Jordan Edwards: ML Engineering and DevOps on AzureML

72 MIN · YOUTUBE · HTTPS://WWW.YOUTUBE.COM/WATCH?V=XMPYLDAMGKA&LIST=PLWFLAA-F1PGPVZIWTKJJOMZ670OAVEWSP&INDEX
https://www.youtube.com/watch?v=XmpylDAmGkA&list=PLwFLAA-F1PgpVziWTKjjomz670oAVEWsp&index=3

SUMMARY

Jourdan Edwards, a principal program manager on the Azure Machine Learning team, discusses the evolution of machine learning (ML) DevOps and its integration into enterprise environments. He emphasizes the importance of treating ML as a software engineering discipline while addressing the unique challenges it presents, such as increased technical debt and the need for effective governance.

- *ML DevOps aims to streamline the machine learning lifecycle, enhancing collaboration among data scientists, engineers, and operations personnel.*
- *The maturity model for ML DevOps outlines stages from no ML ops to fully integrated systems, emphasizing the need for reproducibility and governance.*
- *Tools like Microsoft Seal for homomorphic encryption and White Noise for differential privacy are being developed to enhance data security and model fairness.*
- *The conversation highlights the importance of interpretability and the need for rigorous testing and governance in ML models, especially when they impact critical decisions.*
- *There is a growing recognition of the distinct roles of ML engineers and data scientists, with a focus on collaboration and shared accountability in model development.*
- *The integration of strong typing and schema definitions for model inputs and outputs is essential for improving the interoperability of ML systems.*
- *The future of ML involves addressing adversarial threats and ensuring responsible AI practices while maintaining the flexibility of data science workflows.*

welcome to the machine learning Street talk YouTube channel with me the X Microsoft tech lead dr. Tim scarf Conor shortened and Yannick Kilcher today is a special episode we have Jourdan Edwards principal program manager on the Azure machine learning team Jordan has almost single-handedly transformed the reputation of Azure ml into an enterprise ready credible platform for doing machine learning DevOps DevOps is all about people process and

platforms to enable continuous delivery of value to your end-users it's about accelerating the velocity of software deployments it's about taking all of the friction out of the process to help you be more entrepreneurial to help you fail fast to create data science labs and environments automatically with infrastructure as code if you view machine learning as a software engineering discipline there is a specialization of interests and expertise across the life cycle there are data scientists there are engineers

there are ops people there are governance people machine learning is the next generation of software

engineering now we shouldn't be hyperbolic about machine learning it's not really intelligence it's just a computer program with mo DevOps we can treat the machine learning lifecycle exactly the same way as we do for software engineering but there's a key difference machine learning produces a lot more technical debt in production than traditional software so there's a lot more things we need to do we need to

think about governance at the company level we need to think about the handshake between all the different personas and the software engine life cycle we need to think about ethics and fairness we need to think about machine learning engineering and testing in the interactive phase and the non interactive phase data scientists famously like to work in Jupiter notebooks which is antithetical to software engineering and for a long time I thought that the best solution was to

get the data scientist to behave like software engineers I've now been thoroughly disavowed of that persuasion figuring out ways to get the very different stakeholders in this process to work together effectively is one of the key challenges of ML DevOps I really enjoyed talking to Jordan Edwards a key driver to innovation and machine learning is the velocity of our iteration cycles autonomy of data scientists and being able to get models into production machine learning is just

like software but it produces more technical debt in production the models might make important decisions and therefore need to have rigorous testing and governance ml DevOps is about orchestrating the non interactive phase of software deployments for machine learning this is extremely hard to get right because of the different personas involved in the lifecycle of model deployments this is Formula one in the 1950s as you can see it was a manual error-prone process they were engineers hitting

parts of the car with hammers there was no monitoring and their automation this is what Formula one looks like today there are many different types of engineer working together effectively it's fast it's continuously monitored and improved the whole process has been automated and orchestrated this is a machine learning DevOps mo DevOps is all about increasing the

velocity of software deployments it's about reducing all of the friction out of the process to help people in the organization fail fast to go from vision to value and to take exciting ideas around machine learning and get them into production we've had this horrible problem for years now it's called data science myopia and that is that models are created in Jupiter notebooks and they don't go anywhere and it's impossible to do governance at the company level because there's such a

complex interaction between all of the specialized roles in the data science process this is what mo DevOps is for it's about orchestrating the interactive and the non interactive phase of software deployments it's also about testing and monitoring for example with the data scientists might create interactive explain ability tests to understand what the model is doing a machine learning engineer might create a non-interactive semantic test to test the model behavior there might also be load tests integration

and so on one of the other things we look at is Jordans mo DevOps maturity model and this is super interesting

because I think mo DevOps can become an intellectual exercise and no progress is made so I think it's really valuable to come up with a maturity model so there in your company you can have different projects in flights that are at different levels of maturity and that's established governance is one of the things that interests me the most so

many models that we put into production may have life threatening decisions of consequence in which case we need to have a structured process to decide what kind of explain ability we need whether or not we should use machine learning at all maybe we need an expert system and what kind of check-in process should we have throughout the lifecycle of a machine learning project mo DevOps also allows us to have this entrepreneurial fail-fast culture in business it means

that a business person can spin up a data science lab we can template eyes previous solutions that we've already created we can ingest data from an analytical data store like our data Lake maintaining our role-based access control and all of our data governance procedures my dream is that people in the business can create a data science lab by saying ma locks in it they can scaffold the project they can say I want to have a structured project I want to

have this kind of production target I want to have this kind of environment I want to have this kind of governance process I want to have this kind of explain ability and we can scaffold that in to end and remove all of the friction out of the process that's the dream that we want to achieve with mo DevOps as a researcher usually whenever I have trained the model that performs really well my job is done I'm happy and this

is all good but Jordans job only starts at that point he takes these models that researchers produce and tries to get them to production his team basically does anything that you need to get a machine learning pipeline up running reproducible and up to scale to ship it out how to make machine learning into a real product that's kind of the topic of our talk today and I was super excited to learn about a site that I

