*Please note the following timestamps are approximate.*

Jeff Reback (Guest) (00:00):

So I think pandas can definitely serve a really compelling use case here for, “Just use it.” If it doesn't work, get a bigger machine. If it still doesn't work, go distributed.

Peter Wang (Host) (00:12):

You're listening to *Numerically Speaking: The Anaconda Podcast*. On this podcast, we'll dive into a variety of topics around data, quantitative computing, and business and entrepreneurship. We'll speak to creators of cutting-edge open-source tools, and look at their impact on research in every domain. We're excited to bring you insights about data, science, and the people that make it all happen. Whether you want to learn about AI, or grow your data science career, or just better understand the numbers and the computers that shape our world, *Numerically Speaking* is the podcast for you. Make sure to subscribe; for more resources, please visit anaconda.com. I'm your host, Peter Wang.

Peter Wang (Host) (00:48):

This episode is brought to you by Anaconda, the world's most popular data science platform. We are committed to increasing data literacy, and to providing data science technology for a better world. Anaconda is the best way to get started with, deploy, and secure Python and data science software, on-prem or in the cloud. Visit anaconda.com for more information. All right, so welcome, and I'm very, very excited today to have a fun and lively, I'm sure, conversation with Jeff Reback, who is a longtime pandas maintainer, and he is a managing director at Two Sigma. So if any of the listeners on this call, if you use pandas, they are benefiting from many, many hard years, many years of hard work, by Jeff on the project. So thank you Jeff, and welcome to the Anaconda podcast.

Jeff Reback (Guest) (01:36):

No, thank you so much, Peter. Very, very happy to be here talking to you about pandas, and work, and all these other nice things.

Peter Wang (Host) (01:43):

Looking forward to getting your perspectives on all sorts of interesting and fun things. But let's, to get started, for the listeners who are not as familiar, can you tell us a little bit about how you got into programming, what your background is, and then also how you got involved in the pandas project, and your role as it's evolved over the years?

Jeff Reback (Guest) (02:00):

Sure. So I guess a lot of other young folks, I started on Wall Street in the '90s. That speaks to my age of course. And for years, ironically, we actually...I chose Perl for a long time, we would do really everything, and this was maybe went on for about 10 or 15 years. I literally wrote everything—back ends, front ends, everything. It was just very convenient to do. Perl was the thing, and then around, I would say around 2008/2009, I switched the role I was doing and I said, "Oh, hey, let's take a look at this Python thing." I started looking at Python and at the time I was working at this new fund and I said, "How do I store data and store everyday's data, and then I want to retrieve this," okay? Do queries on it. I stumbled upon PyTables?

Peter Wang (Host) (02:44):

Yes.

*Please note the following timestamps are approximate.*

Jeff Reback (Guest) (02:44):

And it actually introduced me to the HDF5 format and I was like, this sounds awesome. I stumbled upon pandas too, and I think at the time it was maybe...I don't know if it was .1, or it was like .2 or something. This was back in 2000…maybe '10 or '11 or something. So it was way before Python was popular. Actually, ironically, when I first picked up Python, I started with the "let's do Python 3000" book or something. And of course that was...in retrospect, we had to go back to Python 2, but that was totally fine. And anyways, so I started using pandas, this was way back then, and I wrote locally an extension to write my data to HDF5 and retrieve it, and it worked great. I was all excited, got my work done, and I think about six months later, what I was doing, I changed roles again, started working at a different company.

Jeff Reback (Guest) (03:29):

And I said, "Okay, let me take a little break. And hey, maybe they want this in open source." [At the] time, it was just email a guy and that guy happened to be Wes McKinney, the original founder of pandas, and I don't think I even did a pull request. I think I may have sent him the code at that point. Maybe GitHub was around, I don't even super remember, but eventually, with very limited back and forth, I mean, I had tests and everything, but I had not done any open-source programming at this point. And so I just sent it to him. He accepted it, it was like, "Oh, that's pretty cool." I think then at this next role I was doing, I worked with some former colleagues. I ended up, I was building trading systems, and so I take data, and do transforms on it, do machine learning on it, and then actually do trading on it. And so it turns out that I was doing a lot of simulation. So I would sit there and wait, I would submit some jobs and then wait.

