EPISODE 1086

[INTRODUCTION]

[00:00:00] JM: The lifecycle of data management includes data cleaning, data extraction, integration, analysis and exploration, and of course, machine learning models. It would be great if all of this data management could be handled with automation, but unfortunately that's not an option. For most applications, data management requires a human in the loop. A human in the loop might be responsible for working in a spreadsheet or labeling data as a mechanical Turk or creating an algorithm for data labeling and snorkel, and data scientists and data analysts are humans in the loop as well. They're studying large datasets.

Aditya Parameswaran is an assistant professor at UC Berkeley and he studies human in the loop data analytics. He joins the show today to talk about the work and the projects that he is focused on, including data spread, which is an alternative to Excel; and OrpheusDB, which is a relational database versioning system.

If you want to reach 30,000 unique engineers every day, consider sponsoring Software Engineering Daily. Whether you're hiring engineers or selling a product to engineers, software engineering daily is a great place to reach talented engineers, and you can send me an email, jeff@softwareengineeringdaily.com if you're curious about sponsoring the podcast addict.

[INTERVIEW]

[00:01:22] JM: Aditya, welcome to the show.

[00:01:23] AP: Thank you for having me.

[00:01:24] JM: You're an assistant professor at UC Berkeley and one of your concerns is human in the loop data analytics. Can you explain what human in the loop data analytics is?

[00:01:38] AP: Sure. Human in the loop data analytics is a new area of research that many of us have started coalescing around. Human in the loop data analytics refers to the idea that it's

not just the data that's important during data analytics, but also the human who is performing the analytics, the analyst or data scientists if you will. Tools, infrastructure, techniques have to be designed with the end user in mind rather than simply thinking about the scalability of the data or the infrastructural aspect. It also needs to be taking into account the fact that the end users, many of whom may not have substantial programming expertise, should be able to easily get insights from data. That's the high-level principle.

How does that happen? Well, if you were to build tools to make it easy for end users to get interesting insights from data, the tools need to now do more of the heavy lifting. The tools need to be intelligent. They need to be support automate – They need to be able to be more automated. They need to expose intuitive mechanisms for the end users who may not necessarily know how to program. Interactive intuitive mechanisms so that these users can actually make sense of data without having to do a lot of work, and that work could be writing code, digging through lots of data, or what have you.

[00:03:06] JM: What human in the loop data analytics means in practice? Is this like I am a data scientist and I'm actually doing work munging data, or I am a data analyst, I'm staring at an Excel spreadsheet? What does this mean in practice?

[00:03:21] AP: Yeah. I mean, literally in every field, ranging from industrially, to the sciences, to government organizations, to medicine, you have this notion of a data scientist wanting to make sense of their data. Often, they have a lot of it. Often, unfortunately, the tools fail them. If you have a spreadsheet, for example, and the tool that you're comfortable with is spreadsheet, and spreadsheets are incredibly popular and for good reason. I mean, they are very intuitive. They are reasonably powerful. But when you try to load a large dataset into a spreadsheet, the spreadsheet fails. You can't really load a dataset that goes beyond a million rows into a spreadsheet.

A spreadsheet is a perfectly fine data and analytical tool, but it doesn't seem to work on large datasets. Likewise, many of the other tools that people try to use for big data, either they require a lot of programming or they don't work on large datasets. It's either pick your – Choose your adventure. You can't get the best of both worlds. Really, human in the loop data analytics is trying to build intuitive. It's about building intuitive tools that also scale so that you want the

expressiveness and the intuitiveness of spreadsheets, but also working on very large datasets, right? You want to be to specify at a high level what you're looking for, like a Google search interface, wherein you can type in a few keywords and the system have to find interesting webpages. What if you could do that with your data? That's the sort of goal.

In practice, yes, people are using tools like spreadsheets. They're sometimes using computational notebooks, like Jupyter. They are all writing code in languages like Python, or Scale, or Java. Often, if you're not super comfortable with programming, this can be pretty onerous and pretty overwhelming to end users. We really need to meet them in the middle with tools that can help them reduce their burden and sort of help do a lot of the heavy lifting for them. Do some of the automation for them.

