

The background of the cover is black, decorated with several abstract, brush-like strokes in red and yellow. These strokes are concentrated in the top right, middle right, and bottom left corners, creating a dynamic, artistic feel.

O'REILLY®

The Care and Feeding of Data Scientists

How to Build, Manage, and Retain
a Data Science Team

Michelangelo D'Agostino
& Katie Malone

REPORT

The Care and Feeding of Data Scientists

*How to Build, Manage, and
Retain a Data Science Team*

*Michelangelo D'Agostino
and Katie Malone*

Beijing • Boston • Farnham • Sebastopol • Tokyo

O'REILLY®

The Care and Feeding of Data Scientists

by Michelangelo D'Agostino and Katie Malone

Copyright © 2019 O'Reilly Media, Inc. All rights reserved.

Printed in the United States of America.

Published by O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472.

O'Reilly books may be purchased for educational, business, or sales promotional use. Online editions are also available for most titles (<http://oreilly.com>). For more information, contact our corporate/institutional sales department: 800-998-9938 or corporate@oreilly.com.

Development Editor: Alicia Young
Acquisitions Editor: Melissa Duffield
Production Editor: Katherine Tozer
Copyeditor: Octal Publishing, Inc.

Proofreader: Christina Edwards
Interior Designer: David Futato
Cover Designer: Karen Montgomery
Illustrator: Rebecca Demarest

June 2019: First Edition

Revision History for the First Edition

2019-06-20: First Release

The O'Reilly logo is a registered trademark of O'Reilly Media, Inc. *The Care and Feeding of Data Scientists*, the cover image, and related trade dress are trademarks of O'Reilly Media, Inc.

The views expressed in this work are those of the authors, and do not represent the publisher's views. While the publisher and the authors have used good faith efforts to ensure that the information and instructions contained in this work are accurate, the publisher and the authors disclaim all responsibility for errors or omissions, including without limitation responsibility for damages resulting from the use of or reliance on this work. Use of the information and instructions contained in this work is at your own risk. If any code samples or other technology this work contains or describes is subject to open source licenses or the intellectual property rights of others, it is your responsibility to ensure that your use thereof complies with such licenses and/or rights.

978-1-492-05396-5

[LSI]

Table of Contents

Introduction.....	v
1. Your Mission, Should You Choose to Accept It.....	1
The Periodic Table of Data Scientists	2
The Structure of Scientific Teams	4
2. How to Win Friends and Recruit Data Scientists.....	7
Getting Grilled	8
Who, When, Where, How	11
3. Interview with the Data Scientist.....	15
The Interview	16
Choose or Lose	18
All Aboard	19
4. Fear and Loathing in Data Science.....	21
The Journal Club	22
How to Hack	24
Getting Outside	25
5. To Agile or Not to Agile.....	27
History of the Agile World, Part I	28
Process, Process Everywhere	29
6. Chutes and Career Ladders.....	35
Technical Footholds	36
Organizational Footholds	37

Personal Footholds	37
Usage Manual	37
Example Career Ladders	38
7. Final Thoughts.....	47
Additional Resources	47

Introduction

Over the past 5 to 10 years, data science has grown tremendously. But as young as data science is as a discipline, the craft of managing data scientists is even younger. Many of today's data science managers were thrust into management roles out of necessity ("battlefield promotions") or because they were the best individual contributors, and many come from purely academic backgrounds. At some companies, engineering or product leaders are being tasked with building new data science functions without any real data science experience of their own. More and more people find themselves managing data scientists without the necessary toolset or role models or mentorship to do the job well.

This report aims to fill that gap—to become a resource that data science leaders (whether they're data scientists or engineers or product managers) can use to understand how data science management is both similar to, and distinct from, other types of management and to learn concrete tips for building and sustaining their teams. It's also aimed at anyone trying to decide whether managing data scientists might be for them someday.

Getting data science leadership right is important. Data scientists are difficult to hire. But too often, companies struggle to find the right talent only to make avoidable mistakes that cause their best data scientists to leave. The statistics on data scientist retention bear this out. In **surveys**, data scientists report that they stayed at their previous jobs for only one to two years. And they have an ever-expanding array of options. Half of data scientists **report** being contacted *at least once a week* by outside recruiters about new positions. Given this landscape, your best people won't stick around long if you're

messing it up. However, we believe that the mindful application of a few tenets of data science management can (inexpensively) keep your team happy, engaged, and productive.

In this report, we take you through the journey of building, managing, and sustaining and retaining a data science team. **Chapter 1** begins with an exploration of the different flavors of data scientist and how you should think about structuring a data science team for your business. In **Chapter 2**, we offer suggestions for building a pipeline of great candidates. **Chapter 3** covers interviewing and onboarding for success. In **Chapter 4**, we look at the issue of “FOMO” in data science and its cure—continual learning programs. **Chapter 5** tackles the issue of *how* your team should do its day-to-day work by delving into Agile methodology. And, finally, **Chapter 6** covers the crucial topic of designing and communicating a career ladder for your team.

This report doesn’t cover every single aspect of data science management—for example, how much a manager should code, how to have productive one-on-ones, how to give feedback, how to manage other managers, etc. We believe there are other great **resources** out there that cover some of those more universal management issues. We’ve tried to focus this report on what’s most unique to data science. We hope you find it to be a useful guide to the care and feeding of data scientists.

Acknowledgments

Thank you to everyone who gave us valuable feedback on drafts of this report, especially Skipper Seabold, Ali Vanderveld, and Conor Jensen. Particular thanks to Greg Ball, who heavily influenced the example career ladder and wrote the organizational and personal elements. We’d also like to thank our respective spouses for giving us the time and space in which to write this report. And finally, last but certainly not least, thanks to the numerous data scientists who we’ve managed over the years on whom we’ve made the countless management mistakes that have informed any wisdom that we have to share.

Your Mission, Should You Choose to Accept It

So here you are, pondering the running of a data science team. Maybe you're a data scientist who has risen through the ranks and been put in charge of a growing team. Maybe you're an engineering or product leader tasked with building a new data science function from scratch. Or maybe you're being proactive, sizing up what data science management looks like and figuring out whether it's for you some day. To do your mission well, you'll need to understand how to balance and structure a data science team, how to recruit and interview the best people to fill that team, and how to keep them productive and happy after they're in place. It can be a daunting task, but this report is here to help. And luckily, it won't self-destruct at the end. Over the following pages, we cover each of these topics in turn, with a focus on concrete, actionable tips that you can actually use.

To begin, it's crucial to recognize that there are several different flavors of data scientist who generally will fill different niches within your organization and need to be managed differently. Your first job as a data science leader is to quickly take stock of your team (or the team you're tasked with building), match it up against the goals of your organization, and target your efforts around building the right kind of data science team. In this section, we cover the different flavors as well as thoughts on how to structure your team within the context of the rest of the organization.

The Periodic Table of Data Scientists

“Data scientist” is an overloaded term. Different data scientists have different strengths, interests, and ways in which they contribute to your organization. Breaking the role apart into a few types of specialist archetypes helps your data scientists to understand how they should approach their work and can clarify expectations to make everyone happier. You don’t necessarily need to formalize these specialist roles and titles for your team (some data scientists don’t want to be pigeonholed), but being aware of these different archetypes will help you better understand your team and mentor them on career development.

If your team is small, young, or just starting out, this might be more detail than you need to be successful: don’t overthink your roles. However, these distinctions become important as your data scientists become more senior and you need to give them structured thoughts on what kinds of opportunities will help them grow (see [Chapter 6](#) on career ladders for more on this).

Operational Data Scientists

Some data scientists are operations oriented in the sense that their strength is applying data science to the everyday functioning of your business. An *operational data scientist* focuses on optimizing the organization’s decisions, using data, so that business goals are met more effectively. They work closely with business stakeholders, understanding what they do and helping them to do it better with data. An operational data scientist is a great communicator of technical concepts to nontechnical colleagues and helps translate business goals into data science models and metrics. They’re also fluent in all of the data systems that power the organization, from customer relationship management to enterprise resource planning to marketing automation. They know what the company’s top-line goals are for the year and write slide decks and applications with an eye toward getting recommendations adopted by the organization. At a certain level of seniority, an operational data scientist begins to look like a strategist, marketer, logistics coordinator, or other functional leader—but they’re extremely effective because of their ability to support recommendations with data and models, not just intuition.

