

## Lessons learned managing a data science research team.

BY KATE MATSUDAIRA

# The Science of Managing Data Science

“WHAT ARE THEY doing all day?”

When I first took over as VP of Engineering at a startup doing data mining and machine learning research, this was what the other executives wanted to know.

They knew the team was super smart, and they seemed like they were working really hard, but the executives had lots of questions about the work itself. How did they know that the work they were doing was the “right” work? Were there other projects they could be doing instead? And how could we get this research into the hands of our customers faster?

And that was my job. You do not hire a VP of Engineering unless you need a VP of Engineering.

Over the next year we addressed all these questions, and we were able to shorten our timeframe from idea to product from six months, to 2–3 months. In the rest of this article I want to share with you some of the lessons we learned managing a data science research

team that worked closely with a product team. And I hope they give you some ideas and insights you can integrate into your own work and process.

When you have a team of people working on hard data science problems, the things that work in traditional software do not always apply. When you are doing research and experiments, the work can be ambiguous, unpredictable, and the results can be difficult to measure.

As a manager coming from the standard software development process, I was well versed in agile, scrum, and the other processes du jour. However, the research work we were doing did not fit neatly into those boxes. Things were a bit too messy.

We were proposing ideas, investigating hunches, and testing hypotheses. It was very difficult to estimate the work, and we could not guarantee the results would be good enough to ship. So we had to come up with a new process that would account for all of this uncertainty.

Back to our original question: “What are they doing all day?”

I could have easily gone and just asked everyone and provided a status report, but that would not have really addressed the actual question.

The heart of the matter was that we needed a clear process that would work well for research, while still making collaboration with our product teams easy. We needed a process where we could take advantage of the synergies of a group of people who know a lot about the data and a group of people that know a lot about our customers, and get them working together as one team.

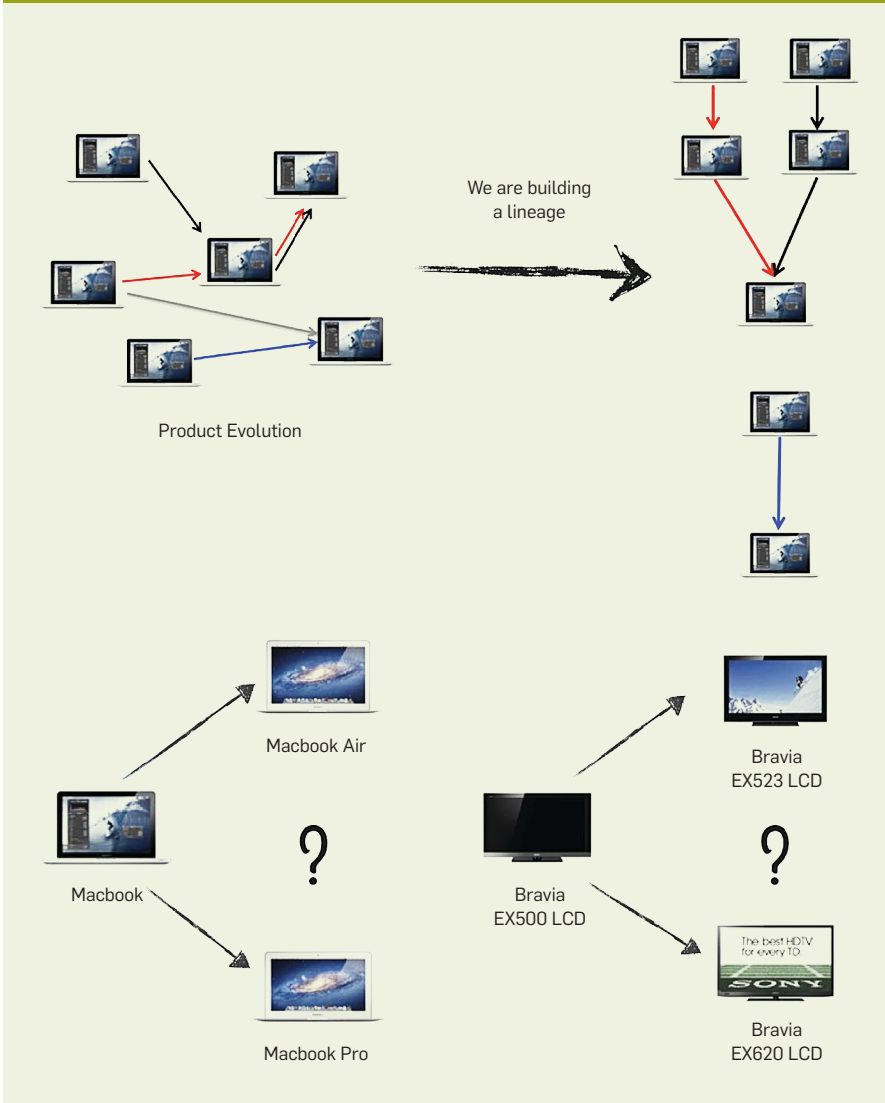
We needed clear channels of communication and a process that would support our future work. And with that in mind, we focused on the key areas of communication and logistics.

