SDS PODCAST

EPISODE 998: IN CASE YOU MISSED IT IN MAY 2026

Jon Krohn:	00:00	This is episode number 998, our ICYMI in May episode. Welcome back to the SuperDataScience Podcast. I'm your host, Jon Krohn. This is an ICYMI episode that highlights the best parts of conversations we had on the show over the past month. My first clip is from episode 989 Anthropic's recent Claude Mythos preview has reset expectations of what AI can do in the wrong hands, finding and exploiting software vulnerabilities at machine speed. I speak to Anneka Gupta and Cal Al-Dhubaib of Rubrik about why the old cybersecurity playbook of prevention and detection is no longer enough and how AI agents themselves are becoming a new source of data exposure inside organizations. And then now more recently, just a couple of weeks ago at the time of recording, on April 7th, Anthropic announced Claude Methos preview. And so this is a model that famously they haven't released to the public.
	01:04	There's only people testing it privately. Anthropic's own estimate, however, is that mythos class capabilities will proliferate to other labs within six to 18 months with open weight versions to follow. So yeah, the reason supposedly why they didn't release it to the public is because of its prolific ability to identify cybersecurity issues and exploit them if it's not being used by a friendly person. Now I also do think it's brilliant marketing.
	01:35	To say that we carry it to the public. But yeah, from inside Rubrik, is Mythos the inflection or just the most visible point on a curve you'd already been planning against?
Anneka Gupta:	01:48	I think when you look at Mithos, it's definitely the most visible point on the curve that we've already been planning for. Because even if you look at models like Opus, for instance, they also are able to find vulnerabilities. They just require a little bit more handholding to get there. Whereas Methos, you can point

it at an open source repository and tell it to find me all the vulnerabilities and it will find you the vulnerabilities. And this is just the nature of where AI is going. The challenge is that the cybersecurity industry as a whole has been super focused on attack prevention and attack detection because it used to be the case that attackers would enter into your system. They would sit there for weeks, if not months, sifting through your data, finding the opportune time to actually exploit this vulnerability and really perpetrate the attack and then actually do that in a visible way such that the company then has to react to it.

02:50 But with the reality that now with AI agents, you can find and exploit these vulnerabilities much, much faster and at machine speed that requires machine speed response. And no longer can you hope that if you detect an attacker, detect a breach, okay, maybe you can cut that off before the attacker has done damage. Instead, you have to assume that you've already been breached and you need a plan for how are you actually going to recover and ensure that in that recovery you're minimizing any impact to your overall business. And that's the business that Rubrik has been in. So I feel like there's never been a more important time for what we've been doing because what we're ensuring is that you can actually restore your applications, your data, your identity, your infrastructure back up and running extremely quickly. And in the case that an attacker gets into your system and destroys your data, your identities or your infrastructure.

Cal Al-Dhubaib: 03:56 What I find really interesting is it's like this twofold problem and we're kind of feeding into it from the enterprise perspective. There's still massive FOMO. Everyone has to use AI. Everyone has to be more productive and it's coming from leadership down. And so we're very quickly opening up actually the surface area that models that are like Methos can actually chain together vulnerability. So it's one, you've got this problem

of adoption as increasing surface area. And then two, these models themselves, and I know we're going to nerd out a litle bit more about it, but I get so passionate about this. We don't yet have the right trust infrastructure in place in most cases to prevent models from causing harm on their own. So you don't even need the bad guys.

Jon Krohn: 04:37 Right. Yeah. It's pretty wild. I mean, it was Anthropic themselves that actually had, there was some flag, like a Boolean flag that they got wrong in some ... Somehow it was possible for them. You guys might be able to explain this better than me because you actually are cybersecurity experts. Cal's been one for three weeks, but there was like a flag that was set the wrong way in some code that was pushed to GitHub. And so that allowed people to see the entirety of the Claude code base. How does something like that happen?