Jeff Reback (Guest) (04:19):

And so I had some extra time. I was like, "Oh, let me go look at this pandas thing." And I did, and I found some bugs, and I'm like, let me go see if I can fix it. It was a challenge to me. I eventually started fixing a few bugs here and there, and I think this was still maybe 2011 or 2012, and I did more and more. And then I started following the issue tracker, and I was like, "I am good at this," and I actually liked it, too. So I did this for maybe a few more months, and pretty soon I was just responding to questions, doing PRs, really quickly I became known very shortly as, a bug would pop up and it would be fixed five minutes later I was like, "Oh, I just like doing that. I like being responsive." And so that's how I got started in pandas, really. I was an enthusiastic user in some sense. And then I actually liked the technology. I liked how it was being used and liked to contribute to that. I had time, it was—

Peter Wang (Host) (05:10):

Well, you made time, it sounds like. I mean, you had a day job. Most people on Wall Street, they're quite busy.

Jeff Reback (Guest) (05:15):

Exactly. Well, this was very much a research-y time when, as they said, I was just in a room with a few guys, and I had a bunch of boxes and we were able to do some simulations. So I continued contributing to pandas. This was, I think, in 2012, something like that, and I found that I was doing it more and more and more. And I eventually got involved in some of the early release cycles. I think pandas may have been, I don't know, .8 at this time or something. I think maybe the next year is when I actually started doing some releases myself—

Peter Wang (Host) (05:46):

Nice.

Jeff Reback (Guest) (05:47):

And really taking, I'm not going to say over the project, but more really reviewing other people's pull requests, and so on and so forth. And coincidentally at the time, Wes started to step back from the project, maybe because I was contributing. And so around that time I ended up starting really doing my own code, or doing lots of pull requests myself, reviewing other people's code and doing the releases. And I did that for, I would say several years actually. Really full time, mostly contributing code.

Peter Wang (Host) (06:15):

When you did the first pandas release, the first release to pandas that you did, how many other people at the time were authorized to do that?

Jeff Reback (Guest) (06:23):

Right. So you're basically asking who were other committers?

Peter Wang (Host) (06:27):

Well not just committers, but people who actually were authorized socially, had the social capital or the whatever it is….Commitment is one thing, but to actually say, "Yeah, I'm going to go and do a release." At that time, how many other people could actually do a release, and were authorized to do so by Wes?

Jeff Reback (Guest) (06:43):

Right. I think it was Wes, and maybe one or two other people. And so it was a very small group at that point. As you know, in open source, people come and go a little bit. But I happened to stick by pandas, and I was encouraged because we had lots of issues, and this is the hallmark of a successful open-source project, is that actually people are using it, and therefore when they use it, they find lots of bugs. They find issues and make feature requests. I like doing it, actually. That's why I was drawn to this.

Peter Wang (Host) (07:12):

And I think there's something around this, which is you're obviously through your career on Wall Street….Well, so let me back up one step. A lot of people who end up in open source, especially in, let's say, the Python scientific numerical ecosystem, they come from science, they come from math-y things, engineering, or a lot of astronomers that end up in there. Not a lot of people end up coming into it from Wall Street, or from the finance sector, and I've got my own theories as to perhaps why that is. Certainly Python is used a lot in that area, and many people there have the skills to contribute, but we don't end up with a lot of long-term contributors or certainly maintainers in the ecosystem who come from FinTech.

Peter Wang (Host) (07:52):

Of all of the people you've worked with through your career, and you started on Wall Street, obviously as a research quant and programmer and all these things, you have met many people who I would assume you would feel are qualified to do this kind of contribution. But you are the one who stuck by it, who was answering bug reports and questions late in the night. Is there something particular about you? How many of your colleagues do you feel could have also done the same, or might have had a susceptibility to doing the same, but just didn't for whatever reason?

Jeff Reback (Guest) (08:20):

You raise a very interesting point here. I think that there's definitely the skillset on people who work on Wall Street or in FinTech. I think there's a couple of factors, though, that generally play against this. One is people obviously are super, super busy. You just simply don't have the time. It's ironic because I had a little built-in window. This is a funny story, where I would….So I lived in New Jersey at the time, and I was commuting actually into Manhattan, and I would do it by ferry. So I would leave at six o'clock in the morning. I had this hour ferry ride, so it's sort of a built-in time where you're awake or you're…well, I don't know if you're wide awake, but you're awake, and it's like, I have an hour to do stuff, so let me do it. And so I did that actually for several years. I think it was four or five years, this was when pandas got done, to be honest, [to] a large extent. You could do a lot in an hour of uninterrupted time.

Peter Wang (Host) (09:06):

Especially in the morning when you're fresh—

Jeff Reback (Guest) (09:08):

Exactly, exactly.

Peter Wang (Host) (09:08):

When you haven't had the emotional drain of the day, like a vampire sucking away your creative juices, right? Yeah.

