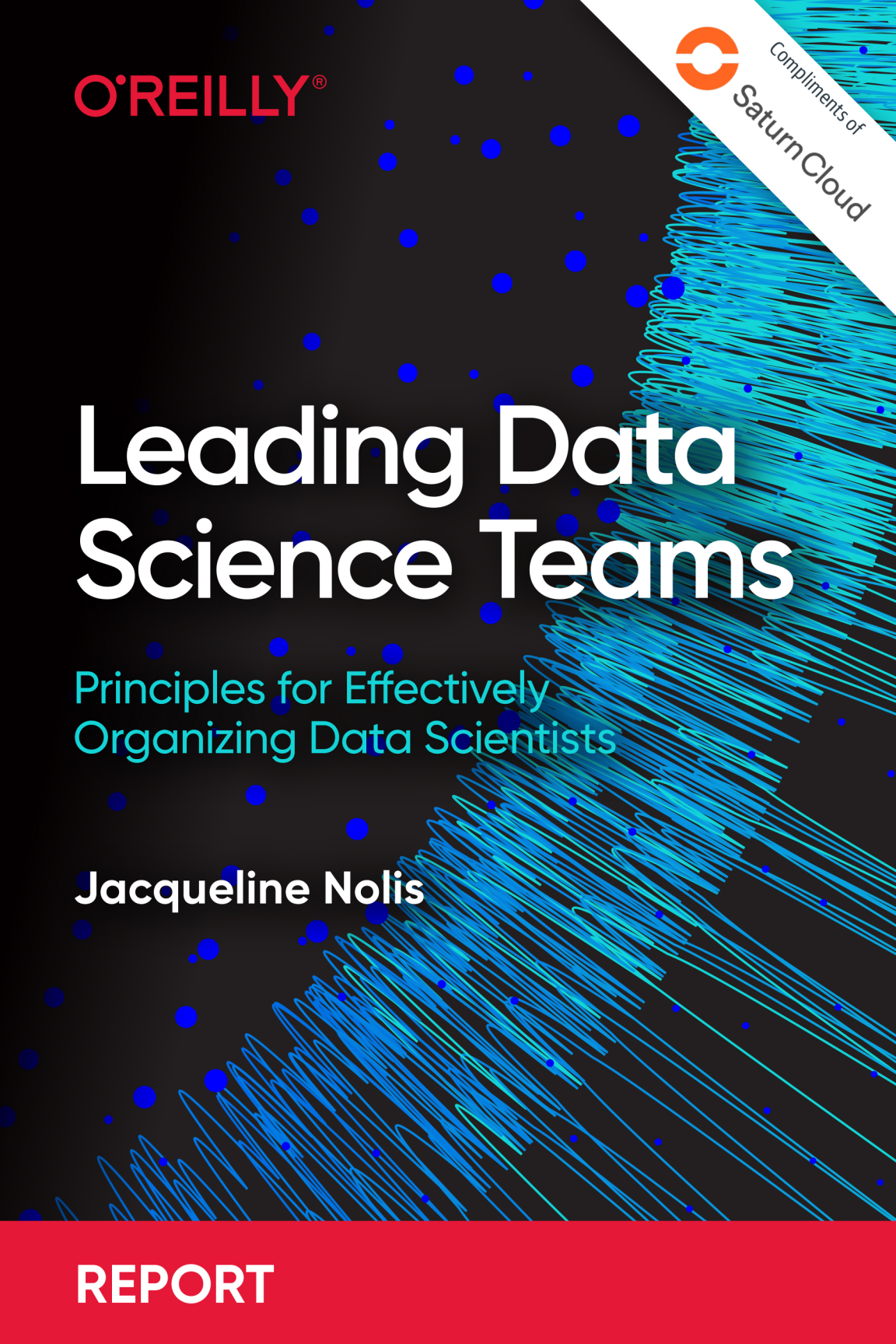
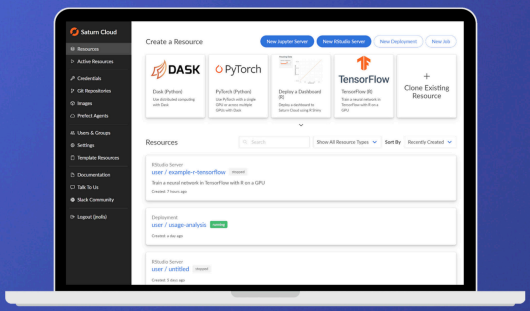