### Communication: Bringing Transparency

The very first thing we had to do was improve our communication. Our dif-



## Model lineage examples.



ferent teams were not speaking the same language.

Answering someone's question was not as simple as just explaining the right answer; it was about learning to use the right vocabulary to reach true understanding. *Communication isn't just about the words, but the meaning behind them.*

**How are you communicating results?** For example, one thing that always tripped us up was the question "What does it mean to be finished?" The data science team was measuring their results, and we would complete an experiment, but how did we know when an algorithm or model was good enough to be integrated into the product? We needed more than the measurement of results; we needed success criteria.

When we were creating a model,

we would measure the precision and recall of an algorithm and communicate our results that way. However, not all executives speak in those terms. So we started changing the way we talked about experiment results. Instead of putting things in terms of algorithm evaluation metrics (like precision and recall), we started communicating results in terms of the customer experience and business metrics.

Before, we would communicate results like:

Precision: 80%

Recall 25%

And after we would communicate results this way:

For the top search terms\* accessories appear 25% of the time on the first page.

For top search terms the head products are present 90% of the time in the top 3 results, 98% of the time in the top 5.

\* Top search terms are the 1,000 most popular queries on our website over the last 30 days.

The second way made it clear what these measurements meant. And even people without understanding of the algorithms could understand and ask questions about the results.

How are you measuring the results? Another issue we ran into was we were using a random sampling to measure our results for an algorithm or model. And while that is certainly one way of testing the results, it did not necessarily map to the experience on the website. The algorithms we picked would perform well, but then completely miss the boat on an important product.

So we changed the way we evaluated our results by weighting them with actual customer usage. We would take head products, or products that had customer views, and then use those as the samples to measure our results. This gave us a better approximation of the real customer experience.

**Show improvements.** Another key when you are doing experiments to improve an algorithm, like tweaking search results, is to show the before and after. What changed with the experiment? Sometimes certain results would be improved, and others would decline, so it was equally important to understand and communicate both sides of the coin. With all the information we could make the best decision about how to proceed.

**Solving difficult problems.** Besides all the problems with communicating about the data and results, another miscommunication seemed to happen when we would embark on new ideas. The conversation would go something like:

An executive would say "I have this idea to create a model lineage for a product. Then we can show customers past products and use our algorithms to predict when new ones will be released."

A data scientist would take the requirements and go investigate the idea, work on it for a while and come

back with: “That is going to be really hard; I don’t think we should do it.”

But these conversations were not always consistent.

A lot of things were “really hard” and sometimes features would get pushed through anyway (frustrating the data science team that had to work on them), but other times something that was “really hard” would just stop the feature in their tracks—resulting in missed opportunity.

Since there was not a way to follow up on these conversations, or problem-solve as a group, hard problems only became harder.

