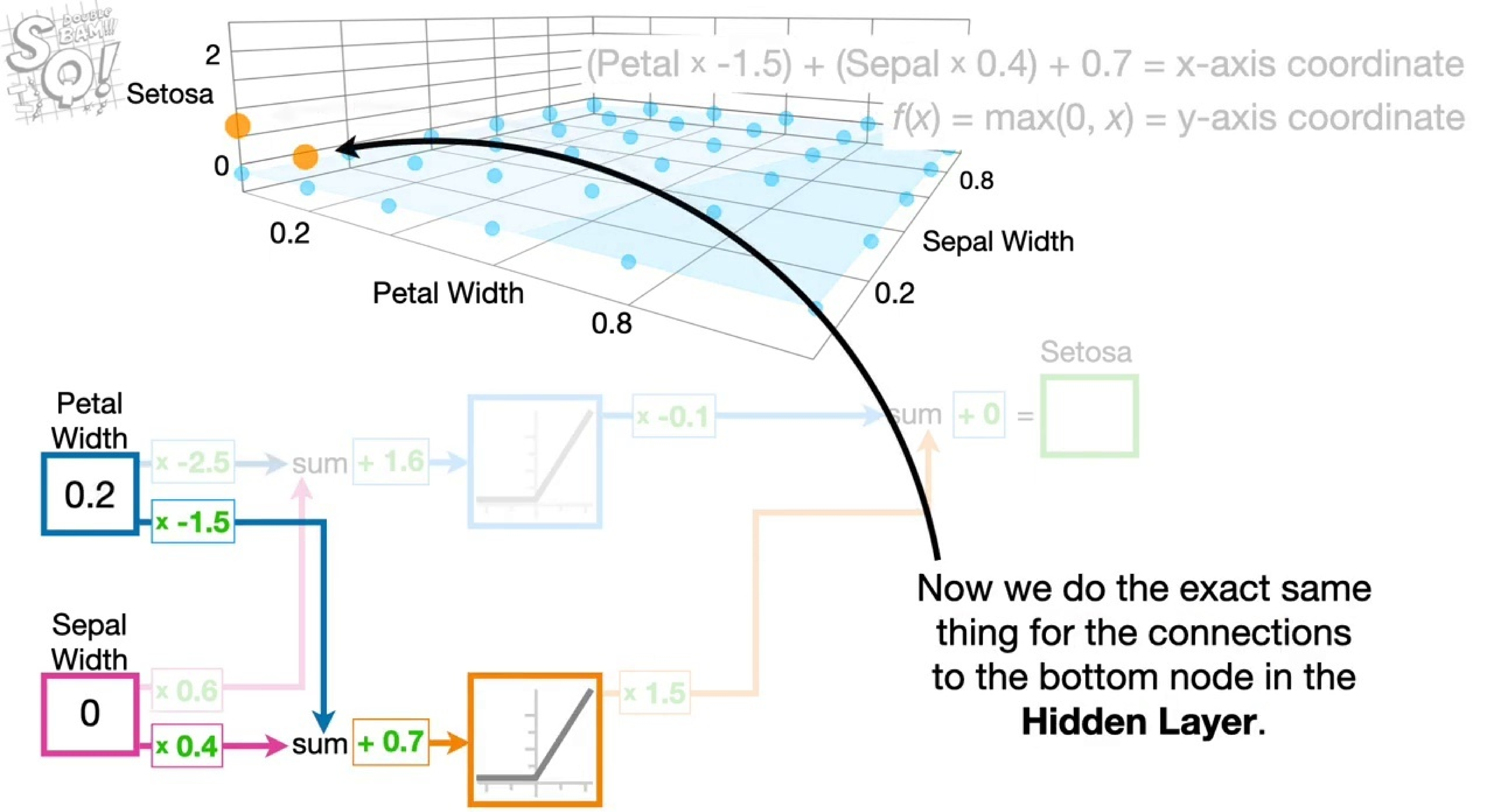




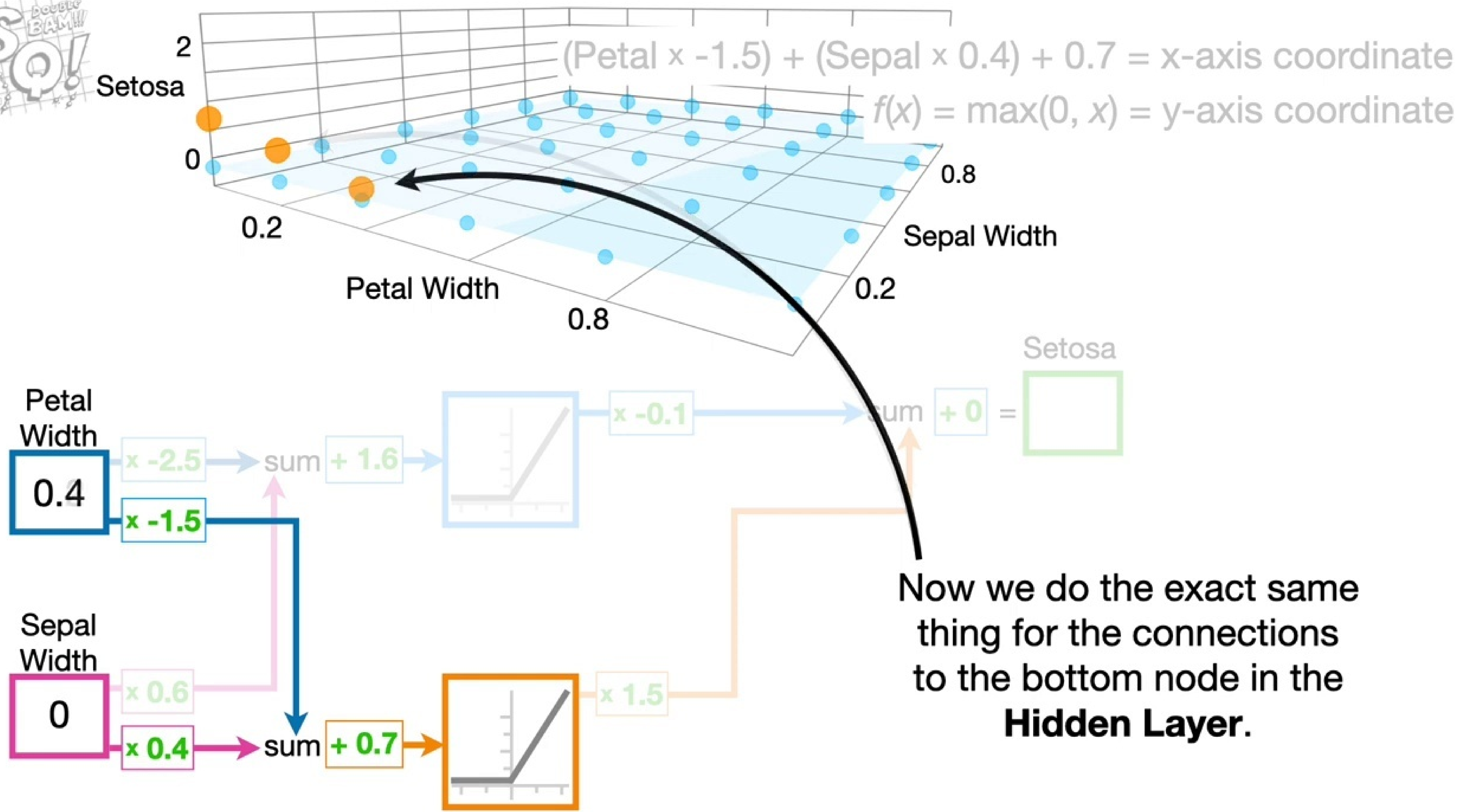
Now we do the exact same thing for the connections to the bottom node in the **Hidden Layer**.



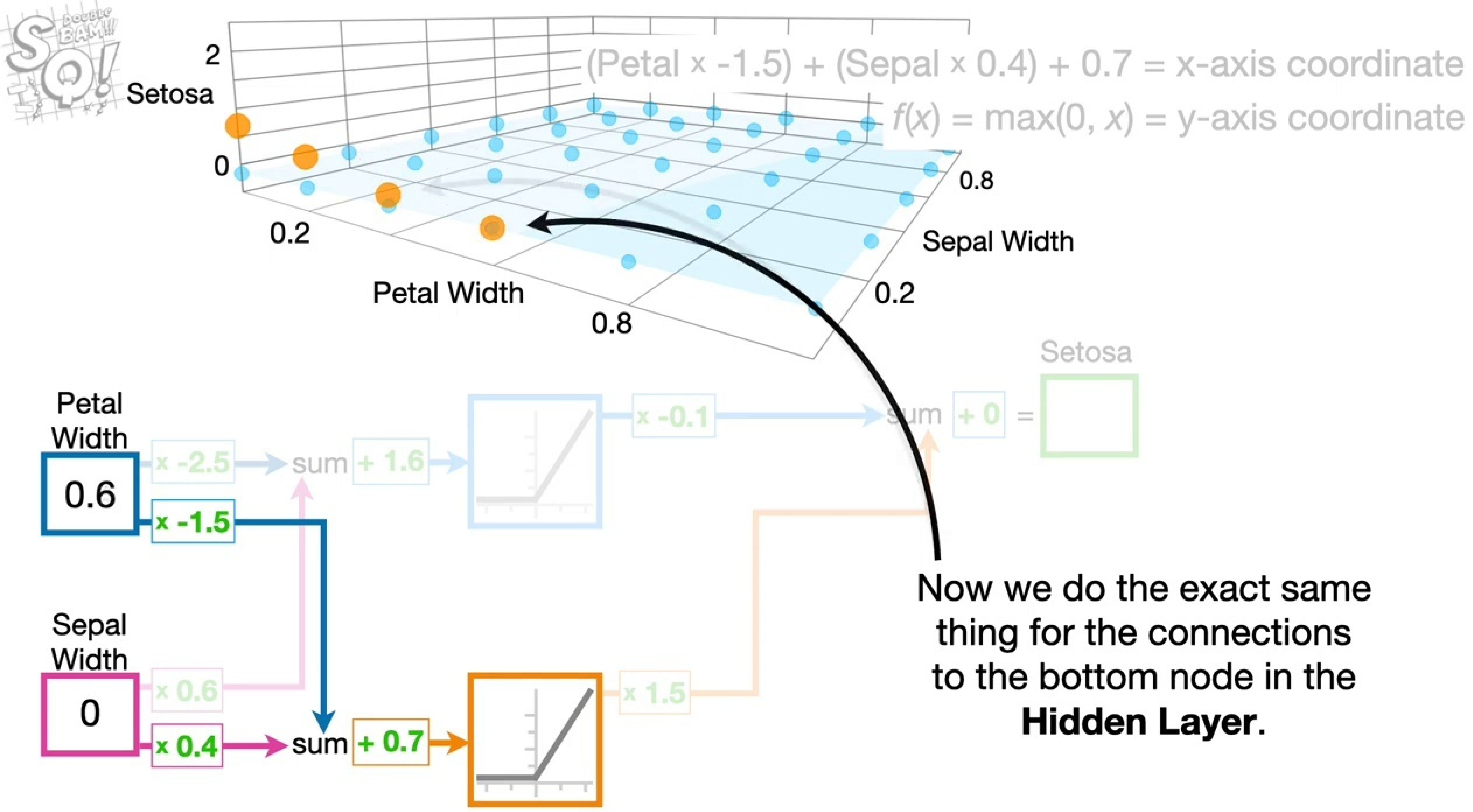
(Petal x -1.5) + (Sepal x 0.4) + 0.7 = x-axis coordinate
f(x) = max(0, x) = y-axis coordinate
2
Setosa
0
0.2
Petal Width
0.8
0.8
Sepal Width
0.2
Setosa
Petal Width
0
x -2.5
sum
+ 1.6
x -0.1
sum
+ 0
=
x -1.5
Sepal Width
0
x 0.6
x 0.4
sum
+ 0.7
x 1.5
Now we do the exact same thing for the connections to the bottom node in the **Hidden Layer**.



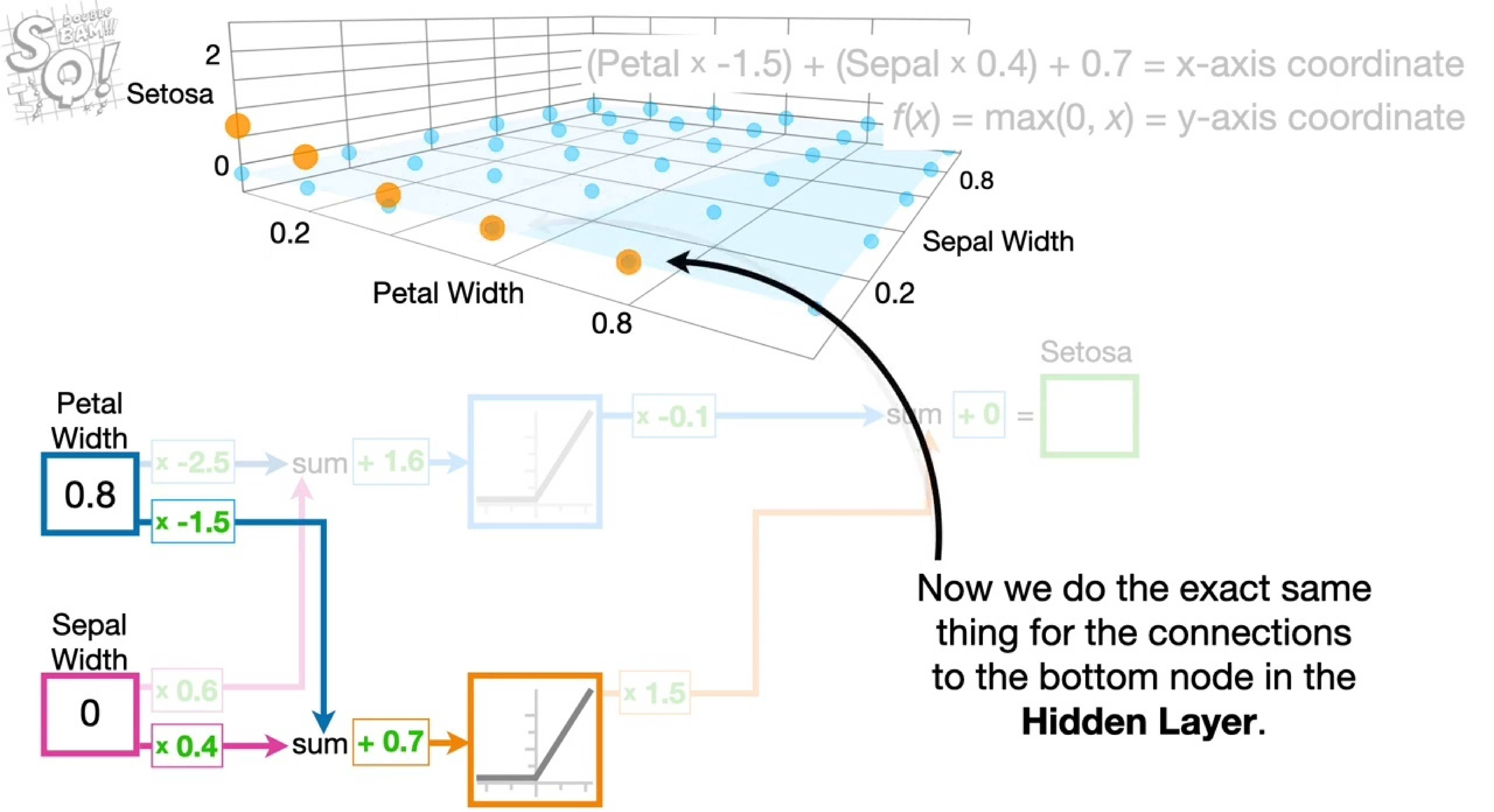
(Petal × -1.5) + (Sepal × 0.4) + 0.7 = x-axis coordinate
f(x) = max(0, x) = y-axis coordinate
2
Setosa
0
0.2
Petal Width
0.8
0.8
Sepal Width
0.2
Setosa
Petal Width
0.2
x -2.5
sum
+ 1.6
x -0.1
sum
+ 0
=
x -1.5
Sepal Width
0
x 0.6
x 0.4
sum
+ 0.7
x 1.5
Now we do the exact same thing for the connections to the bottom node in the **Hidden Layer**.

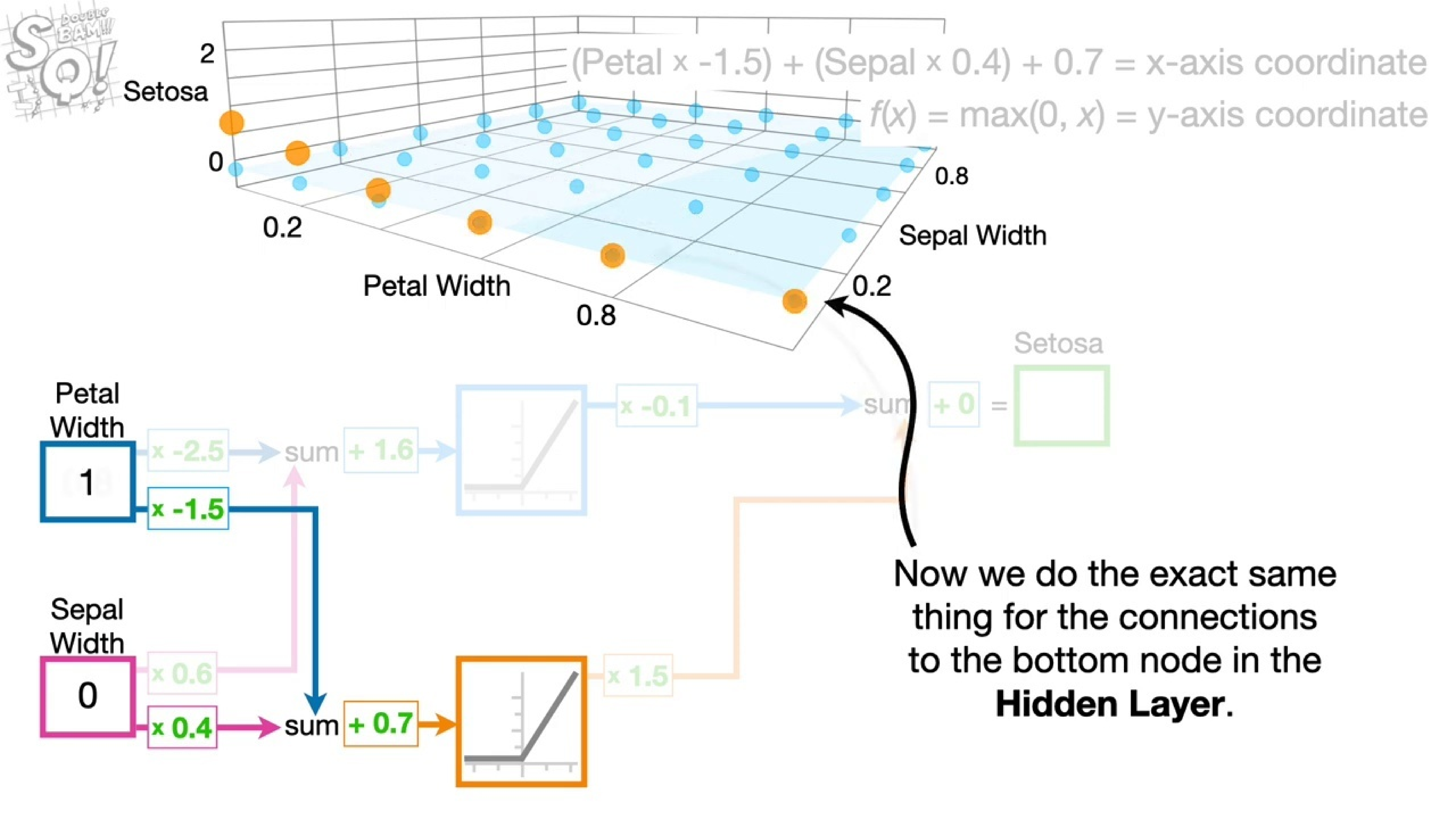


Now we do the exact same thing for the connections to the bottom node in the **Hidden Layer**.

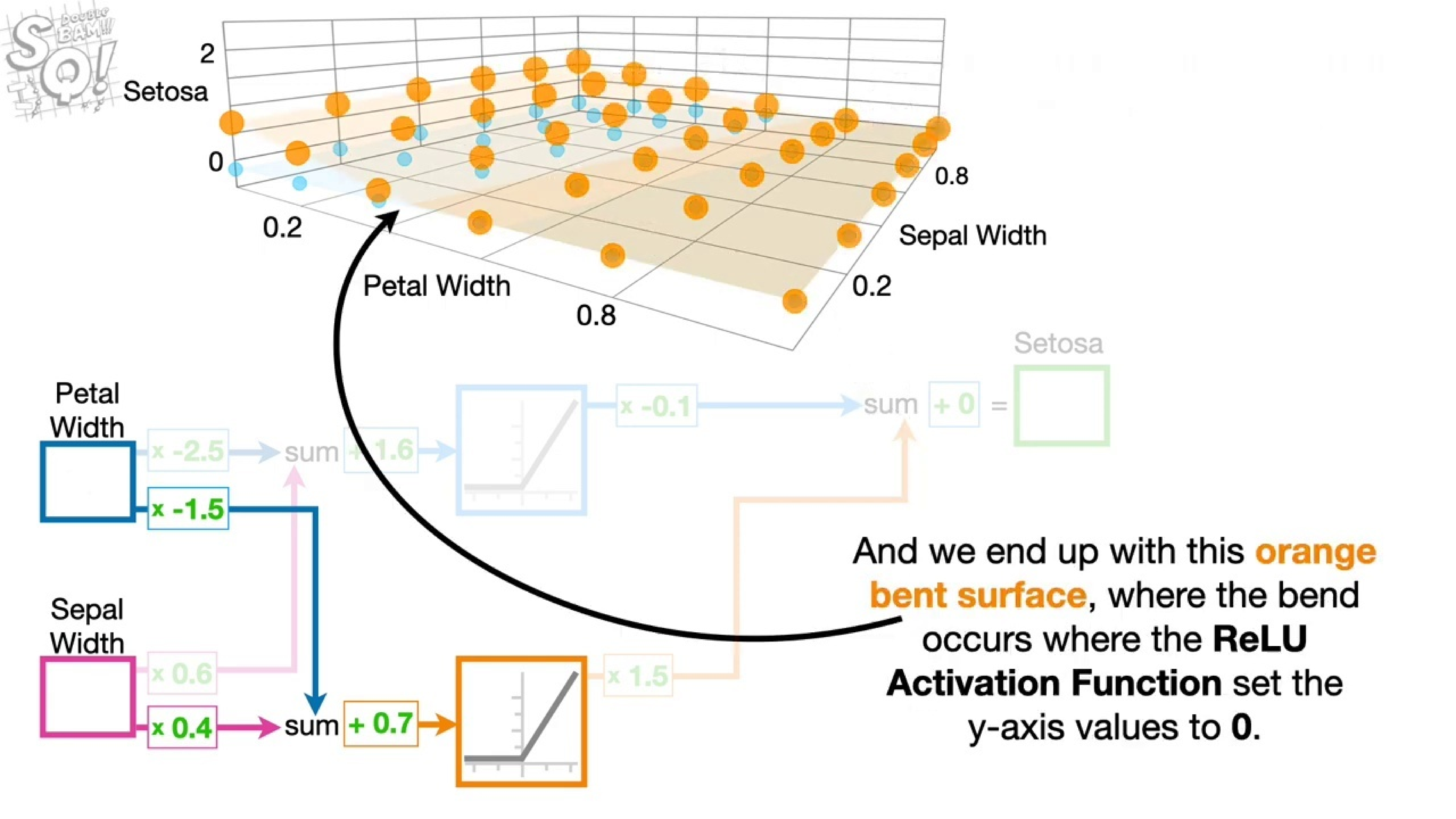


Now we do the exact same thing for the connections to the bottom node in the **Hidden Layer**.

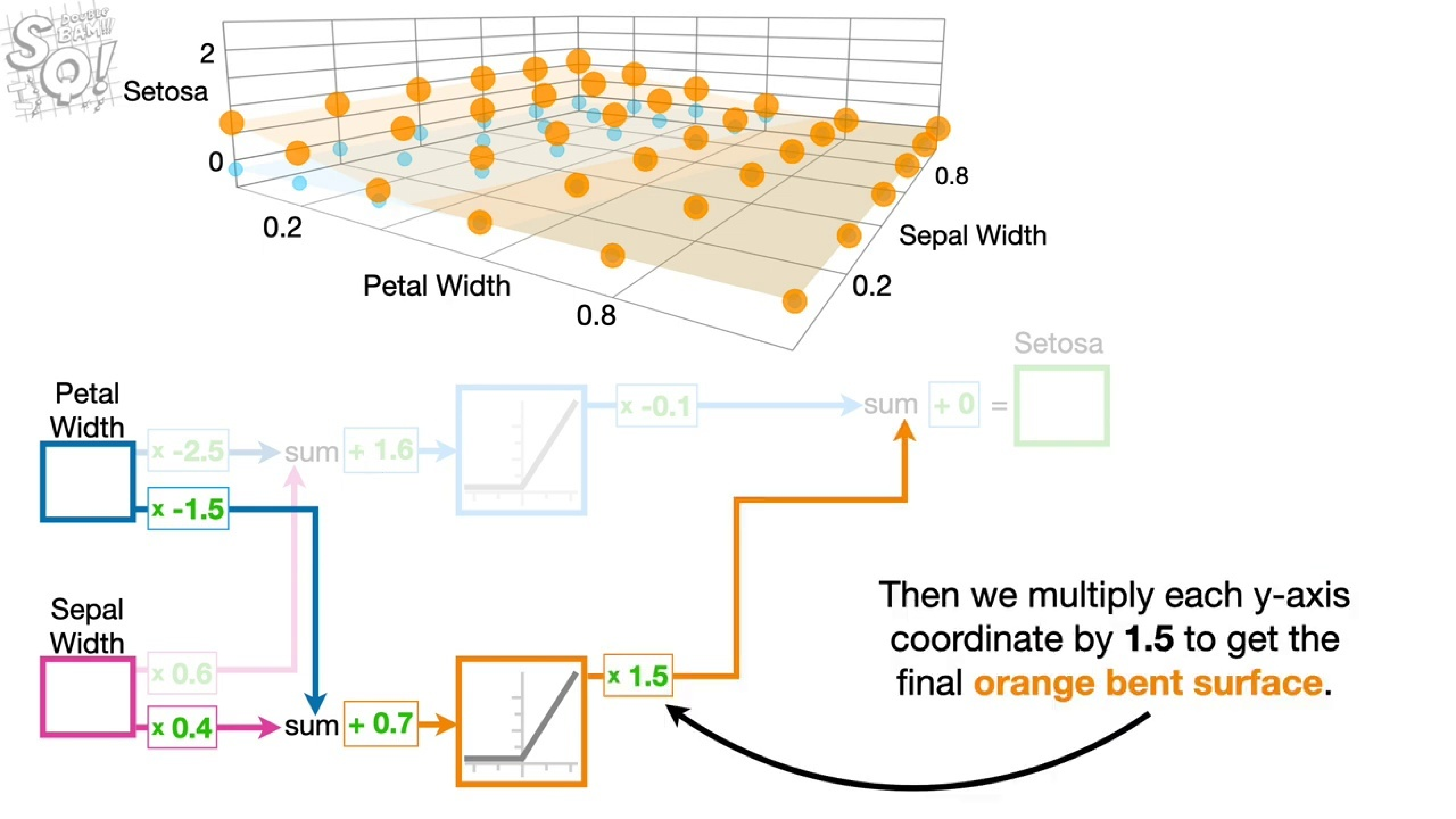

(Petal × -1.5) + (Sepal × 0.4) + 0.7 = x-axis coordinate
f(x) = max(0, x) = y-axis coordinate
Setosa
2
0
0.2
Petal Width
0.8
0.8
Sepal Width
0.2
Setosa
Petal Width
0.8
x -2.5
sum
+ 1.6
x -0.1
sum
+ 0
=
x -1.5
Sepal Width
0
x 0.6
x 0.4
sum
+ 0.7
x 1.5
Now we do the exact same thing for the connections to the bottom node in the **Hidden Layer**.

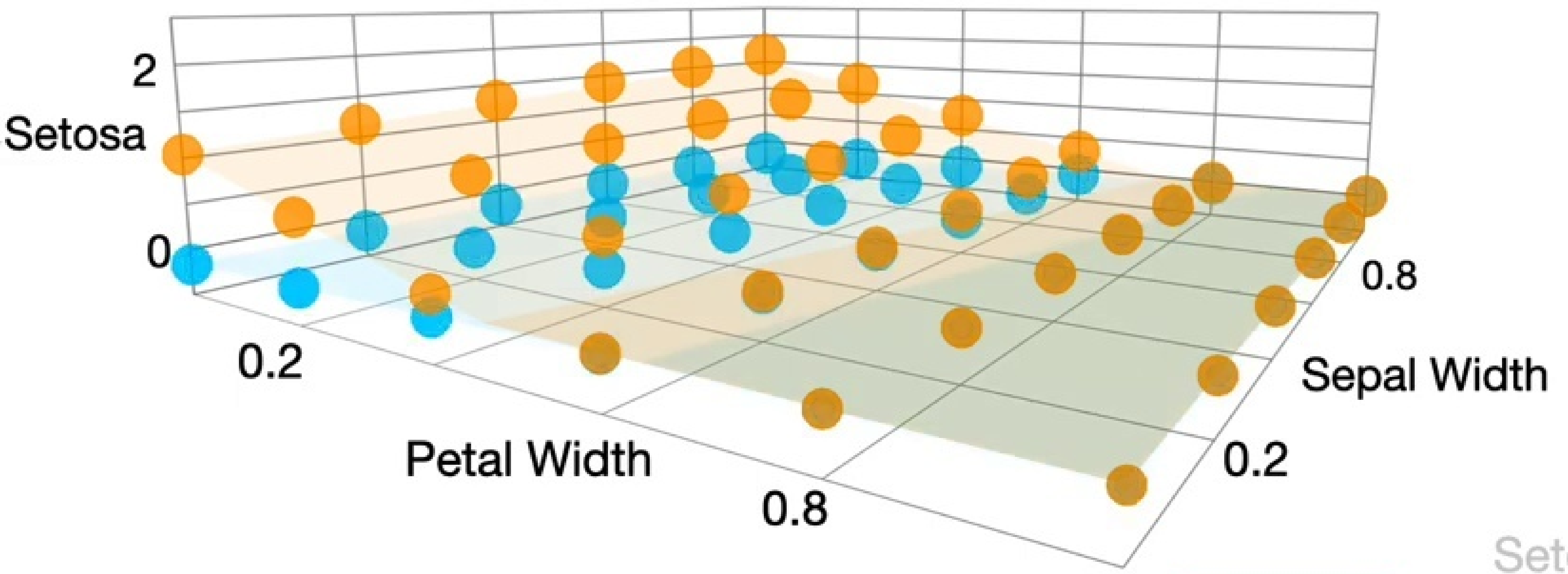


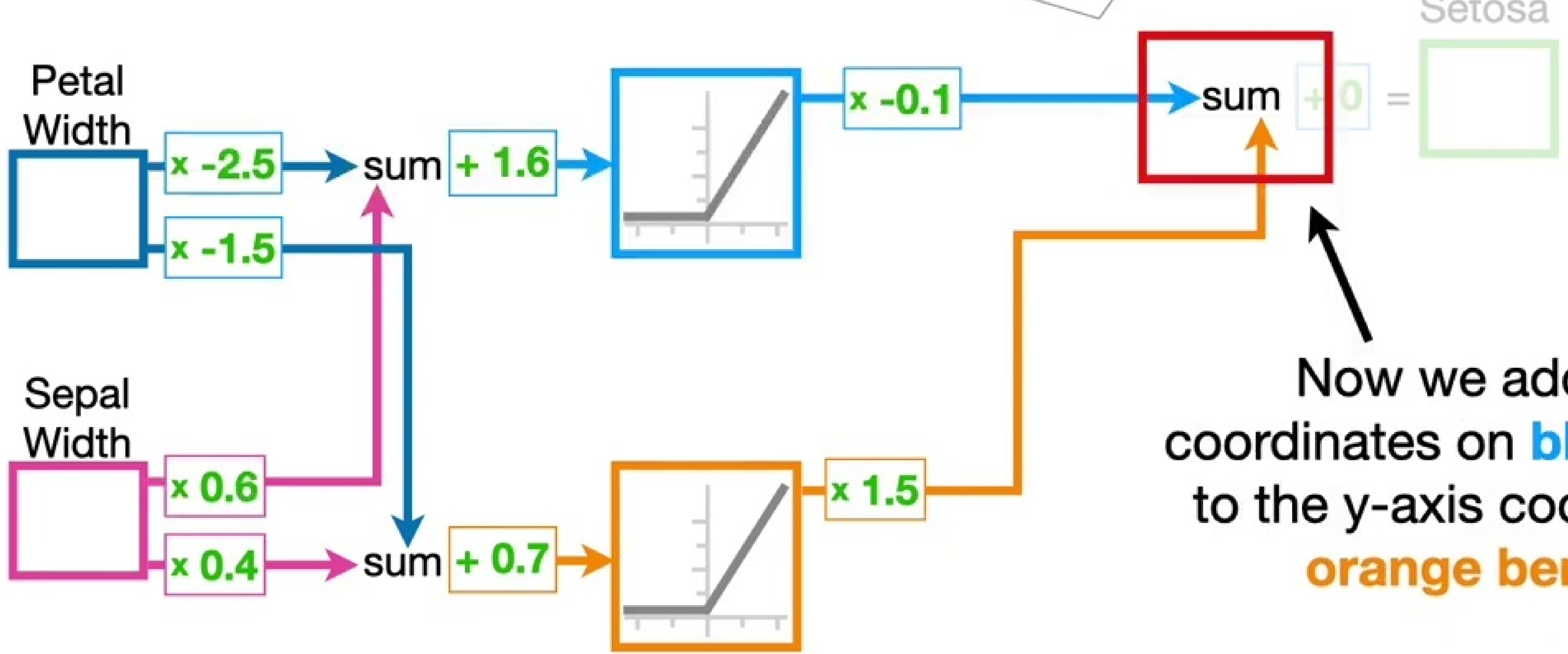
(Petal × -1.5) + (Sepal × 0.4) + 0.7 = x-axis coordinate
f(x) = max(0, x) = y-axis coordinate
Setosa
2
0
0.2
Petal Width
0.8
0.8
Sepal Width
0.2
Setosa
Petal Width
1
x -2.5
sum
+ 1.6
x -0.1
sum
+ 0
=
x -1.5
Sepal Width
0
x 0.6
x 0.4
sum
+ 0.7
x 1.5
Now we do the exact same thing for the connections to the bottom node in the **Hidden Layer**.



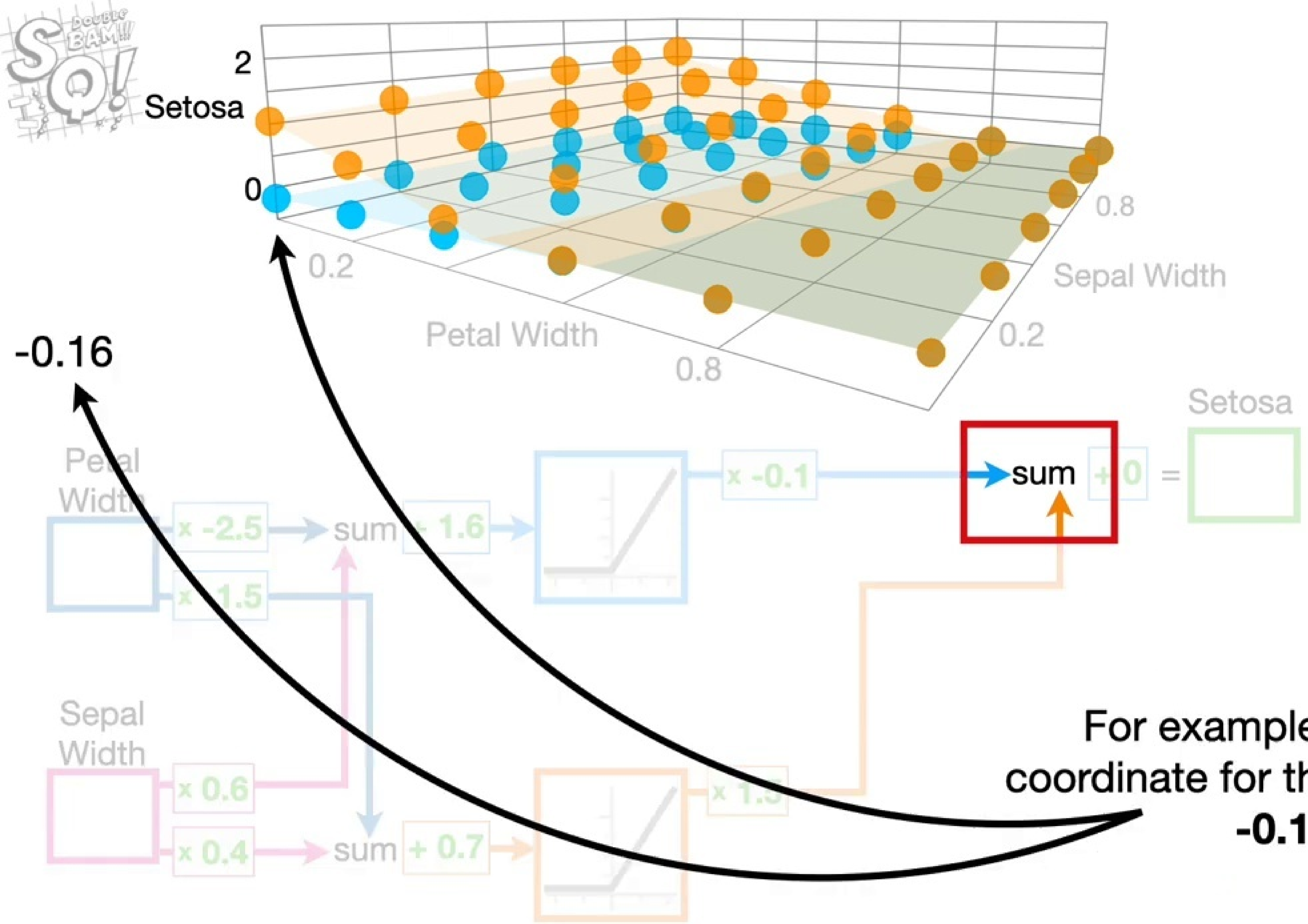
Setosa
2
0
0.2
0.8
Petal Width
Sepal Width
0.2
0.8
Setosa
Petal Width
Sepal Width
x -2.5
x -1.5
x 0.6
x 0.4
sum
+ 1.6
sum
+ 0.7
x -0.1
x 1.5
sum
+ 0
=
And we end up with this orange bent surface, where the bend occurs where the ReLU Activation Function set the y-axis values to 0.


Setosa
2
0
0.2
Petal Width
0.8
0.8
Sepal Width
0.2
Setosa
Petal Width
x -2.5
sum
+ 1.6
x -0.1
sum
+ 0
=
x -1.5
Sepal Width
x 0.6
x 0.4
sum
+ 0.7
x 1.5
Then we multiply each y-axis coordinate by **1.5** to get the final **orange bent surface**.