[00:05:35] JM: I think about Google suite of business tools becoming more machine learning aware, like typing in, giving you an auto complete in your Google Docs, for example. That's one way that tooling could become more effective. What are some other ways that human in the loop data analytics tooling could become more effective?

[00:05:59] AP: Sure. There are various ways. The first is preserving the interface while increasing that the interface supports, right? Again, going back to the spreadsheet. You want to preserve the spreadsheet interface. You want people to use a spreadsheet just like they would while also being able to support billions of roles, or trillions of roles. In some sense, that's just one way. Just increasing the scalability could be really valuable for end users.

In fact, I have this anecdote of a genomics group that we were collaborating with at Mayo Clinic. This genomics group would generate very large datasets. They would generate a datasets around gigabytes large, and often these are people who are not super comfortable with programming. They won't be able to actually inspect these datasets to check for correctness. They would ship this to their bioinformatics collaborators simply to just have them open it, check it for correctness and then tell them, "Hey, you know what? We are good to go on this dataset."

Even something as simple as opening and looking at your dataset is impossible on current tools that domain users, end users will be able to be comfortable with. Spreadsheets wouldn't suffice.

They're super comfortable with spreadsheet, but spreadsheets wouldn't suffice for the simple task. The first goal would be improving scalability.

The second goal would be making it easier or more intuitive to express more complex needs, and this is where some of that intelligence comes in and guesswork comes in on the part of the system. The system could, like you mentioned, in email could help you auto complete your emails. That is an example of an auto completion. It's an example of intelligence. It's an example of automation. In the data analytics or data science phase, this could be trying to guess what the data scientist is trying to do and extrapolating from those examples.

One example of a system that we built is a tool that allows you to sketch a pattern with the system trying to find matches for that pattern. Let me give you an example. So let's say I am looking for a product whose sales has been increasing, but profits have been decreasing. To do so, you might provide a canvas where the user would actually sketch a curve on a chart saying, "Hey, I want something that's increasing for sales, but decreasing for profits. They draw a line on a chart and give it to.

Now, this is – I would call this a query. This is a query that the user has given to the system. Now, the system has to figure out whether any subsets of data match that query. Is it product A or product B? That actually matches that user provided specification.

The system has to do the matching process, and there's a lot of ambiguity in this matching process too, because if product – What does it mean for the product sales to be going up? There could be a few sort of bumps along the way. How do I classify a product as having matched that pattern? There's a little guesswork on the part of the system in meeting the users midway. Honestly, I think the two key mechanisms are in improving scalability, and the second is in intelligence and automation, meeting the users midway.

[00:09:40] JM: You built a number of tools that are seeking to realize some of the conclusions that you have about human in the loop data analytics. Could you give me an example of a tool that you've built and how that exemplifies what you're trying to do with human in the loop data analytics?

[00:10:01] AP: Sure. Returning to spreadsheet. I know that spreadsheets are too many. Spreadsheets are deeply unsexy topic, but it's one of the topics that's close to my heart. In the realm of spreadsheets, we built a tool called data spread. Data spread was – The starting point for us was to try to say, “Hey, can we preserve the spreadsheet frontend?” The intuitiveness, the flexibility of the spreadsheet frontend while also providing the benefits of databases. As we all know, databases are extremely expressive. They are powerful. They are scalable. Can we bring in some sense the two together? To preserve the frontend, replace the backend?

We build this tool called data spread, which is trying to do a holistic marriage of these two modalities. Preserve the features that people really crave from a spreadsheet’s endpoint, but also adding scalability. Now, the first challenge that we dealt with was the fact that, “Hey, look. Spreadsheets can't support very large datasets.”

A spreadsheet would act as a tiny window into the dataset, which is actually stored in your database. The entirety of your dataset would be stored in your database. The spreadsheet would basically be a frontend to the database. Next, we realized that just because you can look at a million rows in your spreadsheet, doesn't actually mean anything. How do you even make sense of a dataset that has a hundred million rows or a billion rows? Imagine scrolling through a spreadsheet with a billion rows. It's impossible to locate anything of interest.

