Architecting the Cloud, part of the On Cloud Podcast

Mike Kavis, Managing Director, Deloitte Consulting LLP

Title:	**Leverage MLOps to scale AI/ML to the enterprise**
Description:	Artificial intelligence and machine learning (AI/ML) have spectacular potential to provide transformational insights. However, it's often difficult to scale AI/ML models to the enterprise. In this episode, Mike Kavis and guest, Deloitte's Sudi Bhattacharya, discuss the emerging discipline of MLOps and how it's helping organizations develop sound models and then scale those to enterprise production—thus closing the "train to production" gap for AI/ML. According to Sudi, MLOps uses a three-step approach: continuously training the models, developing a robust deployment framework (including the right team), and integrating the AI/ML models into the business processes. One caveat: the approach to AI/ML should be problem first, technology later.
Duration:	**00:33:14**

Operator:
This podcast is produced by Deloitte. The views and opinions expressed by podcast speakers and guests are solely their own and do not reflect the opinions of Deloitte. This podcast provides general information only and is not intended to constitute advice or services of any kind. For additional information about Deloitte, go to Deloitte.com/about. Welcome to Architecting the Cloud, part of the On Cloud Podcast, where we get real about Cloud Technology what works, what doesn't and why. Now here is your host, Mike Kavis.

Mike Kavis:
Hey, everyone, welcome back to the Architecting the Cloud Podcast, where we get real about cloud technology. I'm Mike Kavis, your host and chief cloud architect at Deloitte. We discuss all the hot topics around cloud computing, but most importantly, we talk with the people in the field that do the work. Speaking of people who do the work, my good friend Sudi Bhattacharya is here. He is the cloud machine learning leader and managing director here at Deloitte. I should probably introduce myself. Sudi, you're a repeat offender. We had you on maybe a year ago or so. Welcome back to the show. Just remind everyone who you are and what your background is, and the cool stuff that you work on today.

Sudi Bhattacharya:

Sure. Thank you, Mike. Thanks for the introduction. I feel quite fortunate to be working at the intersection of three cool technologies, so cloud, public cloud, big data. So, how are you actually going to build these big data systems on cloud? I also work in AI/ML systems, building AI/ML platforms and deployment platforms, and eventually API integration. I also work mainly in the financial services industry. There are a lot of interesting things happening in those areas.

Mike Kavis:
You also work at the intersection of the most confused buzzwords in our industry: cloud, big data, AI/ML. With that said, before we get into it, how would you define AI, because we're going to talk a lot about that, and how would you define ML, so it's a clear picture as we go into the rest of this conversation?

Sudi Bhattacharya:
A year ago, I was giving kind of my street definition and I will repeat that a bit. It's just an easy way to understand AI/ML. When we are talking about artificial intelligence systems, then we are actually talking about machines or computer programs trying to mimic us, trying to mimic things that we actually do quite well. For example, if I look at an image, the recognition of an image, if I'm looking at a piece of text, understanding the meaning of that text, I don't need any help to do that. Machines can't really do that well. But with the latest advance in deep learning in technology, we actually are getting better in each of these areas. So, that's artificial intelligence, mimicking humans.

Machine learning, on the other hand, deals with problems that we can do theoretically, but we can't do it in any meaningful time. So, if you give me lots and lots of data about, let's say, credit card usage and shopping behavior, and then want me to predict what is this particular user going to be buying the next time that person logs onto your website. So, it's all using algorithms, mathematical, statistical, probabilistic algorithms. So, one should be able to do that, but it would take me five months to do it. Machine learning models do that in seconds. So, machine learning models are essentially for predicting things or inferring things from data that are hard for human beings to do because of the calculational complexity involved.

So, that's kind of a simple, street way to think about it. If you look at these services, some of the cloud providers are also making them separate. They are talking about AI services and ML services. The border is a little blurry, but the definition I gave you is a good way to just think about it.