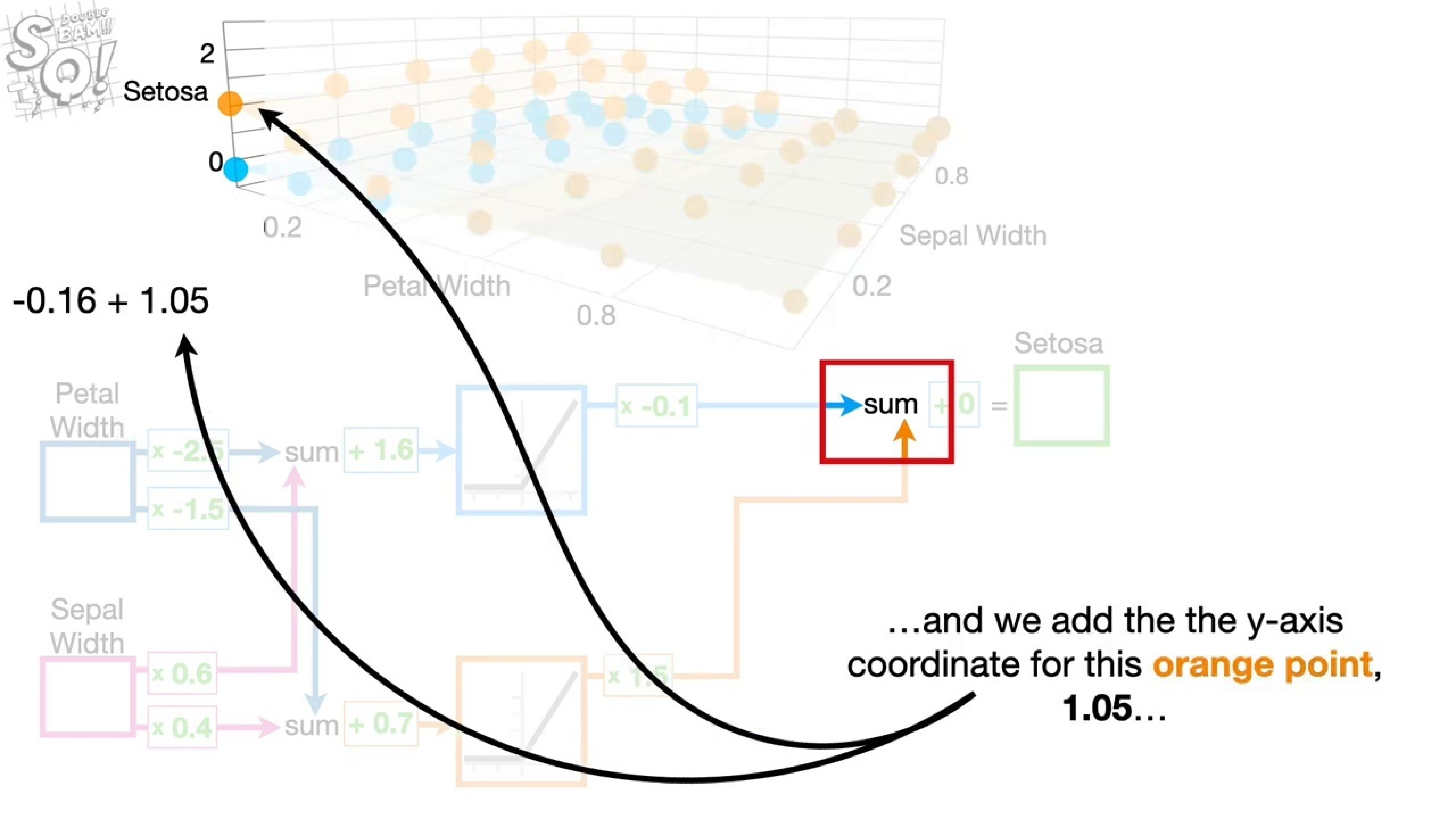




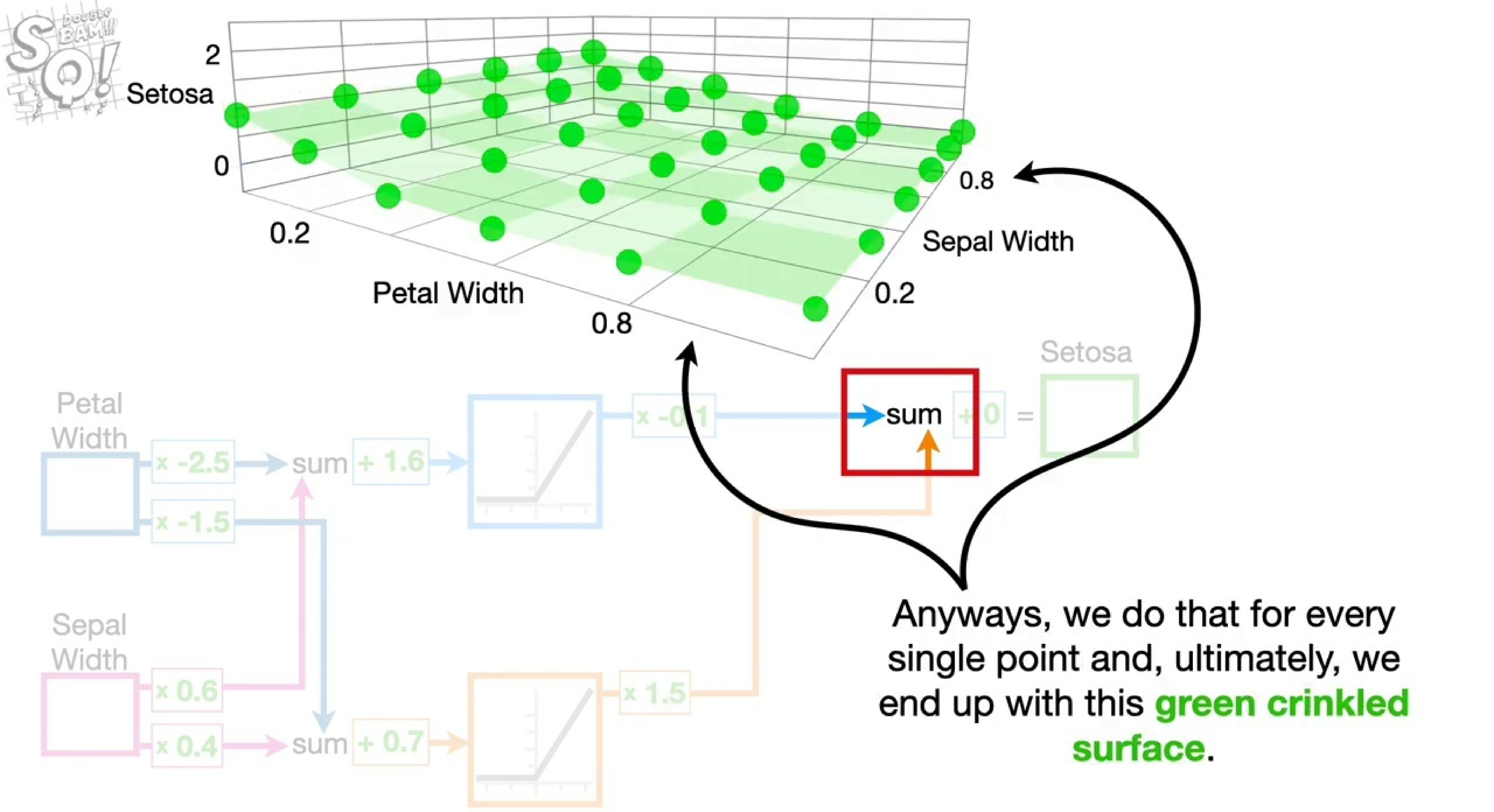
Now we add the y-axis coordinates on **blue bent surface** to the y-axis coordinates on the **orange bent surface**.



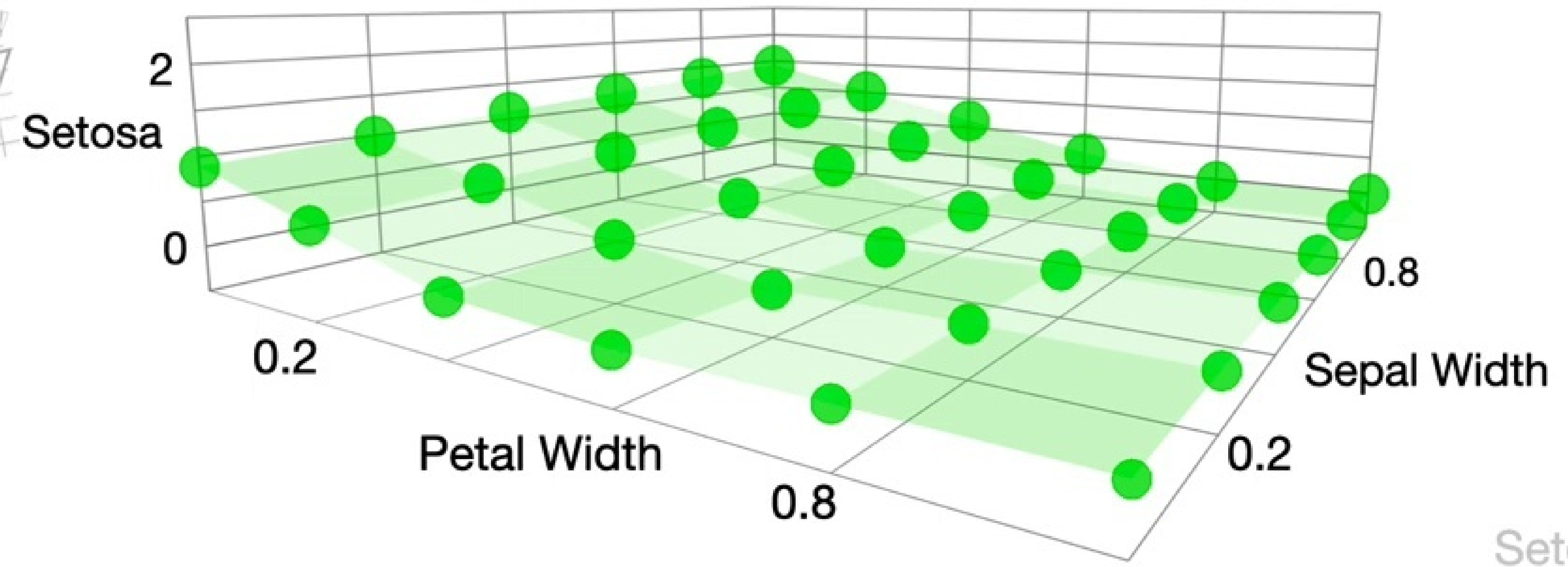
For example, the y-axis coordinate for this **blue point** is **-0.16**…

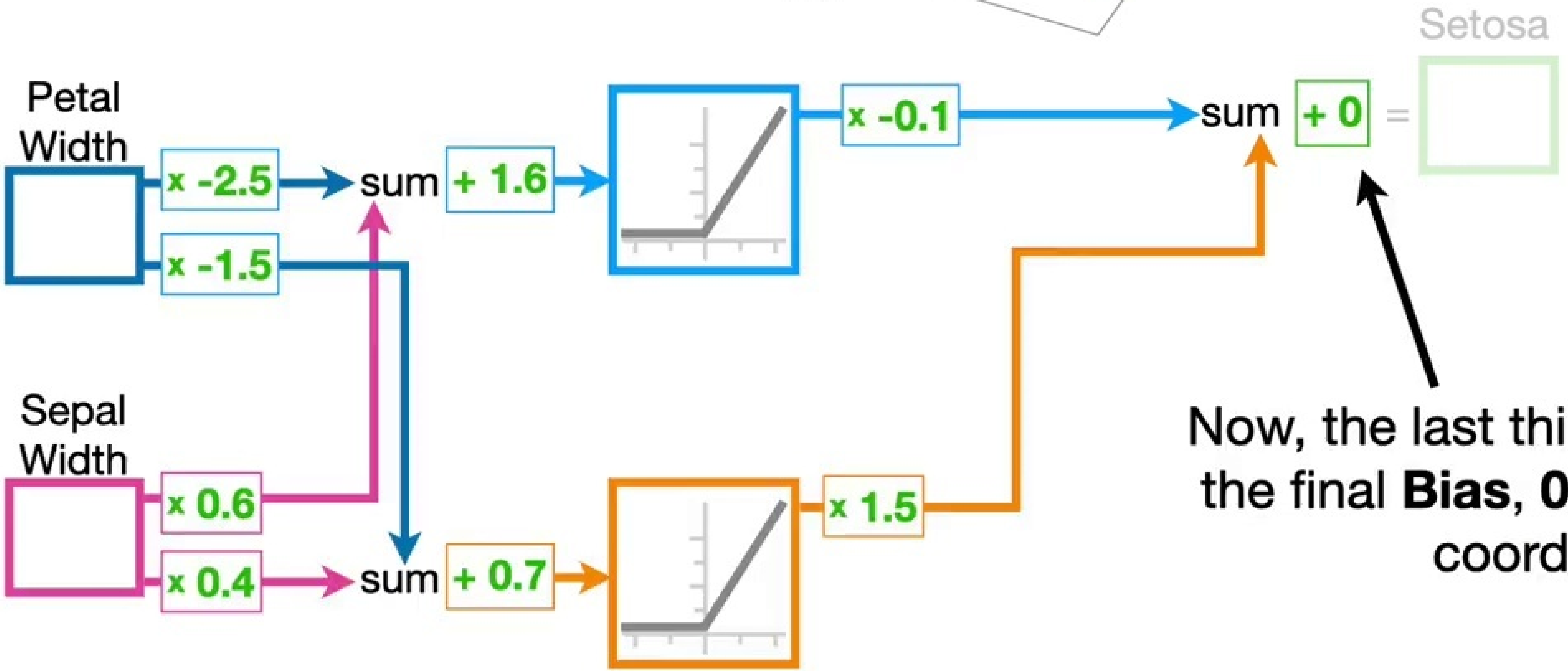


2
Setosa
0
0.2
Petal Width
0.8
0.8
Sepal Width
0.2
-0.16 + 1.05
Setosa
Petal Width
x -2.5
x -1.5
sum
+ 1.6
x -0.1
sum
+ 0
=
Sepal Width
x 0.6
x 0.4
sum
+ 0.7
x 1.5
…and we add the the y-axis coordinate for this **orange point**, **1.05**…

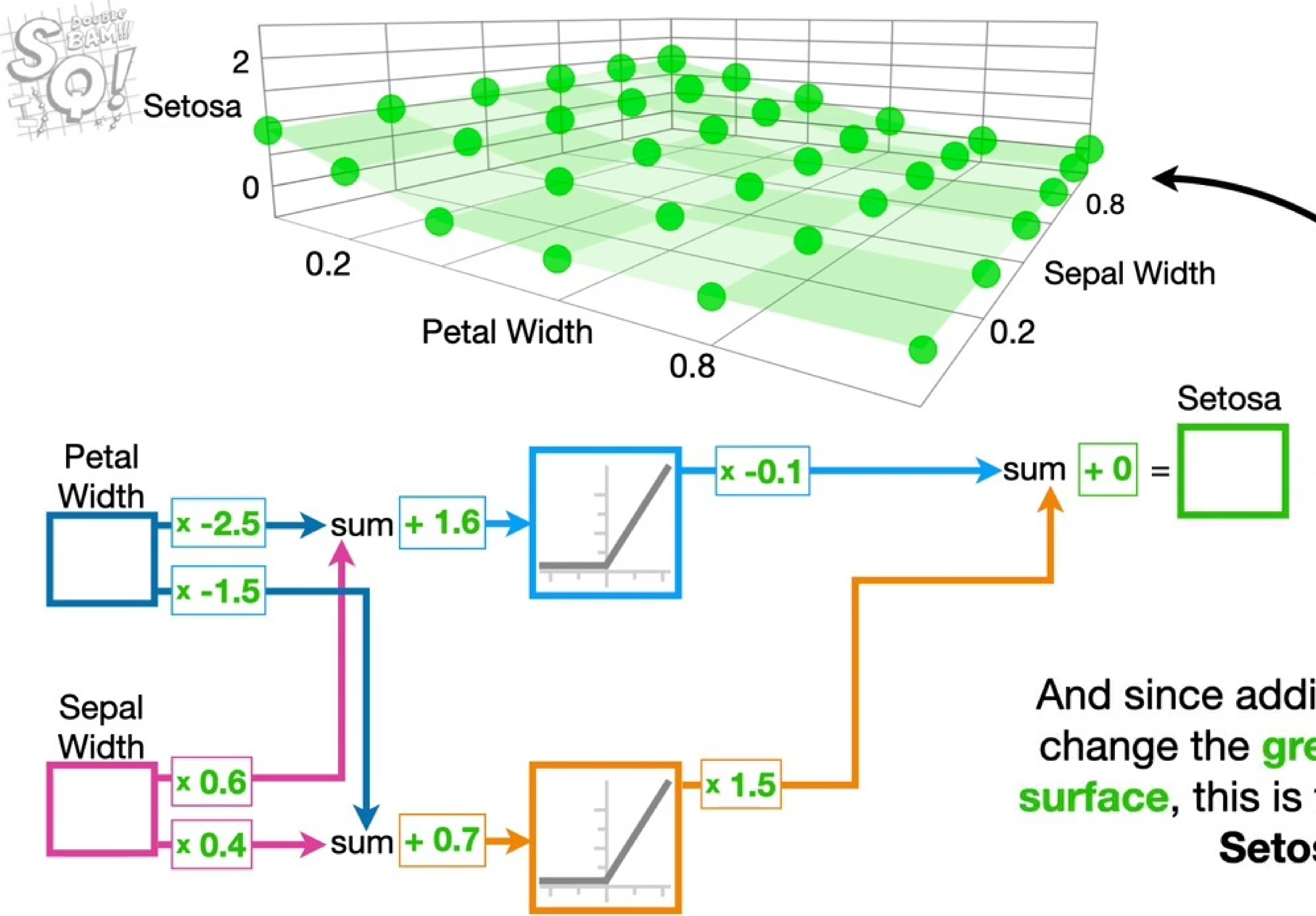


Setosa
2
0
0.2
Petal Width
0.8
0.8
Sepal Width
0.2
Setosa
sum
+ 0
=
Petal Width
x -2.5
x -1.5
sum
+ 1.6
Sepal Width
x 0.6
x 0.4
sum
+ 0.7
x 1.5
Anyways, we do that for every single point and, ultimately, we end up with this **green crinkled surface**.

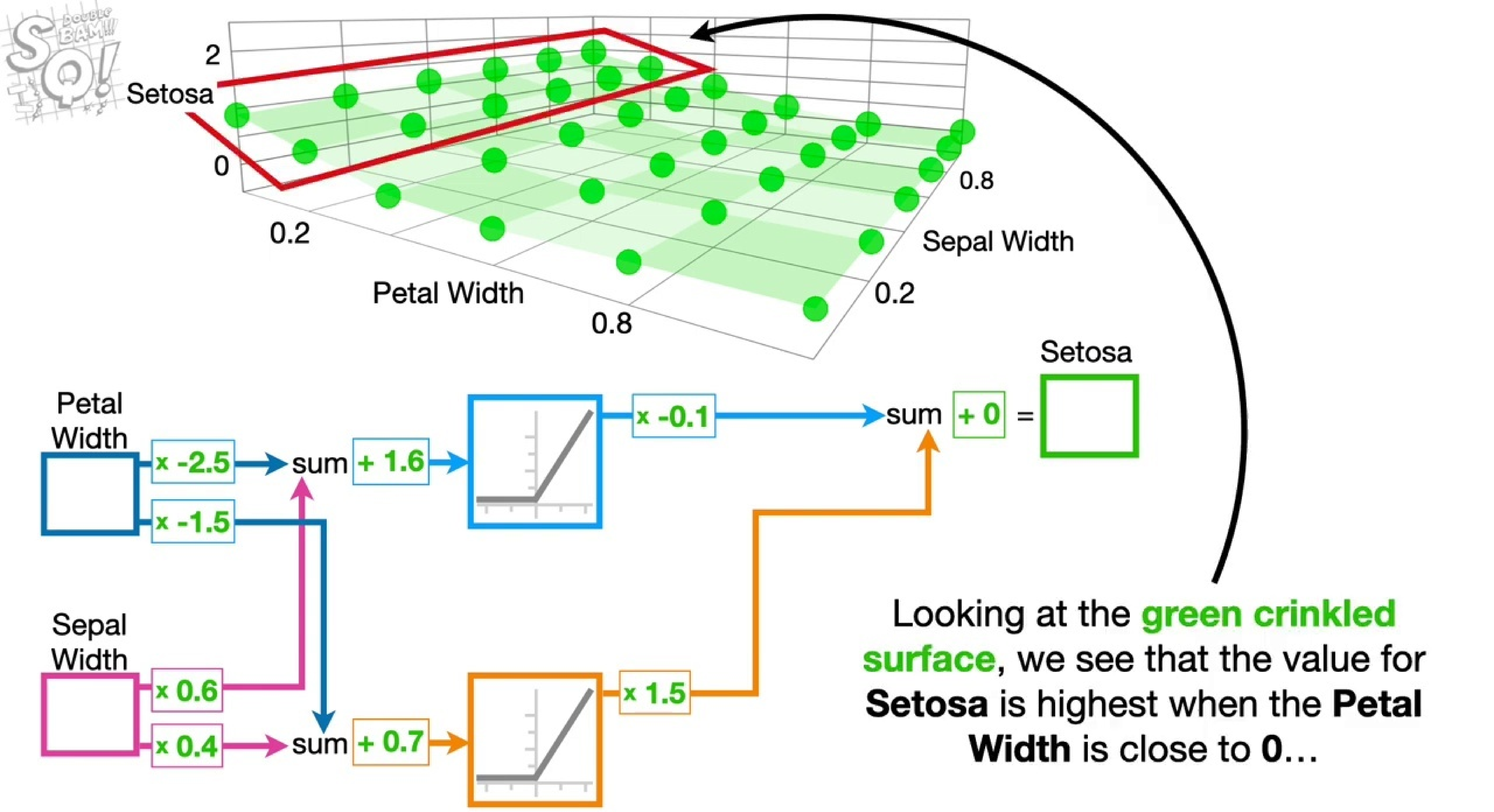




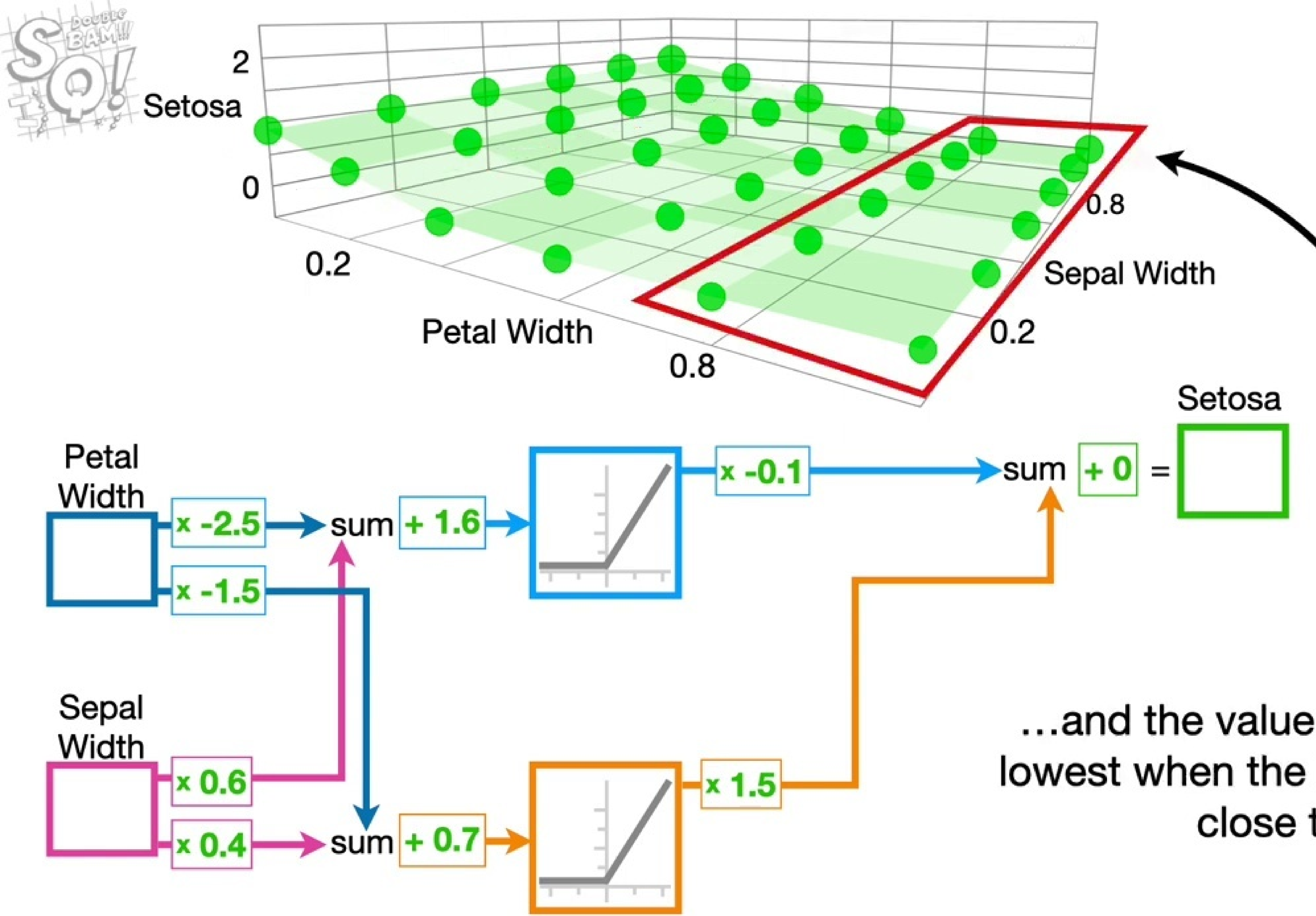
Now, the last thing we do is add the final **Bias**, **0**, to each y-axis coordinate.



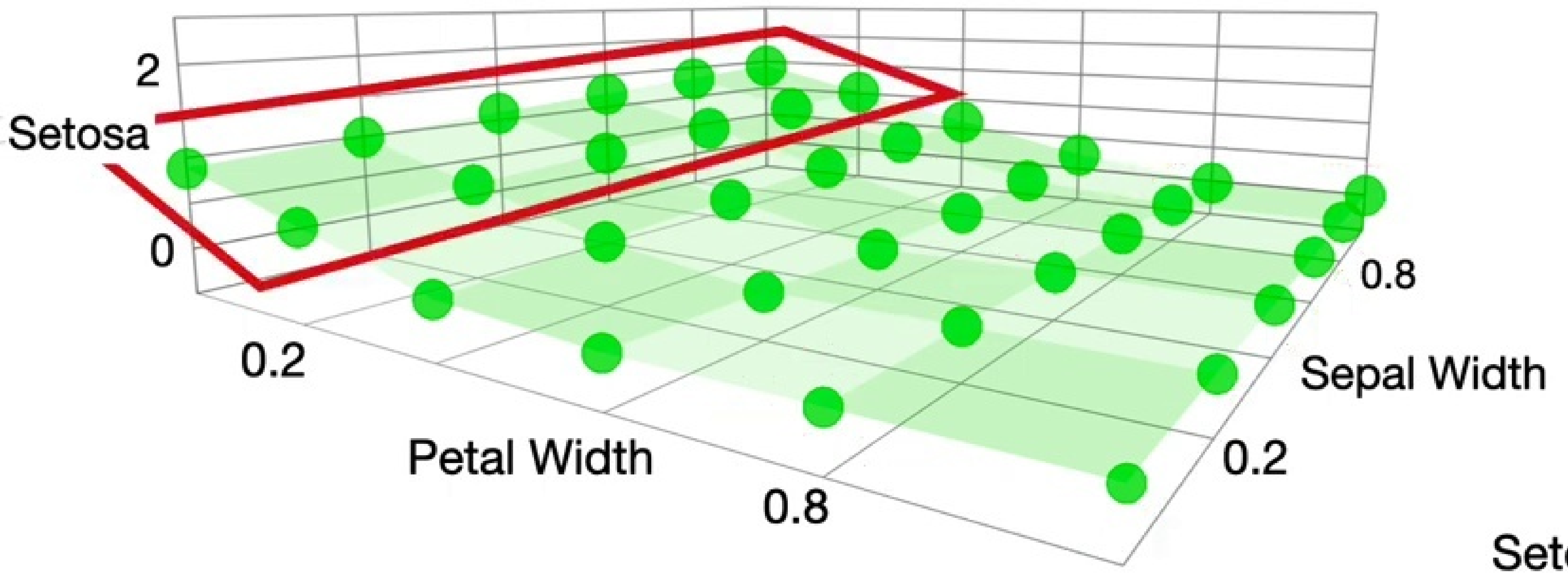
And since adding **0** doesn't change the **green crinkled surface**, this is the output for **Setosa**.

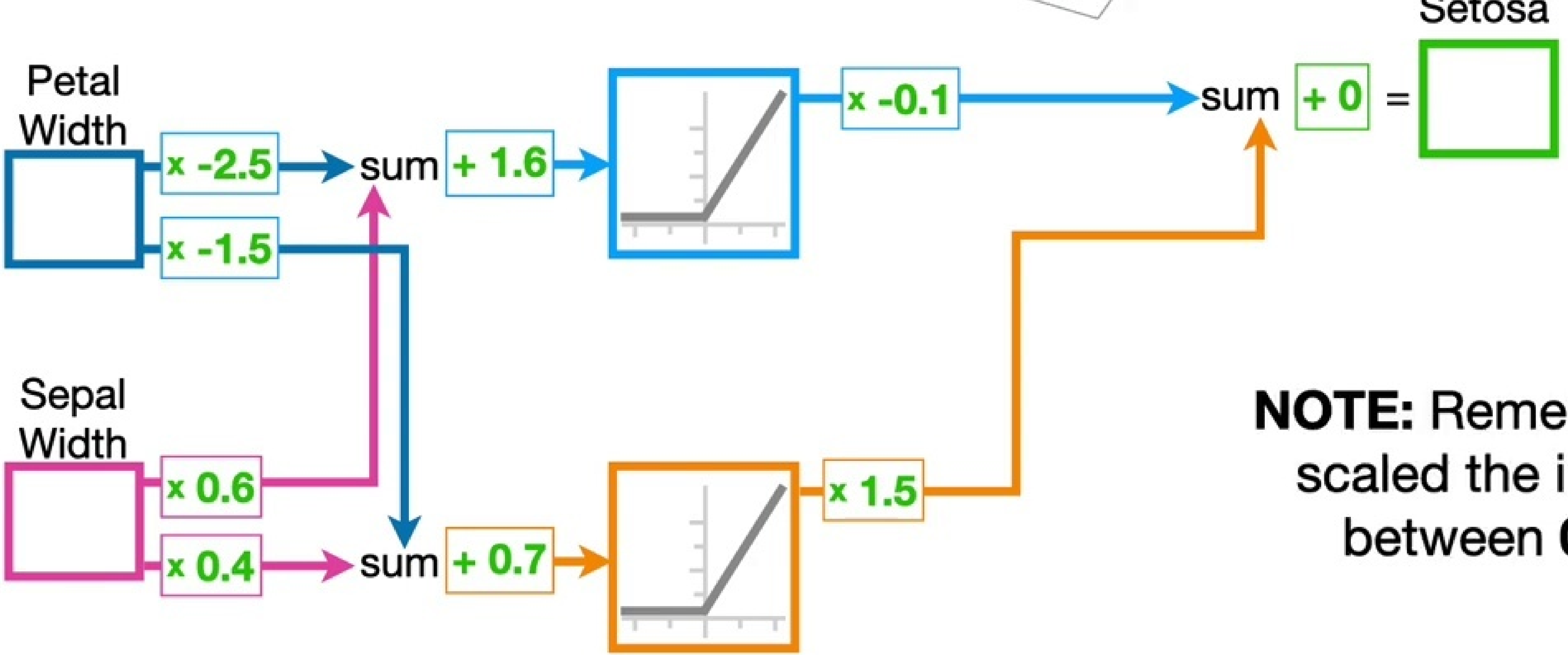


2
Setosa
0
0.2
Petal Width
0.8
0.8
Sepal Width
0.2
Petal Width
Sepal Width
x -2.5
x -1.5
x 0.6
x 0.4
sum
+ 1.6
sum
+ 0.7
x -0.1
x 1.5
sum
+ 0
=
Setosa
Looking at the green crinkled surface, we see that the value for Setosa is highest when the Petal Width is close to 0…

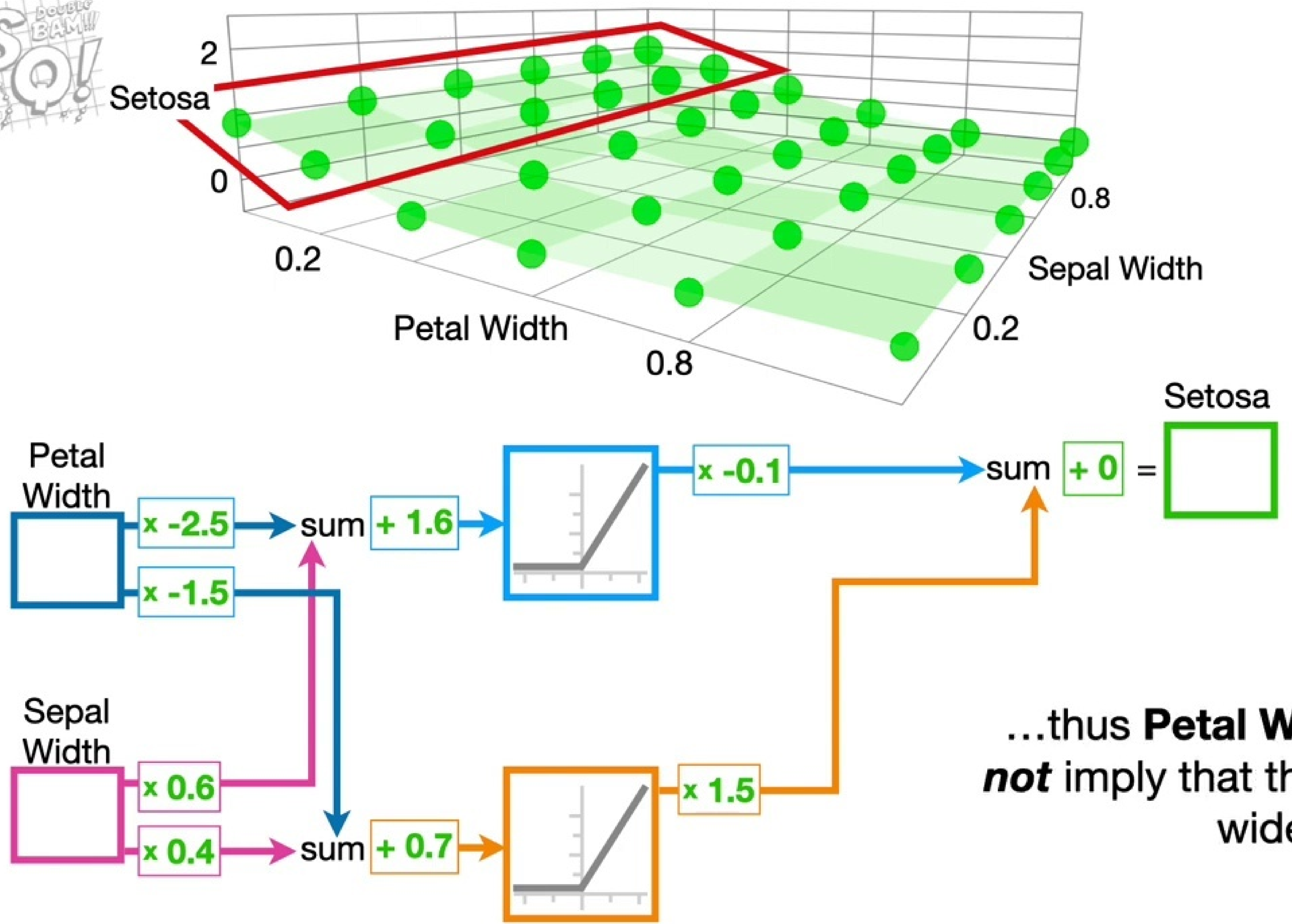


…and the value for **Setosa** is lowest when the **Petal Width** is close to **1**.

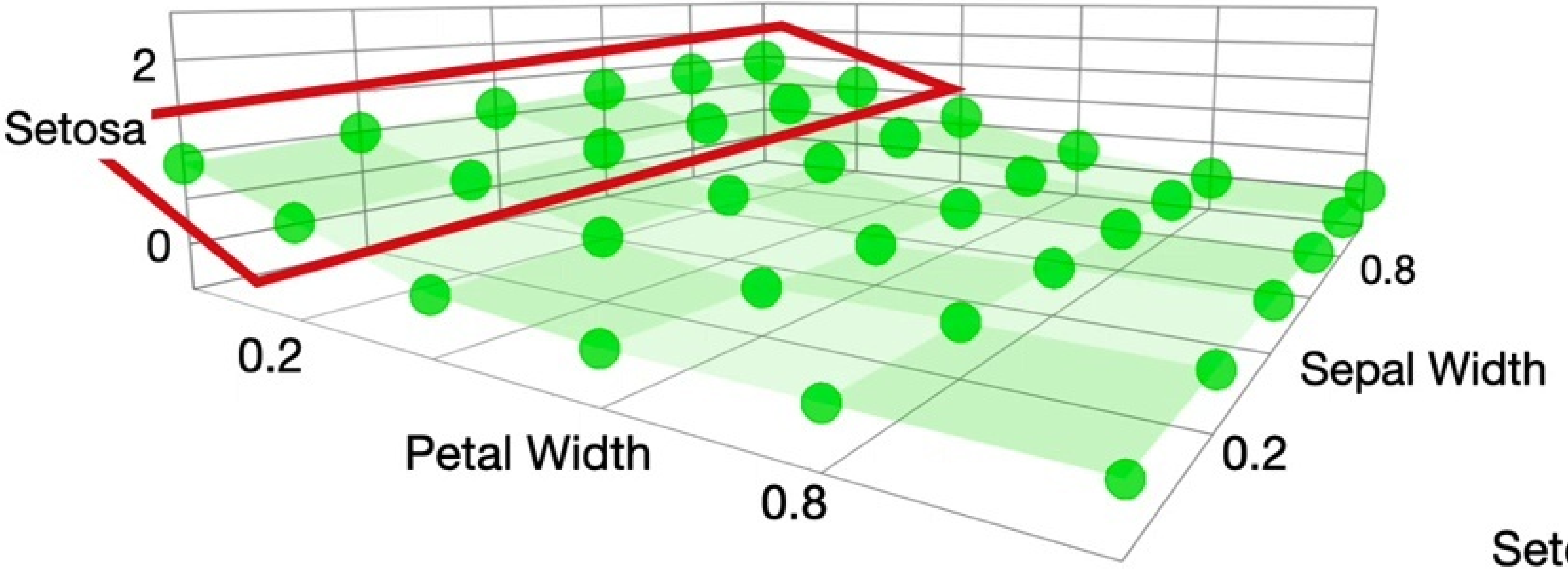


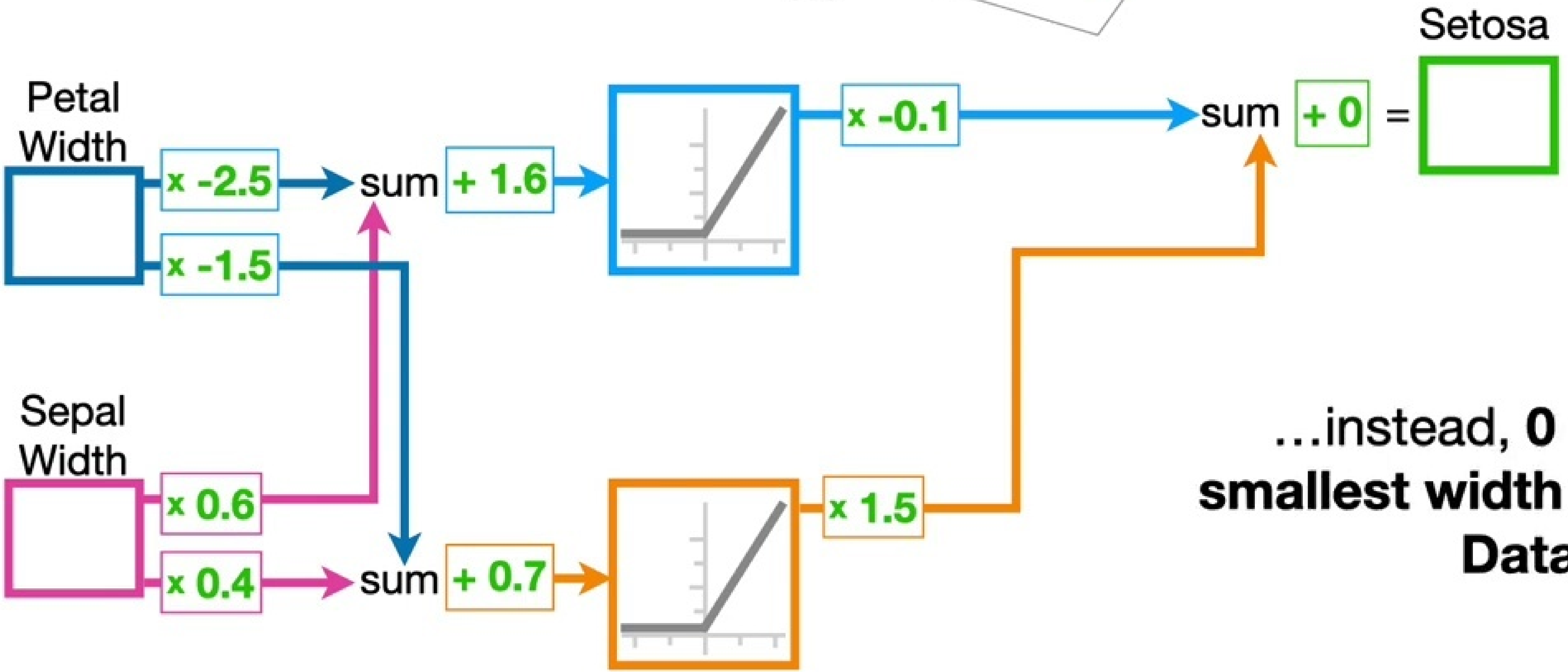


NOTE: Remember that we scaled the inputs to be between **0** and **1**…

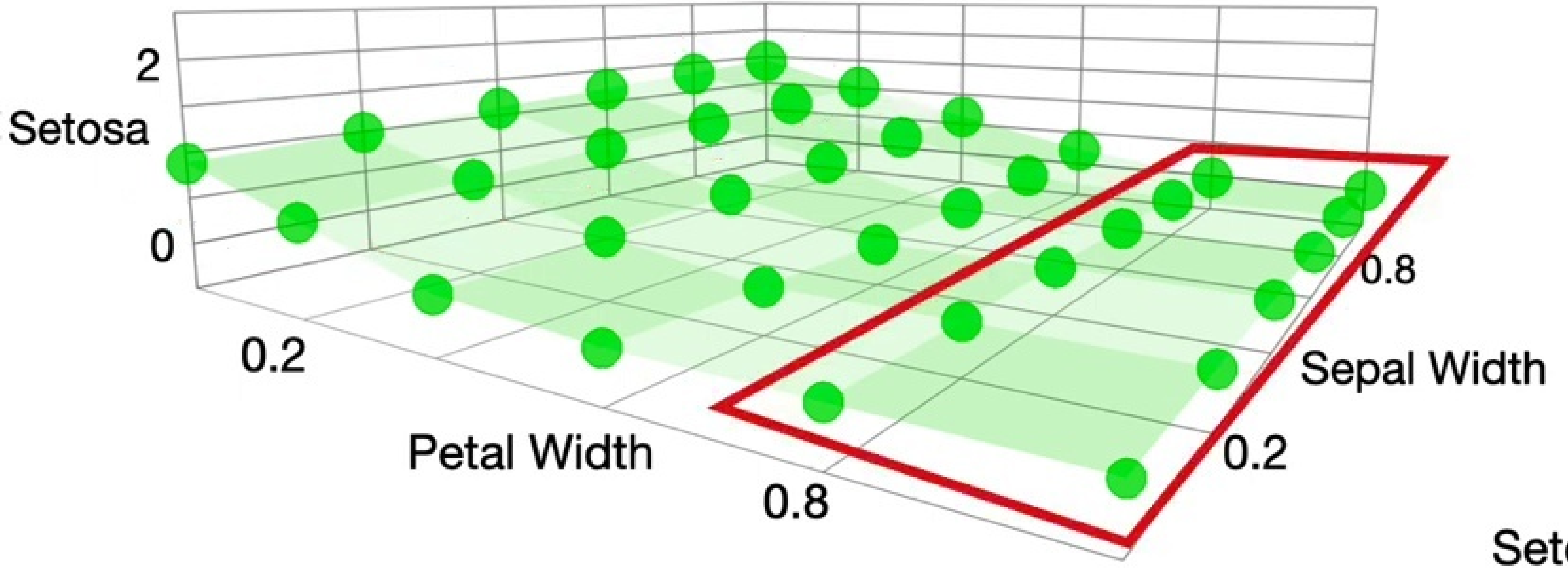


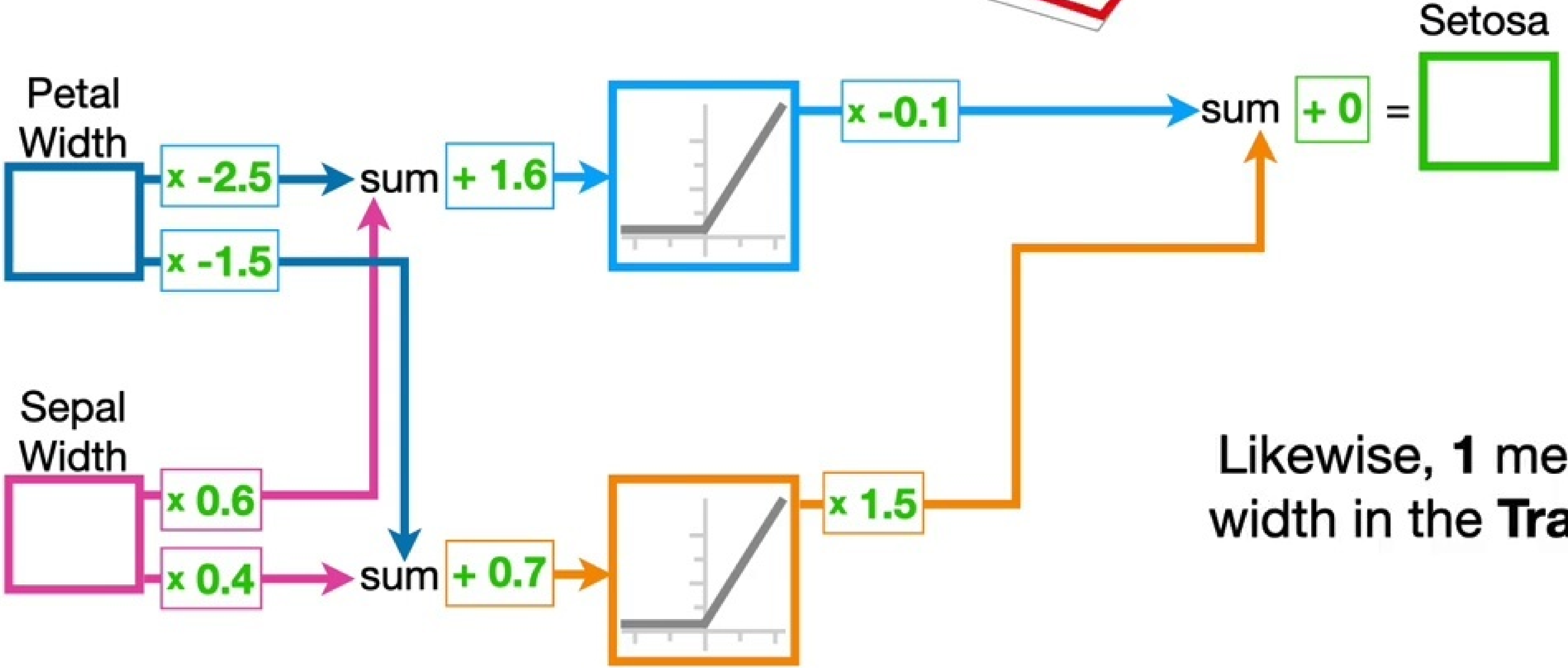
…thus **Petal Width = 0** ***does not*** imply that the petal is **0** cm wide…



