Transcript provided as a free resource by

Data Mesh Radio Episode #120: Applying ML Learnings - Especially About Drift - To Data Mesh

Interview with Elena Samuylova
Listen (link)

Transcript provided as a free community resource by Starburst.
To check out more Starburst-compiled resources about data mesh, please check here: https://www.starburst.io/info/data-mesh-resource-center
To get their Data Mesh for Dummies book (info gated), please see here: https://starburst.io/info/data-mesh-for-dummies/

0:00:00 Scott Hirleman
The following is a message from George Trujillo, a data strategist at DataStax. As a reminder, DataStax is the only financial sponsor of Data Mesh Radio, in the Data Mesh Learning Community at this time. I work with George and I would highly recommend speaking with him, it's always a fun conversation.

0:00:18 George Trujillo
One of the key value propositions of data mesh is empowering lines of business to innovate with data. So it's been really exciting for me personally, to see data mesh in practice and how it's maturing. This is a significant organizational transformation, so it must be well understood. Empowering developers, analysts, and data scientists with downstream data has been part of my personal data journey that reemphasized the importance of reducing complexity in real-time data ecosystems, and the criticality of picking the right real time data technology stack. I'm always open and welcome the opportunity to share experiences and ideas around executing a data mesh strategy. Feel free to email or connect with me on LinkedIn if you'd like to talk about real time data ecosystems, data management strategies, or data mesh. My contact information can be found in the notes below. Thank you.
LinkedIn: https://www.linkedin.com/in/georgetrujillo/
Email: george.trujillo@datastax.com

0:01:11 Scott Hirleman
A written transcript of this episode is provided by Starburst. For more information, you can see the show notes.

0:01:18 Adrian Estala
Welcome to Data Mesh Radio, with your host, Scott Hirleman, sponsored by Starburst. This is Adrian Estala, VP of Data Mesh Consulting Services at Starburst and host of Data Mesh TV. Starburst is the leading sponsor for Trino, the open source project, and Zhamak's Data Mesh book, *Delivering Data Driven Value At Scale*. To claim your free book, head over to starburst.io.

0:01:48 Scott Hirleman

Data Mesh Radio, a part of the Data as a Product Podcast Network, is a free community resource provided by DataStax. Data Mesh Radio is produced and hosted by Scott Hirleman, a co-founder of the Data Mesh Learning Community. This podcast is designed to help you get up to speed on a number of Data Mesh related topics, hopefully you find it useful.

Bottom line up front, what are you going to hear about and learn about in this episode? I interviewed Elena Samuylova, who's the co-founder and CEO at the ML model monitoring company and open source project, Evidently AI. This bottom line up front is quite a bit different from other recent bottom line up fronts. I'm gonna add a lot of color on not just what was said, but how it could apply to data and analytics work, especially for Data Mesh, so taking a lot of learnings from machine learning and applying it to data and analytics, especially Data Mesh. It's a bit theoretical in nature, but I think it provides a lot of food for thought. So some key takeaways or thoughts this time specifically from my point of view, based on what I learned in this conversation.

Number one, a good rule of software that applies to ML and data, especially Mesh data products, if you build it, it will break, set yourself up to react to that, act accordingly. Number two, maintenance may not be "sexy" but it's probably the most crucial aspect of ML and data in general. It's very easy to create a data asset and move on, but doing the work to maintain is really treating things like a product. Number three, ML models are inherently expected to degrade. When they degrade for a number of reasons, they must be retrained or replaced. Similarly on the Mesh data product side, we need to think about monitoring for degradation to figure out if the Mesh data product is still valuable or worth the spend especially, or how to increase that value, in the return on investment, not just the return. If it's becoming more and more expensive to maintain, should we move on from it?

Number four, data drift, which would be changes in the information input into your model, such as a new prospect base, so what you were trying to actually react against has changed. It's a crucial concept in data and analytics too, are we still sharing information about the things that matter in a way that is understandable? Are we encapsulating what's happening in the real world in our Mesh data products? A lot of what Andrew Padilla talked about in his episode. Number six, concept drift feels similar to semantic drift in the analytics world. So we can look to potentially take deeper learnings from how people approach and combat concept drift from ML and apply it to Data Mesh. I haven't seen anybody specifically writing how to do that, but I think it's important to start to look to.

Number seven, how can we monitor degradation in Mesh data products and prevent that degradation in our data and analytics work? Historically, reports drifted further and further from reality with no intervention because the pain of change was so high. Are we fully reliant on the domain to know when something has degraded? Can we use software to help us detect semantic drift? Very early days on that one. I've heard it come up in a lot of conversations, but I don't have a good answer. Number eight, ML models are designed to do one thing very well. Unfortunately, we don't have a good framework for reuse at that model level in ML that we could apply to Data Mesh. Maybe at the ML feature level? I'll talk about that in a second. Number nine, ML models have expected performance metrics. Those expectations need to be set through conversations between the business team and the ML team. So this would be like how well are we looking to raise revenue from using this model or what's our expected conversion rate? So measure using KPIs. Can we use a similar approach to expectations, at least for some specific use cases, for a Mesh data product?

Number 10, When building an ML model, you need to consider scope, business purpose, expectations, measurement against expectations, etc. Similarly, when doing any data work, you should consider the same. It is somewhat hard to measure the impact of most Mesh data products, but it doesn't mean you shouldn't try. What are you trying to achieve with the data product and is it meeting those expectations? Is the business need still relevant or has it changed? This is kind of important when you're thinking about, should we actually create a data product? If we serve the use case, is that going to provide a lot of value?
Number 11, regarding graceful evolution and preventing breaking changes due to changes in sources or causing downstream breakages from changing the ML model and/or its inputs and outputs. ML unfortunately does not have any answers that we aren't already using on the data and analytics side. Good communication, contracts, monitoring and observability etc. No silver bullet or MLMFD, which would be Machine Learning Magic Fairy Dust. Doesn't exist, unfortunately. Number 12, the concept of a feature in an ML, a smaller component of the model that might be reusable across multiple models, could be interesting to consider in Data Mesh. It would likely break with Zhamak's view of each data product owning its own transformation logic, but could create almost proto transformed data, almost like a service bus to easily serve data products. It probably has a lot of drawbacks. I do not recommend this, but it is interesting to consider and think about.

