Draft Version

MACHINE LEARNING YEARNING

Technical Strategy for AI Engineers, In the Era of Deep Learning

ANDREW NG

deeplearning.ai

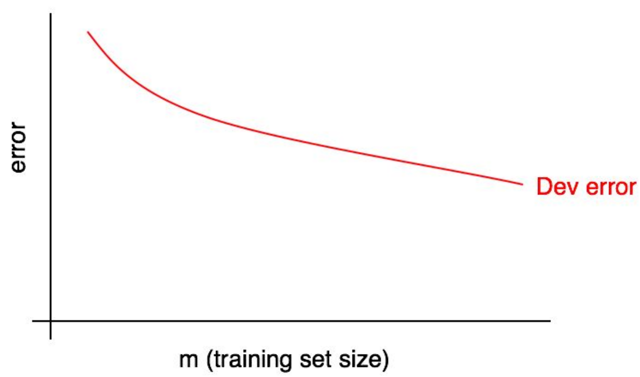
Machine Learning Yearning is a
deeplearning.ai project.

Learning curves

28 Diagnosing bias and variance: Learning curves

We've seen some ways to estimate how much error can be attributed to avoidable bias vs. variance. We did so by estimating the optimal error rate and computing the algorithm's training set and dev set errors. Let's discuss a technique that is even more informative: plotting a learning curve.

A learning curve plots your dev set error against the number of training examples. To plot it, you would run your algorithm using different training set sizes. For example, if you have 1,000 examples, you might train separate copies of the algorithm on 100, 200, 300, ..., 1000 examples. Then you could plot how dev set error varies with the training set size. Here is an example:



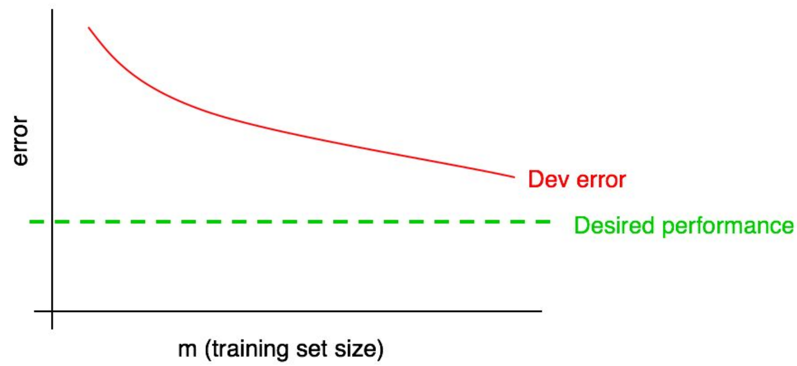
As the training set size increases, the dev set error should decrease.

We will often have some "desired error rate" that we hope our learning algorithm will eventually achieve. For example:

- If we hope for human-level performance, then the human error rate could be the "desired error rate."
- If our learning algorithm serves some product (such as delivering cat pictures), we might have an intuition about what level of performance is needed to give users a great experience.

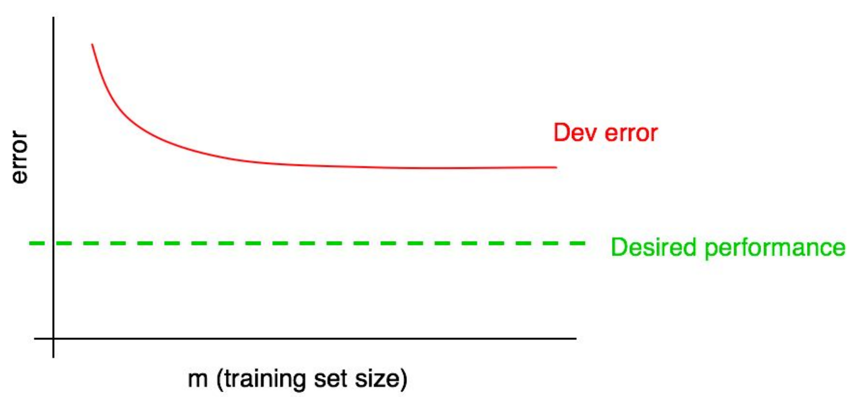
- If you have worked on a important application for a long time, then you might have intuition about how much more progress you can reasonably make in the next quarter/year.

Add the desired level of performance to your learning curve:



You can visually extrapolate the red "dev error" curve to guess how much closer you could get to the desired level of performance by adding more data. In the example above, it looks plausible that doubling the training set size might allow you to reach the desired performance.

But if the dev error curve has "plateaued" (i.e. flattened out), then you can immediately tell that adding more data won't get you to your goal:



Looking at the learning curve might therefore help you avoid spending months collecting twice as much training data, only to realize it does not help.

One downside of this process is that if you only look at the dev error curve, it can be hard to extrapolate and predict exactly where the red curve will go if you had more data. There is one additional plot that can help you estimate the impact of adding more data: the training error.

29 Plotting training error

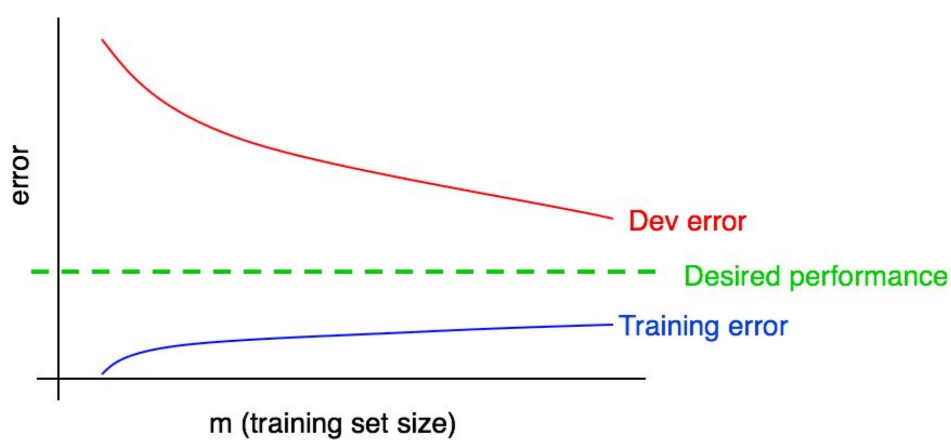
Your dev set (and test set) error should decrease as the training set size grows. But your training set error usually *increases* as the training set size grows.

