Q1. Could you describe your process for developing an ML system?

"Ok how do I go about it? Ok, so usually I don't know, maybe I'm a bit old-fashioned or something, so if that's how it comes across, then I guess that's the way it is, but I'm usually in my workflow first and foremost very classic, just Jupyter Notebooks. And start the same way. So I open the notebook, get data of course. Usually in the beginning, when you work with customers, the data is always kind of compiled. Very often, customers just have CSV files lying around and say: Oh yes, here's a bunch of data, why don't you do something with it and see if you can somehow manage it, see if you can somehow get to know it first.

And yes, usually there are quite a lot of meetings with the customers to clarify what the data is and to get some technical input and then you iteratively work through the whole thing, so relatively straight forward, so I usually don't use any special tools at this stage, if I'm honest, so without any cloud stuff or anything, it depends on how big the problem is scaled, but so far I haven't had the situation that I had to allocate a class of 16 v 100 to check if it's even possible. Most of the time the whole thing requires very classical approaches and usually also relatively classical solutions. So you can't necessarily build a Transformer architecture or something and then most of the time I'm like he throws <an ML service> at it and sees what happens and usually the customers just want to know how good can we get? Is that good enough to continue? And therefore mostly relatively classical does the notebook produce some plots with matplotlib. And then somehow try to sell the customers that this is somehow worthwhile. So you feel a little bit forward, so the customer usually gives clear information about what he wants to do with it. Something like ok, we want to estimate something here from data that actually have nothing more to do and to save us so to speak processes or sensors or whatever.
At least with <companies> for example often the case that one has something like we measure a few accelerations here and want to determine then actually elongation there and there one can make that somehow about a machine learning model, for example that is such a relatively yes current object at the moment and then is just then the question ok can we do that? In the context of accuracies that are useful for this analysis? And then you have to ask the customer what is useful and then he says something like under 6% should be ok. And then you would continue until you touch this threshold, so to speak, and as soon as that is reached, then you would continue and say ok, now we can bring the whole thing to the next stage, so to speak, and elaborate the whole thing a bit more. These first analyses are all relative, so there's just a lot lost because none of it is retained for long. The first approaches are usually not useful in that sense and most of the time everything is thrown away again, because as soon as it makes contact with the real infrastructure that is there on site, there is not much left of this old notebook.

Then you're usually in the next stage, where you have to make friends with the customer's infrastructure a bit more, so that you can easily get to the data, etc., because this CSV story, that's in order to be able to iterate as quickly as possible at the beginning and to quickly get an estimate of what's possible.

For example, we work a lot with Azure and that there is a lot of infrastructure in Azure that could work for example so that I say ok I deploy, my node model for testing even just directly in the cloud and allocate a VM or something and make that on there and can then extend that directly around such an ml experiment or something. I am aware that this also works in the notebook etc. and that there is also this SDK and so. But I find it partly just really

cumbersome so. So here's actually the real problem when you start working with customers and the real problem at the beginning is that you have to get your login data. That all has to be explained with IT and then you have to get some access to keywallets to somehow access all the databases. That's what holds up so much at the beginning and that's also the reason why most people, in my experience most people, don't solve this in the way that you say, we're going to make this so fluid that you can say at the beginning we're going to do everything on VMs and then everything is deployed almost immediately.
Yes, that doesn't actually happen, because most customers say, yes, here's my CSV that I pulled last week and yes, just make sure that it works somehow.

Yes, and that's why there is little left, simply because this whole, so this infrastructure around authentication is just partly incredibly slow and very, very often, so at <company> we know the problem now also that then very much is set on security of course and then you are also a bit paranoid, so I mean, I mean, the problem is clearly there and it is also right that security is emphasized and so on, but for me, as an external user, it is of course super frustrating when you sometimes really wait a week for your access data and during the week I could have simply started with other data and then presented something on the next Monday. Not possible if you wait forever, etc. Yes, so as a rule, as I know it, it is just so that the customer then wants to have something like a pipeline that he can maintain himself halfway, he wants to have it lying around in the cloud somewhere and that then does something. To give you an example, I had to shimmy along, somehow I don't know 6 wind turbines, which are equipped with special sensors and they post a stream of data continuously on some server. And now I want to take quasi in a weekly cycle for example this data again, make my machine learning estimate and then dump that somewhere, however and that should just be practical for the customer. How you do that depends entirely on how the customer can work with it.
So you want to have such a pipeline, which then runs with a schedule and my customer has then of course the possibility to look in such a UI / graphical interface, which is preferably what people want to have, then: is the model running, has it run through the last time. How long does the model take, how much does the model cost me?
And exactly, and that's then exactly where it mostly runs to and now in the case of Azure, for example, that would now be an ML experiment that you would then just run. "

Q2. Do you continue experiments after deploying the ML system? If so, how do you go about doing this?

