### SEXY SCIENTISTS WRANGLING DATA AND BEGETTING NEW INDUSTRIES

CHRIS WIGGINS (The New York Times)

CAITLIN SMALLWOOD (Netflix) Amy Heineike (Quid)

Jonathan Lenaghan (PlaceIQ)

# DATA SCIENTISTS AT WORK

Roger Ehrenberg (IA Ventures)

> Erin Shellman (Nordstrom)

VICTOR HU (Next Big Sound)

John Foreman (MailChimp)

CLAUDIA PERLICH (Dstillery)

DANIEL TUNKELANG (LinkedIn) KIRA RADINSKY (SalesPredict)

ERIC JONAS (Independent Scientist)

Yann LeCun (Facebook)

ANNA SMITH (Rent the Runway)

JAKE PORWAY (DataKind)

André Karpištšenko (Planet OS)

S E B A S T I A N G U T I E R R E Z foreword by peter norvig (Google)

### Contents

Foreword by Peter Norvig, Googlevii	
About the Author	
Acknowledgments	
Introductionxv	
Chapter I:	Chris Wiggins, The New York Times
Chapter 2:	Caitlin Smallwood, Netflix 19
Chapter 3:	Yann LeCun, Facebook
Chapter 4:	Erin Shellman, Nordstrom
Chapter 5:	Daniel Tunkelang, LinkedIn
Chapter 6:	John Foreman, MailChimp
Chapter 7:	Roger Ehrenberg, IA Ventures
Chapter 8:	Claudia Perlich, Dstillery 151
Chapter 9:	Jonathan Lenaghan, PlacelQ
Chapter 10:	Anna Smith, Rent the Runway
Chapter 11:	André Karpištšenko, Planet OS
Chapter 12:	Amy Heineike, Quid 239
Chapter 13:	Victor Hu, Next Big Sound
Chapter 14:	Kira Radinsky, SalesPredict
Chapter 15: Eric Jonas, Neuroscience Research	
Chapter 16:	Jake Porway, DataKind
Index	

## <u>CHAPTER</u>

## Eric Jonas

### Neuroscience Research

**Eric Jonas** is a research scientist in computational neuroscience and signal processing, pursuing two main lines of investigation. The first focuses on how learning and memory work: how does the brain take in new information, build internal models, and do things with those models (such as save them to the neocortex). His second line of research focuses on building new types of machine learning models using signal processing technology to detect patterns in this kind of data that could not be detected natively. In both endeavors, Jonas has to grapple with two distinct sets of problems. The first set involves finding the right models and techniques to interpret and work with incredibly noisy, high-throughput neural data. The other set of problems involves bridging the gap between the scientists running the experiments and those creating the mathematical models.

While Jonas has been continuously fascinated with the brain since he was seven years old, he has alternated brain science with entrepreneurship. He co-founded two start-ups, both of which were backed by Peter Thiel's venture capital firm, Founders Fund. The first, Navia Systems, developed a new class of probabilistic computers to make inferences under uncertainty. He was CEO of the second, Prior Knowledge, which created a predictive database that learned the deep structure of the data it contained, and then used that knowledge to generate predictions based on probabilistic inference. Just three months after being featured as a TechCrunch Disrupt Finalist, Prior Knowledge was acquired by Salesforce.com's Chief Predictive Scientist. In early 2014, Jonas returned to pure research, an environment he knows well from his career at MIT, where he took four degrees: a BS in electrical engineering and computer science, and a PhD in brain and cognitive sciences. He is currently a postdoctoral fellow in EECS at UC Berkeley.

Jonas is an exemplar of the data scientist who is building the tools of tomorrow. This comes through as he talks about creating instrumentation to record signals from brain cells, his frustration with the slow pace of tool development in the past decade, and his vision of having the right tools to analyze the deluge of brain data that will be available in the future. His desire to build tools that help scientists and non-scientists make sense of their world through data—together with his views on how quantitative and computational the life sciences will become going forward energize his interview.

**Sebastian Gutierrez:** You recently co-founded a startup and then sold it to Salesforce. Tell me about this journey.

**Eric Jonas:** Some friends from graduate school and I started Prior Knowledge [P(K)] in August 2011 with the goal of building developer-accessible machine learning technology. Our vision was building ubiquitous machine learning. We were really inspired by companies like Heroku and Twilio and the way they had democratized access to a lot of what at the time was fairly cutting-edge technology. We felt it was crucial to preserve uncertainty—the ability to say "I don't know the answer"—when putting this technology in the hands of normal people. At the time, that was a really radical thought.

Certainly, the Bayesian statistical community had been doing this for a long time, but a lot of machine learning methods that were out there were just all about giving people an answer. It is important to ask whether you can trust this answer or not, especially since often whether or not you can trust an answer varies greatly on the question you happen to be asking. Machine learning systems will quite reliably tell you that there are two genders, male and female, but they won't always be accurate in predicting which one you're talking to. So that was really the goal.

We did the normal startup thing—we built the product, we demoed and made it to the finals at TechCrunch Disrupt, and we raised funding from venture capital firms. Peter Thiel's group, Founders Fund, backed us. We then were acquired in December of 2012 by Salesforce, where we found a great team of people who very closely agreed with the vision of a ubiquitous predictive platform, where at the end of the day saying "predict" should be as easy as saying "select." I was at Salesforce for over a year, where my team and I continued to work on machine learning technology. I was promoted to Chief Predictive Scientist of Salesforce and oversaw this area.

**Gutierrez:** You have now left Salesforce to do computational neuroscience research. Why neuroscience research?

**Jonas:** The brain is fascinating because it computes. The liver is interesting too—it's this complex metabolic soup, and when it breaks you die, so there are people studying it. However, as a CS person, it's amazing that there's this blob of goo in my head that somehow is doing all this intractable computing.

So these days I spend my time doing two things. First is working with neuroscientists and answering neuroscience questions about how learning and memory work. How does your brain take in new information, build models internally, and then do things such as saving that to the neocortex? Then second thing I spend my time doing is building new types of machine learning models to find patterns in exactly this kind of data. I think that's really where the field is headed, especially as we see a tremendous set of interests in neuroscience data now.

I have been interested in the brain for most of my life. Even before college, I was already fascinated with it. Then when I arrived at MIT, I double-majored in electrical engineering and computer science [EECS] and brain and cognitive sciences [BCS] as an undergraduate. I then did a master's in EECS, and then finished it off with a PhD in BCS. So my background is computer science and brain and cognitive sciences, and I have been studying it for many, many years. This area is something I've been thinking about even when I was working in the industry.

Going from being a startup CEO to working in a large company is a jarring transition, because suddenly you have extra time after work. In this extra time, I finished my PhD, and I continued doing some of the neuroscience research I did in my PhD. I was always very interested with the idea of building machine-learning signal processing technology to find patterns in neuroscience data that we just couldn't find natively. As undergraduates we're taught things like the Fourier transform or principal component analysis, but those are all like fifty years old. Clearly the machine learning technology that's out there has become a lot better at finding those sorts of patterns in data. So lately I've been working with researchers at UCSF, Northwestern, MIT, Harvard, and other academic institutions.