Although there can be overlap between an operational data scientist and someone in an analytics or business intelligence role, there are crucial differences. Like the product-focused data scientist, who we look at in the next section, an operational data scientist is more likely to be doing modeling and engineering work than an analyst. Unlike the product-focused data scientist, their models and code are inward facing, analyzing and optimizing the function of the business itself, and their customers are sales or marketing teams rather than external customers.

Product-Focused Data Scientists

Another type of data scientist is the *product-focused data scientist*. This flavor of data scientist works closely with a product team, and their work might even be embedded within the product itself, using data science and machine learning to increase its value. For example, they might build models that power recommendations on the company website or app, or they might work with the product designers to answer questions like, “How do we know our new product will fill a need in the market?” “What feature should we prioritize on our roadmap?”, or “How will we measure whether this feature actually works?” A product-focused data scientist needs to be business savvy, like the operational data scientist, and keep macroscopic company goals in mind, but they also need to be more technically rigorous because of the quality and quantity of data that powers many modern products. Their code and models are more likely to reach the organization’s customers via the product, and, as a result, they are deeply user centric.

Engineering Data Scientists

A third way a data scientist adds value to an organization is via engineering work. The *engineering data scientist* builds and maintains the systems that power the work of the product or operational data scientists. They run production code and machine learning systems smoothly and efficiently, processing and analyzing datasets at scale and solving data-intensive problems. An engineering data scientist maintains a high standard of technical excellence, and in the case of a more senior engineering data scientist, they begin to play a role that looks a lot like a lead or principal engineer. Increasingly, the name of this role in the market is morphing to *machine learning engineer* or *data science engineer*, signaling the technical depth

required of someone doing this flavor of data science job. Technical project management and leadership and code review skills are crucial for this role.

Research Data Scientists

Finally, some data scientists are expected to play more of a purely research role. A *research data scientist* is solely tasked with advancing the state of the art, often in a field like deep learning or computer vision or natural language processing, without any explicit expectation that their work will be immediately useful to the company. Be careful and use this role sparingly: it's very unlikely that your team is a pure research data science team. Typically, these roles are more like the research scientist roles who are found at places like Google or Microsoft Research and are staffed (often exclusively) with PhDs. Very few organizations have the need for such a team or the structure to support them. Nonetheless, it's probably the most glamorous kind of data scientist, and it might be the expectation of people on your team. Manage those expectations carefully.

Another good way to think about these flavors is with the “Type A” versus “Type B” dichotomy. In Type A, the “A” is for “analysis,” mapping nicely onto the operational data scientist as we’ve described them. In Type B, the “B” is for “building,” and this maps to the product, engineering, and research data scientists.

The Structure of Scientific Teams

Although many companies know that they “should have a data science team,” there’s often no clear consensus on what that team should do or how it should interact with the rest of the organization. One of the best ways to cut through that fog is to use these archetypes to align your data scientists with more traditional departments and roles like operations and marketing, product and engineering, or research. That will help your less data-savvy colleagues understand the goal of the data science program and how to interact with it, which will make you more successful.

Given these different flavors, should you seek to balance your team with different kinds of specialists or load it with generalist “full-stack data scientists”? The specialist-versus-generalist debate continues to rage, but as we explain later in Chapter 6 on career ladders, we prefer the philosophy of the *T-shaped* data scientist. Look to hire

folks who are broad enough to hack their way through the basics of each function but have (or will choose to develop) depth in a particular area that aligns with one of the flavors.

Data science is becoming pervasive enough that, in many organizations today, data scientists need to serve multiple parts of an organization. However, this introduces a thorny question: how should you distribute your data scientists? Should they all sit together in a core data science team or department? Should they sit with their functional teams, closer to the business or product problems? Or is there an in-between model that works?

A centralized data science team can lead to a strong sense of identity, program coherence, and knowledge sharing. Data scientists love to talk shop with one another, and they take pride in their work, so having them sit and work together leads to high team cohesion as they push each other to explore and try new things. An upside of this is high job satisfaction and, hopefully, good team culture and high retention. The downside of this model is that it places your data scientists further from the problems that need to be solved out in the rest of the business, making it easier for them to end up adrift or down rabbit holes. A data science team needs to be effective as well as happy.

Another upside of the centralized model is that decisions about which workstreams to tackle (and by whom and how) are more likely to be made by trained data scientists. Having one centralized team with a centralized way of handling requests from outside teams can help the entire company to ensure it's using its precious resources efficiently.

A fully distributed data science team with the data scientists embedded within other business units or teams flips this dynamic around. The data scientists are closer to the actual work on the ground and gain crucial business context and awareness of the needs of their colleagues across the organization. However, even though the organization might benefit more from their work in this arrangement, the data scientists themselves often end up wishing for more like-minded people to help them think through data or software or methodological problems. Anecdotally, we've known a number of data scientists who have left organizations because of the isolation and lack of mentorship caused by the fully distributed model.

A good compromise is a hybrid of the two models. Sometimes this is called a *center-of-excellence* model, but whatever name it goes by, it's characterized by a central data science team that pools best practices and shares ideas while the data scientists themselves are primarily "loaned out" or embedded within functional teams for "tours of duty." In other words, although the data scientists spend most of their time accountable to the organization more broadly, there are structured chances for them to get together, share ideas, talk about what interesting problems they've uncovered, and generally create some of that culture that makes the centralized model so appealing. In a product or engineering-focused organization, this model can look like a centralized data science team where members are embedded in product or engineering teams, attend planning meetings and standups, and even sit with the team for the life of a project, but eventually return to their home base.

One thing to consider is how your team structure might evolve as your team grows and scales. When a team is just starting out, it's unlikely that a fully distributed model will be successful for reasons we've already mentioned. But at Facebook and Google scale, there will likely be enough data scientists to spread around that loneliness, as such, isn't a real factor. If you're somewhere in the middle, we recommend the hybrid model.

How to Win Friends and Recruit Data Scientists

You've spent time thinking about the type(s) of data scientists who are best for your particular organization and how you want to structure your team. Now how do you go out and find them, get on their radar, and recruit them to join you on your team's journey?

When you're trying to find data scientists to hire, the first piece of the puzzle is knowing where they congregate so that you can put your company's message in front of them. Data scientists tend to cluster, and they love to learn, so invest in community outreach in data-centric venues: go to meetups, speak on panels at local colleges and universities, and give talks at conferences. With demand for data science talent at an all-time high, prospects are (justifiably) picky about where they want to work, so you have to show them why your organization deserves their attention. Because data scientists love to learn and they want to work at places where they'll be able to continue learning as they work (see [Chapter 4](#)), think of yourself as a community data science educator rather than as a recruiter or a salesperson. This means that your meetup conversations, panels, and talks must first and foremost center around teaching the data scientists around you something new—whether it's something technical, methodological, organizational, or business related.

We've personally made use of two strategies that aren't for every organization but are extremely effective if you have the resources to do them well:

- Contribute to open source software either through packages that your team uses or by open sourcing some of your own code to the community. By its very nature, open source attracts intrinsically motivated individuals and immerses them in a rich community of volunteers looking to collaborate. Getting “street cred” with the open source community, which you get through working with them on valuable projects, gives you visibility and access to a great network. And it's the right thing to do, particularly if you're working at a firm that uses open source tools. One way to carve out the time for this is to use team *hack weeks* to encourage open source contributions (see [Chapter 4](#) for more on hack weeks).
- Teaching up-and-coming data scientists is another great recruiting option. This can take a lot of forms, like teaching night classes at the local college or university or online, writing and publishing technical articles or blog posts, or hosting a podcast. We've personally done all three and have directly hired great candidates as a result of each method.

Getting Grilled

If things go well, you'll have candidates reaching out to you (or at least responding when you or a recruiter sends that cold email or LinkedIn message). If you're having an initial conversation over coffee or a phone chat as part of an interview (see [Chapter 3](#) for more on interviewing), be prepared for some tough questions. In our experience, it's the intent behind the questions that really matters. Understanding how to interpret that intent will help you to put your best foot forward. Here are a few questions that you should anticipate:

What's your company strategy? What's the role the data science team plays in the company strategically?

This question is a candidate's way of gauging how integral data science is to the organization and, as a result, how central their place would be. As the data science leader, the candidate is looking to you to demonstrate that you have a vision about where

you're going and that you can convince someone (the candidate, in this case, but in other cases it will be your boss or the executive team or the board) that your vision is worth investing in. Have a concise, true, elevator-pitch answer ready here. A wandering, rambling answer or one that doesn't get to the heart of the organization's data strategy in a meaningful way will turn off sharp, ambitious candidates.

How do you decide what the data science team will work on? What's the highest priority project for your team to work on right now?