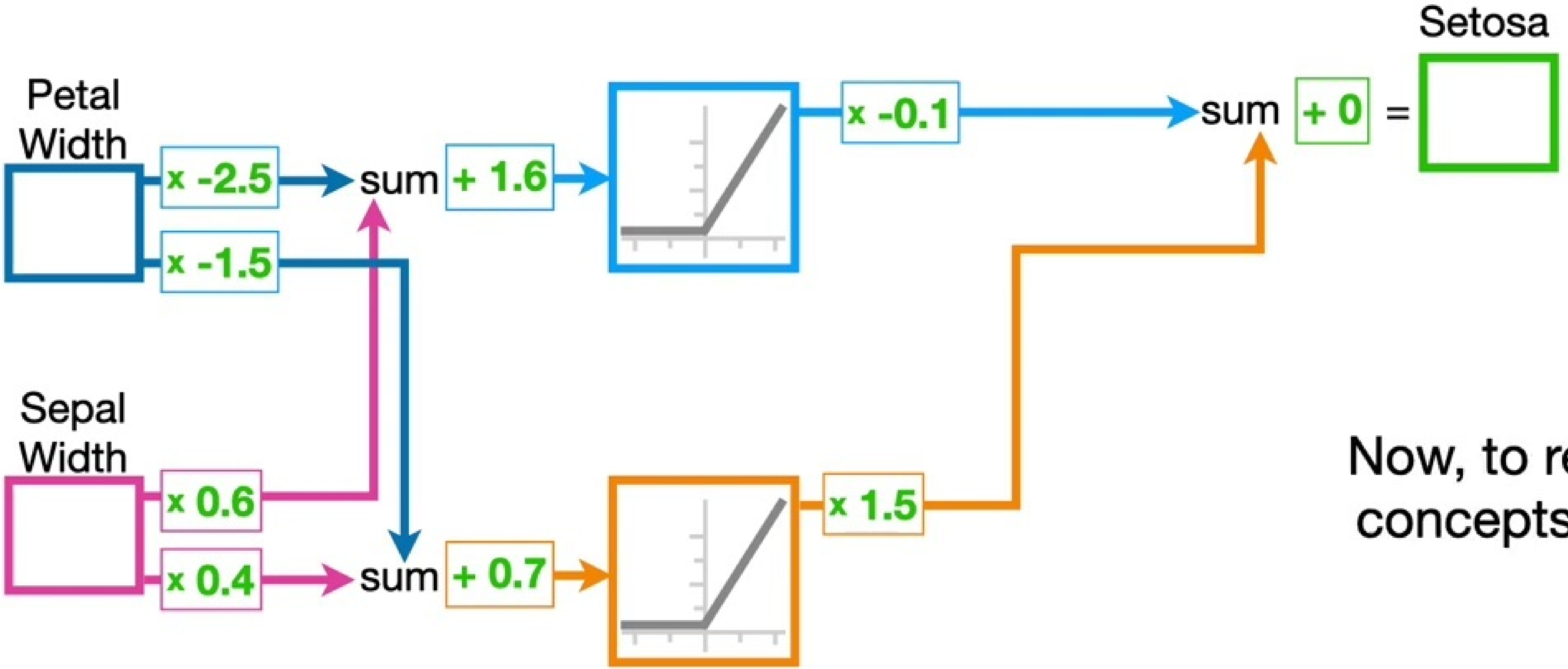


…instead, **0** refers to **the smallest width in the Training Dataset.**

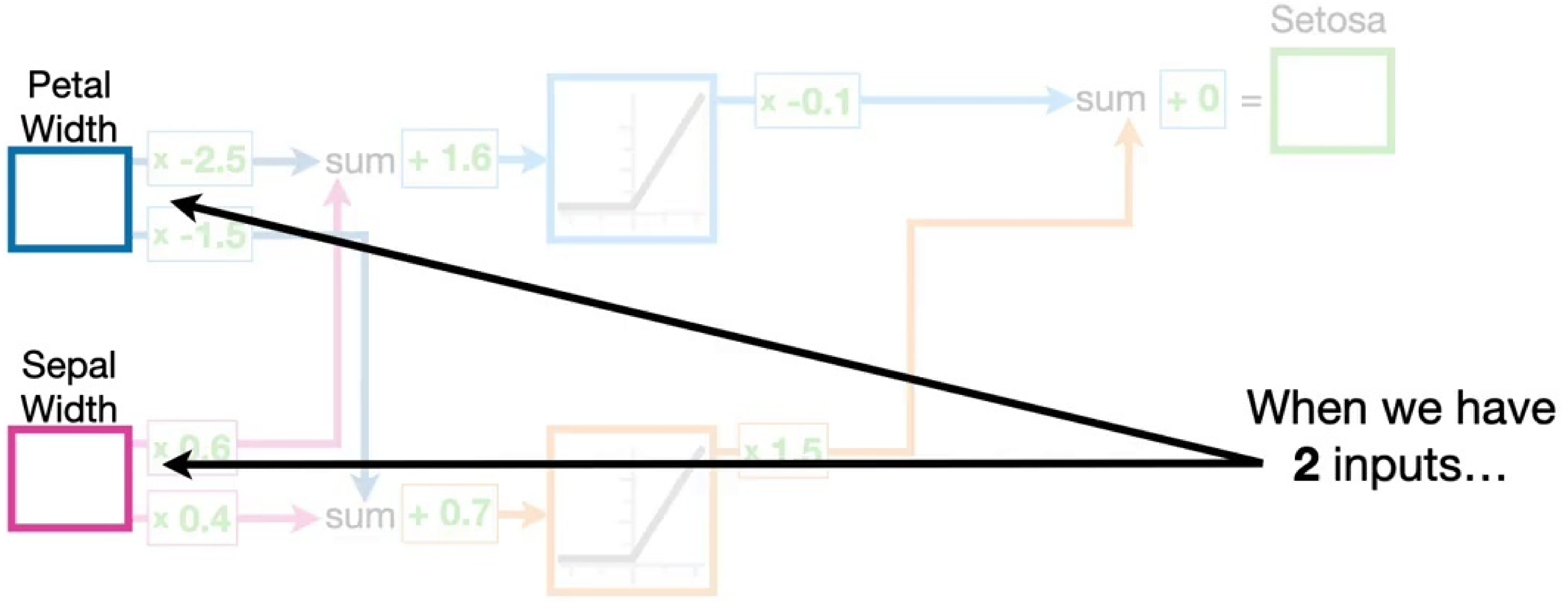


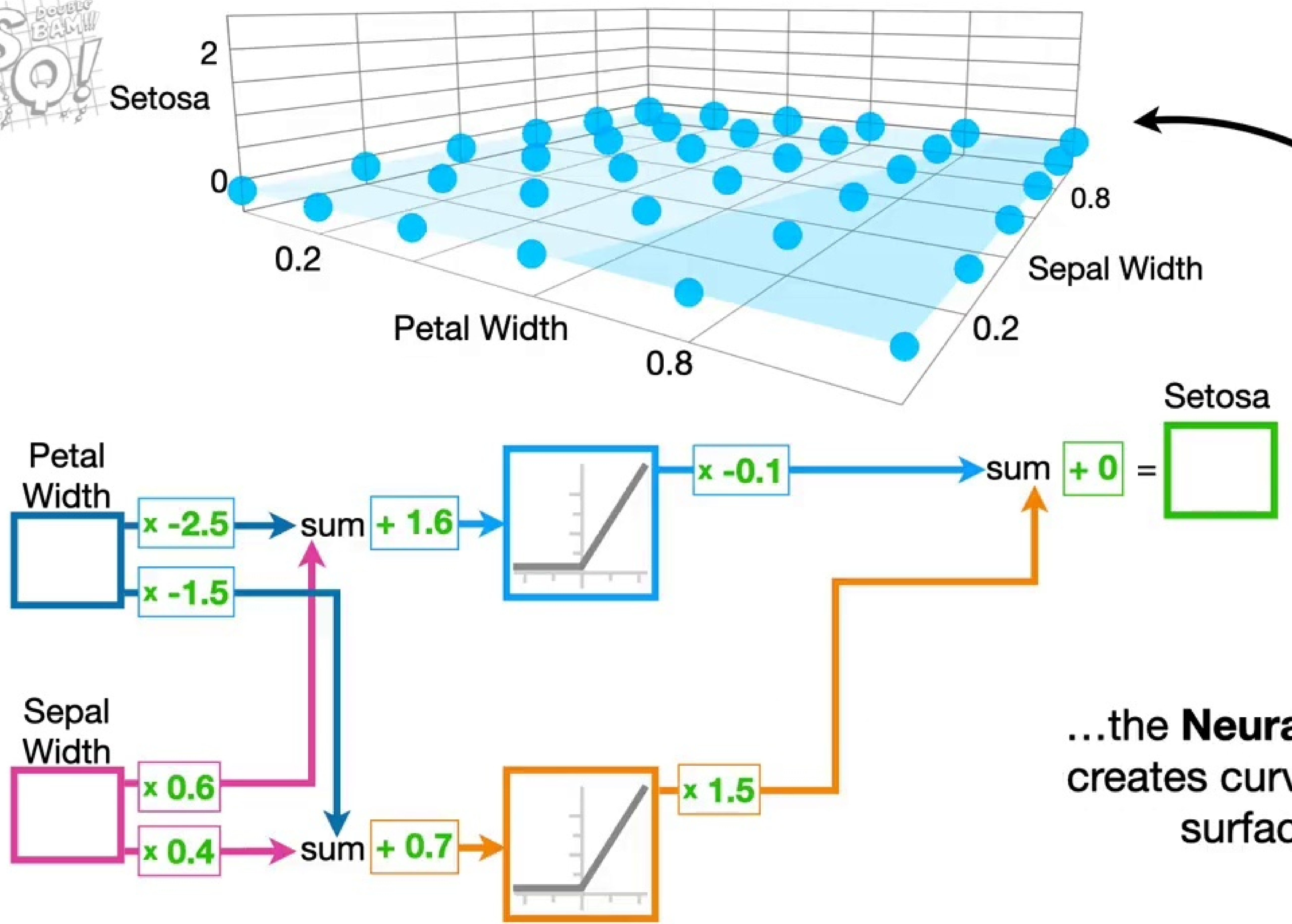


Likewise, **1** means the largest width in the **Training Dataset**.

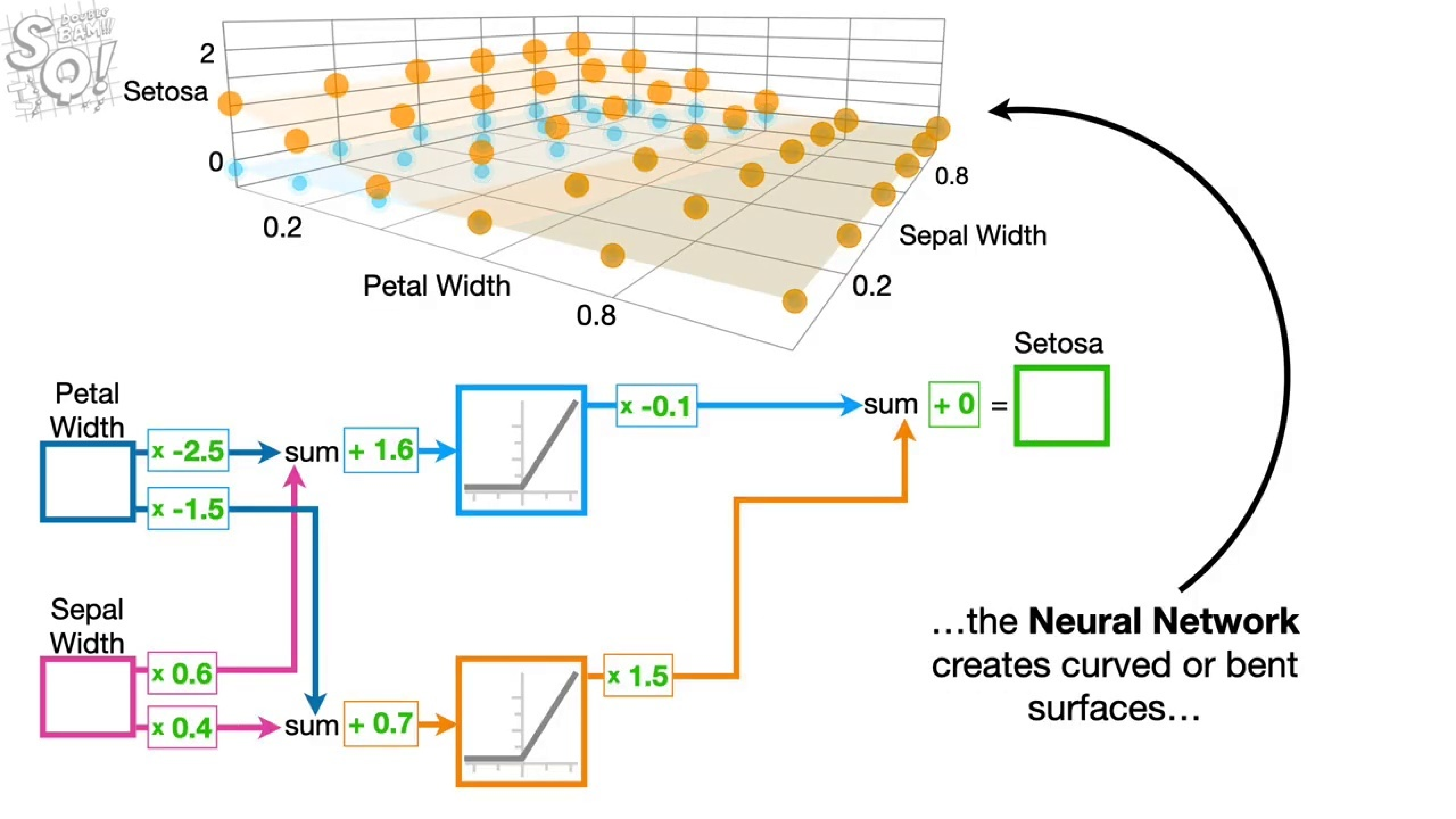


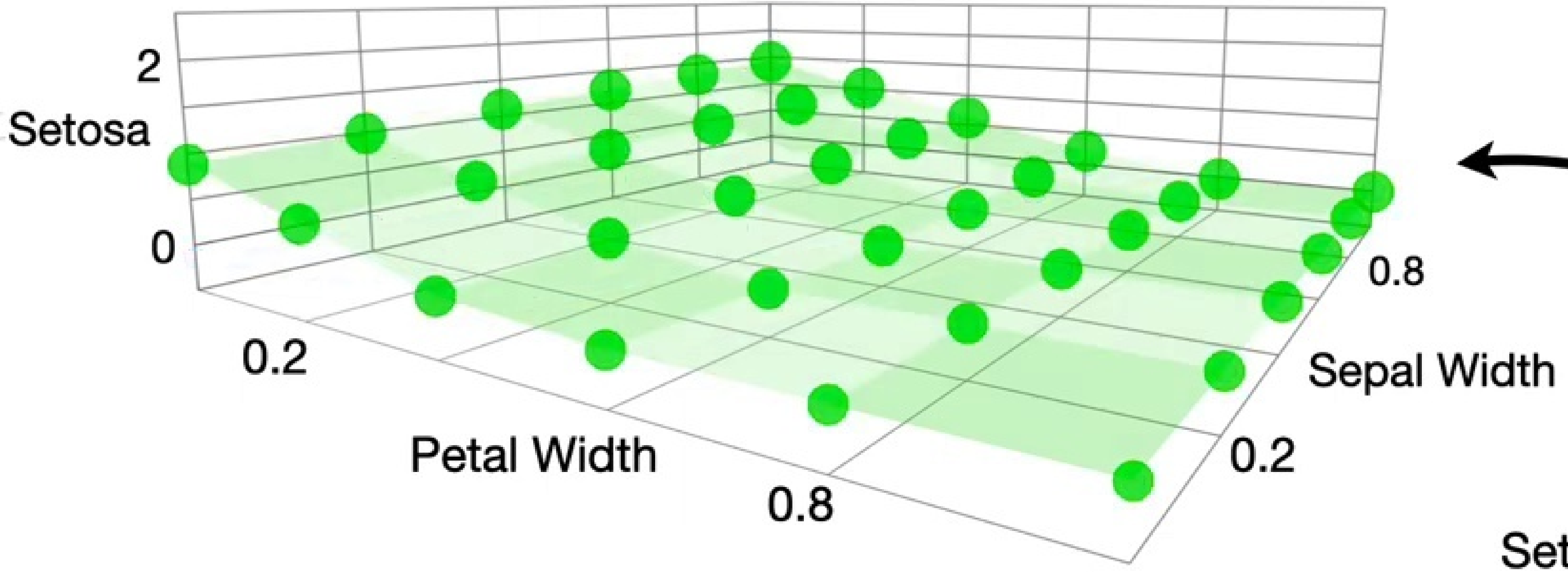
Now, to review the concepts so far…

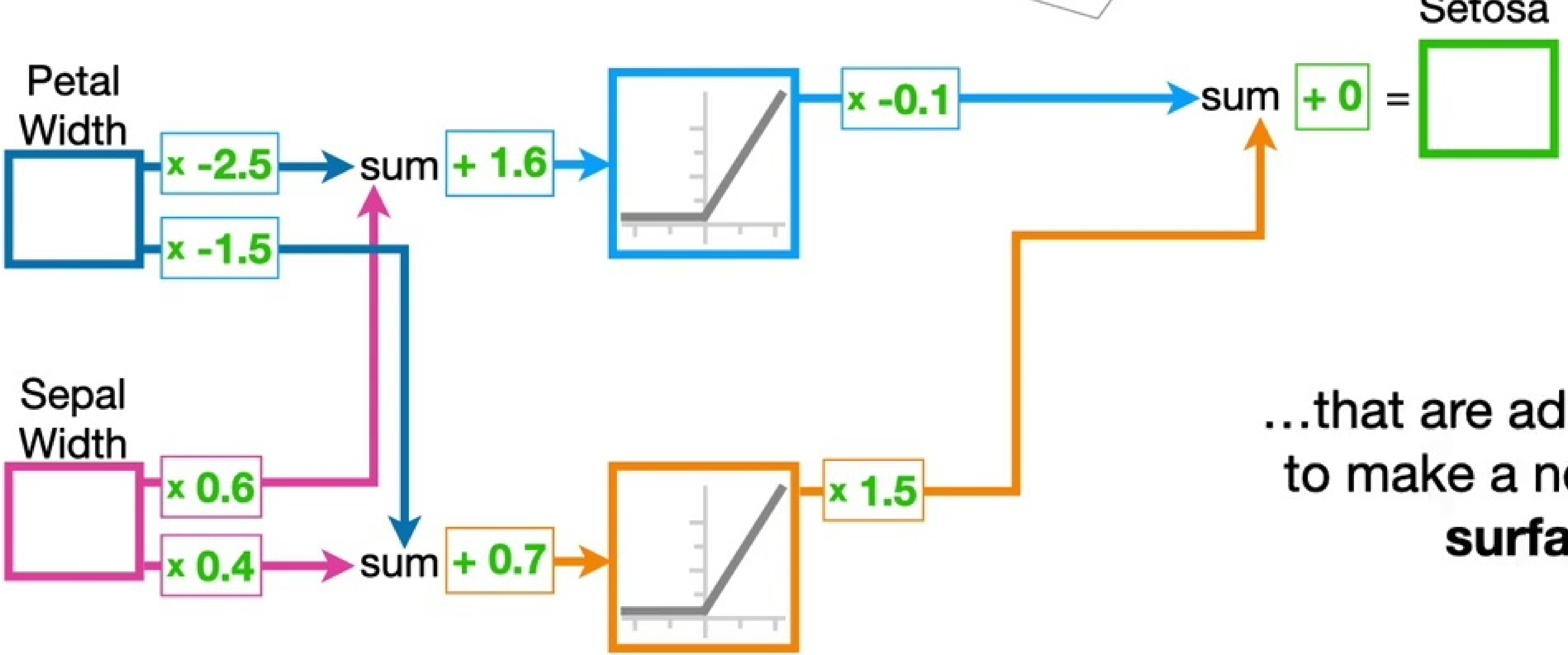

Setosa
Petal Width
x -2.5
sum
+ 1.6
x -0.1
sum
+ 0
=
x -1.5
Sepal Width
x 0.6
x 1.5
x 0.4
sum
+ 0.7
When we have
2 inputs…



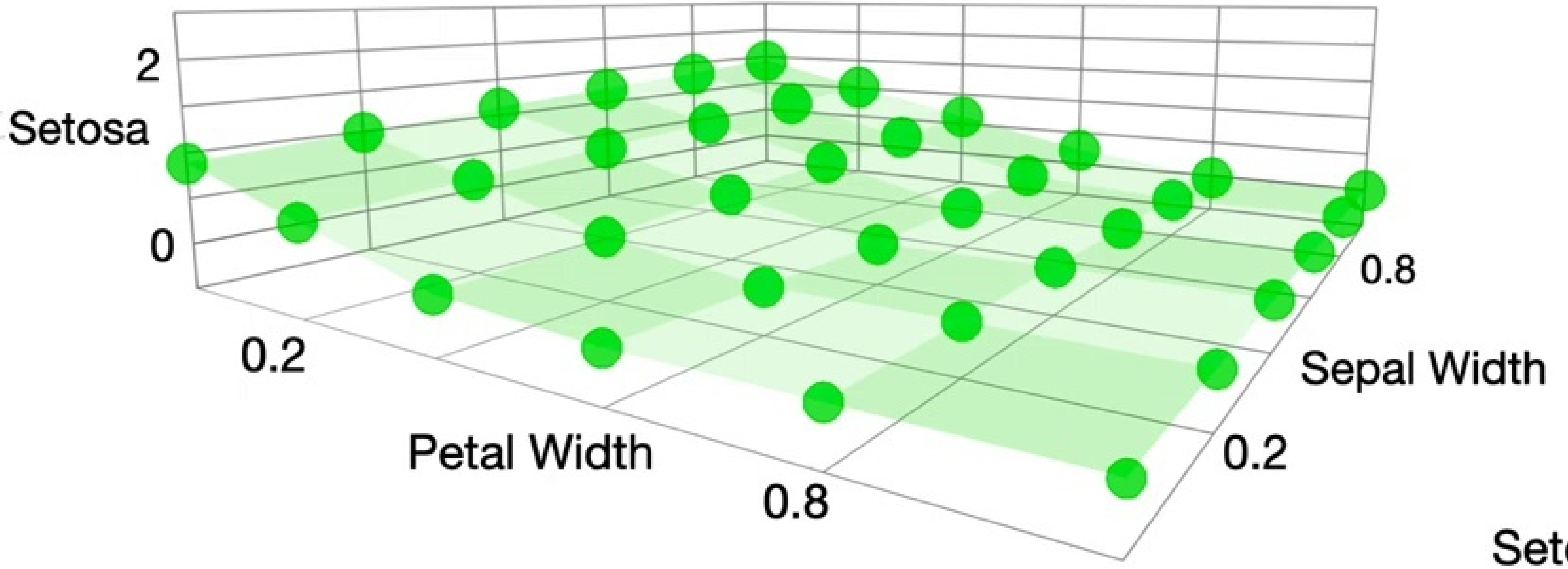
…the **Neural Network** creates curved or bent surfaces…

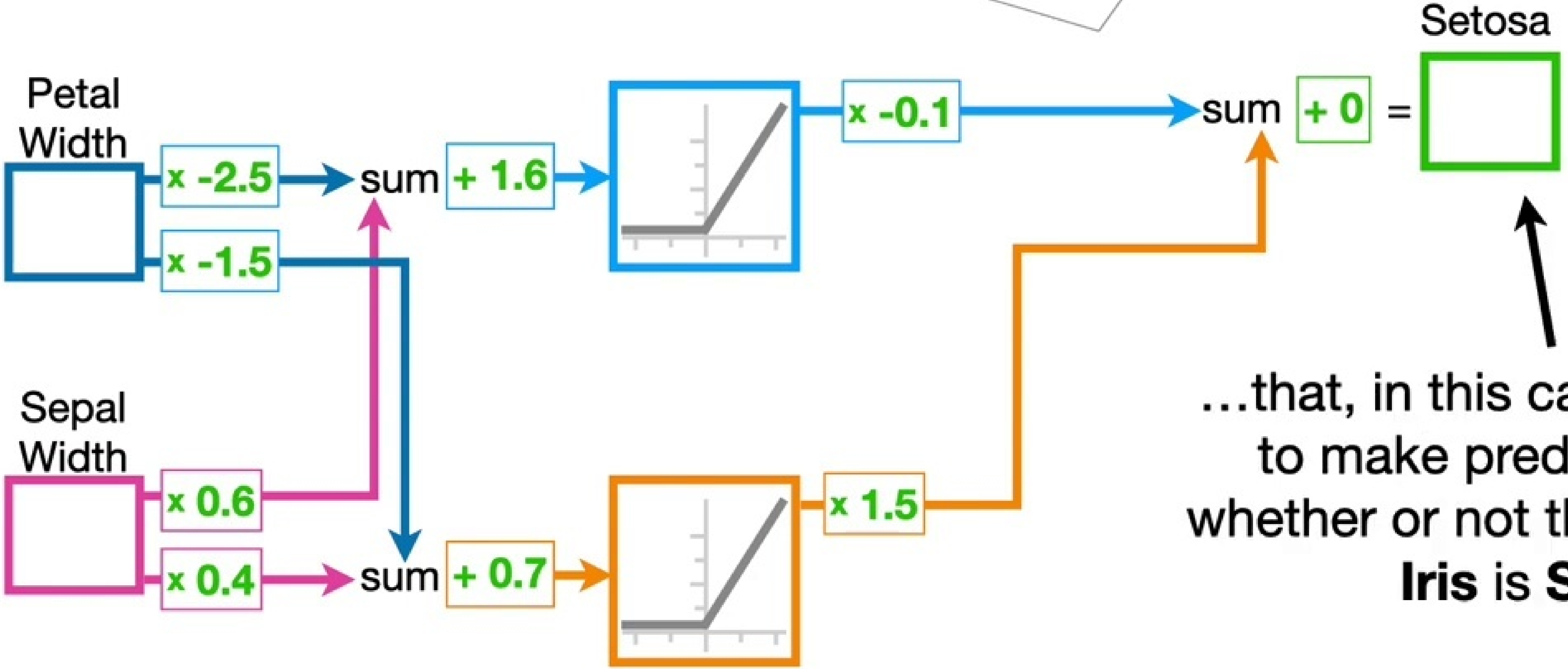

2
Setosa
0
0.2
Petal Width
0.8
0.8
Sepal Width
0.2
Petal Width
x -2.5
x -1.5
Sepal Width
x 0.6
x 0.4
sum
+ 1.6
sum
+ 0.7
x -0.1
x 1.5
sum
+ 0
=
Setosa
…the **Neural Network** creates curved or bent surfaces…



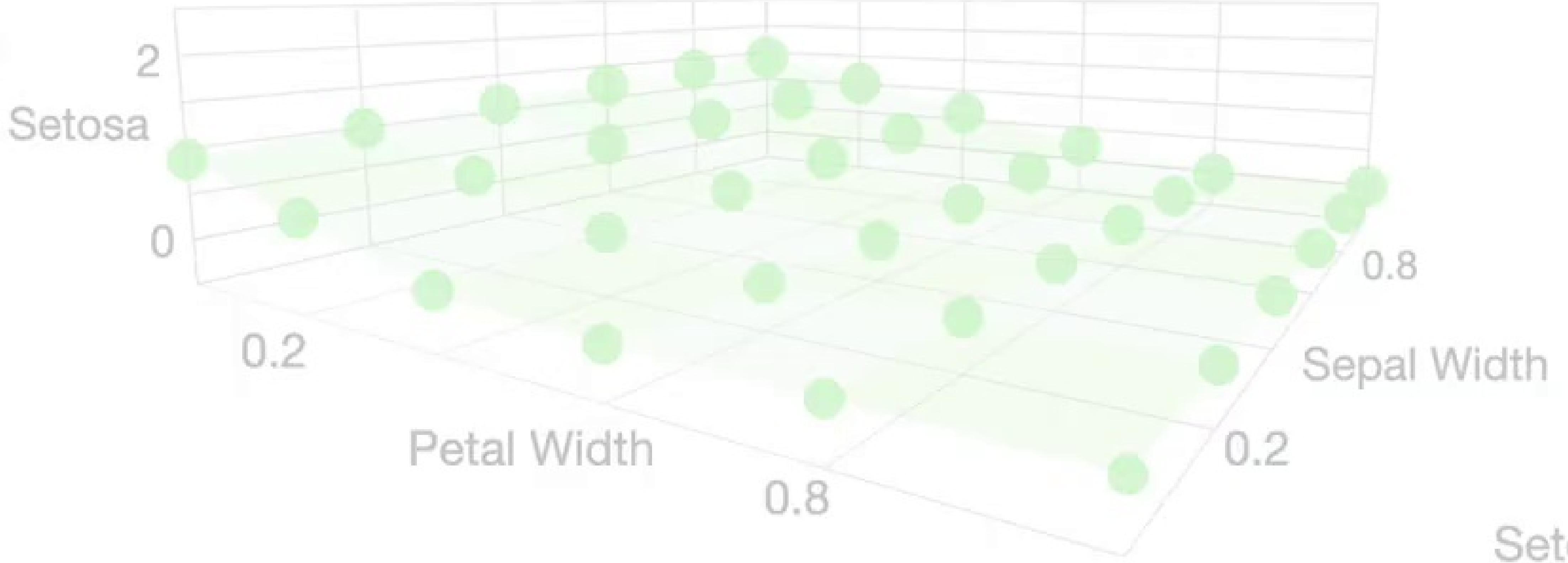


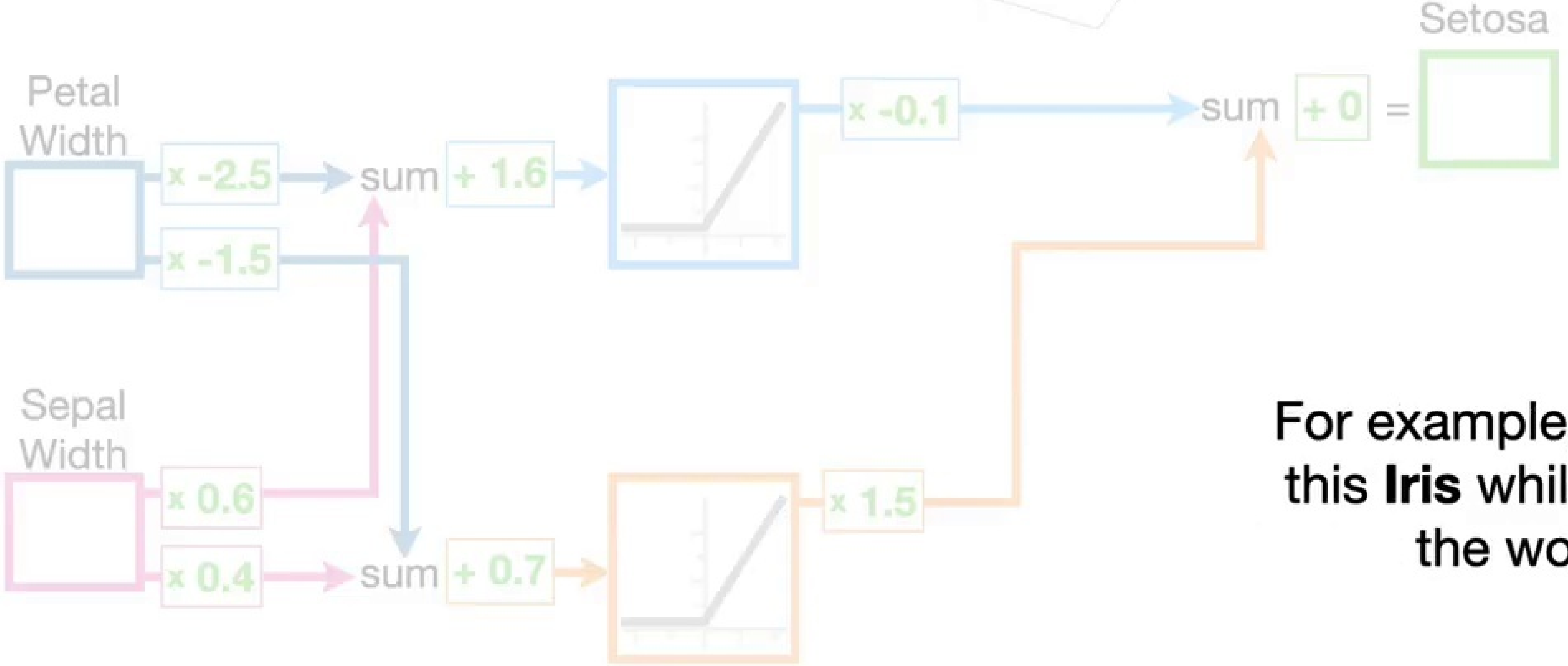
…that are added together to make a new **crinkled surface**…

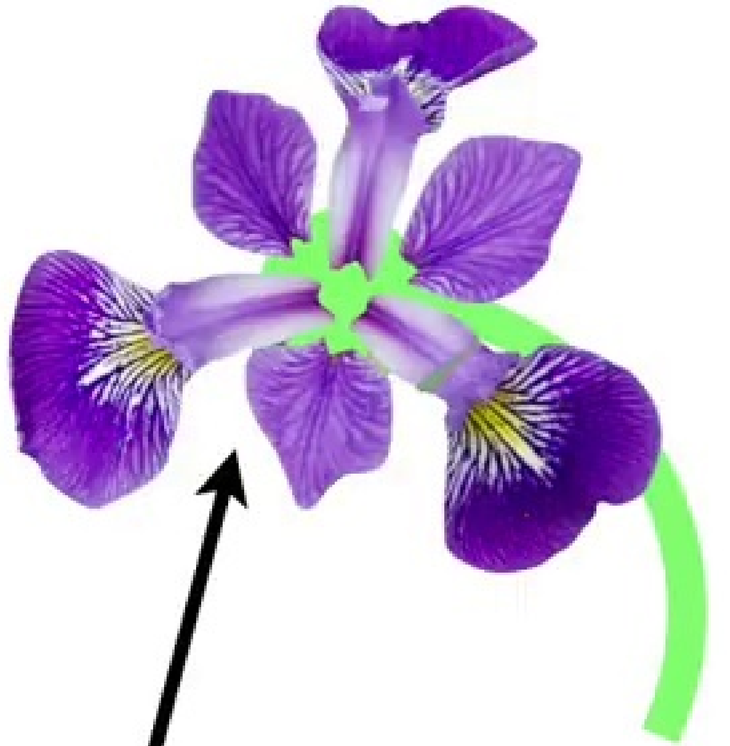




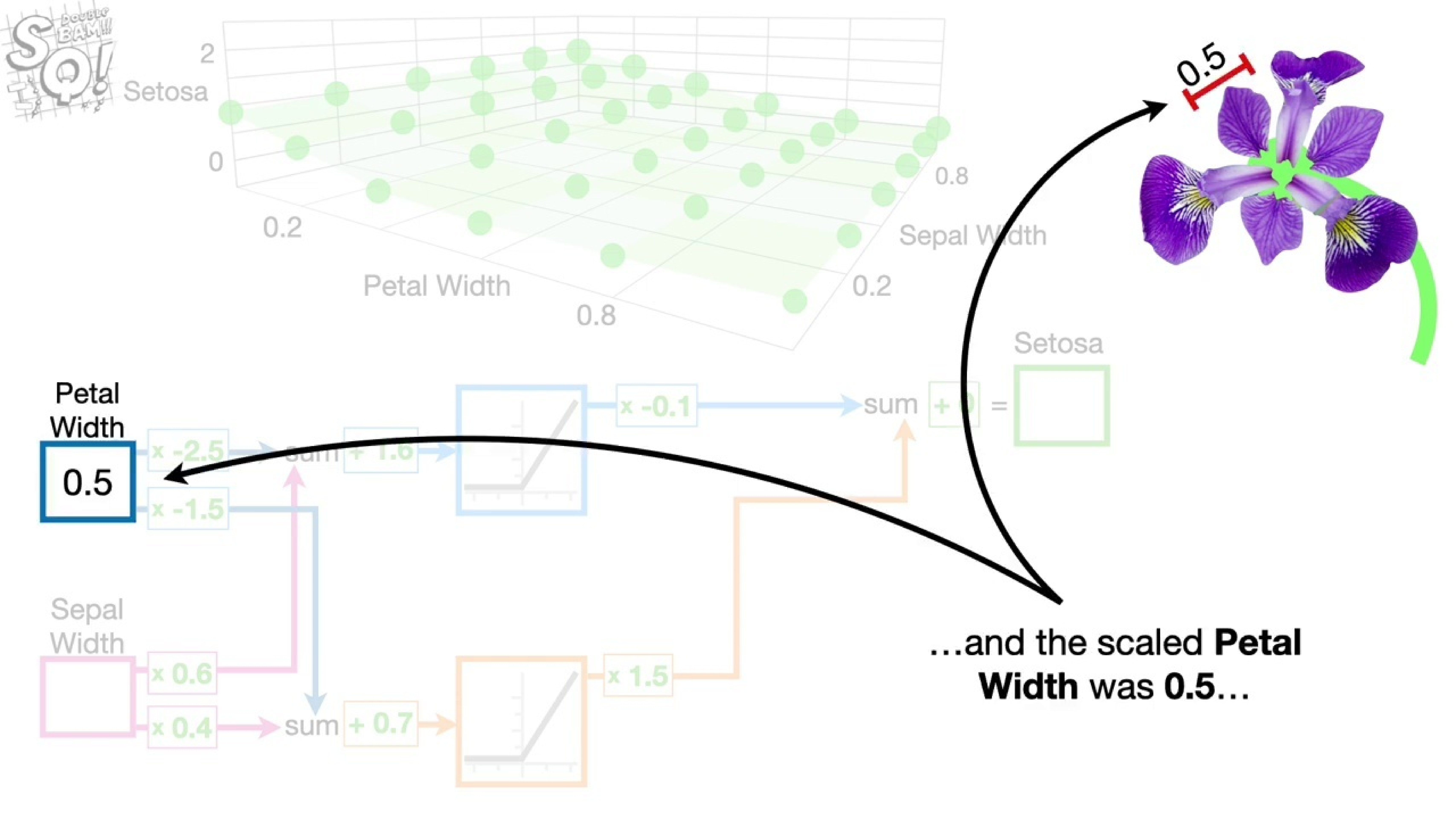
…that, in this case, we can use to make predictions about whether or not the species of an **Iris** is **Setosa**.