Number 13 guardrails on ML models help to keep the models from doing things like reacting to data that is out of the norm. As Elena said, if an ML-based recommendation on our website is a bit off, the recommendation engine, the conversion rate falls, but that's not the end of the world. If Amazon recommends that you buy another toilet seat after you just bought a toilet seat, when you're like, "Why would I buy another toilet seat?" Okay, their conversion rate falls, but it's not a big deal. But what if you are dealing with big dollar decisions, should we look to proactively put in guardrails into our data products? Probably, yes, if they're driving crucial decisions. Consider failure modes and what to do in those cases. We need to really think about, should we have guardrails to make sure that even though we might think that the data processing is correct, that we don't want to push data out that really has changed significantly. This is a big thing in the observability space. And finally number 14, getting to fast incremental value is crucial when developing ML models. There needs to be very good trust and communication, so people understand the initial quality level might not be great as you iterate towards a better model, or Mesh data product. This is becoming a common theme. How can you release a version 0 or even 0.01 of your Mesh data product and still drive value now while getting it to that quality level of 1.0?

So I think there's a lot to learn from this episode and think about how could we apply what we've learned in good practices, best practices in machine learning, that we could as well to data and analytics specifically in Data Mesh. With that bottom line up front done, let's jump into the interview. Okay, very, very excited for today's episode, I've got Elena Samuylova, and I apologize if I murder the name, please feel free to correct it, but who's here. She's the CEO, co-founder of Evidently.AI, which is one of the few that I've seen that is actually open source software in the ML, MLOps, ML monitoring space, which I think is awesome that there's some open source out there, and that's kind of what we're trying to do with the podcast about open source information and all of that.

So we're gonna be talking about a concept that's been coming up more and more in general, in Data Mesh talks about drift. Because historically, the cost of change in data has been extremely high. We haven't been able to actually monitor for when things are changing because, well partially, we haven't really monitored for when things are changing because the world hasn't changed nearly as quickly as it is now. But also the cost of change was so high that people just said, "Okay, I get it, something has changed. We're gonna wait until it's totally broken, or totally not relevant to make a change," and that's what we're trying to get away from within Data Mesh and these gentle evolutions as to make sure that we're staying on path with what's actually happening in the real world, instead of just keeping the same report going.

So kind of a long intro to this, but I think it's really important for people to understand these concepts and the ML space, because of how much information that's flowing through it, has so much more depth around drift. And so Elena has been very, very kind in offering up her time to help explain this topic, and I kinda kept the newbie mindset on this rather than trying to dig too deep. So hopefully, I'm not talking too stupidly about this stuff, but I know some of the audience members, it might be a new concept to them. So again, too long of an intro, but Elena, I'm very excited about this. If you don't mind, if you could give people a bit of an introduction to yourself, and then we can jump into the conversation at hand.

0:13:09 Elena Samuylova
Thanks Scott. It is indeed a very exciting topic. So as mentioned, myself and co-founder, we are working on an open source tool that helps you want to machining models in protection. So we're starter founders, probably what is interesting is where we're coming from, because before we embark on this journey to build Evidently, we have been working together for many years, creating machine learning solutions and deploying them in production. Actually, we started fairly early, as long as 2014, when it was still a fairly new topic. People didn't even call it "AI" back then, it was still “big data” sort of thing, and we are working on deploying solutions in different industries, from manufacturing to retail to finance, you name it.

In our previous startup we specifically focused on industrial sector, working with companies like oil and gas, steel, manufacturing companies of all sorts, helping them create solutions that optimize industrial processes. So what we've seen and what I've seen personally and participated in was basically the development and deployment of these models in the real world, and witnessed all the proliferation of tools that appeared recently in the ML space and evolution of the industry. So I'm happy to share my insights and my learnings from this, and specifically on the topic of model maintenance, because that's what I'm excited about, basically everything that happens after you create a machining model and you finally start using it. Which is a somewhat boring topic, but I find it really exciting.

0:14:35 Scott Hirleman
But I think this is really crucial because people think that you model once and then it's done, but exactly what you're talking about of what's happening in the real world. Has this changed? What's going on with this? What's actually happening relative? If we're doing something that's so crucial to the way we interact with the world, the way we interact with our customers and our information in real time, it's so crucial. So let's start with a general overview of what is drift? What is it? And what causes it?

0:15:14 Elena Samuylova
So first of all, let's define the terms, because I'm sure there is some understanding of what's drift is in traditional data analytics. In machine learning, it might have a slightly different understanding. So usually we talk about machine learning models drift, which is like a general term that says that the model performance degraded. It suddenly or gradually starts behaving worse, and it doesn't bring the value that you expect it to. And there are usually two reasons for that, that you might choose from.

One is “data drift,” which is the situation when the model is now applied to different data that has not been seen when you created this model. So imagine that you were predicting something and now you're making these predictions for a new population, because maybe you have customers from a different location or with different behavioral patterns, and your model will not perform as well. And another situation is so called "concept drift", when basically the real world relationship that you model is modeling are changing. For example, it might be a change in how you log the data in your app. Or maybe there is a pandemic and everyone and everything is behaving differently.

So there are multiple examples depending on specific applications that you can think of, but all of these changes affect the performance of the model, leading to it not performing as you want. It might be less accurate, that might be giving incorrect predictions, and effectively it doesn't bring the return on investment and the business value that you expect it to. And this is something that with machine learning, you expect to happen. You know that no model lasts forever. This will degrade and become worse with time, so you have to factor it from the very beginning. How are you gonna update and retrain this model when it is in operation?

0:16:54 Scott Hirleman
How do you... I think this is something that when you think about product thinking, ML seems like it's a much faster kind of product concept of churn and burn, that these models come in and go, and come in and go, but how do you start to think about how long models typically last? Is it just completely all over the map? Or with how fast things are changing in the real world, how do we really think about that?

0:17:29 Elena Samuylova
Of course it depends. So there are some models that are pretty stable, so for example, in manufacturing that I was already mentioning, you would often work with a very stable environment, so things don't change as drastically on a production line. If you're using this data that's coming from some sensors, then it might not be changing in a minute, right? In some situations when you're working with behavior data like user data, it might be changing every day quite a lot, so you would need to factor that in. And actually you can even measure it, so when you're creating your model, you're using some historical data that you're training your model on, and you can actually relate how different it is and how it changes with time.