O'REILLY®
Compliments of
SaturnCloud
Leading Data Science Teams
Principles for Effectively Organizing Data Scientists
Jacqueline Nolis
REPORT

Saturn Cloud

A Flexible Data Science Platform For Your Team



ALL-IN-ONE WORKSPACE

- Scalable data science with Python, R, and more
- Use up to 4TB RAM, large GPUs, and Dask clusters
- Cloud-hosted JupyterLab & RStudio
- Easily share work with colleagues
- Straightforward deployment environments
- Run jobs and deploy dashboards and APIs
- Connect from existing cloud resources
- Admin reporting tools

Trusted by

NVIDIA. snowflake® checkout.com Cellarity VITAL PROTEINS® Senseye

AND MORE

Join thousands of data scientists today on our hosted platform or with our enterprise option:

LEARN MORE AT SATURNCLOUD.IO

Leading Data Science Teams

Principles for Effectively Organizing Data Scientists

Jacqueline Nolis

Beijing • Boston • Farnham • Sebastopol • Tokyo O'REILLY®

Leading Data Science Teams
by Jacqueline Nolis

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*.

Acquisitions Editor: Jessica Haberman
Development Editor: Melissa Potter
Production Editor: Katherine Tozer
Copyeditor: nSight, Inc.

Interior Designer: David Futato
Cover Designer: Karen Montgomery
Illustrator: Kate Dullea

March 2022: First Edition

Revision History for the First Edition
2022-03-10: First Release

978-1-098-12736-7

[LSI]

Table of Contents

Introduction

Leading a data science team presents a lot of challenges. Compared to other functions of an organization, data science tends to be highly speculative. A leader of a data science team won't know how a project will turn out until it's well underway. The data might be missing, there might not be a signal in the data, or the models might just not be practical. Sometimes leaders have to commit their data science teams to doing work before it's even well-defined. For example, a project will start to use data to figure out who the "best" customers are before data scientists have time to define what best means. Once the work is going, a leader has to manage the data scientists to make sure they stay on track and manage stakeholder relations when surprises crop up. Above all, data science teams are constantly being given more tasks to work on and last-minute, must-have deliverables often well beyond their bandwidth to handle. Those issues only scratch the surface of what a data science leader has to think about.

This report is a handguide for thinking about the problems data science leaders face when running a team. It's a reference to help think through the current challenges of a team or to be proactive about future scenarios. It's also a guide for people who have not yet had the opportunity to lead a team but want a head start for when they might do so in the future. Reading this report should give you a better understanding of what sorts of challenges teams might face and the tools required to think through them.

Unfortunately, this report doesn't provide an easy checklist of exactly how to successfully lead a data science team, because there are no easy answers. As you'll notice throughout the report, most of

the commentary will at some point say, "it depends on your particular company and team." But by pointing out what the challenges are, you should be better prepared for when your team encounters them.

Chapter 1 covers who should be on a data science team. Chapter 2 discusses managing the workflow of the team so that the many tasks a data science team might work on are handled effectively. Chapter 3 discusses some of the technical infrastructure decisions a data science leader must think through. Finally, Chapter 4 covers how to help the data scientists deliver value. We hope you find this reference helpful in managing your current team or a team you might manage some day in the future.