Jeff Reback (Guest) (09:14):

Absolutely. So this, I think, is one factor. The other factor is I happened to….So I started at an interim where I was working at a smaller firm, where it was kind of my own firm. So actually we didn't have any restrictions on contributing to open source at all. And then when I went to subsequent firms, actually I had the ability to contribute to open source. I did work at Continuum, which was the precursor to Anaconda at the time. And then I worked and I'm working with Two Sigma, and we actually have a very open policy of being able to contribute to open source. So I think that is one of the big factors here. Most, well, I shouldn't say most, but to my experience, a lot of folks who are very, very qualified people, and they love open source, and they use it quite a bit, but they may not necessarily have as an open environment. So I think that's one big restriction that a lot of folks have. I think that's changing nowadays, but for many, many years this has existed this way.

Peter Wang (Host) (10:03):

Yeah, yeah, it's really good. Because 10 years ago-ish, nine years ago, something like that, I remember having a pretty intense conversation with a CTO of a hedge fund, trying to argue for the use of open source. And not only that, but allowing some of his employees to contribute back, as opposed to the mindset on Wall Street is very zero sum. Everything's proprietary. There's no benefit to releasing this stuff. Why would we give our competitors a leg up? Make them pay for it, make them pay for devs to fix their own stuff. And so there was this kind of conversation that was happening over a very fancy steak dinner. I was trying to make the point to him…

Peter Wang (Host) (10:38):

I said, "Look, at this point in time, everyone we talk to..." And again this was 2013 timeframe, pandas was clearly ascendant but still ascendant and not dominant. And at the time I was saying, "Look, every

single firm that we get called in to talk to, they've created their own crappy version of a pandas-like thing around NumPy, around probably HDF5, a PyTables for some persistent stuff. They've all created their own crappy version. They've got their own little bespoke datetime libraries that they've wrapped, and some other calendaring stuff. Every new person you hire coming in the door is going to know pandas and not your own internal crap. And maybe pandas has some bugs, but your internal stuff has bugs too. But there's hundreds of thousands of people using pandas, looking at the bugs. Yours has just, whatever, the few dozen quants in your firm."

Peter Wang (Host) (11:25):

And so some of these arguments, it was so hard to make those arguments, and I don't think I particularly won that particular discussion. But the point is, at the time, just to reiterate, I think, your point about how much the temperature has changed in that industry, at that point I was having to argue just to use the fricking open source. And now you've got people like Two Sigma, and of course Two Sigma is a very forward-thinking, long-term-looking firm. And so they are investing in the open source and they have been a long time contributor and supporter of NumFOCUS, and so we love that. So it's really kudos to Two Sigma for doing all the right things there for the ecosystem. But I think your point about doing the work on the ferry is very interesting.

Peter Wang (Host) (12:03):

And so I have a question a little bit around this, which there's this theory that people need at least three spaces, the home space, the workspace, and a play space. And so when you're on the ferry, you're between spaces, you're not home and you're not at work. And so you have utter liberty to play. You could be on your phone playing a game, or you can read a book, or you could hack on some code that isn't part of your day job. When we shift the working on open source under the realm of the work, even though it's sanctioned, it's allowed at work, but it still falls into the work umbrella, is there a different flavor to that somehow? Does it change at all? Maybe for you, because you got into it starting with the play space, it's the same. But do you see a difference in how people approach it among maybe younger colleagues who are doing open source for the first time on the clock, so to speak, at Two Sigma?

Jeff Reback (Guest) (12:50):

I mean you absolutely raise a good point. I think that in my personal experience, this was definitely…it was almost my downtime to be very honest. I read a lot, for example, but I do that of course when you're not supposed to, at night when you're trying to fall asleep. And so my downtime, I would read on the ferry, true. To me, sitting in a chair for an hour on the ferry, it actually was, to me, I can get stuff done and I'm like, I feel productive. And it feels good to contribute to open source right there, and it feels good to do it right then. And you're right, I'm not distracted by other things. So I think that is definitely a big contributing factor, how I viewed this so-called play space. Yes, I play video games, but look, I didn't want to do it on the ferry. To me this is something that's in between work, it's kind of related to work.

Jeff Reback (Guest) (13:37):

Now, going back to a point you made before about, before you mentioned why certain companies may have had a, we'll call it a negative attitude toward open source, and now have moved full spectrum toward really not just consuming open source, which I think pretty much everybody does nowadays, but actually contributing back. And so my personal experience of this is, I think, one of the reasons people have embraced this. I think when they first start using open source, people tend to take the packages that are out there, like pandas, and they fork them because they're like, "Oh, I have this bug that I fixed

internally and I can just take my own fork of this and put it on top." And that works for about five minutes, until you realize I actually would now want to rebase to upstream and I want to use the upstream and then I have to put my patch on top.