Let's illustrate this effect with an example. Suppose your training set has only 2 examples: One cat image and one non-cat image. Then it is easy for the learning algorithms to "memorize" both examples in the training set, and get 0% training set error. Even if either or both of the training examples were mislabeled, it is still easy for the algorithm to memorize both labels.

Now suppose your training set has 100 examples. Perhaps even a few examples are mislabeled, or ambiguous—some images are very blurry, so even humans cannot tell if there is a cat. Perhaps the learning algorithm can still "memorize" most or all of the training set, but it is now harder to obtain 100% accuracy. By increasing the training set from 2 to 100 examples, you will find that the training set accuracy will drop slightly.

Finally, suppose your training set has 10,000 examples. In this case, it becomes even harder for the algorithm to perfectly fit all 10,000 examples, especially if some are ambiguous or mislabeled. Thus, your learning algorithm will do even worse on this training set.

Let's add a plot of training error to our earlier figures:

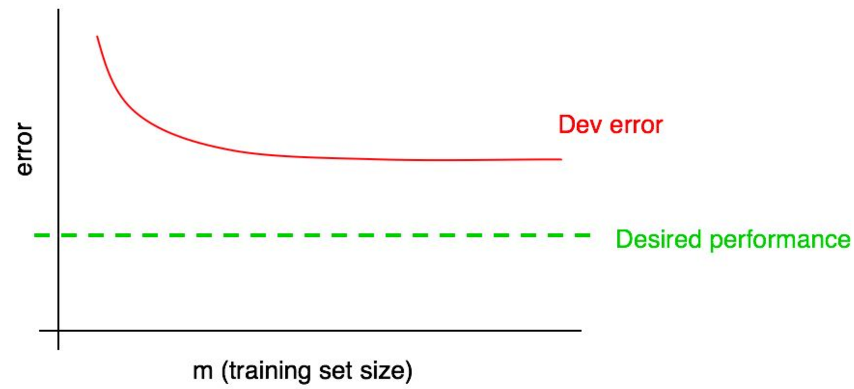


You can see that the blue "training error" curve increases with the size of the training set. Furthermore, your algorithm usually does better on the training set than on the dev set; thus the red dev error curve usually lies strictly above the blue training error curve.

Let's discuss next how to interpret these plots.

30 Interpreting learning curves: High bias

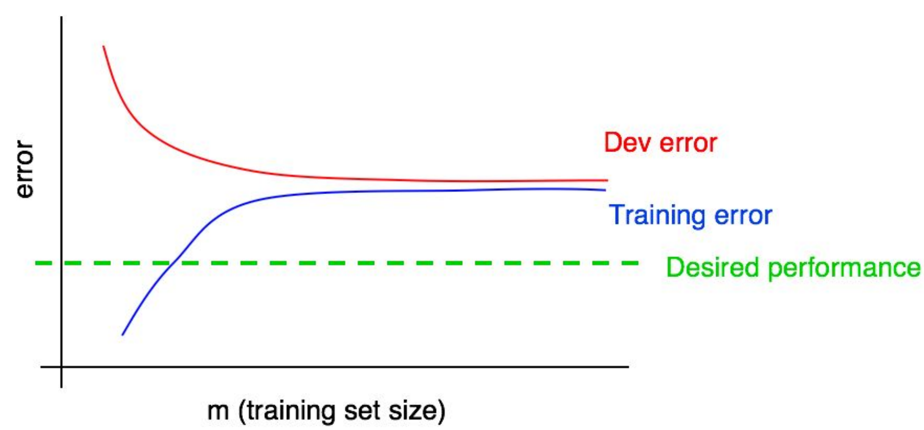
Suppose your dev error curve looks like this:



We previously said that, if your dev error curve plateaus, you are unlikely to achieve the desired performance just by adding data.

But it is hard to know exactly what an extrapolation of the red dev error curve will look like. If the dev set was small, you would be even less certain because the curves could be noisy.

Suppose we add the training error curve to this plot and get the following:



Now, you can be absolutely sure that adding more data will not, by itself, be sufficient. Why is that? Remember our two observations:

- As we add more training data, training error can only get worse. Thus, the blue training error curve can only stay the same or go higher, and thus it can only get further away from the (green line) level of desired performance.
- The red dev error curve is usually higher than the blue training error. Thus, there's almost no way that adding more data would allow the red dev error curve to drop down to the desired level of performance when even the training error is higher than the desired level of performance.

Examining both the dev error curve and the training error curve on the same plot allows us to more confidently extrapolate the dev error curve.

Suppose, for the sake of discussion, that the desired performance is our estimate of the optimal error rate. The figure above is then the standard "textbook" example of what a learning curve with high avoidable bias looks like: At the largest training set size—presumably corresponding to all the training data we have—there is a large gap between the training error and the desired performance, indicating large avoidable bias. Furthermore, the gap between the training and dev curves is small, indicating small variance.

Previously, we were measuring training and dev set error only at the rightmost point of this plot, which corresponds to using all the available training data. Plotting the full learning curve gives us a more comprehensive picture of the algorithms' performance on different training set sizes.