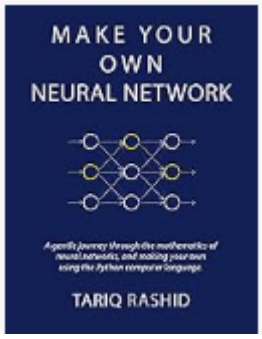
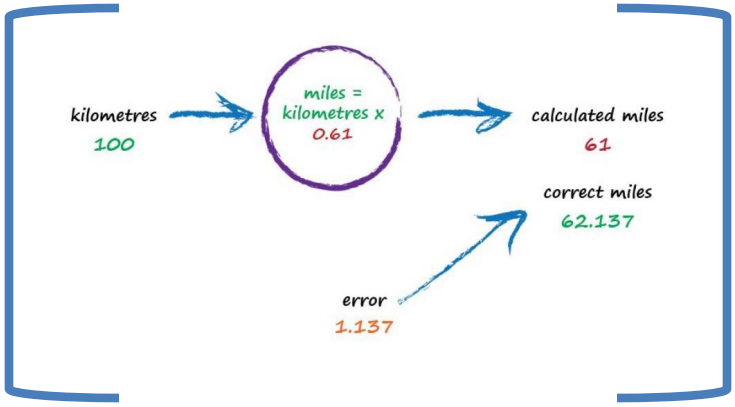
“SIMPLE” EXPLANATION: HOW A NEURAL NET LEARNS TO SOLVE A “CLASSIFICATION” PROBLEM

(From “Make your Own Neural Network”, T. Rashid, 2016)

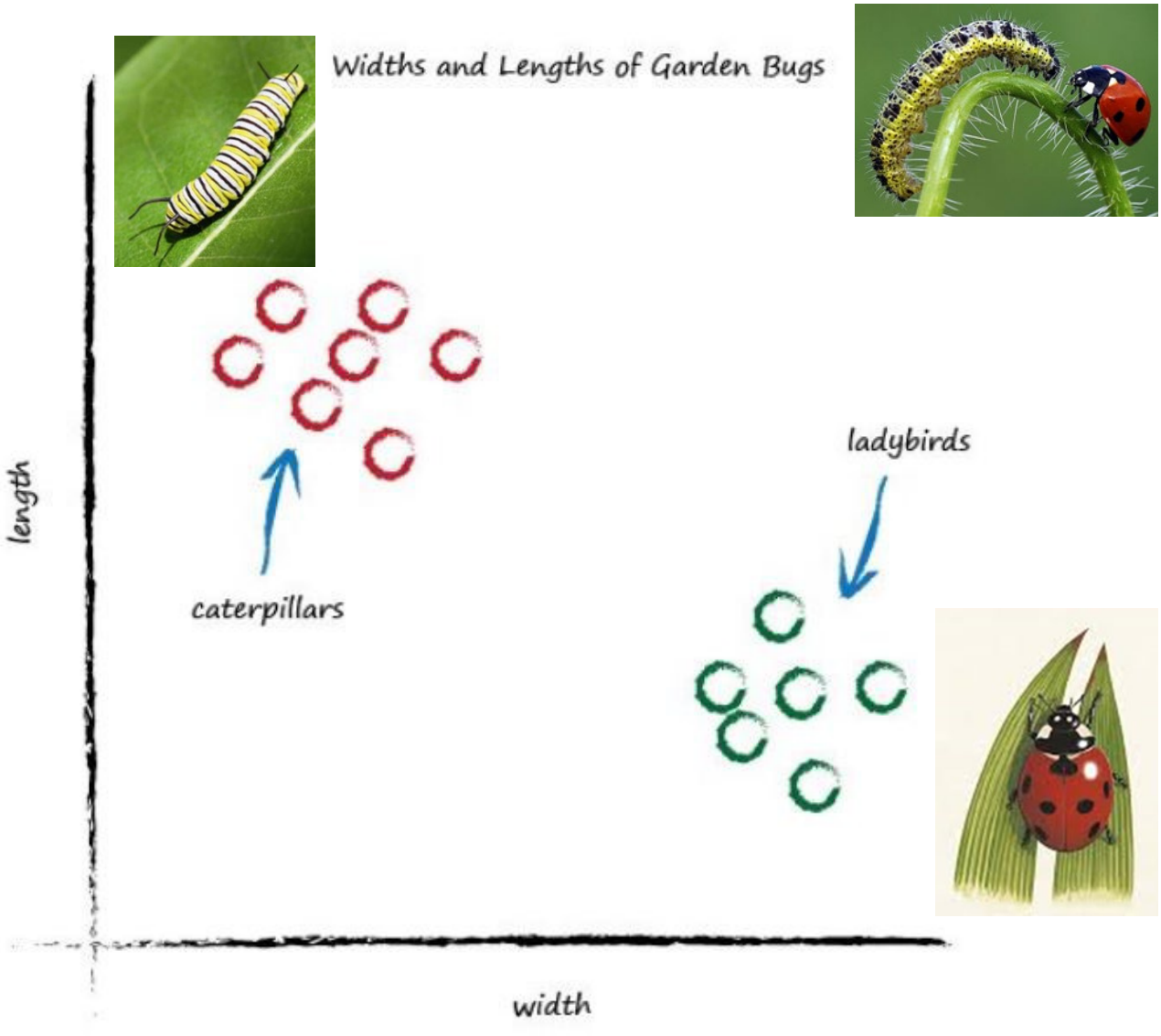




Classifying is Not Very Different from Predicting *(Regression)*

We called the above simple machine a **predictor**, because it takes an input and makes a prediction of what the output should be. We refined that prediction by adjusting an internal parameter, informed by the error we saw when comparing with a known-true example.