Jeff Reback (Guest) (14:23):

It became a tremendous amount of work to actually do that. And I know many people who have tried to do that, and it's always a failure. And so I think a lot of firms realized that it's so much easier to try to just simply fix upstream, get it in there, this is the critical fix, this is the thing you need. And at the same time you do get obviously the benefit of all the other worldwide contributors to pandas, I mean, there's thousands of people who have contributed and fixed bugs, which you didn't even know that you're picking up inside. And so, I think the attitude has shifted because you basically can get the almost free labor of open source, and get your fixes too, without having to contribute an enormous amount on your side from a technical point of view. So I think that attitude has really shifted over time.

Peter Wang (Host) (15:08):

Right, and that's something that you bring up a very good point—something that a lot of people who are in the open-source community, but don't ever enter the tall skyscrapers of big enterprises…they may not realize the degree to which this is standard practice. Most businesses actually have a lot of private internal forks of open-source packages. They don't talk about it. Why would they? And the only people who know about it are people who don't sign up for a full-time gig, or contractors, and then you see, oh my god, you guys are maintaining a pile of patches and it's going to slow you down because you're the only one maintaining them. There's something, and let me try to figure out the right way to frame this philosophically.

Peter Wang (Host) (15:51):

In the open-source ecosystem a term I've used a lot around this is the word generativity. It's one of the few things…because it's information artifacts. If you go back to is software IP, is software…you go to the Jeffersonian, "My candle does not diminish when I light the flame of yours," or whatever, my flame isn't diminished by lighting your candle. When someone else clones a copy of pandas, they don't make your copy of pandas any less valuable. It's one of the very, very rare things, I mean, we'll flex the metaphysics a little bit, but a pile of source code, software projects, are one of the very few things in the world that we can point to where the more you give it away, the more valuable it becomes, right?

Jeff Reback (Guest) (16:31):

Yes. This is like the old network effect, I think, is what you're describing.

Peter Wang (Host) (16:33):

It's a network. Yes. Because there is a network effect of it. There's the very basic level, which is if you use it and someone else uses it and you have a problem, now you at least have someone else to talk to say, "Hey, did you have this problem?" Or "How did you solve this problem with this set of tools?" If all of us are using wildly different incompatible tools, none of us can help each other work those tools better, even if they're great tools. And then you add to it, then there's additional levels of network effect and network benefit, when things are in coherence, or in space coherence, or in space concordance with each other.

Peter Wang (Host) (17:05):

And there's harmonization as value, versus taking, and taking away from someone else. Because taking always implies taking away in the physical world, you have a very different view of what is value, and what is damage, or what is additive versus what is subtractive. And so I think there's something really counterintuitive about this that most, certainly managers, don't have a way to account for when their people contribute to open source, so when they release software into the world. But it's also, to me, it just, again, I don't want to ding the finance industry too much, but so much of the prevailing mentality there is zero sum, is not abundance mentality, or generativity, right?

Jeff Reback (Guest) (17:40):

This is absolutely true. I was just thinking of two more reasons why this actually has been changing recently. So one is that using the standardized software, it means that when you hire new folks and we're always hiring new folks, every single bank on Wall Street hires new folks constantly. They are already pre-trained in this standard API.

Peter Wang (Host) (17:59):

They're pre-trained.

Jeff Reback (Guest) (17:59):

This is super fantastic because now you don't have to teach them. And then secondarily, whenever you have a problem, there's a global resource: Stack Overflow. Every single question you've ever asked about pandas is there, easily searchable. So again, whenever you have a problem, that is your first resource. I mean no matter how many experts people have in pandas inside, there's a hundred times of them outside. And that is just such a great benefit to society, I think. And in fact, that's actually one of the reasons I think pandas is now so-called standardized. I mean, people are using it, they're copying the API. That's a whole separate discussion we can have of why they're doing that. But I think it has become almost a utility in some sense. There's good things about that, and there's not so good things about that, but that's, I guess, just the way it is nowadays.

Peter Wang (Host) (18:44):

The utility, you're one of the linemen flying around dangling off a helicopter trying to work the high voltage lines. It's a utility, but for some people it's still a job, and it's a tough job to maintain and support all those different use cases. Let's talk a little bit about how your role has evolved over time, and how the team has evolved over time. What does it look like now when every change involves a lot of thought about the impact of the change? I'm sure the velocity is not what it was when you only had a 10th or a 100th the users. So tell me about that.

