Single number evaluation metric

(SPEECH)
Whether you're tuning hyperparameters, or trying out different ideas for learning algorithms, or just trying out different options for building your machine learning system.

You'll find that your progress will be much faster if you have a single real number evaluation metric that lets you quickly tell if the new thing you just tried is working better or worse than your last idea.

So when teams are starting on a machine learning project, I often recommend that you set up a single real number evaluation metric for your problem.

Let's look at an example.

(DESCRIPTION)
Title, Setting up your goal. Single number evaluation metric.

Text, Using a single number evaluation metric. A figure shows three arrows in a circle, labeled Idea, Code, and Experiment. A table with headers labeled Classifier, Precision, and Recall. First Row, A, 95%, 90%. Second Row, B, 98%, 85%.

(SPEECH)
You've heard me say before that applied machine learning is a very empirical process.

We often have an idea, code it up, run the experiment to see how it did, and then use the outcome of the experiment to refine your ideas.

And then keep going around this loop as you keep on improving your algorithm.

So let's say for your classifier, you had previously built some classifier A.

And by changing the hyperparameters and the training sets or some other thing, you've now trained a new classifier, B.

So one reasonable way to evaluate the performance of your classifiers is to look at its precision and recall.

The exact details of what's precision and recall don't matter too much for this example.

But briefly, the definition of precision is, of the examples that your classifier recognizes as cats, What percentage actually are cats?

(DESCRIPTION)
Annotation on Precision column, of examples recognized as cats, what % actually are cats?

(SPEECH)
So if classifier A has 95% precision, this means that when classifier A says something is a cat, there's a 95% chance it really is a cat.

And recall is, of all the images that really are cats, what percentage were correctly recognized by your classifier?

So what percentage of actual cats, Are correctly recognized?

(DESCRIPTION)
Annotation on Recall column, what % of actual cats are correctly recognized?

(SPEECH)
So if classifier A is 90% recall, this means that of all of the images in, say, your dev sets that really are cats, classifier A accurately pulled out 90% of them.

So don't worry too much about the definitions of precision and recall.

It turns out that there's often a tradeoff between precision and recall, and you care about both.

You want that, when the classifier says something is a cat, there's a high chance it really is a cat.

But of all the images that are cats, you also want it to pull a large fraction of them as cats.

So it might be reasonable to try to evaluate the classifiers in terms of its precision and its recall.

The problem with using precision recall as your evaluation metric is that if classifier A does better on recall, which it does here, the classifier B does better on precision, then you're not sure which classifier is better.

And if you're trying out a lot of different ideas, a lot of different hyperparameters, you want to rather quickly try out not just two classifiers, but maybe a dozen classifiers and quickly pick out the, quote, best ones, so you can keep on iterating from there.

And with two evaluation metrics, it is difficult to know how to quickly pick one of the two or quickly pick one of the ten.

So what I recommend is rather than using two numbers, precision and recall, to pick a classifier, you just have to find a new evaluation metric that combines precision and recall.

In the machine learning literature, the standard way to combine precision and recall is something called an F1 score.

And the details of F1 score aren't too important, but informally, you can think of this as the average of precision, P, and recall, R.

Formally, the F1 score is defined by this formula, it's 2/ 1/P + 1/R.

And in mathematics, this function is called the harmonic mean of precision P and recall R.

But less formally, you can think of this as some way that averages precision and recall.

Only instead of taking the arithmetic mean, you take the harmonic mean, which is defined by this formula.

And it has some advantages in terms of trading off precision and recall.

But in this example, you can then see right away that classifier A has a better F1 score.

And assuming F1 score is a reasonable way to combine precision and recall, you can then quickly select classifier A over classifier B.

So what I found for a lot of machine learning teams is that having a well-defined dev set, which is how you're measuring precision and recall, plus a single number evaluation metric, sometimes I'll call it single row number.

Evaluation metric allows you to quickly tell if classifier A or classifier B is better, and therefore having a dev set plus single number evaluation metric distance to speed up iterating.

It speeds up this iterative process of improving your machine learning algorithm.

Let's look at another example.

(DESCRIPTION)
Text, Another Example. A table with the headings Algorithm, US, China, India, and Other. First Row, A, 3%, 7%, 5%, 9%. Second row, B, 5%, 6%, 5%, 10%.

(SPEECH)
Let's say you're building a cat app for cat lovers in four major geographies, the US, China, India, and other, the rest of the world.

And let's say that your two classifiers achieve different errors in data from these four different geographies.

So algorithm A achieves 3% error on pictures submitted by US users and so on.

(DESCRIPTION)
More rows are added to the table, Third row, C, 2%, 3%, 4%, 5%. Fourth Row, D, 5%, 8%, 7%, 2%. Fifth row, E, 4%,

5%, 2%, 4%. Sixth row, F, 7%, 11%, 8%, 12%. Another heading is added,

(SPEECH)
So it might be reasonable to keep track of how well your classifiers do in these different markets or these different geographies.

But by tracking four numbers, it's very difficult to look at these numbers and quickly decide if algorithm A or algorithm B is superior.

And if you're testing a lot of different classifiers, then it's just difficult to look at all these numbers and quickly pick one.

(DESCRIPTION)
Average column appears. A, 6%, B, 6.5%, C, 3.5%, D, 5.25%, E, 3.75%, F, 9.5%.

(SPEECH)
So what I recommend in this example is, in addition to tracking your performance in the four different geographies, to also compute the average.

And assuming that average performance is a reasonable single real number evaluation metric, by computing the average, you can quickly tell that it looks like algorithm C has a lowest average error.

And you might then go ahead with that one.

You have to pick an algorithm to keep on iterating from.

So your work load machine learning is often, you have an idea, you implement it try it out, and you want to know whether your idea helped.

So what was seen in this video is that having a single number evaluation metric can really improve your efficiency or the efficiency of your team in making those decisions.

Now we're not yet done with the discussion on how to effectively set up evaluation metrics.

In the next video, I'm going to share with you how to set up optimizing, as well as satisfying matrix.

So let's take a look at the next video.