EPISODE 363

[INTRODUCTION]

[0:00:00.4] JM: Instacart is a grocery delivery service. Customers log onto the website or the mobile app and pick their groceries. Shoppers at the store get those groceries off the shelves. Drivers pick up the groceries from the stores and drive them to the customers. This is an infinitely complex set of logistics problems paired with a rich dataset given by the popularity of Instacart.

Jeremy Stanley is the VP of data science for Instacart. In this episode he explains how Instacart's foresighted marketplace business is constructed and how the different data science team break down problems like finding the fastest route to groceries within a store, or finding the best path to delivering groceries from a store to a user, or personalizing the recommendations so people can find new items to try. It's a really interesting episode about some applied data science.

If you're looking for old episodes of Software Engineering Daily but you don't know how to find the ones that are interesting to you, check out our new topic feeds in iTunes or wherever you find your podcasts. We have sorted all 500 of our old episodes into categories like business, and blockchain, and cloud engineering, machine learning. We have a greatest hits feed that has all of our best episodes, our most popular episodes, and these will be updated overtime. You could subscribe to all these categories instead of subscribing to this main feed if you want or subscribe to a subset, whatever's interesting to you. You can check the show notes for more details about these, and I hope you like today's episode.

[SPONSOR MESSAGE]

[0:01:48.4] JM: Artificial intelligence is dramatically evolving the way that our world works, and to make AI easier and faster, we need new kinds of hardware and software, which is why Intel acquired Nervana Systems and its platform for deep learning.

Intel Nervana is hiring engineers to help develop a full stack for AI from chip design to software frameworks. Go to softwareengineeringdaily.com/intel to apply for an opening on the team. To learn more about the company, check out the interviews that I've conducted with its engineers. Those are also available at softwareengineeringdaily.com/intel. Come build the future with Intel Nervana. Go to softwareengineeringdaily.com/intel to apply now.

[INTERVIEW]

[0:02:44.6] JM: Jeremy Stanley is VP of data science for Instacart. Jeremy, welcome to Software Engineering Daily.

[0:02:49.5] JS: Thanks so much, Jeff. Really excited to be here.

[0:02:51.8] JM: Instacart is a grocery delivery service. I use it once or twice a week. I love it. We're going to get into the data science of Instacart in this episode, and let's start by describing how the company works. I've heard you describe it as a four sided marketplace. What are the four sides of the marketplace?

[0:03:15.1] JS: Yeah, I think that is pretty fundamental. The first two sides are pretty obvious, you have on one side the consumer. The consumer is shopping for groceries and having them delivered, and then on the other side you have the personal shopper. That personal shopper is going to the store, picking out the groceries the customer has ordered and delivering them to the customer's doorstep.

The third side of this marketplace is the retailers. Instacart has 160 different retail partners and some of them are household names that everyone in the nation would likely know. They would be names like Wholefoods, or Costco, or Target. We just launched CVS nationwide, but there are also lots of retailers that are more local brands, so it's the publics in the southeast which has thousands of stores. If you life in Florida or Atlanta, you know and love publics, or HEB in Texas as an example.

There are those regional stores all over the U.S. that we partner with, and the retailers are really important to us. Customers already have relationships with those retailers, they trust those

retailers, the retailers are making the inventory available to us. Our shoppers are going to their physical stores, so we have really tight integrations with those retailers and are thinking about them all the time.

Then the fourth side of the marketplace or the products and, really, it's the consumer packaged goods companies behind the products. These are all of the groceries that our customers are shopping for and we care not only about the products themselves and being able to surface those products to customers, find them in the stores, but we also want to offer advertising solutions to those partners. They want to promote their product on our website, and maybe it's something like a featured item in a search result or maybe it's a coupon where that customer gets a dollar off if they buy two of that product. The consumer packaged goods company actually pays Instacart to offer that coupon to the customer. It's a win for us, a win for the customer, and a win for the advertiser as well. Those are the four sides.

[0:05:30.8] JM: I heard an episode of the How I Built This Podcast with your CEO, and one of the things he mentioned in that episode that was pretty cool to me was the fact that at least some of the supermarkets, the deal between the supermarket and Instacart is that the supermarket pays a subscription to have their store indexed by Instacart, and that idea of turning — I mean, what I assumed originally was, "Oh! Instacart does a delivery for store. Maybe they take a cut. Maybe they take some flat fee," but the idea of a subscription had not crossed my mind. I guess that example opened my mind to make me realize that this is just a big marketplace and there are a lot of different incentive structures you can create. How do you think about the importance of incentives in the different sides of this marketplace and how do you make sure that the incentives are aligned?

[0:06:35.9] JS: It's a great question. I think if we're specific about the example you gave — So some of our retail partners, they will pay us a percentage of the gross merchandise value. If you spend a hundred dollars on groceries, the retailer might give us some percentage of that as incremental revenue to Instacart. The reason they do that is they want to work closely with Instacart and have Instacart offered for their retailer to as many customers as possible. They find that when they do that it increases their same-store sales.