But of course, depending on the application, depending on how you've built your model, the speed of model degradation rises. And here we're talking about natural model degradation, so something that we kinda expect to happen because the world changes, the model will not change with it automatically, but then there are also some drastic events that you cannot factor in, and a pandemic is a perfect example, like a telltale of some drift happening everywhere. But it might be something that affects your specific domain, like I don't know, change in interest rate would affect some credit scoring models, and some situations that might just happen one day overnight and it will affect your model, so you need to be able to catch these sudden bugs as well.

0:18:49 Scott Hirleman
I think that's really interesting, and I'm trying to think about how we take that same concept and think about just kind of general analytical models, because what we're trying to do with Data Mesh is share what's actually happening in the real world. And so what's happening in the real world is obviously changing, like you said just with these ML models, but I don't think I've seen many people talk about this, and semantic drift, which is kind of the concept drift. But it's also what the concept of something is changing as well, and I don't know that that's flowing as much into the ML model because you are replacing them. There is planned obsolescence.

And when you start to see models drift, is it that you kind of re-compute them and still try and have them do the same thing? Or is it more people rip and replace? How do you start to think about what typically happens? I know it's probably all over the map, but is it that people just keep trying to replace the same model and then what they're actually trying to do it for is no longer really that relevant or it doesn't drive the needle, and so they're putting a lot of effort towards something that doesn't matter, or?

0:20:13 Elena Samuylova
In this case, the usual action is to retrain the model. Meaning that you basically take the newer data that you've collected and you put it pretty much in the same processing steps that you used when you created the model in the first place. But then it might not work, you might actually need to rebuild the model, right? So it might be like adding additional data sources, using a different modeling approach, maybe testing out different features and so on. It depends. It usually starts with the first one, then if it doesn't work you would try the second one. It might be that the model is just becoming so irrelevant that it's better not to use it at all. Right?

You can switch to some human decision making, if you were trying to automate something, you can go back, right? Or maybe you can use some fallback strategy, which can be like an alternative, maybe rule based decision even. So there are these actions that you can take. Usually it starts with your training. You can also rebuild the model. You can just stop from using it.

0:21:07 Scott Hirleman
And how do you think about something like semantic drift coming in? Like what

we're measuring doesn't matter nearly as much. Is there some kind of way that you measure, is this still having the impact that we expect it? You talk about the model itself degrading, but it might be that the thing that we're interacting with is becoming less relevant. Right?

So you say, "Okay," you're talking about... Like I used to cover semi conductors, and you might say, "Okay, the cost of a semiconductor more and more started to move towards software, rather than the actual physical piece of the silicon. And so the cost of silicon fluctuations didn't nearly as much impact the actual margins of these companies, because less and less, it was less and less important of what was going on. So if you were to really measure that back in the '60s or '70s, that might greatly, greatly affect margins 'cause the companies couldn't raise their prices if things went way up, but they also didn't have to cut their prices very quickly if the cost went way down.

So, do you start to think about... This kind of almost feels like monitoring on monitoring on monitoring, and I think anybody who's familiar with the MLOps space has probably seen the Spider-Man meme of the multiple Spider-Mans pointing at each other with monitoring tools and things. But just I'm trying to understand conceptually, what is the thing that typically creates the issue where the model is no longer nearly as impactful or effective? Is it the degradation, is it the concept drift or data drift, or? What are you typically seeing? 'Cause you're talking to so many different companies, I think it's useful to get that.

0:23:08 Elena Samuylova
Yeah. I think there are two different issues and they're really, really on different sides of the scale. So one is when the model is no longer relevant because you're just solving a wrong problem, right? So you may not need the output of this model because you know how to use this. And it happens pretty often, especially if you started with some sort of R & D approach, you create some model, you deployed it, and then you actually learned that no one is using it, or people are using it in a completely wrong way that you did not foresee initially, right? They just pick the output of the model and use it in a different way. So this is more of a kind of product problem, if I may say. So because a machine learning model is usually either a part of some product or feature of it, or is a kind of stand alone application that solves a specific problem. And if you're solving the wrong problem in the wrong way, then the model is not relevant.

And in this case, the hard part is measuring and solving it, because here we're talking about measuring the business KPI or the product KPI or the business impact of the model, and it's not always that straightforward. You might need to run an A/B test, you might need to work specifically focused on this use case. So this one, this first problem is basically building a wrong problem or using it not in a way that it was intended.

And second situation is when you're actually solving a really important problem, so it costs something to your business, you're either increasing revenue or you're helping decrease costs, you have been told to automate something, but somehow this model is not bringing the desired value, because maybe the quality of the predictions are not as expected, you might be dealing with low quality data, so in the end your model is not working because the inputs are not at the level that you need them. This is something that can be solved, can be worked on, and you need to catch the specific bug, specific issue that the model is facing. It can happen on multiple levels, so it can be purely software bugs, literally like your service might be down, right? It can be data quality issues, and this is I think something that intersects a lot with the whole general Data Mesh and data analytics world, because we're talking about the quality of the data assets. In this case, one particular product, which is the machine learning model that consumes them, and then there are model quality issues, which would be the basically model accuracy or model decay, that you might reveal them late.

0:25:28 Scott Hirleman
Yeah, and I think that "Is it still even relevant?", I think that's a difficult thing to test. Do you have any advice on people trying to... 'Cause I think what we haven't had in the data space specifically is that we've actually had the concept of, "Let's shut this down." That we haven't had that product mindset. I think ML, the cost of running ML, people are much more aware of it because it is something that is degrading and that you're constantly having to kinda keep up and running, versus people are like, "Eh, we're just running this report, it's fine if nobody's really using it anymore, 'cause the cost of the report isn't that huge," or blah, blah, blah. Which typically it is because somebody's using something that's no longer any good.

But how do you measure that relevancy, and how do you think about that again, with was it relevant, to when does it not become relevant? And there are all the memes out there around your P score for 0.501 and "Oh, no," and 0.499, "Yay, we got it under the 5% P value." How do you think about talking to people about the relevancy?