Now look at the following graph showing the measured widths and lengths of garden bugs.

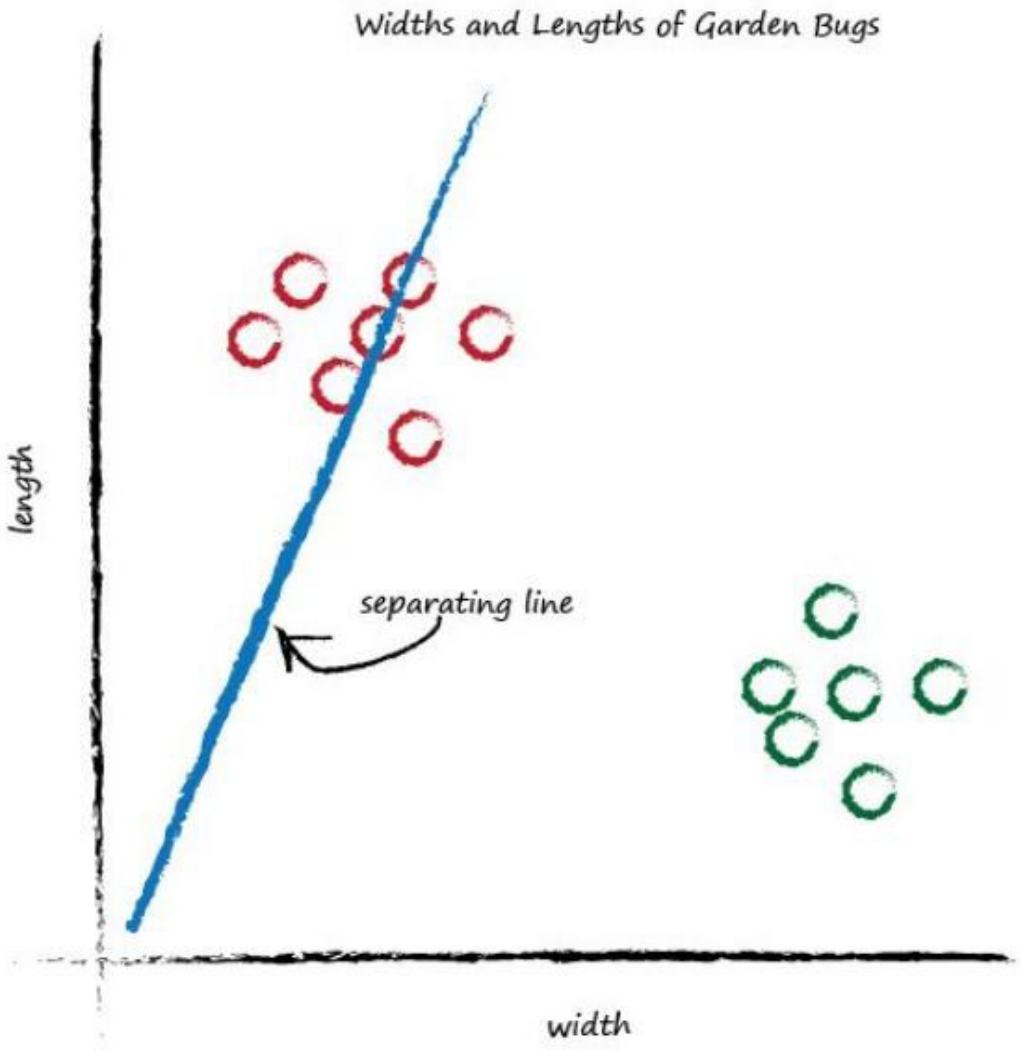


We may propose to “draw a line” in the plot visualizing the 2 characteristics (“FEATURES”) representing the insects to SEPARATE THE CLASSES, i.e., to implement a CLASSIFIER (We will consider that a bug whose representation is “above” the line will be considered a CATERPILLAR. If the representation of the bug is below the line, we will consider it to be a LADYBIRD:

As in the previous situation, we propose that the separating line (“DECISION BOUNDARY”) will go through the origin. Therefore, it would be represented by