CHAPTER 1
Choosing Your Data Science Team

There is a wide variety of things a data science team can do, both because the term "data science" is absurdly broad and because different companies have very different needs. Data science can mean a company having a single analyst who makes data dashboards for executives or a team of people creating production-ready machine learning APIs. A company could need help using data to define a business strategy, to make tailored customer experiences, to forecast investments, or something else entirely! Because there are so many different types of work a data science team might do, there are many different ways to staff the team to meet the needs of the business. This chapter covers how to think through who you should hire for your team.

Every data science team will naturally have data scientists on it, but in addition to the data scientists themselves, there are many potential supporting roles that could be included on the team. Having more positions on the team can give you a wider range of capabilities but also requires managing more types of work. Consider that a data science team needs to do all of the following to effectively work:

Create a vision
: Design a strategy for what the data scientists will work on by communicating with stakeholders about their needs and assessing the capabilities of the team.

Project management
: Keep track of what is being worked on and communicate when timeline issues arise.

Stakeholder management
: Work with stakeholders to help them understand the capabilities of data science and figure out data science opportunities.

People management
: Help the team through performance reviews, feedback, support, and other managerial tasks.

Technical mentoring
: Help more junior data scientists work through issues and resolve technical roadblocks.

Data engineering (optional)
: Have data in a location that data scientists can then clean and use for their work, as well as locations for the outputs of models and analyses to be stored.

Software engineering (optional)
: If necessary, write code that will call the models the data scientists build, such as the UI and backend that show the model results to customers.

The data science work itself
: This is indeed important as well.

That is a lot of tasks besides actual data science—and almost none of the work is optional! Not only that, but a lot of these tasks can be done by people with different roles. This creates a mathematical matching problem for how to design a team, as Figure 1-1 shows.

From Figure 1-1 we can see there are many possibilities for how to cover the needs of a data science team. For instance, you could choose to hire a data engineer to do any data engineering or have a software engineer do software engineering *and* data engineering (and struggle a bit with the latter). The complexity of the problem increases since not all of these tasks need to be covered—you could ignore some and hope your team keeps running OK. Together there are many feasible solutions. So beyond the fact that your data science team is going to have data scientists on it, there are a number of different ways you can structure it, depending on some key decisions around who will be leading the team, who will be managing the projects, what sorts of technical roles will be on the team, and how much specialization will you have.

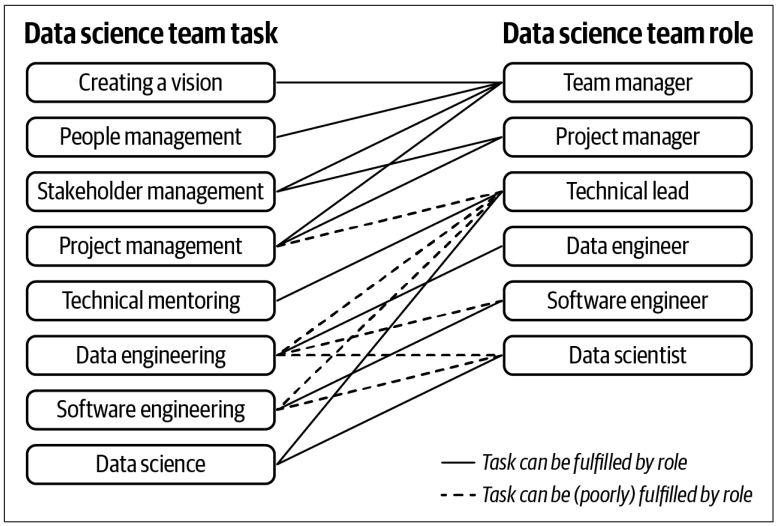


Figure 1-1. An example of the mathematical matching problem of designing a data science team. This is a bipartite graph, but the number of combinations is left as an exercise to the reader.

How Are You Going to Split Up the Leadership-Style Tasks?

A data science team really needs two different types of leadership—managerial and technical. The managerial tasks are those you'd associate with a typical manager. These are tasks like creating the strategy for the team, meeting with stakeholders about requirements, deciding who to hire and promote, and so on. The technical leadership makes the big technical decisions and helping the junior data scientists. These are tasks like deciding what kind of model architecture to use for a major project or mentoring a junior data scientist through their first analysis.