Mike Kavis:
Yeah. That's probably the clearest way I've ever seen or heard it defined. Cool. Thanks for that. So, the first question, and I get asked this a lot, is how do I even get started with artificial intelligence? It's this great big word. It can automate a lot of our business processes by letting machines do it, but it's not just that easy. How do we get started?

Sudi Bhattacharya:
That's a very important question, an interesting question. There are multiple viewpoints. But normally, one trend that you see is the moment one talks about AI/ML and getting started, once one focuses on model development.

So, let's say I'm thinking about an image recognition model. Basically, say somebody is in a car accident and the insurance company wants to assess the damage of the car automatically. I will take a video of the car and then send it, and then my machine learning model in the cloud will capture that video and actually infer the cost of repairing that car. Pretty cool. But there have been scenarios with this type of model, where people have spent a lot of time building these models, but they've never gotten value out of these models, because it very hard to take that model and integrate it into the business process of the claims business process that is supposed to be using that model to be more efficient. And usually, also, business processes are complex. They have a workflow. They have many decision points. Unless you actually have a very clear idea of where in the process you want to use AI/ML, you should not start modeling.

So, I think the recommendation that I would provide to customers, clients, is identify a business process where you believe efficiencies are needed, where there is lots of manual work and lots of decision making, manual decision making. So, once you have a clean map of that process, once you have identified the decision making points and if those decisions could be data-driven, they automatically are points for insertion of AI/ML models, because then you have data and you can train a model to mimic the human decision.

Then usually, also, it's not one model. One process might need four or five different kinds of models to make that all work and fit together. So, unless you have a clean understanding of the process that you are trying to enhance, you are not well-served to start on your AI journey.

Then once you've understood the process, understood the areas of enhancement, then you start looking at data, looking at feasibility, because AI models are probability. They may not be highly accurate when you start building these models, and you have to think about building their accuracy over time as well. So, that I think is the right approach, problem first, technology, even if it's AI/ML, later.

Mike Kavis:
You mean I shouldn't run around with a hammer looking for nails?

Sudi Bhattacharya:
Yeah, exactly right. I know how to model NLP and that's all I'm talking about. Exactly.

Mike Kavis:
I think that's a 40 or 50-year-old problem in IT actually. This kind of leads to the next questions, because we see a lot of experimentation, a lot of companies playing around with these technologies, getting smart, but we don't see a lot in production. My previous podcast that I did, I was talking to Julien Vehent about security and I asked him about AI in security. He basically said not ready for prime time yet. The same type of concept, "A lot of people are doing experiments, but I have yet to see where it's robust enough for me to use." His words there.

So, the same type of thing here. We see that just about every client we talk to has an experiment or a proof of concept going, but very few have production implementations of that. So, how do we bridge that gap between innovation and putting this stuff in production and operating it?

Sudi Bhattacharya:

That’s a great question. I am seeing very similar scenarios, a similar approach to AI/ML. Large organizations have hired robust teams of data scientists who are very smart, who are actually experimenting, innovating, getting some data, working with different cutting-edge models in many cases. But the moment you ask them, “How many models do you actually have in production enabling real business processes?” The number is astonishingly low. It’s really like 10; some would say 15 models. For an organization that’s multi-billion dollars.

But if you also look at an organization that has successfully embraced AI, without naming them – we all know who they are – they have thousands of models. Like I said, a business process needs many, many different models, AI models, ML models, maybe just KPIs to drive the process and make it more efficient. In a large organization, you would think that if it is completely mature in its lifecycle of AI usage, you are talking about thousands of models.

So, how do you go from experimentation and five models in production to a thousand models? There is an expression in this area, in this field, the gap between training and serving. So, you train a model. You get good results. But then how do you serve that model up in production, and how do you do that serving up at scale? Again, it goes back to our concepts and ideas that we are very familiar with in the cloud, that you need automation. You need pipelines. You need a methodology that actually makes these models deployable, so that this is not just on your laptop. You have gotten a little bit of data, trained data, maybe lots of data, but you didn’t really follow a process. Then, it becomes difficult to replicate it. It becomes difficult to version it. It’s very difficult to do it in production.