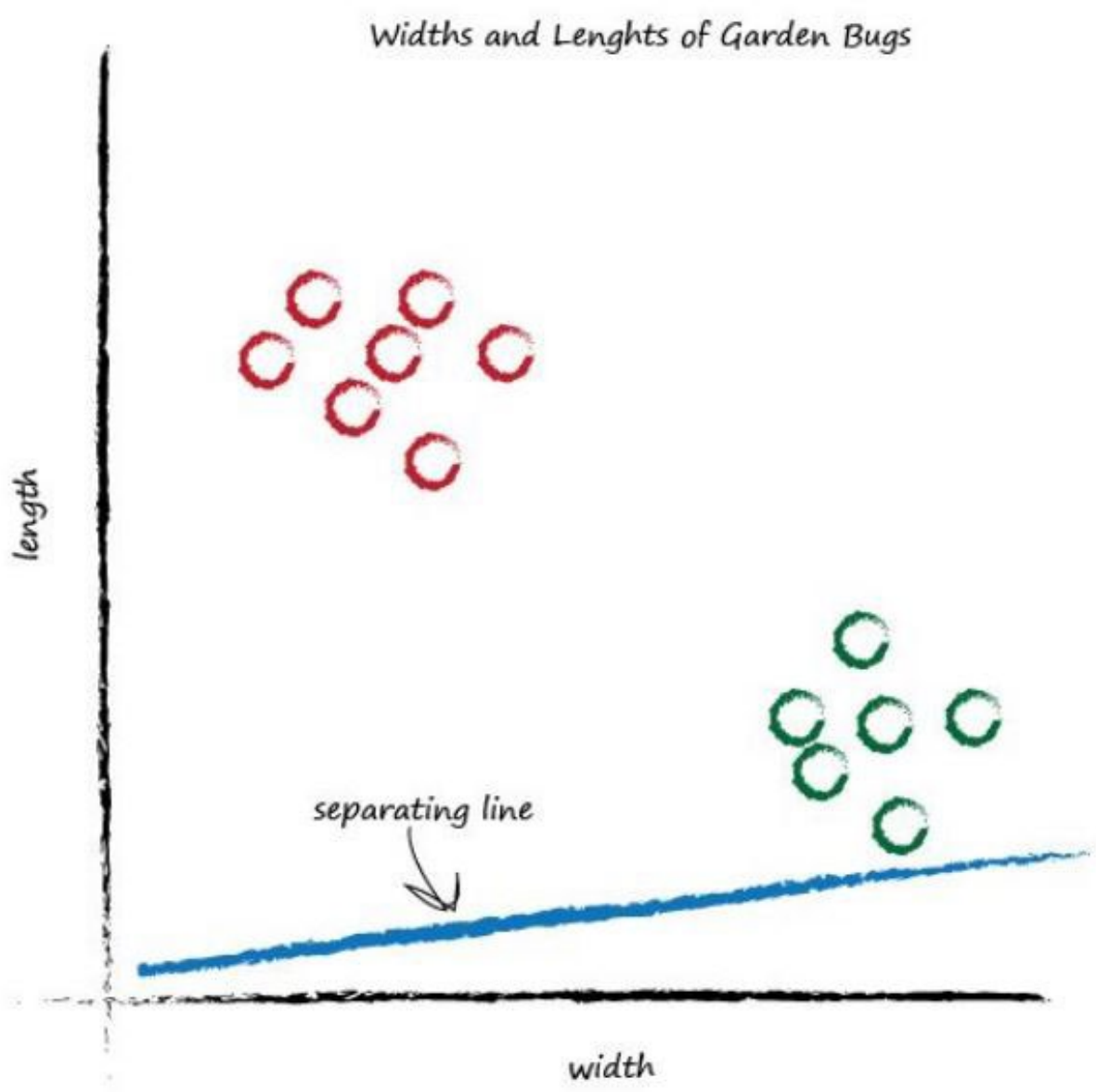
Y = A x …… (length) = A (width).

What happens if we place a straight line over that plot?



We can’t use the line in the same way we did before - to convert one number (kilometres) into another (miles), but perhaps we can use the line to separate different kinds of things.