What we decided was, “Hey, we need to build in intuitive navigation capabilities into a data spread so that you can zoom in and zoom out of a spreadsheet on demand.” Can we bring the analogies, the interactions that are so common in mapping software, like Google Maps, to a spreadsheet? What does it mean to view a spreadsheet at a course of granularity or a fine granularity?” We built in some of these functionalities.

The other opportunity that we realized in spreadsheets at least is that often people – In complex spreadsheets, people end up spending a lot of time waiting for a spreadsheet to complete computation and they end up waiting for 15 minutes, half an hour, and sometimes the spreadsheet crashes, especially if it is complex spreadsheets with financial information or accounting information where there's a lot of nested and complex networks of formula.

In such case, we decided, “Hey! You know what we can do? We can simply apply a UI trick to this. We can replace the in-progress computation with a progress bar,” like you would when you’re downloading files. We replaced the in-progress computation, the progress bar, and then compute this computation in the background as the users continue to explore the spreadsheet.

This is, in some sense, bringing the notion of asynchronous computation to spreadsheets, which has never been done before. It’s not synchronous and that you’re not returning the sponsors immediately. You are doing it in the background, and the user is aware of what is going on, because we’ve replaced the computation with a progress bar. We’re basically improving the interactivity of spreadsheets in this manner. This is one example of a tool that we’ve built. If you like, I can give you another example too, or I’m happy to move to something else.

[00:13:40] JM: I would like to know, have you shown this to people who are in situations where they might be using a spreadsheet in the wild? Have you seen practical applications of it and have you gotten some prototypical users, and have you gotten some feedback from them?

[00:13:57] AP: Right. Going back to my genomics example that I mentioned earlier, we work with those users to provide a spreadsheet interface for them. They were our start starting point for this entire project. They’ve been sort of really good potential users for data spread. Data spread is still very much a work in progress. I wouldn't say it is a finished product, but it’s still a work in progress, and so we’re continuing to work with these genomics users who were our initial target users and continue to be one important target users for us.

The other thing that we are doing is also partnering with folks who build spreadsheet systems. Been working with folks in Google Sheets as well as LibreOffice and trying to translate some of the lessons that we've learned into these commercial or open source spreadsheet platforms.

[00:14:55] JM: When you talk to the – I’m very curious about the interactions with the Google Sheets people. What have come out of those conversations? What do you think lies in the future of their spreadsheet platform?

[00:15:10] AP: What I can tell you is that it's clear that they recognize that there is in need towards democratizing access to very large data and they view very large datasets as an ex-

frontier for Google Sheets as well, because this is coming to them from their enterprise customers. A lot of them who are telling them, “Hey, you know what? We have these very large datasets. We would like to analyze it in Google Sheets. Why does Google Sheets crass, or freeze, or break when I try to load in this dataset?”

They are aware of some of these issues, the fact that in some sense this spreadsheet is a great intuitive, flexible tool, but also constrained by what they have done so far. There’re a lot of backward compatibility issues. In some sense, taking a clean slate approach to the spreadsheet universe, like we’ve done, can inform their – In some sense, an ideal test bed to see if IDS will work or not and if this leads to sort of more adaption, more usability, more scalability. And then they can adapt it in their systems.

[00:16:37] JM: Okay. It's kind of exciting. I mean, the idea of working with really, really large datasets productively in a spreadsheet. I mean, I think it’s something that people would want to do, but it's not really feasible today. Let's zoom out and talk about the lifecycle of data analytics. When I think with the lifecycle of data management, you've got data cleaning, you've got extraction, you've got analysis and exploration. You've got machine learning. You've got a collaboration. Tell me about the outstanding problems in the lifecycle of data management.

[00:17:15] AP: That's a great question, and it’s a really important question. I think it's [inaudible 00:17:18]. In some sense, there are various steps in this data analytics pipeline ranging from cleaning, to exploration, to analytics, to machine learning, like you said, and possibly even deployment of machine learning models and what have you.