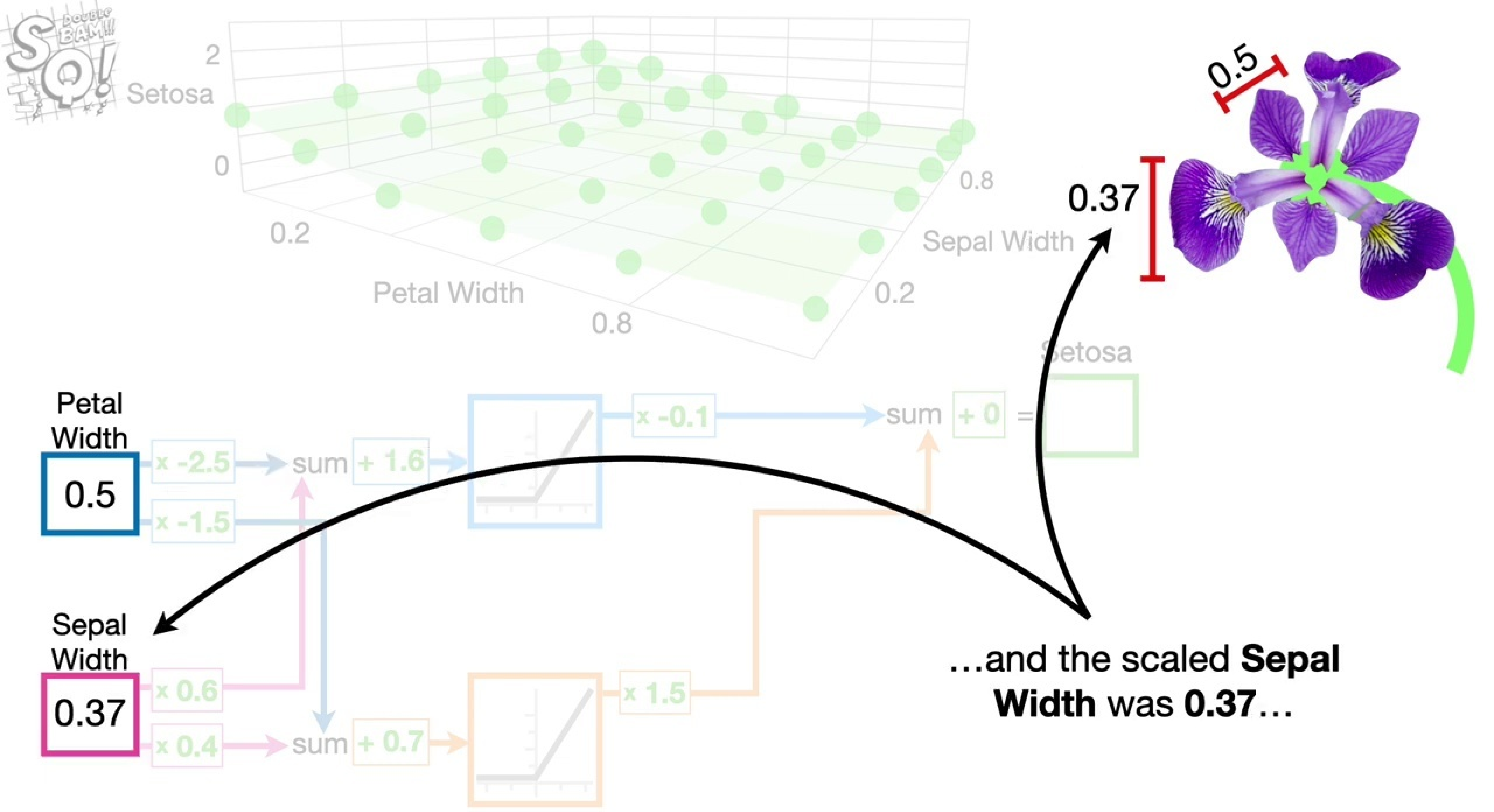




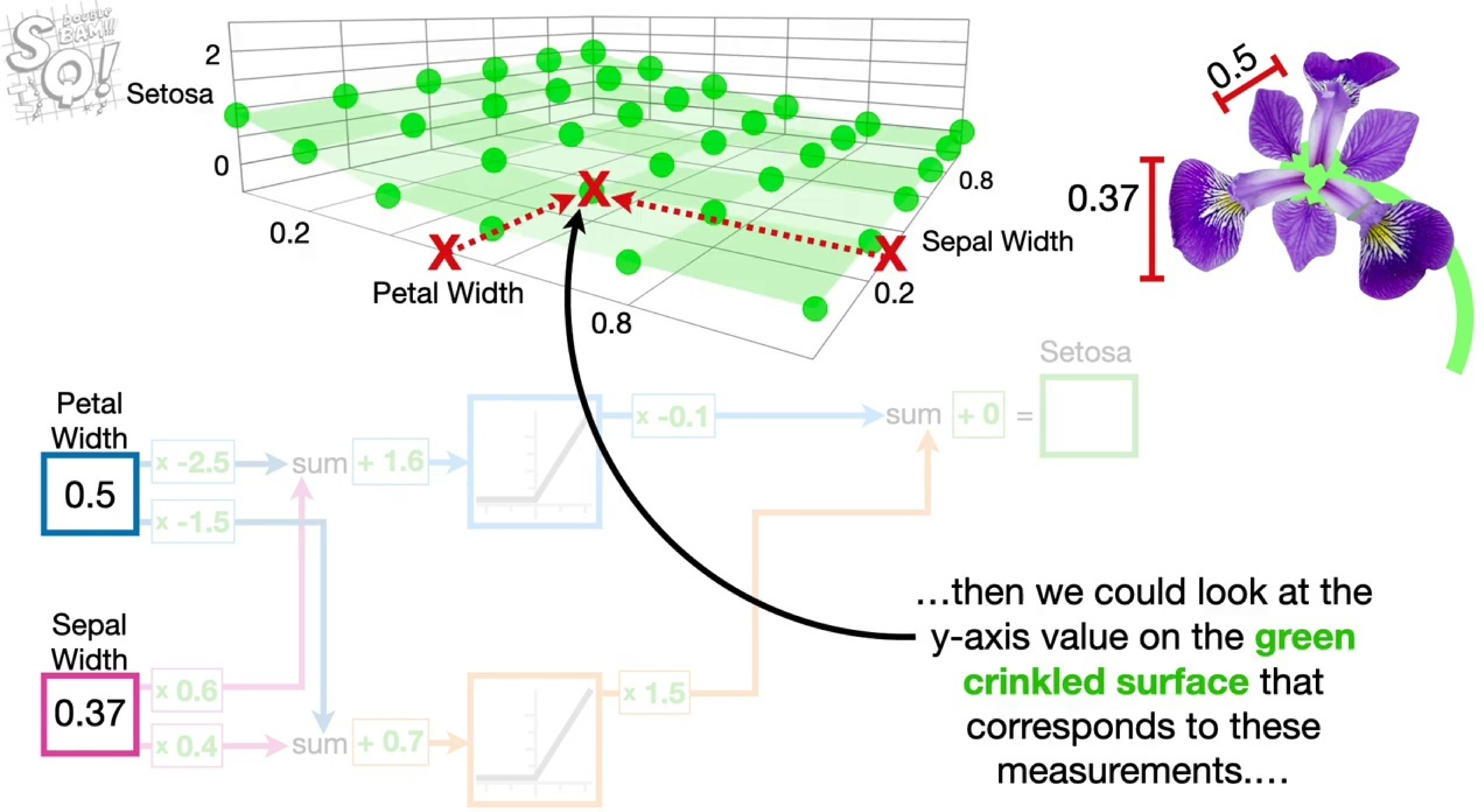
For example, if we found this **Iris** while walking in the woods…

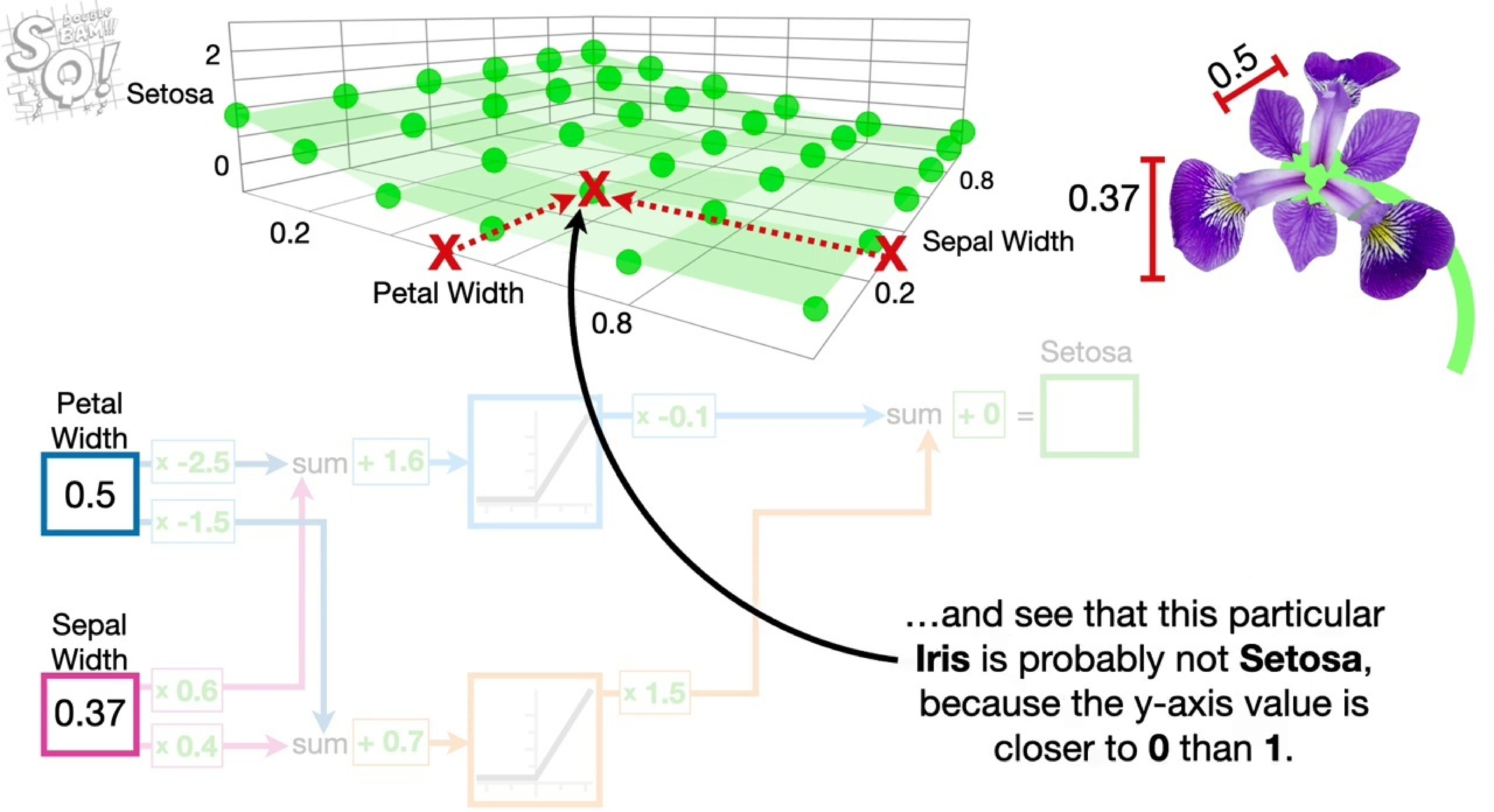


Setosa
2
0
0.2
Petal Width
0.8
0.8
Sepal Width
0.2
0.5
Setosa
Petal Width
0.5
x -2.5
x -1.5
sum
+ 1.6
x -0.1
sum
=
Sepal Width
x 0.6
x 0.4
sum
+ 0.7
x 1.5
…and the scaled **Petal Width** was **0.5**…

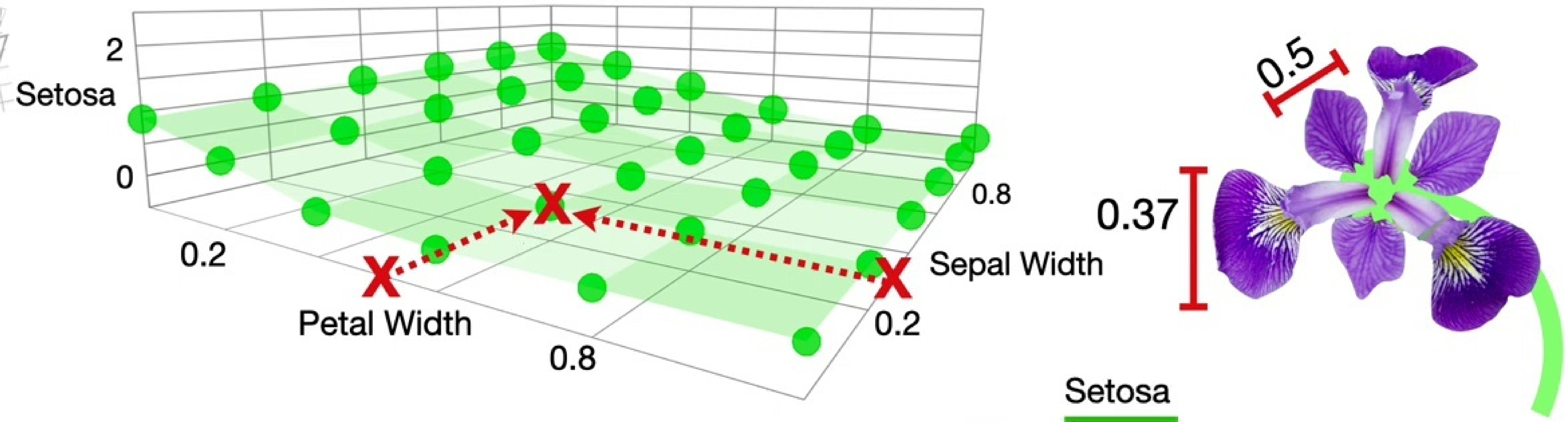


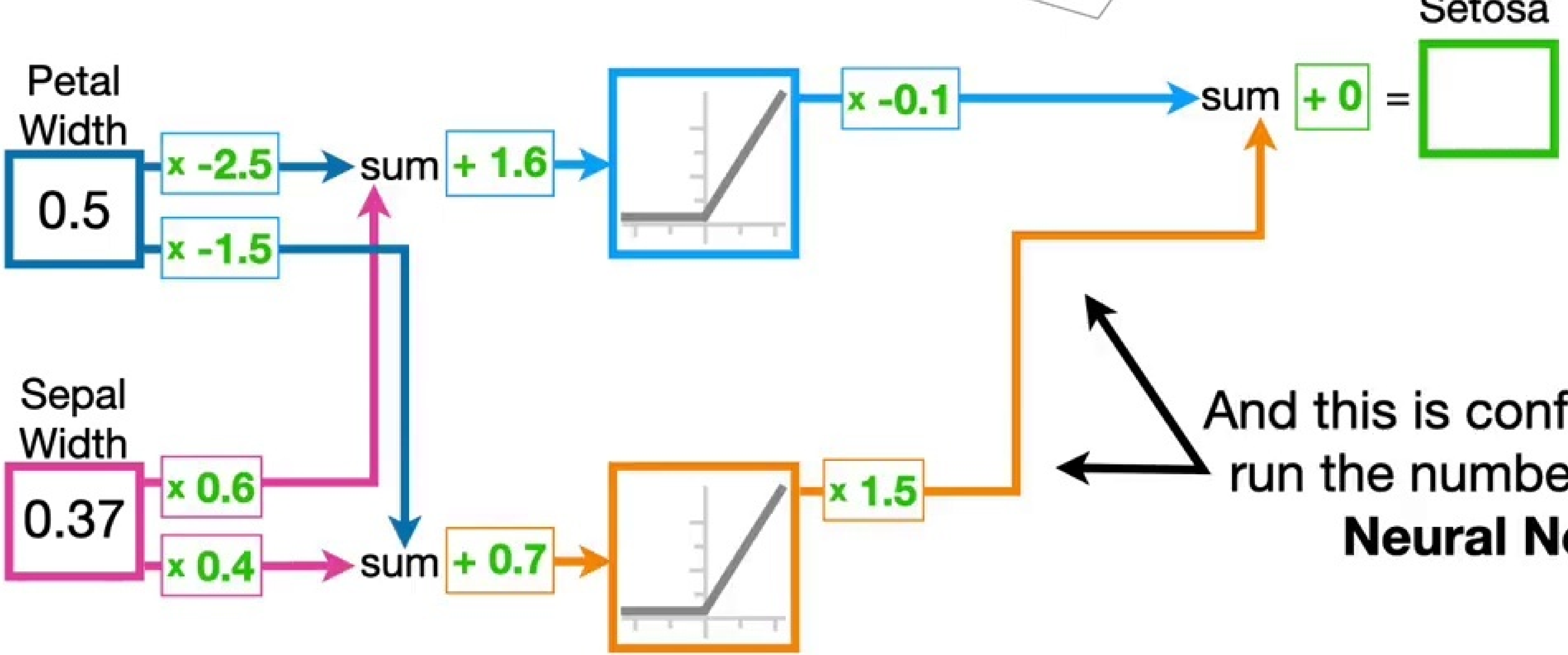
…and the scaled **Sepal Width** was **0.37**…


2
Setosa
0
0.2
Petal Width
0.8
0.8
Sepal Width
0.2
0.5
0.37
Setosa
Petal Width
0.5
x -2.5
sum
+ 1.6
x -1.5
x -0.1
sum
+ 0
=
Sepal Width
0.37
x 0.6
x 0.4
sum
+ 0.7
x 1.5
…then we could look at the y-axis value on the green crinkled surface that corresponds to these measurements.…

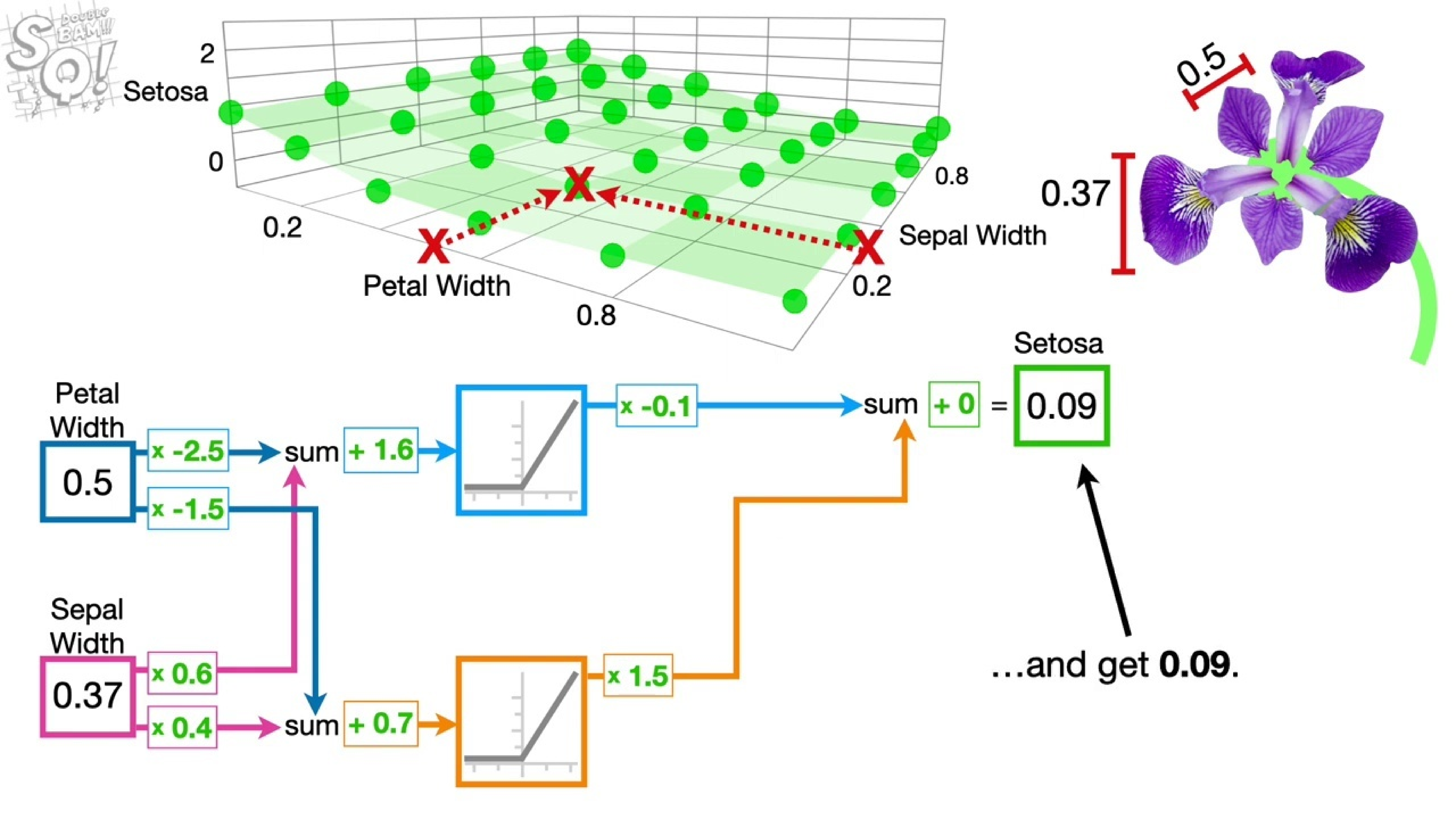


…and see that this particular **Iris** is probably not **Setosa**, because the y-axis value is closer to **0** than **1**.

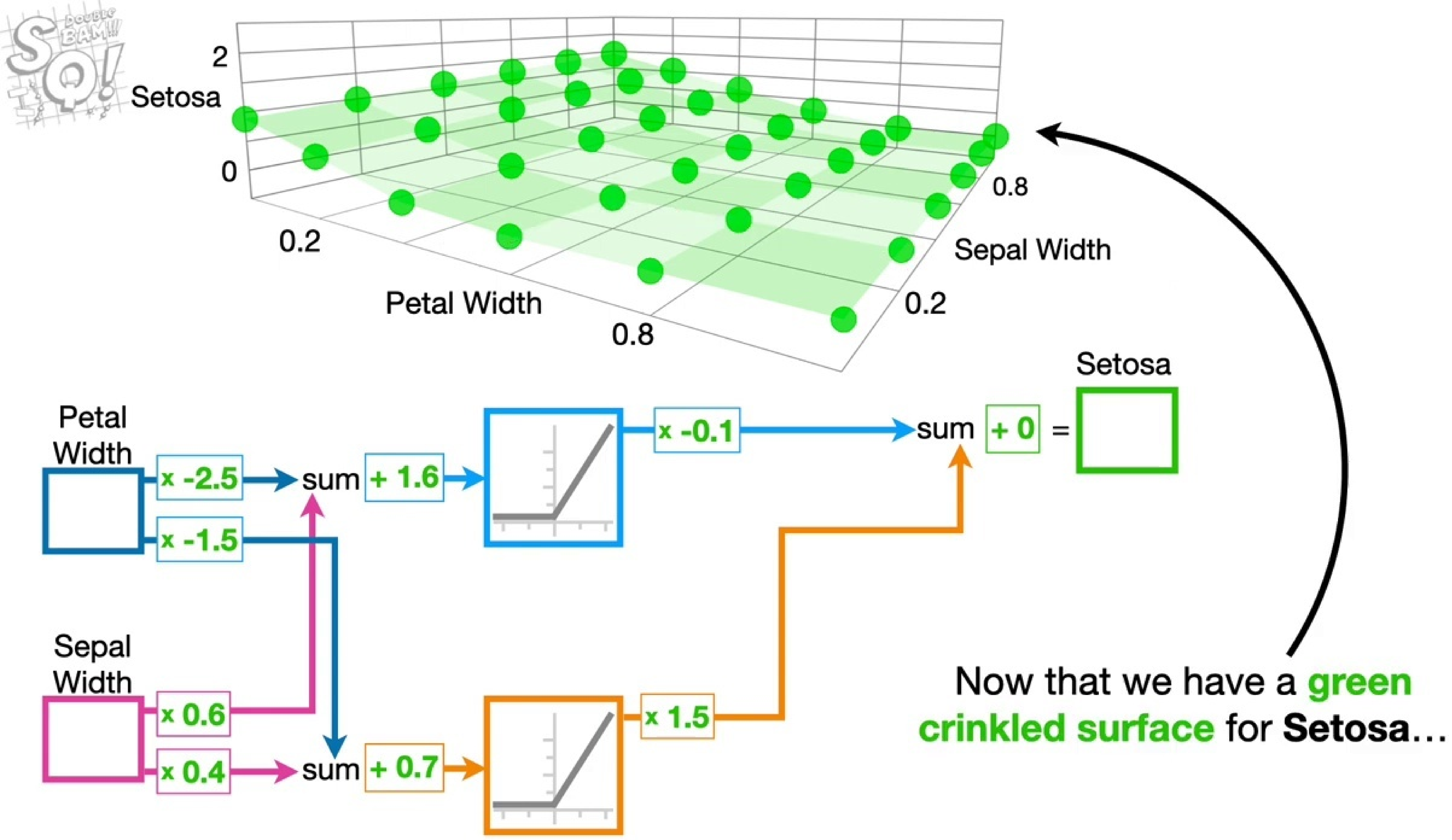




And this is confirmed when we run the numbers through the **Neural Network**…

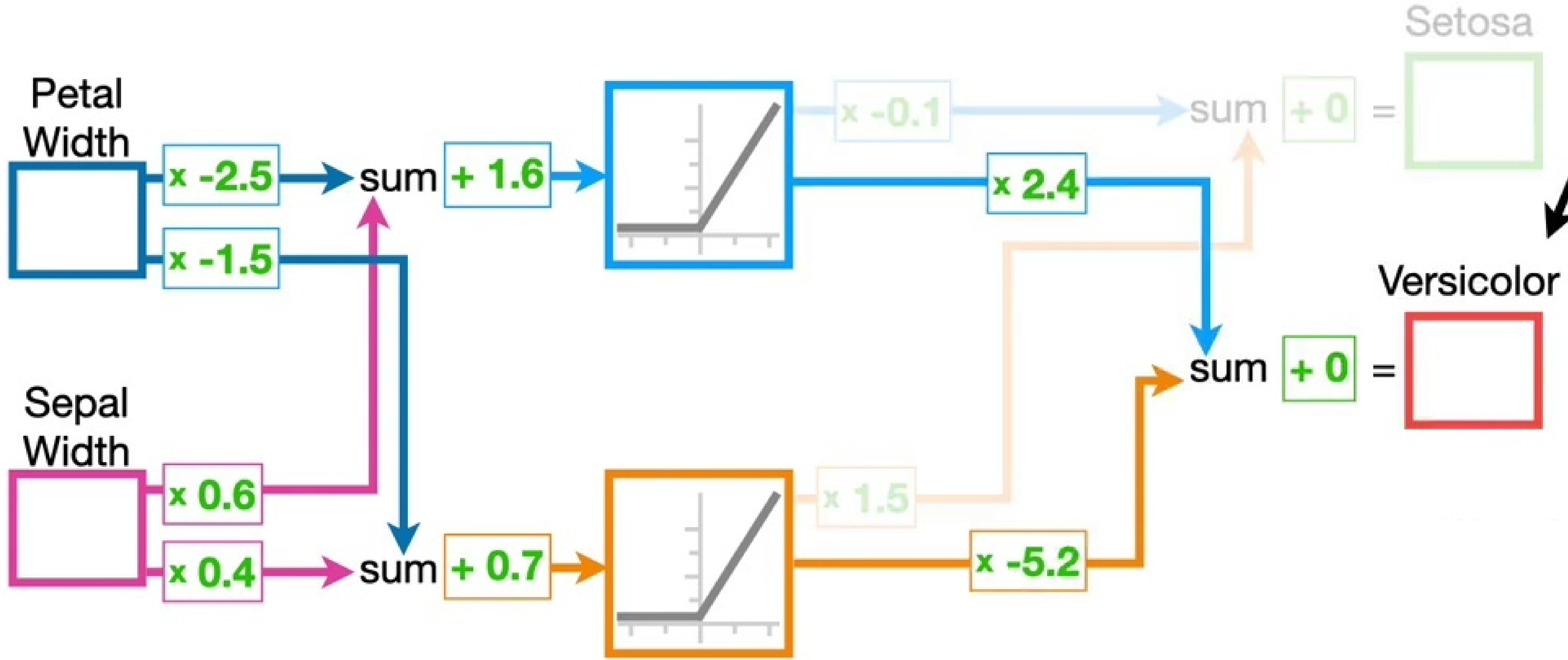


Setosa
2
0
0.2
Petal Width
0.8
0.8
Sepal Width
0.2
0.5
0.37
Petal Width
0.5
x -2.5
x -1.5
sum
+ 1.6
x -0.1
sum
+ 0
=
Setosa
0.09
Sepal Width
0.37
x 0.6
x 0.4
sum
+ 0.7
x 1.5
…and get **0.09**.

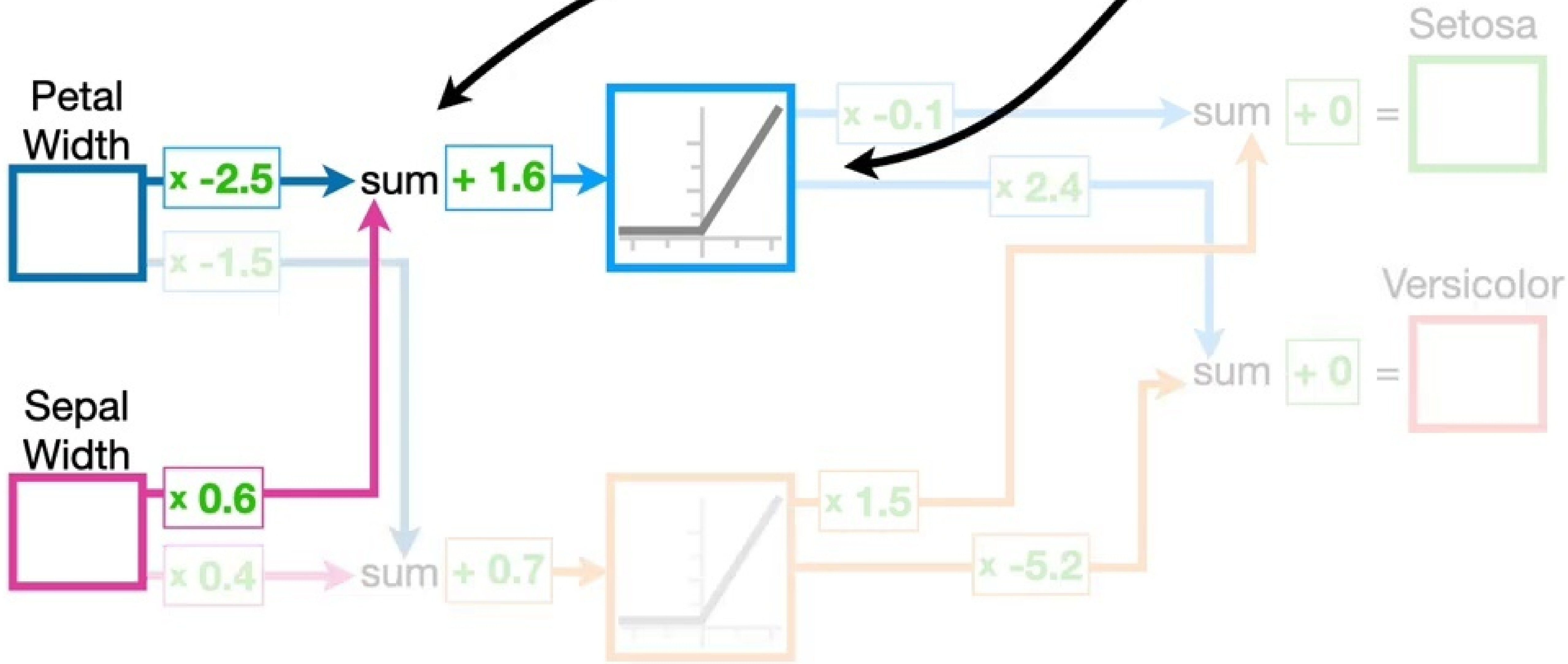


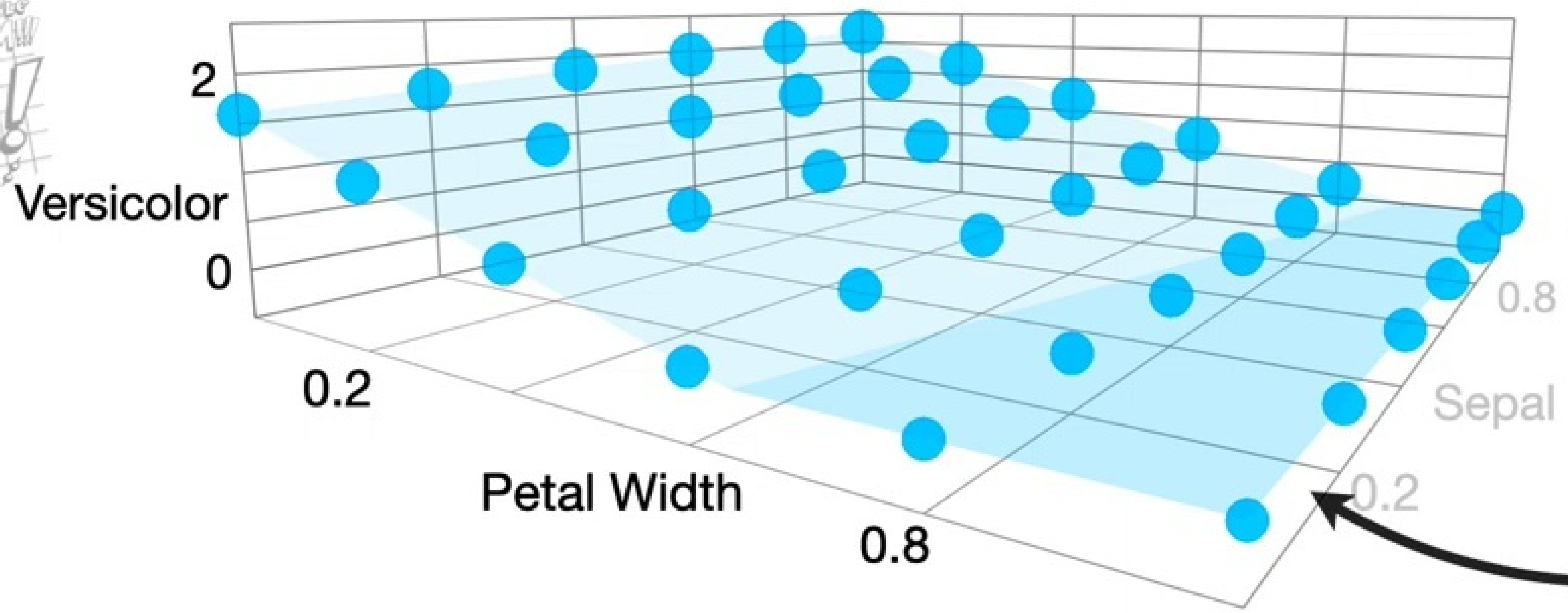
Now that we have a **green crinkled surface** for **Setosa**…

…let's determine the output for the second species, **Versicolor**.

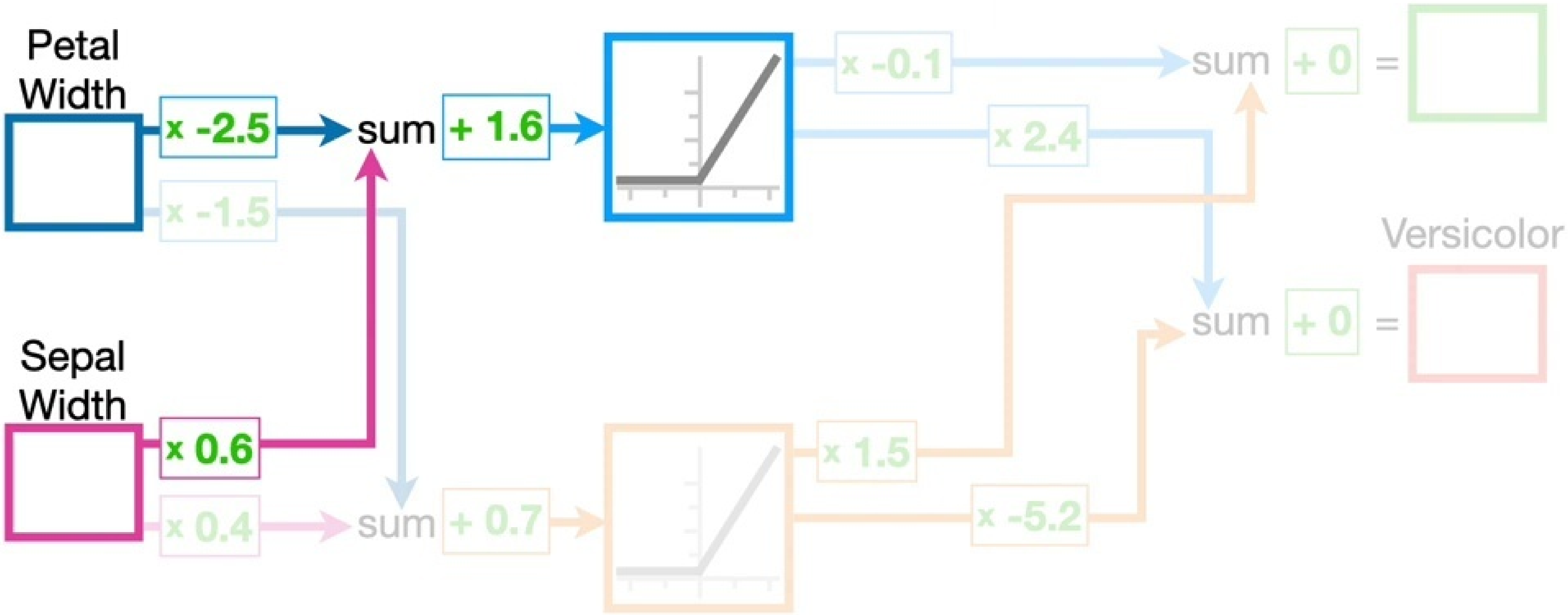


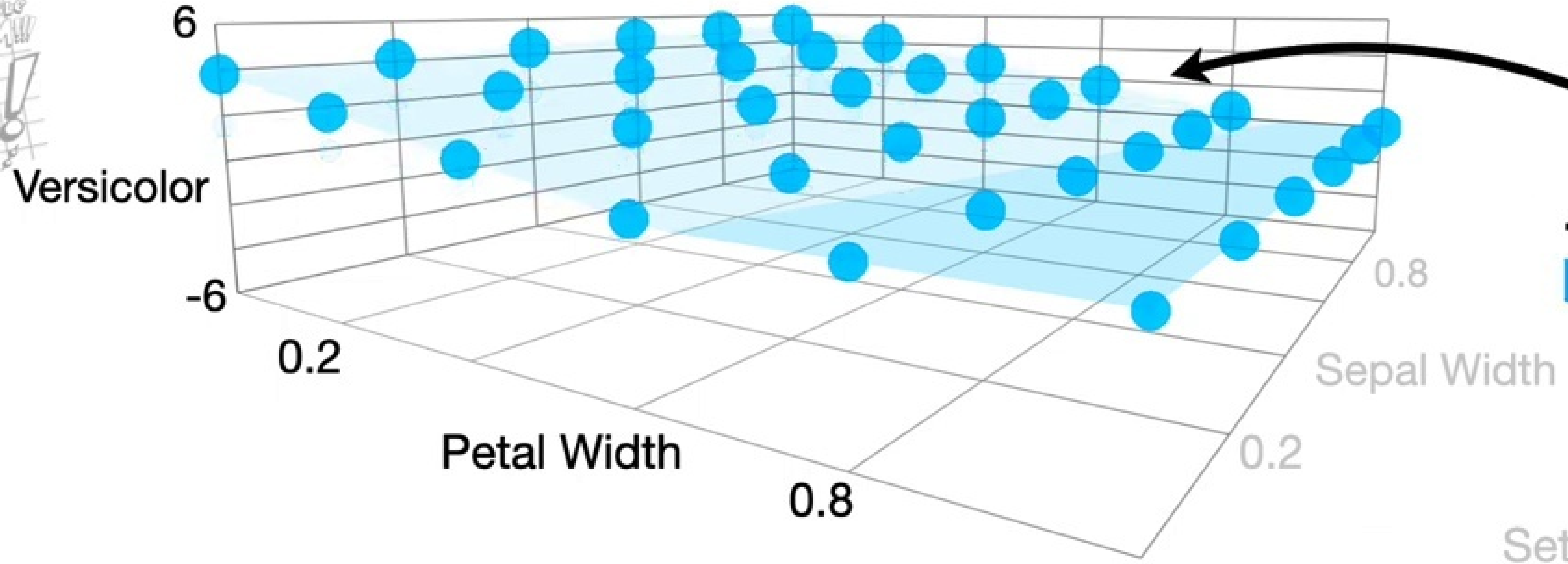
Just like before, we'll start with the connections to the top node in the **Hidden Layer**.