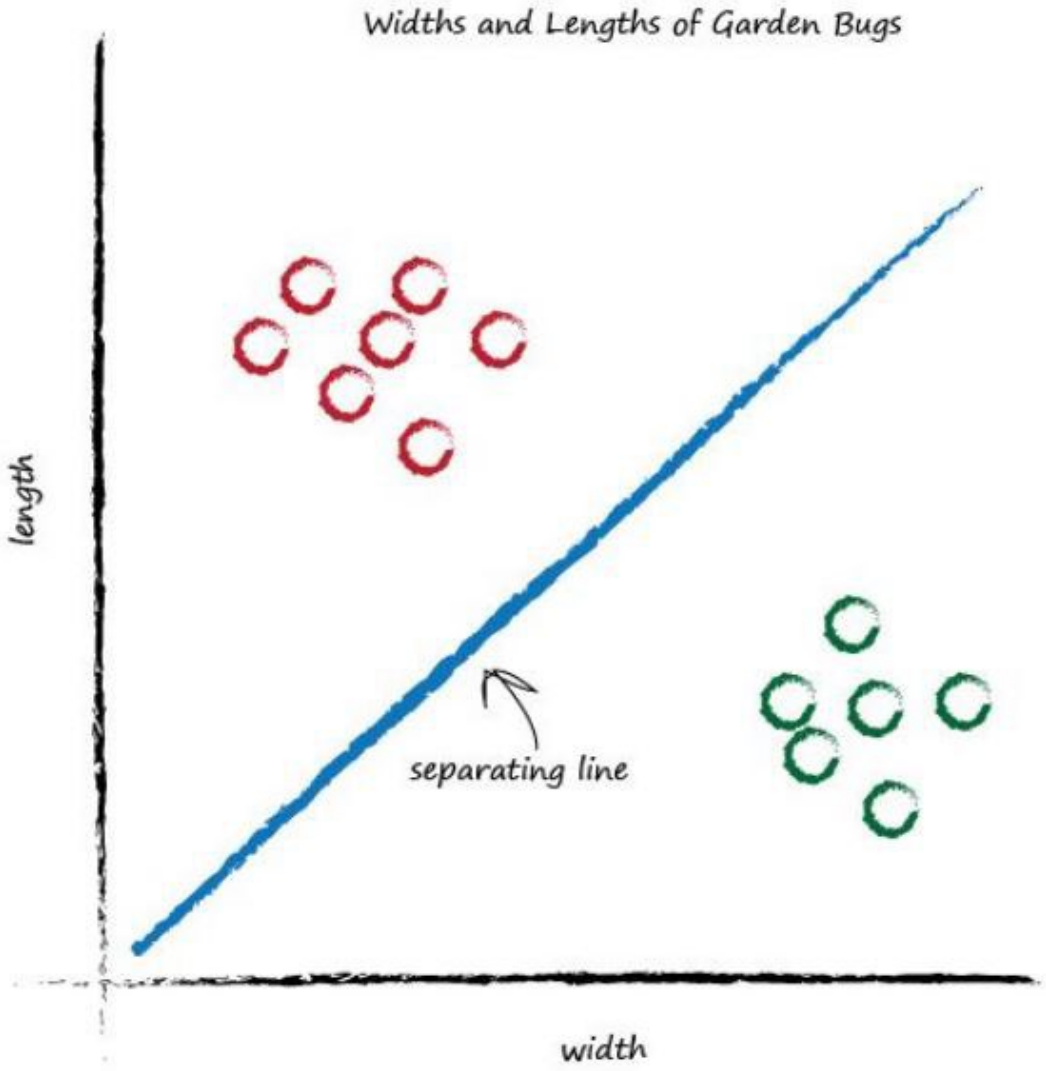
If we start again with a line specified “at random” (e.g., A = 0.5) we may not obtain a correct classification:



This time the line is even less useful! It doesn’t separate the two kinds of bugs at all.

Let’s have another go:

In this very, very simple case, we can draw a “correct” line “by eye” (and using our knowledge of geometry we could find out what was the SLOPE (A) of the line that we draw). BUT ... Could we use AN ITERATIVE “ADAPTATION” (or “NUDGING”) PROCESS TO FIND OUT THE SLOPE OF “A GOOD LINE” ?? – [Hint: that is what a neural network will be doing]



That’s much better! This line neatly separates caterpillars from ladybirds. We can now use this line as a **classifier** of bugs.

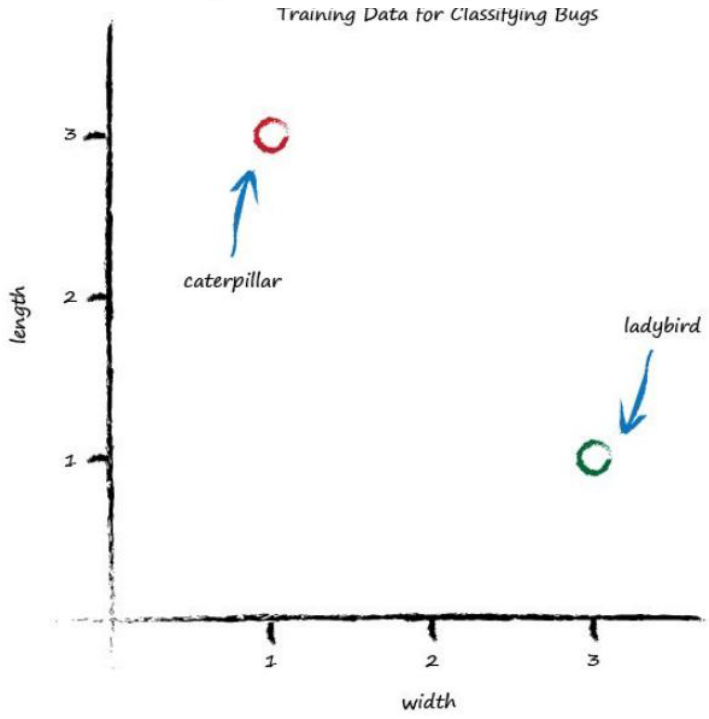
AS IN THE PREVIOUS EXAMPLE, we will PROGRESSIVELY ADJUST THE PARAMETER (slope A) OF THE "DECISION BOUNDARY" BY CONSIDERING EXAMPLES FOR WHICH WE KNOW THE CORRECT CLASSIFICATION, and trying to make adjustments that will allow our progressively adjusted line to produce correct results.