A given store footprint now starts to serve X% more customers, and those X% more customers will be a lot more profitable to the retailer because they've already sunk the cost in managing inventory and having a real estate expense and the people working in the store. Being able to drive incremental sales to new customers that wouldn't have bought already, something that our retailers really care about, they test it, and that's why they're paying us a fee.

Ultimately, they know that customers want to buy groceries online, and Instacart is a great partner for helping them to do that, to ultimately making their customers happy. That's the specific incentive there. I think, in general, it's tough. When you have four sides to the marketplace, any product change that you make, any feature that you ship, any algorithm you put into production, it's going to have an impact on all four sides. Maybe it's increasing basket sizes.

Increasing basket sizes is probably good for the customer. I think they appreciate that. It might be good for the retailer because they're going to have larger basket sizes per customer. It's good for some advertisers, some of the partners, if their goods are the ones that are going into the increment basket size. If your wage structure is fixed per delivery, it might actually be bad for the shopper. That's a really obvious kind of simple example. There are many that are much more nuanced.

I think that it's hard in a four sided marketplace to look for product or algorithm changes that are a win for everyone in the marketplace. Oftentimes, our strategy and what happens is we ship a portfolio of changes that might each be good for two or three of the sides and, yet, in combination, they're good for everyone.

I think the other side to this is that for any of our key performance indicators, there are some counter measure. If we increase how available we are to customers, we're more likely to have our shopper sitting idle. We will always look at both of those metrics. We'll have a key indicator that we care about and we'll look at the counter-metric that if you were to do something really naïve to try to manipulate that key metric, like just staff for shoppers, the counter-metric is going to suffer. Then we try to look for moving the efficient frontier on those two metrics.

[0:09:34.7] JM: The strategy that we've seen lots of marketplace businesses take in recent years, a lot of the on-demand companies. Strategy is typically to grow aggressively in the early days and not think so much about costs, and then once the company reaches a certain size, they start to use the economies of scale that are inherent in reaching that certain size to look for better efficiencies, and I think this gets us towards the data science conversation because it is only with a certain scale that you start to be able to make conclusions within a reasonable degree of sureness about your business. Instacart is at that scale at this point where you can start to make unique conclusions that are backed up by the data. How do you identify the right opportunities in that sea of data that are going to allow you to reduce the cost or improve the efficiency to get the business closer to where you want it to be?

[0:10:43.6] JS: Yeah, in many ways, that's the golden question. If we could all identify the right things to work on 100% of the time, we would be 1,000 times more productive and efficient. I think that data can play a pretty important role there, but it's not the only answer. I think that we try to make all of our decisions using data, but what that really means is we try to think about the business from a first principle's perspective. What is it that are the kind of fundamental physical drivers of the success of our business, and some of that comes down to finance. Some of it comes down to behavior. Some of it comes down to physics; space, time, density, for the efficiency of our delivery.

Once we do that first principles-based analysis of the business, we then start to look for, "Okay, for any given one of these key dimensions, how do we measure it and how we quantify the impact that incremental changes to that metric are going to have on the business? How to measure it is sometimes pretty straightforward, but usually there is a lot of nuance in that. Maybe we don't have the right measurement instrumentation. Maybe we're not capturing exactly the right data, we haven't asked customers the right question. We're not logging in the right frequency. We're not controlling for the right effect. That requires a lot of diligence to go through and try to understand are the data that we're collecting and the metrics we're looking at really closely align to the first principle drivers of the physics of our business? Then you can ask, "Okay, well once we've measured these things, can we prove whether or not an incremental change is going to really matter?"

You can try to do a lot of historical analyses, and we do. We will take any one of the key facets of our business and use all of the historical data we have to try to understand what might be root causes, what might be relationships that would be meaningful that we could act upon?

I would say those analyses are more about inspiration than they are about definitive conclusions, because it's inevitably biased historical data. The sophistication of our systems changes overtime. The nature of our customers change overtime. Who we are delivering from and the retailers changes with time. There are seasonal effects. There are a litany, a long long list of things that could bias the data.

in the end, we use those analyses for inspiration, but what it comes down to is coming up with an idea for a product or an algorithm change, implementing it as quickly as possible, measuring the A-B test and seeing whether or not it really moves the metrics we care about. In the end, that's where I think our deepest insights come from are the successes and failures that we have in experimentation.

[0:13:41.6] JM: You've written about the different teams at Instacart, or at least some of the teams for whom data science is critical, and there's a great blog post about this that I'll include in the show notes. One of these problems is fulfillment. You've got team or teams responsible for fulfillment; the process of getting groceries from the store to my house. What are the different variables that are being weighed and traded off in the fulfillment algorithm?

[0:14:17.4] JS: Great question. In order to understand that, I think let's first make sure we understand the service that we're offering, the product if you will. The customer places an order for, let's say, 15 items from a retailer, and check out they select a delivery window, and that delivery window could be in the next two hours or it could be some one-hour window in the future. Once we've accepted that delivery in that window, it's the job of our logistic system and our fulfillment system to try to fulfill that order as quickly with as high quality as possible. What does that mean?