never get to see otherwise haha we had Microsoft build a couple of weeks ago as well and the folks at Microsoft released some pretty interesting stuff relating to machine learning first of all they released something called Microsoft seal this is something that they've been working on since about 2012 it's all about homomorphic encryption seal is an open-source homomorphic encryption toolkit allowing machine learning to be performed directly on encrypted data this means data can be shared and worked on securely by remote

data scientists Microsoft have also released something called fair learn which is a set of Python packages for assessing and evaluating and also implementing fairness in your machine learning approaches fair learn a package that empowers developers of AI systems to assess their systems fairness and mitigate any unobserved unfairness issues interpret ml is an open source library for creating explainable and fair and machine learning models on structured and text data and finally Microsoft have released something called

white noise which is all about differential privacy white noise is a toolkit that offers strong privacy assurances preventing data leaks and re identification of individuals in a data set so white noise will add a little bit of noise

to some of your records which means it will no longer be possible to determine whether a certain record influenced the model which is super important for privacy Microsoft announced further work on the zero redundancy optimizer with gradient and

activation memory implementations in 0-2 Microsoft's blog post claimed this optimization it support rating a 200 billion parameter language model a week later open a I blew the machine learning world out of the water by actually building such a model train to this infrastructure GPT 3 has 175 billion parameters Microsoft had invested 1 billion dollars into open AI last year and we're starting to see some really cool projects as a result of the collaboration only yesterday GPT 3 was

announced by open AI now last year Microsoft put a billion dollars of investment into open AI and this is where some of the money is going Yanik did a really great video on the speed and zero - it wouldn't have been possible to train this GPT three model if it wasn't for the innovations that have come out of Microsoft that have allowed them to train these ridiculously big AI models so GPT - had about 11 billion

parameters in and Microsoft's project Turing had 17 billion parameters and this new model has a hundred and seventy five billion parameters now Yannick did a really good video talking through the ins and outs of GPT three and admittedly it's a little bit like a memorization machine it's an incredibly good hash table that can look up and find all of the bits of text it was trained on and it can cleverly interpolate between them it was incredibly expensive to train

this thing but it just goes to show there are some amazing things we can do now especially using zero shot one shot and few shot learning additionally another collaboration with open AI showcased the remarkable code completion ability of large language models trained on code completing functions with just the function name and a doc string as input now we covered a lot of ground of Jordan today we talked about some of the build announcements we talked about a machine

learning being the next generation of software engineering and how we can do things like infrastructure as code to take the friction out of machine learning how we can do governance at the company level how we can do automated testing and also we had some interesting ideas about interpretability and how we should do the handoff between interactive testing and non interactive testing I hope you enjoy the episode today remember to Like comment and subscribe and we'll see you back next

week welcome back to the machine learning Street Talk YouTube channel with me Tim scarf my compadres Yannick Kilcher Connor shorten and today we have an incredible guest we've got Jordan Edwards I've known Jordan for several years I used to work with Jordan when I was at Microsoft and he's extremely well thought of inside Microsoft I think it's no exaggeration to say that Jordan has pretty much single-handedly transformed the perception of Azure machine learning into a credible enterprise ready

platform for doing mo DevOps so it's super cool to have Jordan on with us today so welcome Jordan thanks for having me damn awesome so why don't you tell us a bit about what you do and your story at Microsoft sure so

I'm Jordan Edwards I'm a product manager on the Azure machine learning team and I work on what we call ml ops or how do you bring your machine learning workflows to production so bit of background for me I started at

Microsoft around almost eight years ago now in July and I started off by working on the Bing platform doing you know distributed systems application middleware business logic tear things like that evolved from that into working on engineering systems so spend some time as a product manager some time as a developer I started working on engineering systems helped move the Bing organization as well as the office organization over to using git and using you know cloud-based build systems and

really introduced the culture of CI CD to these ml first organizations now as I was doing that you know came to realize that you know while this problem or that has been well largely solved or there are tons of tools available for developers the tooling that's in place for data scientists is much more primitive and also we find that many data scientists are coming in have more of a research background and aren't classically trained as software

engineers so the tools are slightly different you know they were trained to publish papers and not to write programs and so as part of that and it did some more investigation sat with some data scientists inside and outside of Microsoft just to learn their ways of working and started formulating a strategy for how can we take some of the tooling that we've been building in Azure and in the Bing platform and around the company and bring it together

into a cohesive story to enable what we were calling like DevOps for MLA I at the time and then has slowly evolved as the marketing terms have developed into what is you know ml ops and that's been a big big collaboration across the companies so we've you know been working with the azure data team at the developer tools developer division and trying to make sure that we have a cohesive story for how you can wire up your data pipelines

your data catalog your model development or training pipelines your model registry and then your existing you know software development lifecycle application release management tools so it's like how do you really complete the loop because the big the big Delta compared to you know classical software engineering is you bring in data engineers data scientists data models and things yet way more complicated really fast that's yeah I have like an

infinite amount of questions so so first I really want to kind of dive into how how it looks because for most people maybe listening or something like this like you either have the the university researcher right that maybe works on on these models and so on or I'm still going to guess that most data scientists are actually you know doing fairly basic data analysis and so on so I think not many people in the world right now are

doing what you're doing like actually build big products with state-of-the-art kind of machine learning so how how large is actually the the machine learning model what how large of a part of just your effort is that or is it like is it mainly there's 5% of work going into the actual model and then 95% needs to be built around it or right

and how does it really yeah it really varies based on the customer and the scenario

so some customers will come to us and say I have some data and I want to do forecasting on it right and like I don't have any machine learning data science background but I have this data that's at least shaped in tabular form and so how do I you know at least you know try to get a model out and see if the models any good embed the model into a business application and then figure out you know the the trickiest

part is okay I've got this model unlike normal software you can't keep a model around forever right like they tend to have some seasonality or properties of decay to them it's like how do I figure out when I need to create a new model to replace it so you know we come you know in all of the talks that I have with big customers it's like okay here are the you know here's this process maturity model for the machine learning lifecycle

here's stages one two three and four and the usual response I get is like well we're at phase zero like we have we have none of this so and and a lot of that too comes with it's not even necessarily the about the tooling it's more about the people and the process and this is we saw the same thing with DevOps as well right like anybody can go and stitch together open-source tools to set up a machine learning platform but if

the process and the people and the roles and responsibilities aren't set up ahead of time you're just gonna end up with what is more or less a disaster where you've got you know data scientists have a bunch of like CSVs pulled down onto their laptops the way they've trained the models like even they can't figure out how they produced it they've got a pickle file sitting somewhere and then they go talk to their dev and say hey