NOMENCLATURE:

Words used in this example:	Words used in Neural Networks:
Example (each of the 2 insects represented in the table)	SAMPLES, PATTERNS, INPUT VECTORS
Bug Characteristics: width, length	FEATURES
Line	DECISION BOUNDARY (later will be a *"hyperplane")*
Equation of the line:	MODEL
Slope, A	ADJUSTABLE PARAMETER ("weights& biases")
Process of progressively modifying the parameter(s)	TRAINING, LEARNING PROCESS, ADAPTATION, MODEL "FITTING"

In this case we will also use 2 examples:

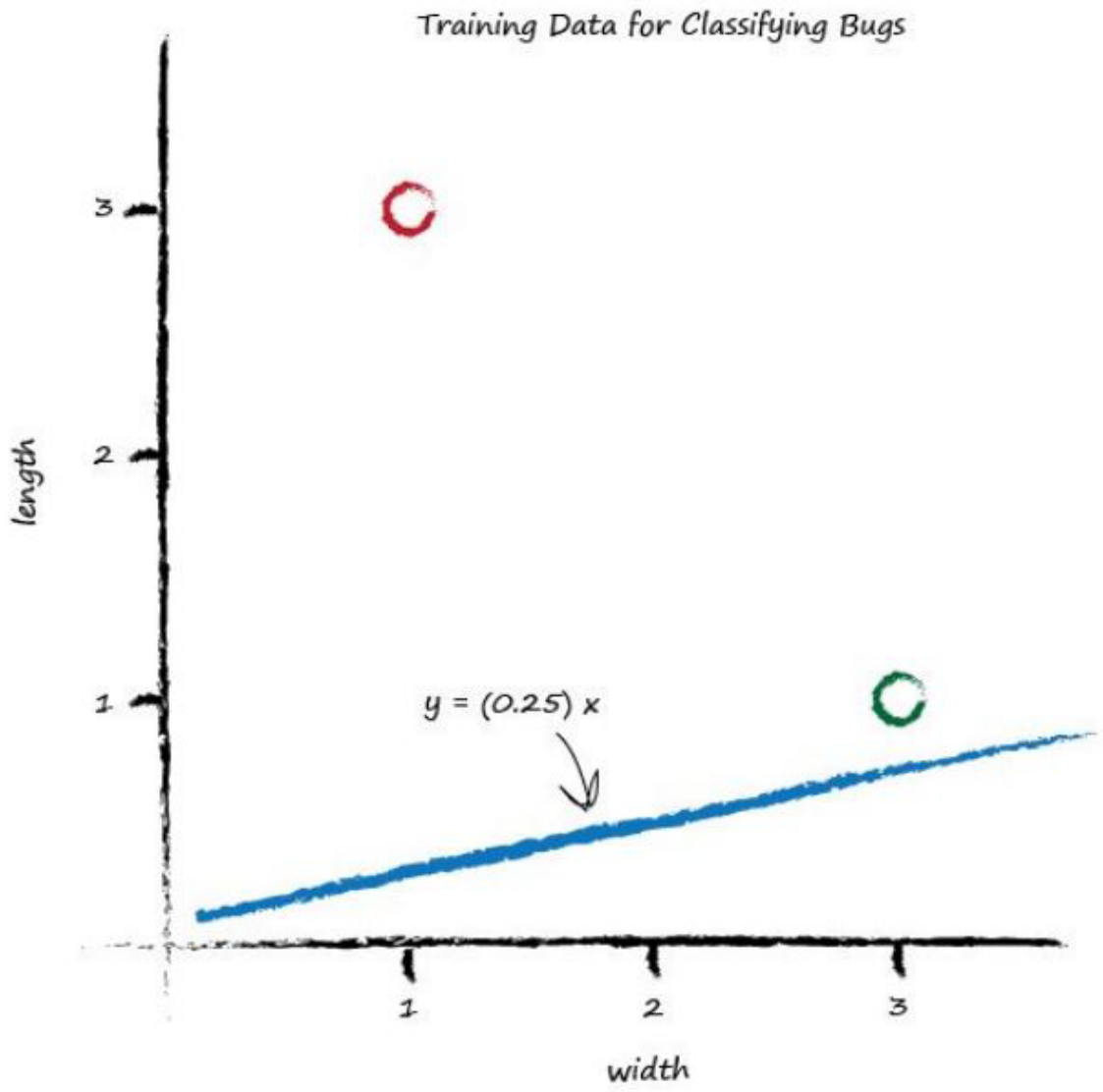
Example	Width	Length	Bug
1	3.0	1.0	ladybird
2	1.0	3.0	caterpillar