The Obama administration has spearheaded the new BRAIN initiative and there's going to be all this high-throughput neural data around soon, but frankly, not a lot of technology to even look at it—especially technology that's accessible to regular scientists. And I say "regular scientists" partly because the people running the experiments and formulating the questions are often actually quite different from the people with the ten years of computational or mathematical training necessary to build and understand the models. I am trying to carve out a space in between them to be at the impedance matching layer, serving as that buffer because I believe that you can be really productive there.

Gutierrez: Why is this important work now and ten years from now?

**Jonas:** One of the reasons that I left Salesforce was in some sense that my team was there, everything was in good hands, and I was looking for the next big challenge. Aaron Levie, the Box CEO, at one point made the comment, "One of the questions you have to ask yourself when doing a startup is why is now the right time to do the startup?" I think Peter Thiel told him that.

So for me, the question is: Why is right now the right time to be trying to build tools to solve these neuroscience problems? One reason is that the data are going to be available very shortly. About five years ago there was a real question of tracing out all the neurons in neural systems and how they can connect, because no one had the schematic of the brain up until then. Connectome projects are projects that are tackling this goal of creating a comprehensive map of the neural connections of the brain. So people have been building these connectomes of really dense schematics of the systems, led by groups such as Sebastian Seung's group at MIT.

However, the problem with biological circuits is that the schematic that you get out doesn't have a nice little box drawn around parts saying "This is an adder" or "This is a register." No, it's just this dense graph of crap. You can imagine what it's like by trying to figure out how a processor works by just looking at how all the transistors are connected. Obviously, that really limits your understanding. So Konrad Kording<sup>1</sup>, a scientist at Northwestern, and I started trying to build models to discover the structure and patterns in this connectomic state. We have a paper that was just sent out for review on exactly this idea of how—given this high-throughput, ambiguous, noisy, sometimes error-filled data—you actually extract out scientific meaning.

The analogy here to bioinformatics is really strong. It used to be that a biologist was a biologist. And then we had the rise of genomics as a field, and now you have computational genomics as a field. The entire field of bioinformatics is actually a field where people who are biologists just sit at a computer. They don't actually touch a wet lab. It became a real independent field partially because of this transition toward the availability of high-quality, high-throughput data. I think neuroscience is going to see a similar transition. I like to say that neuroscience is generally ten to fifteen years behind the rest of biology because, in many ways, it's a harder problem: there's more ambiguity, and getting the data is much, much harder. So the hope is that right now is the right time to strike.

Scott Linderman<sup>2</sup>, at Harvard, and I are organizing a workshop at the 2014 Computational and Systems Neuroscience [COSYNE] conference on discovering structure in neural data, organized around questions like: How do we find these items? And how do we build algorithms to find patterns in this data? In ten years, the data's going to be there, and if people just keep taking the Fourier transform or keep doing PCA on this data, then we're really going to be screwed. There's just no way you're going to understand these systems.

<sup>&#</sup>x27;http://www.koerding.com/.

<sup>&</sup>lt;sup>2</sup>http://people.seas.harvard.edu/~slinderman/.

It takes sometimes a decade to get these sorts of ideas out into the scientific community, because in sciences there is never a kind of transparent ROI, so it's harder to expect people to be really eager to get onboard. But the hope is that if we can start building the right models to find the right patterns using the right data, then maybe we can start making progress on some of these complicated systems.

**Gutierrez:** When did you start wanting to study the brain? And what motivates your current work?

**Jonas:** When I was a kid, I used to build circuits and taught myself how to program a computer. I remember sitting in a 7<sup>th</sup>-grade class and wondering what the assembly language was for my brain, because clearly there's this computing thing going on there. When I arrived at MIT, I decided to double-major in EECS and BCS sciences partly because I just couldn't give up the engineering side and partly because I had some smart faculty members tell me that if you actually want to do this sort of work, you really need to have the hardcore quantitative background.

Then when I began graduate work at MIT building instrumentation to actually record from the cells, I realized partway through that what we really needed were the tools to understand the data that was being generated. I've come to the slow realization over the past several years that I'm not really the kind of scientist who has the patience to sit there and methodically explore a given system with existing tools. Usually, after a few weeks of working, I get frustrated with current tools, and I'm like—no, let's just build a better tool. And it's this frustration that's part of what is really motivating me to help build better tools.

The other part that drives me is being able to understand the brain faster. The area of the brain that we study is called the hippocampus: it's kind of like the RAM for your brain. There was a postdoc in a lab I worked in who discovered a phenomenon called "reverse replay"<sup>3</sup> basically by staring at the data for five years using traditional methods. I remember thinking I could have written an algorithm five years ago that would have just found that. It had me thinking of how much understanding is sitting on the hard drives of people in various labs just needing these sorts of analytic techniques. So that really continues to motivate me when I look at these data sets. I'm like, "Look, the answer is in here somewhere!" The challenge for us is to actually build the tools to find it. It's no different than trying to build a telescope to find something far away or build a microscope to see something really small.

<sup>&</sup>lt;sup>3</sup>David J. Foster and Matthew A. Wilson, "Reverse replay of behavioural sequences in hippocampal place cells during the awake state," www.nature.com/nature/journal/v440/n7084/abs/nature04587.html.

### Gutierrez: When did you realize you were on the right track?

**Jonas:** The real aha! moment for me was realizing that a lot of other people are doing computational neuroscience. Their work was just being completely ignored by the actual people who were getting papers on the cover of *Nature*. It's an interesting question of why we have this entire community of very smart people putting out all these papers in these low-tier journals, and over here we have the actual experimentalists, and they never seem to communicate. Is it because everyone is insular and territorial? Or is it because what's being produced by the computational neuroscientists isn't of use to the experimentalists? Why, if I'm an experimentalist, would I go and learn all these additional techniques if it's really not going to give me any sort of new capabilities? I saw that happening in 2004, and then I came back to it ten years later and still seeing the same thing is very frustrating. This has to change. Someone just has to do this.

There are more and more people waking up to this realization, and that's part of what motivated us to get a workshop going, because I'm going to be wicked bummed if these data sets come out and no one knows what to do with them. I think everyone—including the funding agencies—is going to be very frustrated. We don't want to find ourselves again in kind of the regime that genomics found itself at the end of the '90s where it's like, "Well, okay, we have this data. What do we do with it? Where's this clinical miracle that everyone was promising?" Part of that, of course, is because biology ended up being fantastically harder than we ever anticipated. But part of it was also that we just weren't really sure of the questions that we were going to ask of these large data sets.