Many candidates ask this question when what they're really wondering is *who* calls the shots on what the data scientists work on. A good data scientist wants to be relevant and integral to the most important functions in the organization. Talking about the core product, business, technical, or other goals of the organization, and how the data science team is contributing to them with their work will speak to how their work will be valued when they're on your team.

On the other hand, data scientists also appreciate autonomy and want to see how they can bring creativity into their work. Being valued by the organization is great, but there can be too much of a good thing to the point that a data scientist ends up spending so much time servicing ad hoc requests from the organization that they can't make satisfying progress on the next great thing. The best way to walk this line is to speak to how other parts of the organization inform *what* the team works on, but balance that by speaking to the team's latitude both in *how* they solve a given problem and how they can generate their own ideas. Data scientists appreciate both clear objectives and autonomy, so the best answer will show a balance of the two.

Relatedly, you should also be prepared with answers for "How do you choose who works on what?" and "What would be the first project I'd likely work on?"

How do your data scientists work together and collaborate?

Many data scientists come from academic backgrounds, and there's often a deep loneliness in academia and in a lot of junior data scientist positions. Candidates who ask this question want to know that they'll actually have like-minded colleagues to connect to and work with.

What does onboarding look like? How are performance reviews and career paths handled?

Like most humans, your average data scientist probably appreciates clear direction and understanding about what they need to do to be successful at work. With many ways to contribute, though, sometimes it's not self-evident what should be prioritized, which can lead to data scientists being overextended and burning out. These bad experiences become compounded because data science is a new-enough field without strong norms around career progression that many data scientists find themselves needing to play internal politics to advance professionally. A good way to reassure someone with a rigorous analytical bent—someone who values the idea of dispassionate, data-driven decision making—that they're joining a meritocracy is to show them a set of written criteria outlining how they'll be evaluated. Asking about onboarding and career progression in general is a candidate data scientist's way to probe whether you've thought this through and will be rewarding them for the quality of their work versus other factors.

NOTE

See [Chapter 6](#) for our thoughts on how to structure career progression with a carefully crafted career ladder.

A second subtext of this question centers around diversity. The reality is that data science currently skews more male, and white, than the population as a whole. We talk more about the importance of team diversity in the next chapter. But we've found that candidates who would diversify your team or who value working with colleagues who aren't exactly the same as them use questions about performance reviews or career paths to really probe a classic place for bias to creep in. Having a strong onboarding and career pathing policy that treats everyone fairly and speaking to its importance when candidates inquire communicates that you value fair evaluation and diversity on your team.

What tools does your team use? What does your tech stack look like?

Data scientists often see an organization's tech stack as a signal for the kind of data science the organization does. The most

attractive jobs to most data scientists are those that use open source tools, especially those based on R or Python, for the bulk of the data science work. Of course, you're probably not in a position to completely retool your organization's tech stack just to make it more appealing to data science candidates, but be aware that your technical choices can communicate more than you realize.

Another facet of this question to be aware of is how your team gains access to the computing resources it needs. Do you have access to your own servers and scalable computing clusters? Or, is there an IT department or DevOps team that plays gatekeeper and can keep data scientists waiting for weeks or months? Data scientists who have been burned by this in the past will ask for details. Having a dedicated cloud account for the data science team or access to a self-service data science platform will put minds at ease.

Who, When, Where, How

Who you want to recruit in the first place will affect where and how you recruit. The lowest-risk hire is an experienced data scientist who already has a track record of accomplishments that mirrors what you expect of your new teammate. However, experienced senior data scientists are especially difficult to find because the field is relatively new, and you will likely find yourself needing to reach out to them proactively—they already have jobs, and many are older with families and other responsibilities that preclude being active on the meetup scene or in open source. However, they're disproportionately valuable because they have done data science before and can use their previous experience to wrangle technical, methodological, and organizational ambiguity. Experience and comfort with ambiguity are especially valuable in the early days of your team.

Increasingly, there are also more new-to-data-science candidates on the market. They can be easier to recruit because there are simply more of them, and they're more open to compromise because they're looking to get their foot in the door.

When can you recruit less-experienced candidates and train them up, and when must you find seasoned data science hands?

Our advice is to begin by hiring candidates who are as experienced as possible when you are starting your team, and branch out to more junior candidates to fuel the growth phase. The experienced first hires are the people who will set the expectations and culture, define and follow through on the first sets of deliverables, and create the structure that allows themselves to scale. The second wave, which includes candidates from data science bootcamps, graduates straight out of school, or those making transitions from other careers, will be navigating a lot of cognitive overhead and ambiguity already. Giving them a few experienced mentors who can guide them toward success will greatly increase the chances that they make the right decisions as they grow. That ultimately makes them, and you, more likely to succeed. In addition, we've known many junior data scientists who end up leaving companies when there aren't senior data scientists or leaders in place from whom they can learn.

We recommend against filtering too hard on background, given that great (and terrible) fits are organization specific and can come from anywhere. That said, here's a high-level (but by no means exhaustive) guide to typical strengths and weaknesses of candidates from different backgrounds to give you an idea of where you might want to focus your recruiting efforts and probe more during the interview stage:

PhD graduate in a quantitative field

- *Pros:* Proven ability to go deep on a problem. Self-directed in defining and executing on an ambiguous program of research. World-class expert in something (although it might not be something directly relevant to their day-to-day data science work). Potentially deep in the coding and statistics expertise that a product, engineering, or research data scientist needs.
- *Cons:* Might lack awareness of business problems and so might not make a strong operational data scientist. Potentially accustomed to working at a much slower pace than you're used to or than the business demands. The specific quantitative methods of their academic discipline might not translate well to machine learning and data mining.

Graduate of a data science master's program or data science bootcamp

- *Pros:* Usually has exposure to many different algorithms and technologies, although likely some of them at a superficial level.

Has demonstrated a willingness to invest in a data science career.

- *Cons:* Might not have well-developed instincts for how many algorithms and technologies work “under the hood,” which means that they might need lots of direction and supervision.

Software engineer turned data scientist

- *Pros:* Stronger engineering skills than the other types of data scientists. Likely to be experienced at maintaining legacy codebases and understanding “development” versus “production” status of projects. You should consider this person for an engineering data scientist role.
- *Cons:* Might not have strong intuition for statistics or business problems.

Data analyst turned data scientist

- *Pros:* Most likely to know what kinds of business problems are the most valuable and to empathize with nontechnical stakeholders. Could make a strong operational data scientist.
- *Cons:* Often needs significant mentoring in software engineering and so is less likely to be a strong product or engineering data scientist.

Finally (and this should go without saying), remember that the people who you’re recruiting are *people*, too. Data science isn’t a huge field, and you will likely run into some people repeatedly over the years. Moreover, a year or two is a long time in data science, and someone who wasn’t a perfect fit for your organization the last time you were hiring could be the ideal candidate now. If you treat recruiting as an exercise in community building, if you get to know your peers at other organizations, and if you treat your candidates well throughout the interview process, you’ll find that your reputation will begin to do a lot of the work for you.

Interview with the Data Scientist

Your recruiting strategy has finally paid off! You’ve hooked that candidate—the one you’ve been trawling for and piqued their interest. How do you reel them in?

It all starts with the interview, and the interview starts before your candidate even applies: it starts when they read your job posting. A well-written job posting convinces candidates that you and your company actually get what data science is all about and have realistic expectations for the role. We all know that the best data scientists wear many hats, but writing a concise job posting that emphasizes the most important things and foregoes an overly long list of “nice to have” requirements shows that you know what you’re looking for and helps the right candidates apply to your role with confidence.

Begin by having an honest conversation with your team—and with yourself—about which skills are really essential for someone to have right off the bat versus those that they can pick up after they join the team; unicorns are great, but they also don’t exist. And, importantly, **research has shown** that women and minorities are less likely to apply for a job when they don’t meet every single qualification that you list, so be careful not to scare away diverse candidates with a needless laundry list.

A handy framework for organizing your thoughts is to classify what you want in terms of *knowledge*, *skills*, and *dispositions*. Knowledge is, in a sense, book learning—familiarity with machine learning algorithms, knowing what a t-test or *p*-value is, or having previous experience in a particular industry or domain. Although knowledge

is about what someone knows, skills are about what that person can do—their street smarts rather than their book learning. Finally, dispositions encompass things like the candidate’s attitude, their mindset, and how they respond when things don’t go their way.