One option for running a team is to try and have a single person do both of these types of leadership work. However, generally, there are few people who are well qualified to do both the managerial leadership tasks and the technical leadership tasks. Even if a data science team does have a person with both skill sets, they're likely to be too busy to do both of those tasks well. So it's dicey to put all of this on a single person.

Thus, another option is for a team to have two separate roles, a *manager* and a *technical lead*. The manager works on all of the managerial tasks, including maintaining connections with the stakeholders. The technical lead comes up with the best technical solutions for the team to implement and helps mentor the team. Having two senior people is expensive, and if they aren't working well together, it can create a host of problems, but working in sync is a great option that lets the two leaders do what they do best.

Finally, a data science team can choose to have a leader who only focuses on one of those two types of leadership and accept that there will be gaps in what is managed. A data science team with a manager but no tech lead would need the data scientists on the team to be able to work without much mentorship, which usually means hiring more senior data scientists. A data science team with a tech lead but no manager would need to be able to trust that the other leaders at the company were still positioning their team well strategically. Both of these options are possible but not ideal. If the data science team is small, you might need to go with this option and then hope the rest of the team fills in the gaps.

If you are a leader reading this report, it is very important for you to understand your strengths and weaknesses and the best ways for you to support your team. If you come from a technical background and really wish part of your leadership job let you keep coding, then make sure you have strong support on the managerial component. If you do not have a technical background in data science, make sure when you hire data scientists, you have senior ones on your team and you listen to them. More than almost any other section of this report, the question "how are you going to do leadership-style tasks?" requires deep introspection and a willingness to share power and collaborate.

How Are You Going to Do Project Management?

Project management (organizing the tasks the team needs to accomplish and ensuring things are being completed on schedule) is critically important to ensuring that the team actually completes tasks at all. In addition to keeping the work itself organized, all parties must be kept informed and up to date on changes. Whether or not a person is explicitly assigned to the task of project management, this will

inevitably end up being done by someone, somehow. Maybe you have a single person who has the sole job of project management, or maybe a data scientist will create an Excel spreadsheet of tasks, but someone somewhere on the team is doing project management.

Oftentimes the project management is being done by the same person who does the people management. This makes a lot of sense: the person who is managing the team and meeting with stakeholders is well suited to also keeping track of what is getting done and communicating it with others. Unfortunately, a manager is often quite busy with meetings and their own work, so the project management can fall by the wayside, to the detriment of the team as a whole. Having the manager of a team do the project management requires lots of discipline from the manager and works best when the team is small.

Alternatively, you can hire a person to do the project management (an aptly named *project manager*). This frees up the manager to focus on other tasks. A project manager may or may not have a technical background. Having a technical background makes it easier for them to understand what the data science tasks and blockers are, but a nontechnical person should still be able to communicate with the data scientists enough to understand what the situation is and how to help.

If the project manager is an entirely separate role on the team, it's critically important that the project manager has organizational influence. There will be many times that the project manager will have to help the data scientists by going to other parts of the organization and fixing blockers like a lack of access to data. The project manager may also have to be firm with the data scientists—like telling them to stop working on the fun-but-noncritical task and instead finish the job that is needed by the end of day. If the project manager has influence within the organization, these sorts of situations will be possible to navigate (although still difficult). If the project manager does not have influence, it will be impossible. The best thing a team with a dedicated project manager can do to help them succeed is to back up the project manager and listen to them as much as they can.

As a leader of a large team, say around eight people or more, you'll almost certainly need to hire a separate project manager. As a leader of a small team, say less than five people, you'll probably not have

the budget and will need to do it yourself. If you have a project manager, the best thing you can do for your team is to hire the right person for the job and empower them to do it well. If you're doing it yourself, then the best thing you can do for your team is ensure you are consistently keeping up with project management.