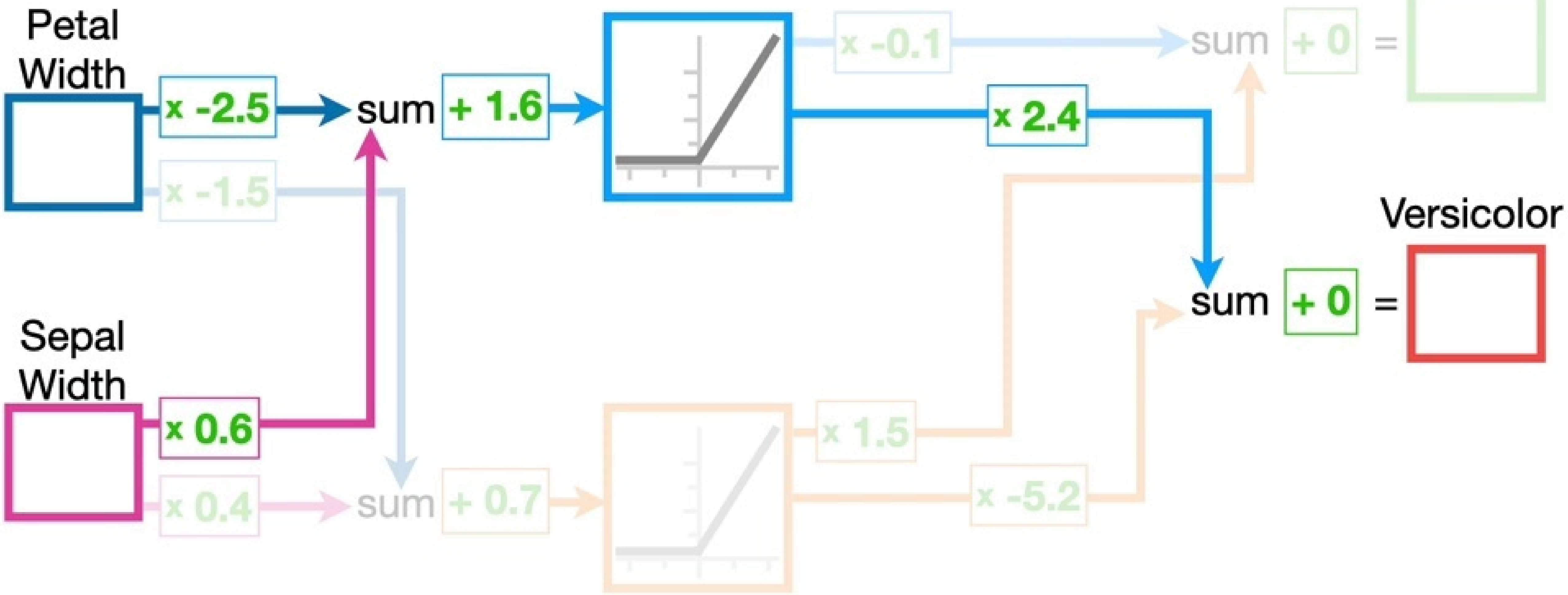


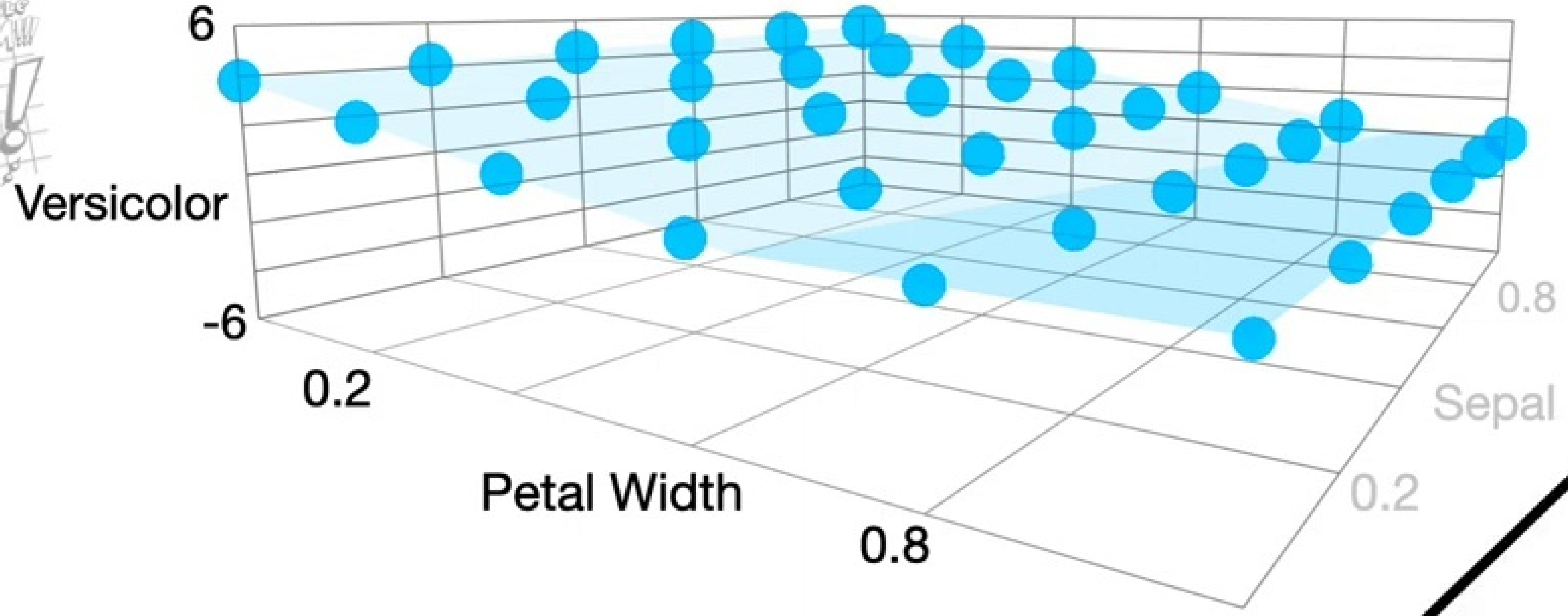
And because the **Weights** and **Biases** are the same as before, we start out with the same **blue bent surface**.



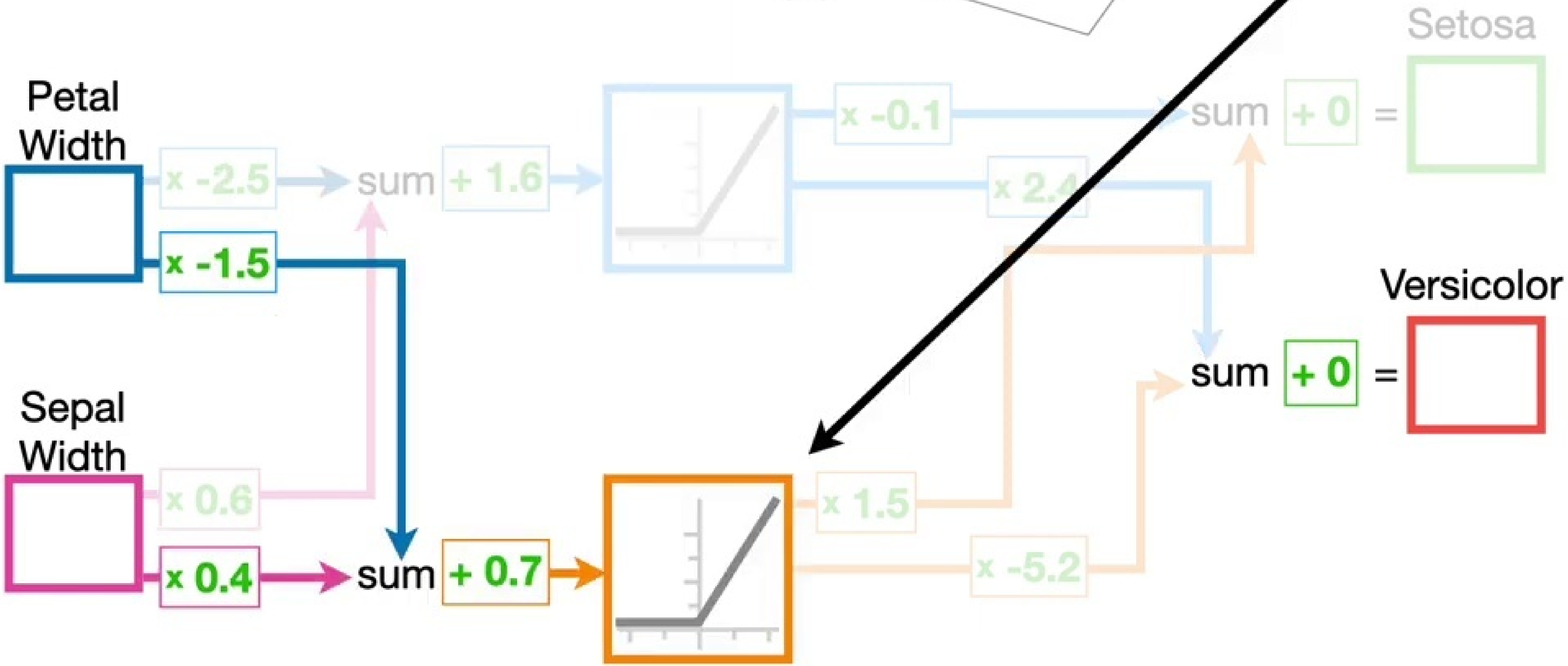


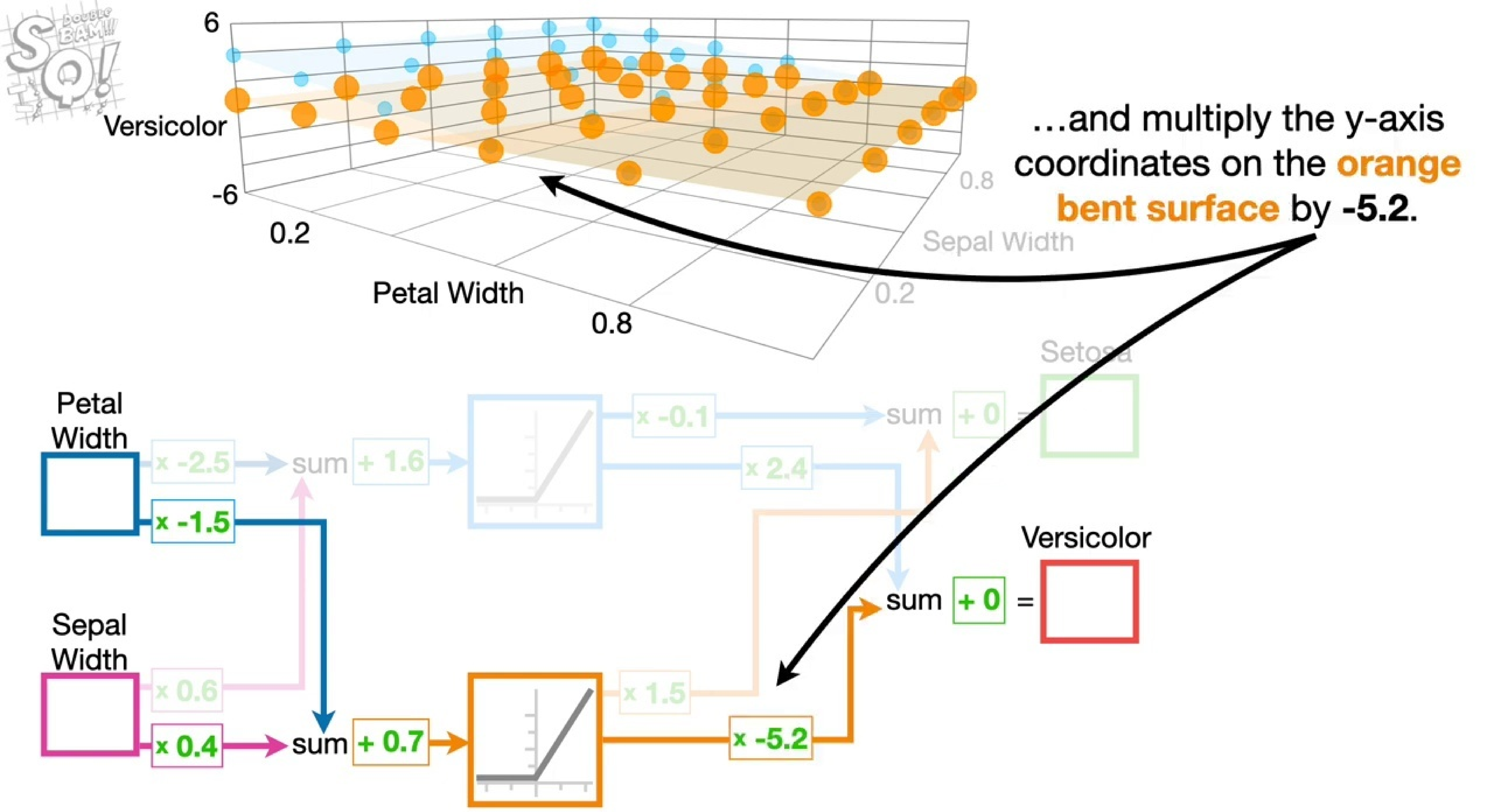
…gives us this final **blue bent surface**.



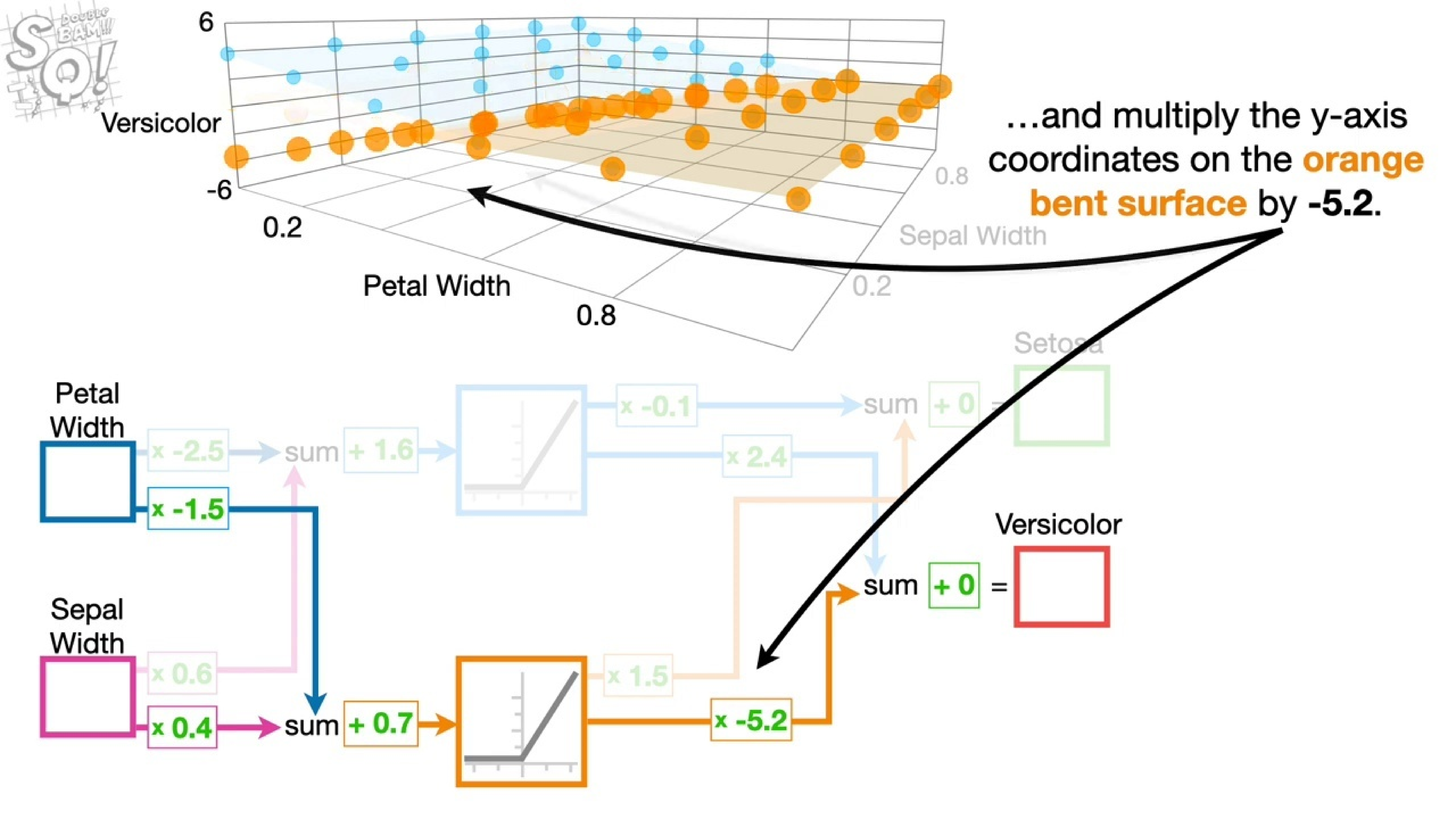


Now we create the **orange bent surface** from the bottom node in the **Hidden Layer**…

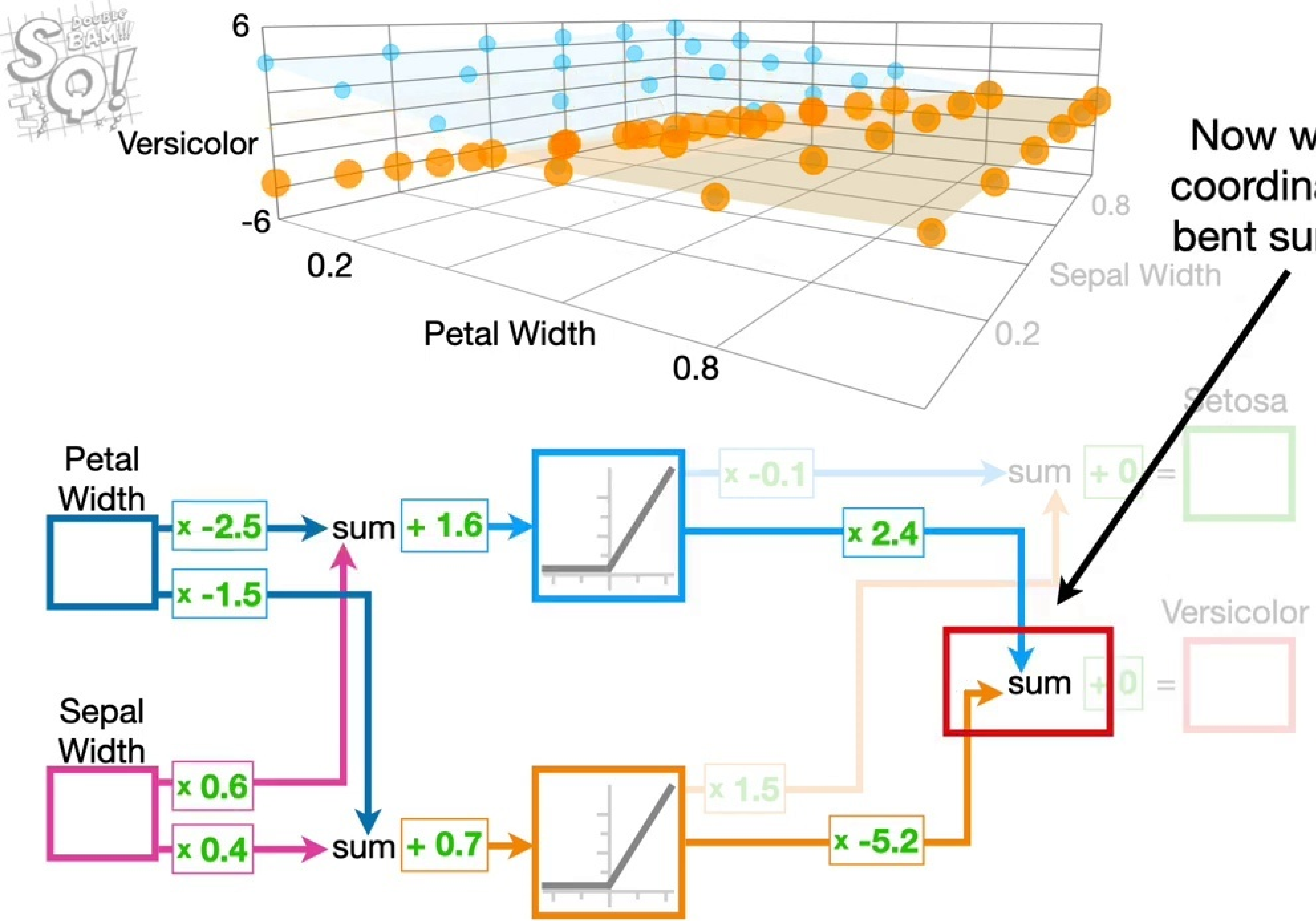




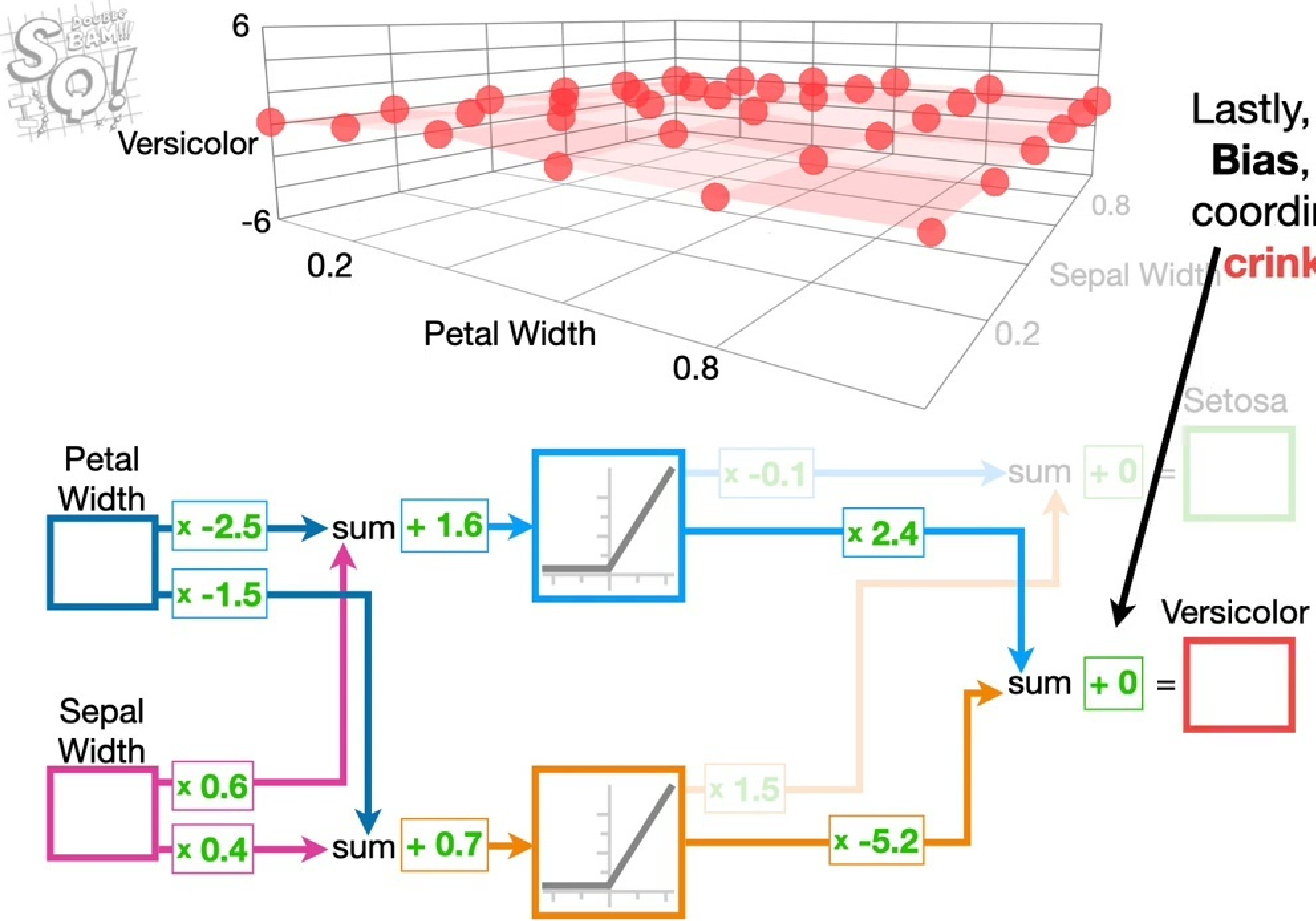
6
-6
Versicolor
0.2
Petal Width
0.8
0.8
Sepal Width
0.2
…and multiply the y-axis coordinates on the orange bent surface by -5.2.
Setosa
Petal Width
Sepal Width
x -2.5
x -1.5
x 0.6
x 0.4
sum
+ 1.6
sum
+ 0.7
x -0.1
x 2.4
x 1.5
x -5.2
sum
+ 0
=
sum
+ 0
=
Versicolor



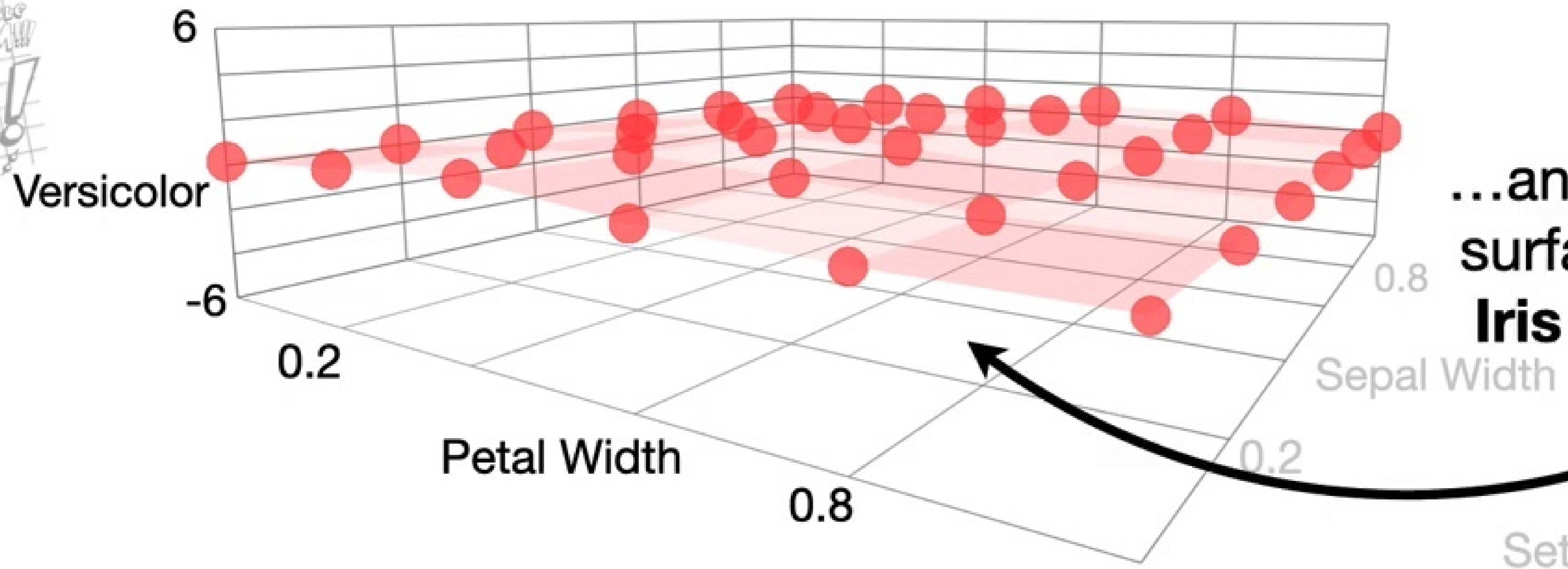
…and multiply the y-axis coordinates on the **orange bent surface** by **-5.2.**
6
Versicolor
-6
0.2
Petal Width
0.8
0.8
Sepal Width
0.2
Setosa
Petal Width
x -2.5
sum
+ 1.6
x -0.1
sum
+ 0
=
x 2.4
x -1.5
Versicolor
sum
+ 0
=
Sepal Width
x 0.6
x 1.5
x 0.4
sum
+ 0.7
x -5.2



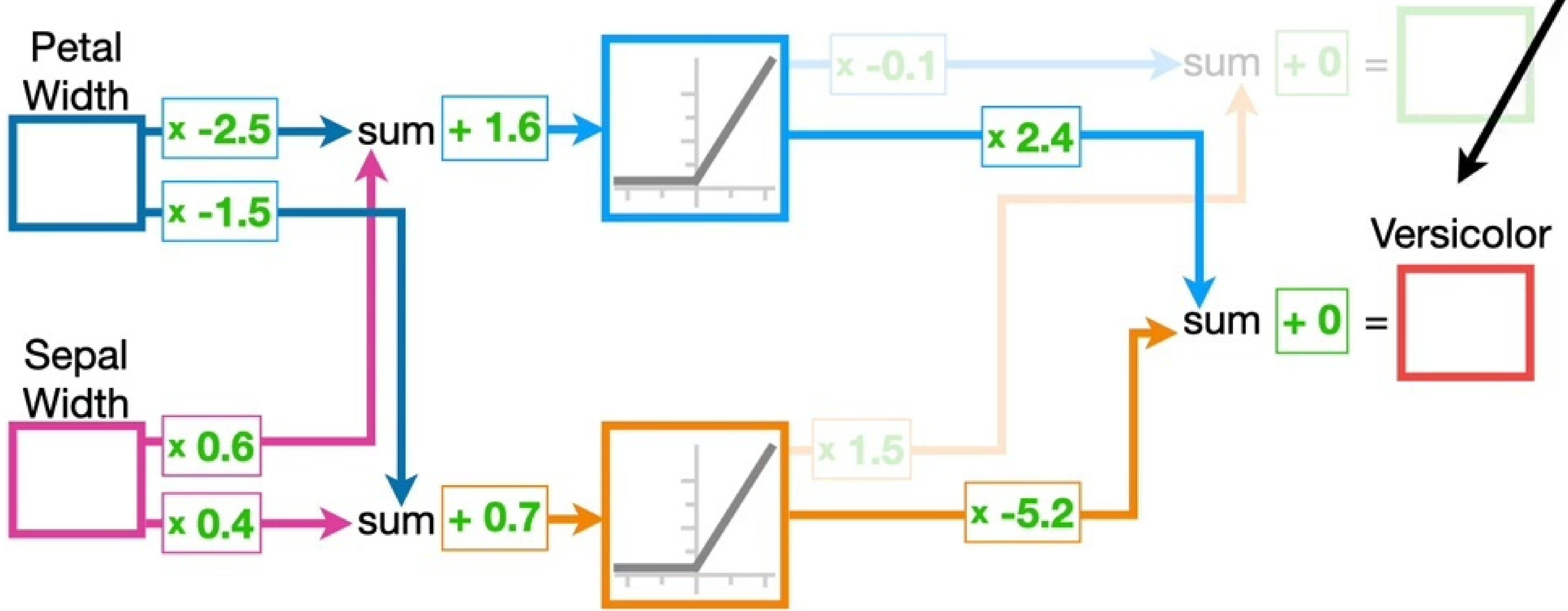
Now we add the y-axis coordinates from the two bent surfaces together…

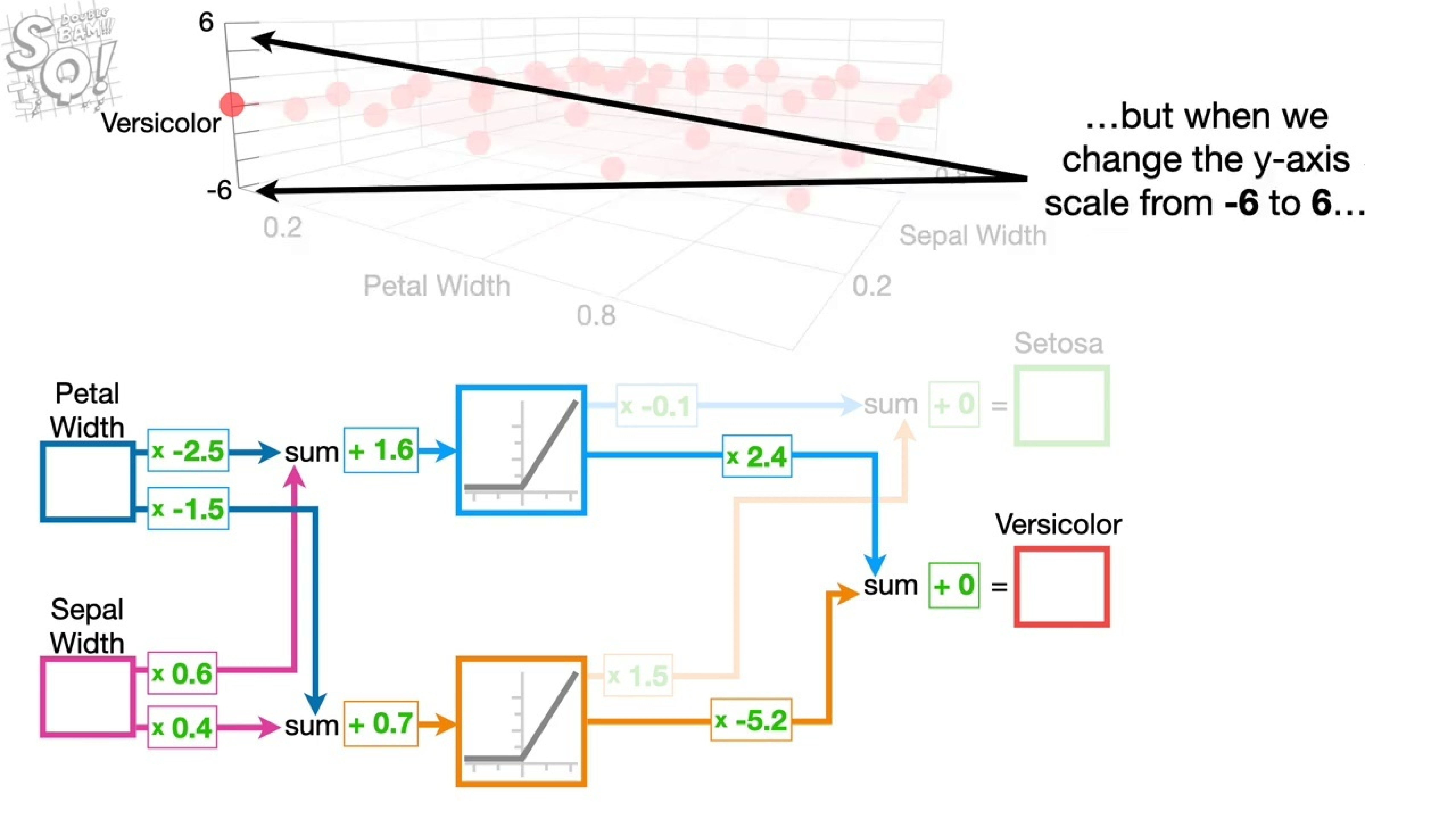


Lastly, we add the final **Bias, 0,** to the y-axis coordinates on the **red crinkled surface**…

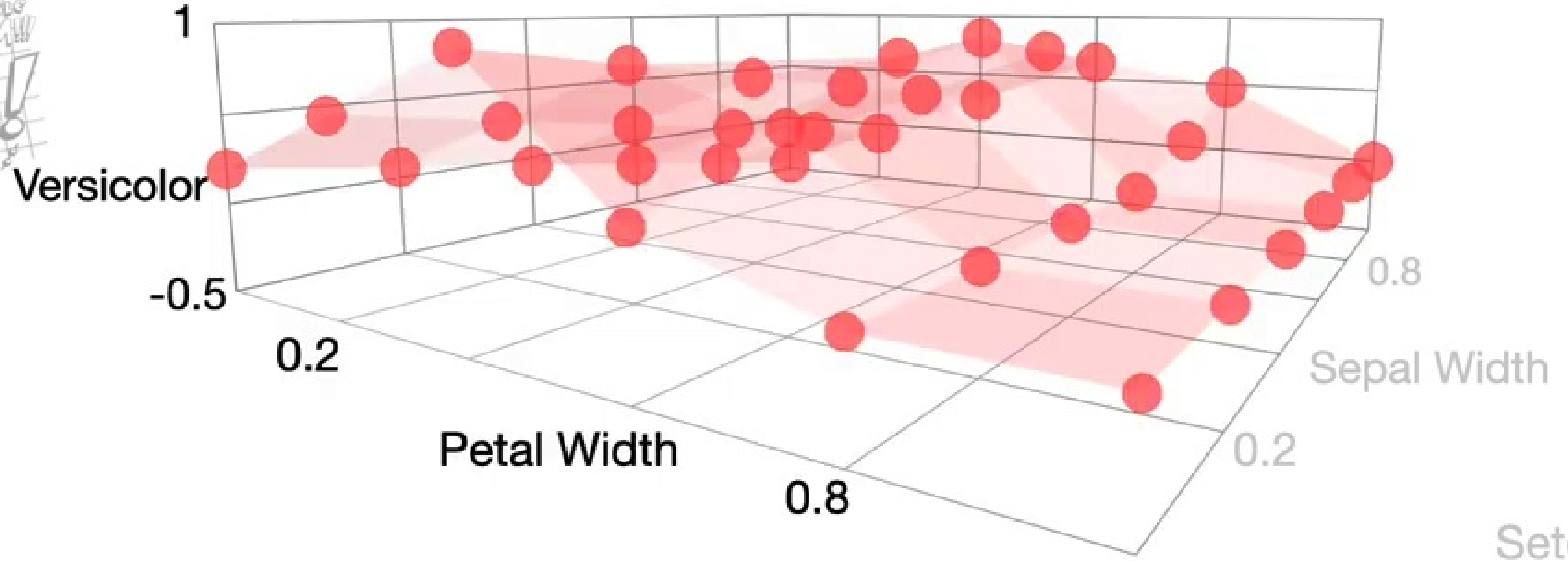


…and that gives us the final surface for predicting if the **Iris** species is **Versicolor.**



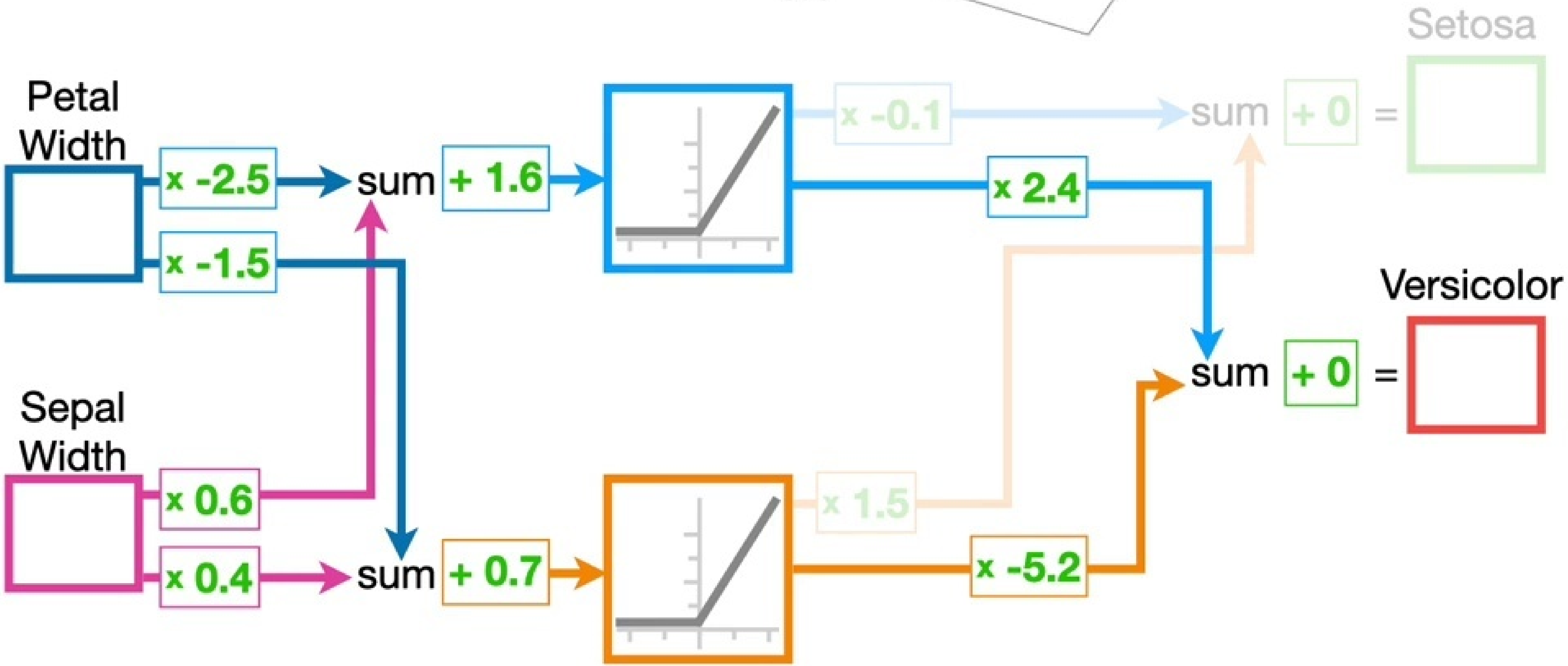


6
Versicolor
-6
0.2
Petal Width
0.8
Sepal Width
0.2
…but when we change the y-axis scale from **-6** to **6**…
Petal Width
Sepal Width
x -2.5
x -1.5
x 0.6
x 0.4
sum
+ 1.6
sum
+ 0.7
x -0.1
x 2.4
x 1.5
x -5.2
sum
+ 0
=
Setosa
sum
+ 0
=
Versicolor