Jeff Reback (Guest) (19:13):

Absolutely. I think from a personal point of view, maybe from 2013 to maybe, I don't know, 2016 or '17, I was a very heavy contributor of actual code, like commits, and I obviously did some level of code review. Then I would say probably in 2017, I changed almost 180. Since then, my contribution has been almost 100% code review. My level of contribution is virtually the same, meaning the number of contributions, or whatever you want to call it. So I mean, that's really indicative of the fact that we've had many, many, many either drive-by or repeated contributors. Our committer code base has…the number of folks who

commit to large parts of pandas, I would say has grown from just a handful to now maybe 30 or 40 plus. We're a fairly generous package, in terms of giving out commit rights. We love for people to commit.

Jeff Reback (Guest) (20:01):

But pandas, actually, interestingly enough, has a very interesting property. It's very wide, meaning there are so many areas to it. From reading CSVs, to doing indexing, to all these different I/O connectors, that it actually allows people to specialize and that's almost de facto what you have to do. You cannot get your hands around everything. And so this allows folks from a lot of different walks of life, some are scientists. I think you made a point earlier that most are former scientists. Yes, that's probably true, but the science that they come from, neuroscience, and there's a chemist, and there's a fellow who...there's a number of software engineers as well, but they're very, very diverse, which is fantastic.

Peter Wang (Host) (20:42):

It's fantastic, yeah.

Jeff Reback (Guest) (20:42):

I mean, this really contributes a lot to the ecosystem itself, and the community around pandas. But I'll say we've had entirely volunteer contributions, I think all the way through about 2019, 2020 timeframe. And that is only the very first time when we actually had funds available through various grants to actually pay maintainers. But even that did not ramp up until very recently. So as of 2022, we actually now have three full-time funded maintainers who actually contribute, do code review and triage issues, and so on. So pandas has moved into really a different phase at this point. It's fairly mature, lots of edge cases that we're dealing with, but the model's changed. Like a little company almost, at this point.

Peter Wang (Host) (21:25):

In this talk that you gave at Two Sigma, which is I guess now publicly available on YouTube, right?

Jeff Reback (Guest) (21:30):

That's correct.

Peter Wang (Host) (21:31):

You have a slide in there where you indicate that just given, looking at the level of commits, the amount of review necessary, the amount of work necessary, pandas really needs to have 10 full-time developers working on it to support just the care and feeding of the project.

Jeff Reback (Guest) (21:45):

That's right. pandas is, as I said, I think we have over 3,000 open issues, but what that generates is between 10 and 30 or 40 PRs per week.

Peter Wang (Host) (21:55):

Wow.

Jeff Reback (Guest) (21:55):

This is an extremely active project, and these all require…some are very simple, but you have to have the expertise and the deep knowledge to really review these things. And so we've grown that over time. But it is non-trivial. This has grown up into a very large project, and I don't actually know what would happen if we didn't have funding for maintenance at this point. I mean, it's hard. As I say, I think I used to review every single PR. That is recently, I have gone down that reviews and I still do some reviews, but it's gone down a lot recently.

Peter Wang (Host) (22:26):

Well, and I think this is something that is not…a lot of users, of course, never look at GitHub contribution statistics or things like that. But I do want to just draw awareness for the listeners on the podcast of the degree…I mean, Jeff, you're a pretty modest guy. You're a very talented guy, but you're also a pretty modest guy. The degree of Jeff's contribution is significant, let's call it that. I mean you have just been nonstop pounding away on stuff for a very long time. And it is truly…I don't think you're celebrated enough for your accomplishments in this regard. The world really does owe you a tremendous, at least many, many rounds of beer, and a lot of good wine for how much, at a minimum, for how much work you've done there and growing the team is non-trivial.

Peter Wang (Host) (23:13):

A lot of open-source projects do flame out, because they can't manage to cohere a team, and then the original author or maintainers move on, do other things, and then the project withers. But the pandas project is a really nice example of what happens when you do grow that team intentionally. So tell me a little about how you think about…you did say there's a lot of places where people can specialize. That's one aspect of how the team grows. But when you're growing the team, and as you're mentoring people, you say you're liberal with the commit bit, which is great, but then of course not everyone is necessarily ready to take the next step and become an active maintainer. What do you look for and what wisdom could you give people maybe who are interested in contributing, either to pandas or to any other project? Or the things you've seen that are really good, tips that you could give, or mindset, or whatever kinds of advice that you could give to people?