Chapter 2 is focused on the methods for managing the data science work for a team, which is a primary focus for whomever is doing the project management. The chapter will discuss in more detail the topic of working with stakeholders and how to keep the work moving. Chapter 3 is about helping data scientists perform their best, which includes helping them take on their own project management. Strong project management is *quite* important to running a successful team.

Will Your Data Science Team Have Non–Data Science Technical Roles on It?

In some organizations, it makes sense to have non–data science roles on a data science team, like software engineers or data engineers. They can help get the data organized for the data scientists, write code that acts as wrappers around the models, and more. By having these types of people on the actual data science team, the data scientists will be well supported. The downside of having other engineering roles on the team is that an inconsistent amount of work might need to be done. For example, some weeks might not require any data engineering work, and it's not clear what a data engineer should be doing in those weeks. There also might be organizational redundancies. For example, if the data science team has data engineers and the data engineering team has data scientists, who does what?

This sort of decision is largely outside the data science team itself. If you're in an organization set up around projects, your team will likely have multiple roles. These organizations tend to be filled with teams of many different specialties who all closely work together to do projects. If you're in an organization set up around roles, a pure data science team makes more sense. Role-based organizations tend to have departments for types of workers like an engineering department, a data science department, and so on.

Data engineering is a particularly common case for having a non–data science technical person on the team. In some organizations, there are entire data engineering teams dedicated to storing data and making it accessible to other teams, in which case your data science team won't need data engineers. In other organizations, generally smaller ones, there aren't teams dedicated to it, and thus the burden of maintaining databases and keeping them up to date will fall in other places, like possibly your data science team. But it's worth thinking through data engineering in particular because without some sort of data backend, your data scientists cannot work. There are many tales of data scientists being hired at companies only to realize the company doesn't have any data for them to work with. If you choose not to have data engineers on your team, make sure there is an infrastructure in place for your data scientists to get what they need.

Should the Data Scientists on the Team Specialize?

If your data scientists are specialized in particular areas, such as forecasting, experimentation, and optimization, you in theory should be able to accomplish more as a team since you'll have more areas of knowledge covered. On the other hand, if everyone on your team acts as a data science generalist, so that any body of work assigned to the data science team can be done by any member of the team, you should have a much more robust group. A team of generalists will be able to review each other's work better, handle teammates having time off or quitting, and chip in if projects require extra help. Together, this creates a push and pull between having more specialization so the team can accomplish more and more generalizing so the team runs smoothly.

There are also different types of specialization:

- Technical topics like forecasting, experimentation, and optimization
- Domains like fraud, marketing, and logistics
- Particular parts of the company, like a particular dataset

Depending on the organization, there may be almost no risk to certain types of specialization. Having data scientists who are experts

in marketing is probably just fine for a marketing company. But specializing in a particular technical topic or dataset within the business might lead to more situations where there isn't enough work for the data scientist to do.

As a practice, with all other things being equal, generalization is probably better. On a day-to-day basis your data science team will succeed based on how smoothly it runs, not how many obscure techniques the team knows (see Chapter 4 for a more in-depth discussion of this). Further, your data scientists will naturally specialize on their own. Data scientists *love* learning new things, and so if a project requires concepts and methods that data scientists don't know, they'll go learn them. You don't need to structure the whole team around only certain people knowing how to do certain things.

That said, there are situations that require such finesse that you'll want specialists to do them. A classic example of this is optimization and reinforcement learning, which generally isn't included in the standard data science curriculum and takes a while to learn. As a data science team gets larger, you'll find more situations where you feel obligated to organize the team so that some people have roles related to their specialty. But try and avoid this for as long as you can, and if you do hire specialists in a particular field, don't hire just one. A single specialist won't have anyone to bounce their ideas off of and, worse, no one to check that what they are doing is what they say they're doing.

