Using fast.ai to sort text documents (part 2)

Now, let's try to use a model to distinguish whether a tweet is fact checkable or not fact checkable.

OK, we've made our language model and we saved the values from that language model. Inside this file called fine_tuned_enc. So we're all set for the next step, which is to build the classifier. And remember, the classifier is the model that does the sorting between whether a tweet is fact checkable or not fact checkable based on the language that is being used in the tweet. And we hope it understands language better because we just made a language model. OK. So for the next step, we need the classification data. And remember, the classification data was in the CSV called hand coded Austin tweets. That's the one that actually had both the tweet text as the text column and the checkable column was true or false, and that was the labeled column.

So we're going to grab that into this data_class. And now we're going to make a learner and that learner is going to be a text classifier learner. That's the task we have at hand and that's the kind of learner we're going to use. We're going to use our classification data from the hand coded tweets. This is our wiki text model. This is a hyper parameter that we're not going to go into right now, but just know that it's it's useful to have this value here. And then we're also going to load in that fine tuned encoder. Remember, that was the language model information. So we're going to load that into the learner as well. So all of that's being pulled together. We're going to use that classification data, those hand coded tweets to train the classifier. The difference between a fact checkable tweet and a not fact checkable tweet, and we're going to do that with a series of cycles. These are all with different learning rates, which I'm not going to dive into completely right now, but they are in the fast.ai documentation and in the fast.ai courses. And you can actually really pretty much rely on these rates for tasks like this.

Okay. At the end of all of that training, we now have an accuracy of 94 percent on our data. That's pretty great. From what we were giving it, it seems to be able to determine whether or not a tweet is fact checkable or not about 94 percent of the time. Let's see. Here's an example tweet. "Four states have two universities represented in the top 20 highest paid executives of public colleges. Texas has six." Ok, looks like there's some facts in there that could be checked. Let's see if the computer agrees. We've got the example here. And then we're going to do learn.predict on the example. And true is fact checkable, so it is determined that it is true that this statement is fact checkable. In fact, this number here is actually the percentage confidence that it has up to 99 percent confident we can actually open the black box a little bit and see what words the computer focused in on. Interestingly, universities represented seems to be something that is indicative of a fact checkable thing.

So that's my introduction to machine learning for journalists. Thank you for joining me on this journey, it's been a lot of fun. I really encourage you to check out the fast.ai class called Practical Machine Learning for Coders. With the background that you have in this class, I think that class might be something that you could get into, even if you're not a particularly good coder. Also, if you're feeling like a little bit more advanced or if you have a little bit of coding experience, check out the other notebook in the A.I. workshops folder, hh-finding-similar-documents. The code is a little bit more advanced, but it's how you might look for documents like something having to do with a topic like homelessness in a large

trove of documents. So that's another really good thing to explore. Otherwise, thank you for joining me for hands-on machine learning solutions for journalists.