First, let's talk about quality component. There might be three or four different store locations that we could fulfill that order from. The first question that we ask is; what is the probability that each of those 15 items is going to be available at those store locations? We can, for any one of

the store locations, then ask the question; which store location is going to have the best chance of giving the customer all of the items that they want?

The second is we want to make sure that our shoppers can move as fast as possible. In order to do that, we can do a few different things. First, the shopper who is closest to the store location we want to fulfill from, they'd be a great candidate to fulfill this order. Second, we can actually shot multiple orders simultaneously and we can do that in a couple of different ways. One way is to actually send a shopper to the store and they will pick two different orders at the same time and then deliver one and then other, or maybe even three at the same time and deliver the three in sequence. We may also have what we call in-store shoppers at that store who can pick order after order after order and then stage them in refrigerators, freezers, in pantries that we have staging areas at the store.

Once those orders are staged, we can send a shopper to the store to pick up a set of, say, five orders and deliver those in sequence. Our optimization challenge is to find the shoppers that are closest to the store location, we're going to pick up as many orders as possible and deliver them in a sequence such that that sequence is close together in space and time and the deliveries are delivered on time. That the last component. We want to make sure that we're not five minutes late or even five minutes early. We want to be 10 minutes early from the end of the delivery window, because if we show up in the last few minutes of the delivery window, the customers aren't that happy. They'd rather us show up 10 or 15 minutes before the delivery window ends.

There's this giant vehicle routing problem with time windows that has to be solved to figure out which shoppers can do what sets of orders in what sequence such that we move as fast as possible such that the deliveries are all done on time and we ensure that customers have the highest chance of getting the items that they want.

[0:17:25.7] JM: In that complex sequencing of different events that is the fulfillment process, there's a lot of different data sources and I'm interested in how you aggregate those data sources and how you keep them clean. What are those data sources? What are the ones that are internal and what are the ones that are external?

[0:17:50.3] JS: I think that there are two key things to think about from the perspective of data in the end-use. One is; what is the current state of the system? The second one is; if we do X, what do we anticipate to happen? The current state of the system is really encapsulated by the shoppers' phones and the GPS signals that we're getting from those phones. From within our app, the state that they're in their process, have they accepted an order? Are we still waiting for them to acknowledge an order? Are they delivering to a customer address? Are they sitting idle?

We will measure that state. We'll measure their GPS location, and so that gives us the current state of the shoppers. The current state of the customers, for all of the customers that have checked out, we know the items that they want to have delivered and we know the time window that we've committed to them.

We also know all of the customers that are on the site in process, they're building their carts, we're waiting for them to check out. There's a probabilistic chance that any one of those customers are going to convert. That's the current state.

The production piece is, "Well, if we were to give this shopper this order, how long would it take them to pick all of the items? How long would it take them to drive from this location to this store, or from this store to this customer address, or this customer address to a subsequent customer address? More than just how long would it take them to drive? What's the distribution of those drive times?"

If you think about the API, all we need for the optimization is the state measurement and the set of predictions. The state measurement from a data perspective is pretty straightforward, but the predictions I think are the interesting component because that's really where we get the leverage a bunch of different data. We can use all of the historical data about our shoppers, their locations, how long it's taken for them to do different things. We can use weather information. We can get real-time traffic information. We can look at the stores and we can understand at any given store how long it takes to pick any given item and how long it might take to pick a batch of items, given there's 15 or 20 or 50 or whether or not this is a fast shopper or a slow shopper.

Then we're getting all sorts of third-party data from our retailers about what they expect to have in inventory, and we're seeing our shoppers go to stores and not find items, and so we can use that information to predict whether or not they'll find an item in the future.

[SPONSOR MESSAGE]

[0:20:39.4] JM: Dice.com will help you accelerate your tech career. Whether you're actively looking for a job or need insights to grow in your current role, Dice has the resources that you need. Dice's mobile app is the fastest and easiest way to get ahead. Search thousands of jobs from top companies. Discover your market value based on your unique skillset. Uncover new opportunities with Dice's new career-pathing tool, which can give you insights about the best types of roles to transition to.

Dice will even suggest the new skills that you'll need to make the move. Manage your tech career and download the Dice Careers App on Android or iOS today. To check out the Dice website and support Software Engineering Daily, go to dice.com/sedaily. You can find information about the Dice Careers App on dice.com/sedaily and you'll support Software Engineering Daily.

Thanks to Dice for being a loyal sponsor of Software Engineering Daily. If you want to find out more about Dice Careers, go to dice.com/sedaily.