So, we are basically here talking about the discipline of machine learning operations, MLOps. You really need to follow the discipline to develop the model, train, validate, and finally, you are happy with the model. Then you need another infrastructure. The way we deploy models is very simple. We essentially “Dockerize” the model artifacts, expose endpoints, and then you have a deployed model.

So, I will actually give you an example of my conversation. Maybe three days ago I had this conversation, where people were talking to us about MLOps, and the client was actually telling me that all our previous attempts failed, and I was asking, “Give me one reason why it failed.” It failed because they just did what I described. They took a model, “Dockerized” it, deployed it, and they never scaled, because they didn’t have – the moment you just have one Docker container, it’s limited. Then what happens is people like your model. They start hitting that endpoint. The moment from two hits and then you go to 100 hits and 200 hits, the Docker container essentially fails. You cannot sustain that load.

So, you need that whole idea of autoscaling, that whole idea of container orchestration. You essentially need a Kubernetes infrastructure, which are all available. Every cloud provider does that, but some people are scared and there is this gap. So, data scientists don’t understand Kubernetes. IT is getting up and running in Kubernetes, but they don’t quite understand the workload of a machine learning model. So, that’s where the gap remains a gap between training and serving.

So, firstly, to summarize, you really need the discipline to train it in a robust way, following software lifecycle design best practices with proper pipelining, proper versioning, et cetera, testing. Then on the back of it, you need deployment infrastructure and discipline to do that. Thirdly, a deployed model gives you zero ROI unless people are using it. So, you also have to think upfront about how are you going to integrate that model into the process that is supposed to be benefiting from that model. So, that means APIs. That might mean many things. There are many ways to integrate. Usually, in a large corporation, you might need all of those patterns to do it right. So, all these three things have to come together to turn a large company into like a machine learning AI factory, where they can go from deploying two models every six months to, in two years, going to the thousand-model paradigm.

Mike Kavis:
I want to understand the scope of MLOps. I love all these something ops; “this ops,” “that ops.” A project I worked on last year was around the data scientist experience. There’s a lot of work that goes around until they get to the model. There’s a lot of pulling data and running jobs and doing all this stuff. So, we kind of built a platform on top of one of the providers to make that like one place to go to experience all that. They don’t have to save all these files on their hard drive. It’s all being done in the cloud. Then there’s, “Okay, now I have a model and I need to deploy that.” So, is MLOps all of that or is it from the point of now there’s a model. How do we get this model into production?

Sudi Bhattacharya:
It’s all of that. If you look any MLOps framework, it will talk about initial data pipelining that is needed for training, because that’s a crucial part, because large deep-learning models might need lots of data. There are performance efficiencies and then lots of experimentation, which means that you need to – machine learning, the difference is it’s not just code. It’s data also. If I have success with a trained model in Version 1, that means that I have code for Version 1 and I have data for Version 1. The next version might have a different set of data and different code. So, you have to track both.

Let’s say you deploy a model. What needs to happen to actually have a true production model deployment? The first thing is we say, “Okay, I have done continuous integration. I have done continuous delivery. But now I have to start thinking about continuous training.” So, that’s an additional thing that MLOps adds to the picture.

So, what is continuous training? Models are based on the data that has been used to train them, and that data goes stale. Imagine if I trained a model pre-COVID to predict something about buying patterns or behaviors. Once COVID has hit, that has changed, completely changed. There are major macroeconomics shocks that have happened. So, I need to go back and retrain my model with the current data. In many scenarios, that retraining may happen at a much faster pace. In some scenarios, it might be every two months, whatever. So, how do you do that? You actually have to automatically track your model drift, that your model is becoming stale.