Jeff Reback (Guest) (24:03):

Thank you for those kind words. Yes. I've been a long-time contributor to pandas and yes, I've done a lot over time. I mean, as I said, I think pandas right now, it's definitely moved from the, “let's build this cool project and break things fast and make it really super useful,” to now it is really a mature product. It is very…we have to view it in a very different light. Whenever I'm like, "Okay, I want to deprecate this thing," we have to think about, this is going to cause millions of people to change code. That doesn't mean we don't do things, of course, but we're just a little bit more thoughtful and retrospective about things. I definitely think that growing, or getting support, really ongoing support, obviously the monetary aspect helps. There are folks who they may have day jobs, but they are able to contribute hours to really work on pandas.

Jeff Reback (Guest) (24:48):

But I mean, I think the heart and soul of pandas is really attracting newcomers. And of course we absolutely love diversity among newcomers. It's interesting, we had one of the maintainers, I think it was two years ago, maybe three years ago, we ran this worldwide sprint, I think it was 24 hours for just documentation. And I think we had, gosh, we had 50 or 60 contributors literally worldwide from 30 countries, it was amazing.

Peter Wang (Host) (25:12):

Wow.

Jeff Reback (Guest) (25:12):

And I think things like that really help encourage folks to do this. We want more and more. Because what happens is you have 50 contributors, whatever, and then maybe one of those becomes a maintainer. I mean this is really the, I think the—

Peter Wang (Host) (25:26):

There's definitely a funnel.

Jeff Reback (Guest) (25:27):

Yeah, exactly. This is really the hard part, and it's because it's a combination of, you have to have your own internal drive, just like any other job that you do. You have to have drive in that job and it has to be interesting to you. And I think that's actually where pandas can really shine. We have so many areas that you can find some little…it could be a little niche or a big niche. We have one contributor, all he works on is the Styler. This is the way we output formatting. But he's done such an amazing job on that, and he loves doing it. I'm like, great, more power to you. It's actually, in some sense, it's almost now a mini project. He will come back and review all the PRs that are associated with that.

Jeff Reback (Guest) (26:01):

So I think if I had to say what people should do, it's find a project, it could be a big one, it could be a small one. I think big ones are actually easier to approach simply because they have lots of guidelines already of how to do things. It makes the getting started part really, really easy. I mean, this is one of the things over time we spent enormous amount of time and effort really making the documentation, how to contribute, actually very accessible. I think that's one, if you can just run your three commands and just boom you can build the project, and you have instructions on how to do it, it's a really, really great thing to contribute. And then eventually, if you have the drive, maybe your employer sponsors you, maybe you're just doing it in your part time, you can become a maintainer of a project.

Jeff Reback (Guest) (26:44):

It's not tremendously hard, but it's also…I'll divide the maintainers into two camps here. One is people who are really super awesome [at] doing code, they're awesomely detailed in code, and that we absolutely love and appreciate. We also appreciate folks who can code review other people and give nudging comments to contributors. Because not everyone is an expert in open source, not everyone knows how to build their environment. And so helping folks out, and even contributing to things like documentation, community, these are all super important things. If I have, my nugget of advice: Find a project, start contributing, find a community you like, and just help out.

Peter Wang (Host) (27:23):

That is an interesting thing about approaching the larger projects, they are more likely to have some things in place to help you get started. And they also appreciate the importance of introducing and being welcoming to new maintainers and whatnot. Yeah, it's definitely not…smaller projects, if you're passionate about it, if you have the passion, it's hit or miss. You might find a maintainer that's very

welcoming. You might also find one that, for whatever reasons, it could be something as simple as language difficulties…

Jeff Reback (Guest) (27:52):

Yes, absolutely.

Peter Wang (Host) (27:53):

Maybe you don't speak that language or something. All sorts of reasons why, then the collaboration doesn't really work. But if it does work, it can be really great, because then you're closer to the core development of the project.

Jeff Reback (Guest) (28:02):

I was going to say, I view it as the small town versus the big town. You could go to a small town and it could fit you like a glove, and that is super awesome, great. Or maybe not so great if it doesn't fit you, because you know everybody in that town. Whereas a big city, you could be somewhat anonymous, but you can find your niche. And so there's different schools of thought here, I think on what projects but I guess just the advice is just try.

Peter Wang (Host) (28:25):

Just try it, yeah.

Jeff Reback (Guest) (28:26):

Yeah.