[INTERVIEW CONTINUED]

[0:21:59.3] JM: You have a problem with so many variables like this. I think one of the revolutions that we've had in machine learning or data science recently is the spreading of knowledge in how to deal with these really complex systems. Whereas in the past, companies might have gotten bogged down with a rigid rule-based system that was architected in a monolith and all the code is in a monolith and is just a disaster to try to add something new or try to understand what's going on or to try to isolate a single variable. Can you describe what you have learned personally about building a system that allows you to zoom in on one of those data source or isolate one of those data sources? What are the best practices when dealing

with a highly complex system with all these different streams of data? How do you know that you are assessing it accurately and not getting confused?

[0:23:10.0] JS: I think that is — You're right. We have made a lot of progress as a field, and there's been a lot of great articles and research talking about the complexity of machine learning and how do you go from a product that has market fit but very little intelligence to something that really resonates with customers?

If you look at those experiences that people have had, it's rare that you build something really complicated in the beginning. The first iterations should be about something very simplistic where maybe it is just the rules engine or maybe it's a very simple regression and you're really focusing on measuring the outcome and trying to prove that you're evolving product in the right direction and kind of building a framework to do that, and then you gradually increase the sophistication until whatever you're optimizing for in the machine learning. When you deploy it, you find an A- B test no longer has the right impact.

I think that that's a kind of journey that is incredibly important to understanding how to frame the problem up front and start with something really simple, how to measure its efficacy and then how to slowly add complexity to it with time. Then I think there is also the question of how do you subdivide a very complex problem into these constituent parts. I think that that's really really hard. I think that we have, at times, organizationally, divided the problem into this group owns this half of the virtuous cycle and this other group owns this other half of the virtuous cycle.

We're going to iterated on building those halves independently, and we're able to initially make lots of progress and we're able to knockdown a lot of low hanging fruit and everything is working well, but then you reach a point where the left hand doesn't know what the right hand is doing and maybe even the left hand is making faulty assumptions about the right hand and they're at odds with each other. The only way to make additional incremental progress is to break those barriers down and reframe the problem and try from a different perspective.

We've absolutely experienced that, and I think if I could give myself one point of wisdom to take back in time, it would be to look for those issues sooner and pay attention to them faster and

break down the organizational berries even faster. We might have been able to gain two or three months in execution if we had done that in certain cases.

I think the other challenge with this problem is that you can't A-B test a lot of the changes that we make. There isn't a natural unit of experimentation. It's like Facebook. A lot of changes that Facebook makes, they affect pairs of Facebook members. They don't affect just an individual. You can't release a change to Facebook to 1% of the population because that 1% is going to use that change to interact with the other 99%, and it changes the other 99%'s behavior. In fact, if the 99% doesn't have that feature, they might behave in a very weird way. You can't easily do a A-B test for a network.

We have a similar, maybe even a harder network problem and that if we were to take half of our shoppers and try to route them with a specific algorithm, it might starve the other half of the shoppers, or it might lead to a negative outcome or a positive outcome for the other half of the shoppers, and that's not really what we would find if we deployed it to the entire geography.

Our A-B tests, if we're going to conduct them, have to be at the geography levels. We'll change something in San Francisco but leave it in Austin the way it was. Maybe you'll be able to see a significant change by tracking those to geographies overtime, but that's pretty course and pretty noisy.

Increasingly, what we've done, is to build a simulation framework so we can model what the outcomes would be replay history and draw from past deliveries to reconstruct the a hypothetical Monday in San Francisco and deploy five different changes, five different routing algorithms and find out, "Well, what is our planned efficiency? What is our planned late? What's our planned accuracy?" Then we will take the best ones and ship them and look to see whether or not in reality we observe the same outcomes and the same kind of sub-metrics underneath it, does the whole kind of system seem to be behaving in the same way as a simulation? If it is, then that's fantastic. If it isn't, then it means there's something wrong with the simulation. Increasingly, we're trying to do that to make smart decisions.

[0:28:14.9] JM: That's quite an incredible approach. I've seen that approach in Wall Street, but I haven't seen it in these new marketplace businesses, although I don't doubt that there are other on-demand companies that are doing it.

Another question I wanted to ask you was about the supply and demand problem that you also wrote about in this blog post. You need to have the right capacity of customers and shoppers who can receive the orders from those customers, because the whole idea is, Instacart, you get your groceries quickly. You need to have shoppers that are available, but of course the shoppers don't want to become disenfranchised and just sitting at the store not doing anything, not getting enough orders from customers. It's kind of a delicate balance. The main knob that you can tune, from my point of view at least, is the price, which is what you're paying the shoppers, because I don't think that there is variable pricing on the groceries, at least for the stores I've shopped at.

When you're studying the response to a change in market price for the shoppers so that you can get enough shoppers on board but not too many, to serve the customers, are you also doing this simulation process or are you doing something different to model those price changes?

[0:29:40.6] JS: Yeah. I think that it's interesting. The primary way that we deal with new markets is to oversupply then. We will usually offer guarantees to shoppers in those new markets so that if they are sitting idle, they're still making a good wage and they're still happy. That's a temporary cost to us of launching a new market. Once we build a sufficient amount of density in the market, the demand becomes more predictable and we no longer have to have those guarantees. That's the typical strategy.

