# The Return of the Data Scientist

17 MIN · YOUTUBE · HTTPS://WWW.YOUTUBE.COM/WATCH?V=QDQT99CSHJQ
https://www.youtube.com/watch?v=QDQT99csHJQ

SUMMARY

*The discussion focuses on the evolving role of data science within AI engineering, particularly in evaluating AI models. The speakers emphasize the importance of rigorous evaluation practices, caution against reliance on generic metrics, and advocate for a data-driven approach to identifying and addressing model failures.*

- *The concept of "harness engineering" at OpenAI illustrates the integration of observability in AI systems.*
- *Many AI engineers are currently using vague metrics and relying on LLMs to evaluate model performance, leading to poor evaluation practices.*
- *Common pitfalls include using off-the-shelf metrics without customization and failing to validate LLM judges.*
- *Effective evaluation requires understanding specific failure modes and generating application-specific metrics.*
- *Data scientists should actively engage with data, avoiding outsourcing labeling to those without domain expertise.*
- *Criteria drift can occur when teams do not regularly review data, leading to misaligned evaluation rubrics.*
- *Automation should be limited; human oversight is crucial for nuanced understanding of model performance.*
- *The importance of exploratory data analysis and a scientific mindset is emphasized for continuous improvement in AI evaluations.*

All right, welcome everyone. So today we are calling it the return of the data scientist >> and we're going to >> Great. I'm glad y'all are excited about this. >> Okay, so just to set the stage, um y'all probably have seen this article already about harness engineering. Raise your hand if you've seen this. Okay, so about half the people. So if you haven't read this article, I highly recommend taking a look. Um, but what it is about is a team at OpenAI, they use agents autonomously for a few months to build some very substantial software internally. And what they did is using a harness to keep the agents on track.

And when so when people hear harness uh they usually think about specifications and unit tests but one detail that's really important to look at is okay the harness also contains logs metrics and traces. So like the whole observability stack. Um, and what I'm here to convince you of today is that the harness is data science or at least a big part of it is data science.

So, okay, like let's take a step back like how did we get here? Okay, so like let's look at ML engineering data science four years ago. So like we used to examine the data really carefully, look at the data, visualize it. anytime you had a model or prediction, you would be very careful to make sure you align it with human labels. Um, you're very careful to like use the right tool for the task. And when it came to designing metrics, we really take a lot of care to make sure those are aligned with business goals and the metrics made sense.

AI engineering today feels like we've taken a bit of a step back in some places. So now a lot of times we're just using vibes uh to know if we're doing the right thing. A lot of times we're just asking the model, another model if it did a good job or the same model. Um and then we're not putting a lot of thought into metric design. Often we're just using an LLM to just grade something on a scale of 1 to 100 or just using someone's metric package off the shelf without thinking about metrics.

And this is where people get into trouble and it shows up um a lot. You can see it because a engineers often are scared of what they don't understand. And the data sciency pieces of AI engineering, eval retrieval, those have died so many times I can't even count it anymore. Um but this talk is about eval. And so what we're going to talk about today is okay, how do people go wrong with evals and what mistakes do they make and how you can think like a data scientist to overcome those mistakes. Um and so we have taught over 4,500 people evals at lots of companies and we've seen the same mistakes over and over again. So we're going to highlight the most common ones and for that I'm going to give it over to Shrea.

>> Great. Hi everyone, I'm Shrea. I'm going to tell you five big mistakes that people make coming into working with us and how we can course correct it by wearing a data scientist hat. So the first one is kind of using metrics, what we call generic metrics or off-the-shelf metrics to measure the accuracy or alignment of your agents. And it's tempting to use these generic metrics, right? After all, we use generic metrics for, you know, software. We measure things like latency or uptime. But AI needs a different approach.

Even then, you know, what are these so-called generic metrics? You might have heard things like helpfulness or hallucination or coherence. And yeah, they sound like they're important to measure, but when you really think about it, they're quite ambiguous. One, like, can you really tell exactly what hallucination means? Two, if you're building an Hamill is building an app maybe for a medical context or I'm building one for a legal context, our definition of hallucination is going to be different, right? Maybe different tools or different data that's being used in these applications. And it doesn't make sense to use, you know, the same definition or the same way of evaluating hallucination or an off-the-shelf evaluator to do this.