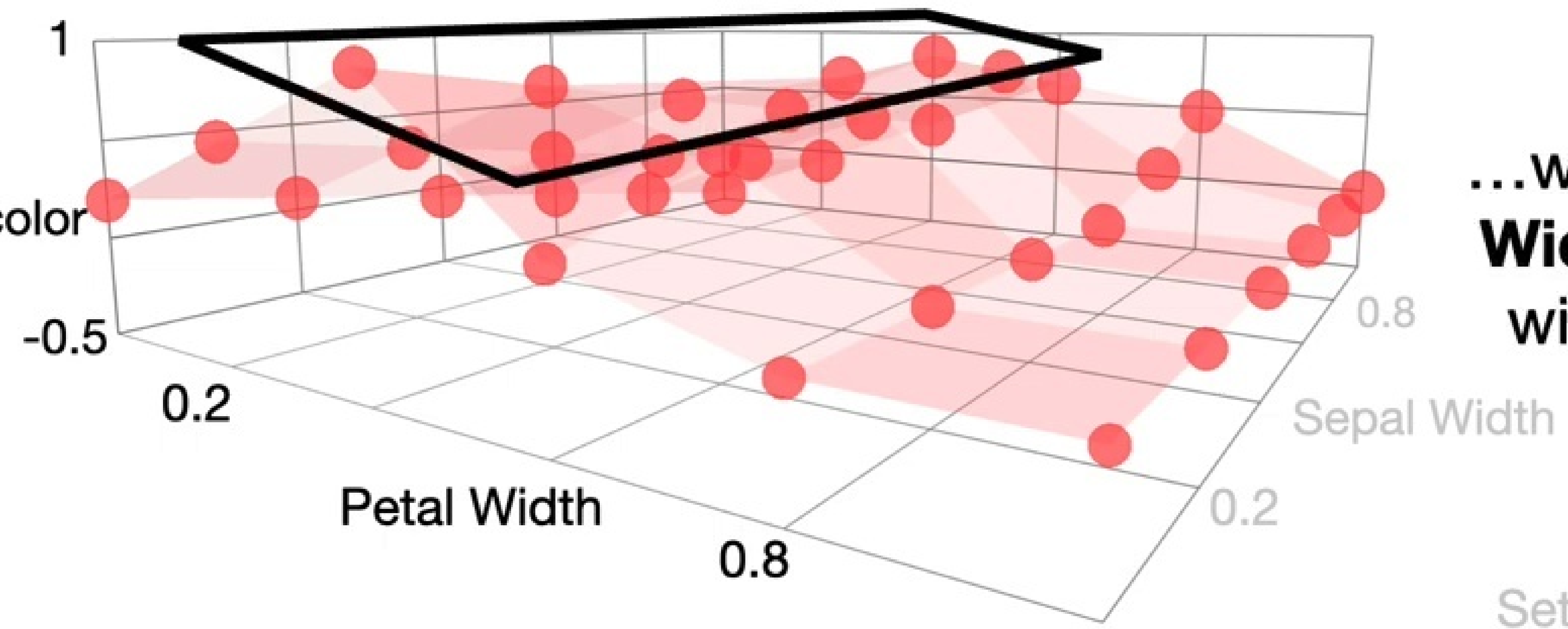


1
Versicolor
-0.5
0.2
Petal Width
0.8
0.8
Sepal Width
0.2

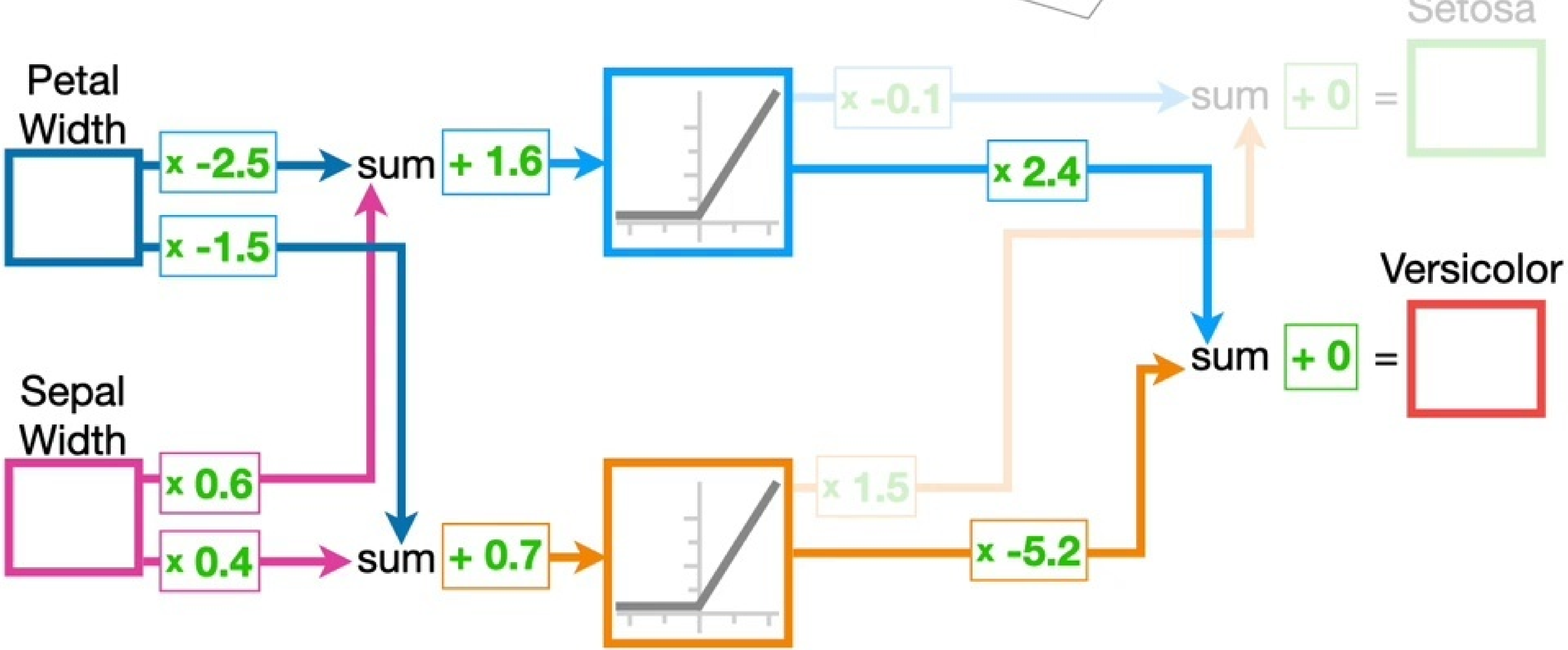
…to **-0.5** to **1**…



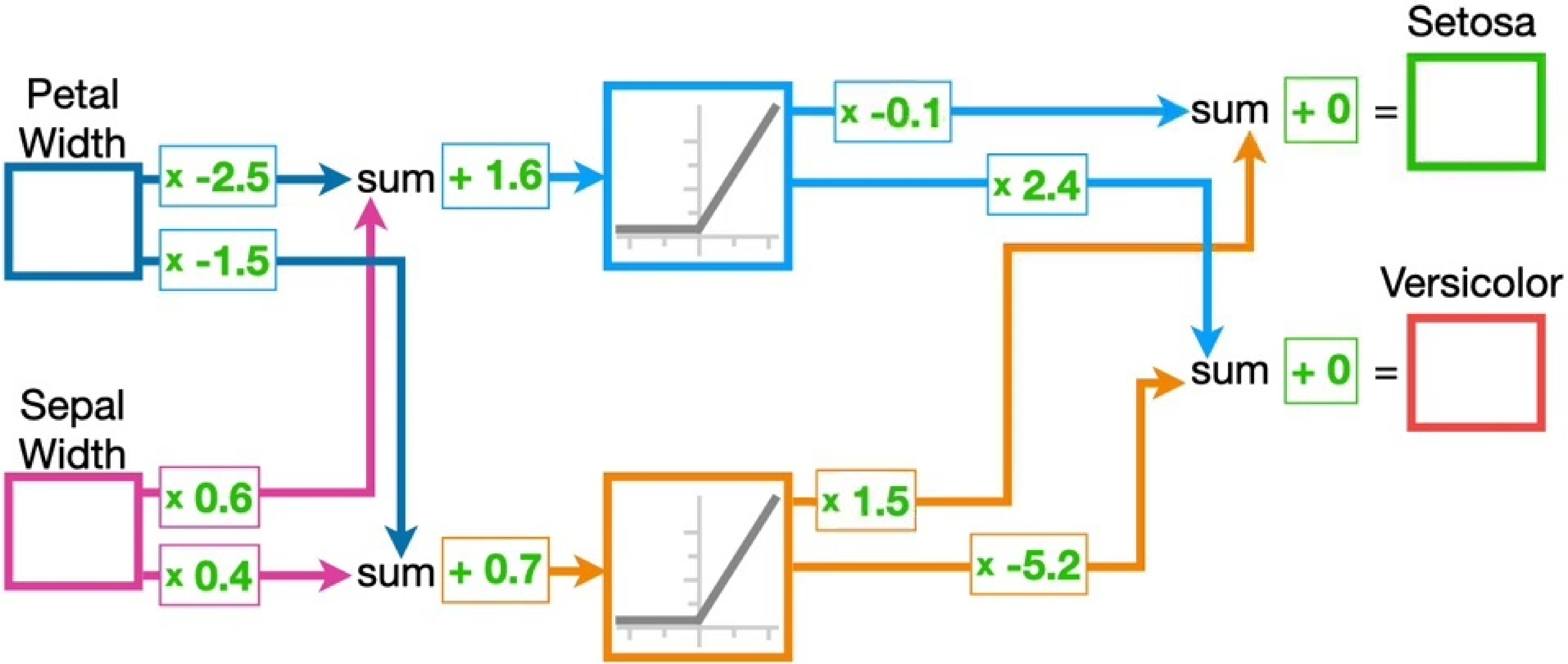
Petal Width
x -2.5
x -1.5
sum
+ 1.6
x -0.1
sum
+ 0
=
Setosa
x 2.4
Versicolor
sum
+ 0
=
Sepal Width
x 0.6
x 0.4
sum
+ 0.7
x 1.5
x -5.2



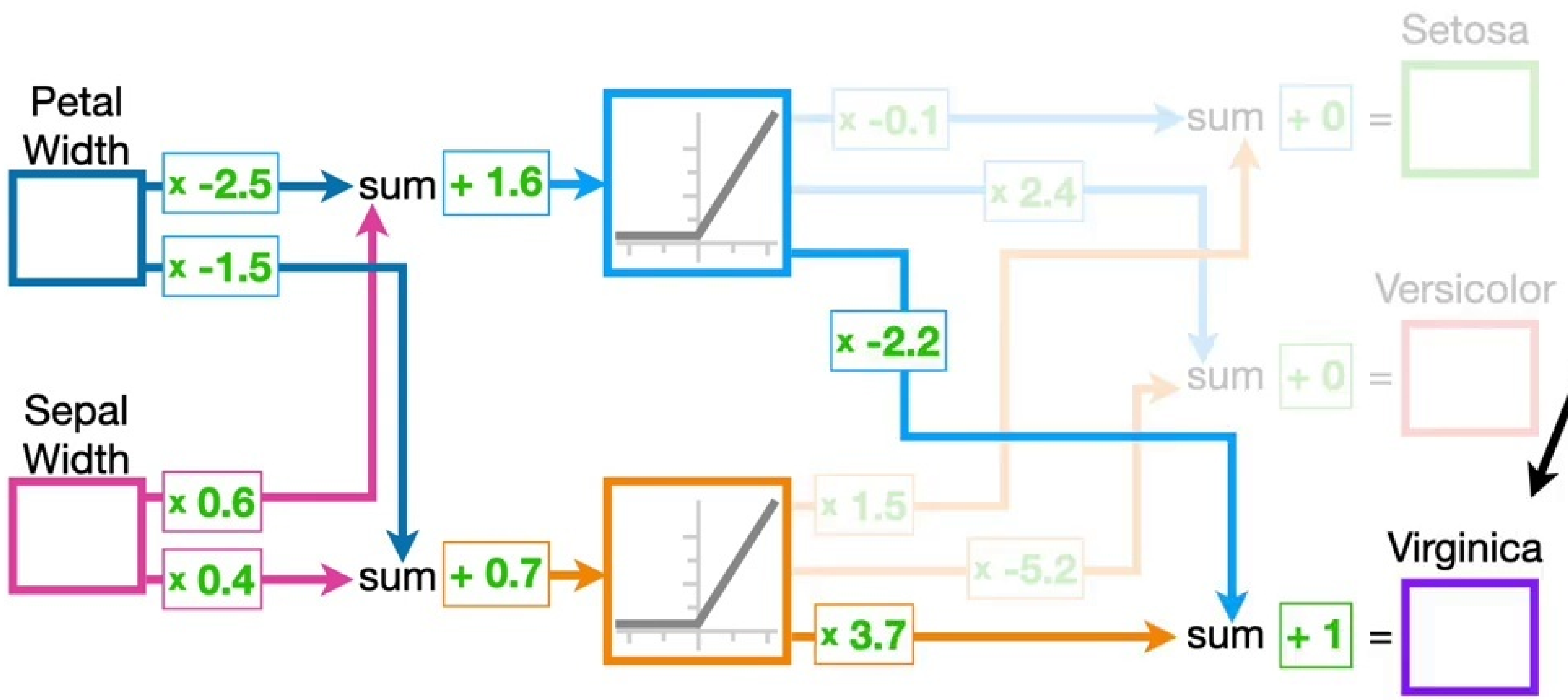
…we see that when **Petal Width** is close to **0.4**, we will get a high value for **Versicolor**.

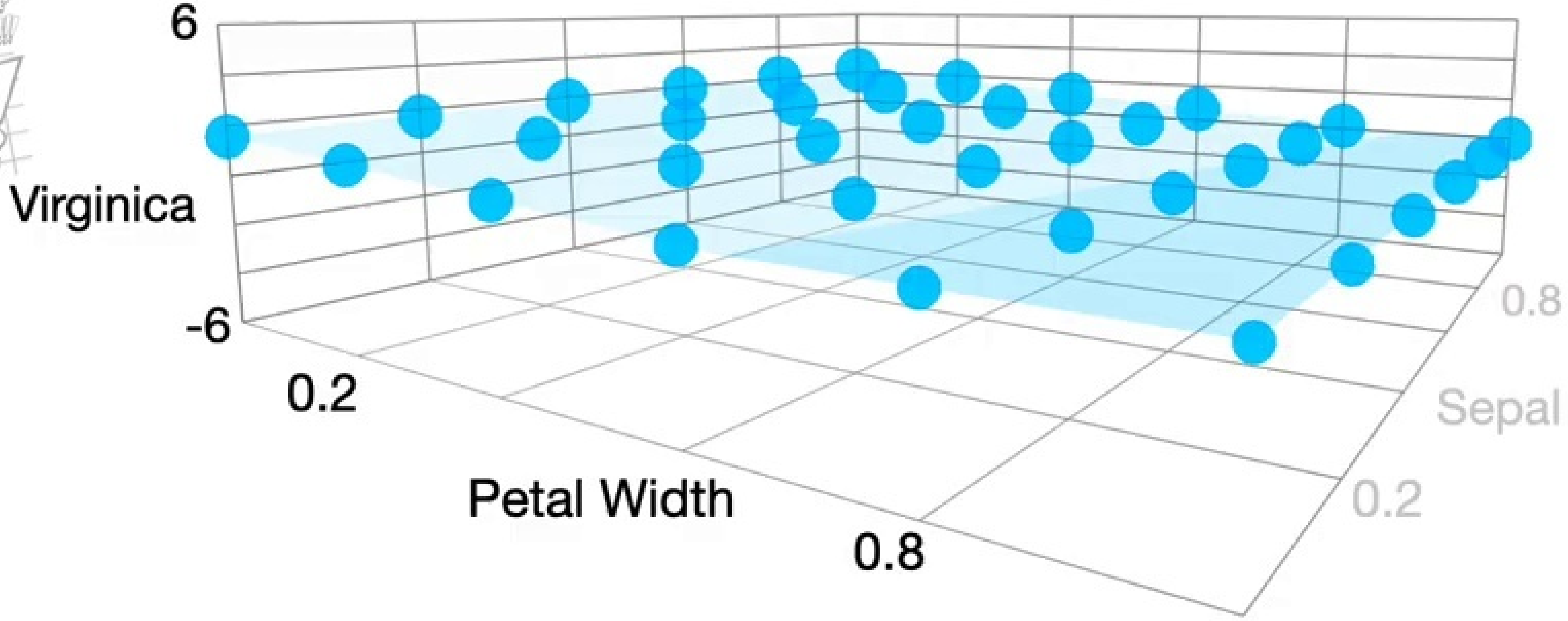


Now, just like we did for **Setosa** and **Versicolor**…

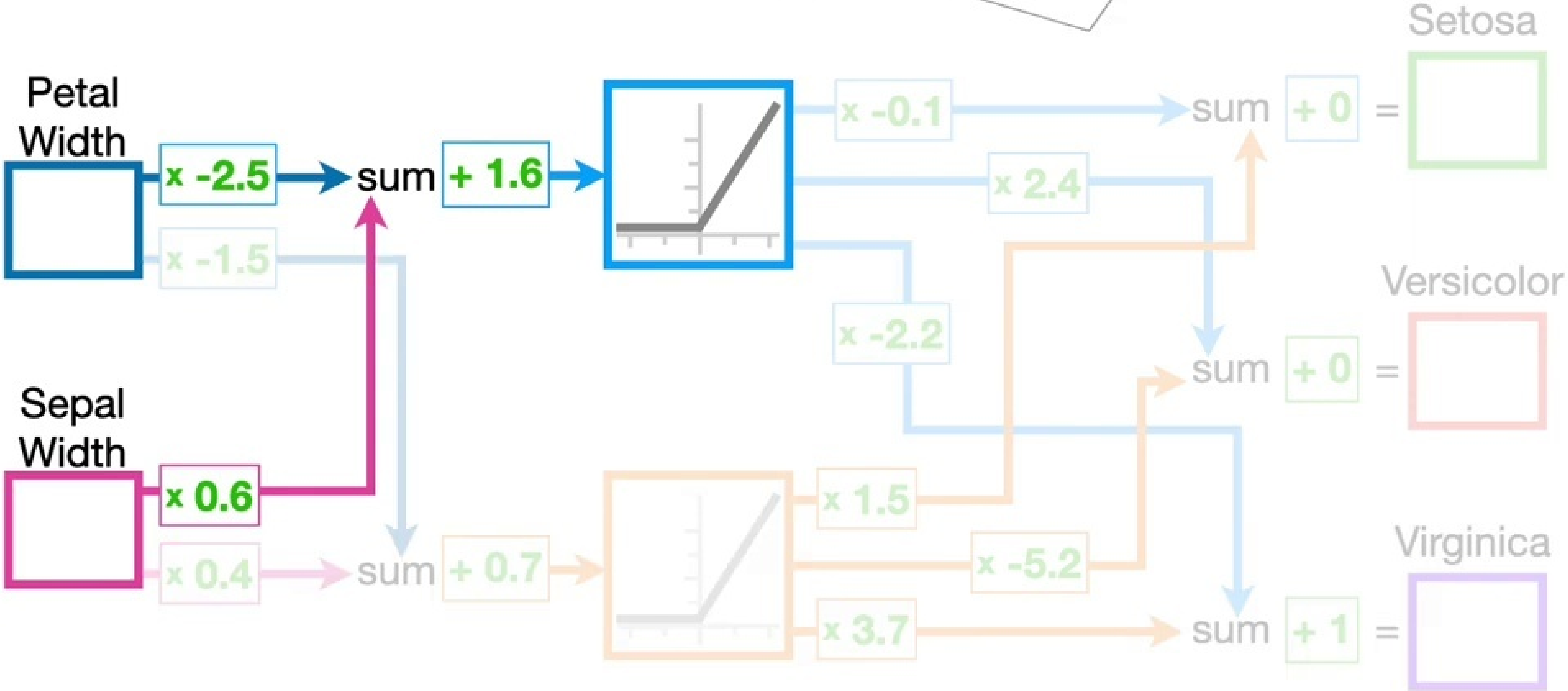


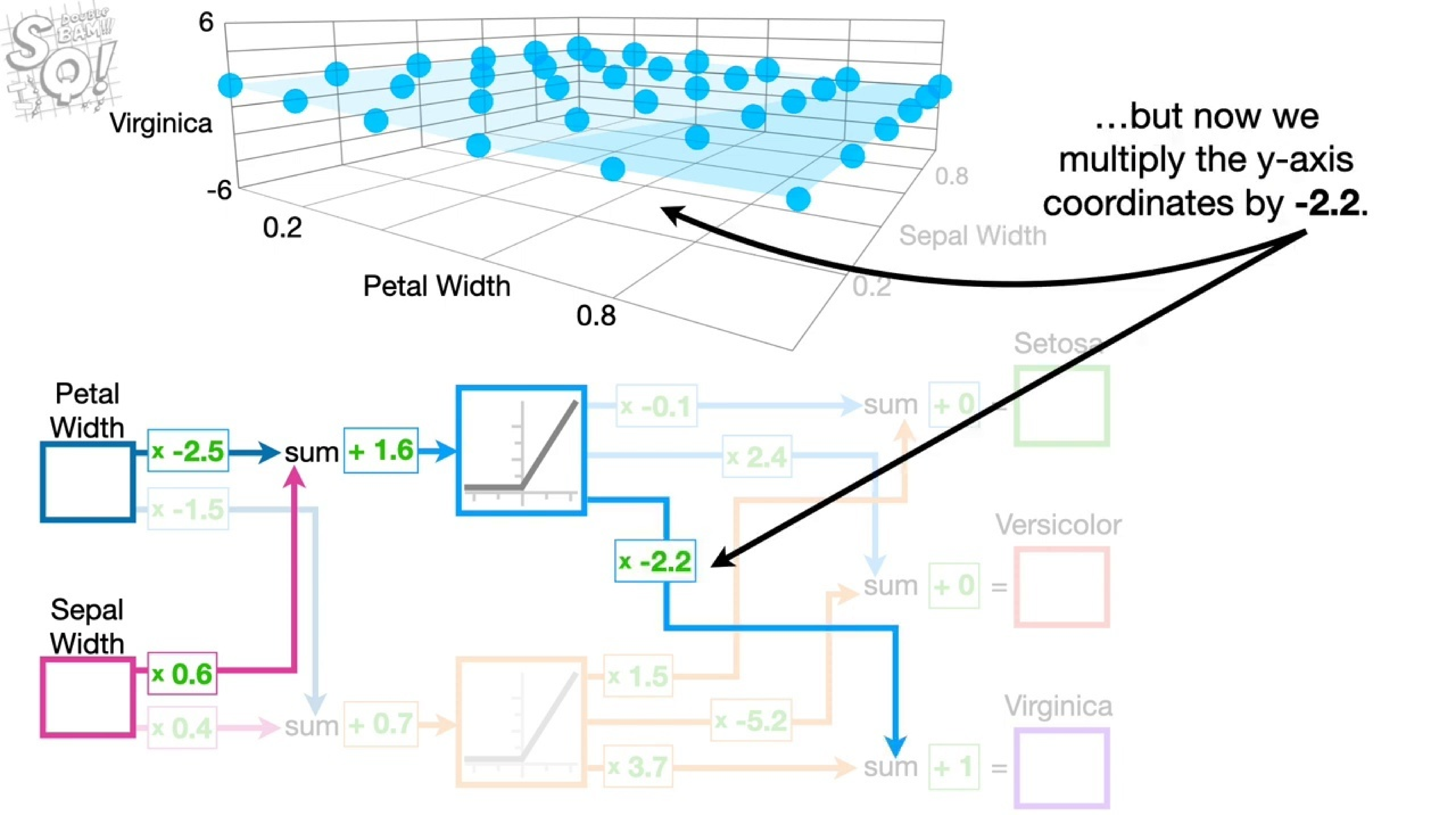
…let's determine the **crinkled surface** for **Virginica.**



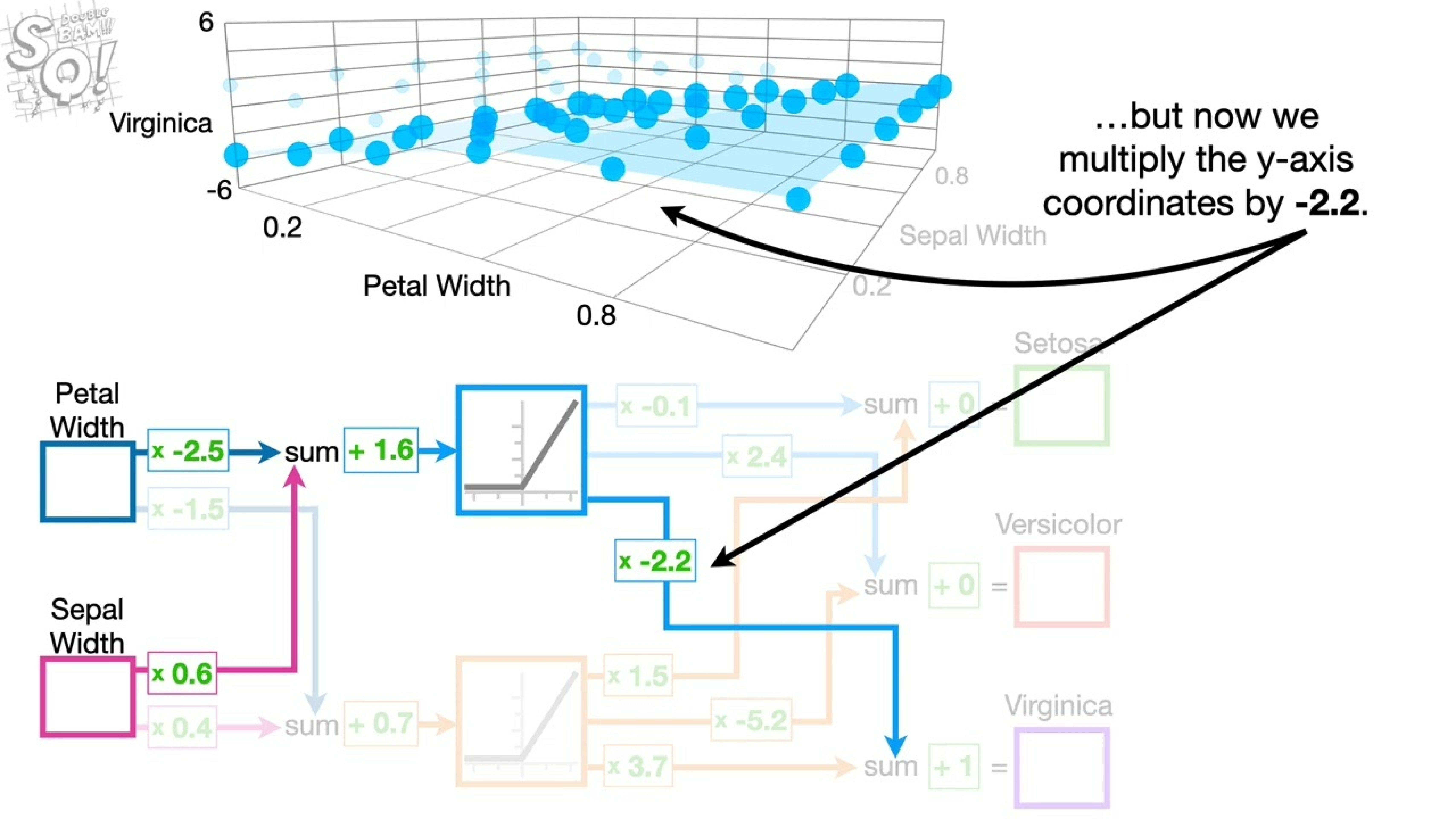


Just like before, we start with the **blue bent surface** from the top node in the **Hidden Layer**…

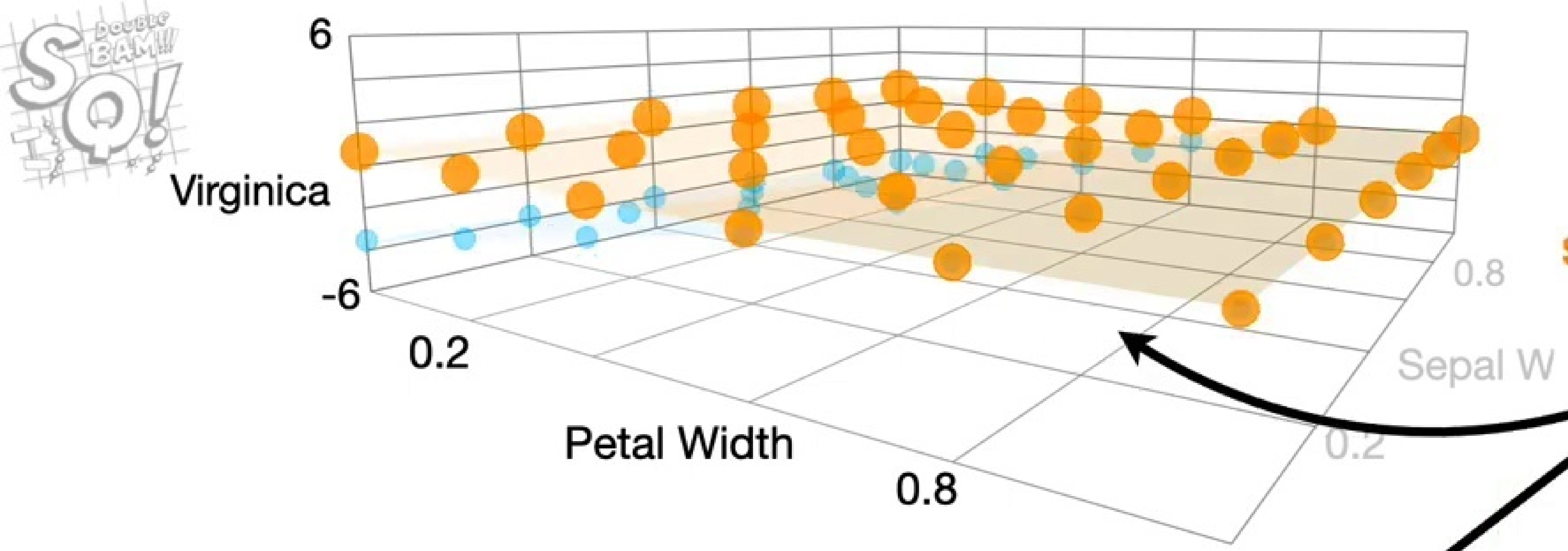




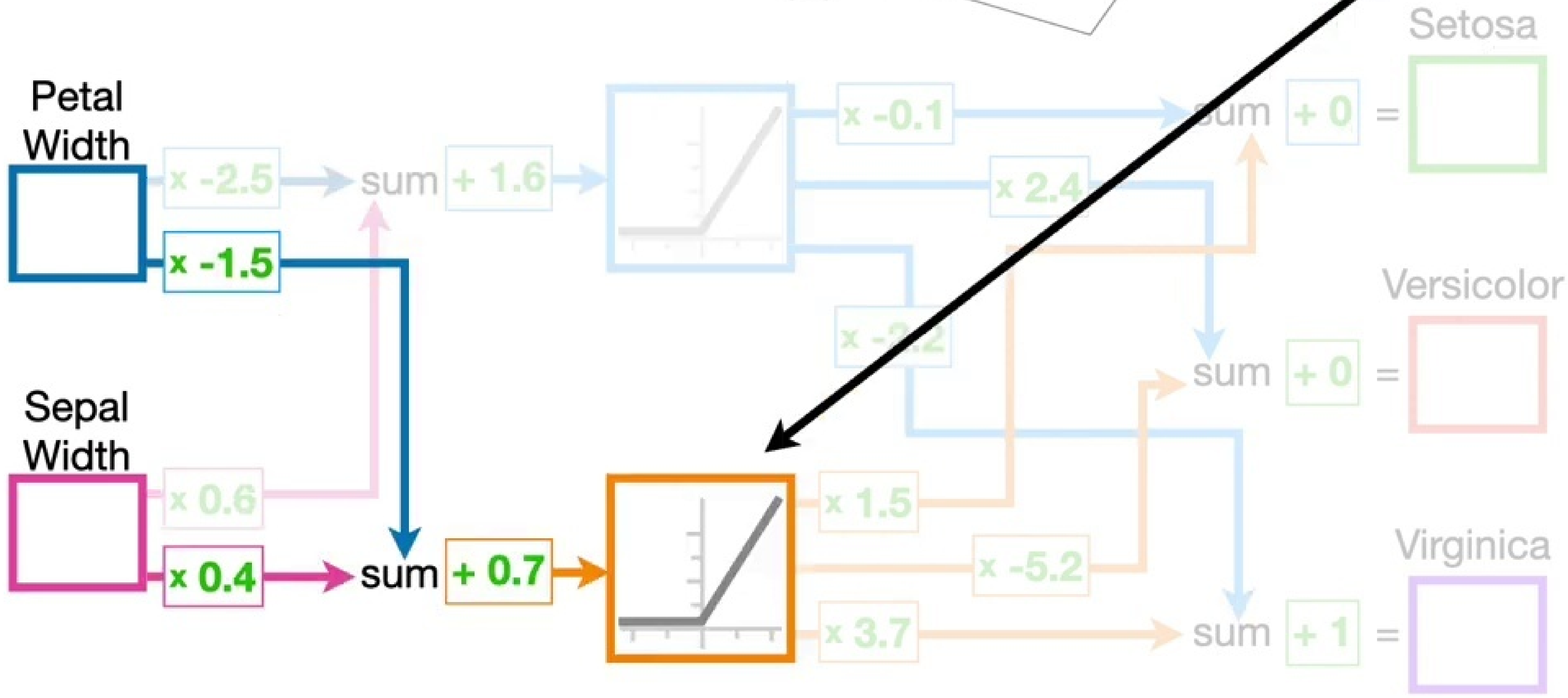
…but now we multiply the y-axis coordinates by -2.2.
6
-6
Virginica
0.2
Petal Width
0.8
0.8
Sepal Width
0.2
Setosa
Versicolor
Virginica
Petal Width
Sepal Width
x -2.5
x -1.5
x 0.6
x 0.4
sum
+ 1.6
sum
+ 0.7
x -0.1
x 2.4
x -2.2
x 1.5
x -5.2
x 3.7
sum
+ 0
sum
+ 0
=
sum
+ 1
=

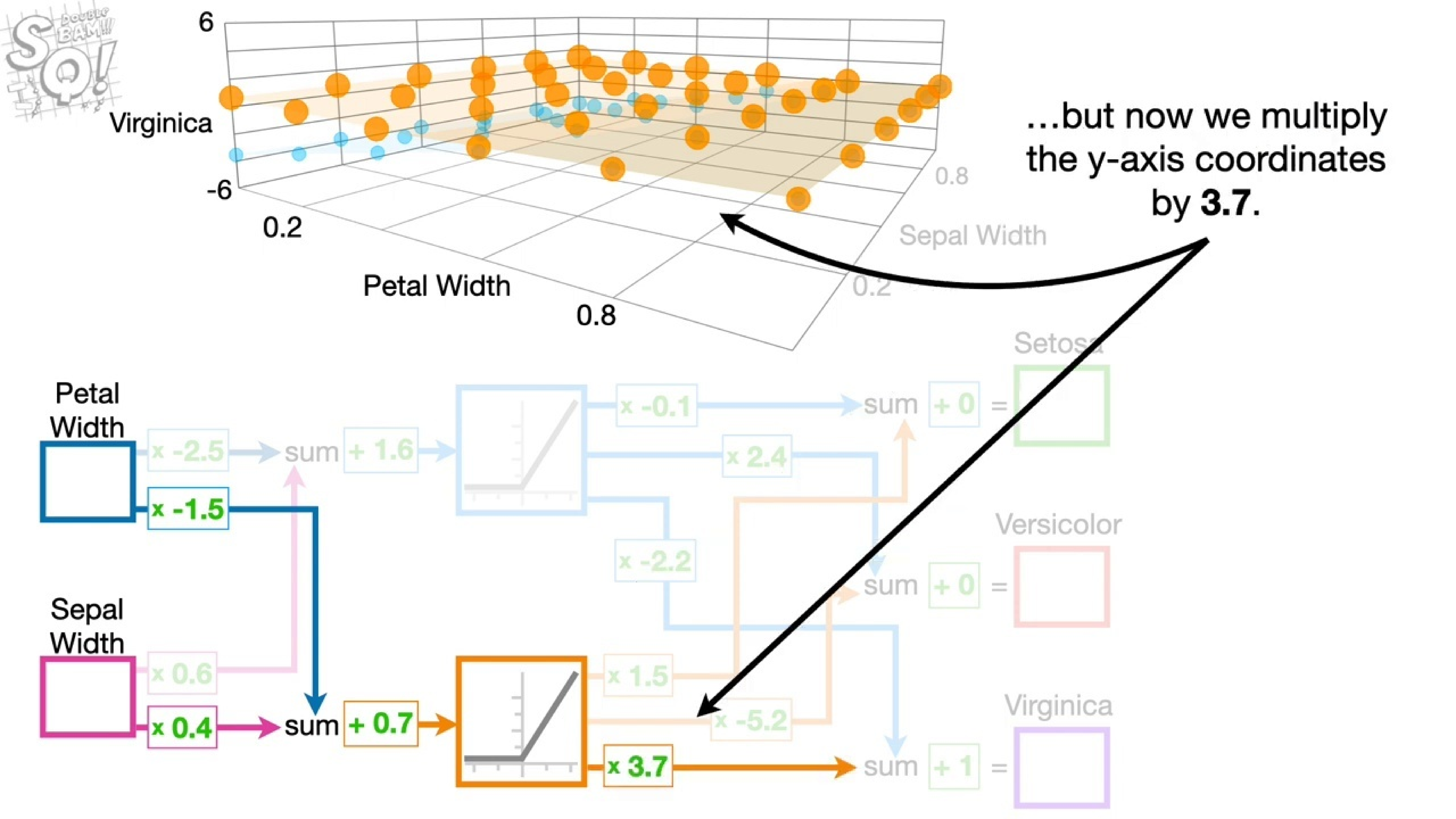


...but now we multiply the y-axis coordinates by **-2.2**.
6
Virginica
-6
0.2
Petal Width
0.8
0.8
Sepal Width
0.2
Setosa
Versicolor
Virginica
Petal Width
Sepal Width
x -2.5
x -1.5
x 0.6
x 0.4
sum
+ 1.6
sum
+ 0.7
x -0.1
x 2.4
x -2.2
x 1.5
x -5.2
x 3.7
sum
+ 0
sum
+ 0
=
sum
+ 1
=

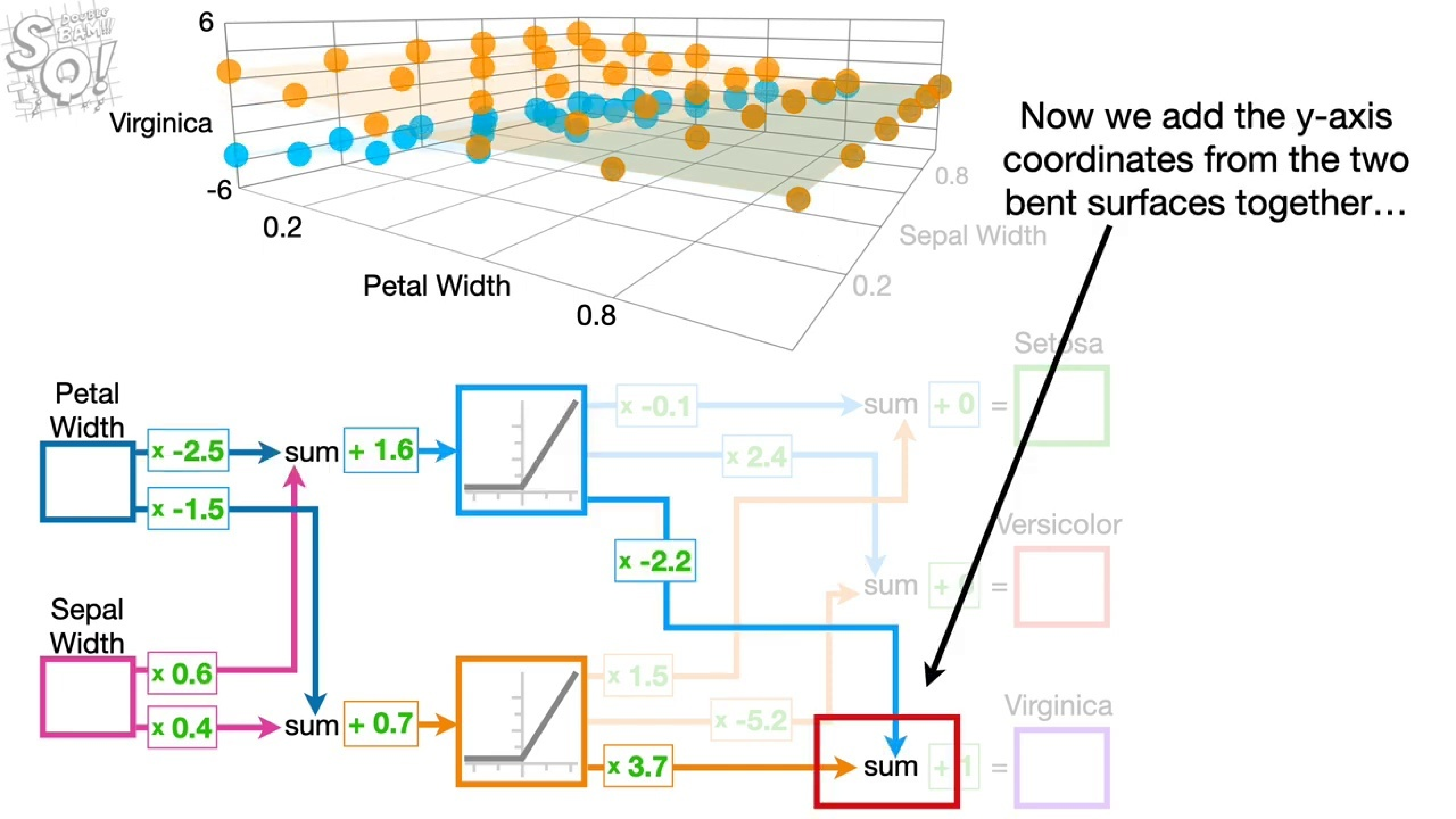


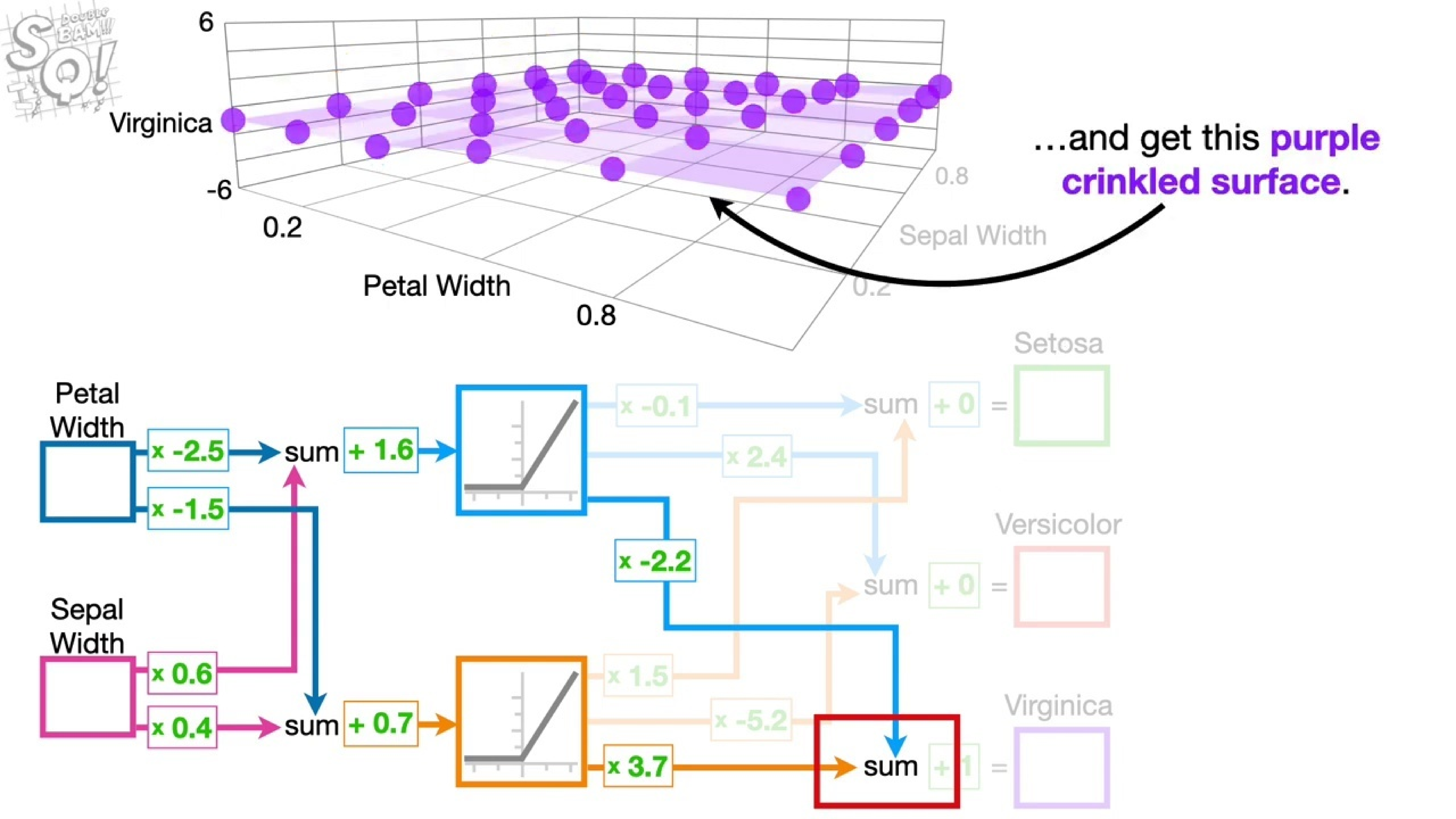
And just like before, we create the **orange bent surface** from the bottom node in the **Hidden Layer**…