**Gutierrez:** What are the main types of problems that people are working on in computational neuroscience?

**Jonas:** There are two main problems people are working on—one part is the wiring of the brain, and one part is the activity of the brain. The set of wiring questions people have—especially with this connectomics data—are: What are the circuits? How are they organized? And what are the kinds of modules that are connected? The set of activity questions are: What are the repeat patterns of activity? And what do they really mean? The problem with figuring out the answers to these sets of questions is that all of the data is incredibly noisy.

One way of understanding the data issue we face is by imagining that you are in a stadium and you can listen to 500 people at once. Your goal is to figure out what's going on in the game just from listening to those people. There are certainly some things you can tell, like who's winning, who's losing, and that sort of thing. But if you wanted to actually understand things play by play, it's actually much, much more difficult. Translating this example back to the brain, we can think of cells in the brain as being individual people. So there are lots of people trying to build algorithms to tease apart those signals in the brain to figure out what the different conversations are that are going on in a given time and understanding how to relate these conversations back to the larger state of the system.

**Gutierrez:** Are other people tackling other biological systems of the body with similar data and algorithm goals?

**Jonas:** Certainly. A lot of my friends do computational biology work where they're trying to understand how proteins interact and give rise to signaling networks. We used to think that each gene was turned into a single protein—that was the end of the story. Now we know that a gene gets turned into mRNA that then gets more or less sliced and diced and then turned into proteins. This kind of slicing process—called alternative splicing—is the reason why it looks like we only have like 20,000 genes in the human genome, but we have vast amounts more of all these different proteins.

There's incredible diversity in proteins. So lots of people—such as Yarden Katz<sup>4</sup> at MIT—are developing algorithms to take this high-throughput data and understand what's actually happening and what the generative stochastic processes are. If you take the naïve computer science view, every cell is basically a little computer, right? It has this chunk of memory, and DNA is the compressed obfuscated buggy binary that runs inside this complicated set of stochastic differential equations. If we're going to figure out how this all works, then we have to start applying better computational techniques. So, yes, it's very much the case that there are people tackling different biological systems with similar data and algorithm goals.

I make the perhaps slightly controversial statement that I don't think humans are going to be able to understand biology. I think our notion of what it means to understand something is going to have to change. We're going to have to be much more comfortable having a complicated model inside a computer, where we only understand parts of it. In some sense, we were incredibly lucky with physics. The fact that Maxwell's equations are four linear partial differential equations that explain all this behavior is amazing and magical. There's no reason to expect that these gross bags of fluids that we call our bodies, which have evolved over 4 billion years, are going to exhibit this kind of aggressive reductionism.

When you watch Steven Boyd<sup>5</sup> lecture, he keeps referring to the 19th-century mathematics that we all know and how this 19th-century approach to science just doesn't work. So we have to start developing algorithms and we have to be using computational tools to redefine what understanding is. In fact, I think

<sup>&</sup>lt;sup>4</sup>http://www.mit.edu/~yarden/.

<sup>&</sup>lt;sup>5</sup>http://stanford.edu/~boyd/.

industry is actually a bit ahead in this regard. An executive from Target the proverbial or perhaps apocryphal Target predictive application—doesn't necessarily care about the underlying causal process giving rise to someone buying diapers versus beer. What the executive cares about is: "Does this model have predictive power and does it let me then go do something else?" I think you're going to see much more of that trend in the life sciences.

**Gutierrez:** For someone who wants to start working in this area, what material should they be consuming?

**Jonas:** On the computational bio side, there are lots of blogs out there actually. For instance, *Nature* and *Science* both run blogs. However, one of the most useful resources, which I didn't appreciate as an undergraduate, are review articles. A review article is an up-to-date survey of some particular subfield. This is something no one tells you about when you're 21 and struggling through some material. It would be so much easier if someone said, "Guess what? Some poor graduate student out there has written a 15-page article on the state of the art in extreme but accessible detail because it's designed for the wider scientifically literate audience. You should go and read it to understand the material." Both *Nature Reviews Neuroscience* and *Nature Reviews Genetics* are both great sources for having these sort reviews. Those are my two go-to resources.

The other thing that people really don't appreciate is that graduate students, perhaps because of an adherence to sunk cost fallacy, often write really great surveys of the field at the beginning of their PhD thesis. Often, when I want to get into a new field or an adjacent field, I go find a recent grad student's thesis and then read the first chapter. This is because they're going to talk about the review of their field in a way that keeps in mind a general audience for that section. So they're often great pedagogical tools. I have lots of printed-out PhD theses around my living room from random graduate students.

**Gutierrez:** Let's switch back to your startup experience with Prior Knowledge. What was it like to be a guest lecturer at Peter Thiel's startup class at Stanford?

**Jonas:** The whole experience was great and the Stanford students were very enthusiastic. To the class I guest lectured specifically, the experience was very fun, as the class was very much focused on AI and not data analytics. What made it a very interesting experience was that I was a guest lecturer with Scott Brown, the CEO of Vicarious, and Bob McGrew, the director of engineering of Palantir. These two companies are pretty much on opposite sides of the space of possible data/AI companies.

Palantir is a very successful company that does nothing Al-related in the slightest. In fact, they tend to be very Al-agnostic. They build tools to let human analysts do better data analysis. On the other hand, you have Vicarious,

who says that in ten years we're going to solve general artificial intelligence. I don't think a lot of people in that class recognized the degree to which the things that Vicarious was saying are the things people have been saying for the last twenty years, as it's very easy to overhype and overpromise on these sorts of technologies. Then you had us, Prior Knowledge, who sat right in the middle of the spectrum.

Personally, I think that the best thing that happened in the class, which I'm not sure it ever made it into the notes, was that Peter asked, "Do you ever worry about your technology being used for evil?" The people at Palantir said, "Well, yes, in fact, we have an entire legal staff dedicated to making sure that that doesn't happen." I think that's the best spin I've ever heard on a compliance department. Having a bunch of lawyers because you like government contracts is very different from saying no, we employ John Connor.

Having Peter teach the startup class was great because Peter's extremely smart and very willing to make long-term technology bets. He's excelled in investing in science and state-of-the-art engineering as well as pure-on commercial consumer investments, like Facebook. There aren't very many VCs who make those kinds of bets, and Founders Fund has always been very much willing to take the risk here for deeper science plays. Of course, I'm obviously very grateful for that. Finally, it was fascinating to watch the class reach a certain degree of infamy due to Blake Masters taking extensive notes and posting them online.

Gutierrez: Could you turn your neuroscience research into a startup?

**Jonas:** I wish it were possible to make a billion dollars developing technology that would make neuroscience better. I wish it were possible to make more money building tools for science because you could hire an engineering staff and a support staff to help speed up progress. However, it's just not sustainable. The reality of the current situation is that science tool companies are a shitty business. If there's a very clear and useful clinical application for a product, then often it's more viable. Illumina, which makes systems that analyze genetic variations and biological functions, is an example of this type of company.