Let's start with a random dividing line, just to get started somewhere. Looking back at our kilometres to miles predictor, we had a linear function whose parameter we adjusted. We can do the same here, because the dividing line is a straight line:

Initial (“random”) attempt: A = 0.25 :

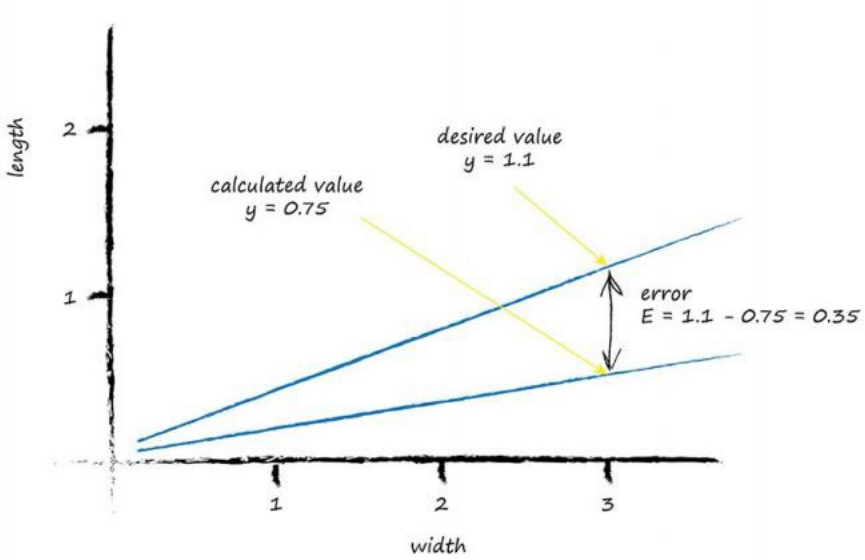
Let’s go for **A = 0.25** to get started. The dividing line is **y = 0.25x**. Let’s plot this line on the same plot of training data to see what it looks like:



Well, we can see that the line **y = 0.25x** isn’t a good classifier already without the need to do any calculations. The line doesn’t divide the two types of bug. We can’t say “if the bug is above the line then it is a caterpillar” because the ladybird is above the line too.

LET'S ATTEMPT TO CHANGE "A" TO GET A BETTER RESULT … BUT BY HOW MUCH?

IF WE PROPOSE A Y COORDINATE VALUE THAT WILL BE "CORRECT" FOR THE LADYBIRD EXAMPLE (E.G., A "TARGET" t = 1.1 , WHICH WILL BE 'ABOVE' 1.0) , THEN WE CAN QUANTIFY HOW "FAR OFF" WE ARE, I.E., AN ERROR, "E"

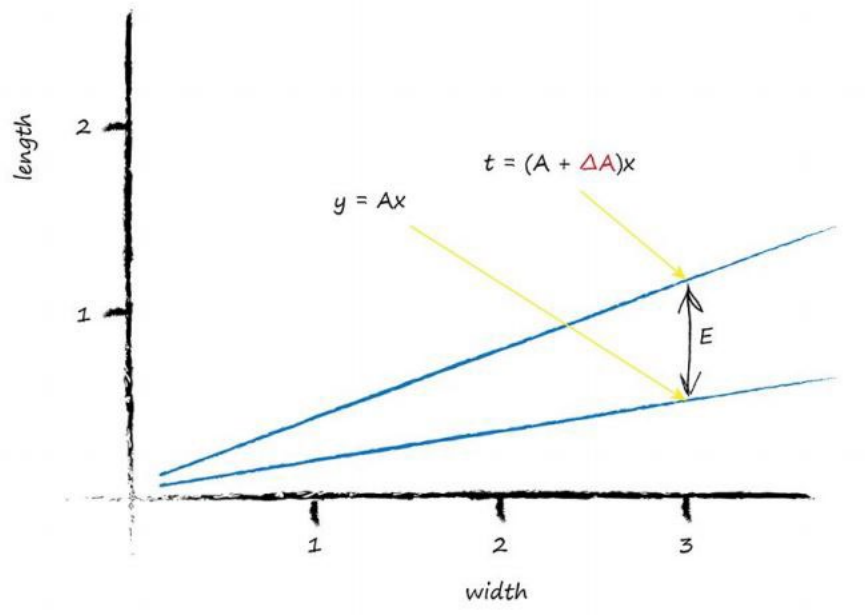


Now, what do we do with this **E** to guide us to a better refined parameter **A**? That's the important question.