The underlying question of, "Well, how do we model supply and demand and how do we react to it?" It's interesting. I can take you through the history of it. We really started figuring out how many shoppers to put on shift with humans, individual analytics people in our operations organization within each city. They would study what was happening in the market and they would put together by hand schedules that we would then copy into our system and distribute to decide how many shoppers we were going to try to get on shift.

We realized that that wasn't scalable. It's not scalable to have people manually trying to make this happen. Once we start to launch 50, 100, 500 different markets, you can't have an individual doing this for every one of those every day. Not only that, you're not going to learn. You're not going to be able to take the insights from one market and use them to make other markets better.

We started out with a software-based approach to doing this that was relatively simple and we tried to follow what the best people were doing in the field to match this staffing problem. It really came down to look into the past and ask the question, "This time last week, did we have too many shoppers or not enough shoppers? What about this time two weeks ago? Well, let's react to those changes and make incremental adjustments."

Initially, when rolled this out, it made everything worse, because it was too simplistic and it wasn't taking into account all these special cases and idiosyncrasies. That was a really interesting journey where we really needed to have a lot of faith in the fundamental approach that we could automate this and use data and use algorithms and it would get better over time. The business bore with us through that journey, and about six weeks in, things started to get significantly better.

Overtime, we weighed really big changes in managing the efficient frontier of keeping our shoppers utilized but still having availability for our customers. We reached, then, a plateau where we realized what was happening is we were too often just chasing our tail. We were kind of overreacting to what had happened in the past, and because we have different role types, we might find that a role type sits since idle or is busy and we would add more people to do more of that kind of work the next week, but then our system that did fulfillment would just find some way to use those people even if it was using them efficiently.

That was an example where the system that was staffing the shoppers wasn't thinking about what the system that was optimizing the routing of the shoppers were doing and they were kind of working at odds with each other. We have moved to a new approach that does use simulation. We take forecasts for customer behavior into the future and for shopper behavior and the distributions of those outcomes. Then we construct many alternative realities for what next Thursday in San Francisco is going to look like, and we solved for what staffing should be

at every store location for every role type in order to try to optimize the average outcome across all of the simulation routes.

[0:33:31.6] JM: Then does that mean that you have to also be doing testing of the accuracy of your simulation?

[0:33:38.9] JS: Yeah. That can come a couple of different levels. One level is just we have many different forecasts and distributional forecasts going into the simulation, and so we can compute the RMSE or the likelihood of those forecasts as a KPI for the accuracy of the inputs. Then you have the question of, "Well, is the simulation infrastructure itself accurate?" Taking all of the inputs and making the decisions.

I think that's much much more difficult to understand and to directly measure. I think the only way we found to audit that is to show that when you make the change and in reality, that it tends to move in the same to the simulation suggested it would move, and so that's still a fairly manual process.

[SPONSOR MESSAGE]

[0:34:36.4] JM: The Women in Tech Show is a podcast featuring technical interviews with prominent women in technology. The most recent episode features Karen Walker, former VP of Operations at Compaq. She talks about working on the largest supplier of PCs in the 90s. With the Compaq Portable, they took on IBM and became one of the fastest growing companies in American History.

Other great recent episodes have covered subjects like robotics, distributed systems, and ethical questions of artificial intelligence. Check out The Women in Tech Show on iTunes or wherever you get your podcasts.

[INTERVIEW CONTINUED]

[0:35:24.4] JM: There's another area of data science that you explore in this article, which is search and personalization. In my experience, the personalization and search engine on

Instacart is quite good. I think about it as a problem versus something like Netflix or Amazon where I think the average item that I purchase on Amazon is higher price than the average item I purchased on Instacart. On Netflix, I'm going to be devoting two hours to watching a movie or perhaps 13 hours if I'm going to imbed myself in one of these addictive series. This is different than the calculus that a customer on Instacart is going to make, because on Instacart you can make more impulse purchases. For example, I bought a kiwi on Instacart for the first time in a few weeks — Sorry. For the first time in a few years and I found that I really liked it and it was only — It's like a dollar of an experiment. Perhaps the degree to which you can nudge people to make little experiments with their purchasing behavior is greater on Instacart.

I guess I just use that as an example because I'm curious about any examples of surprising customer behavior that you've observed on Instacart and what you're doing around personalization. Is it collaborative filtering or what other methods are you using to recommend the items to people.

[0:36:58.1] JS: Yeah. Maybe I'll start this one from first principles. If you build Instacart, the only thing that you need to begin with is catalog that has all of the items in it. Literally, what our founders did in the very beginning is they just went to the store and they bought one of everything and they took a picture of it and they put the data directly into catalog, so you can begin really really inexpensively with something you create yourself. Then you need to search engine on top of that, and so maybe that's just elastic search and it's a keyword based search engine.