I've got a model and they're like what is this thing how do I use this so these are the the sets of problems that we really want to want to go after as much as possible and so from a like product strategy point of view the first thing is okay let's look at how data scientists are working today so we have there's this kind of two different schools of data science we've seen you have the folks who are really big on

notebooks and you have the and those tend to be more of your like researchers doing data science and then you have I call them AI developers but it's basically software engineers who are doing machine learning and it they send anymore you know IDE users so you've got your your notebook users and your IDE users they have very different styles and ways of working and so how do you build a solution that can capture and help make the model development

process group reducible for both of those flows so that's sort of the biggest pain point we're trying to go after and tackle right now that's super interesting I think we'll get on to the dichotomy between science and engineering shortly the articulate some of the goals that I see for ML DevOps I think it's squarely about seeing machine learning is the next generation of software engineering so that's not necessarily about these there are some data scientists that are just doing data

science on their own machine interactively we're not talking about that we're talking about that road to

production so in my opinion that goes of mo DevOps it's about increasing the velocity of software deployments it's about increasing the quality and security of software orchestrating the non interactive phase of software engineering and the handover between the interactive fresh and the non interactive phase it's allowing companies to develop this kind of entrepreneurial culture where they can go from vision to value they can fail

fast we'll talk in a little while about template izing and being able to use infrastructure as code and spin up data science labs and so on it's about introducing governance and continuous monitoring and having risk management for governance or something at that I think we can talk about soon as well because some models need to be verbalized able some model ready to be explainable and there should be some kind of a process that manages that putting guardrails there sometimes it's

very important because we're coming on to the really big thing here which is this is bigger than any one person or team or projects if there is a web of accountability and one of the big things with DevOps is managing the handshakes between different personas involved in this software engineering lifecycle and there is a specialization isn't there of interest and expertise that the data scientist does not care about governance and the the ops person doesn't know

anything about data science and the data scientist might know about building models but might not know about being doing the kind of area testing that an mo engineer might do so with the best will in the world even if you're a unicorn and you can do almost everything you probably really we need to orchestrate exactly yeah and there there is no you know one man shop to do production Enterprise ml and like you talked about many things

but you didn't even bring up you know the whole responsible AI responsible ml part of it which is you know say I'm building a model for a financial institution how do I guarantee that model is not you know discriminating and certain user verbs like bringing responsibility into the picture and fairness is is also super important so yeah anybody's the governance the governance bucket yeah it's it's a bigger question right it's like how do you how do you verify or how do you do

Quality Assurance ran a software that you basically have no clue how it's working right especially if we go into the more deep learning areas of machine learning you don't know and how do you so how does how does something like just something basic like unit testing you know continuous continuous testing and so on how does that work going from staging - you know beta to actual release is it just you have these bunch of test sets and if it scores well then

it gets through or how do people I mean then that's sort of the most primitive way of doing it is you brute force it with your golden set of test data to validate that it works well across all the scenarios you'd expect but we have a bunch of tooling that we've been working on in the open-source community some of like the interpret ml stuff for you bring in shap and lime and you know as you're training the model you try to

figure out which features are being used the relative importance of those features and you know you can do some fuzzing and bias busting where even in the case of deep learning you can remove just remove one signal

and see how that changes the output of the model right so say I just take color out of an image for example and see how that changes what what the model creates yeah that that's super interesting and I'd love

for us to drill into this a tiny bit yes there are so many different tests the lifecycle of production writing these models now on interpretability the Yannick famously said a few weeks ago that he thought it was like reading tea leaves I'm fascinated by it for example I interviewed Marco Rubio he created life on my YouTube channel and the idea is that you come up with a semantically relevant perturbations and then you build a linear model and you figure out

which of those perturbations flips the switch on a classifier but the problem is many of these methods are open to interpretation and especially on unstructured models if you're using these methods on language and vision model Rea they may require another model just to classify whether or not they do what you want I've been developing this idea that I think interpretability methods are useful in the interactive phase but they're not necessarily in the non interactive phase and so you can

have a blended approach and this is why I think we need to have some kind of model risk management process which will take into account the unique circumstances of every single model to give you an example yeah in banking when they do credit decisions if you apply for a credit card they won't use a model they'll use an expert system and the reason is it needs to be explainable but not in the sense that you would think it

you need to be able to verbalize why it's doing what it's doing so they use an expert system and they say what's your net credit balance and how many defaults have you had in the last 12 months and it's code it's an expert system so no machine learning model is explainable it's not verbalize aboard you can explain the what but you can't explain the how so surely then there are there are certain cases where you need

to have an expert system and then there are some cases that have less severity where you would use explained ability yeah I'd totally agree and I think that you know training time fairness analysis yeah bias busting that's where the tools that we have right now are most useful unfortunately a lot of times when the model is actually running in production it's it's more of a reactive signal where it's you know either reported or a labourer comes in and says like no this

is wrong and then you have to feed that signal back into the model training process it's like oh you know this this signal is being weighted incorrectly or this signal just needs to be dropped completely from the training process that's getting to that degree of maturity that requires a lot of muscle and we're still you know just starting to turn the wheel on that one and that's that's also why for a lot of many of these highly regulated institutions have

been very slow to adopt machine learning and in particular deep learning just because if you can't explain it no regulator is going to approve you know making potentially life-or-death decisions with a black box well could we rip on that just for a minute yeah we we don't require humans to explain how they know things we can't verbalize how we know things an aeroplane pilot cannot tell you how he fly and how he flies the plane what we

do is we do testing and

human intelligence has much broader generalization so less testing is needed we can assume with less testing that the pilot would be able to fly in more situations deep learning is a bit like a hash table given a sufficiently good sampling of the input space there is some interpolation and and given sufficient testing we would be able to infer that the model would run in production as expected so surely given the correct amount of testing we could

rely on deep learning models yes assuming that the case is something the model has seen before but that's the best the key thing right great I love this methodology that Marco Rubio came up with around doing error testing and model so he said things like lime and shopping are great for the the interactive frame but in the non interactive phase we need to be building expert systems to test the models they need to be able to run as code and the