Unfortunately, current tools for each of these stages are not tied to each other. You often have different tools for each step. You may start with – There are tools like spreadsheets, which are great if your data has already organized in a developed fashion. You have other data preparation and data cleaning tools. Trifecta is a great example of that. There are tools that allow you to do visualization. Tableau is a great example of that. Looker is a great example of that. And then you have tools from machine learning and data science. At least data science increasingly is happening in computational notebooks.

Now, why are these tools all so different from each other and why are they catering to different audiences? Now, to me, I think there's a missed opportunity here. Can we think about a tool that combines the benefits of all of these tools out there and also caters to a broader fraction of the population? Is there that of magical tool that combines the benefits of spreadsheets, computational notebooks, interactive visualization tools like Tableau, as well as like data prep tools, like Trifecta? Why should I use separate tools for each of these steps in the pipeline?

Another challenge here is the fact that I even as someone who's fairly proficient in programming do use spreadsheets. I do enjoy using spreadsheets once in a while when I want to quickly go and edit some data. At times, I might still want to use my data preparation tools, like Trifecta. I might want to – I often work in computational notebooks. I actually use all four of these types of tools, but still in some sense, I have to pick between them given the problem at-hand, and I have to move data across them, which is not ideal.

The Holy Grail would be a tool that combines the benefits of all of these tools Interactive visualization capabilities, that ability to write code and see that results immediately like in computational notebooks. The ability to quickly go and edit your data like in spreadsheets and also sort of get the data into the shape that you want like in Trifecta, right? Those are the sorts of things that I would, as a data scientist, really like.

Machine learning is the next step of the pipeline, and there's a whole sort of can of worms in terms of how machine learning training happens. How model development happens? How deployment happens? None of this is done rigorously. Again, this happens across a bunch of different tools. There is no clear emerging consensus in how this should be done well.

Overall, I think there's a huge opportunity here for people to come and consolidate this space. Thinking about the end-to-end data analytics pipeline, but also thinking about the skillset of users. You want tools that cater to both novices in terms of data science, but also the experts. And experts often also want to use tools meant for novices. Experts also want to use interactive visualization tools like Tableau and spreadsheets. You want these functionalities to not just be constrained to one faction of the population.

[00:20:51] JM: Okay. The other tools that you have built, there's Zenvisage, OrpheusDB. There're a number of other tools that come to realization in your research. How do you have so many projects in flight?

[00:21:10] AP: I would say it's part of the joy and the dangers of being a professor that you can have many PhD students who each of whom drive an independent project. Often, groups of two or three PhD students end up getting together and driving a given project. Like in the case of OrpheusDB or Zenvisage, it's often a small cohort of two to three students who in some sense bring different skills to a project. One of them might bring interaction, or HCI, human computer interaction skills. Another one might bring system skills. The third one might bring machine learning skills. You have a small cohort of people who brings sort of various skills to the project and then they build out a prototype and then try talking to end users and seeing the missing sort of missing aspects and improving on it.

Really, that is how we do things in my group. We sort of have different projects ongoing at a time, many different projects actually. I try not to have too many. I try not to have more than five at any given time, but for each project, there're usually one or two students, PhD students, who are driving it and they're really the force behind the project in terms of doing the implementation, generating ideas and setting the vision for the project. At some level, I feel like all that I'm doing is providing very high-level guidance and making sure that there are no roadblocks in their way.

[00:22:40] JM: Okay. As we can continue through this world of data analytics, I want to refocus on human in the loop data analytics and just get a better understanding of how you think the role of the human in the loop changes overtime, because obviously the tooling is going to improve, and I think a lot of the – Some of the roles that humans play today can be supplanted or replaced by machine learning tools, like image classification, for example. And I just want to get your perspective on how the role of the human in the loop changes over time.

[00:23:25] AP: I think that is a fine line between seeding too much autonomy and seeding too much productivity to the to the system and also sort of to make the most of the system capabilities while also preserving some amount of control. The way I would think about this is if you have a system wherein everything is automated. If you're moving towards automated machine learning, for example, this from the perspective of an end user would be good,

because they don't have to do much work. They might not have to do feature engineering. They might not have to do data preparation. They might not have to do parameter tuning and so on and so forth. This is less work on the part of the analyst or a data scientist who's doing the analysis or machine learning. The system does most of the work.