The reason for that is customers are going to come to a site like Instacart and they've already been to Wholefoods. They know the kinds of things that they would buy there. They're just going to want to take the list of things that they have in their head and quickly express them and find those products and then buy those products. That the very first phase.

The second phase is once you've shopped at Instacart two times, you quickly realize, "I don't want to search for everything every time I use Instacart." Let's take the past products that a user has purchased and lets curate those for them to help them purchase the things that they want to buy again in this session.

The very simple thing to do better is to just rank them by maybe the frequency with which you've purchased them, or the recency with which you've purchased those items. That's a really simple buy it again interface.

The third one is the customers is in the middle of buying a specific item and maybe there's something else that a lot of other people will buy when they buy that item. Maybe they're buying fresh mozzarella. Many people who do that are also going to buy basil, or pizza dough, or pizza sauce, maybe eggplant. Let's go and make it easy for them to immediately navigate to the things that most people would buy next.

The fourth one is the customer has bought all of the things that they know they need and they're in a discovery mode. What are the items that they might want to try that they haven't tried before? All four of those problems you can start with something pretty simple, like even the last one for discovery, you can just take the most popular products purchased on Instacart, remove anything that you've bought in the past and show that as an aisle, and it will be a pretty reasonable place to start with discovery.

The question then is, "How do you start to add sophistication and how do you make the customer even happier, be able to move faster, allow them to buy incremental things?" The algorithms are going to vary across all of these approaches.

I think we can talk about some of those specific algorithms if you're interested and specific approaches. Maybe just to draw the analogy to other types of recommendation problems, I think that's a pretty interesting thing to think about. I would say that on Amazon, in general, you're going to buy something once and only once. That's true with books. It's true with many household goods.

I think with groceries, it is quite different because you're going to potentially purchase something over and over again but in different contexts with different frequencies. There's a lot of interesting dynamics there. The value of an incremental purchase on Instacart is actually quite high even though that product might only cost you $5 if you're going to buy a new — A jar of pasta sauce. That's maybe worth hundreds of dollars if you take into account the fact that if you

like that, you might switch to purchasing that and make it a part of your pantry that you'll repurchase over and over and over again.

This discovery problem, while it's not as common, because on a per item basis, it's actually really impactful and it's pretty common on a per visit basis. I think that the cost of ordering something and not liking it is higher. People don't like to have something that — They don't like to throw food away. It's generally bad for the environment. It feels like a waste of resources. Versus if I go on to Netflix and I start watching a series and I immediately know that it's bad, I'm not too upset about that. Now, if I watch the first two or three episodes and then I realize that it's bad, I might be more upset that I wasted all of that time. But, hey, maybe it is on me. I invested the time.

I think the kind of quality and relevance of the recommendations that we make has to be higher. Ultimately, I'd love for us to be able to change how people buy food. Today, if you think about how we find and purchase food, either we try something in the specific setting, and a friend tells us about it. We go to their house and we try it there and we decide we want to try it in the future. That's great, but it's a pretty limited set of opportunities for that inspiration.

You're at the store and maybe you sample something or maybe something just catches your eye and you pick it up and you look at it and you decide, "What the heck? I'm going to try this thing." I think that's a very manipulating and kind of random approach to discovering things. It's manipulating not necessarily in a bad way, but the retail stores, they're going to have end-cap displays, they're going to highlight specific products. They're trying to influence what you're going to purchase, but they're doing it in a way that is the same for everyone. It's not at all personalized to your tastes or preferences.

The brands are very selfishly motivated to try to get you to try their product, and so they're doing everything they can to change the packaging, the words on the packaging, the nutritional information, the size of the packaging in order to try to get you to try that product. They're spending television advertising, all of these other things.

I think Instacart has an opportunity to use data to let people discover food in a very different way. It will be more scalable than trying something at your friends and it will be far more

personalized than running into something in the store. I think that that will be really interesting for us as consumers. It will also change, I think, the nature of the products that are offered. Maybe there will be more fragmentation of products. There'll be more types of pasta sauces and more variety because each one will be able to find their niche more easily and it won't be reliant upon a large T.V. media spend to build a brand. I think it's going to be really interesting to see how all of that changes and how we use data and algorithms to help shape it.

[0:43:28.3] JM: That's a great optimistic viewpoint. I'm with you on that. We've seen this rise of niche brands just because of the internet. Even the internet serving customers in what is still pretty naïve, if you compare what is the current marketplace of the internet relative to where it could be, we're still in a pretty naïve state. If you think about more sophisticated stuff that's coming down the line, you can get a lot more specialized products that are a lot weirder because you can — Those companies can come back to the economies of scale. They can get to the economies of scale faster and get their costs down to a place where customers are going be willing to buy those weird products.

Switching from the optimistic world to the more pessimistic world, then we'll talk about some data engineering stuff. I've talked to a lot of these marketplace businesses about the problems of fraud detection and chargebacks. We did a show with stripe that was very popular about their fraud problems, and every marketplace business has a whole host of fraud problems. It's nothing new and it's probably not going away from the internet anytime soon. It turns out to be a problem that tends to involve so many different layers of the data science pipeline at these marketplace companies because you often need to synthesize data from so many different places in order to get the right signals for who might be a nefarious consumer. Can you talk at all about fraud detection and how the antifraud teams are set up instacart?