Taking these three categories as a starting point, write out bullet points for what are must-haves versus what are nice-to-haves in each category. If you have a sense of what flavor of data scientist you’re hiring for—an operational, engineering, product, or research data scientist (see [Chapter 1](#))—it will be easier to write the job requirements. A well-written career path document (like the one we’ll discuss in [Chapter 6](#)) can also serve as a great starting point.

The Interview

There are many (often contradictory) opinions out there on the best ways to interview and assess candidates. But if you are mindful about how you design your own process, the interview itself can serve as another selling point for your team and your company. It communicates that you take hiring and growth seriously and actually know how to evaluate data science talent.

Each assessment method has its pros and cons. Whiteboarding lets you see how candidates think on their feet, but can be unrealistic and anxiety provoking. Take-home challenges let you see how someone codes but can be prohibitively time consuming for candidates who have day jobs or families. Talking through code samples from GitHub provides great insight into a candidate but can disqualify people who work primarily in private repositories. Asking someone to give a presentation on their area of expertise makes it difficult to compare across candidates. We could go on.

That having been said, a few things have worked well for us and our teams:

- We’re advocates of interview questions that are concrete rather than hypothetical. For example, to gauge a disposition around curiosity and learning, ask the candidate about a time they had to learn something on their own instead of, “How would you learn something new on your own?”¹

¹ This is referred to as *behavioral interviewing*.

- Design technical screens to be as close as feasible to the actual job you will ask the person to do when they join your team.
- Does your team toss around ideas and whiteboard problems a lot? Then whiteboard away. But don't ask FizzBuzz² or bubble sort. We've found a good whiteboarding prompt to be: "Let's imagine that we're teammates and have been asked to tackle this business problem. Here's a dataset and the question or problem we've been asked to look at. Let's brainstorm and try to work through it." This single question gets at knowledge and skills, and how the conversation goes will tell you a lot about various dispositions.
- Do you do lots of pair programming? Pair your candidate up with a data scientist or engineer and have them solve a problem together.
- Will this person need to maintain or extend software built by someone else? Give them a code sample and ask them to explain it or debug or refactor it.
- Is communication of paramount importance for you? Pair the candidate with an engineer, product manager, or business person and ask them to teach a new topic the interviewer knows nothing about.
- If you do decide on a take-home challenge, let the candidate complete it over the course of a week or more rather than forcing them to do it in a single sitting, with the guidance that they really shouldn't spend more than a few hours on it over the course of that period. This accommodates people with families or other responsibilities and also prevents candidates who have more free time from going overboard. Use the take-home challenge as a prompt for further conversation in the interview process rather than as a test or a filter. Ask about particular decisions or pieces of code or how they would take an aspect of their solution one step further.

Make sure that your interview process is designed to get as many different opinions as possible, and pair interviewers to get two takes on each session. We can't tell you how many times someone had a

2 For perhaps the greatest data science interview mashup on the internet today, we highly recommend Joel Grus's "[FizzBuzz in Tensorflow](#)" blog post.

negative interaction with a candidate when their session partner said, “Actually, that’s not how I interpreted that at all...” and then the resulting conversation ended up in a different place.

Like most machine learning models, there’s no interview process out there that has perfect recall for the candidates who you want on your team without any danger of a false-positive bad hire. You need to decide where along this precision/recall curve you’re willing to sit. Do you need to err on the side of getting someone to do the job right now, even if there’s a higher risk of ending up with someone who isn’t the right fit in the long run? Or do you have the luxury of waiting and rejecting some candidates that would have been good to be sure that you’ve achieved a great fit?

Choose or Lose

If you’re lucky, you now have a choice to make. However, data science is such a new field with so many people entering it from so many sources that you can end up with several good candidates for the role who are very, very different. One might be a strong software engineer who’s coming up to speed on statistics and machine learning. Another might be a recent PhD graduate with excellent scientific skills but little coding experience. A third might be a candidate with data analysis experience and business awareness but weaker software and statistics knowledge than the other two. How do you break the tie?

If there were a single, clean answer, you would already know it. Hiring always involves judgment calls, and your role as the leader is to maintain confidence in the face of that ambiguity. Here are a few tips, though.

If your team is new or small, chances are your data scientists will spend most of their time building data pipelines and figuring out what they should even be measuring. Having a few seasoned hands around at this stage can be disproportionately valuable. As your basic infrastructure (technical, process, cultural) becomes more established, hire for more niche skill sets like advanced machine learning. When you begin to find that your high-value senior data scientists are spending increasing proportions of their time maintaining models and pipelines or answering one-off questions from the rest of the organization, that’s your sign to invest in junior candidates. The jobs are well defined enough at this point that they’ll be

pretty likely to succeed, and as the junior folks come up to speed, it will free up bandwidth for your most valuable senior people.

One other point worth mentioning around hiring data scientists is diversity: a diverse team is valuable to have but is exceedingly difficult to build. Nonetheless, there are steps you can take during the hiring process to make it more likely:

1. Encourage the interviewers to keep their opinions of a candidate private until they've had a chance to write them down and enter their feedback. This is one of the best ways to avoid groupthink and protects your candidates against biases creeping in where the loudest or most opinionated interviewer ends up dominating the overall sentiment toward the candidate.
2. If you have a process of debriefing with the hiring group after an interview, have the highest-status interviewer go last in offering their opinion: having them go first sets up the team for a dynamic in which everyone else in the room will simply agree with the boss.
3. Try to have a diverse panel doing the interviewing. This makes it likelier that your panel will consider more angles on the candidate, and it will differentiate you to the candidate who, most likely, is interviewing at other places.³

All Aboard

After you've made your hire, you'll need to set them up for success. A good onboarding plan is one that lays out, as clearly as possible, what you expect of the new data scientist and what they should do to live up to those expectations. Strong career paths help here (see [Chapter 6](#)) because they lay out the expectations of the role, and if you interviewed well, you might have ideas about where the new

³ Be thoughtful in this approach, though. Bad dynamics can develop sometimes where, say, the one woman on the team ends up interviewing every candidate, making it more difficult for her to ship as much code or do as much data science as her male peers. Our preferred mode of dealing with this problem is to explicitly value interviewing in a way that's comparable to technical tasks. Hiring new team members is one of the most important things your data scientists will do, and you should reward good interviewing accordingly.

hire has skill or knowledge gaps and can set up an onboarding plan to address that.

For example, if someone is starting on your team fresh out of grad school in statistics and with minimal software engineering experience, you might want to pair them up with a more experienced engineering data scientist to maintain an important software tool. After they close their first few bug tickets, maybe they'll graduate to adding a new feature. As they ramp up, acting with more autonomy and making more high-impact decisions, they should be getting regular feedback in one-on-ones and performance reviews during which they can chart their progress and know where to focus next. Maintain a growth mindset⁴ here, and give positive reinforcement when they reach a milestone. This will help ease anxiety and establish **psychological safety**.

Tactically, we are big fans of 30-day and 90-day onboarding plans. A new data scientist will likely need at least 30 days just to learn their way around, and writing 30-day goals will keep them from feeling overwhelmed in those first few weeks. Goals might include, for example, having coffee with everyone on the team, working with a designated colleague on these three bug tickets, or giving one 15-minute presentation on a data science topic in our Journal Club or team meeting. The 90-day goals serve to get the new hire the rest of the way up to speed, as 90 days is enough time to get over the initial transition and begin really contributing. At the end of 90 days, your new hire should have accomplished a sampling of tasks similar to what your other team members are expected to accomplish in their day-to-day work.

One other important function of 90-day goals is that they give you, the boss, a chance to check in on the new hire's performance. Hopefully you've hired well because your interview process does a good job of testing the knowledge, skills, and dispositions that you care about, but a bumpy onboarding is often a sign of trouble to come. At 90 days in, you have enough data to know whether anything is amiss, and you are early enough in your relationship with the new hire to course correct.

⁴ *Mindset: The New Psychology of Success* by Carol Dweck is a valuable read for any manager.

Fear and Loathing in Data Science

Now that you've gone through all of this effort to recruit, interview, and hire a data science team and have structured that team for success, how do you make sure that they stay happy and stay with you?

We'd argue that one of the most crucial factors for retention is understanding a key psychological fact about data scientists: FOMO is a real phenomenon within data science. FOMO—or the fear of missing out—is that quintessential emotion of our social-media age, that anxious feeling in the pit of your stomach that you're sure something more fun and cooler than what you're currently doing is happening somewhere else out there in the world. In data science, it can take the following forms: *"My company isn't doing any cool machine learning projects right now."* *"We don't really have big data... we just have a MySQL database, but I hear everyone else is using Spark."* *"We're not even doing deep learning...or reinforcement learning...or GPU-accelerated Bayesian inference for time series modeling."* Or whatever the next big thing is.