Peter Wang (Host) (28:26):

You did say something about how, just earlier, there's a comment you made that pandas has really grown up now. It's a very stable product, it's of course got very, very wide…it's a standard, it's quite ubiquitously adopted. There are people cloning the API, right, for other kinds of things. But what pandas fundamentally still is, it hasn't strayed from its original thing, which is it's a library that developers use, or sorry, that data scientists or quants or whatever, use generally locally, single machine, to process some data, to play with their data, oftentimes exploratory scenarios. So now in recent years there's really been this emergent, or let's say resurgence of focus on data engineering.

Peter Wang (Host) (29:07):

And I mean, what used to be a lot of data-warehouse-type stuff has now shifted into, with the advent of cloud data warehouses and whatnot, there's been a shift in [the] data engineering discipline, and the emergence of the “modern data stack” is now the term of art that people use. And there's many popular tools and scripts to schedule this kind of pipeline, or do that kind of distributed compute, and all these things. I guess my question, and it's kind of open ended, but what do you see as the role of pandas, which is very, I would say, still quite focused on the single-user, single-node experience. What is the role of pandas in that kind of a world? What do you think?

Jeff Reback (Guest) (29:45):

This is a great question. I mean this is of course the endless debate of, hey, my tool doesn't fit my use case. So let's just, instead of trying to improve upon the existing tool, let's just go write a new one. And

that's what a lot of folks have done. Fundamentally, when I'm working with data or really looking at something, I want to do the simplest thing possible. And so the simplest thing of course is to just do single-node, just immediate eager execution. Just show me the data, let me look at it, let me transform it, let me play with it. And if your data size is up to…so maybe five years ago maybe having a five- or 10-gigabyte data frame was actually quite exceptional. I think nowadays maybe that is much more common, but even still, everything that size and less is super common.

Jeff Reback (Guest) (30:29):

Having data bigger than that surely happens, so maybe instead of happening in 5% of cases five years ago, now it's maybe 15% of the cases. But still we have that 85% of cases where pandas just works. It works well, it's actually quite performant, and it has a lot of bells and whistles. Everybody, you can write your syntax and really get out what you need. It's funny, everybody's like, "Oh, my script is too slow." I'm like, "Okay." Or maybe it doesn't fit in memory. I'm like, "Okay, buy a faster computer or get more memory." And actually that is a really, really good case today. I mean, I can spin up a node, my single machine for two terabytes for a dollar an hour or something. Why do I need to even do distributed compute in large cases? So I think pandas really, it's not just hobbyists, it's basically replacing Excel.

Jeff Reback (Guest) (31:16):

That's the competitor today for pandas, to be honest, to a large extent. Yes, obviously we are seeing convergence among the so-called big data and the data science stack. It's been happening for years. We're getting closer and closer. I mean what's been happening is people like the Spark folks and the BigQuery folks now they all, and even the database folks, they all allow you to run Python code as user-defined functions. And you can use Python or pandas inside using a chunk of data. And of course the Python ecosystem grew Dask to do exactly this type of thing. It uses pandas internally. And so there's absolutely a role for both here. I think when you truly need to process a massive amount of data, and that could be…maybe massive means, it's 100 megabytes of data, but maybe it takes an hour to chew on.

Jeff Reback (Guest) (32:07):

I do want to do things in a parallel way, and I do think pandas can become even more performant, dispatching some computation in a more distributed way, maybe just on its local machine. I think that's going to happen. I think that's the direction we're moving, using some various engines, like we use Numba, we're going to be using [inaudible] a little bit to do some of this, we'll call it local distributed compute. I mean one of my favorite things to do is spin up a Dask cluster, but on a single machine. It feels like I have amazing superpower. I can do anything I need to do and I don't have to worry about networks, and spinning up extra machines. So I think that's a really big use case. That is almost my ideal. Have a single big machine, use pandas. It just works. Maybe with Dask or something. But so I think pandas can definitely serve a really compelling use case here, for just use it. If it doesn't work, get a bigger machine.

Jeff Reback (Guest) (33:02):

If it still doesn't work, go distributed. Now of course, the problem here, and I do touch on this a little bit in this YouTube video, is that you effectively have to switch your API at that point. And that is a shame. I mean it would be great if we could all use one single API, and just call it a day. pandas has become that API, but it is eager execution. So whenever you go distributed, you really have to…it leaks, basically, it leaks the distributed nature, and so you really have to switch. It's unfortunate. And so actually inside Two

Sigma, what we do is we use pandas in a capacity at the single-node level. And if we have to go distributed, we do. And we can still use the power of pandas to really do the hands-on manipulation. So I think the world is converging. I think the world can still use pandas in a lot of scenarios. So I think it'll continue to exist for a while.