0:26:53 Elena Samuylova
Ideally you should talk about this before you build a model, because identifying the business KPI and how you're gonna measure the success, like if this model is built and put in production, is the critical problem that you have to solve. Sometimes you might figure out that actually you don't know what you're doing, exactly because you're expecting just some miracles, like some oracle that you can ask any question

and it will predict you anything.

I think we have already moved from this space, but it still happens sometimes. Managing expectations of the business stakeholders, understanding how to solve a particular business problem with machine learning, is a huge issue, right? But in an ideal world, when you create this model, you already choose something like a business KPI and you translate it back to the model metrics. So accuracy is like a simple example, they can be more specific metrics depending on the type of problem you solve and so on. But when you create this model, you would then test or evaluate it on some historical dataset or during the experiment, and then you would still need to monitor this quality metric. It is very important to choose the metric right, because there are many caveats, there are metrics that are not good. You know, if you want 99% of something, you can find a metric that will say 99% of something. It depends on how you frame it. But yeah, so you should have a metric, and I think this is a big difference between machine learning and the data world, because in machine learning you always have some metric because your model is optimizing this metric. Right?

0:28:16 Scott Hirleman
Yeah. I know it's a little bit difficult to try and pin you down on, okay, tell us exactly how you measure relevancy in ML so that we can apply it to data and analytics, but outside of the KPIs, do you have things that you've seen where people could look for red flags or anything like that around relevancy? Because I think especially when you start to talk about even the features within a dataset. Is this still relevant to what we were trying to... This is almost the semantic drift concept to me. Is this still relevant to what we are trying to display, what we're trying to share of what's going on? And that's slightly different than is this still relevant to the business?

And so that semantic drift around, "Does this still say what we're trying to get it to say?" is there anything that you've seen in the ML world? I don't know, is there anything where there's not necessarily even a direct tie or a direct correlation, but anything that you've learned from? 'Cause you've seen so many times when you put in new interesting things around data, like the ML world, you're putting way more out there more commonly, more frequently than anybody else.

0:29:38 Elena Samuylova
I still think there is a pretty big difference here between machine learning and data in general, because every machine learning model is purpose built. Most of the companies do have, not millions of these models. I sure can mention companies having thousands of data tables and dashboards easily. When we talk about machine learning models, it can be dozens, in big companies maybe hundreds, but these are not like unlimited numbers.

That's why every time you create the single model, you start with defining this KPI, and this is probably different from a case when just recording data just in case, or creating tables because you can, creating data first because you can. But circling back to the question of how to identify if the model is relevant, or in general if you're solving the right problem, I think the biggest issue is always the communication with the business stakeholders. Especially with data science, it used to have some sort of R & D attitude, when you're just like trying to solve interesting problems, looking at the data, trying out interesting algorithms, and there is a certain disconnect between this and what can actually move the needle for the business, right?

And how you bridge this is always through communication with business stakeholders, and business stakeholders sharing information, actually helping data scientists to formulate the right problems, to choose the right features, to interpret them the correct way. And sometimes there is a sort of an issue, you know when people just throw data to the data science team, let's say machine learning team and say, "Hey, your AI is very smart. Just figure it out." Trying to solve this kind of communication gap is usually a big deal, and I think embedding data scientists and machine learning engineers in business teams is something that works, and we have seen successes when companies move from the centralized data team to embedded teams, because it helps exactly with this communication. So how do you build the right model, how do you solve the right problems, how do you interpret the features that you're using in the business context, without just expecting algorithms to figure everything for themselves. So it is always a communication problem, that's probably shared across so many domains, not just machine learning.

0:31:55 Scott Hirleman
Yeah, I'm laughing here 'cause I'm like, I guess I am kind of asking you for the ML fairy dust of, "Here, just what can we sprinkle on this to make it so that we know whether what we're measuring is still the thing that we thought we were, what we're sharing?" And so I think that it's funny how many of the podcast episodes of people who have really dug into what works and what doesn't, it just keeps coming back to communication more than anything else, and so it's pretty funny that that's such a thing.

One thing I think that ML has really, really figured out quite well, that the data space is still trying to figure out, is again, the cost of change around data has been so expensive, and ML has figured out how to do graceful evolution. I think some of that has been learned from the software side, where software, I don't know that it's even graceful evolution, 'cause a lot of times it's just like, "We're gonna just drop all this, these 10 columns and whatever, and it's not a big deal 'cause we're not really using them anymore," and blah, blah, blah. But is it that you're planning to re-train and so people should think about that they've got to just completely rethink the model, but then that creates issues for the consumers. So how you think about that graceful evolution of... You are also, ML is dependent on upstream, and then downstream things are dependent on it. So you're kind of in the middle, and that's kind of where data is gonna be with Data Mesh, is you've got your upstream sources that are evolving, and then you've got your downstream consumers where you can't just break all their stuff, or that you've got to give them affordances to make it so that the conversion and things like that.

So big big question, kind of an obnoxious one to try and throw it all into one thing, but just we'd love to kinda start the conversation around that, maybe dig into a couple of different aspects?

0:34:00 Elena Samuylova
Surprisingly, machine learning hasn't figured that out yet in many cases. I've talked to companies who said like, "Hey, we built a model and then at some point randomly understood that it's not working, because we're actually did not monitor and it was the end user who came complaining." So these things still happen, so usually would wait for the first disaster like this to start thinking about monitoring. With the exception of the companies who have a really strong engineering culture and they deal with realtime models because it's over their software service, you will inevitably have monitoring, at least on the software side.

But yes, you're absolutely right, that there's those kinds of downstream and upstream issues that you need to take into account and prepare to catch, and you start of course with enabling this monitoring, so how do you actually know what something is wrong. You come from both sides, so one is from the data ingestion side, you need to have some sort of data validation and data contract, probably like a good example of reference to Data Mesh here, right? So you need to have an agreement with the data source that your model is using, and to make sure that you can check and catch some bugs and also maybe incorporate some fallback. With fallback, I mean that there is some way of making the decision in case you, for example, don't receive the data. If you're dealing with a very risky use case, you need to be able to predict something even in this scenario, right?

