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

"Mostly there are 2 variants. One variant is that there is already data, e.g. via Kaggle and Co. and you only have to take care of the modeling, i.e. how do I formulate the problem, what problem do we have. Is it classification, regression, etc. What are the base line models or what naive approaches exist that you have to look at first. Then you sample, you look at the data, you have an exploratory process, and you might already find statistics, insights, whether the data is completely random or whether there's correlation there.
Normally, at this point or in this phase, a lot of communication is necessary, you talk to specialist departments, who first describe the problem to you, what they need and so on, in order to penetrate the whole thing a bit, and then you start trying out the first models. You then have to evaluate that, the evaluation is then specific to the problem that the customer is currently having. It is also important to find a demarcation, i.e. what do I have an influence on and what do I not have an influence on with the model. For example, a customer comes in with some KPIs and a ratio that is bad right now, and wants to know if you can improve that. But sometimes you can't optimize everything data-driven.
Then you try out a few models and find some that work reasonably well.
We don't spend an infinite amount of time on prototyping either, sometimes you start with a trivial model just to have some model, and then you're already building your way all the way towards production, that you're already making the first cut, so to speak. And then iteratively improve the model and push it with each iteration simultaneously in the direction of production. This then becomes Continuous Delivery-like. As a rule, not only the individual model is connected to it, but also many data paths, so you have a model and have already connected a small subset of data sources, but still have 3 other data sources, e.g. something external, such as weather perhaps, and then you successively connect further data sources and integrate them into the model and then the next data source comes and you go another round. So prototyping is not separate from putting it into production.

Sometimes customers approach us with a POC and ask if it's feasible. So you can take a data-driven approach. For example, in the area of computer vision, it is not always entirely clear whether a certain usefulness can even be expected for a problem. In that case, we first have a prototyping phase, and we first look in a POC to see whether it works at all, and then we can possibly go through a few cycles until we have an idea at some point. Then you say okay, this could work and then the phase begins where you then bring a prototype towards production. Then you have a clear point defined where it starts.

But we've also had other projects where we actually moved toward production right from the start. That means we really built up the complete stack and built in a naïve model at the beginning to map the entire route. You only do that if you're already very sure that you definitely want to do what you're doing and that it's likely to bring something. "

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

"That's actually already the standard, I would say, so that the experimental environment and the prototype remain in place for the time being. And then you continue to develop and always push the last version, or a version where you say, ""Okay, that's good now,"" in the direction of production.
A model is never finished. That means you always find something to improve it, whether it's a deeper neural network or trying out a different architecture for it. Or you try a new class of

model, you just want to find an even better model. The second reason is that you want to connect new data sources.
In real projects, you rarely start with the complete data landscape, you usually start with a subset of the possible data and then successively add new data sources. The process is often also very time-consuming, e.g. whether you push data to a data lake first. And then the model has to be able to handle this new data, so you go back to the drawing board. [How do you ensure conformance between experimental and production-ready code?] We don't do that, but it would probably make sense. That's the main difficulty you have. The further apart the stack, so to speak, when you build the prototype and then have the production system, the harder it is to customize it.
A colleague of mine had a project where he defined a JSON DSL where the functions were specified, and then tests were run to make sure that both stacks were basically doing the same thing, and then you had that conformance in terms of the functions. Now an ML pipeline is more than just feature generation, and that's certainly something that has its limitations. It's important to have conformance so that you get the same result. On one side, you're experimenting like crazy trying to find a good model, and if it behaves differently on the other side, you're already taking it a bit ad absurdum. This conformity is very important. We don't have a standard solution for it."

Q3. Which tools do you use?
Personally, I'm a big fan of R in combination with Spark when the data is a bit bigger. Also especially feature engineering with Spark SQL, you can do a lot with it. Otherwise, Python is very widespread in the company, about 90% of the people use it. As IDE RStudio or PyCharm/Jupyter notebooks. [Interviewer asks if plugins are used] No, I've never used plugins.

Q4. What are some common challenges you face in this process form a tooling perspective?
"Spontaneously I would say the famous language gap, there I would like to see improvements. That is, I prototype in Python, but the productive system is JVM based, so you have to implement everything in Java or Scala again. Thereby the question arises, how do you do this translation properly?
In any case, one would wish that one says: ""Ok, I have now thought up 25 features in a Jupyter notebook, I don't want to program them all individually again in Java"", but in the ideal case one has a descriptive description. And then you just translate them or compile them or whatever. "

Q5. Which issues do you encounter when working with Jupyter Notebooks?
Testing is difficult and extremely important. Cleaning and refactoring are also very important, because the classic Data Scientist hasn't learned to code cleanly, and you can see that accordingly. It would be very helpful if they could get help. In terms of the idea, you could also make a few design patterns available. The reusability of the code must be greatly improved.