kind of thing he came up with was if it was an unstructured problem being able to turn it into a structured kind of like a unit test and he had this like this concept of semantically equivalent adversarial examples so for example you could have a rule which would turn what followed by an hour into which followed by a noun and then you would see in your test set did it flip the switch on more than four

percent of the examples or in the fairness example you could do data grouping and you could say well split it into men and women and is the accuracy higher on men than women in which case let's not put this into production so what we're doing is we're coming up with expert systems right to determine whether or not so I suppose verbalize herbal tests to determine whether or not something does or doesn't go into production so we're

taking the magic out of the process and I think the trick to doing that well and effectively is having a good semantic understanding of data in the first place so one big gap I've seen from lots of lots of companies and we even have this problem in Microsoft in some places is like we just don't have enough labeled data or enough knowledge of what we're working on to be able to solve that because you know for for adversarial

testing your basic example is like okay so say I'm looking at dogs and cats right and I'm building it but dog or cat classifier but like let me throw in you know a baby now and just see what happens they're like you throw something totally out of the blue and see what the behavior looks like when it's never seen it before or you could just start flipping certain signals around like same for my my you know my dogs sample

it was all golden retrievers right and then my cat sample is all black cast now if I feed in a gold-colored cat which way is it gonna go on the classification so it's like how do you the semantic understanding of data is super important and I think that is where we're gonna see the most development going forward especially as we get more advanced on the simulation side right so when you can use simulators to generate these

data sets that allow us to test and the simulator can even apply labels to those generated test sets that you can then use for a variety of different applications so that's I think yeah the the growth of the data state to include you know test data for ml is gonna be a very cool space to watch so do you see data augmentation playing a huge role in that like I've seen with like the dermatology gand paper

they use a pix depicts image to image translation network so they can have all the different like skin colors lighting conditions so do you think like data augmentation is going to be a huge workhorse and kind of like balancing out the data set and then sort of doing this kind of generalization I think that's gonna be probably the biggest compute spinner in the space going forward to tease you know and and Azure we often talk about like okay you know this this

problem were working on how much you compute do we need to solve it but I think that Jenna yeah generating data pulling out semantic properties of data and extracting the layers of data so that you can look at like you know layers of something as a feature store like that's gonna be a very cool problem space for us to really really hammer on so sorry is most of this like kinda add a distribution detection is most of this

still kind of manually encoded like with the during an example we're manually saying of like lighting and skin color and the size of this skin is the key thing you see like automated added distribution kind of generalization metrics being useful in the near future near future is tough to say it's gonna depend on on the company like for for example I would imagine that someone like Facebook is looking very closely and how to do this at scale and we can

see that making its way into PI tours for example as a toolkit to help with this type of problem but I think that for you know for now we've seen a lot of ml work on the tabular side and we're just scratching the surface when it comes to whether it's text for NLP or images or audio or you know video data which is kind of the the synthesis of all of these put together right and you

know since we're since we're on a call now like the problem of how do you do noise suppression effectively when you're doing remote calls is a very fun ml problem where you can think about how do you pull out you know all the layers of an audio stream coming in and figure out which ones are and aren't relevant in this context like that's that's a real a problem that I'm I'm excited by some of the work we're doing inside Microsoft

and the work that's going on externally in that space in terms of compute how I don't know if you've if you've been at your current position for super-long but how has how has deep learning changed the game of this entire let's sell ml products and so on it's GPUs everywhere right yeah so I mean the problem with deep learning I've seen is a lot of times data scientists like to try a the newest trendiest thing

and deep learning is overkill for a lot of these problems like if I'm building an anomaly detector on a tabular data a simple like regression or a simple scikit-learn like classical ML model or an ensemble of those will probably solve me this faster and cheaper than a deep learning model well that's the the base problem is it's it's it's being used as too big of a hammer to solve a lot of these issues but in terms of like the

the usage on the machine learning side we're definitely it you know well over half of our machine learning jobs are using deep learning today do you think it's different because that a company like Microsoft's you have access to these incredible data sets you have access to all of this compute so it seems more appropriate I would like to play devil's advocate to what you said before because I am a bit of a deep learning zealot and I am of the opinion

that you can apply it to almost anything and to a certain extent it's because of the compositionality of deep learning you it's like it's a unifying computational paradigm which means you can use the same set of ideas and tools to solve any problem and you can compose together models a bit like Lego bricks and you can use some pre trained ones and you can use some stuff them over here and stuff them over there so it's very inviting I can see why

people would think yeah I only need to have deep learning of course people talk about the no free lunch theorem which is basically saying that you do need to have specialized algorithms and in order to get the best out of it it will probably end up there eventually as the deep learning frameworks continue to evolve but definitely right now in turn of you know ease of entry ease of getting started like if you're learning ml starting with classical ml before you

jump off the deep end of the pool it looks to me also like the no free lunch thing is still like very much at play you could imagine network architecture search data augmentation hyper parameters learning rate tuning you could easily have the five percent difference in tuning the deep learning model so I think it's still that automated hyper parameter search still makes it the no free lunch theorem even with deep learning right it's also interesting talk about where you want to

use a deep learning model is the deep learning model your endgame or is the deep learning model for example just being used as a feature Iser we've seen that be on the automated machine learning side a nice way to ease customers into deep learning is throwing in like bert bert-- encoders as things that we can run on on text data but i think the key is going to be not exposing the inner workings evolve the derivatives inside of the deep learning

framework to people trying to train models right away it needs to be explainable understandable some of that you're touching on something interesting though which is one of the reasons why we've struggled to have an ml DevOps framework for years is every single ml DevOps project is different the computer's heterogeneous you might have distributed inferencing you might be doing data engineering effectively using deep learning models the permutations of these architectures is mind-blowing ly large so I'm interested to know how you

get a handle on this really because one of the things that excites me is I genuinely want to have a way of template izing and scaffolding this I want someone in the business to say I want to create a data science lab and I want to have this I want to have this it's an unstructured problem here's my analytical data store I want to do this engineering I want to use this pre-baked model that someone else has done I want