How do you do that? Before you track drift, you have to actually track your model performance. Let’s say your model has not drifted. But is this model high-quality? Meaning your model is in production, people are using it. They’re getting predicted values. But how accurate are those values? How do you do that in an automated fashion? Because otherwise, it’s a project, and then you have to capture those and you have to compare it with what should have been the predicted value. So, model performance tracking. Then against the performance tracking, the model becoming stale, drift detection. Against drift detection, then you have to decide that, okay, now the drift is enough for me to retrain, and then push a train pipeline that goes and trains the model again and deploys the model back. So, that whole part is the backend of MLOps.

Then, there is another interesting aspect here. Usually, because models are probabilistic, and you are always improving the models when you are deploying the model – it's like an application. You rarely deploy just one version of the model. You do multiple deployments – sometimes you need to do multiple deployments because maybe you have one particular group of customers. You have trained your model for their data and you want a version of the model to just address their needs, et cetera.

So, how do you actually track that deployment? Then, you also have this idea of spiking deployments, right, to Cyber Monday and Black Friday. I think it's called the BF/CM problem. If your model is a sales-oriented model, that's when it crashes, because that's when people are there and you have to recommend in real-time work, and then you are not ready for that kind of scale. So, all these things together actually form the discipline of MLOps. I think it's a very rapidly evolving area, a lot of experimentation, a lot of research, a lot of methodology. Quite exciting, but I wouldn't say that we now have answers to all these questions. Early stages still.

Mike Kavis:
Let me ask you a question. A lot of companies have people dedicated – they call them – my least favorite term, DevOps engineers – but people who work on the pipeline. But the pipeline for models is very different. Does it require a different skillset to build those pipelines, or is it just the people who do that type of work just need training on machine learning, AI, and those concepts?

Sudi Bhattacharya:
My sense is – and there might be controversy in this – but my sense is that this is – obviously, it's not quite the same, but many things are similar. Many things are similar in the sense that a pipeline is a pipeline. It has steps. It has a workflow. It needs orchestration. It has dependencies that you need to manage through orchestration, et cetera. But there are specific things in a machine learning pipeline that makes it more performing. Basically, you are loading data and you are – let's just talk about a training pipeline and you are training the pipeline. Usually, you are training on GPUs, so hardware accelerated clusters. Then you have to be very aware of how to build this pipeline, so that you are not wasting time, that your GPUs are not wasted because they are loading data. So, you have to interweave your load of data and training. There are small, little architecture performance tweaks you can do.

But these things are based on fundamental concepts of computer science. There is nothing super-special about how to make these perform, so retraining is quite possible, but it does need to happen, because if you just ask a DevOps person without any idea, just, "Go build a training pipeline for a distributed training model," that's not fair. If that person is smart, they will probably figure it out, but I think some additional training would be very, very helpful in this context.

Mike Kavis:
So, along those lines, we have people with a role called machine learning engineers and we have data scientists. We were talking about Kubernetes and now we're talking about complex pipelines. In my mind, a data scientist shouldn't have to worry – they should just assume that someone is taking care of that, but a lot of times, they're having to get down into the technology, which is not their skillset. So, describe what's the difference between the role of a machine leaning engineer and a data scientist.

Sudi Bhattacharya:
Let's start with an AI application or an AI-enabled application. Let's say I have a process, a business process, where I am getting thousands and thousands of e-mails for my customer support, and I know that my response rate is directly related to customer satisfaction. If I could meaningfully respond to these e-mails quickly, I know that my customers are going to be happy.

So, normally, without any machine learning, AI, et cetera, these are very manual. So, somebody has to go and look at the e-mail and respond. There is no way to do it in a very efficient way. So, with AI, what people are doing is they are actually building NLP deep-learning models that will look at – they will learn from these e-mails, and then they will actually route it to appropriate people, route it to appropriate departments, sometimes start writing automated answers based on this. So, it significantly improves the response rate, the quality of response and, consequently, customer satisfaction.

