**Transcript:**

**Taylor Linton:**
With open source. There's really two categories of benefits with using open source models, and it's really broken down into strategic and tactical reasons. To start off on the strategic side, it really comes down to ownership of an ip. Meta has done a great job commoditizing the models and other open source model trainers. So the fact that these companies can take their proprietary data and customize these models, that's a great opportunity to be really a competitive differentiator from their competition, having their own models that they own that are trained on their proprietary data. Another big reason on the strategic side, it really comes down to cost. And what I mean by that is sometimes your incentives aren't aligned when you're using a proprietary model because they charge based off of input data and output data. So the more value you get, the more charged. And so the fact that you can build an open source model and you're only paying for the commodity hardware, your goal is to get as much usage as possible, whether it's getting it in front of more customers or getting more AI assistance used from your employees, not only are you able to train your own model and keep it in your environment, but there's a lot of security benefits.

[Related: Role of Open Source in AI]

Customer trust is huge. Responsible AI is huge, and the fact that you can feel confident that you are sending your data to a model in your environment, it's a great opportunity to really de-risk it. Another part on the tactical side comes down to latency. If you don't own the model and you can't control the infrastructure that it's running on, there's many times when requests might take four or five seconds to get a response. So when you have ownership of the open source model, it allows you to make sure that you get that ultra low latency to support your requirements.
So that's one of the things that people really appreciate with open source models because you can download the model and run it in any environment you want to. So it could be their preferred cloud provider, it could be on-prem, it could be in their Nutanix environment too. But really it comes down to where do they want to run it? Where's their preferred environment? We often suggest our customers to go grab their own proprietary model, test out the use case, see how it performs. It offers a good benchmark to know roughly where AI is to be able to support the project from an accuracy standpoint. Now, once that POC has been built, that's where we work with them to say, okay, how should we build this with an open source approach? Now, open source AI is not a silver bullet. There's really trade-offs too. So when you start looking at cost, these models, they scale very well from a cost standpoint, open source models, and the reasoning is because you're paying for the commodity hardware that it's running on.

[Related: Enterprise IT Teams Jump Into AIOps]

The downside to that is if you don't have a large volume use case, it could require some expensive compute to host that model, and it might not make financial sense to go deploy a model for low usage. In those instances, it totally makes sense to be able to use a closed source proprietary model because you're only charged for input data and output data. So it can be really affordable in lower volume projects. But of course, as you do scale it out, that's where it gets very expensive and the scale tips pretty quickly. It's where it does make sense to deploy an open source model on your own hardware.

Related: IT Team Modifies Open Source NetBox to Help Manage Hybrid Multicloud]

So it's nearly impossible to try to automate the process of knowing which model to use and the best way to train or adjust the model for a use case. And so our engineers sit down with customers and look at what exactly are we trying to build, what type of data are we going to be sending to this model? What constraints are there, whether it's cost or latency. And so we sit down with them to understand what is the best open source model for this particular project. And then from there we can sit down and figure out how should we customize this to improve performance, whether it's speed, latency, or accuracy. These are all trade-offs that really you kind of have to pick one or the other. So we do our best to help them make the best decisions there.