Now, the downside of that is that the system may make mistakes and you may end up with the objective. You may end up with the system ending up developing a model that is optimizing for the wrong objective. Having that fine-grained human control while the machine learning this happening is still very important to ensure that the system is not going completely awry, right? As an example, this is anecdote of a machine learning algorithms that was chained on images across two different hospitals, and one hospital was a bigger hospital that was catering to the broad population. This is chest X-rays I believe. One hospital that was catering to the broad population. Another hospital that was catering to – That was a research-oriented hospital that was targeting the more "deadly cases".

Now, data from both these hospitals was fed into a machine learning model, and the machine learning model, perhaps somewhat surprisingly, was able to do a really good job on this radiological task surpassing human experts. It was able to detect problematic cases way before way better than human experts.

Once humans decided to go back and dig into why this machine learning algorithm was doing so well, they realized that all that the machine learning algorithm was doing was detecting that the images that were produced in the machine in the hospital that was a research hospital also had a tag corresponding to the machine that was generating it.

The machine running algorithm was simply looking for this tag and then saying, "Hey! These are images of patients who had more problematic conditions than those that did." It was just over-fitting based on signatures output by the imaging software as supposed to actually doing something on the core task. This problem would not have been caught if there were no humans in the loop. You really needed humans in the loop to be able to understand where this problem came from and how did the accuracy end up being so high.

Overall, think there is this balance of how much autonomy should a system have in the data analytics space is still an open one. I think I still think that leaving everything up to the system is a dangerous one. At least at the initial stages of exploration, there is a really a lot of context, insight and input that needs to happen from the data analyst or the human in the loop providing inputs to the system. Making sure that the system is not doing something that is not what the data scientist or human in the loop intended.

Really, that guidance needs to be a lot more at the start. Maybe towards the end when it's close to deployment, the guidance can be eased off, right? So you can move towards more automation. At least the start, [inaudible 00:27:28] objective, set the context. Making sure that the system is doing the right thing. There is a need for a lot of human input.

[00:27:36] JM: A project that you've been working on that is closely related to data science is Modin, which is a faster alternative to pandas that uses the same API. We actually have a show coming up on Modin, but I'd love to get some perspective on your end. What are the shortcomings of Pandas? I guess you could explain what Pandas is and describe the shortcomings of it and then we can talk about Modin a bit.

[00:28:06] AP: Sure. Pandas at a very high-level is data frame library. It's really a library for processing and working with tabular data in an intuitive manner. Data frames have their history in the statistics community. The R programming language had a data frame implementation, and really you could view this as a set of observations. These are your rows. And aspects of those observations, these are your problems, and it's really just a table if you think about it that way.

The difference between data frames and tables from a database or a relational context is that data frames are a lot more flexible. You don't need to rigidly obey a schema like you would in a database. You can have it be more ad hoc and more flexible and it can continually evolve over the course of data science.

Now, the fact that it's more flexible means that it's a good fit for data science, because at least the start of your data analysis process, your data is often not structured in a way that you would want. You want to transform it so that it fits the more that you want so that you can then do the analysis or then do the machine learning. Pandas is really valuable in bridging this gap going

from a somewhat structured dataset to a fairly structured dataset that you can then apply to your machine learning pipelines.

That was the original goal of Pandas, and Pandas is a Python implementation of a data frame library. Again, data frames came from R in the statistics community specifically. Pandas over the course of the last decade has had many, many contributions from various open source contributors adding more and more functionality. Right now, Pandas has like 250 operators that you can apply to what looks like a table. You can, for examples, filter your table. You can join your table with other tables. You can transpose it. You can – There are various sort of types of operators that combine both linear algebra as well as relational algebra and expose all of that functionality to a table.