Unfortunately, at this time, I don't think there's a way to build a startup around neuroscience tools. Even if I could accomplish something as dramatic as building a tool to record from a billion neurons, which I think everyone recognizes would be one of the biggest breakthroughs in neuroscience in the past 50 years, the total addressable market for this tool would be around 20 labs, each of which has a couple of million dollars in funding a year from the NIH if they're lucky. It's not a market that can sustain large companies. I wish it was—but no, alas.

### Gutierrez: What did your typical day look like at Prior Knowledge?

**Jonas:** There was no such thing as a typical day at P(K). As CEO, it meant that I worked with customers, managed the team, owned the product vision, and dealt with investors. The responsibilities and day-to-day work in each of these areas changed very rapidly as we went from idea to execution to being bought all in the span of eighteen months. Each day I would work on all these areas and, before I knew it, my entire day would be gone. Sometimes I would feel like I had done nothing. After all, as an MIT person, I was used to thinking about my day in terms of how many equations I had written and how many lines of code I had committed, so it was very tough. Some weekends I'd go to the office to code because I missed doing the actual technical work so much.

One of my extremely smart friends, Alex Jacobson, who was the entrepreneur in residence at Founders Fund, who made the original introduction for us, at one point told me, "You're never going to get to do new technical work again because now everyone knows that you can manage technical people, and in many ways that's a far more valuable skill. Most people don't know how to evaluate technical people or how to convince them to do anything. So the fact that you can manage a group of 24-year-olds, and get them to do something real is all that anyone's going to want you to do". This was somewhat disheartening to me because I don't want to be pigeonholed away from doing technical work.

The only way that I survived the CEO experience was that my team was amazing. My cofounder Beau Cronin handled the product side. My cofounder Vimal Bhalodia handled all the COO-type work, and planning, and execution. My cofounder Max Gasner handled all of the transactional work and was the person out on the road fundraising with me. Cofounder Cap Petschulat, my best friend from high school in Idaho, was our lead architect. So it was just a really great group of people that kind of helped me through that. The first two hires we had—Jonathan Glidden, a Berkeley undergrad, and Fritz Obermeyer, a PhD from CMU—were also fantastic. It was my first experience actually managing people who were obviously much smarter than me. It was always great to come back at the end of the day to find out how much technical progress they had made.

Gutierrez: What did your typical day look like at Salesforce?

**Jonas:** At Salesforce, a lot of our challenges revolved around overall integration, integration with the existing systems, and talking to customers. This made life much less hectic. The challenges inside of any large company are very different from startups, as the incentive structure is so different. In a startup you can move very quickly. The Facebook mantra of "move fast and break things" works at startups because when you break things, no one cares—because generally you have four customers, whom you likely met through a friend of a friend, or a friend of a VC. So if you break something, you can call up the CEO and say sorry. In a big company, you can't do that, so it becomes more of navigating those waters and understanding how to play that game. Though a different structure, it was still an important challenge for the team and me. I think we pulled it off successfully, as evidenced by the fact that most of the team is still around even after the one-year mark.

Gutierrez: What does your typical day look like now that you do research?

**Jonas:** I wake up most mornings and go to Philz Coffee, where I sit and code or read papers for a couple of hours. These days I'm trying even harder than ever to aggressively defend large blocks of time. I've discovered that if I want to get anything technical done, I need a four-hour contiguous block. Otherwise, work just doesn't get done. Even just one 15-minute phone call can totally mess that up.

It's been a bit of a struggle being an independent researcher without an admin or a team to work with. I come home and—like, wow—the accounting paperwork isn't all magically done for me. So the big challenge for me right now is trying to figure out how to be technically productive while still getting admin tasks done. Similarly, travel also gets in the way of being technically productive. For instance, I'm going back to MIT next week. The week after that there's the workshop that we're running at the COSYNE conference. Then two weeks after that there's a machine learning conference in Iceland and then one in Copenhagen. It's just so easy to have all these little things kind of eat away at your time as you're trying to be technically productive. I think that's often why in academia, graduate students and postdocs do all the real work. There's no way that a principal investigator, given all of their responsibilities, could possibly ever sit down and do real technical work.

Gutierrez: How did you view and measure success as a PhD student?

**Jonas:** "Poorly" is the most honest answer I can give. I've yet to meet a graduate student who doesn't make the mistake of confusing inputs with outputs, especially in the experimental sciences. We think that because we're there at nine in the morning and stay until midnight, where we sit in front of a computer all day, and maybe even actually coding instead of hitting reload on Reddit or Hacker News, that we're being productive. We work very hard and think that's successful. We say things like, "I haven't slept in 48 hours!" And everyone's like, "Ooooh." There's this tendency to think that's the metric that matters as opposed to the number of papers you write or how close are you to actually graduating. So in grad school, I didn't have the best view or measure of success.

Gutierrez: How did you view and measure success as a startup CEO?

**Jonas:** Fortunately, in industry, especially in startup world, people won't let you get away with that view or measure of success. If you have a smart group

of people that you're working with, like I did, all the traditional startup vanity metrics are ignored in favor of the question of "Are dollars coming in?" That measure of success is all that matters, especially when you're VC-backed, because the dollars-going-out number is typically very large. So for us at P(K), the primary focus was: How many customers, doing real things with the system, are paying us money? Of course, when we were very early on in the process, we took the wrong view of measuring success by focusing on much more technically questions, like: What's our uptime look like? How long does it take to process a job? What's our predictive accuracy? and similar questions. However, we very quickly realized that no one gives a damn. What really matters is who's actually using and paying for it.

For example, there's all this talk about predictive analytics. Kaggle became this big thing because everyone seems to think that predictive accuracy matters. In reality, almost no one actually cares about predictive accuracy because in almost all the cases, their starting point is nothing. If you have something that gets them 80 percent of their way there, it's an infinite improvement and they will be so happy. The number of industries where the difference between 85 versus 90 percent accuracy is the rate-limiting factor is very small.

Sometime in the future, after everyone has adopted these sorts of technologies, the predictive accuracy will start to matter, but at this point it doesn't matter as much as people think it does. Sure there are some areas like quantitative hedge funds that are fighting tooth and nail over that last epsilon, but most people are not in that position. So it really comes back to the question of "What value are we providing?"

**Gutierrez:** How do you view and measure success now that you've transitioned back to research?

**Jonas:** As I've transition back to research, it's been very important for me to keep the startup experience view and measurement of success at the top of my mind. Our first employee who we hired out of Berkeley, Jonathan Glidden, wants to go back to graduate school. I'm really excited for him because I don't think he's going to make any of these cognitive errors in graduate school, having now gone through this process. He really understands, in some sense, how to ship. But it's hard because a lot of academia tends to value novelty in a way that I think is actually very counterproductive.