Anneka Gupta: 05:10 Well, I think that's an interesting case. There's a lot of diferent scenarios where things have been exposed publicly that wasn't supposed to be. There are cases where code has been exposed externally because the AI agents posted it into the wrong repository. That's not what happened in the Anthropic case. In anthropic case, there was, again, a bullying flag that was misconfigured and sent out. But the reality is that the challenge that we have today is that AI agents, they're very outcome focused. They're going and trying to say, how do I get this job done in the least steps possible, the fastest way possible? And they're not taking into account the same rules that us as humans, as employees working in an organization know to follow. They haven't gone through all the security training that we've gone through. They haven't gone through the developer best practices training.

06:01 Now, obviously they have that information in their models broadly, but they're not optimizing for the same thing that we're optimizing for. And what that means is that

inevitably AI agents are going to make mistakes. Inevitably, they're going to do things that you didn't want them to do in their pursuit of this outcome of a task that you've given them.

Cal Al-Dhubaib: 06:22 I find it really interesting that these models, because we've been investing in the capabilities for them to iterate autonomously, think of it as an exhaustive search that's now happening across all of the data assets and tools and configurations of that tool. Humans in the past, we actually had a lot of security through obscurity. We might have had permissions. And I've been studying my cyber stuff. You have. We had security through obscurity and we had access to files and permissions, but we didn't broadly use these permissions to take large scale actions. And now we have these very hyperproductive agentic systems that can do an exhaustive search of everything that's available to it to accomplish the task. And so a lot of cyber controls really were never designed with this reality in mind.

07:14 And it's not just about the mistakes that AI can make or what it can publicly expose. There are other kinds of data exposure risks that you have even internally, right? You could say, Hey, salaries of an employee could be accidentally shared with another employee because you've hooked up your AI agents to your employee and compensation systems. And that is really scary too. So there's a lot of implications both internally and externally to exactly what Cal is talking about. And as the models themselves are able to take just more and more and more and more steps independently, it's really hard to diagnose where in that 20 step process did this happen and what caused it. And you're certainly not going to be able to fix that after the fact. You have to figure out how you're actually going to fix this while this agent is actually running.

Jon Krohn: 08:11 While agents are creating new cybersecurity risks, they also are reshaping how we work productively with data. In episode 991, I speak with Dr. Trevor Mantz about why code notebooks have become the natural working memory for AI coding agents. Trevor walks me through the Marimo Pair skill, which lets you drive a notebook from your agent collaborating with ClaudeCode or Codex in real time as you load, explore, and visualize your data. Beyond the coding, the development experience, you also obviously have a lot of experience doing data analytics, data visualization, data science in kind of a notebook environment. So maybe even walk through for us, you've now talked us through what your workflow is like when you're coding, but when you're exploring data, when you're doing data visualization, it could be interesting for our audience to hear what a power user of Merimo, how they use that kind of tool.

Trevor Manz: 09:08 Yeah. I've always thought that notebooks hold this really special place in the ecosystem because they are this environment that marries your code and data together. And so although I've been sort of this bare bones text editor for my traditional software development, I've always used some kind of notebook when I wanted to do data work or some sort of live Repel. I think my first thing I was in was our studio and then I switched over to Python and started using Jupyter and now I'm a MarineMode user. So I usually would spin up a notebook from the command line and just jump in and start loading my data. And now I find myself with the Marimo Pair skill.

09:49 Anytime I get an issue or often the place that I start to work on a problem is now in my agentic coding tool. So for me, that's often cloud code. If that looks like some tasks that I want to load some data, explore that data, I'm going to start asking to start up a Marimo pair session and then start describing a very high level sort of, these are the data sets that I want to look at. These are libraries I think that I want to do. And sort of being a lot, like I was

saying before, a lot more declarative about the approach that I want to take. I know exactly what I want to do with my data.

Jon Krohn: 10:19 This is maybe a really dumb question, but when you start doing that Marino pair session, that's in the command line or that's in like a notebook environment?