Now, what Modin was trying to do – While it's great that there's so much functionality. So you have 250+ operators. Part of the challenge that many of these operators are not optimized. They're not efficient. They don't work on very large datasets. Again, these are all community contributions. These are all open source contributions. Every time someone identifies the need, they will implement something and add it to the codebase, and now you have a new function. But this function is not necessarily optimized.

In some sense, what our goal with Modin was take a step back and think about how do you optimize a space? Provide similar functionality to Pandas, but also allow it to scale to large datasets. This is a team and a lot of my work. Really thinking about preserving functionality that end-users like, but allowing it to scale up to larger datasets.

We are applying once again database principles to allow us to optimize Pandas queries or Pandas expressions, right? Can you apply the familiar techniques of query planning and query optimization that we've developed over the course of the last 30 years in the database community and have it bear on this important problem, which can have an enormous impact because of the popularity of Pandas.

Pandas, people say is a reason why Python is so popular. It's called as a lingua franca of for data science. It's important for us to make it better. It's important for us to make it faster. It's important for us to make it more scalable. That's why we built Modin.

[00:31:55] JM: The work around Python's data science ecosystem – We had a show recently about Dask, and that's a way of building these distributed scalable objects for manipulating large datasets. How does Modin compare to the Dask project and the scalability approaches to data science there?

[00:32:20] AP: Sure. The Dask project as well as another project called Ray, which is another distributed executor, are both what I would consider as very capable backend for Modin. Modin preserves the Pandas API while just trying to scale up the execution thereof. Modin's secret sauce in some sense is a translation from the Pandas API calls into a set of operations that can be then distributed and executed using a distributed scheduler like Dask or Ray. You can imagine that Modin sits between Pandas and Dask, or Ray, and in fact can use both of these as backends.

[00:33:08] JM: Okay. The value of being able to use either Ray or Dask, what is that? If I can just get a better understanding for what that middleware layer is.

[00:33:21] AP: Right. Unlike why does Modin need to exist given Dask or Ray, well, the first thing is that MOdin preserves the Python API. You don't need to do anything – Sorry. Pandas API. You don't need to do anything different. Anything that you want to do in Pandas will continue to execute in Modin.

Now, for Dask and Ray, the reason why we wanted to do this was the said, "Look, let's ensure that Modin works with any distributed execution framework, not just Dask or Ray, so that if there is a new distributed execution framework that comes by, we would be to use it. In fact, one of our original goals was to actually use databases as a backend to Modin. Again, have the databases be a third option to Dask or ray.

Now, you may debate as to whether it's a good idea. It may not be, but the goal was to not be tied to any one executor and have it be more generic, and that was the reason why we decided to go with both Dask and Ray to provide some flexibility and generality for Modin.

[00:34:30] JM: What are the other outstanding problems in data science, and particularly data science scalability that you're concerned with?

[00:34:41] AP: Sure. I'd like to return to some of the projects that you've mentioned and tell you about some of the things that we are working through on those projects. One of those projects is Zenvisage. Zenvisage, the goal of Zenvisage was to really say people spend a lot of time looking at data visualizations. Often, we found that our scientific collaborators spend all of their time pouring through these visualizations trying to find patterns.

Our goal was can we accelerate this search for patterns? Because if you're spending hours looking through visualizations trying to find patterns, that is not a good use of your time, and this is the reason why they were spending this much amount of time was because often they don't have the programming chops to be able to quickly write some software that would help them look for these patterns. They were looking for these patterns manually in a tool like Tableau or Excel.

I think, really, thinking about fast forwarding to their patterns or fast forwarding to the desired insights is a key question that hasn't been fully addressed yet. We have made some progress on that question, but I don't think it's fully addressed yet.

The other question which I think is really important is that data science is happening in a really, really ad hoc manner. You have sort of workflow-specified and computational books, but these workflows are often not reproducible. The datasets that you generate over the course of these workflows are often not recorded in a principled manner. So you have many, many versions of each dataset floating around. We all encounter this. If you ever looked at our downloads folder or our files folder, you'll have many, many versions of each of your Word files, and your text files, because you have stored your datasets at previous stages of experimentation.