Roughly 40% of data scientists say that challenging work and learning opportunities are their top two motivating factors for changing jobs. And in fields that are moving and changing at lightning pace, as data science and machine learning are, FOMO masks a real, valid fear at its core: *"If I'm not learning at my job, I'm going to be left behind and soon I'll be irrelevant."*

So how do you make sure that FOMO doesn't lead to loathing your leadership and leaving your company?

It's your job, as a data science leader, to hire data scientists who first and foremost care about having a real, measurable impact on your business, not just about using the newest, shiniest methods and tools. And it doesn't stop with hiring—you'll constantly want to teach and mentor this part of the job, especially given that a lot of data scientists come out of graduate school where the push is to stay on the bleeding edge (with the added bonus that you don't need to stick around to actually maintain any software...).

But assuming that you've done that, you're still going to need to give your team opportunities to practice continual learning. Luckily, there are relatively inexpensive ways to keep your data scientists engaged and learning and feeling like they're working on the cutting edge, even if your business isn't. These learning opportunities also give you the chance to upskill a team member in a particular area that your carefully designed interview process might have uncovered as a weakness.

The Journal Club

Our first piece of advice is to institute a *Journal Club*. A Journal Club can be very informal—we usually do them over lunch—and it's nothing more than people reading a particularly interesting journal article or technical blog post, discussing it, and explaining it to one another. We've found success when the person who selects the paper does a short presentation to kick things off, but that's optional. Open up your Journal Club to anyone in the organization who wants to come. It's a great opportunity for cross-pollination and for evangelizing data science to interested engineers and other analytics-minded people in your organization.

Where to Get Ideas for Papers

We're big fans of The Morning Paper, which delivers a writeup on an interesting computer science or machine learning paper every morning. Data science newsletters, weekly link aggregators, and podcasts are also good sources of material. Here is a list of our favorites:

- [The Morning Paper](#)
- [Data Machina](#)
- [Data Science Weekly](#)
- [DataElixir](#)
- [The Data Science Roundup](#)
- [The Analytics Dispatch](#)
- [Import AI](#)
- [SkyNet Today](#)
- [The Gradient](#)
- [O'Reilly Data Newsletter](#)
- [O'Reilly AI Newsletter](#)
- [PyData Newsletter](#)
- [FlowingData](#)
- [Data Is Plural](#)
- [The Wild Week in AI](#)
- [Deep Learning Weekly](#)
- [KDNuggets News](#)
- [DataTau](#)
- [Linear Digressions](#)
- [DataFramed](#)
- [Talking Machines](#)
- [Data Engineering Podcast](#)

As a leader, set the tone that reading widely and sharing new learnings is a part of what it means to be on the team by sharing your favorite articles on Slack or an internal team email list. Make sure to participate in the Journal Club. Show your team what's important to you through your most important tool: setting aside valuable time on your calendar. Show up and ask the dumb questions. Be vulnerable and curious. This will help establish that crucial feeling of psychological safety that we mentioned earlier.

As an alternative or supplement to a Journal Club, consider hosting a data science “movie night” (or movie afternoon, to be more family friendly). Pay for some popcorn and watch a video of a conference talk or tutorial session. Many of the big conferences make their videos freely available (PyData is particularly good about this). We find that people often accumulate a long list of talks that they want to watch but just don’t have the time. Watching them as a team both carves out the time and fosters that culture of continual learning.

How to Hack

Next, we think it’s essential to give your team unstructured “hack time” to actively work on something new or speculative. Of course, Google popularized the concept of “20% time,” and the stories about Gmail and other popular products emerging from it are legend. Over the years, we’ve experimented with various forms of hack time for data science. A half day every week or a full day every other haven’t worked well, which can be explained in part by **Google’s dirty secret**: it’s just too difficult for people to actually pull themselves away from their day jobs. Plus, it’s challenging to get a meaningful piece of data science done in such short chunks of time. Whole team hackathons can solve this problem, but they don’t let each person explore what they’d most like to learn.

For these reasons, we’ve eventually landed on a preference for individually scheduled “hack weeks,” during which each team member can explore a new software package or language, a new statistical technique or tool, or do something with the company’s data that they just haven’t had time to delve into. Give your team members the guidance that they should treat a hack week as they would a vacation—plan for it far enough in advance to get clearance from any projects or meetings. To be productive, the hack week must have a concrete outcome planned ahead of time—an application, a software prototype, a notebook documenting the research process for others to read, or a blog post.

We’ve found that some light dosing of accountability helps make hack weeks successful. The project should have a small amount of daily planning in a tool like JIRA and daily check-ins on progress with a hack week “buddy.” At the end of the week, mandate a presentation to the rest of the team so that they can also learn from the experience. In addition to giving your team the opportunity to learn

something and stay on the cutting edge, we've personally had a shockingly large number of projects that started as hack weeks blossom into important new capabilities for our teams. Keep a running list of these, which you can use as justification with your boss for setting aside the hack time.

Hack weeks are also a great opportunity to encourage your team to contribute back to open source. Explicitly condoning contributions to open source is a solid retention mechanism, and your team will expect it if you used open source as a recruiting tactic.

Getting Outside

Finally, we think it's important, to the extent that your company can afford it, to have a clear policy on conference attendance. We encourage abstract submission to conferences and pay to send a team member if their talk is accepted. That's both a great opportunity for your team member but also for generating PR (and valuable recruiting points) for your company. If it's in your budget, fund one conference a year just as a learning opportunity. If conferences aren't in your budget, you can rally your team to speak at and attend local meetups and can also use the aforementioned "movie night" concept as a substitute.

Getting your data scientists outside your company's walls is good for career growth and for gaining valuable perspective that can (ideally) alleviate FOMO. When your team gets out there and talks about your projects and hears about the experiences of others, team members might just realize that your team is doing cool and interesting work and that the problems it faces are actually quite common across the industry.

To Agile or Not to Agile

You’ve recruited and hired your team. You’ve set them up for continuous learning to keep them engaged and performing their best for your business. Next, you’ll want to assess *how* you’re going to work together as a team to achieve those fantastic business outcomes your boss has been dreaming of (and telling *their* boss to expect...).

Over the past few decades, “Agile” has taken the software world by storm and become a cottage industry in and of itself. Data science teams in companies with lots of software engineers face a choice about how they’re going to work: to Agile or not to Agile? Perhaps even more so, data science teams that don’t work regularly with engineering teams should ask themselves the question: Should we learn from the experiences of our engineering sisters and brothers, or does data science require a different way of working?

In this chapter, we make a distinction between the Agile mindset or philosophy and all of the trappings that have become the Agile process. We maintain that although the Agile mindset is a great fit for data science work, teams must beware of adopting all of the formality of the Agile process lest it suck the life (and ultimately the exploratory creativity) out of a data science team.

History of the Agile World, Part I

The Agile movement traces its origins back to the *Agile Manifesto*, which was written by a group of software engineering process thinkers at a ski resort in Utah in 2001. In its summary, the group wrote the following:

We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

- Individuals and interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

That is, while there is value in the items on the right, we value the items on the left more.

The group went on to enumerate some core principles behind its thinking:

- Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
- Business people and developers must work together daily throughout the project.
- Simplicity—the art of maximizing the amount of work not done—is essential.

The manifesto, and Agile itself, developed as a response to *waterfall* software development. In the rigid, linear waterfall process, the team collects and documents all product requirements upfront, then produces a comprehensive design and project plan, and then finally goes off to implement the plan and verify the work. Although waterfall works well for designing bridges or airplanes, the proponents of Agile argue that it doesn't work well for modern software—it's just too slow and rigid. In the real world, requirements change and stakeholders often find that the end product isn't quite what they were expecting and needs to change. With Agile, a team iterates quickly, shows working software as often as possible, and responds to stakeholder feedback and requests for change early and often.

Process, Process Everywhere

Like most revolutions, Agile eventually became so accepted as to become almost anodyne. Few software development teams these days would say they implement waterfall, and an entire industry has grown up around Agile, with its own lingo, roles, processes, and certifications.

There's *kanban* versus *scrum* (two different “flavors” of Agile). There are meeting ceremonies—*backlog grooming* and *sprint planning*—to parcel out the work, *standups* and *retros* to check on progress and to evaluate the health of the team, and *demos* to show work in progress and gather feedback. There are formal roles to enact: *product owner* and *scrum master*. Tasks are *pointed*, and the *scrum master* measures *velocity* and *burndown* to understand how the team is performing and how it can improve (for more on the lingo, see the upcoming sidebar, “Agile Terminology”).