And another one is the monitoring of the actual model performance, so as soon as you have the true labels, you know if what you predicted is right or wrong. Or sometimes even before you know that, you can already look at the model output and believe if it's sufficiently good to act on it, you need to be able to monitor this downstream impact, model quality, business KPI, and again, factor this in. If something is wrong, you should go and rebuild the model, stop the model, or somehow intervene. So you have to monitor both side and prepare to build processes and tools that kinda keep this going, so the data scientist who builds the model stays engaged in the model, because if something is wrong, he would be the person or she will be the person to take care of it. So it's not a one and done thing.

This is something that many companies still have to internalize. Right now there is no such role, with the MLOps engineer focused on monitoring and maintenance, it spreads around the teams. I'm sure in the future we're gonna see maybe with companies that have a lot of models, that will be like a separate team to take care of the model maintenance. But this is a big piece. It's really time, right? When you have the model, it finally pays off and brings you the business value that you expect it to bring. This is the most crucial part to ensure that this happens.

0:36:41 Scott Hirleman
When you think about... You talked about the monitoring and the contract agreement with the source, but that source is going to still have evolutionary changes. Right? And so within Data Mesh, a big part of this is that you don't have unexpected changes or that you have much fewer of them, that people actually understand somewhat their downstream impact. But there still is going to be changes. Are there any engineering techniques that people have developed? Or is it more just like, "Oh, our source is changing, we gotta figure out how to deal with that."? Is there a way to gracefully prepare for that so that you are... You talked again about the contracts against the monitoring and against fall back of, "When this happens, do this."

And so it's almost like a rule based around your ML, and I've talked about this where you can't model out all the potential issues, so you just say, "If it's not within this bounds, ignore it. Or react in this way if it's not in this bounds." If you're talking about real time pricing and it's like, "It's not within this bounds." Okay, well, you're not gonna drop the price 99% 'cause you got this weird thing that flowed through, don't do that, that's just stupid. Versus maybe you do spike your price and it's like, okay, nobody is just gonna buy, but you're not also killing yourself from a profitability standpoint.

Have you found that there are good techniques or ways that people can look at, so that they are prepared? Because again, with Data Mesh, the operational model that we're taking our data from changes, and we're now having more intentionality around that, where the people that are handling the operational model or the operational database or whatever, that they're understanding that their changes have impact and what those impacts are. But like, there are still gonna be changes that happen. Have you found anything where people are okay, or make it so that those changes aren't nearly as painful?

0:38:55 Elena Samuylova
It does depend on the use case, because I mean there are some situations, maybe it's just content recommendations, if you show some irrelevant recommendations, I mean, it's not a big deal, right? You maybe lose some conversion, but no one gets hurt. And then there are the use cases when you can actually lose a lot of money by giving this 100% discount on something, right? Or if you're working with a production line, I mean like your whole production line becomes crap. So of course, depending on the risks of the use case that you're dealing with, you need to have different levels of guardrails and monitoring and this whole alternatives available to tackle this. And of course, the more important the use case, the more you have to think about this. And I truly believe, it's kind of a product problem, because it is the solution design that goes beyond machine learning, that factors in the specifics of the use case.

And unfortunately, most of the solutions right now are more of a manual kind of policy layer. There might be some smart solutions, like for example, you like to fill some averages, instead of the data that you didn't receive, that might make sense, right? But it still requires understanding of what you can do in the specific use case. Maybe you can apply some default prediction. Maybe you can just send it for manual review to someone who can make an expert judgment instead of using machine learning in this case. So there are all these options, but they're mostly in the products field rather than engineering. So the engineering solution is to catch this. Do not miss it out, right? To not just generate some prediction that you should not act on.

0:40:24 Scott Hirleman
Again, I'm looking for you to have the fairy dust.

0:40:27 Elena Samuylova
I know.

0:40:28 SH:
"Here's how we can just deal with changes and be very resilient to change." But it's also good to hear that these challenges aren't solved somewhere else and we just haven't really extracted them from it. Because that is kind of the way a lot of the things around data have been. With Data Mesh, it brings in like, DevOps and microservices concepts and stuff of, "Hey, really the only way to continually scale these things when you're at a significant scale and things is loose coupling, right?" You can't have things that are tightly coupled and then you just keep adding more and more tightly coupled things, because then everything is rigid and nothing can move. So you've gotta move to loosely coupled and moving to loosely coupled is not easy, but it's the only way that you can have this stuff really start to scale.

Yeah, so I think one other question that I had. You've been on the ML side rather than the data side. What are some good ways that you've been thinking about how ML and data can better collaborate, interoperate with each other? Originally, when we were first talking about this, you were thinking about what can ML learn from Data Mesh and I was thinking about what can Data Mesh from ML. And so how can we better collaborate, not just information sharing, but if you had a thing that data people do really poorly for, or ML, is it that they're always expecting the model to be perfect and that the upstream changes do make it change? Or how can ML help you... Or how can the data folks help the ML folks? And how can the ML folks help the data folks and collaborate better?

0:42:23 Elena Samuylova
Yeah, I really hope that there will not be such division, you know, like ML folks and data folks actually sitting in different roles, because I do believe that they actually need to work together. Take a business use case like churn prediction, you might be able to solve it with a predictive model, which will be fancy and use machine learning algorithms. You can also learn it with traditional analytics. And how they actually decide who's gonna apply this or that algorithm if you pre define machine learning? You might be a hammer in search of a nail. So you should actually, I think approach many data problems with an open mindset. And sometimes the solution is to create a machine learning model and system on top of it. Sometimes it's just traditional analytics. So actually, I would not build a fence between these two worlds.

But what's interesting is that somehow, for example, in terms of tooling, we do have a modern data stack, or probably already postmodern data stack. We have an evolving main learning stack, but they are somehow not exactly sticking together, they're developing independently, because there are two groups of very smart people that are working on these things, and somehow they don't really talk a lot to each other. So I would actually imagine on the industry level, it would be interesting to collaborate a bit more. And on the company level, I think you kind of hit the right point with the communication and like data updates, because this simple situation when someone changes something without taking into account the potential downstream impact of who else is using this data stores and so on, or just throwing the data that you have without explaining what the field works or what the fields mean, this is so commonplace, right? It's almost ridiculous that every time when we talk about fencing, monitoring algorithms, a lot of the problems that we find with this are little problems, we know like someone didn't update someone about this thing changing. And I think this is everywhere, right? In machine learning, in data and probably in many other analytical fields.