Trevor Manz: 10:27 That's in your agent. So it'd be like open code, cloud code, codex. You can just say like, "Hey, I want to start working on my notebook that the Marimo Pair skill teaches the agent how to start Marimo or connect to Merimo for you and then you're driving Merimo from your agent."

Jon Krohn: 10:44 Whoa. So you'd have it open in a browser window and so you'd be working in, say you could have Claude code running a terminal and so you're typing into that terminal, but then the agent is changing things in your Marimo Notebook. Exactly. It's open separately.

Trevor Manz: 11:01 Yes. And you can also collaborate on it. So you have the Marimo Notbook interface. So if the code that it generates or there's some tweak that you want to make to a cell, then you can go edit that by hand in the Marimo Notebook, but then you can ask Claude, you say, "Hey, I circled something that's interesting in this notebook. Can you tell me about it? " And Claude has access to that state now because of the way that they share this context. So it's like all that rich information that historically has been trapped in that environment now is more context that you can offer to the agent beyond your file system.

Jon Krohn: 11:32 That's so cool. I love that. Does it do things like if you ... So if it had kind of been developing a notebook, had been getting going on some data analysis for you and then you go over to your Chrome browser or whatever and you make a change to a cell, could you end up in a situation where either in the notebook experience or in the terminal

window, it's kind of like it's watching what you're doing and it's just kind of like, "Hey, I like what you did

Trevor Manz: 12:01 There." Yeah, right now, well, we're trying to get our better feedback loops of if you've made edits for us to stream those back. There's a feature inside of MCP that's called channels I think we might explore in that direction. At the moment, it's a lot. Claude has, or your agent has access to running code inside the kernel. So if you've made a change, it can grab any state that it wants inside the notebook. It can see your cells, it can take screenshots of cells now and look at those. So if you say like, "Hey, there's something interesting here, I'm not quite sure what it is, go off and use the other tools at your expense to help me understand what this is. " At your expense. Yeah, exactly. Then that's now context that you can feed back into getting somewhere with your data. So I like to think of it as when I started really adopting agentic coding tools for traditional software development, ideas for like software ideas started to feel a lot cheaper in the sense that, okay, that's not going to take me an afternoon, I could do that really, maybe just do this proof of concept.

13:01 And I've had so many of those times inside of notebooks where there's something I know that I should look at or explore, but I don't want to figure out the API or I can't remember exactly this thing. And instead I can just say, "Maybe we should try these different validations first and then let it work on that problem and then make some plots and bring me back when I can make my decision."

Jon Krohn: 13:19 So far we've looked at using AI agents and securing against them, but what does it actually take to build the foundation models behind these agents from scratch? In episode 995, Jasmia Henry walks me through her work as a full stack foundation model builder. We cover all four stages of the process, the often unglamorous slog of data curation, building bespoke tokenizers and embeddings,

model training and reinforcement learning and then finally the inference layer that serves it all to end users. You describe yourself as a full stack foundation model builder, which makes a lot of sense to me. So like end to end, all aspects of it and you're at the very cutting edge of machine learning research and language modeling research we're going to get into NERIPS papers that you have for listeners who don't know NERPS, it's the most prestigious academic AI conference that there is.

14:10 And so yeah, tell us what you mean by full stack foundation model builder. What is involved other than finding out how many Marlboros it takes to get to a rig, what does it involve?

Jazmia Henry: 14:24 Yeah. So really four steps. The first step is data curation. So this is a step that everybody hates. I actually yesterday sat down with my CTO and we had a very large whiteboard where we were just kind of going through and coming up with a multi-stag plan. We've been working on just data curation over the past six months. We have people who focus specifically on that. And what that means is I might get a textbook of a thousand pages and within that textbook, each paragraph might have a highlighted word and then around the highlighted word, it has a definition and then it has different type of topics and a topic might start off in chapter one and not be talked about again until chapter five and to chapter 20. And so how do you get AI to be able to look at that document and be able to synthesize of like, okay, yes, this thing might have been brought up in paragraph one, but that doesn't mean that paragraph two talks about this thing.