Positioning a Data Science Team in an Organization

In addition to deciding who should be on a data science team, there is also the question of where the data science team should be located within the hierarchy of an organization. Placing a data science team has two competing interests: having the data scientists throughout a company easily communicate with each other, and having the data scientists communicate with their stakeholders. Given these two competing needs, two different philosophies of data science team placement have arisen: data science as a center of excellence or as decentralized teams.

A data science center of excellence is a centralized group of data scientists whose work spans across the entire organization. By having a single centralized data science team, the data scientists can switch between helping different departments within the organization as

the demand for data science work changes. The main downside of a center of excellence is that the data scientists can easily become insular and disconnected from the needs of the organization.

The opposite approach of having one core centralized data science team is to have the data scientists decentralized and spread across the entire organization. With decentralized teams, each different department (marketing, logistics, engineering) has its own data scientists. This allows the data scientists to be embedded with their stakeholders and gain a deep understanding of the problems they are trying to solve.

The downside of decentralized teams is that you can have incredible amounts of siloing. Different data science teams will use different analytics methods, different technology stacks and programming languages, and even different definitions of the same thing. For example, it can be really difficult to run a company if one half of the company reports at the monthly level and the other half reports at the weekly level!

As you can see, there are countless ways to organize data science teams. There is no universally correct way to do so; it very much depends on the particulars of the company environment. Some situations may call for a single large team of specialists serving the entire company, while others require data scientists to be spread throughout. The data science teams may be large and filled with many roles like managers, project managers, and data engineers, or they may be small, with just a few data scientists and a technical leader. The role of a good data science leader is to understand the context of the organization and make the best decision for the team within it.

With a well-designed data science team, the data scientists should be ready to handle the work from stakeholders. Unfortunately, for most data science teams, the volume of demands put on them for deliverable work far exceeds the capacity of the team. In the next chapter, we'll discuss how a data science team can manage their workflow and strategically think through what types of work to do.

CHAPTER 2
Managing the Work

Data science teams are a pretty busy bunch. There are constant requests for new reports, new models, and past analyses to be redone with new data. For most data science leaders, this means having to make thoughtful decisions around what to actually work on and what to deprioritize or not do entirely. It also makes it very important to manage how long tasks are taking because if one particular task takes longer than expected, something else has to be pushed back.

To be able to successfully manage the workload of your team, you'll first need to have a very clear view on what the goal of your team even is. Are you helping make the product better with machine learning, providing strategy advice with data to a particular department, or something else? You'll then need a clear project management process for how to keep track of the work, and while data science is similar to software engineering, some software engineering principles, such as Agile, don't always directly map to data science. You'll also need to have clear communication with your stakeholders so that, as new developments arise, everyone has the information they need.

The task of managing the work of a data science team must be a concern for data science leaders on teams of all sizes. On smaller teams, the leader may do the project management tasks directly. On larger teams, there will be a project manager who is in charge of managing the tasks. But as the data science leader, you'll still be responsible for the work actually getting finished, and when

something goes wrong, it will be on you to fix it. So this chapter lays out how to think about managing the tasks of a data science group and best practices for keeping your team running smoothly.

What Is the Goal of Your Team?

Before you can effectively prioritize what to work on, you need to have a strong understanding of what the goal of your team is. Since you'll be having lots of requests coming in from many directions, you'll want to focus on what aligns with your goals and discard the rest. The goal of a data science team generally falls into one of three categories:

Help the organization use data to make decisions
: When this is the team's goal, it'll be doing lots of analyses of data and presenting them to stakeholders.

Use data as part of the product to make it better
: These data science teams help train machine learning models to put into production.

Keep data within the organization flowing
: To meet this goal, the data science team helps move data from databases into reports, dashboards, and other locations where the business can use it.

Each one of these goals requires a very different type of work: decision science, machine learning, and data engineering, respectively. The best-case scenario for a data science team is that they focus on exactly one of these goals and one particular part of the organization to help with (like marketing). In practice, you'll find your team often being drawn to do multiple goals, either from external stakeholders with unmet needs or from your own desire to have the team have more influence and bring data to new places. But unless your data science team is massive, with many data scientists and managers within it, it is unlikely you'll be able to do it all.