to have this production ization target and then it would just create it for me and then I just check some code in

or I just create my as your ml pipeline then it will just do it how far away are we from that we're getting to a better spot for that we're starting to I arrived at the point now and you're seeing you know technologies emerge in the wild whether it's cube flow ml flow whatever where the concept of machine

learning pipelines and being able to define strongly-typed inputs and outputs two steps in the pipeline and having the ability to move those inputs and outputs from one compute fabric to another like that's that's getting more more and more fluid over time because you know a pretty common scenario as you'll see data preparation is done on spark right and then you'll want to use you know GPUs to actually go and train the model and then maybe CPUs to evaluate or compare the

two the two Delta's like that's a pretty common scenario we hear pop up and so that getting to a clean interoperable format between different steps in the ml pipeline is is super important and that you know when it comes to that so for example like do we want to go all-in on Parque or do we think that there is a different standard we should be looking at for how to federate data across these computes then there's also the question

of you know data flows so how do I design a to look at a patchy beam as an example right like how do i define a data flow that can run locally but can also scale up to to massive scale as I go beyond like okay I think this model is good against this you know 0.1% of my data but now let me try against the full thing and see what I get so it's it's heterogeneous compute and it's how do

you scale what was an interactive experiment to run on your full data corpus without shooting yourself in the foot and you know for that's where it comes in to for each of the compute targets that you're running on being able to closely monitor and analyze like are they using resources effectively you know like is how much of the time on the GPU is being wasted right now because this operation isn't taking advantage of it like this is all of the fine tuning

that goes into proper ml engineering and setting up like real end-to-end ml pipelines hi I just want to say as a researcher every single attempt at scaffolding or whatnot has failed every time there is a framework that's like oh just plug in your data here and your model here and boobity boo and I'm like yeah but now I want to multiply each data point by two at this step or gradient from from this point write it like every every single

time it costs me more to learn the intrinsics of the model because I want to do something that the model that the framework designer hasn't you know thought of so I need to like okay wait watch what do i need to sub class right now it's just it's just kind of painful every time yeah it moves too fast right if you're outside the scope of the operators in your framework then you're kind of you know largely stuck and that's where I

actually think docker is particularly important and useful for machine learning because with docker and containers you can actually design a heterogeneous compute PI plan where you know sure all of the operators in your framework can run together in memory whatever but say I say I want to mix like tensorflow and PI torch together right like they have some fundamental and compatibility so how do I build that bridge well spin up two containers just pass the pass the output in an

interoperable format whether you put it in you know a shared memory bus or you flush it to the disk whatever but I think combining containers with containers running on heterogeneous compute with an Orchestrator that can handle that is how you solve this for real I don't think there's ever gonna be like an uber framework that has every operator a data scientist would ever want to use it's just not realistic another trade-off is a the dichotomy between having control and governance

and having freedom and this is part of what we're getting to with this cultural dichotomy between science and engineering when I was at Microsoft I was surrounded by incredibly gifted and bright people who were multidisciplinary they they were data scientist but they were also brilliant software engineers so I would be forgiven for having the the misconception that we should conflate science and engineering and my strategy on several mo DevOps projects was to encourage the data scientists to be code first to use

IDs to use software engineering tools like git and C ICD and so on to participate in in the full lifecycle of the engineering project and I had some resistance at the time because people were saying well these data scientists they they might be physicists or something they they might not know anything about software engineering they yeah but I was making the argument that well it shouldn't be any different right because even if you're if you're designing a computer science algorithm

you go to a coding interview at Google what are you doing well you're whiteboarding it and you're realizing it and why is data science any different because the big thing they resist is any attempt to control the way they work especially in a scrum methodology because they see it as a kind of truncation or censorship of their fluid process of thinking and so on and I used to be very madly against that but now now I kind of I understand it I they are

a different breed of human being had the same problem as you when I first started on this too I'm like wait they don't understand get but it's so easy like what's going on here but I think what it finally struck me after spending you know three or four weeks with just like embedded with a team of data scientists and looking at how they work I mean seriously like it's it's so iterative and so much of just trying different

things and seeing what sticks it's not like here's this task go solve it like go implement this class like that's it's it's experimental and so the workflow for interactive experimentation is very very different than the software engineering workflow yeah that's so true but I suppose you could call it a hypothesis driven development and it's a it's a different I think already that the curbs the different cuz even in software engineering there's the famous coastline of Great Britain problem and so that the

coastline has this fractal property so every time you zoom in there's more detail so you tend to underestimate the duration of every software project by a factor of three and I think for data science projects is more like a factor of ten I was saying everybody yeah yeah even higher in it's because if you ask 10 different data scientist how to solve a problem they'll give you ten different answers right and just that the curves are different if you look at it as a

software engineering project that the curves are different it's much harder to predict what's going on I think you have to look at it as a research project and I think for most organizations you really have to have a separation between your data scientists and your ml engineers like that's the the other big miss we've seen is most organizations don't actually have ml engineers they have data scientists and software engineers who can't talk to each other and that's why they get stuck so you

know we've we've talked to a bunch of customers now and even helped some of them kind of right JD's to pull ml engineers into their organizations really what you what you want in that use cases you want you know your your AI developer so you want a software engineer who understands DevOps and understands enough about the ML problem that they can take what the data scientist built and operationalize it right it's it's like a computer scientist versus a software engineer

it's the same thing so I wanted you to spend more time talking about what the ml engineer does because I think this is the role that if you look in a lot of industry at the moment they don't have the notion of an ml engineer sometimes if you look at some of the vendors providing machine learning platforms that they also remove the notion of a machine learning engineer whereas I think it if you if you think of a

software engineering project it's something which accrues technical debts in fact we're going to talk about this paper called machine learning the high interest credit card of technical debt that's got some super interesting bits in it well what I would like to do quickly though is is share my screen and you've got this wonderful deck where you talk about a maturity model of a machine learning so let's just do that now this is wonderful this is a set of

slides that came from Jordan and I think this this beautifully articulates a maturity model from ml ops and the reason I think this is useful is I think we run the risk of intellectualizing this process when we start talking about ml ops we take it to the nth degree we start talking about sophisticated risk governance procedures we start talking about AI ethics and and legals and automated retraining and things like the canary deployments and zero downtime