So, how might we kind of get circumvent using off-the-shelf metrics? How would we do this as a data scientist? That's where we would actually explore the data, look at what's breaking, and try to really specifically name what the failure modes are to our system. So, more concretely, what we found that's really useful is using tools like codecs or clawed code or cursor AI assisted tools to build custom interfaces to load up traces for your agent, actually read each message in each trace one by one, and talk about, you know, what might be going wrong. pretend you're the user of your system, talk to your PM or talk to someone else and be like, "What are the mistakes here?" Write them down as open notes, which we don't show here. But then over time, as you build these collections of open notes, you can then categorize them into the bespoke failure modes that you need. So, for example, in this a real estate agent tour app, maybe some of the bespoke failure modes were it was rescheduling tours whenever there was no or they were hallucinating times for the tours, right? That's something very specific to the app. Pamela and I always say look at your data. And this is exactly what we mean by look at your data. It's it's not just oh look at it and don't say or think or do anything about it. It's build your own interfaces and try to find failure modes and try to do this at scale.

The second metric or this sorry the second failure mode that we commonly see um is really using LLM judges to try to figure out what's going wrong. Maybe you looked at your data and you found a failure mode, but then you simply go and ask an LLM, hey, how often is this occurring in my data? without any notion of trust in that LLM judge itself. So I'm not saying that using LLM judges are bad. I'm saying that using LLM judges kind of blindly without validating your validators is bad. So what do most teams do when they use LLM judges? Well, they ask an LLM, hey, I have this failure mode that I'm trying to uh measure.

Maybe rate how good or bad something is on a one to five scale and then give me the numbers. and then they kind of see this histograms of one and five and then try to make some business decision about that. That's not really good. We found that it's really difficult to trust those numbers or even turn them into actionable business metrics. So what would a data scientist do? Or if you were to wear your data scientist hat, what would you do? Well, you would kind of say, okay, the LLM judge is an imperfect classifier and maybe I should treat finding or training this classifier as I would any machine learning model in the past. So concretely that means taking examples of labeled traces for each failure mode and trying to partition data into you know train development and test sets. If you've you know been in machine learning you're very very familiar with this the idea of separating your data trying to figure out what prompt or what model works really well on your trainer development set and then making sure it doesn't overfit the alignment on a test set. So really what we found is there's no difference here with LLM applications. you still want to go through the same rigorous process. You want to make sure that you don't overfit um your LLM judges.

And in many classification tasks, interestingly, you know, LLM judges, fitting them is an imbalanced classification task, which is a fancy way. There's a lot of jargon for saying that only a small fraction of the trade set is a failure or a failure mode. Um so when you measure alignment with your preferences, don't simply measure accuracy. use some of these metrics that we've designed for imbalance classification tasks like precision or recall or false positives or false negatives to make sure that your LLM judges are aligned.

So those are two pitfalls. The third pitfall that we see is really really broad but bad experimental design really. Um and I'll talk about two ways that we see it. One is people like to generate synthetic data. I love doing that. Um, but they'll do so in a way where all the traces end up looking something like this. Why does that happen? Well, it's very easy to get there when you just ask an LLM for synthetic data like give me data or five generic questions if you're trying to, you know, train a document question answering chatbot. Um, okay, maybe that's not good. So, how might we think like a data scientist to improve this process? We might think about ways in which we want to systematically generate the synthetic data and use LLMs for very small parts of the gen data generation process. So hypothes hypothesize which dimensions vary in the data that users bring into the system. Generate combinations of this data. Review all your synthetic data for quality. Make sure there's good diversity. Look at the data again. And there's a lot of different ways you can think about you know application specific uh ways to generate synthetic data. So one thing that we like to one exercise we like to do with people is just get into your application look at some traces and co come up with at least three different dimensions that vary across users. So sometimes that's the persona of the user. Maybe they're a novice or there maybe is there somebody who's very more much more experienced. use LLMs to generate different values for each dimension and then kind of for lack of a better term take the cross productduct of all of these dimensions to generate your synthetic data.

All right, so that was one way in which you could have bad experimental design and generating synthetic data poorly. Another one is in designing metrics. Um and I talked about why we need application specific metrics but also how we evaluate the metrics is very important. A lot of times people evaluate metrics like score things on a 1 to5 scale or one to 100 scale. Um, and that's not very interpretable or actionable for us. How might we revisit this problem as a data scientist? Well, we'll try to make the problem as least complex as possible. We'll try to make things a binary classification problem.

We're trying to ask the judge for whether something fails or not. A binary classification problem rather than one to 100 or one to five scoring for something like failure. We also want to make these actionable or align with business outcomes. And really, this is really difficult to do, right? It's not going to be something that you can oneshot or get first. Um, especially when you're trying to align LLM judges. Often we see people really struggling, even ourselves, like we can never write a good LLM judge prompt on the first time. Um, but some things that help us are make these LLM judges very very narrow in scope towards binary tasks like passing something or failing something and then label lots of data ourselves to try to me measure alignment with that.