15:28 This is not necessarily important. You need to skip over that when it comes to this and find the next time that's important in chapter six. You can do it different ways. You can come up with different graph databasing strategies and stuff, which are strategies that we use, but

to enrich that data, you can also have somebody who's a subject matter expert who goes through and goes, "Oh yeah, yeah." So the title DV tool means this right here. And then somebody might say DV in this other chapter over here, and then somebody might just say tool over here, but you know it means that when it's talking about hydrostatic pressure. So you want to have somebody who can help make sense of all of these things so that when we're creating that graph database, everything's appropriately attached. So that's the first big portion that constantly changes and we kind of make jokes sometimes that at the end of the day, our company is just as much of a data company if not more than an AI company.

16:27 A lot of it's just building data models.

Jon Krohn: 16:31 Something that we talk about on the show a lot. In fact, in a recent episode of the show, episode 993, we had two book authors on the show. They have a brand new book called Architected Intelligence. Their names are Jacob Miller and Jeremy Mumford. And one of the key takeaways from that episode, if not the main takeaway, is that successful AI deployments are about having the right data for your model. And so data engineering can be a bigger part of any successful AI deployment than the model itself. So agreeing with you 100%.

Jazmia Henry: 17:02 No, absolutely. And then there's that next step, right? What shape do you want the data to go into when you're using these data pipelines? I have built custom tokenizers in our office so that when we have people who are pulling things from our ... So first off, starting off with bespoke tokenizers, but building on top of the embeddings models so that when we have people who are doing other processes, which might not necessarily have to do with AI specifically, but still they need that knowledge base from that graph, they're able to use that and be able to, "Oh, okay, well I have a document, a drilling document." And that drilling document is able to better be chunked and

OCR works better because it's able to identify the information that it needs to pull from that document versus us actually going through and having to put bounding boxes across multiple things within a document we can better avoid that on a text-based level versus having to always defer to a more expensive vision model.

18:13 So that's another portion there. And then there's the fun part, I guess, the part that everybody gets excited about, which is building the foundational model. There's multiple different ways that we do it. We do some continued pre-training, just taking a model that we do benchmarks constantly and experiments constantly on the different types of models that are out there, both open source and the big box ones that you get behind an API. We compare which ones you're doing better at certain tasks. And then for those ones that are open source models that we feel like we can move forward with, do some continued pre-training specifically on oil and gas to make sure that we can extend that ability forward. But then there's also doing some reinforcement learning. A model doesn't understand the world around us and so being able to put physical validation around it by creating physics environments and saying, "Okay, go ahead and do these things.

19:15 I want you to use Python Rebel to create some code that's doing the mathematics." Yes, but it has to be defined within this space of physics. I can take a very long time to try to teach some model physics, but then it's going to overfit to physics, especially if I have a smaller model and then somebody put oil and gas on it and everything kind of falls apart, or I can just have the very basic rules of physics that are unchanging inside of an environment. And so doing that type of stuff, figuring out what things can we strip out from model training and instead put it in reinforcement learning is something that we do a lot of stuff with. And then the last part is inference, which is the I find most exciting, but a lot of people find less

interesting. The way that you train a model is going to affect the way you have to serve a model.

20:04 And so if I do some really cool environment that's super amazing and makes the model do better than all the other models out there and I'm like, okay, I love this reinforcement learning thing, it's great. Well, reinforcement learning models are bursty. And so that's going to affect the compute level that we can use when we're doing inference. That's very pricey. So suddenly you have to have this inference thing on so that you can keep your uptime at 99.9% because you have SLAs you got to stick to, but you have this huge cluster that's supposed to be able to handle that burstiness. So being smart with how am I building the model and how can I serve it to the public so that they can get the best results from the model?