One of the things I struggle with is that comparative advantages are actually complicated, because they intersect with your utility curves. There are problem domains where I know that the models that I have would actually be transformative. But I feel that that's not the most important domain for me to be working on right now, or someone else will do that, or that I can come back later. And so it's hard when you've been so trained by graduate school to think that what really matters are papers. So it's novel to recognize that, yes, some papers aren't actually worth your time to write. So a lot of it right now is how can we uniquely have an impact? Where can we have these force-multiplying effects? Even for some of the things I care about, are papers the right metric?

Roughly nine months ago, I decided that what I really needed was a board for myself. So I have this group of four friends that I send weekly status reports to who monitor my progress and help me set goals. Sure it's a little Type A, but it's tremendously helpful. One of the things that we've talked about is that there are a lot of technologies that technically exist. You can read *Scientific American*, or *Newsweek*, or other publications, and you think these technologies are out there. Sadly, you can never really buy one off of the shelf. This problem comes from the fact that so much of what academia is oriented toward doing is getting a prototype that basically only works once and then getting that result out there by writing a paper and then moving on. There's a real gulf between that and actually having impact in people's lives.

I think the rise of university press offices has actually been a double-edged sword, because my mom reads an article in *Tech Review* or from the MIT news office about research and it always says, "And this may lead to something for cancer" or some other impactful result. I then have to be like, "Mom, when they say that, what they really mean is that they had to put that in there for the grants. In reality, this protein may lead to curing cancer in the same way that you living in this house may lead to you being very, very rich, through homeownership, but it's probably not going to happen."

Since I returned to science, I've really started trying to track the kind of weasel words that scientists use. Of course, no one's trying to be disingenuous; it's just that part of our jargon includes phrases like "may show a relationship to," or "may share a common cause with," or "strongly suggests that," and none of these are definitive. The general public interprets these statements as being far more certain than we, the scientists, intend. Are we actually learning anything? No. We're waving our hands around a lot. The real metric should be: "Do I know something now that I didn't know yesterday?" And a lot of times for a lot of results, the answer is, "Slightly." So for me personally, the question is still out as to whether or not that's rewarding enough to keep waking up in the morning. We'll see.

Gutierrez: How do you choose what to study and analyze?

**Jonas:** My list of goals is to learn everything, be able to build anything, save everyone, and have fun doing it. That's a nice simple list. It's nice to have application domains that I'm actually passionate about or questions that I'm really curious about. On the neuroscience side, I actually do care a great deal about how the system works. And so—while there are other application domains in epidemiology and genomics and other domains that are also very interesting—when times get tough, I'm not going to drag myself out of bed for those problems. So a lot of it is kind of intrinsic interest, and then part of it is also clinical impact.

I've started working with clinical schizophrenia data partly because there are people out there suffering from this disease and, in some sense, that's absurd. The fact that disease is a "thing" is absurd. We're mechanistic, we should be able to fix ourselves, and the world I want to exist in 20 years will solve that. So going back to the goals, it's important to ask if we are closer to this world or not?

Startup culture teaches you to be like Steve Jobs, in that you're right, everyone else is wrong, and your vision will power through. Academic culture teaches you that you're dumb and that you're probably wrong because most things never work, nature is very hard, and the best you can hope for is working on interesting problems and making a tiny bit of progress. Just doing that is seen as an amazing career. So the question is: How do you reconcile these kinds of things? I don't know, I struggle a lot with reconciling these two cultures in myself.

Some of the best scientists out there are the ones who are extremely opportunistic—when they see novel ideas and how things suddenly fit together, they drop everything else and work on that for a while. Others are consumed by a single, all-encompassing vision and aggressively pursue that forever. The downside, in many ways, is that the academic funding system really rewards the former, in that if you have three *Nature* papers with no clear coherent tie to them, it doesn't matter. "You have three *Nature* papers—congratulations, Professor!"Whereas if you've been working on the same problem for 10 years, but only making incremental progress—"Well, sorry, you're not getting tenure at a place like MIT. I really hope you enjoyed working on the problem."

That's one of the reasons why I'm not necessarily excited to go back into academia, because the incentive structures are so confused around this issue.

Gutierrez: What kind of tools have you used and do you use now?

**Jonas:** From a technical point of view, I'm almost entirely a Python and C++ person. I do C++ for the heavy numerics and Python for basically everything else. It's an extremely productive environment. It's nice because, as someone with computer science training, I can do complicated things in Python. It's not like MATLAB, where you have to jump through a million different hoops. And I can drop down in C++ when I need it for speed.

Mathematically, a lot of what I work on is Bayesian models using Markov chain Monte Carlo to try and do inference. I really like that universe because the world is so simple when you think about it probabilistically. You can think of a stochastic process that you can condition on your data and do the inference. That's great! It means the set of math I have to know is actually shockingly small, especially because often the problems that I'm working on don't have data from a billion neurons yet—we have data from 100. And so I'd much rather spend my time building complicated correct models and then, when the data gets larger, figure out how to simplify those systems; rather than start out with something simple and later rework the models when the data set size grows. That's really been my tack thus far academically and in the startup world. I think everyone at P(K) shared that bias as well.

**Gutierrez:** As you're building these tools for yourself, is there any chance you'll go and build another tool company?

**Jonas:** I've thought of it. I actually just received funding from DARPA to fund some of the construction of these tools and to hire some engineers to specifically build them, which is kind of exciting. But I don't know. The problem is that it's really hard to do a tools company, especially an open-source tools company. Tools companies don't fly anymore, especially for these sorts of things. Of course, the entire big data ecosystem is all built on top of these tools.

But how large is the space for companies like Cloudera? I'm not really sure. I think most people who buy things from Cloudera are buying them because Mike Olson did a good job selling them something and there's a CIO someplace whose CMO has turned to him and said, "I want to do what Target does. We need a Hadoop." So they call up Cloudera and they're like, "I'm buying two Hadoops." And then they're like, "Great. We just bought two Hadoops. Now we have some Hadoops." And you're just like, "There's no value being created here." I think that's how a lot of the most successful tool companies have managed to get off the ground.

The other thing I really learned at P(K) is that the right thing to do is to not build a tool company but to build a consultancy based on the tools. Identify the company, identify the market, and build a consultancy. Later, if that works, you can then pivot to being a tool company. If you're selling to the enterprise, which you should as they're the only people with money, you're never going to make headway without a substantial services arm. So start with the services arm first because it's quick revenue, it's nondilutive, and it's great. If that works and gets traction, then you can go down the standard Silicon Valley VC trajectory.

I think there are a lot of data-related startups right now that, had they started by doing that, would be in much better shape. What you don't want to be doing is burning through VC money just to figure out who your customer is going to be. It's a painful truth and it's hard work, but it's much better to approach building a company that way. We had meetings at P(K) around this question of turning to consulting or continuing to build the platform even after we had taken VC money. Ultimately we decided to shoot for the moon and it worked out very well for us. However, having gone through that, I now think the right thing to do is to start out with a consultancy.