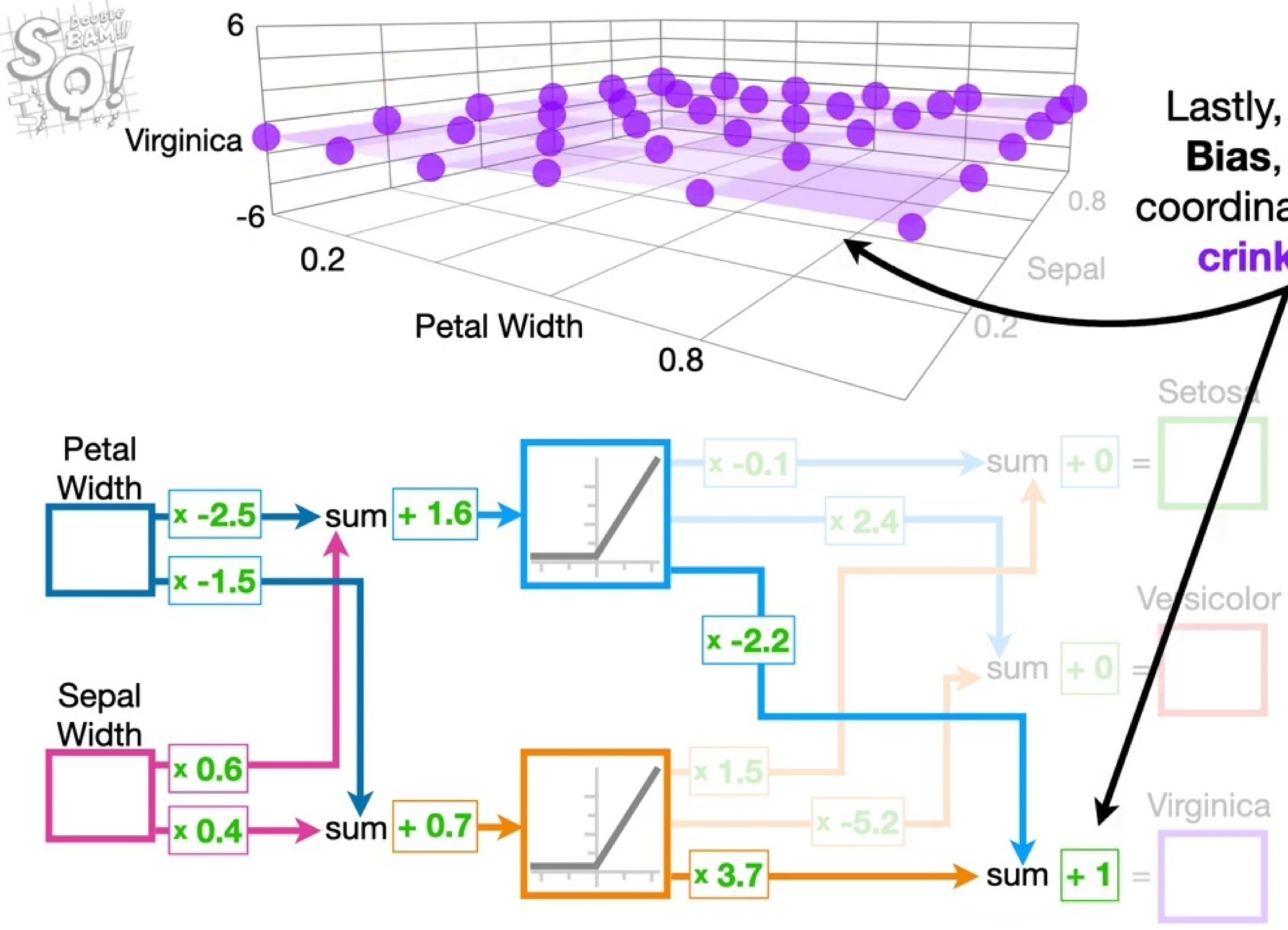


6
Virginica
-6
0.2
Petal Width
0.8
0.8
Sepal Width
0.2
…but now we multiply the y-axis coordinates by **3.7**.
Petal Width
x -2.5
x -1.5
sum
+ 1.6
x -0.1
x 2.4
x -2.2
sum
+ 0
=
Setosa
Versicolor
sum
+ 0
=
Sepal Width
x 0.6
x 0.4
sum
+ 0.7
x 1.5
x -5.2
x 3.7
Virginica
sum
+ 1
=

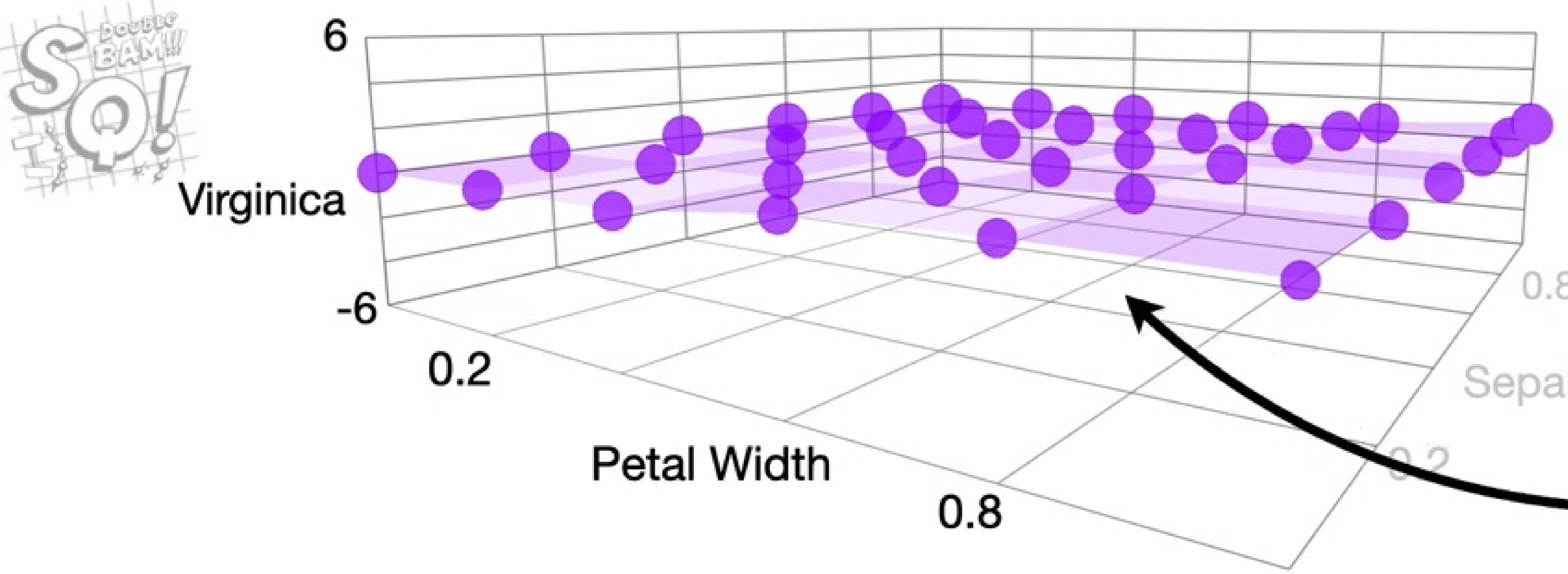

Now we add the y-axis coordinates from the two bent surfaces together…
6
-6
Virginica
0.2
Petal Width
0.8
0.8
Sepal Width
0.2
Petal Width
Sepal Width
x -2.5
x -1.5
x 0.6
x 0.4
sum
+ 1.6
sum
+ 0.7
x -0.1
x 2.4
x -2.2
x 1.5
x -5.2
x 3.7
sum
+ 0
=
Setosa
sum
=
Versicolor
sum
=
Virginica



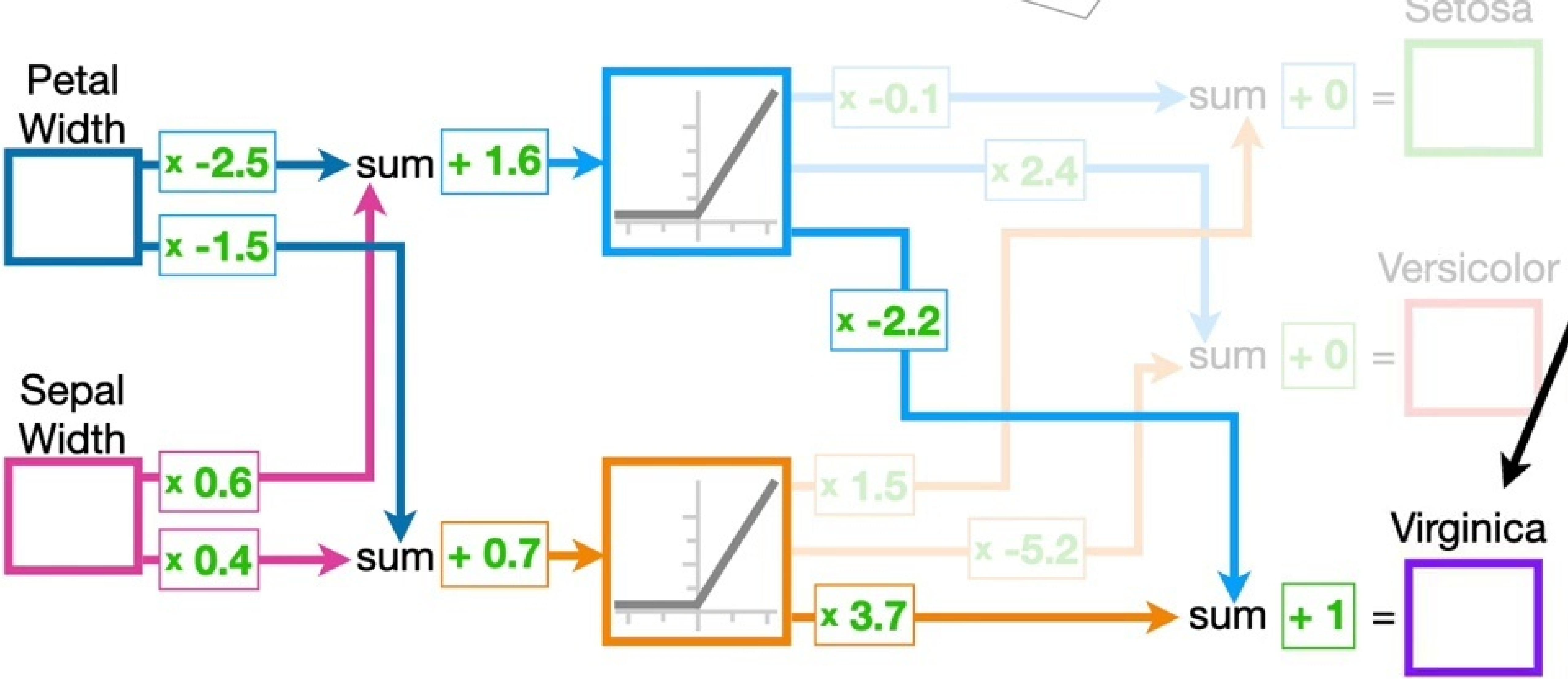
…and get this purple crinkled surface.
6
Virginica
-6
0.2
Petal Width
0.8
0.8
Sepal Width
0.2
Petal Width
Sepal Width
x -2.5
x -1.5
x 0.6
x 0.4
sum
+ 1.6
sum
+ 0.7
x -0.1
x 2.4
x -2.2
x 1.5
x -5.2
x 3.7
sum
+ 0
=
sum
+ 0
=
sum
+ 1
=
Setosa
Versicolor
Virginica

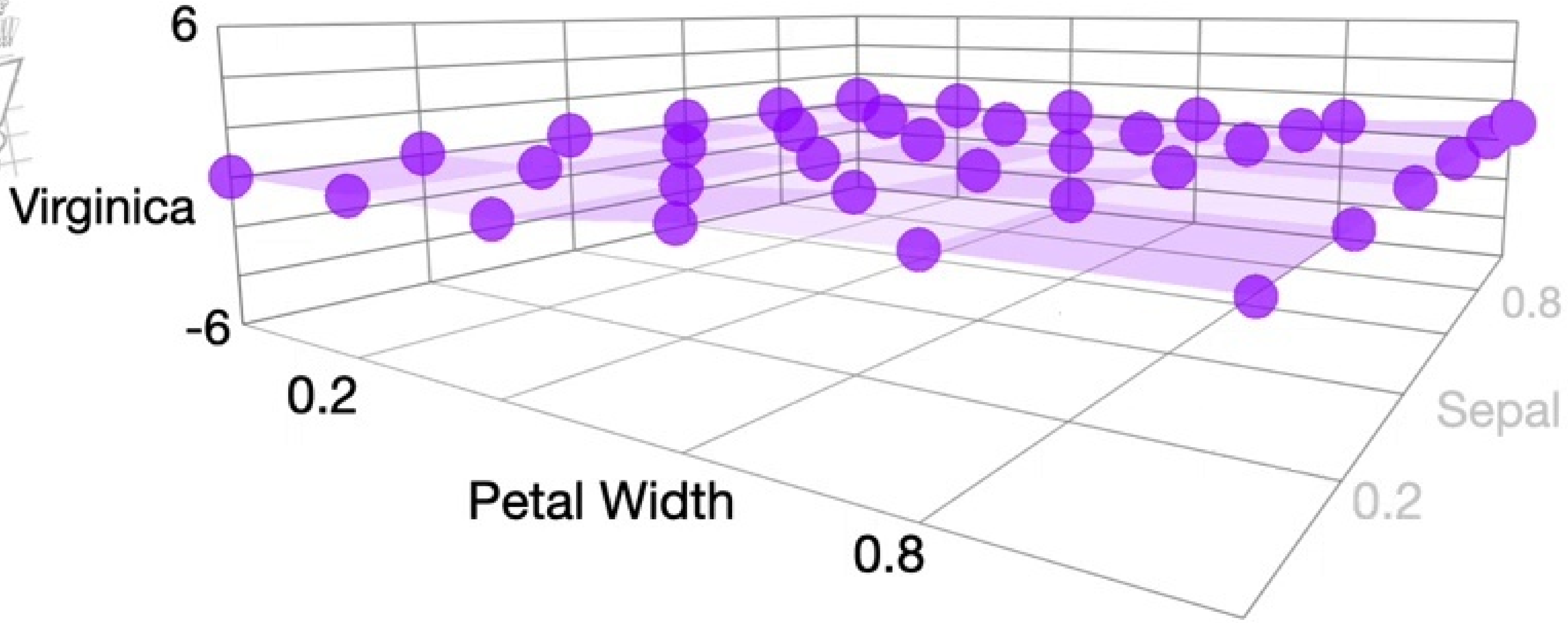


Lastly, we add the final **Bias**, **1**, to the y-axis coordinates on the **purple crinkled surface**…

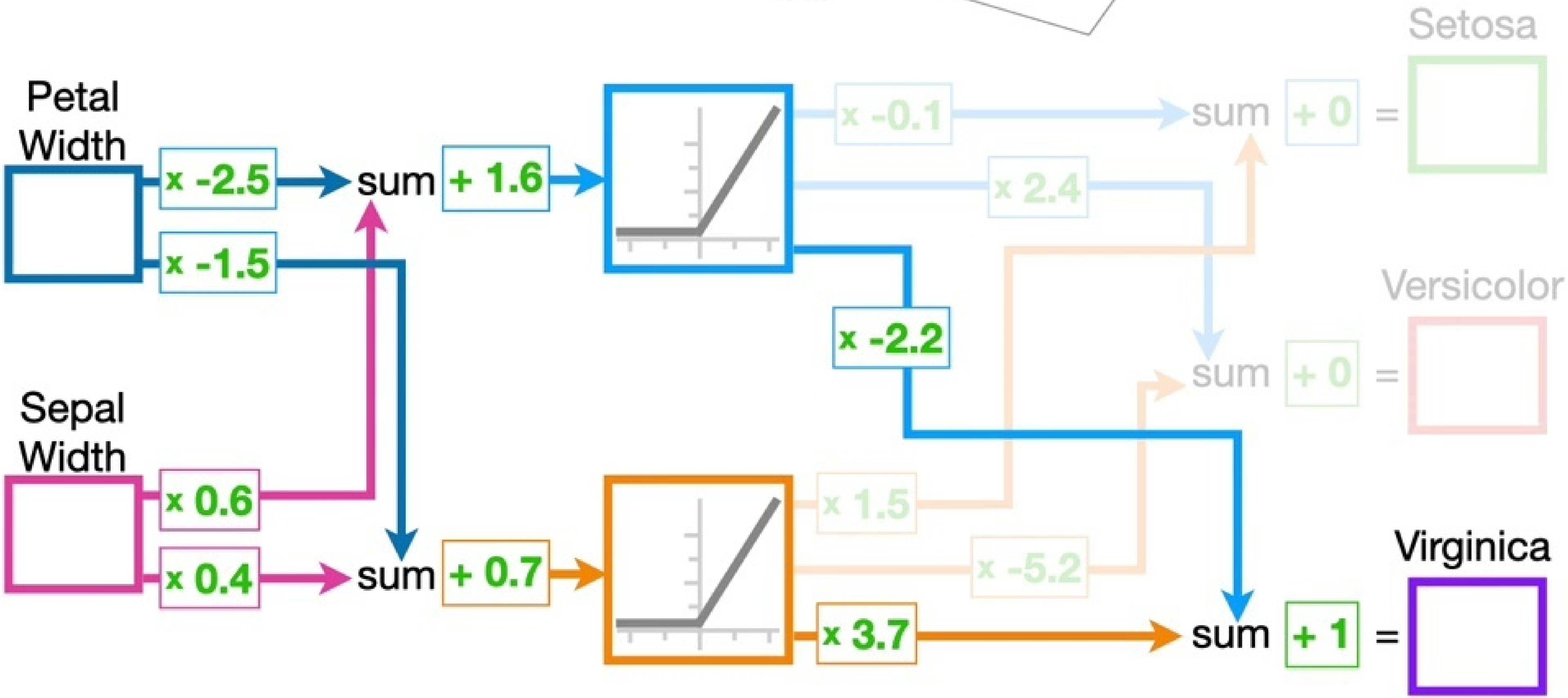


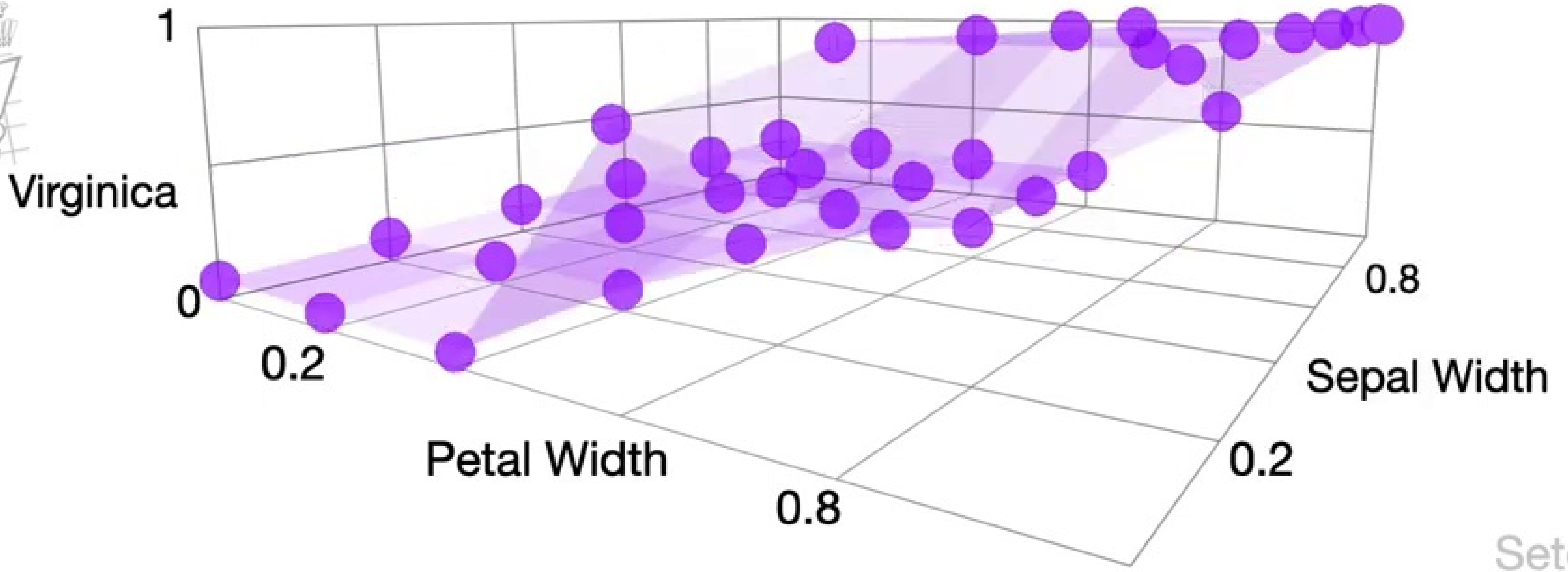
...and that gives us the final surface for predicting if the **Iris** species is **Virginica.**





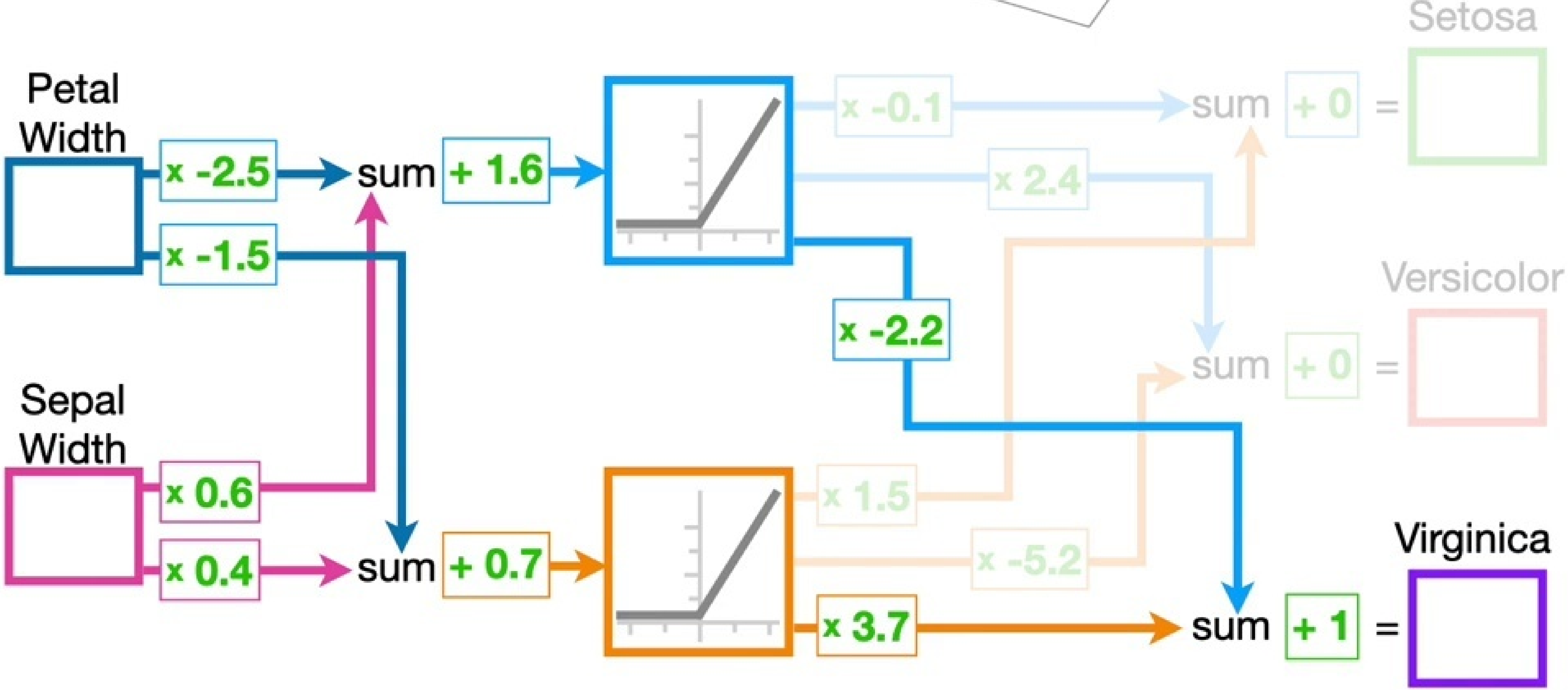
Now, so we can see what's going on, let's change the scale for the y-axis from **-6** to **6**…



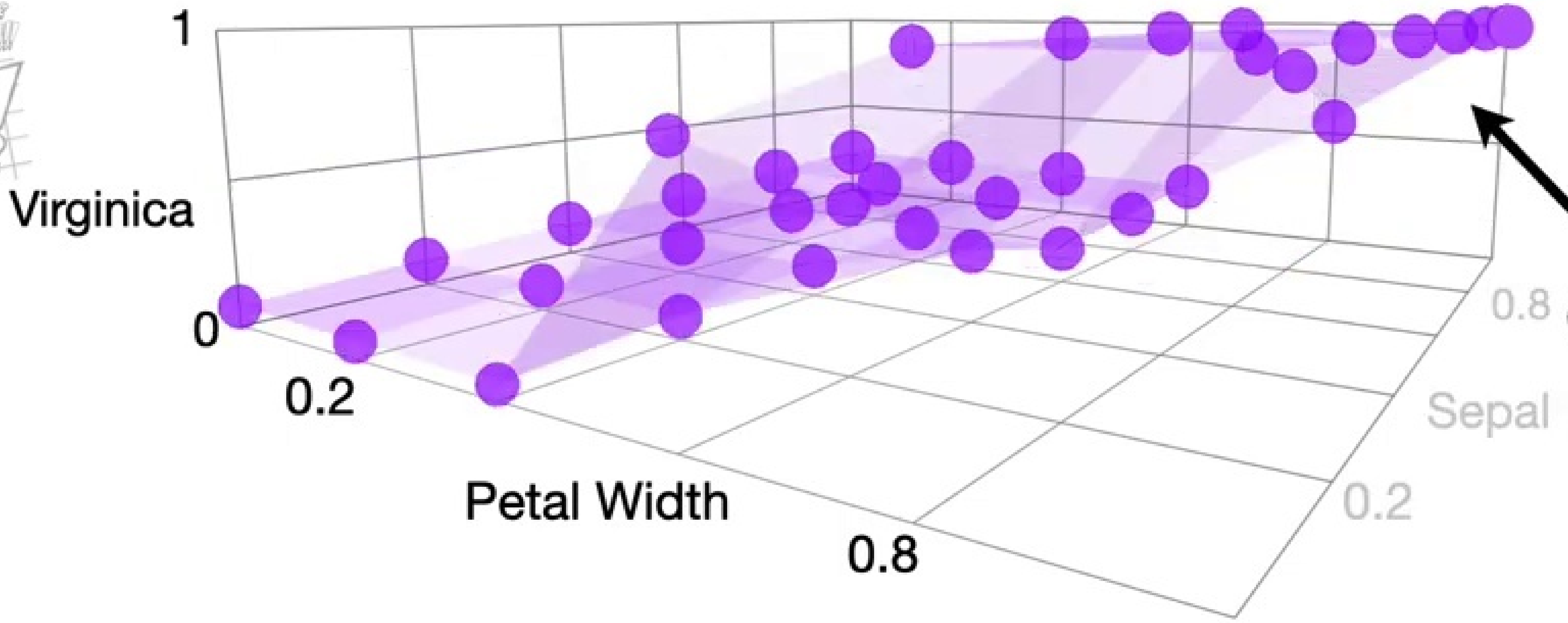


1
Virginica
0
0.2
Petal Width
0.8
0.8
Sepal Width
0.2

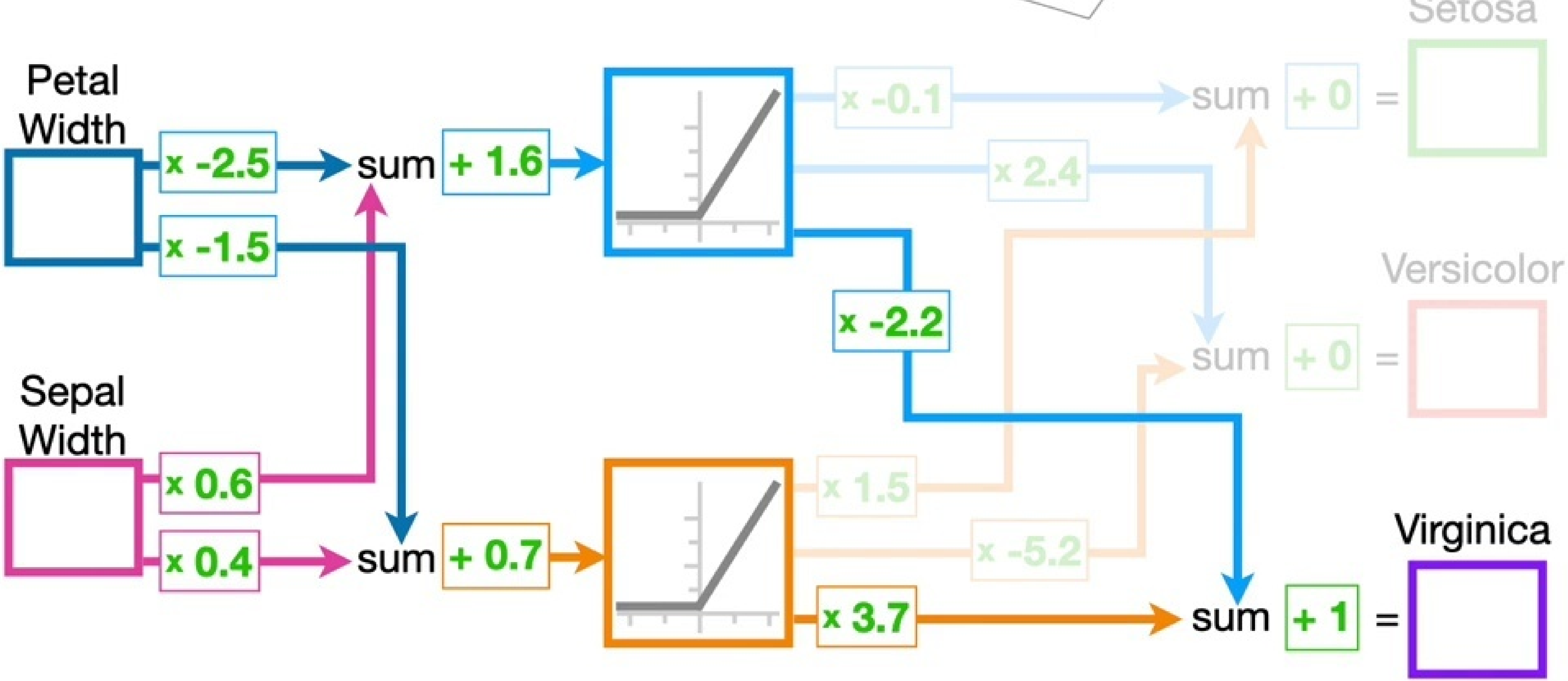
…to **0** to **1**.

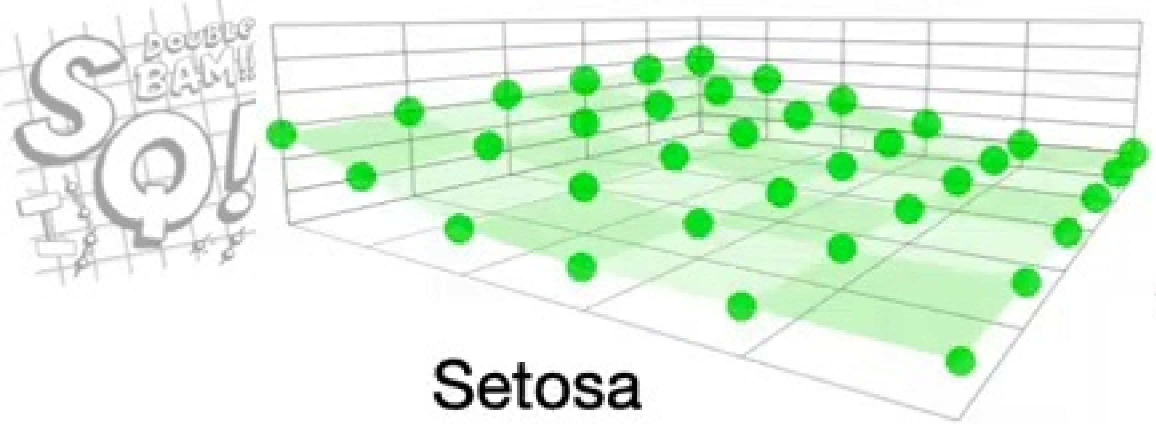


Petal Width
x -2.5
x -1.5
sum
+ 1.6
Sepal Width
x 0.6
x 0.4
sum
+ 0.7
x -0.1
x 2.4
x -2.2
x 1.5
x -5.2
x 3.7
Setosa
sum
+ 0
=
Versicolor
sum
+ 0
=
Virginica
sum
+ 1
=

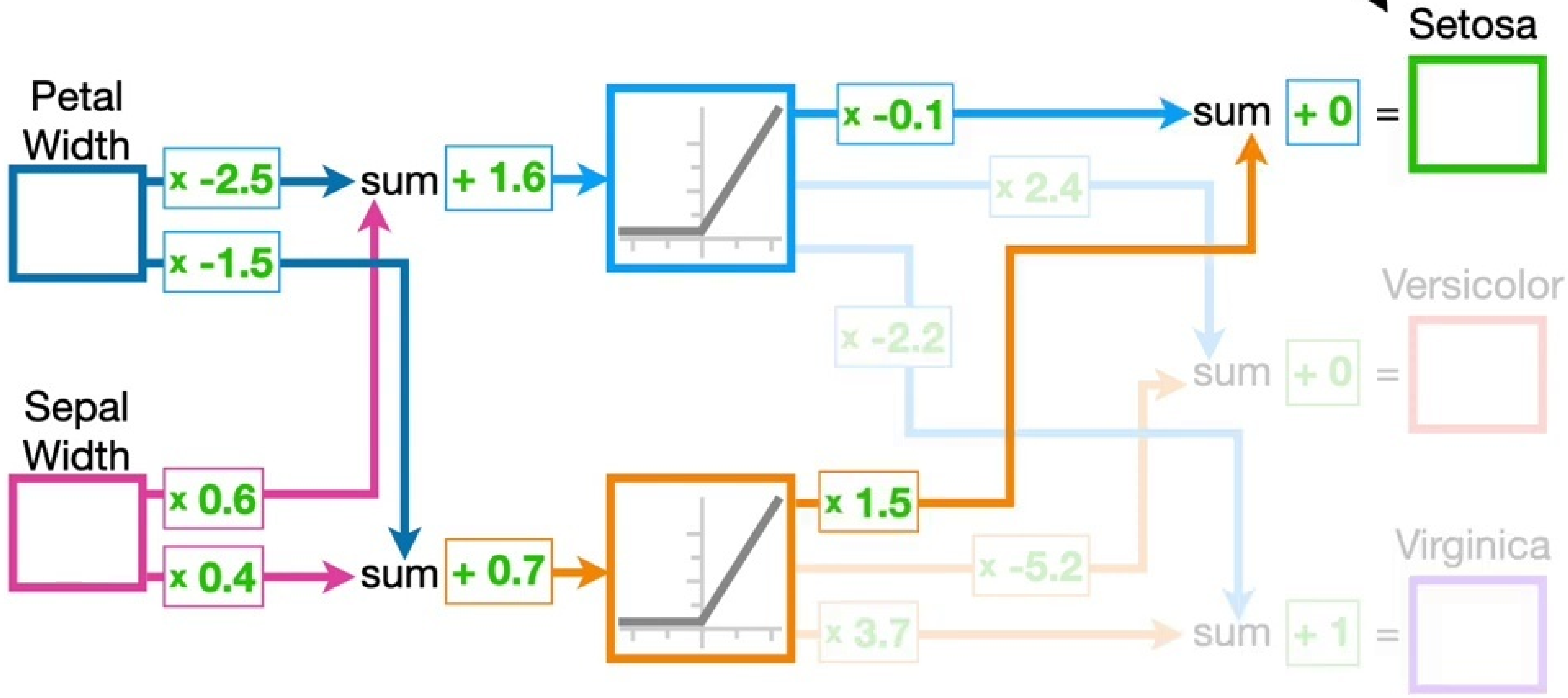


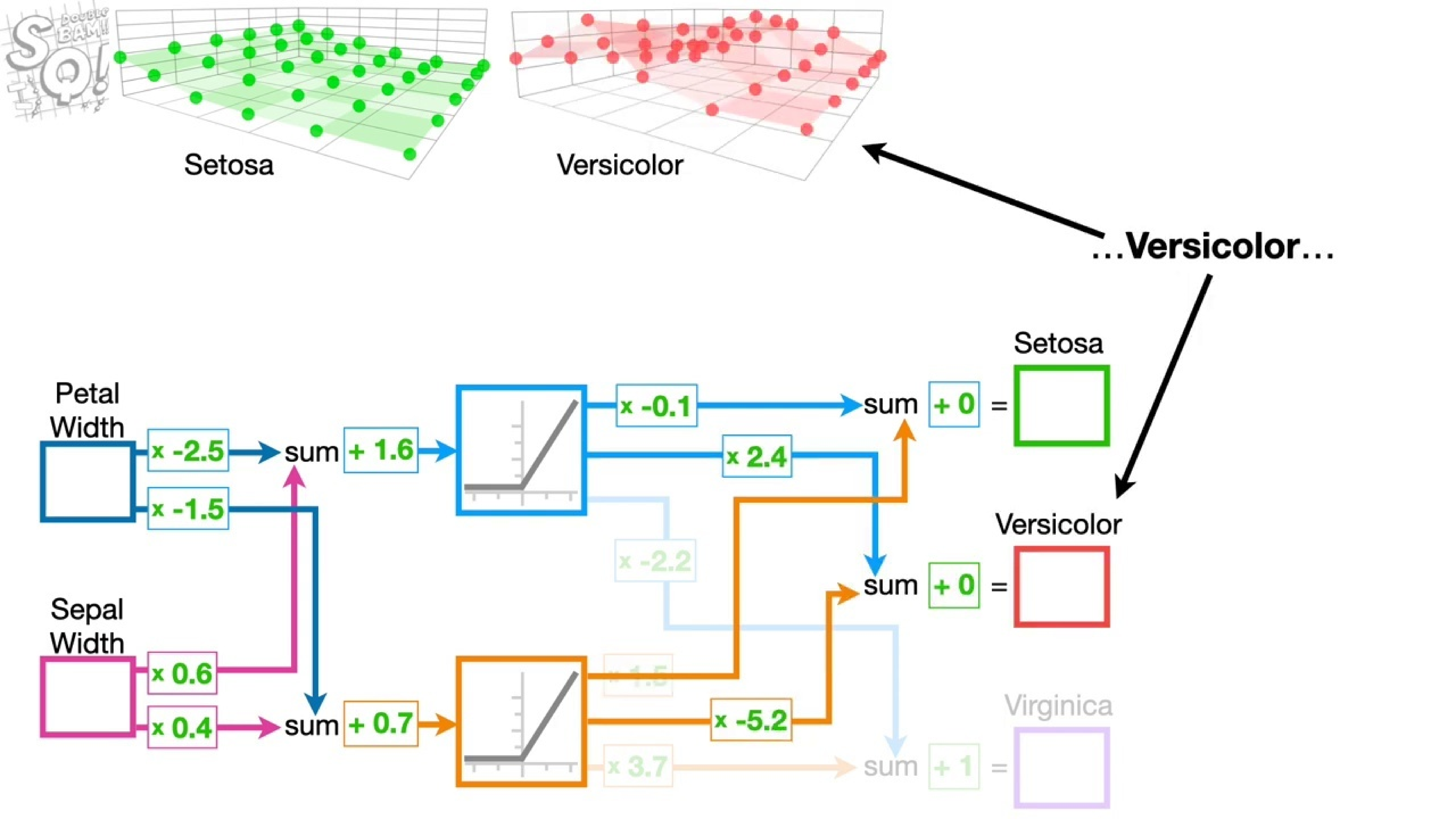
Now we see that when **Petal Width** is close to **1** (the widest), then we will get a high score for **Virginica**.