0:44:20 Scott Hirleman
Yeah, if you see something, say something. If you think that... And we haven't had the

communication to know who, and the tooling to know who is actually consuming from downstream, in a lot of cases. I talk about this with the software developers. They have to make changes to actually do their job, and in most organizations, they can't know what those changes are gonna do 'cause they don't know who's consuming downstream, and not just even first level downstream, but 10th level downstream. So their changes are gonna 'cause something, but they can't, like be in the analysis paralysis of, "Okay, I can't actually change it because what's gonna happen?"

And I think that communication is really necessary, but I like what you're talking about as well, of the stacks are kind of evolving. I think one thing with Data Mesh is... And you talked much earlier about purpose built. ML models are purpose built, and I think depending on what the needs are, if you really are optimizing for low latency, of actually not the model information that it's trained on, but the model that it's reacting to, most ML models are in real time of the interactions. And so a lot of people are trying to say everything should be general use, but what have you seen around people trying to, I guess almost get too cute with ML models, of trying to serve too many purposes and that they're trying to serve the same analytical things from the ML model? Or what have you seen around that, that's kind of caused issues in between the two different spaces?

0:46:11 Elena Samuylova
I think that it is the use of the model output that sometimes causes issues, because you might be using the output of a machine learning model in some of your analytical work without understanding what exactly this model was created for. This is sometimes happening because you didn't define from the very beginning what is the problem that you're trying to solve. So maybe you have a machine learning model that predicts the probability of conversions of some particular products. And then you take this model and you start using it for some completely different purpose, like putting some random inputs and trying to get some knowledge from the inner workings of the model to inform your marketing campaign. Which is a completely different purpose that you should not use the machine learning model that is precisely built to predict conversions in the most accurate way for some other purpose.

I think this misuse and maybe just misunderstanding of how the systems work, and this clash between analytical approach when you try to come up with first principles, make some assumptions about causality, like how correlations, what influences what, a machine learning model that just takes the data and tries to predict this thing in the best possible way, it might use the wrong features and still work. I think this understanding of the differences of the domain, that is very important, and both in the side of educating the business users and business analysts that might collaborate to avoid this kind of like unplanned use of the model and potential issues that stem from it.

But I think the purpose built point that you mentioned is something that comes very naturally with machine learning, and this is great potential to use within the data domain. Because with machine learning, it forces you to think what is the KPI that you want to optimize. If you start with this, this is a great start for pretty much every problem that you want to solve, what exactly are you doing and how you're gonna know that you succeeded.

0:48:11 Scott Hirleman
Well, and I think it's in my question as well would be, are you seeing that there is pressure to not do purpose built? That it can do X, but it also has the ability to do Y and Z, because they're kinda all interconnected. And so I can pass through, instead of passing through these 15 columns into A, and these 15 columns into B, but there's 12 columns overlap and 15 columns into Model C, but again, there's 10 overlap with A and nine overlap with Model B. And so why don't I just put them all together in there, whatever that ends up being like 20, 22 columns or whatever. But it's much more efficient to do it that way, but what we're trying to move to with Data Mesh is, well, in some organizations, is still initially purpose built, but that that purpose expands much more as you have additional users for this. Versus ML, the purpose is to stay essentially, at least from what you're telling me and what I'm interpreting, is to not start to have that scope creep of, "Well, it could also solve, and it could also solve, and it could also... " Instead of like, "No, we need it to focus on the specific thing."

So there is that scope creep within Data Mesh, that's actually kind of a good thing on the Mesh data products, because then you're serving additional use cases. But with ML, it's not. So I'm not even sure exactly the question I'm trying to ask, but I'm just trying to get my arms around, like is that something that you would really strongly push back on the ML side, and so that it is an antipattern that we want to not try and do the same thing with ML because it's gonna get yourself into trouble?

0:50:14 Elena Samuylova
The reality is just simply would not work. If you will try to build different models, like although you try to use the same model for a different purpose, it just will not work. What you can reuse though is features, and there are actually some developments around feature stores which help you do that, so you might have different models today using the same features and they're basically served as a data product for different machine learning models. You'll still create different machine learning models, because this is how the technology works, right? You need to train the model for a specific purpose, to optimize a specific goal, to predict the specific thing, so it kind of forces you to operate this way.

But in terms of scaling the impact of machine learning in general, absolutely right. So if you can reuse features across different models, companies are doing this, this is great. This also basically helps reuse the effort that you put into creating and maintaining the quality of these features. Just in some cases you want to be solving very different programs so you can inevitably end up with different data sources. You might be using some external data source here, a different table there, to create, to solve different problems. So it can naturally come this way just because that's how technology works, but there is some, let's call it economies of scale, in future access, in reusing the architecture. And in some cases, you might be building very, very similar models just like each of them still exist as a Model X, Model Y.

0:51:41 Scott Hirleman
It's interesting 'cause I hadn't thought about this and 'cause I really haven't dug in very deeply into ML, but I guess I never really thought of what features actually meant. But on the data side, you could think of it as almost a proto data product, right? Like where you would think about, "Hey, this same information is gonna flow into 10 different data products, and it's going to flow in this way." And so, at least we're gonna do the initial analysis here. And it has one source where it is reused and that's kind of a proto thing that isn't designed necessarily for data consumers to directly use, but it's almost a proto data product, Mesh data product for other data products only to consume from. Because it's, instead of redoing the same... Tony Baer wrote an article about how if you're redoing the same analysis across 15 different data products, that's very expensive and do you really wanna do that? And do you really wanna have them change where, okay, well, one of these needs it on an hourly update basis and the rest don't? So then do you have to necessarily do every single one if it's downstream of two or three different data products, that every single one above it has to be refreshed on an hourly basis or not?

And so, I think this is really interesting because I'm starting to rethink how we do the production cycle around data, because I think it's interesting and important. But is there anything that you have been kind of hoping to learn from the data side? 'Cause again, I'm learning so much from the ML side, but is there anything where you think ML could learn to take some of the practices from the data side that might be helpful?

