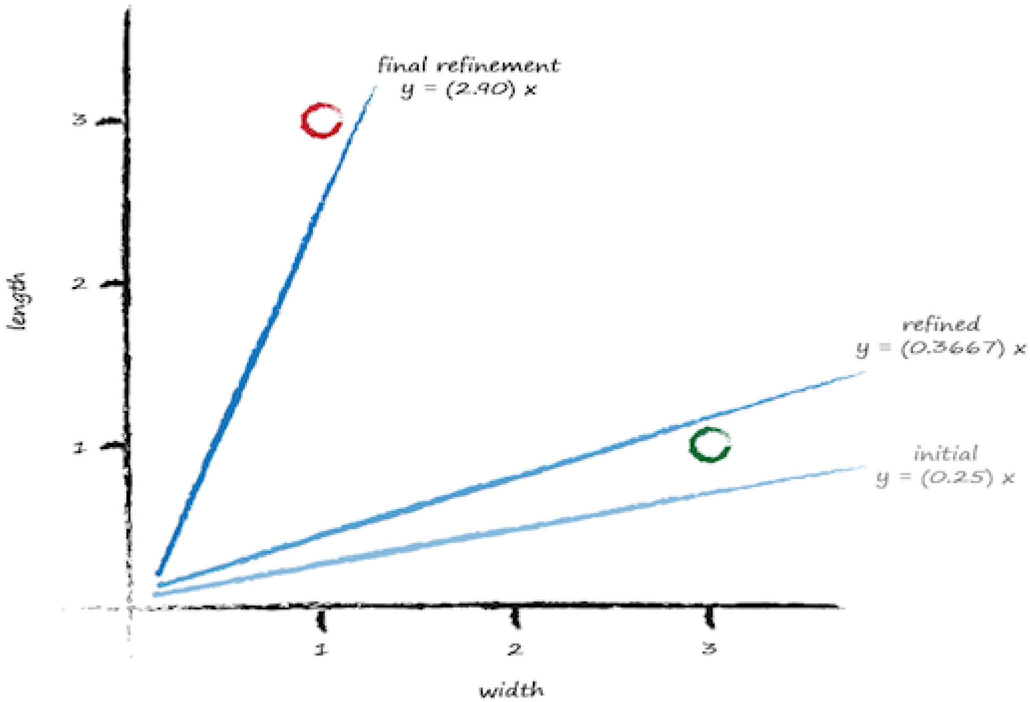
Setting up Learning Rate in Training Classifier

In this lesson, we will find out the use of learning rate while training a classifier.



How do we fix this problem?

Easy! And this is an important idea in machine learning. We *moderate* the updates. That is, we calm them down a bit. Instead of jumping enthusiastically to each new **A**, we take a fraction of the change Δ**A**, not all of it. This way we move in the direction that the training example suggests, but do so slightly cautiously, keeping some of the previous value which was arrived at through potentially many previous training iterations. We saw this idea of moderating our refinements before — with the simpler kilometers to miles predictor, where we nudged the parameter *c* as a fraction of the actual error.

This moderation has another very powerful and useful side effect. When the training data itself can't be trusted to be perfectly true and contains errors or noise, both of which are normal in real-world measurements, the moderation can dampen the impact of those errors or noise. It smooths them out. Ok let's rerun that, but this time we'll add a moderation into the update formula:

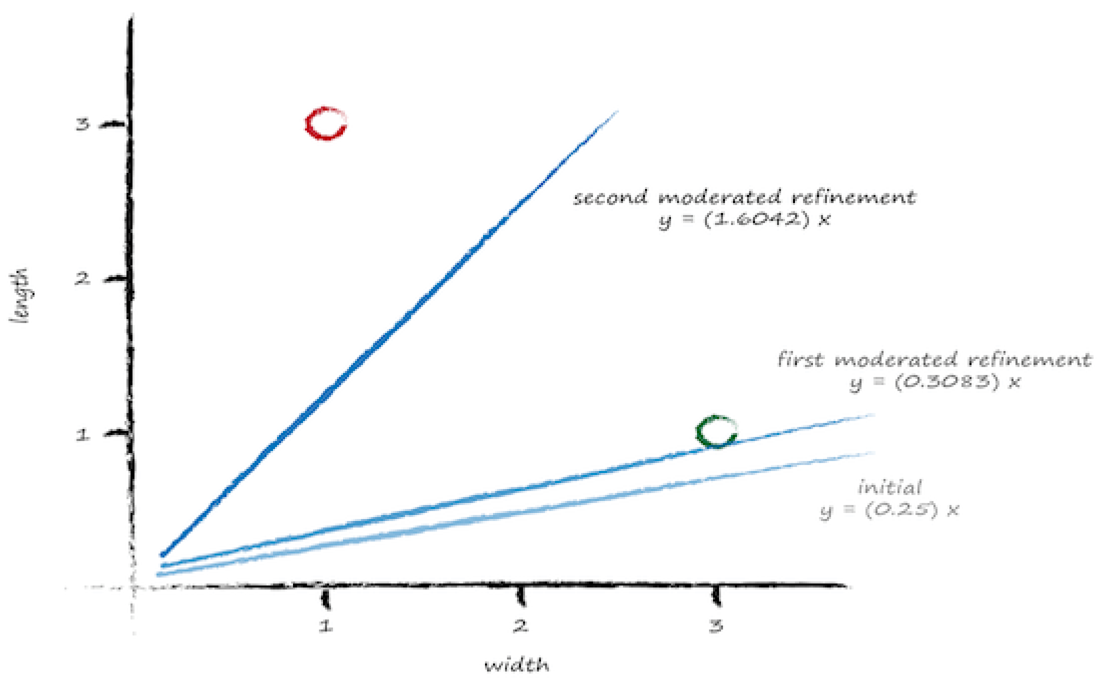
```
ΔA = L (E / x)
```

The moderating factor is often called a *learning rate*, and we've called it **L**. Let's pick $L = 0.5$ as a reasonable fraction just to get started. It simply means we only update half as much as would have done without moderation.

Running through that all again, we have an initial $A = 0.25$. The first training example gives us $y = 0.25 * 3.0 = 0.75$. The desired value of 1.1 gives us an error of 0.35. The ΔA = L (E/x) = 0.5 * 0.35 / 3.0 = 0.0583. The updated **A** is $0.25 + 0.0583 = 0.3083$.

Trying out this new A on the training example at $x = 3.0$ gives $y = 0.3083 * 3.0 = 0.9250$. The line now falls on the wrong side of the training example because it is below 1.1, but it's not a bad result if you consider it a first refinement step of many to come. It did move in the right direction away from the initial line.

Let's press on to the second training data example at $x = 1.0$. Using $A = 0.3083$ we have $y = 0.3083 * 1.0 = 0.3083$. The desired value was 2.9 so the error is $(2.9 - 0.3083) = 2.5917$. The ΔA = L (E / x) = 0.5 * 2.5917 / 1.0 = 1.2958. The even newer **A** is now $0.3083 + 1.2958 = 1.6042$. Let's visualize again the initial, improved and final line to see if moderating updates leads to a better dividing line between ladybird and caterpillar regions.



This is really good! Even with these two simple training examples, and a relatively simple update method using a moderating *learning rate*, we have very rapidly arrived at a good dividing line $y = Ax$ where **A** is 1.6042.

Let's not diminish what we've achieved. We have achieved an automated method of learning to classify from examples that is remarkably effective given the simplicity of the approach. Brilliant!