When it's executed with all of its rigid formality—where work is excessively ticketed and specified and pointed with every last detail spelled out—Agile can sometimes feel heavy and like working on an assembly line. Especially if you come from academia or a research background, all of it can feel a bit like [this e-card meme](#): “Congrats on being just another cog in the capitalist machine, turning forever until the day you die!”

Most data scientists joined the field because of the thrill of poring through datasets to find new things and to build new insights and products on top of that data—things that were unexpected or not planned for. As Stitch Fix's Eric Colson writes, data science is more than just an assembly line executing neatly planned work as efficiently as possible:

Alas, we should not be optimizing our data science teams for productivity gains; that is what you do when you know what it is you're producing—pins or otherwise—and are merely seeking incremental efficiencies. The goal of assembly lines is execution. We know exactly what we want—pins in Smith's example, but one can think of any product or service in which the requirements fully describe all aspects of the product and its behavior. The role of the workers is then to execute on those requirements as efficiently as possible.

But the goal of data science is not to execute. Rather, the goal is to learn and develop profound new business capabilities. Algorithmic products and services like recommendations systems, client

engagement bandits, style preference classification, size matching, fashion design systems, logistics optimizers, seasonal trend detection, and more can't be designed up-front. They need to be learned. There are no blueprints to follow; these are novel capabilities with inherent uncertainty. Coefficients, models, model types, hyperparameters, all the elements you'll need must be learned through experimentation, trial and error, and iteration. With pins, the learning and design are done up-front, before you produce them. With data science, you learn as you go, not before you go.

— Beware the Data Science Pin Factory

If it causes your team to lose its creativity, the full-tilt Agile process can also be a great way to lose some of your best, most talented data scientists.

On its face though, this is quite contradictory. Agile is supposed to be lean and nimble. If it seems that all of the trappings of Agile ceremonies and processes will sap valuable creativity from your team, return to the useful wisdom in the original manifesto and the philosophy itself. For example:

“Responding to change over following a plan”

This is a nice summary of Colson's “learn as you go” data science versus the upfront design of the pin factory. As good data scientists know, our projects are iterative and constantly change depending on what we find in the data or discover in early model results. Data scientists need to be able to go where the data leads them, and they need to have the freedom to explore new ideas iteratively. This mindset is actually at the core of the Agile philosophy.

“Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.”

This is a frequent mistake that we've too often seen data science teams make. They go off to the top of their mountain and endeavor to come back months later with the tablets that will wow their stakeholders. Meanwhile, their stakeholders get restless and wonder why the team isn't delivering any value. To combat this, data science teams need to show their work. In an Agile ceremony, there is usually an end-of-sprint “demo” during which the team shows work from the previous few weeks to stakeholders for feedback. For data science teams, this means that they should show intermediate progress and results as often as possible. Demonstrate notebooks and work in progress. Build

simple apps or internal visualizations and let your stakeholders play with them to get an early feel for model and data product results. Incremental presentations are essential to getting buy-in for data science work.

“Business people and developers must work together daily throughout the project.”

Data scientists must work hand-in-hand with the business to understand how data is generated, how models will be used, and what kinds of projects will provide the most value. The model where data scientists go off into their corner to do their magic just doesn't work. The proponents of Agile realized this, as well.

“Simplicity—the art of maximizing the amount of work not done—is essential.”

This is another mistake that we too often see teams make. Deliver a simple logistic regression baseline before going off for six months to build your deep learning model. (Actually, as we mentioned two paragraphs ago, don't ever do that.)

So what's a data science manager to do? Our advice is, above all, to keep the Agile mindset and philosophy close to the heart of your team's operations. Deliver and show work frequently to your stakeholders, work closely with the business and respond to its changing needs, and keep your work as simple as possible.

When it comes to the formal process, we have two pieces of advice. First, unless you're at a brand-new startup, your engineering, IT, or analytics colleagues probably already have some processes in place for how they work. Because data science is so cross-cutting, think carefully about how you can make your team's process fit in with what the rest of the business is doing. If you don't, there might be an impedance mismatch in the different working styles.

Second, adopt as much, or as little, formal process as your team feels comfortable with, and constantly check in with them in retro meetings to develop your process. But be mindful of overdoing it and stifling exploration and creativity. Especially for new managers, light sprint planning, task pointing, and standups can be an effective way to run a data science team. As Camille Fournier has [tweeted](#):

Thinking today that the reason that Agile/Scrum are so persistent despite their shortcomings is that there is very little else that teaches engineering managers how to actually run their teams from an operational perspective.

For a few more thoughts on the role of Agile in data science, we highly recommend *Agile Data Science* and “A Manifesto for Agile Data Science” by Russell Jurney, as well as a talk by Jennifer Prendki, “Agile for Data Science Teams,” from O’Reilly.

Objectives and Key Results

For larger projects or initiatives that will take multiple two-week sprints to execute, you’ll want to look beyond the Agile framework for guidance. We recommend Objectives and Key Results (OKRs) as a way to align this longer-term work. OKRs are a management framework pioneered by Andy Grove at Intel and later made famous by Google. In brief, an objective is the *what*—it’s the overarching goal, the thing your organization is trying to achieve. Key results are the *how*—the milestones or achievements along the way that, when they’re all complete, will add up to the objective being accomplished. Objectives are the North Star, and key results are the measurable steps you take to get to that North Star.

One particular tweak to OKRs that we love is to frame each key result in three different versions or levels. The first is the “shooting for the stars” version. If you hit this level, you would be unbelievably ecstatic. The second is the “shooting for the moon” version. You’d still be jazzed if you accomplished this level. Finally, write a version that you believe you can achieve with “business as usual.” The expectation is that most of your key results will end up at the “business as usual” level. However, thinking about the moon and the stars gives your team something to strive for, and you can celebrate with them on the rare occasions when you hit these higher levels because you spelled out what excellence looks like ahead of time.

To take a concrete example, “surprise and delight the customer with personalized shopping experiences” could be an objective for your team. A key result could be to increase the click-through rate on your home-page recommendation widget by 2% by better tuning your existing algorithm. The “shoot for the moon” version might be to increase the click-through rate by 4% by incorporating additional data sources. And the “shoot for the stars” version could be to increase the click-through rate by 10% by deploying a new deep learning model that also includes additional data sources.

The advantage of this framework is that, by design, it helps guide people in your organization toward communicating what key results

they're working on and understanding how they contribute to the larger company objectives. If the objectives express the most important priorities in the company, and everyone in the company aligns with component key results, you're in the happy managerial position of having everyone rowing in the same direction. There's some further process that goes beyond the scope of this report, and if you're intrigued, we recommend *Measure What Matters* by John Doerr as a good introduction.

Agile Terminology

Kanban and scrum

Two different flavors of Agile. Kanban focuses on limiting the amount of work in progress, discovering bottlenecks, and continually improving the team's processes to increase throughput. Scrum is a more structured approach with certain roles where work is iteratively completed in fixed-time *sprints* (often two weeks long).

Pointing

Tickets, or units of work, are given point values that roughly describe their size. The rate at which points are completed in a sprint is measured and is known as *velocity* or *burndown*.

Scrum meetings

Scrum prescribes a certain set of meetings. *Backlog grooming* makes sure that tickets are properly specified and pointed. In *sprint planning*, the team makes a commitment to complete a certain set of tickets in the upcoming sprint. *Standups* are quick daily meetings to discuss progress and blockers. *Retro-spectives* are a chance for the team to reflect on the previous sprint and to fix any process or team dynamics issues. *Demos* show stakeholders the incremental progress made in the previous sprint and collect feedback.

Scrum roles

In scrum, team members are assigned to various roles. The *product owner* represents the product needs and decides what features the team should work on. The *scrum master* keeps things moving smoothly. In one **description**: “If the Product Owner is captain of the ship, then the Scrum Master is first mate. The Scrum Master is responsible for crew welfare and making sure team members follow protocol.” On your team, you're likely to be the overall product owner, responsible for your team's

output. You could serve as scrum master, deputize a team member, or consider hiring a project manager of your own.

Chutes and Career Ladders

If you're ex-academics like we are, you might find yourself unfamiliar with the notion of a career ladder. *"People should just do good work, and they'll end up in the right spot!"* is the mentality on which you might have been raised. One of the best things you can offer your team is what they might have lacked in academia: true mentorship and career guidance.