Peter Wang (Host) (33:56):

That's a great answer. And I think from my perspective, there is the raw capability aspect of, yeah, data sets are bigger now, but machines are also bigger and also people are more familiar. I would say, data scientists, not just software developers, but people who are less sophisticated from a software development perspective, they're more familiar with cloud, and so they're going to spin up some cloud resources to do things. But then the other thing I think about is that data science is crossing into mainstream production use cases inside a lot of businesses. And so when that happens, the management of your compute infrastructure falls, then, under a different team than the data science team, than the exploration research teams. And that other team is maybe not so sanguine on this idea of dynamically spinning up a bunch of random stuff that they don't really know how good is your failover, versus we want to watch all the nodes.

Peter Wang (Host) (34:46):

How are you doing some Kubernetes provisioning on your thing, versus how we want to do it? And all this kind of stuff. Or we have standard machine sizes that you're supposed to get. I know in your research cluster you have all these things you could do, but not in the production cluster. And so, one dynamic I'm seeing is that a lot of data scientists, I feel like they realize, look, if this stuff going into production is going to have to end up in this kind of infrastructure mode anyway, then I'll just start using those tools and do my exploration in those tools. And I will use, even though my data set may not start requiring a Spark, Databricks, whatever kind of environment, I'll just start doing exploration there because I'm going to have to end up there anyway. Likewise with Snowflake and Snowpark and whatever it might be. So there is that school of thought, I think, or that kind of pragmatism, that is changing people's approaches to how they even do the exploratory mode.

Peter Wang (Host) (35:33):

But the other thing I would say, almost as a counter to that, there is fundamentally this idea, however, that whether the production/deployment environment is different or not, empowering, and obviously Python has always been about this, the Python data ecosystem at least, empowering the domain expert, the subject matter expert, to go as far as they possibly can, to reduce the friction as much as possible, to reduce their cognitive burden as much as possible, so that they can self-service their computational needs to answer their problem, and to basically cycle as fast as their brain can run. That's always what I think, for me, been the dream of this stuff. And that for me is the reason why tools like pandas, tools like Numba, or even a single-node Dask cluster, are so important. Because it lets that single person, that single brain with all the insight and all the questions, maximally take advantage of a low-friction compute environment.

Jeff Reback (Guest) (36:23):

Absolutely. You hit on really two key points here. One is…the second point, I don't think I have anything to add. Less friction is extremely important in development, but I think your first point of making a frictionless environment from development all the way through production, that is the hardest thing, not just the average corporation, but I think most places this is hard. And you hit upon the answer really is to

use the same tooling in both places. And if that tooling is using pandas and analyzing this data set, well, let's put that exact environment in production and just make it work.

Jeff Reback (Guest) (36:57):

So then I don't have to change anything. I don't have to worry about, there's differences between here or here. And that tooling, by the way, is actually really easy to use and I already know how to use it? More power to it. And so I think these are some of the convergences that we see, and this is why, I think, some of the vendors, Databricks, or you mentioned Snowpark, they actually embed the ability to run these processes inside their systems. And I think it's super powerful. Combining these—

Peter Wang (Host) (37:24):

Snowpark embeds Anaconda, actually, quite specifically. So that's…yeah.

Jeff Reback (Guest) (37:26):

Absolutely, yes. You have the entire data science stack. And then, oh, by the way, if you need to go distributed, well, it's straightforward to do.

Peter Wang (Host) (37:33):

Yeah. This is great. Well, there's a lot more I'd love to talk to you about, but we're running out of time, and maybe we'll have a follow-up. And I'd love to talk to you about work-life balance and other kinds of things, and some of the other fun things between pandas and databases, and Ibis, SQL, whatever things. So hopefully in the future we have another conversation about some of those fun things. But Jeff, this has been an absolute pleasure and delight to chat with you, and thank you for taking time out of your busy day hacking and coding and managing and all these things, to chat with us. Thank you so much for joining us today on the podcast.

Jeff Reback (Guest) (38:03):

Thank you so much, Peter. This has been a great pleasure, and best to you and your family.

Peter Wang (Host) (38:08):

Thank you. Thank you for listening, and we hope you found this episode valuable. If you enjoyed the show, please leave us a five-star review. You can find more information and resources at Anaconda.com. This episode is brought to you by Anaconda, the world's most popular data science platform. We are committed to increasing data literacy and to providing data science technology for a better world. Anaconda is the best way to get started with, deploy, and secure Python and data science software, on prem or in the cloud. Visit Anaconda.com for more information.