All right, I'm going to hand it back to Haml for the last two. All right, some other pitfalls are and this one happens quite a lot is okay, so as much as we talk about looking at data, a lot of people for whatever reason don't want to look at data and they just outsource it to someone else. And one really common failure mode is okay, let's let the AI engineer or the developers label data. And unless you're building a coding app, it's kind of a bad idea because often those people don't have the domain expertise in whatever you're building or the problem you have. And so you don't want to do this. So, you know, most teams make it someone else's problem, but putting your data science hat on, don't trust anything. You don't trust the labels. Don't trust the people labeling the data. You make sure like whoever's doing the labeling actually has a domain expertise. And then you need to look at the labels because often times there's some kind of problem and what's happening that you are not anticipating.

And another reason to look at the data is this phenomenon called criteria drift. And this is from a paper who validates the validators. Shay has an author on this paper. Should totally check it out. But criteria drift is a very simple concept. It's just they found that people don't know what they want unless they look at some data. Um this idea that you're going to specify a rubric up front and that's enough is really problematic. You need to look at data and you really need to force people to look at the data. Uh and it can feel like this. It is kind of like this honestly.

Another pitfall is automating too much. So you might be thinking in your mind like oh all this eval stuff like can just can't just have Claude do it for me and the answer is no. Like Claude can't read your mind. Cloud doesn't know all the different product nuances of what can go wrong. Sure, there's some lowhanging fruits that LLMs can find in terms of like things are obviously errors, things are obviously broken, but there's a lot of context that you need to externalize to get to some of the like more important things often that are broken, especially the kind of more product failures um you know that might have be happening with your users.

So, there's a bunch of other pitfalls. We don't have we don't have time to go through every single pitfall exhaustively, but let me just give you a taste. So, misusing similarity scores, okay? Like rude, blue, whatever.

Like, you see that on a lot of eval dashboards and it doesn't make any sense. Like, do you really want to be measuring like similarity? Uh, you know, but they come with a lot of off-the-shelf eval frameworks. Um, I always see also number two here, asking the judge, is this helpful? Just like really bad LLM judge prompts that are not specific to your product. Another pitfall is making annotators read raw JSON. You should take all the friction out of looking at data. So, you should build your own data annotation interface that makes reading data delightful. Um, fourth one, reporting unccalibrated scores. So you also want to make sure okay like if you have LLM judge you want to make sure like you study the alignment of that judge with a human being otherwise it's just anybody's best guess.

You have to be also be careful number five to make sure you understand what data criteria drift is happening. Um number six you want to make sure you're not overfitting your judges to data. So kind of think of it like machine learning like you don't want to just take a set of data and hill climb against that data over and like you know iteratively and overfit. You need to set aside some data and make sure that your evals are generalizing.

You want to also make sure that you're sampling data effectively. And you want to make sure like what if you're going to put a metric on a dashboard, be really careful that that metric is earning its place and actually has signal. Um this not necessarily a problem just with AI. This always been a problem, but it's definitely more acute in AI. And so if you really like take a step back like okay a lot of the things that we talked about today there's a correlary kind of skill in data science that helps alleviate this problem. So first thing we talked about error analysis or data analysis looking at your data you want to find patterns in your traces you want to make sure you're reading those traces and analyzing the patterns in them. That's a lot like EDA or exploratory data analysis. We talked about um making sure you have good metrics that are scoped to actual problems.

Um and that's metric design validation like what you would like model validation how you would do in machine learning. You want to make sure your LLM as a judge is aligned with human judgment. Um and then you know you want to make sure that you are curating your data properly, your test data. And then also you want to make sure you're you know you do your monitoring and observability. Um and then you want to approach this whole thing with a scientific mindset. You want to experiment and then you want to measure and try to improve. And like that muscle of a being a data scientist is really helpful here.

So we talked a lot today about a lot of different things. Um, I have these skills that'll help you audit your evals to see if you're doing anything incorrectly. So, don't worry about writing this down, I'll share a link. But most importantly, what if you want to come away with something from this talk is to always look at your data. Now, there's one question everyone has anytime Shrey and I give a presentation is where are the slides? Can I are you sure to share the slides? So, you can get the slides by looking at this QR code. So, we'll send you the slides, but also send you all the memes. There's like a lot more memes, which is really cool. So, thank you.