[0:45:18.2] JS: Yeah. I think, first, we use a variety of different third-party services in order to score fraudulent transactions. Those are great. They have access to far more transactions from many different types of companies, much greater density on a per customer basis, and so those are very useful to us but they don't take into account any of the idiosyncrasies of our marketplace business. They also aren't going to be particularly relevant on the shopper's side.

I can't talk about specific examples, but you can just imagine that there's hundreds of creative ways of trying to defraud a complex product like Instacart. To be honest, we've been able to handle most of our fraud issues with people looking for specific types of patterns, and we haven't invested a lot yet and using machine learning to make those processes scalable. I think that's something that's on our horizon we're going to have to do purely from like an overhead management perspective. You can't have armies of people reviewing transactions all the time. It doesn't make sense. But it hasn't been as important a problem for us as some of the other places we've used data science.

[0:46:38.5] JM: Okay. I want talk about the data engineering then, and also your broader process of data science across the company. How do you look at data science versus data engineering? Do you have a date engineering infrastructure that all the data science teams have universal access to or are the data science team standing up their own data engineering pipelines?

[0:47:06.4] JS: I think the first place to answer this is how we organize. Say, the engineering organization at Instacart is 120 people. Then out of those, about maybe 45 or 50 relate to data. Out of those about — Let's say, 50 out of those, about 10 are in data engineering and then about 20 do analytics, about 20 do machine learning. The 10 that do data engineering, they're predominantly focused on two things. One; taking the data from PostgreS secondaries and AWS, or from other data stores, like Cassandra, Redis, production data stores, and moving that data into Redshift, which is the primary place that we use for all of our analytics and for a lot of our data science workflows. Any data that is too big to work with in Redhift, they will put into S3 and then we'll have other ways to manipulate the data off of S3. Say, using something like Spark.

The other thing that the data engineering team is doing is to ingest all of the third-party data that we get, data from retailers, data from data providers around products, data from our advertising partners. Many many different types of streams of data coming in that are useful for our business. Along the way, they're taking all of that data either for more production systems or from third parties and they're enriching and structuring it and organizing it, making it easier to access and manipulate.

The analytics team might be called a data science team at Facebook or at Airbnb or other places. They have a pretty sophisticated skillset. They are using R or Python and doing lots of visualizations and analyses. They are building dashboards for public consumption, the kind of canonical representation of KPIs, maybe using something like Tableau. They're setting up A-B tests and analyzing those A-B tests to help make better decisions in the business. I really think about those analysts at Instacart as being data scientists who are focused on decision science.

Then the remaining people are doing machine learning or they're doing operations research science, and we call them here data scientists, but would be called a machine learning engineer at other companies perhaps. They are integrated into product teams working side-by-side with engineers to shift changes to the product that use data and algorithms.

There is a team that's focused on balancing supply and demand and there are machine learning engineers there that are producing the forecasts and they are building control systems for managing our capacity and how we do busy pricing on the site. They're working side-by-side with engineers to ensure that we're logging all of the appropriate data, and then they'll work with the data engineering team to ensure that that's been structured into Redshift in the right way, and they're working side by side with the engineers to change the nature of the product and the consumer experience. Maybe to change the APIs that we've set up to make it easier to ship products in the future, and they're watching the evolution of their system change overtime and making it better and better.

Broadly, I think that's how we work together. The actual data engineering — Like I said, a lot it begins with Redshift, and we'll be pulling that data directly into, say, Python and then manipulating it from there and using other tools from there, or if the data is too big to really store and manipulate effectively in Redshift, we'll put it into S3 and use other tools like Spark to get out of that scale.

[0:50:43.8] JM: Tell me more about the deployment and updating process for your machine learning models.

[0:50:51.8] JS: Yeah. That will vary. Many of our models are — I would say they roughly belong in three different types of camps. One is they're essentially a batch process the that is running

every hour or every minute or every day, and the end result of that is to push data into a table, and that table is a canonical representation of the truth for the predictions or outcomes of that process that any other system can use. Essentially, the database is the API.

We try to avoid making updates and instead making those outputs logs and you can version the table structure, have a — Not version the table structure, but have something in the table structure that allows you to version your changes to whatever it is you're writing to. That can be a way of manipulating or versioning your release.

One of the nice things about systems like that is the database is a fallback, and so you can always rely upon stale data in the database, fallback to that data with whatever production system you have downstream that depends upon it.

The second type of deployment we have is where the algorithm has to be used in memory in the process that is doing some other computation. A typical example for that would be our fulfillment engine. If we're looking at thousand deliveries and a geography and we're trying to optimally route 500 shoppers to do those thousand deliveries, we can't call out to a service or call out to a database to get predictions for any arbitrary combination of a shopper, a delivery, and a set of other deliveries. There's just too many of them, and so we have to compute them in real-time in the process. In that case, the entire application that in this case is doing routing will ship changes to that application and the algorithms will go along with it, and so that has to be managed more carefully.