SO, WHAT WE WANT IS TO CHANGE THE SLOPE BY AN AMOUNT ΔA, SUCH THAT:

$t = (A + \Delta A)x$…… THIS MEANS THAT $E = t - y = (A + \Delta A)x - (A\,x) = \Delta A\ x$…… i.e., $\mathbf{\Delta A = E/x}$

This tells me that I MUST ADJUST THE SLOPE, A, BY AN AMOUNT PROPORTIONAL TO THE ERROR THAT I OBSERVE, and this will help me towards eliminating the error, i.e., towards "hitting the target"



IF WE PROCEED ACCORDING TO THIS "CONCEPT"

Initial A = 0.25

"PRESENT" "SAMPLE 1" (ladybird) with a "TARGET" = 1.1 : : x = 3 ….y = (0.25)(3) = 0.75 ; E = 1.1 – 0.75 = 0.35

So, lets increase A: $A = A + \Delta A = A + (E/x) = 0.25 + (0.35/3) = 0.366$

< TRY AGAIN>

"PRESENT" "SAMPLE 1" (ladybird) with a "TARGET" = 1.1 : : x = 3 ….y = (0.36)(3) = 1.1 ; E = 1.1 – 1.1 = 0

(Great!)

But: What happen when we NOW TRY with "SAMPLE 2" (with a target t = 2.9) ??

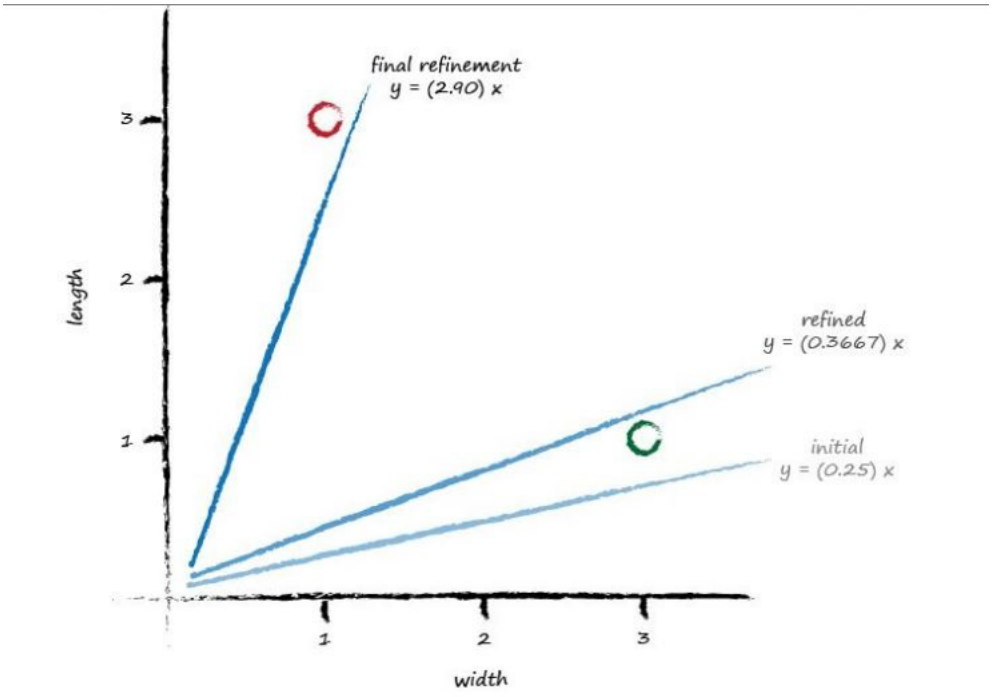
"PRESENT" "SAMPLE 2" (caterpillar) with a "TARGET" = 2.9 : x = 1 ….y = (0.36)(1) = 0.36 ; E = 2.9 – 0.36 = 2.53

So, lets increase A: $A = A + \Delta A = A + (E/x) = 0.36 + (2.53/1) = 2.9$

< TRY AGAIN>

"PRESENT" "SAMPLE 2" (caterpillar) with a "TARGET" = 2.9 : : x = 1 ….y = (2.9)(1) = 2.9; E = 2.9 – 2.9 = 0

We got a value of y that matches our proposed target! This seems good at first, … but THEN WE REALIZE THAT OUR METHOD IS ADJUSTING THE SLOPE TO FULLY MATCH THE "CURRENT TARGET" and SEEMS TO DISREGARD ADJUSTMENTS ACHIEVED WHEN WE PRESENTED PREVIOUS SAMPLES …mmm .. *that aspect* is not good.



Wait! What's happened! Looking at that plot, we don't seem to have improved the slope in the way we had hoped. It hasn't divided neatly the region between ladybirds and caterpillars.

Well, we got what we asked for. The line updates to give each desired value for y.

What's wrong with that? Well, if we keep doing this, updating for each training data example, all we get is that the final update simply matches the last training example closely. We might as well have not bothered with all previous training examples. In effect we are throwing away any learning that previous training examples might gives us and just learning from the last one.