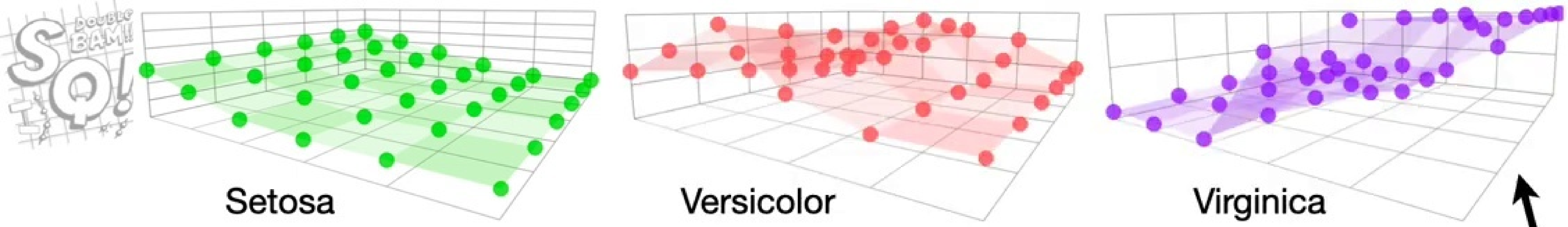


At long last, we have **crinkled surfaces** for **Setosa**…

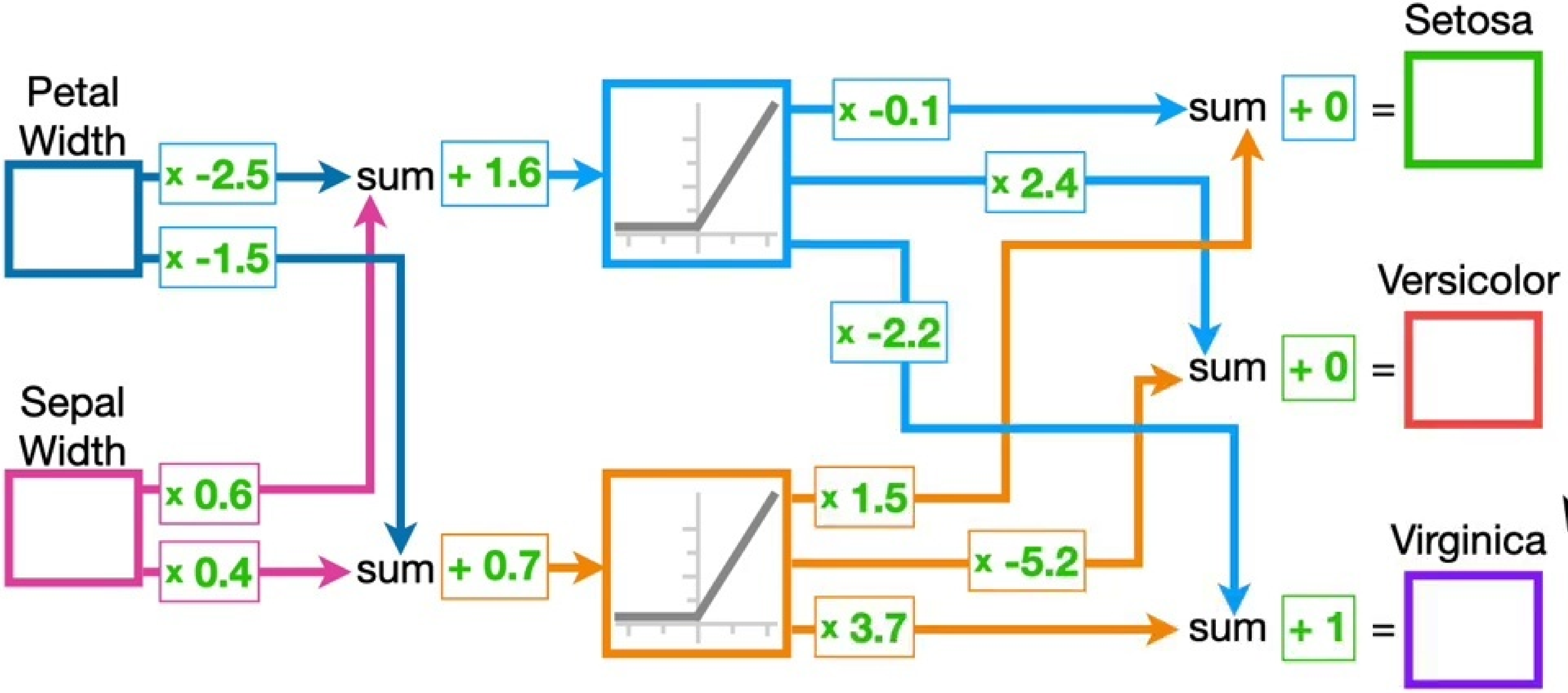


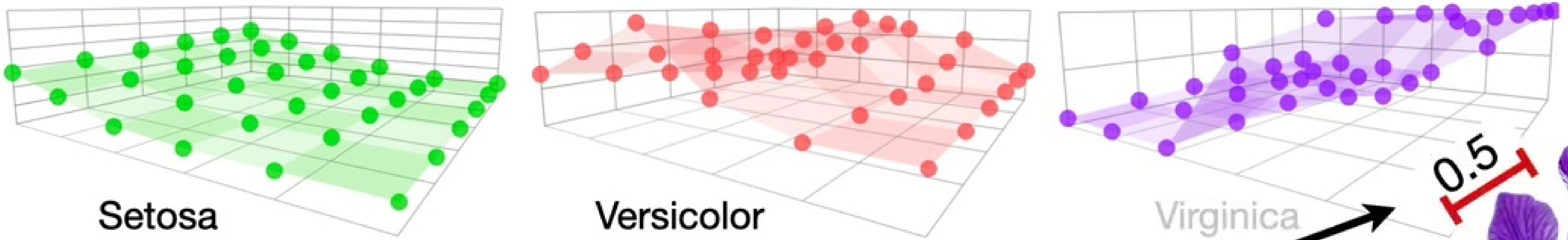


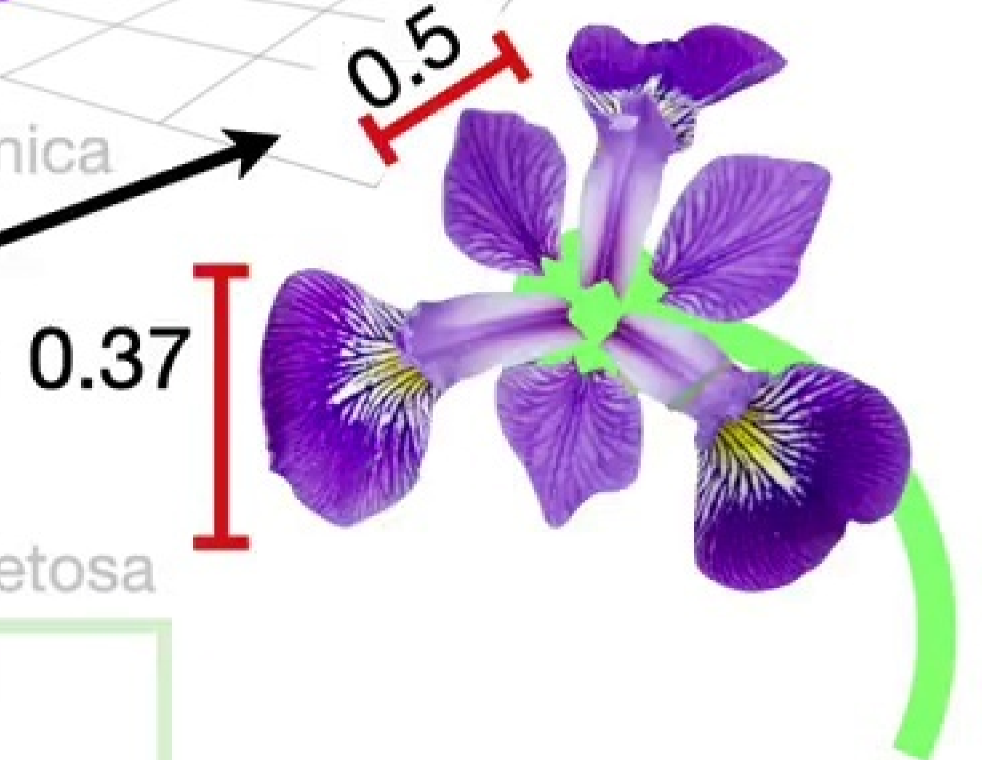
Setosa
Versicolor
…Versicolor…
Petal Width
x -2.5
x -1.5
sum
+ 1.6
x -0.1
x 2.4
x -2.2
sum
+ 0
=
Setosa
sum
+ 0
=
Versicolor
Sepal Width
x 0.6
x 0.4
sum
+ 0.7
x 1.5
x -5.2
x 3.7
sum
+ 1
=
Virginica



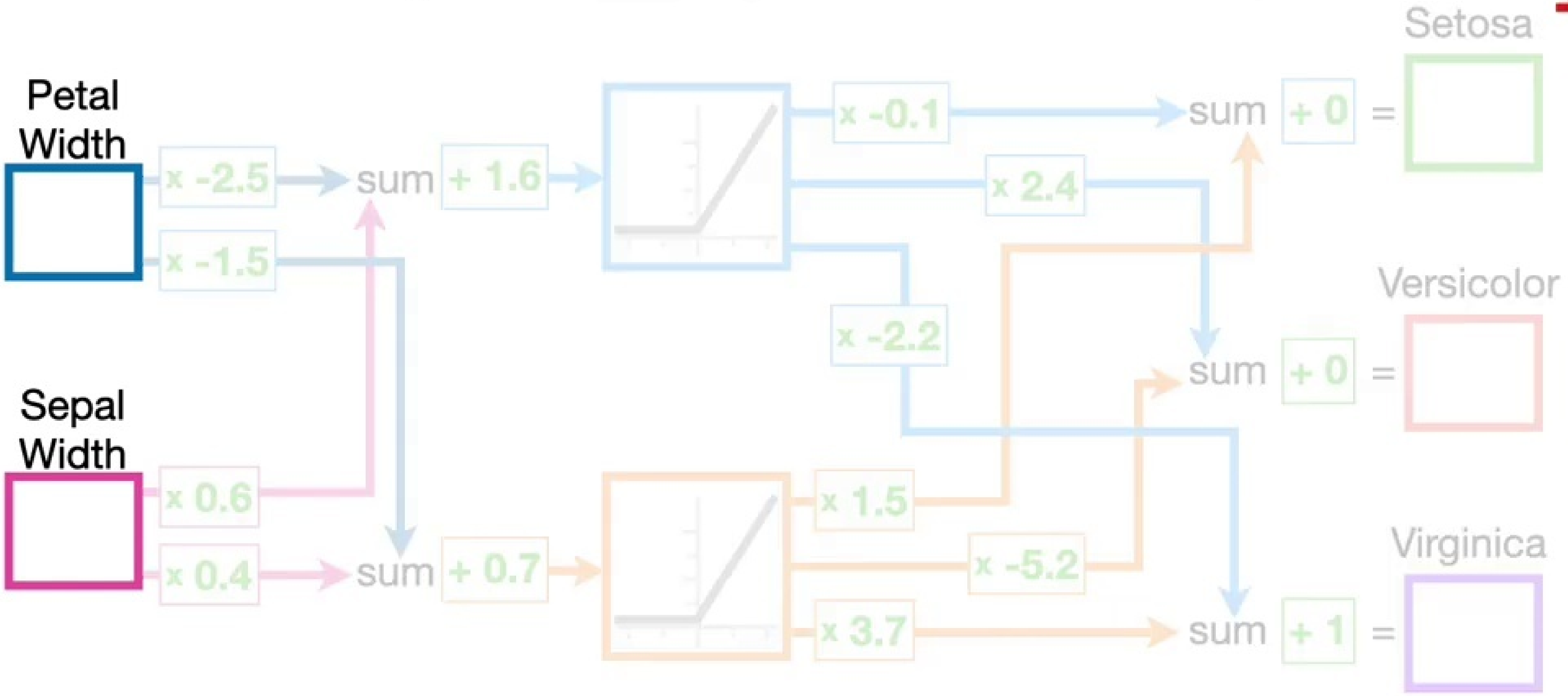
…and **Virginica.**

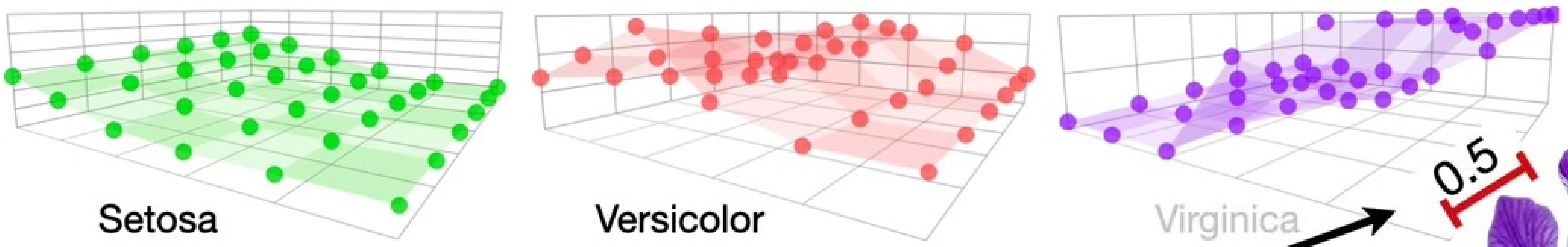


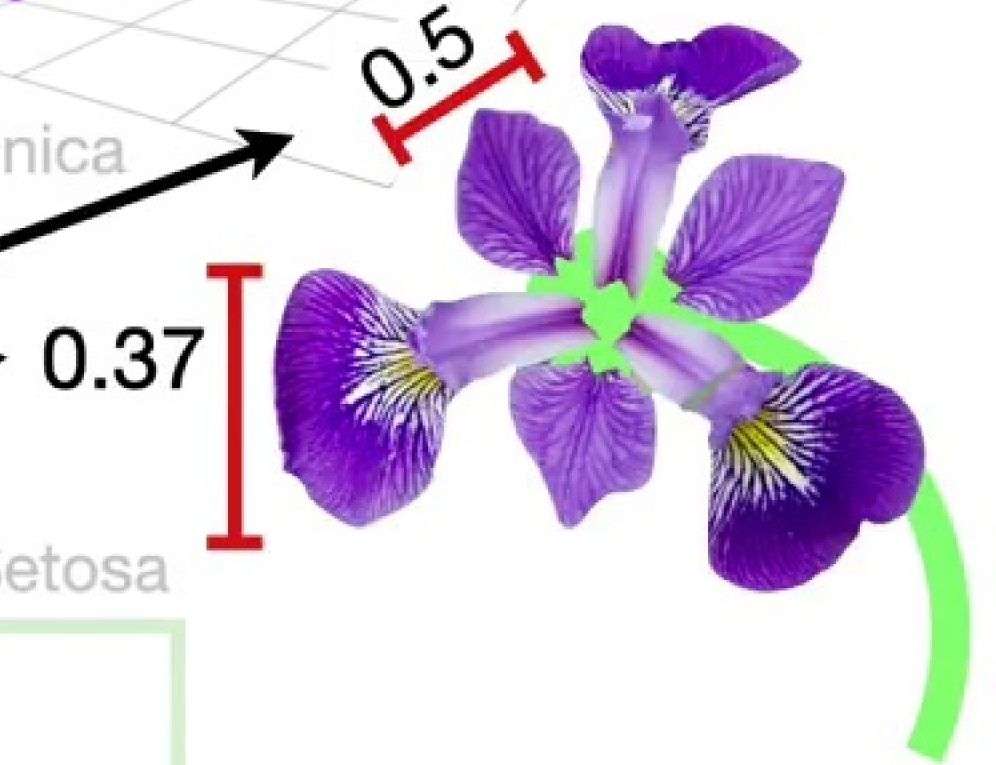




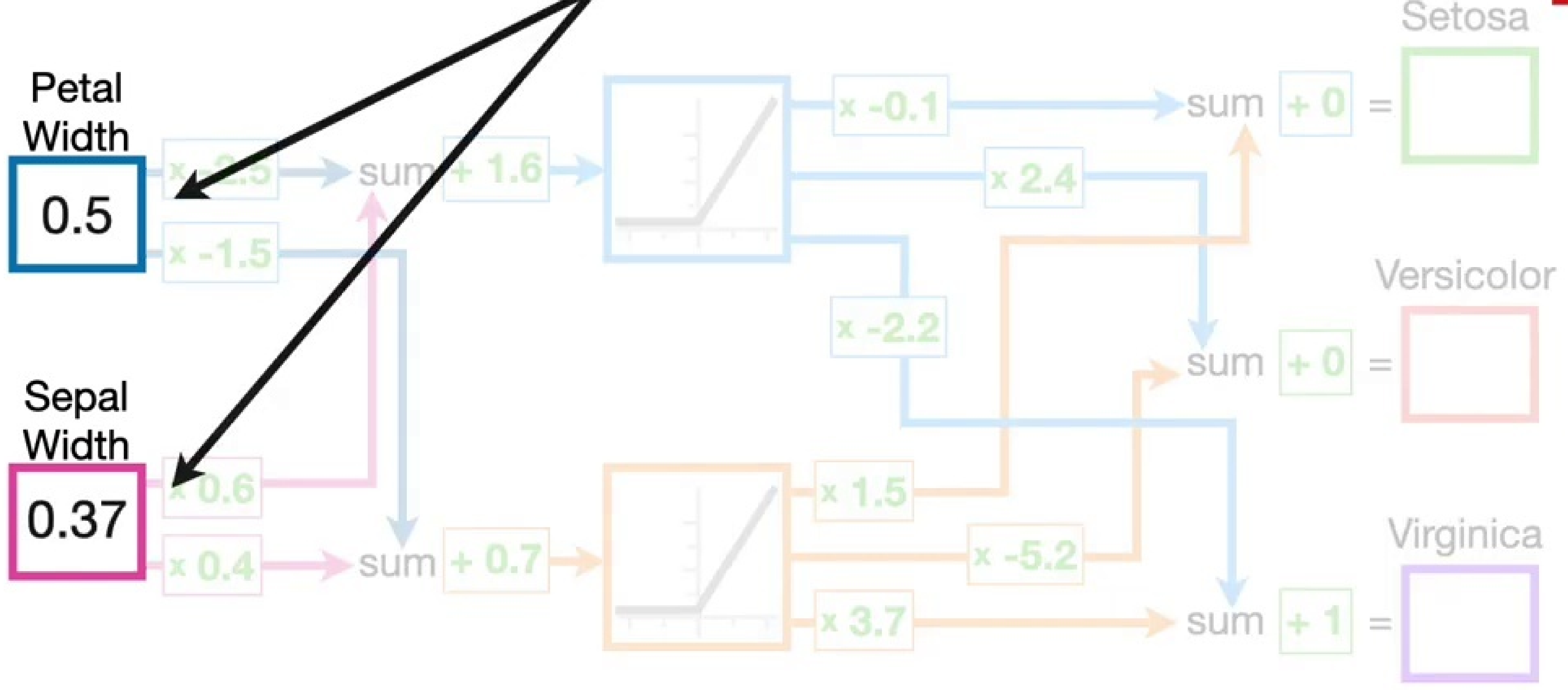
Now we can plug in the **Petal** and **Sepal Widths** from the **Iris** we found…

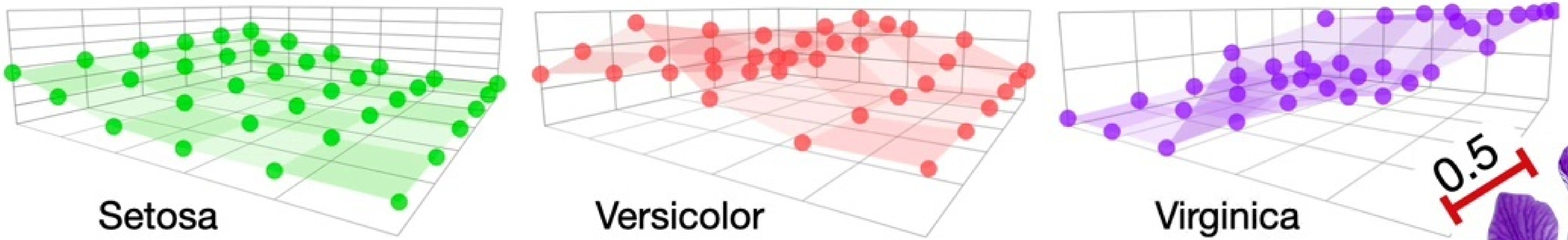


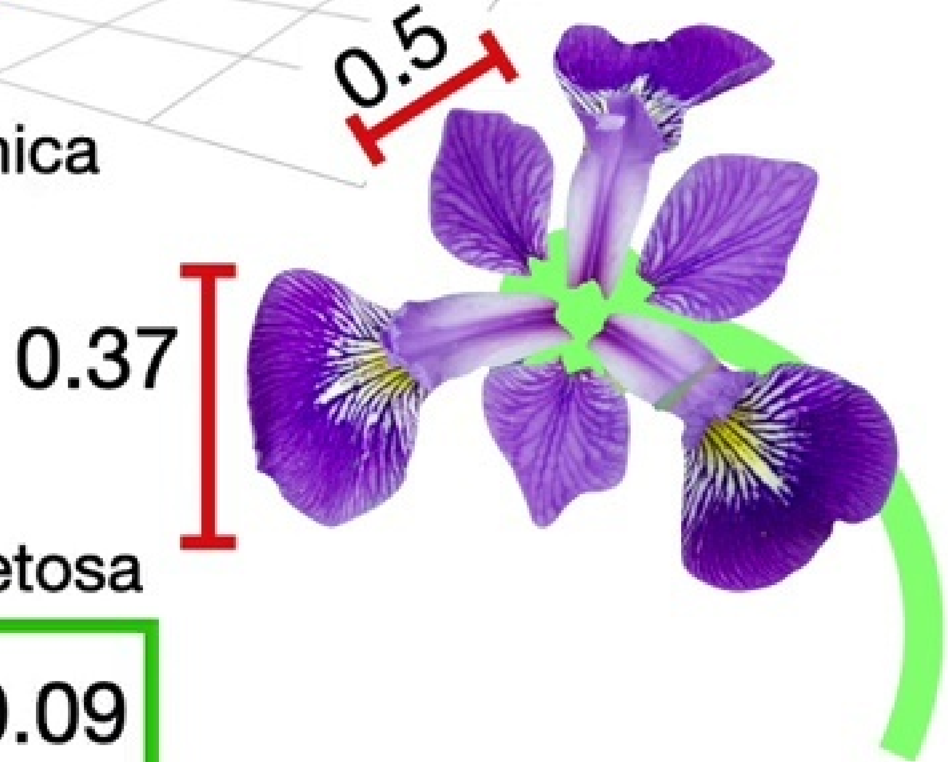




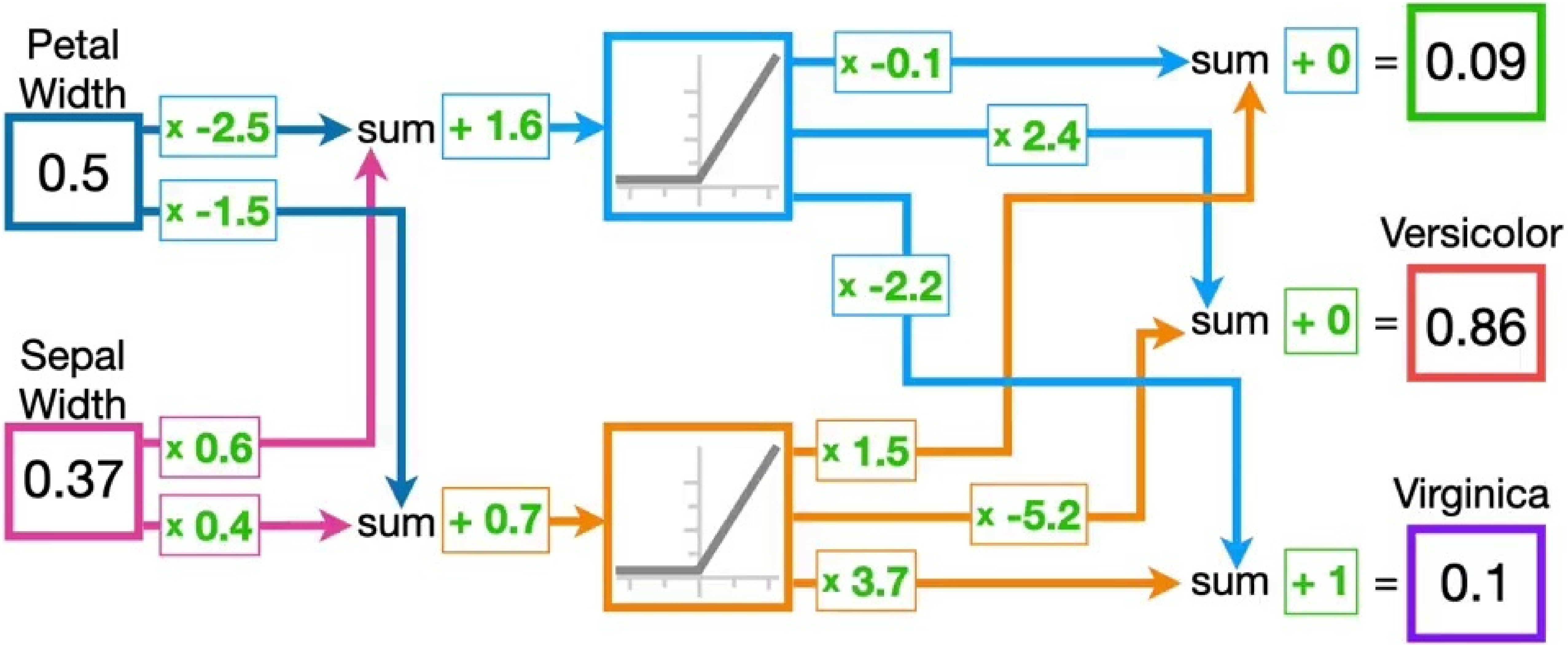
Now we can plug in the **Petal** and **Sepal Widths** from the **Iris** we found…

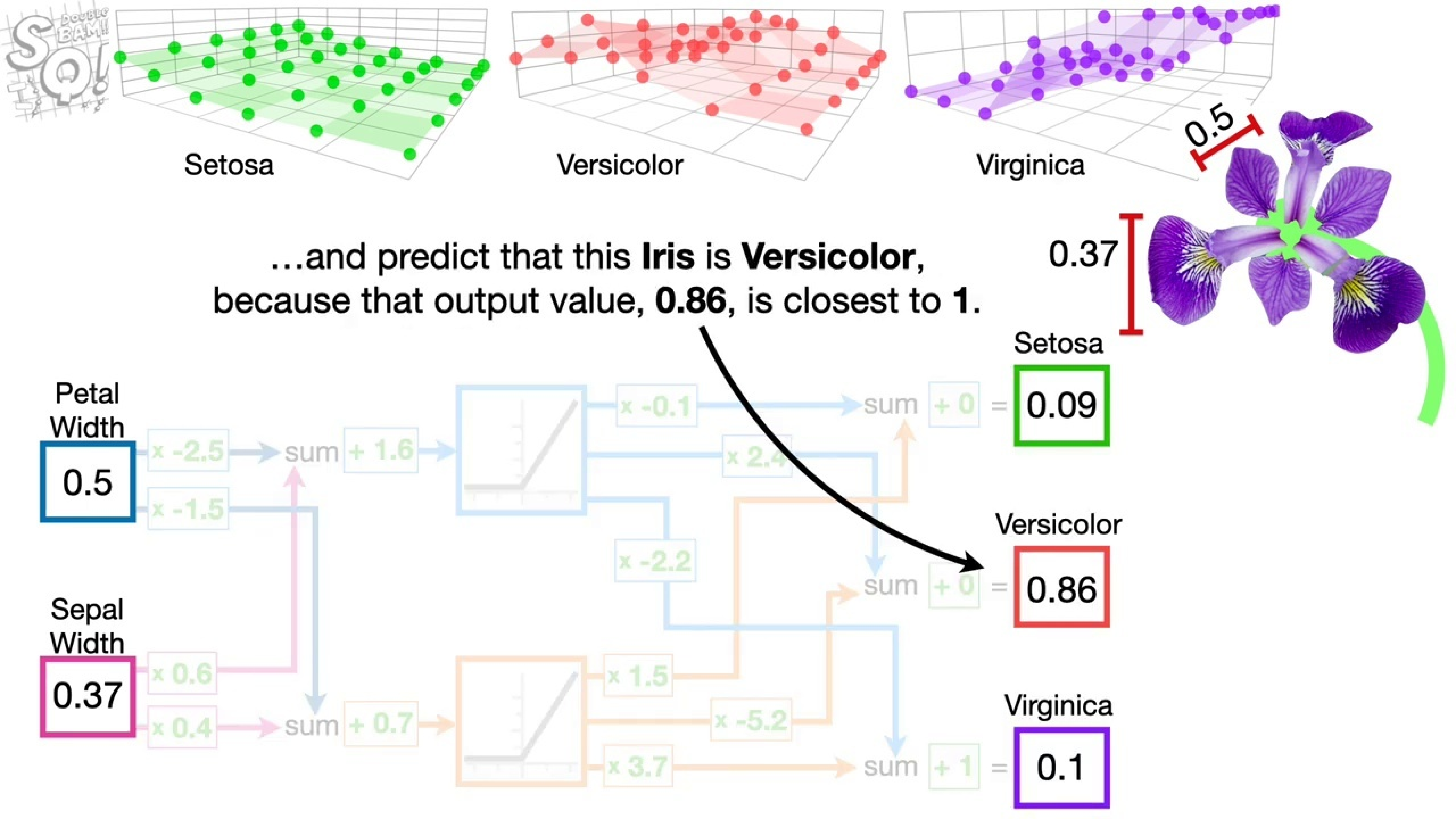


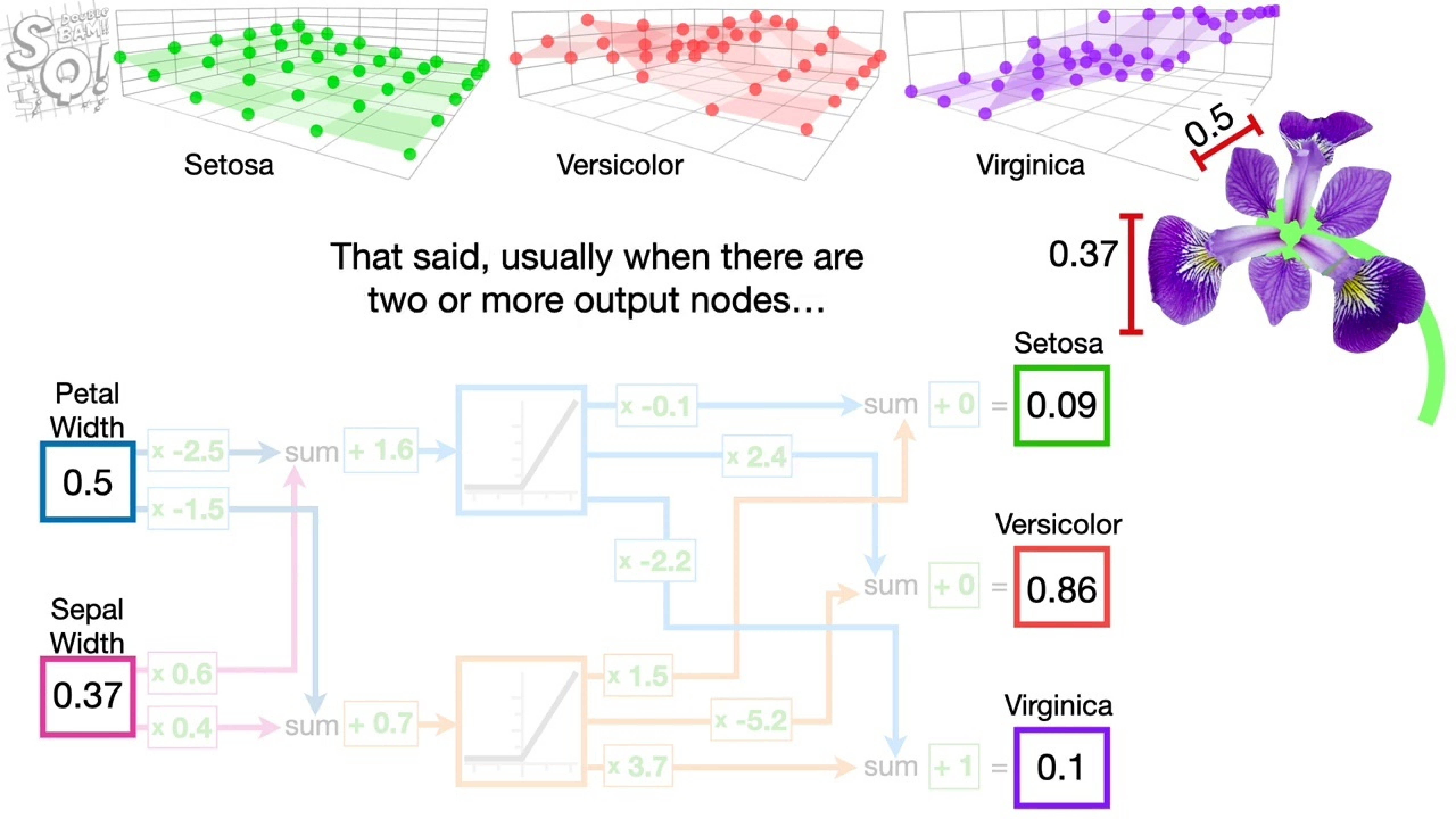




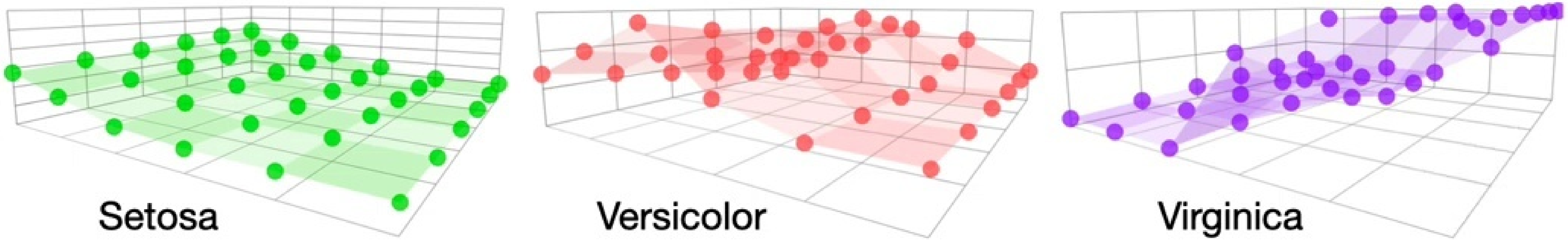
...and run the numbers through the **Neural Network**...



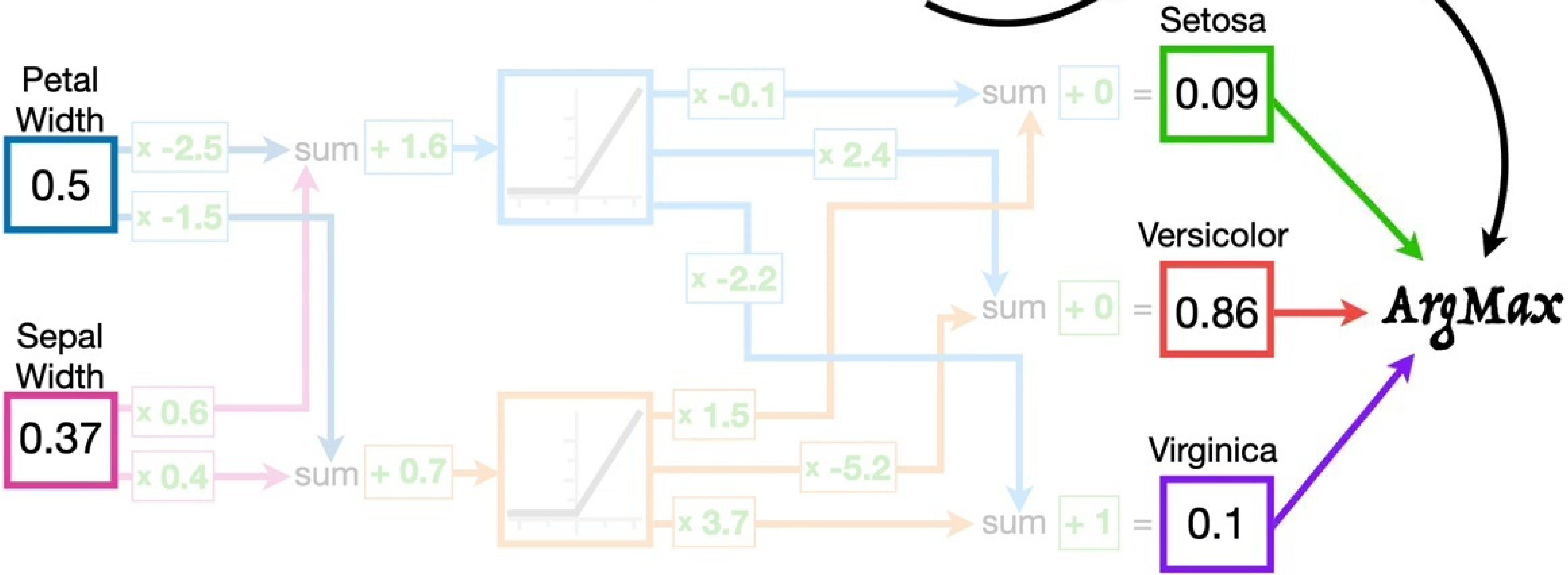

Setosa
Versicolor
Virginica
0.5
0.37
...and predict that this **Iris** is **Versicolor**,
because that output value, **0.86**, is closest to **1**.
Petal Width
0.5
x -2.5
x -1.5
sum
+ 1.6
Sepal Width
0.37
x 0.6
x 0.4
sum
+ 0.7
x -0.1
x 2.4
x -2.2
x 1.5
x -5.2
x 3.7
sum
+ 0
=
Setosa
0.09
sum
+ 0
=
Versicolor
0.86
sum
+ 1
=
Virginica
0.1

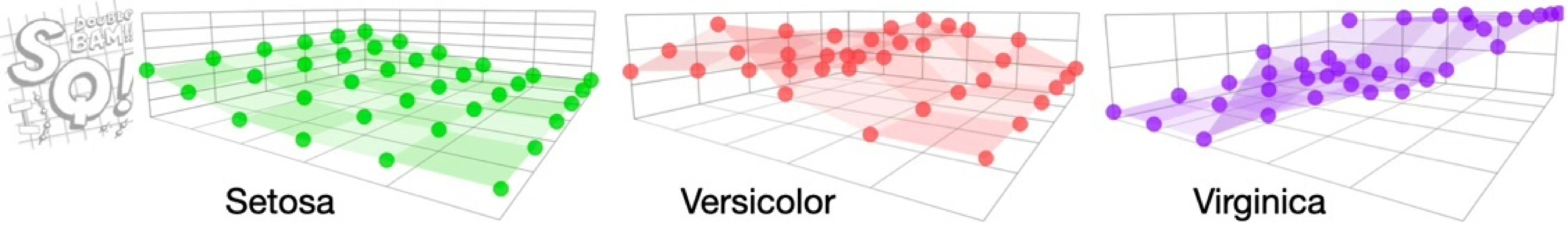


Setosa
Versicolor
Virginica
0.5
0.37
That said, usually when there are two or more output nodes…
Petal Width
0.5
Sepal Width
0.37
x -2.5
x -1.5
x 0.6
x 0.4
sum
+ 1.6
sum
+ 0.7
x -0.1
x 2.4
x -2.2
x 1.5
x -5.2
x 3.7
sum
+ 0
=
Setosa
0.09
sum
+ 0
=
Versicolor
0.86
sum
+ 1
=
Virginica
0.1



…the output values are sent to either something called **ArgMax**…





…or something called **SoftMax** before a final decision is made.