After starting to manage a data science team, you'll very quickly find that your data scientists want to know what skills they should be working on or what it takes to get a raise or to get promoted. They'll want to know what their future at the company looks like a year or two years down the line. Over half of data scientists **say** that growth and career advancement are their top two motivating factors for changing jobs. So, if you don't have a good answer for them, they'll find someone who does. A recruiter will come calling with the siren song of that sweet new senior or lead role, and they'll take the exit chute from your company because you didn't provide them a ladder.

In the early days of a team, you might be able to get away with vague, general advice in one-on-ones. However, nothing substitutes for a carefully constructed career ladder document that lets your people know exactly how you envision their career trajectory on your team. Especially because data science roles are so fluid and the same titles can mean vastly different things at different companies (see **Chapter 1**), your career ladder is your chance to stake out your position on what it all means at your company and on your team.

A career ladder is more like a guidebook than an instruction manual—it's not a comprehensive, rigid set of checkboxes, but it must generally lead you in the right direction. And while no two career ladders are going to look exactly the same, we think there is a universal set of dimensions that a good ladder should cover:

- First, it's essential that you provide both a technical track and a management track up your ladder. You should make it clear that someone can succeed and grow in both influence, project scope, and salary both by becoming prolific individual contributors and by managing people. Nobody is well-served by forcing your best technical data scientists to manage people just so that they can move up the ladder (even if that's how you ended up where you are...).
- Second, in our view, you should lay out the skills and attributes that someone on each rung of the ladder should exhibit in three key “footholds,” if you will: the *technical* dimensions of their work, their *organizational* interactions within and across your team, and the *personal* characteristics with which they do their work.

Technical Footholds

It's clear that you need to tell your team what sorts of technical skills they're expected to master at each stage in their career, and in many ways, this is the easiest and most straightforward part of your ladder. For this component of the track, we personally love the concept of the “T-shaped data scientist” from the O'Reilly report “[Analyzing the Analyzers](#)”:

We feel that a defining feature of data scientists is the breadth of their skills—their ability to single-handedly do at least prototype-level versions of all the steps needed to derive new insights or build data products (Mason & Wiggins, 2010). We also feel that the most successful data scientists are those with substantial, deep expertise in at least one aspect of data science, be it statistics, big data, or business communication.

Correspondingly, the technical component of our ladder first focuses on developing breadth across the different facets of data science (DS) expertise—the crossbar of the “T.” In our ladder (see the example that follows), we use “Machine Learning (ML) and Statistics,”

“Software Engineering,” and “Data Systems” to describe the different facets because our teams tilt toward product and engineering data science more than operational. As one progresses up the ladder beyond the senior level, we expect them to develop excellent depth and focus in one of those areas—the vertical part of the “T.” The ladder lays out what we think increasing depth means for each area.

Organizational Footholds

This part of the ladder is all about shaping expectations around how members of your team interact with one another and with other teams across the company. For example, a less-senior data scientist should be working effectively with peers primarily within the team and communicating status to managers and peers effectively. A more-senior data scientist should be mentoring, collaborating with product managers to proactively develop new data product ideas, serving as a sought-after expert, and beginning to communicate their expertise to the outside world through blog posts and talks. See the example career ladder that follows for more concrete examples of the organizational dimension.

Personal Footholds

Finally, what are the personal traits—the ways of working or the mental habits—that someone should develop at each level of the DS trajectory? For example, more-junior data scientists should learn to ask for help and unblock themselves and balance various demands on their time. More-senior data scientists should deliver quickly with minimal oversight, be persistent in the face of roadblocks, and go out and seek new sources of technical information and evangelize them to the team. The most-senior principal data scientists will have an organization or company-wide ownership mentality.¹

Usage Manual

After you’ve written your own career ladder, it’s important to socialize it well with your team. Hold a meeting and let everyone ask candid questions about what the different aspects mean.

¹ For a great overview of general technical career growth, especially its personal dimensions, we recommend Camille Fournier’s book *The Manager’s Path* (O’Reilly).

Use it in one-on-one meetings and performance reviews to guide career development conversations. Focus on the areas of the next rung where you think someone is excelling and which areas they need to spend time improving on if they're going to get there. If you know that a member of your team needs to work on a particular technical skill or organizational tactic, you can pay careful attention for day-to-day opportunities to upskill them.

If you're an employee, the career ladder presents an opportunity for self-reflection. One member of one of our teams likes to color code the elements of the ladder with green, yellow, and red to indicate his perspective on where he stands in each aspect. If both employee and manager do this exercise, it's a great prompt for an honest conversation about where you agree, but perhaps even more so about where you don't.

Done right, your career ladder will let your team know how to grow and advance in their careers, so they won't be forced to look to someone external for answers.

Example Career Ladders

General, multipurpose career ladders for data scientists can be difficult to write because of the multiple types of data scientists. That said, some themes are common enough to be applicable to most or all data scientists. We've captured some of these common themes in the career ladder templates below, but you should take these as a starting point and customize them to your team and organization.

Individual Contributor Track

Individual contributor data scientists are those who do not have explicit person management responsibilities. Many data scientists, like their peers in engineering, will appreciate a path for technical and professional growth that does not require going into management. Their career track should guide them toward technical and business leadership, as shown in Tables 6-1 and 6-2.

Table 6-1. A technical skills career ladder for individual contributor data scientists

Level	General Skills	ML & Statistics	Software Engineering	Data Systems
Data Scientist	<ul style="list-style-type: none">• Learns quickly and progresses consistently while requiring minimal feedback from senior data scientists• Develops the basics across all three of the technical areas	<ul style="list-style-type: none">• Can reliably apply fundamental techniques of supervised and unsupervised ML	<ul style="list-style-type: none">• Quickly learns to use fundamental tools of software engineering like version control, unit testing, Docker, and CI/CD	<ul style="list-style-type: none">• Learns to access and process data on a variety of database and parallel data processing technologies
Senior Data Scientist	<ul style="list-style-type: none">• Leads team through making well-reasoned design decisions• Develops and defends an approach while considering pros and cons of multiple implementations• Applies knowledge of industry trends and best practices and picks up emerging technologies quickly• Develops deeper expertise across all three of the technical areas	<ul style="list-style-type: none">• Develops deeper knowledge of multiple ML techniques, their strengths and weaknesses, and when each technique is appropriate• Reads and understands statistics and ML research papers	<ul style="list-style-type: none">• Writes clean, well-formatted, and well-structured code• Understands how to organize and develop a new software package• Can deploy custom APIs	<ul style="list-style-type: none">• Understands the inner workings of various database and parallel data processing technologies and their pros and cons

Level	General Skills	ML & Statistics	Software Engineering	Data Systems
Lead Data Scientist	<i>Sr. DS fundamentals plus:</i> <ul style="list-style-type: none"> • Excellent depth in one of the three specific technical areas to the right • Owns and operates large sections of codebase • Deeply understands the entire architecture of the data organization and can clearly articulate its scaling and reliability limits 	<i>Sr. DS plus:</i> <ul style="list-style-type: none"> • Understands cutting-edge ML techniques and can quickly apply them to new problems • Deeply understands the internal workings of one or more ML frameworks • Reads blogs and scientific papers and implements their new methods in code • Potentially develops expertise in experimental design and causal inference 	<i>Sr. DS plus:</i> <ul style="list-style-type: none"> • Leads the development of complex codebases worked on by multiple DS and/or engineers • Guides team philosophy on software engineering best practices with version control, testing, and CI/CD • Understands and is a go-to resource for the team on AWS, DevOps, and cloud computing topics 	<i>Sr. DS plus:</i> <ul style="list-style-type: none"> • Serves as an acknowledged leader for understanding the inner workings and the debugging of complex queries and workflows on parallel data processing systems • Designs systems with new and varied data types and underlying data architectures

Level	General Skills	ML & Statistics	Software Engineering	Data Systems
Principal Data Scientist	<ul style="list-style-type: none">• <i>Lead DS level in two of three technical areas</i>• Externally recognized as expert in an area of data science or technology• Speaks, writes, or evangelizes the use of data science or technology within the industry• Contributes to key open source projects• Promotes technology community• Pushes forward key technology or architectural change within the organization, generating momentum and enthusiasm along the way• Identifies and promotes opportunities for new products or transforms existing products via data or technology innovation			

A career ladder for individual contributor data scientists should also guide them toward taking on more leadership, and higher-impact roles, across the organization as they progress.