### Gutierrez: How do you know you're solving the right problem?

**Jonas:** It really depends on what you're going to call the "right problem." For me there are two parts to that question: "Is the problem right in some sort of global sense?" And "Is the problem going to be one that I'm going to be willing to see through?" That latter question is just as challenging as the former. So a big part of it is: "Am I intrinsically interested in the answer here, and do I think other people are going to be interested in it as well?" Then I can ask, "Am I having fun, and is this the best use of my talents?" I have a clock that shows the estimated number of days until I'm 80, which is a reasonable life expectancy. It helps to remind me that each day actually really matters.

So on the neuro side, I talk to my neuroscience friends to try really hard to make sure I'm not just doing math-wanking. Sometimes I still veer in that direction because math is fun and solving technical problems is fun. Sometimes you veer in that direction and end up finding another path back. However, science is hard, most stuff doesn't work, and you have to be willing to stare at shitty, ambiguous results and be like, "I'm going to keep doing this for another 8 hours today, anyway."

Some of the research projects I've started lately, I've worked halfway through them and realized that I'm not that into them, so I stop. People are very understanding about that sort of thing. Everyone likes to talk about how if you're not failing some fraction of the time, you're not trying hard enough, so at the very least, I console myself by thinking, "Well, I tried this thing and it didn't work, and that's okay." Megan McArdle has a new book out about the role of failure and success, and she makes the same argument that, in some sense, it's figuring out how to both fail and recover is really crucial to all of this innovation stuff that we do.<sup>6</sup>

Specifically to the question of "How do I know I'm currently solving the right kinds of problem right now?" is the good response for our workshop, which is a good indicator that people give a damn. Private and public funding agencies are getting really excited about funding this type of work, which also suggests I'm on the right track. On the other hand, the universe is very fad-driven. In 2008, I might have thought hacking on Hadoop was going to be this really big thing, and now I'm like, "Well, honestly, what's the real value there when most people probably could have done their data analytics on top of PostgreSQL?" You don't really know if what you're working on is the right track.

**Gutierrez:** When you're thinking and solving problems, do you approach it modeling first or do you approach it data first?

<sup>&</sup>lt;sup>6</sup>Megan McArdle, The Up Side of Down: Why Failing Well Is the Key to Success (Viking, 2014).

**Jonas:** If I'm not familiar with data, then I generally don't even start. I recently met Winfried Denk, who invented two-photon microscopy and is a very smart applied-physicist guy who's received many, many awards. His comment to me in this area was that the number-one thing you have to be able to do is actually know what questions to ask. And so I try not to get involved in projects where I don't know what the right questions are. And then generally, if I know the questions, I understand the data well enough to then start thinking about the modeling. The nice thing about modeling is that you can fairly rapidly turn around and try a bunch of different things, then it's very easy to be led astray.

Gutierrez: How do you look at the data?

**Jonas:** Matplotlib in Python. I make a bunch of initial plots and then play around with the data. A lot of the data I work with looks very different from the kinds of data that show up more on the industry side of things. No one in science really uses a relational database, because we either have time series, or graphs, or images, or all these weird things. Rarely do we get relational facts. So I don't end up using SQL that much. It's much more about writing a bunch of custom scripts to parse through 100 gigabytes of time-series data and look at different spectral bands or something similar.

Gutierrez: What do you look for in other people's work?

**Jonas:** On the research side, my answer is different from many of the people I work with and other people in the field. One of my colleagues told me, that I read more papers than anyone they know. I don't actually really read most of the papers. I read the title and the abstract, look at the figures, and then move on. For example, when I evaluate machine learning papers, what I am looking to find out is whether the technique worked or not. This is something that the world needs to know—most papers don't actually tell you whether the thing worked. It's really infuriating because most papers will show five dataset examples and then show that they're slightly better on two different metrics when comparing against something from 20 years ago. In academia, it's fine. In industry, it's infuriating, because you need to know what actually works and what doesn't.

So a lot of what I look for are: "Do I think that their approach was valid? Do I know them?" The degree to which I will read papers from people I know and trust far is far higher than those whom I don't know. People complain that it's hard for new people to break into fields. Well, that's partly because at any given time, 99 percent of the time people are all new and they're cranks. So a lot of it is: "Do I find the structure of this model to be interesting? Do I think they did inference properly? Did they ask the basic questions? Do I believe those results? Is the answer something that I would have believed before

I read the paper?" If the answer is yes, then I'm more likely to have believed it. "Am I surprised? What is the entropy on the result?" The more surprising results, the more thorough I want to be when evaluating the work.

On the industry side, I think that the ability to do software engineering is something that is very important, but isn't really taught. You don't actually learn it as a computer science undergraduate, and you certainly don't learn it as a graduate student. So for me it's very important that someone has learned it somehow—either by themselves or from someone else. I basically can't hire people who don't know Git.

There's a universe where companies at a certain scale can afford to hire people who have tool deficits. For almost anything I've been associated with, that's not really the case, which is unfortunate because it means that you leave out some very smart people. You want people to be productive on day 10, not on day 100. I also really think that the Bayesian approach to machine learning has incredible legs. I think that it encourages a certain kind of precise thinking. If you're not doing it, it's very easy to confuse yourself and lead yourself astray. So I generally look for people who are at least familiar with that part of the universe.

**Gutierrez:** What have you've changed your mind about with regard to using data?

**Jonas:** I started dating my best friend in undergrad around 2001. We were together for nine years and then broke up. So I found myself single at 29, and realizing that I was going to have to learn how to date. I decided that if I'm going to start dating, I should keep data on it. So I create a dating spread-sheet, and then went on something like 100 first dates and ended up with this massive chunk of data. Coming out of MIT, you think you're stupid because everyone else was smarter than you and you think all these sorts of negative things about yourself. However, after looking at the data, it turned out that I'm actually kind of a catch—which was a great thing to change my mind about.

I found that if I could get to the second date, then generally I could get to the fifth date with someone. There's this initial evaluation process, but I was generally pretty good at that. So I was actually in a much better position dating-wise than I had ever thought. You wake up and 30s are around the corner, and your friends are marrying and you're single, you're like, "Oh, my God—I missed the boat!" With the dating spreadsheet it was easy to see that no, actually the data says the opposite. This was this phenomenal. Psychologically it's so easy to fall into these kinds of anchoring effects where you always remember the last date that went poorly. You don't think of the previous 10 out of 15 that went well. If I hadn't been keeping track of that data, I think it my confidence would have taken a big hit, which interplays with all these other things in my life. So I'm dating a great person now and I don't think that would have been possible had I not had this epiphany that the data shows that some people do in fact like me. The dating spreadsheet has become somewhat of a joke now, but it actually really helped. Everyone talks about "quantified self" and everyone wants to track themselves. But no one's writing down a lot of the interpersonal interactions that actually matter—who cares about how many steps you took last week, who did you kiss? So I think the dating spreadsheet is a good argument for the quantified-self approach in this kind of data.