I think the third part is where you really set something up as a service, and so it might be ranking the products for picking them in the store or it might be something like how we think about making recommendations on the site. In some cases, the latency requirements aren't that low, and so if we've got minutes to sort an order for shopper, we can set up a relatively simple service that has a queue in front of it that makes these predictions and then responds back with them. It's okay if that takes 100 milliseconds or more.

In other cases, if we're making recommendations on the site, it has to happen instantaneously. For those cases, we tend to deploy a pretty simple algorithm in production that's using cached data to make a final decision. Maybe we've learned an embedding about products, and we

know that that embedding represents how similar products are. In real-time, we can compute nearest neighbors in that embedding space for a candidate set of a thousand products and do that for a few milliseconds. That process can be pretty robust and independent to changes in a batch process that's updating those embedding. That's another approach that we use.

[0:54:23.3] JM: You're the VP of data science at Instacart. How much of that job is about hiring and people operations and how much of it is about solving engineering problems?

[0:54:38.2] JS: I probably spend a third of my time on hiring, and maybe half of that time is spent building our brand and the other half of that time is executing on the hiring process and refining it. I spent a lot of my time with candidates. I spend a lot of time helping to think about how we evolved the process that we're using. Then I spent a lot of time on podcasts, like this one, evangelizing in order to try to get people to consider us as an employer. We're growing really rapidly and we really have, at the same time, very high standards for the people that we hire. It means that branding is super important.

I spend another third of my time working with the team, and what that actually means, the individual data scientists here don't report to me. They report into the product team that they work with, which might be managed by a technical lead who could be an engineer by training, or a machine learning engineer by training, or an operations research scientist by training and they're managing an integrated team of skillsets and they've got a very clear mission.

While I don't have people that report directly to me, I still spend a lot of my time meeting with all of the people that are either data scientists or managed data scientists, and I am helping them with individual projects being someone for them to talk to about the ideas, helping mentor them in their career growth. I spend a lot of my time on that.

The remaining third of my time is spent, I would say, on the combination of things. Some of it is engineering and thinking about how to solve those problems, but not a lot of it. I would say more of it is a strategy and how to break down the complex problems we have into components that we think we can use data to solve. In some cases, I like to dig in myself and work with the data and try to build something simple to understand what might be possible in a new domain.

Because of the way we organize around the machine learning engineers, they are only in a team and dedicated to the team. There are going to be, at any point in time, a set of teams here with missions that don't actually have any one dedicated to doing machine learning. I can be a part of the 0 to 1 bridge for those teams helping them to think about, “Well, what data are they going to be capturing now, and how do they think about architecting their system for data collection and use? How do they think about their product strategies such that when we add a machine learning engineer, that person will be able to hit the ground running?”

[0:57:04.7] JM: All right. Just to close off, Instacart recently released an open-source dataset of 3 million orders. Given that you just released that and also the fact that you're looking for engineers, if there is a data scientist out there that's looking to get noticed, what are some problems you would love to see solved against that dataset of orders?

[0:57:29.1] JS: One; we’ve launched a [inaudible 0**:**57**:**30.4] competition, and so that's a place to go and compete against others. The specific challenge we’re asking for there is to predict the next order these users will make and to predict specifically what are the products they will repurchase in that next order? There's a very clear objective metric for that, and you can use the [inaudible 0**:**57**:**54.5] site to compete in that. I think that’s one direction to go in.

Another very interesting direction to go in is what products will they discover in their next order? For you, you recently purchased a kiwi. Could we have predicted that that was something you would more likely to try than an apricot perhaps.

[0:58:17.1] JM: You did.

[0:58:19.5] JS: That’s another really interesting challenge. I think the final one is to try to predict what products are most likely to be purchased together and what products are least likely to be purchased together. An interest way to think about this is maybe 1% milk and 2% milk are really unlikely to be purchased together. They’re essentially substitutes for each other. The example I gave before; mozzarella and tomato sauce are quite likely to be purchased together.

[0:58:50.5] JM: Okay. Jeremy, thank you so much for coming on the show. This has been a really interesting episode for me personally, and I'm a huge fan of the product. It saved me a ton of time and I love it. Keep up the good work.

[0:59:03.6] JS: Well thank you, Jeff. I really appreciated all the great questions and I'm glad to have such a loyal user.

[END OF EPISODE]

[0:59:12.4] JM: Thanks to Symphono for sponsoring Software Engineering Daily. Symphono is a custom engineering shop where senior engineers tackle big tech challenges while learning from each other. Check it out at symphono.com/sedaily.

Thanks again to Symphono for being a sponsor of Software Engineering Daily for almost a year now. Your continued support allows us to deliver this content to the listeners on a regular basis.

[END]