Table 6-2. A “soft” skills career ladder for individual contributor data scientists

Level	Organizational Skills	Personal Skills
Data Scientist	<ul style="list-style-type: none">• Works effectively with peers primarily within the team and communicates status to managers and peers effectively• Accepts feedback graciously and treats every project as a learning opportunity	<ul style="list-style-type: none">• Completes well-defined subtasks with little to no issue• Knows when to ask for help and how to get unblocked• Demonstrates time management skills and is able to balance various demands on their time
Senior Data Scientist	<ul style="list-style-type: none">• Works effectively with peers both on the team and off the team; influences peers beyond the immediate team• Understands the business impact and benefit of their product areas and projects• Collaborates with product teams to uncover new business opportunities• Contributes to team’s technical understanding by delivering documentation, tech talks, etc.• Mentors via pair programming and code review• Identifies problems and takes ownership of resolutions• Leads by example, demonstrating good teamwork and being a good teammate	<ul style="list-style-type: none">• Delivers quickly with minimal direction or oversight; responsible for end-to-end, high-quality delivery of complex projects• Demonstrates persistence in the face of roadblocks and dispatches them efficiently by pulling in others as necessary• Recognized as a leader in their team/functional/technical area by their peers• Proactively seeks out new sources of technical information in blogs and scientific papers and evangelizes them to the team
Lead Data Scientist	<ul style="list-style-type: none">• Leads and influences peers across all teams and functions• Sought for technical guidance and recognized as a prolific contributor• Introduces policies and procedures that increase the effectiveness of the entire business• Shares knowledge with the outside world and guides the creation of this content• Implements improvements across the organization	<ul style="list-style-type: none">• Consistently reduces complexity of projects and processes to get more done with less work• Thinks and acts critically, pragmatically, and consistently to deliver large, cross-team solutions on time and at a high level of quality• Recognized as a leader in multiple functional/technical areas by their peers

Level	Organizational Skills	Personal Skills
Principal Data Scientist	<i>Lead DS plus:</i> <ul style="list-style-type: none"> • Recruits externally, networking with interesting candidates to strengthen the talent pipeline • Proactively seeks out new challenges in new growth areas; prioritizes opportunities for maximum benefit, risk/reward • Creates enthusiasm around innovation, technical growth, and learning 	<i>Lead DS plus:</i> <ul style="list-style-type: none"> • Has an organization-wide ownership mentality and seeks out new partnerships across the organization for technical initiatives • Identifies issues with talent management, including how to mentor, evaluate, and promote career growth in the data organization • Communicates consistently across the engineering organization and the company

Management Track

As you likely know, moving into management isn't just an evolutionary step: it's an entirely different job from being an individual contributor. Many less-experienced data scientists might not fully appreciate that, so compare the career ladders in Tables 6-1 and 6-2 to the following Tables 6-3 and 6-4 as a way to preview both tracks.

A career ladder into management should encourage technical development in the early stages like the individual contributor path but then introduce more person management in the middle and upper levels. This management career ladder assumes progression through the data scientist and senior data scientist levels and then branches from the individual contributor track at DS manager.

The technical development of a data scientist is the same in the earlier levels, regardless of whether the person is bound for management or continues down the individual contributor track. If managing a team or department, a data scientist should be a credible technical authority but is generally not expected to build systems or develop methods of the same complexity as an individual contributor—after all, a manager has person management responsibilities that an individual contributor does not.

Table 6-3. A technical skills career ladder for a management track

Level	General Skills	ML & Statistics	Software Engineering	Data Systems
Data Science Manager	<ul style="list-style-type: none">• Helps to unblock, debug, and discuss statistical, software engineering, and data issues with reports• Mentors, pair programs, and conducts code reviews	Sr. DS level	Sr. DS level	Sr. DS level
Director of Data Science	<ul style="list-style-type: none">• Can quickly understand and vet the work coming out of the entire DS department for quality and correctness• Identifies and promotes opportunities for new products or transforms existing products via data or technology innovation• Speaks, writes, or evangelizes the use of DS or DS management within the industry	Lead DS level	Lead DS level	Lead DS level

Management requires organizational and interpersonal skills that make it a distinct role with a different set of tasks than an individual contributor. New managers especially should build a new mindset around their work: their effectiveness is measured in the output and happiness of their team, not just their own individual contributions.

Table 6-4. A “soft” skills career ladder for a management track

Level	Organizational Skills	Personal Skills
Data Science Manager	<ul style="list-style-type: none">• Manages the work of a small team, generally focused on a particular business or product area or technical function• Coordinates a small number of project workstreams• Works with several different product managers to define work and make sure it is executed• Conducts 1:1s, reviews performance, and offers career mentoring	<ul style="list-style-type: none">• Develops relationships throughout one or two parts of the organization to increase team’s influence and broaden team’s perspective
Director of Data Science	<ul style="list-style-type: none">• Sets strategic direction for the DS discipline• Active participant in leadership of the entire data group• Responsible for writing and surfacing all DS OKRs or goals and ensuring alignment with the broader data group and company goals• Communicates consistently across the company, including directly to C-level executives as needed• Can communicate with DS counterparts at external partners and vendors• Coordinates with all of the product managers across the company to make sure that their DS needs are being met and to proactively define and build new DS products• Manages the department’s entire portfolio of projects and workstreams• May directly manage a project or product manager for the DS department• Allocates projects and resources to subteams and team members, balancing abilities, interests, and growth• Identifies issues with talent management, including how to mentor, evaluate, and promote career growth among the DS department• Influences resourcing levels across the organization, ensuring resources are going toward key drivers of the business• Actively promotes the company externally, from recruiting, culture, customer, data, and technology perspectives	<ul style="list-style-type: none">• Develops relationships throughout all parts of the company to increase the DS organization’s influence and broaden their perspective• Recruits externally, networking with interesting candidates to strengthen the talent pipeline• Creates enthusiasm around innovation, technical growth, and learning

Final Thoughts

Whether you're a data science manager, a data scientist who has a manager, or an engineering or product manager who oversees data scientists, we hope that you've found something of use in this report. One thing to keep in mind as we leave you: a great data science team will challenge the person who manages it, almost by definition. The data scientists who you work with are people who have chosen a field that's characterized by its impact, by its challenge to its practitioners, and by its possibility. Your job is to capture the best of that and use it to navigate new frontiers for your organization. They will demand a lot of you, and hopefully you now have a few more tips and tricks to live up to those demands. Because, in our experience, the payoff to managing a great team is well worth the effort of cultivating one. Good luck, and go build something great.

Additional Resources

- *Analyzing the Analyzers*
- *Agile Data Science*
- *Building Data Science Teams*
- *Creating a Data-Driven Organization*
- *Competing on Analytics*
- *The Manager's Path*

About the Authors

Michelangelo D'Agostino is the VP of Data Science and Engineering at ShopRunner, where he leads a team that develops statistical models and writes software that leverages their unique cross-retailer ecommerce dataset. Previously, Michelangelo led the data science R & D team at Civis Analytics, a Chicago-based data science software and consulting company that spun out of the 2012 Obama reelection campaign, and was a senior analyst in digital analytics with the 2012 Obama reelection campaign, where he helped to optimize the campaign's email fundraising juggernaut and analyzed social media data. Michelangelo has been a mentor with the Data Science for Social Good Fellowship. He holds a PhD in particle astrophysics from UC Berkeley and got his start in analytics sifting through neutrino data from the IceCube experiment. Accordingly, he spent two glorious months at the South Pole, where he slept in a tent salvaged from the Korean War and enjoyed the twice-weekly shower rationing. He's also written about science and technology for the *Economist*.

Katie Malone is Director of Data Science at Civis Analytics, a Chicago-based data science software and consulting company that spun out of the 2012 Obama reelection campaign. At Civis she leads the Data Science Research and Development department, which tackles some of Civis' most novel and challenging data science consulting engagements as well as writing the core data science code that powers the Civis Data Science Platform and solves person-centric business problems with data. A physicist by training, Katie spent her PhD searching for the Higgs boson at CERN and Stanford and is also the instructor for Udacity's Introduction to Machine Learning course. As a side project she hosts Linear Digressions, a weekly podcast about data science and machine learning.

The background of the entire page is a vibrant red-to-orange gradient. Overlaid on this are several large, semi-transparent, overlapping circles in shades of red and orange, creating a dynamic, organic feel. The O'Reilly logo is positioned in the upper left quadrant.

O'REILLY®

There's much more where this came from.

Experience books, videos, live online training courses, and more from O'Reilly and our 200+ partners—all in one place.

Learn more at oreilly.com/online-learning