**Gutierrez:** What does the future of data science or computational neurobiology look like?

**Jonas:** I know that everyone wants to talk about big data. It's now this phrase that has somehow entered the lexicon in a horrible sort of way. And also that being a data scientist is "the sexiest job of the 21st century"—admittedly said by a data scientist in an article he wrote, so not really objective. Sure it was in *Harvard Business Review…*but come on! I actually think a lot of the future is in small data. Or what my friends at Bitsight call "Grande Data", as in the Starbucks cup sizes—it's neither Tall (short) nor Venti (large); it's Grande (medium). The amount of things you can discover out of a gig of data are often far more interesting than the things you can discover out of a terabyte of data, because with a gig of data, you can ask more interesting questions. You can build more interesting models. You can understand more about what's going on.

On one hand, there's the Peter Norvig philosophy that with enough data you can use simple models, which is true if you are Facebook, Google, Walmart, or companies of that size. Otherwise, most companies have a thousand, or ten thousand, or even a million customers, which is nowhere near what you actually need for Norvig's philosophy. Most people who are buying and using technologies like Hadoop are using it as a recording engine, where they comb through all this data, then stick it in an RDBMS and actually do their data analysis in R and SQL. I think that as the big data hype cycle crests, we're going to see more and more people recognizing that what they really want to be doing is asking interesting questions of smaller data sets.

On the computational neuroscience side, the data are coming and the data sets will be getting bigger over the next ten years. Right now, if we're not building the right models, I think we're going to be a little bit screwed in ten years. What are we going to do—linear regressions? I've talked to very smart, famous machine learning people at Google, and I asked them, "What do you do all day?" And they replied, "Well, you know, we do feature engineering and then run linear regressions on our largest data." "But you wrote a book!" I thought. "What's going on?"

I hate the phrase "predictive analytics." If you think that the world is all about predictive analytics, then the entire universe is in some sense solved and uninteresting. If you care about what's going on inside the box and if you want technology to let you see new things, then that's kind of a green field right now. The data microscopes project that DARPA is funding is all about building exactly these sorts of tools to let you see things in data that you couldn't see before. A great deal of data analytics startups these days are like data visualization technologies in that it's great when you can think of the questions to ask, and then ask and visualize and plot the conditionals and these sorts of things. However, for a great deal of data regimes, we're far beyond that.

When you start looking at these kinds of Grande Data problems, and it's too much to visualize, and it's not enough data to do something Google-scale, and you don't even have the resources to do Google-scale—well, what do you do now? More linear regression? I really think that's going to end up being the future, especially for the sciences. It's inevitable. And I think you're going to eventually see these sorts of data microscope techniques being taught to undergraduates. We're going to see this real transition.

Gutierrez: What data sets would you love to see developed?

**Jonas:** There are two data sets. One is that I would like to have all of the connections between every cell in the brain. This would be the spatial locations and connections of every cell in the brain. Two is that I want the time series from every neuron in the hippocampus. We have to start building these sorts of high-throughput neural data sets. I'm not necessarily content with these being in animal models, as they're going to be for a while, but we'll get there eventually because the systems are there. The data is there; it's just currently inaccessible. We have to change that. Fortunately, there's more and more interest. Somewhere in my brain, there's some glob of goo that knows my phone number. We have no idea what that is, what that looks like, how it even works, and that's ridiculous because it's in there. There's some circuit in there. I want to understand that, and I want the data to exist to help me understand that.

**Gutierrez:** Early in your PhD you built a device to measure this data. Have you or others thought of pursuing this?

**Jonas:** It is something I've thought of pursuing. The question is partly one of comparative advantage. I think that this space is large enough. What I really want to get to is—if you can record all those neurons, how do you then go use that technology to make \$10 billion? Because then we can let the capitalist innovation machine do its work. However, what we're currently on right now are rats. No one really wants to read the mind of a rat. Even pharma companies have no interest in reading the mind of a rat because a rat is basically too big of an animal to do large-scale experiments with. They like mice because they're small, and even then, mice studies are horribly expensive from pharma's perspective. We'll get there eventually. Hopefully, the hype doesn't kill it first. But, I can imagine going back on the instrumentation side. I've spent a little bit amount of time over to Berkeley's AMPLab, working with some people there that do compressed sensing, trying to feel out these areas. We'll see. Hopefully there are lots of opportunities there in the future.

**Gutierrez:** If you could give advice to someone starting out, what would you say?

**Jonas:** I think that if you're an undergraduate today and you don't know how to program, you're basically screwed. If you're doing anything remotely technical, especially on the biology side, you have to learn how to program. That's inevitable. For my liberal arts major friends, maybe it's less mission-critical for their career trajectory. The ability to work with data really ends up being sort of crucial, and to that end you have to know how to talk the language of the computer.

To people starting graduate school in sciences I would suggest reading Derek Lowe's blog, "In the Pipeline."<sup>7</sup> Derek's a medicinal chemist who's been blogging for years, and he's an amazing person of insight. He talks about working medicine, pharma, and life sciences for thirty years and how he's never had a drug he's worked on actually makes it into a patient. He talks about how that's common because the median success is zero. He also talks about how the purpose of graduate school is to get out of graduate school, as there's nothing else that matters. I think that's really true. I would also encourage people to go into the more quantitative programs because it's so much easier to later become less quantitative.

To academics, I would give the advice that startups are not a source of funding. It's surprising the large number of graduate students who approach me with something that basically has no market and say, "Well, I think I could probably get VCs to give me funding for this." My response is always, "You don't understand the game being playing here. VCs are going to want you to focus on things that you don't want to focus on, and it's not going to work." And even when you have VCs who are extremely supportive, like we did, you will eventually realize that these aren't grants they're giving you. They want you to turn around and give them a billion dollars back. If that's not your intent when you take the money, then that's fine, but you need to tell them that upfront when you start.

This whole using-VC-to-fund-science is a difficult and a duplicitous thing to do, and it's very easy for graduate students to convince themselves otherwise. It's easy to say, "I'm going to build this tools company." And you're like, "Well, no. Let's apply the same rigor to this process that I apply to my other science. How many companies are there like this? How have they been successful? What have their real trajectories been?"

It's one of the reasons I think things like pursuing nondilutive capital like DARPA or early consulting gigs for any of these hard tech problems is actually the right way to go. If you look at a lot of the really successful companies

<sup>&</sup>lt;sup>7</sup>http://www.pipeline.corante.com/.

that have been built this way—even, to an extent, companies like Cloudera, where they had already built a lot of the hard technology when Yahoo! spun out Hadoop. Someone has to be footing that bill, and VCs do not have the risk appetite or patience to let you try something for three years. They'll let you try it for a year, and they'll probably still keep funding you over the next two years, but what's going to happen is you'll bleed through your cap table, and you're going to wake up and maybe finally have a success and realize you've sold 90 percent of the company. Oops.

Gutierrez: What do you look for when you're hiring people?