Orpheus, which is another project that we've been spending a lot of time on is designed towards tracking and versioning your data and your code so that you can make sense of what has happened and retrieve quickly on-demand. Can you, for example, say, "This is new erroneous stupor that seems to have been introducing in my dataset. Who introduced it? Why did they

introduce it? When was it introduced?" So can you point at various aspects of your data and ask questions of this type?

Similarly, if you have two versions of your dataset, can you do a dif between them so that you can understand the differences across them? Those are the types of questions where, really, thinking about data lake management, you have a data lake where you have lots of versions of the dataset. You want to make sense if it. You want to have the analysis be reproducible I think is still a very important and open question.

I would say, broadly – Okay. The other interesting and important open question is in terms of managing machine development, and this relates to the data lake management as well. But right now, machine learning is again happening in a very ad hoc manner. One of the opportunities that we identified was that – People are developing machine learning workflows. They'll write a workflow end-to-end. They'll run it, and maybe they will find that their model doesn't have adequate accuracy. They may then go and make a small change. So they may change some feature. They may change some regularization parameter. They'll make some small tweak and then they'll rerun this workflow from scratch, and often this would take – If it took two hours the first time, it's going to take two hours the second time and two hours the third time and it would basically be wasted effort.

Instead, our tool was targeted at capturing intermediate data during the course of machine learning execution so that we can accelerate this process so that we can identify, "Hey, you know what? All that you did was this small change. We'll make the work flow around a lot faster, because we can automatically reuse previous work that you've done."

If it took two hours of first time to run. It might take two minutes the second time to run and two seconds the third time to run, because it's already cached and materialized and recorded stuff that you've done previously and reusing some of that in the future. This is a project called Helix that we've spent a fair bit of time on.

[00:38:51] JM: These research projects, do you have a vision for – Do you see them as like case studies or do you see them having direct and practical application?

[00:39:01] AP: Right. I think it depends on the project. Modin has been used by a lot of folks. It's been adapted and used and also has a lot of contributors. Zenvisage has been deployed in various groups. One group who was doing battery science, another that was doing astronomy, a third group that was doing genomics. These groups have used Zenvisage. Helix, some of the Helix ideas have yet to make their way to end users. We're still in the process of developing that.

Like I said, some of the spreadsheet work was designed with some of our genomics users in mind. Again, we are working with end users. Overall, the way our group approaches these software development efforts is to work closely with end users to make sure that the problems that we're solving are real problems and not none problems, and building tools to make sure it meets their needs even if it's for a small group as supposed to making sure we build a generalizable tool that addresses everyone's needs, which may never exist. We want to make sure that at least a small group of potential end users would be happy using what we're building. That is a philosophy that we adapt. Some of the students that I work with want to write research papers, want to do the research, but don't necessarily want to spend a lot of their time engineering, which is fine as long as they are making tangible research advances that will have impact. I wouldn't necessarily force my students to go one way or the other. It's entirely up to them. A lot of my students tend to veer towards doing things that also can be used by end users.

[00:40:46] JM: Cool. Data science and data analytics, this is a broad term. It's a broad field. How you see the next five years developing in terms of data science?

[00:40:59] AP: I think the next five years are going to be interesting, because I think, overall, there's been such a vast number of data science tools that have been developed both from the open source community, the startup community and in academia, and we're going to see in some sense of coalescing around a small set of tools or a small set of core guiding principles. It might be that everyone coalesces around the computational notebook ecosystem. It might be that everyone coalesces around – I don't know, the spreadsheet ecosystem. It's hard to predict what these coalescing points would be, but I think moving forward, we're going to see a lot of – I think we've sort of expanded in some sense, and you have a plethora of things that you can use, but none of which perfectly address what you're looking for. In some sense, I think coalescing, focusing on a smaller number of tools that are more general that sort of solve

problems end-to-eend I think would be interesting and I think that's where if I were to make a prediction, I would say that's where the industry and academic research is likely to go.

[00:42:10] JM: Okay, very cool. Aditya, thank you for coming on the show. It's been really awesome talking to you about all these stuff.

[00:42:15] AP: Thank you so much for having me again.

[END]