As a leader, it's your responsibility to keep the team focused on a very specific goal and deflect distractions from this goal. By effectively saying no, you avoid overwhelming data scientists with different types of work they'd be unable to effectively complete.

Prioritizing Data Science Work

As a data science leader, it's often difficult to balance what to work on, not only because not all of the work aligns with your goals, but also because each task can have very different levels of effort, scope, and stakeholders it helps. A helpful framework to consider when prioritizing each task is to consider these three questions:

How much could the result of this work influence the business?
: Some tasks could dramatically alter the business, like adding machine learning to a product so that customers can interact with the product in a new way. Some tasks will likely not affect the business much. Even if you absolutely ace a forecast for revenue from a marketing channel, having that forecast probably won't change much of your business. Interestingly, this question of how much value will come from the work isn't a data science question—it doesn't have to do with how you approach the task technically. Instead, thinking about value will require you, the data science leader, to possibly step outside your comfort zone and think from a product perspective.

How likely is the data science team able to pull this off?
: A recurring theme throughout this report is that data science projects are risky and fail in lots of ways. Before starting a project, it's critically important that you as the data science leader take a guess as to how likely it is to succeed. This ideally would be done in collaboration with a principal data scientist. Having a reasonably thought-out idea of how well a project would work will help you decide if it's worth doing. There can be a really wide range of risks for different data science projects: updating an existing report probably has a high chance of succeeding, whereas building an entirely new machine learning model on new data to use in a new product has much less of a chance.

How much will this bring your data scientists joy?
: As a data science leader, you are in it for the long game, and some tasks are more enjoyable for data scientists than others (it's generally more fun to build a new model with an all-new method than retrain the same model for the hundredth time). In a perfect world, the tasks your team is working on align with the particular type of work the data scientists were specifically hired to do and with their particular interests. Not everything

is going to be a data scientist's dream to do, but if none of the work is fulfilling to them, you risk attrition.

See Table 2-1 for some examples of how different projects might score. Interestingly, these three questions are often highly correlated (or inversely correlated). There is often an inverse correlation between how much something could influence the business and how likely it is to be pulled off. For example, lots of stakeholders come to data science teams with a pipe dream project (we use machine learning to predict what flavor of ice cream a customer will order before they walk into the store!) that could be valuable but probably won't work. The tasks that are very likely to work probably don't have value. This often has to be the case because most tasks that would be straightforward to do and have a huge impact have already been tried!

Table 2-1. Examples of data science projects and what to consider when prioritizing

Project	Influence on business	Chance of pulling off	Data science team joy
Build a new complex anomaly detection model to notice any tiny trends in customer behavior the moment they arise	High: could reduce customer support costs	Low: anomaly detection models with minute signals are often too noisy to work	High: building a complex model can be interesting and challenging
Automate a manual report the business uses into a dashboard	Low: since the report already exists, the primary value is savings in employee time	High: with data already existing in the report, there is low risk the dashboard won't work	Low: requires redoing existing work and has the long-term cost maintaining it
Add natural language processing to an existing part of the product so customer input can be categorized	Medium: categorizing input can be helpful but doesn't immediately alter the entire product	Medium: NLP categorization models often work well if you can successfully devise a relevant set of categories	Medium: requires developing new models, although fairly straightforward ones

Similarly, often the totally novel projects are the most interesting for data scientists but also have the least chance of succeeding. Working on something totally new that is so interesting you could make a conference talk on it later is fun! It's probably not good for the business to only try moonshot projects, though.

So, as your tasks come in, try to choose the ones that check off as many of these boxes as possible, and avoid going for too long without tasks that provide coverage of helping move the business forward, have a good chance of succeeding, and make your employees happy.

Managing the Queue of Work

As tasks come in, you'll want to be thoughtfully keeping track of the queue of work and what's being handled at the moment. If you're lucky, you may have a person on your team fully dedicated to managing this queue—the project manager from Chapter 1. Otherwise, this will likely fall on you as the team leader.