rollbacks and so on what's so good about this is you can put a mark in the ground and you can say well I want to have a certain level of maturity for this particular project so level one is know ml ops this is how we started this is when a data scientist will look at the data catalog get some data from an analytical data store select an algorithm but what happens next this is the key problem with ml ops it's

the myopia of the data scientist it's it's where does the model go after it's been created so the first thing that we do here is is we create reproducibility by automating what the data scientists have done and this is super interesting because this slide solves the problem of what we were talking about before because we were saying that we want data scientists to be like software engineers and rather than trying to force them to use reproducible software engineering

tools like git and vs code what's beautiful about this is is as remote has this concept of a pipeline and it's like a

workflow engine and and that's how we get the reproducibility because the pipeline will record their lineage to the data and the hyper parameters and the code and so on so the data scientists from their Jupiter notebook can create this pipeline interactively and then it becomes a non interactive process later on so I think this is the

key thing that solves that problem right and the cool thing about that too is that yeah the whole machine learning pipeline product was a grassroots in-house effort at Microsoft which was started to solve a very real problem which is you had data scientists that were building models on their laptops and they wanted to be able to run the training process somewhere else so it's like how do I have you know I have this code here how do I run it somewhere else

how does anybody else run it and how do I put it into a pipeline without requiring them to learn like you know DevOps wasn't even really a thing when they first started so they they built the tool to solve the problem of course they originally built it you know back before Python for data science of the thing so while it was just C++ being stitched together but then we took that internal core and kernel and baked it into Azure

ml and you know going forward we're also trying to look and see how we can take that that same process flow and logic and bring it to the open source community so that we have you know an an open standard for machine learning pipelines the meaning is just that if I have done something once it kind of records it and it's able to run it again or is it more of a is it more of a if

new data comes in I can run the same pipeline find to find a new model us data scientists won't ever give that up we can't you can't replace us with a with a pipeline every model needs needs its own care from a legal point to be more than anything else it's a you need to have reproducibility you need to be able to lock down how was this model created I think that's the first thing and the second thing is in these

retraining workflows we want to be able to recreate the model again in the future without the data song right I think it's directively creating it exactly it's reproducibility it's also composability though so the next step I think Tim will we'll get to this in a subsequent slide but is okay right so I have I have that have a reproducible model now the ml engineer needs to be able to take that model package it certify it and then release it to yeah

somewhere right so that could be it's deployed out and running in a container on kubernetes it could be it's inside of a sequel database and is used as part of a you know a relational query there it could be it's you know deploy it out to your phone and it's used it's being used for augmented reality or it's being used for you know background noise suppression and when you're talking to somebody so that's that's the the cool space is

when you bring in hey I have the model registry I have the pipeline that trained the model I know all the dependencies I can get it out and then once it's running now okay cool i've you know trained a model to solve this one very specific problem how do i make it easier to not reinvent the wheel over and over and over again it's like any any company with more than one data scientist runs into the same issue where

it's like I know I know I have all these features somewhere in my company but I don't know where they are and I don't know which ones will work with you know with what I'm trying to put the problem I'm trying to solve so that's one of the big areas that we're investing in is you know how do you define an open format for these reusable modules that you can stitch together where the inputs are typed such that I know what types of

data this model will work or this yeah the module will work against I know what outputs it produces and then how do I go and stitch together this graph and then how do I let somebody else fork the graph collaborate on the graph with me and even just take that whole graph and use it as a module somewhere else so the pattern we see in you know in being in office and all these teams doing real

production ml at scale is they have these huge graphs that are not the work of one data scientist it's whole teams who have built feature Iser x' Bilt's you know modeling techniques built parameter tuning optimizations and they're they're all stitched together and like that's that's how you actually get the ml lifecycle started I see I'm gonna go to the high interest credit card paper now and one of them is correction Cascades so there are often situations in which a model a for

problem a exists but a solution for a slightly different problem a prime is required in this case it's tempting to learn from model a prime a that takes a as an input and learns a small correction so you can see here that when you start to develop this web of interdependencies between models just to develop this a bit I want you to articulate your vision for company-wide I think of every single machine learning project as being

independent in in some sense I think of it as a software project it has an owner it has an accountable let's say a product owner and and legally maybe that's where the accountability lies and at the company level maybe there's a model catalog maybe there's a way of composing and publishing and discovering these models together I'm not quite sure how that works but do you run the risk of if you publish something who's responsible for it and do you have this cascade problem

right it's the it's the same problem you see in software engineering right so we had this this issue when you know we call it the one es effort inside of Microsoft and get everybody onto the same tools and systems and promote a culture of internal open-source it's like okay well internal open-source is great but you have to introduce the concept of accountability and ownership and the lifecycle means you need to be able to archive and deprecated things or

indicate should you be building on top of this or not so it's a question of I have my experiment I want to collaborate I want to share I want to then so say I've built a feature Iser I use it for my tiny thing right but now a huge other internal team comes and takes a dependency on it like how do you set up shared accountability responsibility for for these modules that you're stitching together to create these you know the

pipeline's of the ML lifecycle and that's where I started to talk about Oh get gets very interesting right so how do you smoothly bring in get or version control with the concepts of you know tagging and labeling and ownership information and even policies right I have these modules I've created do I allow people to take

dependencies on any version of them or like only the version in master or only the version that's been tagged because it's been ran

through these 50 different validation datasets to make sure it works that's where we're trying to get to is you know directionally we introduced shareable modules make it easy to system into pipelines and then introduce the the lifecycle on top of the shareable modules in a seamless way so that you can go from interactive in a notebook and ID whatever to in a pipeline to in a shared pipeline and that's ml obviously basa so the envisioning might be

something like Doc docker registries for models or for feature icers for parts of pipelines exactly there's you know there's there's docker registries to capture the dependencies there's packaged feeds that have the different libraries that you might want to pull in there's you know jobs that are predefined that says hey you know run this command in this container with this parameterize data like that's the layering that we're trying to go for to

make sure that again it's a composable component is a ball stack that we expose so how do you think about like when there's a merge conflict in traditional software engineering it's usually we just have written different functions and then bang we just put them together it's easy like that but if Tim Forks my model man Yannick Forks the model they each training on a different subset of data how do you merge the network's now it's an excellent question