**Jonas:** It depends a lot on the role. The first thing to find out is if they are an asshole. Life's too short to work with assholes. At MIT we used to talk about how it would take freshman a while to de-frosh. They would come in thinking they were the smartest person ever because they grew up being the smartest person they knew. Then they get out into the real world and realize that no, they're not. They have to have that arrogance beaten out of them. There are some people who never lose that. There are some people who very much think that being smart is an excuse to not have interpersonal skills. And the world is just too collaborative for that to work anymore.

My cofounder Beau Cronin made the comment the other day when I was talking about an academic who I was working with who was a little bit difficult, and Beau said, "The nice thing about doing a startup is you get to say, 'Nope! No! No! Do not talk!" In academia, because of the way the incentive structures are often set up, that's not as much the case, so you might end up working with difficult people.

At P(K), we evaluated a lot of really smart people that just weren't a good fit. Startups spend too much time talking about culture these days, and often culture is a euphemism for "not exactly like me." Which is a terrible way to look at culture. What really does matter, and how we looked for fit was by asking ourselves the following questions about them: Are they excited about the same technical problems as we are? Are they excited about being collaborative? Do they like sharing their successes and failures? Do they possess some degree of appropriate humility and understand why it's important? Finding the right person with the right fit is hard, especially in the machine learning and data space. But that's the most important thing—making sure they are a good fit.

Obviously, a strong math background is necessary. If I have to explain probability to someone, it's going to be a really hard slog for everyone involved. I would rather take someone in the top 20 percent of quantitative skills who also is a great software engineer over someone in the top 5 percent who doesn't know how to code. The quantitative finance model really popularized the notion that raw cognitive talent is all that matters. This is the D. E. Shaw and Renaissance Technologies model of "We're going to take people who have been doing algebraic topology for a long time, and we're going to then teach them quantitative finance, and this is going to be a good scheme." In some sense, it obviously worked out very well for them, but especially on the data side, data analysis is so much messier than actual math. I have friends who work on these topology-based approaches, and I'm like, "You realize these manifolds totally evaporate when you actually throw noise into the system. How do you think this is really going to play out here?" So I would much rather someone be computationally skilled. I'm willing to trade off what their Putnam score was for how many open source GitHub projects they've committed to in the past.

I'm also very skeptical of this notion where a data scientist comes in without the domain knowledge and starts producing work. I think you actually need to care about the domain. I do think that a lot of the interesting problems, especially those I'm interested in, necessitate that you have already been doing work in the area for a while. So rarely do I find myself hiring someone who just has data science experience.

One of the things I've seen a lot in the neuroscience community—or in industry even—is that you get people who really like math showing up and being like, "How can we apply this thing I have to your problem?" They just want to do the math and they don't really care about the application. But if you don't actually care about the underlying problem, then you're not going to be willing to make the compromises necessary to understand how to guide your own work. In academics or industry, if you're not actually speaking in a language that your customers understand, then you will have a nice time talking, but no one will really listen to you.

**Gutierrez:** What is something you know that you think people will be wowed by five years from now?

**Jonas:** Either that Bayesian nonparametric models let you see things in data that you didn't know were there or that Markov chain Monte Carlo actually scales to data at a size you care about. Being properly probabilistic solves so many of the problems we face in machine learning, like overfitting and complicated transform issues that I still don't fully understand. There's an entire set of machine learning work that starts with the predicate that your data are a fully observed, real-valued matrix where the matrix is  $\mathbb{R}^{n \times m}$ . From my point of view, problems almost never look like that. This predicate forces you to do all this stuff with your data to try and force it to look like that. And then, once you have it in that form, you do a bunch of linear regressions. I'm of the opinion that it's better to do slightly more sophisticated modeling here by modeling the likelihood function and taking a generative approach. I think that in five years, that's going to be the way most people do things. I think it's inevitable. However, I think it's going to be a lot of work to get there.

The other thing I think people are going to be really surprised by is how much of a quantitative and computational science the life sciences will become. In some sense, everyone's always saying this—it's kind of a trope at this point, but it's only going to become increasingly true. Every time we look back, we're much better than we were five years ago. We always still hate ourselves though, because we're never where we want to be—but I think we'll get there.

**Gutierrez:** What is something someone starting out should try to understand deeply?

**Jonas:** They should understand probability theory forwards and backwards. I'm at the point now where everything else I learn, I then map back into probability theory. It's great because it provides this amazing, deep, rich basis set along which I can project everything else out there. There's a book by E. T. Jaynes called *Probability Theory: The Logic of Science*, and it's our bible.<sup>8</sup> We really buy it in some sense. The reason I like the probabilistic generative approach is you have these two orthogonal axes—the modeling axis and the inference axis. Which basically translates into how do I express my problem and how do I compute the probability of my hypothesis given the data? The nice thing I like from this Bayesian perspective is that you can engineer along each of these axes independently. Of course, they're not perfectly independent, but they can be close enough to independent that you can treat them that way.

When I look at things like deep learning or any kind of LASSO-based linear regression systems, which is so much of what counts as machine learning these days, they're engineering along either one axis or the other. They've kind of collapsed that down. Using these LASSO-based techniques as an engineer, it becomes very hard for me to think about: "If I change this parameter slightly, what does that really mean?" Linear regression as a model has a very clear linear additive Gaussian model baked into it. Well, what if I want things to look different? Suddenly all of these regularized least squares things fall apart. The inference technology just doesn't even accept that as a thing you'd want to do.

The reason my entire team and I fell in love with the probabilistic generative approach was that we could rationally engineer in an intelligent way with it. We could independently think about how make the model better or how to solve the inference problem. A lot of times you'll find that by making the model better—that is by moving along the modeling axis—that inference actually becomes easier, because you're more able to capture interesting structure in your data.

<sup>&</sup>lt;sup>8</sup>E. T. Jaynes, Probability Theory: The Logic of Science (Cambridge University Press, 2003).

Another great thing about this approach is that you can have two different sets of people working on the same problem space. With the current techniques, if you want to move jointly along this space then you need people who know everything. And that's really hard when hiring. It's hard to find the person who knows optimized C++ numerical methods and really understands all these kernel tricks or similar techniques; whereas with the generative approach I can find people who are really good at modeling but only work with small data, and I can find people who are as not as up on the modeling but know how to do really efficient inference. Then I can put them together and get a lot out.

Gutierrez: What is nontechnical advice you give your friends?

**Jonas:** The biggest thing I think people should be working on is problems they find interesting, exciting, and meaningful. Today I saw a quote on Facebook that said that a data scientist is a scientist who wants to feed his family. This is not entirely incorrect. There are a lot of interesting problems out there that I think a lot of people can get excited about—and life is too short to not be having fun. So I hope that most people are operating in that space. For my friends who are just graduating college, I tell them, "No, don't go do finance if you're not really excited about it. There are so many other interesting things." In thirty years, you're not really going to care about that extra money. It won't be a thing if you work on problems you find interesting and meaningful.