So, if you talk about this entire system and say, "Okay, if I had the code of this entire system, how much of that is model development?" Typically – this is an average of course – but typically, the model part is maybe 15, 20 percent of the entire coding effort. Everything else that's code has nothing to do with machine learning. This is everything to do with pipelining, everything to do with DevOps, everything to do with APIs, integration that we do day in and day out, outside of machine learning in the AppDev world. So, if you see that, then you know that the data scientists essentially are highly skilled. They are usually PhDs or Masters in statistics. They have acquired some skills in data science. Sometimes the people are coming from other areas with a PhD in physics, or a PhD in some mathematics, and then learning all the things necessary for machine learning. They know how to develop models, but they don't know about cloud engineering. I mean they could work in the cloud, but they aren't necessarily super-aware of pipelines, et cetera. So, they are focused on that core, the heart of the problem that is the model development.

Now, an ML engineer is really everything else. I usually have now started using this terminology that it's not just an ML engineer that we need to make a machine learning project successful. We need a full-stack machine learning engineer, meaning that person is not a PhD. It's a technical person, a smart guy or girl, who knows end-to-end this process. So, they can actually automate the infrastructure, create deployment templates, CloudFormation, Terraform, whatever, to create the machine learning infrastructure on the cloud or the way to write APIs to consume it. They are actually model-savvy. They understand the models. They are not developing them, but they know what that model is doing. So, they are the ones that are actually putting this whole thing together into one single whole. So, that's the machine learning engineering discipline, the full-stack machine learning engineering discipline.

Then, what I would say is that also the general trend with these large companies like Google, Microsoft, and Amazon in their cloud strategy is to make machine learning so easy, in the sense that you don't really want or need PhDs to use them, because the whole idea is you want to democratize user machine learning. I should be able to code. You should be able to code, without understanding lots of theory of how – the mathematics behind deep learning should not be necessary for me to use it. So, all the APIs, all the new frameworks that are being developed are being developed with that in mind, that a smart developer can start building these models.

So, eventually, I think, there always will be, obviously, a space for the high-end data scientists, but the general direction if you really truly want to adopt AI/ML across your organization, you cannot hire hundreds and hundreds of data scientists. It's just too expensive. You want to hire some, whatever that number is, but the rest are really ML engineering that makes it real. So, the ML engineering makes machine learning real for an organization, and data scientists really develop the heart of that system. If you wanted a visual, that's one way to think about it.

Mike Kavis:
Let me ask you a follow-up question on that. Are the cloud providers also abstracting all the things you just described the ML engineers doing, or abstracting away the knowledge worker part of building models? But it still sounds like there's an incredible amount of plumbing people are doing. Are we abstracting that away, too?

Sudi Bhattacharya:
There are frameworks that are evolving. Let's say in GCP, they have this Kubeflow, which is an MLOps framework. The ML engineering is the one that is actually still evolving quite a bit. So, what are the best practices around testing a neural network? What are the best practices around detecting biases upfront? What are the best practices around all these pipelines? It's still, like I said, developing.

Eventually, it's going to be there, but we are at an early adopter stage in those areas. Many companies I see are still not MLOps-savvy. They are doing modeling. I am actually working with a client right now, where they have 50 data scientists. They are actively engaged in model development, but their ML engineering is not yet mature. So, that's the reality in most of the large companies.

Mike Kavis:
What about the operations of this? We talked about MLOps, but one running. I'll give you an example. Recently, I needed help on something, and I squinted and clicked on the ChatOps things, knowing that I was going to be talking to a bot. It asked me all kinds of questions. I filled it out. It came back and said, "It looks like this, this and this." Then it said, "Good-bye." I'm like, "What? Wait!" Someone has to be monitoring the performance, not the performance as far as how fast, but the actual, is it doing what it's supposed to be doing. So, is that another area that's untouched that we need to ramp up as well?