Jon Krohn: 20:46 We're rounding up a great month with episode 993 in which I sit down with Jeremy Mumford and Jacob Miller, the co-authors of the brand new book, Architected Intelligence. Jeremy and Jacob argue that the most expensive AI mistaken organization can make is failing slowly, sticking with prototypes long past their sell-by date because the traditional software mindset says you have to. We discuss how the define, build, feedback loop changes when going from zero to prototype is nearly instantaneous and why narrowly defined lanes of ownership are one of the best antidotes to the organizational antibodies that fight against failure. Another part that I like about the first chapters of your book is how you write that failing slowly is costly and failing quickly can become your superpower. So if you're in an organization and we're going to get to speed again, well at least a few more times actually in this episode, speed is kind of a key thing today it seems, but being able to fail quickly is something that's so important.

	21:53	However, organizational antibodies like the health system, something that was designed to keep an organization healthy, these structures in an organization, let's call them antibodies to give it kind of like a biological analogy, they actively fight against failure. They're always trying to make sure that everything succeeds. And in this day and age where things are moving so quickly in terms of technologies, capabilities, what you could be doing with your platform, that historically useful antibody is now, it's like an autoimmune disease. So what are your tips on forcing failure fast on AI projects?
Jeremy M.:	22:39	It's an interesting problem because I think enterprises, they have these organizational antibodies as you describe and they start to really want to stick with projects and ideas maybe longer than they should. And I think also they are kind of accustomed to the traditional way of software engineering and building products, which is that the initial product will maybe not have full functionality, but if you stick with it long enough, you can build out to the point where it's useful. But now building with AI and building AI infused products and features, going from zero to prototype is nearly instantaneous. And so it requires a mindset shift and an organizational behavior shift that can be hard to execute at the enterprise level. Just to give an example, I remember early on at Pattern, we wanted to build a text to SQL chatbot and having interviewed hundreds of people for AI engineer roles and talking to people across the industry, I swear every single company tried to do this.
	23:47	We all tried to build some sort of text to SQL chatbot that would just solve the problem of being able to retrieve data from our endlessly complex databases. And the thing about all these demos is that within like a day you could have an amazing demo that seemed to kind of have something amazing on the surface and so a lot of companies kind of committed to that and they were like, "Well, if we just stick with it, we'll be able to kind of keep

going with it. " I think the key thing there is that now that it's so easy and quick to build prototypes, you really need to prove value on day one. If you can't prove value on day one, then you need to toss it out and start all over again. And that's a really hard thing to tell people to just toss out their ideas, but I think we're seeing that shift.

24:37 I look at AWS just this last week, they've kind of announced that they're doing this massive shift from their old AWS bedrock system to this new project mantle. And they talked about how they had six engineers over like two or three months just completely rebuild their entire AI infrastac from the ground up. That's the kind of commitment and resolving you to have in this new age to just say, "We got to start over from scratch and build something with immediate value from the beginning." And so that's really what I think it comes down to.

Jacob M.: 25:08 With the one approach when we're creating something of value, the product cycle is that you define what we want to build, we then build it and then we get feedback and it used to be that building was the bottleneck. So you could be really bad. So if it's defined build and feedback and if build was always the bottleneck, you could always just be bad at definition and feedback because that was never the bottleneck anyway. And so when building no longer becomes the bottleneck, the critical, the crucial step is definition and feedback. And so one of the organizational changes we've seen help is actually narrowly defining these lanes and say, "Hey, you two, three people, you can make whatever decisions you want. This is what you're chasing after. This is what success looks like. " And you guys can try and fail over and over and over within that lane.

25:58 And by involving fewer people, it ends up that those who are working on the project feel much more comfortable trying things out and failing because you're not failing in front of such a large audience. All

Jon Krohn:	26:09	Right, that's it for today's in case you missed an episode, be sure not to miss any of our exciting upcoming episodes, including upcoming episode 1000 subscribe to this podcast if you haven't already, but most importantly, I hope you'll just keep on listening. Until next time, keep on rocking it out there and I'm looking forward to enjoying another round of the Super DataSciencePodcast with you very soon.