0:53:38 Elena Samuylova
Yeah. Personally, I'm super interested in the data quality side and how these two kinds of pieces interface with each other. Because I still believe you're gonna have both. You have to kinda control the quality of the overall data assets, right? Detect the tables that are not refreshed, detect some missing data and so on, and you need to control the data input to a particular data product, which can be a machine

learning model, at the moment of ingestion because still you need to catch these bugs when these happen, right? And this is kind of solving the same problem from two different sides, from two different angles. Probably, we're gonna do both and most of the companies are gonna do both eventually. I'm just curious how we integrate it and we use it and kinda learn from each other.

I'm personally actually in touch and good friends with a few other founders that are solving data quality monitoring and whereas we are focused on machine learning model quality monitoring. And still there are like... These are different problems, but there are a lot of things that we can share and learn from each other, and the same applies to the machine learning field and data analytics fields in general. I'm super excited about it. I don't have the answer yet.

0:54:43 Scott Hirleman
Yeah, and I kind of have been wondering about this. You talked a little bit about monitoring the contracts and you talked about kind of ML is data in motion, right? It's not that you're... Sometimes you can land it and then react to it, but in a lot of cases, like latency is in the milliseconds. It matters. It's not that you can be like, "Okay, I'm going to be reacting to this thing." I mean, some cases it's not necessarily that, but in a lot of cases, it is, recommendation engines and things like that. "Okay. I'm clicking through to this page on the ecommerce site." I've gotta show the recommendation by the time the person's scrolling down. I can't show it 10 seconds later.

But when you think about the data pipe... And I kinda hesitate to use the word "pipeline" because that has all sorts of other connotations, but you have data that is pulling from source or it's getting pushed into your source, and do you actually want data filters on both sides? Where that producer has to be doing, saying, "If this tries to malform, if this isn't complete or if it doesn't fit these parameters, I'm not gonna put it on the pipeline." And then on the other side, the consumer says, "If it has these malformed or whatever, I'm not gonna consume it from the pipeline and here's how I'm gonna react to it." And that it feels like it's double the effort, which it kinda is, but it's also like the pipeline itself can malform things. And if you just try and make it all... There is this investigation I started to do into consumer driven testing, and it's useful, but it's also kind of an A-hole move. Because you're putting all the onus on the consumer, instead of the producer saying, "I am going to contractually agree that I'm not going to produce things in this bad way. You still need to make sure that this stuff is working well, but I'm going to... " Because stuff can, in the midst of the, while it's in motion, can still malform and get wrong and all sorts of stuff. But that, "I'm going to commit to my best effort to not put bad data onto here." Instead of just saying, "It's your problem if it's bad data that's coming down your path."

Is there anything that you've seen on that side that's useful, especially that data in motion? There's observability of this thing... "Oh, this thing was wrong" versus "Don't react to this." There's observing and there's action. So what's the action on bad data in motion instead of just observing it?

0:57:33 Elena Samuylova
I think it's a bit of a difference between monitoring and testing it, so you might be like tracking some metrics and just throwing out some dashboard and then if they go off, you go and investigate. And in other cases, you actually do real testing, and this is actually one of the things that we recently added to our own tool.

Because still a lot of machine learning models, I agree with you that a lot of them are real time, but a lot of them are batch. And for batch models, when you're just like generating predictions every hour or maybe every day, and maybe even every week, it makes sense to kinda have these checks as tests. You just received a new bitch of data. Is it good enough? I train that model. Is it not good enough? I'll go and figure out what's wrong with it. Your generated predictions, are they good enough to act on them? Okay, I'll send the newsletter that I incorporate the same way. They're not good enough? Okay, I'm gonna re-check my data. So these are two different ways, and I'm afraid we have to do both. Because it's great if the table was not updated and maybe the data producer informed the machine learning team in advance, "Hey, we will not be able to send out the newsletter tomorrow because we just didn't get the data." But maybe they forgot to do this, and you still need to be able to catch it. Or like you said, the transformation was wrong, it was the bug in the feature transformation code, so the data producer was right, it was the data engineering side, the machine learning model that was wrong.

So unfortunately, there are so many places where it can break, so if your model is important, and the final consumer, who is the business user or your clients or your customer, these are the ones who you should care most, and your internal supply chain might have multiple checkpoints.

0:59:09 Scott Hirleman
Again, I'm asking for you just... I don't think I'm asking too much by just saying, "Can you just magically solve this for us? Can you just tell us that ML has solved this and that we can copy paste that to data?" It's kind of refreshing, I think, for a lot of people to hear that the people where this is even more necessary and more impactful, that there isn't a solution yet. Because it doesn't mean the data folks are that far behind. It would be lovely if there were, but it's also like, okay, we're not crazy, we're not the only ones that are seeing the same thing.

0:59:43 Elena Samuylova

Because I deal with real production models, not like some research. If you look at research, there are a lot of fancy things, there are active learning algorithms like sales learning models and what not. But in practice, what you see is usually the most true and tested, the more boring implementations of these technologies, and sadly, everything in technology is still somehow manual, right? You might be training your models automatically, but there is still a data scientist who owns this model, who understands the features and the business use case, that has to intervene if something goes wrong. I don't have a silver bullet. I know that when you say "machine learning" or "AI" some people would expect that. We're not there yet, for better or for worse.

1:00:25 Scott Hirleman
Well, it's kind of also refreshing though, that we don't just turn everything over to the machines and just go, "Go do it," and we don't understand what's going on, those kinds of black box solutions end up being...

1:00:36 Elena Samuylova
We're gonna still keep our jobs for a while.

1:00:41 Scott Hirleman
So I have one last question that I think is kinda helpful for people to think about when they're in this space, which is, how do you think about really measuring... You talked about the KPIs, but how do you think about measuring if an ML model is really worth building or putting into production or things like that? Because when we're trying to think about the Data Mesh side of, is this data product worth building? And is it worth building, and then we find out that maybe it's not as useful, and how do we think about that kind of shutting down? And you did talk about the KPIs but is there anything that you would advise someone on specifically digging into this? Because people, again, want the, "I want to be alerted when it's no longer the good thing," instead of really the high communication level and things like that.