Sudi Bhattacharya:
I kind of briefly touched on that, model performance, the same with conversational AI. One of the key components of conversational AI is absolutely doing analytics on the responses of these systems. The same thing with models. You have to capture the prediction or inferences of your model. Then you have to compare it with the actual ground truth and continuously monitor that metric, seeing how well your model is doing. Now, the problem there is that sometimes you don't know the ground truth or you have to manually enter the ground truth. So, it's not completely automatable maybe, to begin with. So, those are things people are trying to figure out. But without monitoring, and almost on a regular basis, or on a real-time basis, you will not actually succeed for the exact reason that you are talking about.

These systems have developed to be self-learning, but self-learning doesn't mean they learn by themselves. You have to monitor and retrain them. So, there is an active intervention that is required. There are ops people, MLOps people. There are data scientists. If you have that ecosystem, the MLOps people do the monitoring, and then there has to be triggers. You can actually automate the training as well, but most of the systems I have seen, now there is some manual intervention that people actually look at, then decide, and then train. It's not completely automated.

Mike Kavis:
It sounds like we're very, very early days into this space. This space is extremely exciting. There's a lot of really hard use cases for this technology to solve. If we're talking about this three years from now, we'll probably have learned a lot from where we are today. What do you think on that?

Sudi Bhattacharya:
Absolutely. I think this is one of those areas where there is tremendous potential, but many of our clients are also really trying to understand where is the potential, because the problem is, this is not like AppDev. Let's say you have decided on a process automation. You select the piece of software. You implement the piece of software. Maybe it takes you a year, but after that your process is "automated," within quotes, right.

But AI is a long game. It is not something that will immediately give you value, because your models take time to converge. Initially, you may not have enough training data. There is also this continuous learning component. So, there is a finite amount of time that we'll need before you really start developing the skills, developing the process, and maturing the models that are going to be the heart of your system, and developing more models. So, absolutely, I think in three years from now. Now, we see a lot of excitement, but I don't know if we are going to be maturing, because the potential is so huge that in three years there are going to be new developments, absolutely.

For example, Google has a separate group called Quantum AI. So, they are taking ideas from quantum computing and merging with AI, and starting to look at classification problems, and some very specific things that can take advantage of quantum computing to get better. I don't yet understand it. I have to go back and kind of brush up my physics knowledge a little bit before I dive into that, but in three years that would be a good conversation to have, for sure.

Mike Kavis:
Yeah. Thanks for setting that expectation. All right. Sudi, it's always a pleasure talking with you. I know you're not a big Twitter person, which means you're probably happier than most of us who have to read all that stuff. But you are on LinkedIn, right?

Sudi Bhattacharya:
Yeah. I am on LinkedIn. I actually occasionally publish. I have a couple of posts. I publish articles in Deloitte On Cloud Blog and then I link it back in LinkedIn. Three articles have been recently published on machine learning, good things. They are kind of technical. I have avoided putting code in there, but they have technical concepts in them, so people can learn from those things, if they are interested.

Mike Kavis:
Thanks for that. Yeah, check those out. When I need to learn in this topic, I just lean on Sudi, so I don't have to go search too much. But really good content, check that out. That's it for today's episode of Architecting the Cloud. To learn more about Deloitte or read today's show notes, you can head over to www.deloittecloudpodcast.com. You can find more podcasts by me and my colleague, Dave Linthicum, just by searching for Deloitte On Cloud Podcasts on iTunes or wherever you get your podcasts. If you want to catch Sudi's blogs, you can also search on the On Cloud blog, and you can find those articles and

others by me and Dave. So, that's it for today. I'm your host, Mike Kavis. You can contact me directly at MKavis@Deloittel.com. I'm always running around on Twitter @madgreek65. That's it. Thanks for listening to our show. We'll catch you next time on Architecting the Cloud.

Operator:
Thank you for listening to Architecting the Cloud, part of the On Cloud Podcast with Mike Kavis. Connect with Mike on Twitter, LinkedIn and visit the Deloitte On Cloud blog at www.deloitte.com/us/deloitte-on-cloud-blog. Be sure to rate and review the show on your favorite podcast app.