so to be able to merge models together so say I have original pipeline and then pipeline trained on Tim's dataset a pipeline trained on Yonex data set B do you take the same pipeline and run it on both of the data sets together like just a feature space over a lap between the two or do you need to actually run both pipelines in parallel and then build some sort of ensemble in technique where

you'll you know run a loop through both of the models and apply some pre or post processing to it and on ensembl is also you know a big thing that we do internally and we're just starting to scratch the surface on on model ensemble sure you could just merge the neural nets together you can try to fold the layers together but I think ensemble and gives you a more sort of flexible way of doing that because if you think about

you know the way the human brain works for example it's not just like one neural-net it's like every single synapse in your brain is a different neural net and it played ensembles all of those together and its own meta graph and then will spit out a result just to use the your brain is a computer analogy they're the solution that seems obvious to me for this problem would be like this knowledge distillation technique where tim's fork model is gonna label the data then

Yonex fork model is gonna label the data and i'm gonna train on these to label distributions now is that something that like is being thought of as this technique for merging or because ensemble intends like well now you have this collection of models each of each of them being like you know hundreds of megabytes of parameters at least right so do you see knowledge distillation as being like a useful merge tool for these networks I think we'll get there

over time I just think this the space isn't quite advanced enough yet to do it but ensemble is a cheap way to take models that were trained on different data sets and stitch them together well it's it's cheap from a training point of view more expensive from an inference point of view and so that's where you know we're doing a lot of work on the Onix side was like the open neural net exchange to figure out how do you

quantize models and shrink them down so that they can give you good performance on the inference side and that that again that space is still very new because the frameworks are all evolving but it's been essential for us to be able to actually use you know models in real time like the grammar models in office because you know you can't spend five seconds waiting for like suggestions for sentences to come in that's just not realistic there's also

the the interesting and when it comes to a/b testing of models so I have a model running on the client it's very expensive to deploy a model down to the client it's much cheaper to you know run one in the cloud so how do you do an a/b test with a model running in two different places so it's kind of like your cloud Edge champion challenge or ad hoc analysis and so you know one approach we've seen is

you'll have you know different rings you roll a model out - you can start by running the model in you know offline mode or in shadow mode for some time where you don't block on it the model can be even you know denser more expensive to train but you just see sort of what the fidelity loss looks like with different techniques of quantizing the model and that helps you with being able to use it in production it's also

the question of what you're using the models for like are they running once a month okay like who cares like you know hyper optimizing the latency day it's not a big deal but if they're running at 5,000 requests a second it's a very very different problem to go and solve for you've you've mentioned a bunch of times right now these these ideas of strongly-typed inputs and outputs to these models and so even if I like okay let's say we

compare Python to C++ or something where in in C++ i have my strong types and I know everything's how fit together but if I look in terms of machine learning if I do that what's the type of this thing it's just going to be tensor right like what that tells me nothing at all whenever my no IDE can help me there and even even if you think ahead and say maybe we can you know the compiler could

track the shapes of the tensors right and there's like okay 10:05 by through 200 times like sometimes I'll just put like some weird number as the number of dimensions or hidden layers so when it shows up in a tensor like I know oh that's you know it's this indicator it's like whereas where you know what I would want from a typed input would be something like this thing is an image that has been normalized something this

technique ran through this sort of pre-processing right that being the entire type of the data points how do you how do you come to that place so we're very very early here right now one of the things that we're trying to do is you know for the major types of assets in the system so for example like this is a machine learning model and a model has a load for has an init

function predict function on top of it here's some data that goes along with it here is the environment required to run it so there's a question of okay the model is what methods is a model expose the next 11 the next sort of step there is what is the schema to the input to the model right so the the schema could be like okay this is just you know a panda's data frame with these columns inside of it right and so how do you

take a schema of data set flowing in from haniss whether it's coming from no json erp see whatever the protocol happens to be but how do you take that and translate it into something that's compatible with like the tensors on the front in there and that's we're just scratching the surface for now for now what we have is we have I can tell you you know the scheme of the input function to this module in the pipeline

and what that looks like from okay it's a dictionary of you know I expect these names to come in but we're definitely not the full granularity of tensors yet and that's that's something we've been struggling with and iterating on on the inference side and so we've released a public inference schema package but we're trying to figure out how to tie that into for example in you know nml flow and that flow has this model package format where you define like the

flavor of the model which is how to load this thing how to run on top of it but like how do we introduce scheme up those flavors so we can do that strong typing and then how do we extend this beyond just models and to any component inside of the ml workflow and that's that's the next step that we're we're working to get to right now and where we're having to do that because we have the same

problem for you know explanations right like an explanation you need to give it a model and some data as as input so it can explain the behavior there so it's because everything is so new and nascent and there's so many different open source tools I don't think standards have emerged just yet but at least from you know an azure and Microsoft point of view we're trying all of them figuring out which one's work best for our customers both

inside and outside the company and our net goal is to get to a set of open interoperable standards for all of this like we don't know what the shape is just yet but that is what we're aggressively pushing towards because we don't want to have data scientist be like locked into Azure they should choose a sure because it's the best cloud to do their ml work on the subject of ml flow I used to use ml flow because

it gave me a wonderful interactive work flow on my machine before I kind of promoted to you know as your machine learning and but you don't need to use ml flow what's the story that so it's an excellent question right now we have you know when it comes to to logging things in in a job right you can either use the azure ml SDK or you can use the ml flow one and we have you know API parity across across across

the two of them they're both conceptually born from the same idea which is how do you enable reproducibility whether it's through you know tracking the model package format or the new work that we're working with them on for the model registry as well right so I guess we're we're working on figuring out the the answer there I have my thoughts and opinions but I'm curious what what you guys think like do you think it makes sense to push

aggressively on an open interoperable format for logging and tracking or does it make more sense to you know keep keep a proprietary one or to use the Azure machine learning one or to look at other open source tools or other companies that are in this space like comet or weights and biases or Neptune like there's so many tools in this space like what do you guys think speaking personally interoperability and portability is not that important yeah