1:01:41 Elena Samuylova
Yeah, I think quick iterations are the best solution to this. I've spent quite a lot of time, even doing those sort of workshops, you know when you try to do it on different machine learning applications you can create. I was also engaged in a project which took months until you get the first model that you can even look at and evaluate. And this is usually the most wasteful part, because those machine learning, some ideas are good, some ideas are bad. Some ideas are great, but the data is not there. So the best thing that you can do is actually have some sort of like MVP very fast to be able to like kinda back of the napkin evaluate the quality and potential business impact of the model. Because most machine learning models are kind of expensive, you need to make sure that there is a potential for significant savings, significant in

the scale of the company that you're dealing with, to actually create it.

So again, no silver bullet. Experimentation, there are different ways you can do offline testing even without putting it in production, if you have some historical data, sometimes you can run the experiment fairly fast. So learning to do this quickly to rating and allowing yourself to make mistakes and pick wrong ideas, I think is the best approach here. Because again, yeah, so sometimes you have great ideas, but the model, you just cannot build it.

1:02:54 Scott Hirleman
Yeah, and I think this is something that keeps coming up, if you have proper communication with consumers from it, it's possible. If you don't, you're kind of screwed. Because if you're trying to ideate and you're trying to really go through and say, "Okay, let's put together something and figure out is this useful, is this worth it?" And people start to go, "Oh, you put this thing onto the Data Mesh, therefore it is, I can consume from it, trust it, and do all of that." That's such a slippery slope if you don't have that trust that people are going to RTFM, read the freaking manual, usually a different word in there. That trust and that communication is so crucial.

So I've learned a lot from this, I've learned a lot from the ML space, but it's helped me rethink a lot of things in the data and analytics space. This has been really great for me. Is there anything we didn't cover that you think is important to, or any way you kinda wanna wrap up the episode, any one point you wanna kinda hammer on or anything like that?

1:04:10 Elena Samuylova
Probably one point I want to repeat, reiterate a bit, is that this maintenance part is, usually it doesn't sound very exciting, especially if you're coming from this more researchy domain, you want to look at core models, you want to have deep learning, like the biggest, the fanciest model and so on. But this is literally the most important thing. I just want to kinda highlight it, especially if you are maybe a business user thinking about creating this model. You have to factor in the maintenance, the cost of maintenance and the processes around it. It is not a one and done thing, and it's not more or less you create and it lasts forever. And because you want it to continue delivering business value, you have to plan for it.

1:04:48 Scott Hirleman
Yeah. Well, I think the degradation of the model is a good thing to think about. If you just treat it as a data asset, it's something that you have produced. Versus if it's data as a product, it's something that is ongoing with maintenance. I think that, especially when you talked about the ongoing cost, a lot of people think of Data Mesh as an initial implementation. And it's like, no, you've got ongoing...There's the stuff that

you've already built, it doesn't just run itself at no cost, you have to think about that. And measuring does it still have value. It might have had value in the time and it didn't, or it might have been a bet and the bet didn't pay off. That's okay. You made a bet. That's what a bet is.

1:05:40 Elena Samuylova
And if you build it, it will break, I promise you. So think about it from the very beginning. It will break. What are you gonna do and how are you gonna know about it?

1:05:47 Scott Hirleman
Yeah. A lot of people think about it, "If you build it, value will come," but I like, "If you build it, it will break." I like that a little bit more. That's a little bit more of my style. So Elena, this has been super phenomenal. One, where would you like... I'm sure there's gonna be people who wanna follow up, where would you like that to happen? And what would you want them following up about?

As well, if you could work a little bit more about what Evidently.AI does, 'cause again, I think it's a very interesting company, and it's very useful in the ML space, plus its MLS so that's... Sorry. It's open source, so that is near and dear to my heart. Because I think way too many people are trying to jump to just trying to extract the money instead of adding the value. And I think your content and stuff as well has been... I keep seeing it pop up into my space, even though I'm not following that many ML people, people keep just saying, "This is the good content in ML."

1:06:50 Elena Samuylova
We're truly trying to solve this problem, the model monitoring, in a general way. So if any of the listeners have models in production that they want to keep an eye on and they're trying to understand how to do this, reach out to me on LinkedIn, on GitHub, that we have a discourse community, whichever channel you prefer, and let's have a chat.

Because we are truly interested in figuring out the best practices for the industry as a whole, different architectures, how you integrate it with other tools. So if you're right now building a machine learning platform, or you were like, "We're thinking, we're just starting to think about this," let's chat. And yes, LinkedIn is the best for me, but you can find us everywhere, it's Evidently.AI.

1:07:31 Scott Hirleman
Yeah. I've seen interactions and stuff, and I do really recommend anybody out there as well on the data side to kind of go into this conversation and reach out to Elena with your data science leads or ML leads and be like, "Okay, let's figure out how we

are actually thinking about the combination between the data and the ML side." Because ML is that purpose built. Data Mesh, we're trying to not be as purpose built. Or maybe start purpose built to a use case, but also reusable, and how do we differentiate between those and still drive the most value from both. So I think I always learn a lot when I have conversations with you, so I recommend a lot of people reach out. But again, thank you so much for your time today, Elena. And as well, thank you everyone out there for listening.

1:08:24 Elena Samuylova
Thank you for having me.

1:08:26 SH:
I'd again like to thank my guest today, Elena Samuylova is the co-founder and CEO at the ML model monitoring company and open source project, Evidently AI. You can find a link to her LinkedIn as well as the company GitHub and blog in the show notes. If you are interested in ML, they put out a ton of really excellent content, so check it out if you're interested in that. Thank you.

Thanks everyone for listening to another great guest on the Data Mesh Learning Podcast. Thanks again to our sponsors, especially DataStax who actually pays for me fulltime to help out the Data Mesh community. If you're looking for a scalable, extremely cost efficient, multi data center, multi cloud database offering and or an easy to scale data streaming offering, check DataStax out, there's a link in the show notes. If you wanna get in touch with me, there's links in the show notes to go ahead and reach out. I would love to hear more about what you're doing with Data Mesh and how I can be helpful. So please do reach out and let me know as well as if you'd like to be a guest. Check out the show notes for more information. Thanks so much.