That's kind of the rule of how I do it, maybe not the best way, I don't know. But so, that's my workflow as a rule. I have to admit, there is one thing that you can really optimize, so there I'm also sure that there are other people that do it much better than I do. Something like keep iterating on what you originally touched at some point. I'm not that organized, so I'm also a very iterated person I have to admit, so I, I open a notebook and then I do my stuff and then that's abandoned right after that so as soon as my code is somewhere clean in a module, then I never touch that thing again so that's usually how it's done when the customer says Hey, I want to have something else though, can you try that out with the other features or something. Then I would probably usually then just create a new notebook untitled42 and kind of start from scratch there. It all sounds horrible and it's so a little bit bogged down, a little bit scattered, but ultimately I have to admit, I don't really touch old code in Notebooks anymore. Everything that happens in notebooks is so a bit yes, so that's often more work to then fix that again than to just start from scratch or which is probably also not a good thing, but so there you can one hundred percent rather still optimize. No, so as a rule that is then still relatively sorted. In so far that I. Well not, I don't want to say mostly or always proceed so that I version my repo properly, I have a CI pipeline that tests everything and deployed then also automatically. That is, if I change something, then of course I just change it in the repo, push, make a pull request, so really quite classic development hell. So that's really exactly how I proceed, and as a rule I want to have it so that everything happens automatically in the end until deployment. So yes, that's one of the first things I take care of. Exactly yes, that's exactly how I proceed yes, so mostly dev, test and prod. Yes, so quite classically the division. Yes, dev is always run when I push on development, when I push on master, then usually, either on test or prod, depending on what stage in development I'm at, so. That's one of those issues, I think that's dependent on self-discipline and usually so as undisciplined as I am in the early development process so the more it's just important then, later on, you kind of have to learn that. And I also don't think that there is a big detour or a big alternative to it, because somewhere there is still the work to say now I have to pay attention to where I am, in which scope I am and so on and in which stage I am working and so on. I don't think there's any big trick or anything. I think self-discipline is the only solution and that's what I do.

Q3. Which tools do you use?

Yes, I don't do anything from the notebook at first, for example, so the notebook is just for experimenting for myself, to find out a little bit what the whole thing looks like, to generate some plots and so on. That can change arbitrarily depending on how the requirements change and what is also a classic problem is that the customer is not aware of the requirements at the beginning and then figures that out as you go and accordingly it can get arbitrarily ugly how it looks in the end. That's another thing where I would say maybe I can learn a lot myself in terms of like yeah what is, a good SDK for that etc? What is a good framework for it to develop and so sure I'm also kind of quasi professional amateur there but still usually I do it so that I then just slowly work out in the notebook, ok what do I really need to train such a model? For example, what I have not yet had as a use case is that I have trained a model with the notebook, which I then pickled or dumped somewhere and then uploaded to the cloud and that is usually also not maintainable. Nobody wants that, so I don't stand there in front of the customer with that's the Jupiter notebook where all the stuff happens. Then they will surely send me packing, but what is rather the use case: I somehow extract from my notebook the essence of how the model is trained, then slowly work that out. My development work just happens in the notebook, because that's just comfortable, I think, and that's then poured into a script and then quite classically in principle DevOps, so you create a repo, screw in all your code, write tests to it, build CI CD pipeline and deployed, then in the way then the pipeline in such a CI CD pipeline and somehow versioned best. So I just like, it's my personal preference to organize things in Python modules and deploy that way as well, because usually it's as follows it's of course possible with all this Azure/ML/SDK stuff. to write in the script and that then contacts Azure and uploads all the stuff and defines the pipeline and so on, that's also all so of course I effectively do it that way of course, somehow I have to do it, but ultimately.

Q4. What are some common challenges you face in this process form a tooling perspective?

What I often notice is that it just makes for such an ugly way of working. You start to collect everything in some scripts and then somehow you don't have a proper organization in there, so what my goal is when I implement something like that is that this stuff can run locally on my end as well as in the cloud, that it really runs in parallel and I can still import the stuff on my end and then just run a pipeline through it, just like on the cloud. If I had to say what has taken me the most time, overall for all the projects I've ever worked on, it's something like waiting for resources to be allocated in the cloud and all the Docker images to be running and so on. That's still like half hours all over the place that's accumulating. If you could kind of say, I can kind of pick a compute target and I can say, he's doing it in the cloud or you're doing it here locally and I can guarantee if it's running locally, it's running in the cloud. That would be, oh my God, that would be great if that worked, just doesn't work at all right now and that's terrible and debugging in the cloud couldn't be worse, so right now AzureML. Especially AzureML was in the last 2 years always so in a bleeding edge phase, where then always something has been changed, just so on the fly, where you realize only after weeks, oh that was an experimental feature. And it's not nice overall, I'm not a huge fan of AzureML either, if I'm honest. Yeah, so that's just a bit of a hassle overall to get that to work, I would say but yeah, that's just my goal, I kind of want to have everything one module where I can run it locally as well as run it in the cloud and in the end the actual script is kind of pretty small, just runs a few things from the module that I could run just as well in the notebook and then I just upload the wrapped wheel into the pipeline, then install it into Docker and yeah, then run it there just like on the notebook.

Q5. Which issues do you encounter when working with Jupyter Notebooks?

So I use for example VSCode a lot with Jupyter Notebooks and there is relatively much just already somehow with it, which then help with some of these points so something like Code Quality etc.. These are all things that are also checked there, so refactoring or something like that is also possible without further ado. So I rely a lot on that and it works quite well so far. Yes, I can't say that much about the rest, if I'm honest. Yes, I don't use notebooks for that much. In the end, I move relatively quickly to relatively classical development when I notice I'm having problems with my notebooks, because for me that's usually a sign that I've outgrown it and just let it go and not think about it anymore and move on to proper development. Then in any case in a Scrum environment with DevOps and whatever, yes.