the people talked about the need for cloud portability and so on and most companies have a multi cloud strategy and it's completely accepted that you'll you'll need to do ml things on Azure ml in a certain way and you'll need to do say to make is completely different I think there is a need for global governance across the entire company for models but I think that's something that's slightly different I think every single project will have its own model

registry and this is what I want to get your opinion on how does that translate into company-wide governance so I think and one of the things we're working on right now is you know between the Azure AI and your data organizations and looking at open source tools like Apache at lists how do we enable an enterprise-wide data catalog that by default includes all of the modeling work being done so if anybody you know say create a workspace

create a project whatever I pull in some data just by pulling it in the data catalog should know or track like okay these different projects are now using this data anything I output from that data should also get tracked in the data catalog so I think you know the while a model catalog is interesting I think it's really just models our data and it's just a subtype of the problem that a data catalog is trying to solve so you

know from at least my preferences that we should be collaborating together and figuring out how to just make models and and even software applications that you're rolling out like those those should all just fall into this enterprise wide data catalog data governance life cycle experience could you just expand a tiny bit on that because models seem to be slightly less inert than data because models make decisions that have consequence and we were talking about fairness and bias I

mean is the same thing applies to data right people make decisions on data that has consequences so I don't think they're actually that different it's like a model is just some transformed data you may have read a bunch of features or feature answers on it you may have run some different modeling techniques on it but fundamentally it's some data that you spat out and it's gonna have the same problems as any other type of data you're dealing with

but is there a difference the model is the function not the data you can put data through the model does that make a meaningful difference in your opinion I think to a to an extent there I guess the big thing is a model is a combination of the data and the function to run on the data right so it's really more than a data catalog it's a data and feature and model catalog that you can look at all of that

together and then see all of the different jobs whether it's just a data processing job or a model development

pipeline or an inference my plan that are using those assets because that's the whole sticking point of ML versus normal software engineering is you're introducing data like that's that's where all of the complexity comes from right like code code is a commodity data is gold so when you put when you put gold in your applications all of a

sudden things start to matter a lot more what kind of things would you record in this catalog I'm I'm thinking that data has very important dimensions around privacy for example right and it's not necessarily structured data it's even harder if it's unstructured data because how do you possibly record useful semantic information exactly and so right now the things we're looking at capturing is what is the intended use for this data like what is the source of

this data what is the the PII level of this data we have this problem in an office when we're training models so say I want to train a model on top of email data well if I want an interactive experience I can either use public domain email data so like the Enron emails or you know the Hillary Clinton email data sets those are pretty popular ones internally there or there's you know semi eyes on where if for example

internal Microsoft employees opt in then I can run training jobs on top of those or I can even extract my own emails and train emails on top of those but then as you go you know further and further into the p.i level the PII level gets higher the question is how do you how do you even do training on confidential data and we've built a bunch of tools internally and we're actually in the process of open sourcing those right now

to enable you to to do things like log scrubbing where you can scrub PII out of your training jobs or enable you to do model training and a detonation chamber on top of compliant data so I can shape my training pipeline on the public domain data refine it further on mine then start applying it to you know a real enterprise data set corpus and provide the security to your larger company customer whatever that their

real data isn't leaking out because you just train on the train on the samples or on a subset they're one extension of that is and there are some announcements that build last week and maybe you can expand on these as well that there's something called Microsoft seal which is this homomorphic encryption methodology for scenarios when you can kind of allow data scientists or third parties before memo on encrypted sensitive data there's also something announced called white

noise which is a new approach to differential privacy where you kind of like randomly add add noise to certain records so you can no longer be sure that it was their contribution in the model so Microsoft is doing some pretty cool stuff that definitely we have huge investments internally externally and we're trying to as much as possible take the goodness and push it out to open source as soon as we can because we know from you know all of our customers are

asking for this and at least from my point of view I want to make sure that we're building in a way that customers can use it right away and not like you know which doesn't lock responsible AI behind a paywall that to me is just the wrong approach to take so let's say I've hacked into your email system and I have your model copies before and after you go and train locally on Tim's emails and I want to reconstruct Tim's emails that

you've just trained on do you think there would be any way that I could have optimized directly to have a generative model try to produce data that causes that change I think generative adversarial models are super interesting there so that's another validation sweep you can do is say I have this neural net to run basically run a scanner that scans through the whole neural net tries throwing common use cases in or tries to see if it can extract anything PII from

it so yeah just basic example is an organizational acronym dictionary let's say you're training a model for a company talking about their quarterly earnings reports if you start to learn what those acronyms mean and you can reverse engineer it then you're kind of in trouble and you're eroding customer choice on top of it because then you know anybody with access to to that model inside the company could then start running things and try to figure out like okay what's what's changing

over time so that that whole space is gonna be very very interesting to watch evolve on the adversarial ml adversarial AI especially because you know as as Tim alluded to I think deep learning is becoming more and more prevalent by the day classic it classic ml for most customers is where they're starting but they all want to use deep learning but none of them can use it until we have these these types of responsibilities I tools you think there will be like sort

of like you have black cats right now for classic like for servers and so on there there might be emerging an emerging class of a black hat hacker for 4ml models like public sleep publicly exposed models you already see things like this we we have teams internally that do exactly this right so we have teams that will go and look at the models and do threat modeling threat analysis on top of them and we have seen

yeah and the even even in the external community trying to hack into companies machine learning platforms to figure out what they're doing what data their models are trained on but that's going to be the you know the next generation of black hats are gonna be the data scientist of black hats hosted so Jordan it's been an absolute honor to have you on the show this week but before you go do you have any exciting announcements or anything else

that you want to let our viewers know nothing at the moment where we're cooking up a lot of really cool stuff internally obviously kovat is set back some of that and when it's going to be announced and rolled out but let's let's grab some more time in a month or two and maybe we can even show you some of the cool stuff we're working on oh that would be an absolute honor just to finish things off if you could go

back in time and give you a younger self some advice what would it be oh stop trying to talk data scientist out of using Jupiter notebooks yes that is what I should do the same thing that's a wonderful advice Jordan is it's been a pleasure thank you so much for coming on the show and we look forward to getting you back on yeah thanks for having me this is great guys thanks so much bye I really hope you enjoyed the session

today remember to Like comment and subscribe and we will see you back for another exciting episode next week