If you are working in data mining and machine learning research, most of the problems are difficult. The thing is, sometimes the reason why they are hard can be solved in other ways—like better/different data, changing requirements, or adding special cases. So saying a problem would be “really hard” wasn’t a good enough reason not to try to solve it; and yet, it was consistently being used as the reason why things were not happening.

The key was learning to communicate why things were difficult.

For us, the thing that worked best was to use analogies and examples. By getting specific about why it was difficult, it was easy for people to understand the challenges. So for model lineages we used the examples in the accompanying figure.

All of these changes were just about thinking how we were communicating. It was tailoring the message for the audience. If you work in data science or research, it can be easy to forget that not everyone is as immersed in the details as you are, and that you need to communicate things a bit differently to ensure understanding.

### Logistics: Establish a Clear Process

The next step was adjusting how we did the work. Once we had addressed some of the communication problems, I can remember someone asking me, “How can we create a sense of urgency for the research team?” This comment wasn’t made because work wasn’t getting done, but simply because the research team did not really have deadlines. The work was finished when it was ready.

But we were a startup; everyone else in the company had deadlines, and we were racing against the clock to deliver as much value as possible while we had the runway to do it.

**Adding deadlines to research.** Research is called research for a reason, and even if we wanted to have a schedule and wrap up the deliverables with a bow, that does not mean the world would work that way. What we did, though, was create a backlog for our experiments, just like a backlog of tasks in traditional software development. These were all the things we wanted to try, in priority order.

For each of these items, we were able to define success criteria, assign them out, and even cost them (how long should this take?).

This ended up being really beneficial for the team. Instead of each person working on their own on one thing, we became a team working together from the same list of priorities.

**Adding agile demos.** Another thing we did was we scheduled monthly demos, where each member of the research team would share their progress. These heartbeat meetings helped keep everyone on the same page, and helped show the real progress that was being made.

These touch points also helped create a sense of urgency (since the demos created a deadline) and all of sudden we were moving much faster as a team.

Since we were still dealing with the unpredictable nature of research (who knows if an experiment would work or how long it might take to run and tweak the results), we decided to place a focus on speed.


We wanted quick proof-of-concept prototypes and fast iteration. Previously we might have waited until we had concrete conclusions to show results to the product teams, but in this new model we would share data and intermediate results more readily. We would just add lots of caveats.

At first this made the team uncomfortable; they did not want to show things they knew they could still improve. However, once we started doing this more regularly we were able to get feedback even earlier in the process. This early feedback even helped us make adjustments to the product

along the way (since we were able to see how the results actually looked, which was sometimes a lot different than we had expected or mocked up in the designs).

And when the product team started seeing the results, they would get really excited, which fueled the research team to get even more done.

For example, one of the features we wanted to work on was value estimation, where we would take apart the features of a product and assign a value to each one. The goal was to help users compare different products based on the features so they could understand what they were really paying for, and discover cheaper products with similar specifications.

Well after doing a lot of work, it was clear this was a really hard problem. However, using the same data and models, we were able to build a feature that allowed them to accomplish the same goals—comparing products, instead of features. 

### Related articles on [queue.acm.org](http://queue.acm.org)

#### How Will Astronomy Archives Survive the Data Tsunami?

G. Bruce Berriman and Steven L. Groom  
<http://queue.acm.org/detail.cfm?id=2047483>

#### Privacy, Anonymity, and Big Data in the Social Sciences

Jon P. Daries et al.  
<http://queue.acm.org/detail.cfm?id=2661641>

#### Queue Portrait: Hilary Mason

[http://queue.acm.org/detail\\_video.cfm?id=2532200](http://queue.acm.org/detail_video.cfm?id=2532200)

**Kate Matsudaira** ([katemats.com](http://katemats.com)) has just recently made a foray into novel territory as founder of her own company Popforms. Prior to that she worked in engineering leadership roles at companies such as Decide (acquired by eBay), Moz, and Amazon.

Copyright of Communications of the ACM is the property of Association for Computing Machinery and its content may not be copied or emailed to multiple sites or posted to a listserv without the copyright holder's express written permission. However, users may print, download, or email articles for individual use.