How do we fix this? … *We MODERATE the changes (Do not apply them FULLY)*

TO “MODERATE THE CHANGES”, FOR EXAMPLE, LET’s APPLY ONLY 50% (L = 0.5 = “LEARNING RATE”) OF the “INITIALLY SUGGESTED” correction $\Delta A = E / x$. THAT IS, WE WILL MODIFY THE SLOPE EVERY TIME AS:

$$\Delta A = (E / x)\,(0.5)\ , \quad \text{Then:}$$

Initial A = 0.25

“PRESENT SAMPLE 1-ladybird” with a “TARGET” = 1.1 : : x = 3 ….y = (0.25)(3) = 0.75 ; E = 1.1 – 0.75 = 0.35

So, lets increase A: $A = A + \Delta A = A + (1/2)(E/x) = 0.25 + (0.5)(0.35/3) = 0.3083$

< TRY AGAIN>

“PRESENT SAMPLE 1-ladybird” w/ “TARGET” = 1.1 : x = 3 ….y = (0.3083)(3) = 0.9249 ; E = 1.1 – 0.9249 = 0.1751

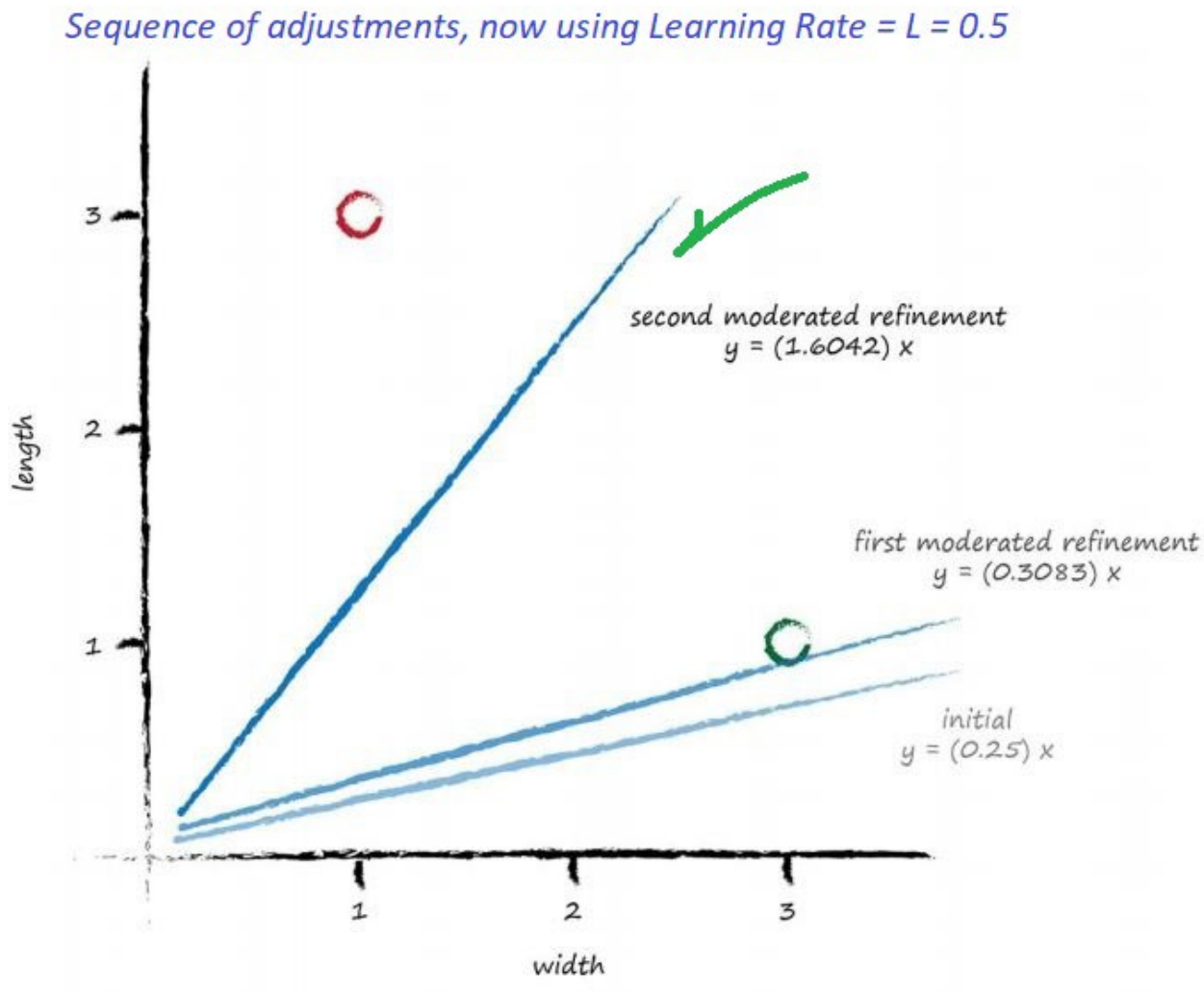
So, lets increase A: $A = A + \Delta A = A + (1/2)(E/x) = 0.3083 + (0.5)(0.1751/3) = 0.33748$

NOW “PRESENT SAMPLE 2”

(caterpillar) with a “TARGET” = 2.9 : x = 1 ….y = (0.33748)(1) = 0.33748 ; E = 2.9 – 0.33748 = 2.5625

So, lets increase A: $A = A + \Delta A = A + (1/2)(E/x) = 0.33748 + (0.5)(2.5625/1) = 1.6$

With this last slope we have a line that divides the 2 “samples” “nicely” (shown in the graph below, with a green checkmark).



This is really good!