It's likely that your organization already has selected software such as Jira, ClickUp, or Trello for project management tasks. These tools keep track of what is being worked on, what will be worked on, and how tasks have been completed. If your organization doesn't have a preset tool, you should quickly pick one and get your whole team used to reviewing and updating the board. While it may be the project manager's (or your) responsibility to maintain, you'll still need the whole team to be editing the status of each task as they work on it. It's common for data science teams to have problems with individual data scientists getting distracted and not putting effort into maintaining updated information in the project management tools. However, you can create a culture of taking the tools seriously. The tools should be updated every time a new task is started, completed, or hits a roadblock. It's even better if discussion around why tasks are started, ended, and so on are all kept on the board.

Your data scientists will also be getting tasks that you might not know about, like when stakeholders make direct last-minute requests for a new analysis. They may also be getting tasks to fix issues with previously delivered work, like bugs found in production models. If you've created a culture of staying organized with project management tools, then the tool will be a record of what is worked on (and why other tasks aren't worked on). This can be valuable for stakeholders asking questions about why tasks haven't been done and will provide security for your data scientists because they have receipts of all their work.

Agile and Scrum

Software engineers have been lucky to have the project management concepts of Agile and Scrum development. These frameworks have made a huge impact on software development because they've let engineering teams iterate more quickly and focus more on delivering value immediately. In Agile and Scrum frameworks, a project is split up into short sprints, each covering a few weeks of time. In each sprint, the work is split into discrete tasks, each with a clear end point. The engineers work through the tasks, and at the end of the sprint, new tasks are assigned. As a data science leader, you may find yourself thinking, "Agile works well for software development, so why can't I use it for data science?" Unfortunately, to date, there hasn't been much success in directly applying Agile to data science.

First, Agile methods require being able to break work into small chunks and complete them sequentially. That's extremely hard to do in data science because there aren't clear boundaries of tasks. For example, you can't split a project into data cleaning, feature selection, model training, and validation because those don't happen sequentially (instead, the data scientist bounces between them). Compared to software engineering, work is much more varied in length, and there is more uncertainty in terms of how long something will take. Also, depending on the data science team, it might be much harder for work to be passed between different data scientists, thus making it harder to find value by splitting tasks into small chunks in sprints. We're not saying Agile *can't* work for data science teams, just that you'll have to make a lot of adjustments to the methodology. To read more on this topic, see the blog post by Sophia Yang at Anaconda, "Don't Make Data Scientists Do Scrum".

Communicating with Stakeholders

Part of managing the work coming from stakeholders is managing the relationships with the stakeholders themselves. A data science team often has connections to many other parts of the company: the engineers who implement the data science models, the marketers who use data to run campaigns, the data administrators who set up the databases for the team, and more. A person in charge of a data science team has to ensure that these people are happy (more or less) with the performance of the team.

When you're in charge of a data science team, stakeholders will be coming to you with all sorts of requests, information, and demands. Not only is it important to effectively prioritize these, as discussed previously, but you're also in charge of making sure that the data scientists on the team are getting the appropriate information. On a successful data science team, stakeholders will feel directly connected to the data scientists and feel comfortable going to you when there are issues, just like the data scientists go to you when there are issues with stakeholders.

If there isn't healthy communication between the stakeholders and the data scientists, there are a few possible scenarios. First, the stakeholders may decide not to talk to the data scientists on the team and instead directly talk to you, the leader. As a leader, this might actually feel like a good situation—constantly getting to hear from stakeholders to understand their wants and needs. But you can become a bottleneck so that only you have information that the whole team needs.

Take, for example, a situation where a stakeholder is in need of a machine learning model to predict which customers are likely to unsubscribe from the subscription service. The stakeholder probably has very precise requirements for the model—how accurate it needs to be, what types of customers it needs to work on, and things like that. Those requirements then have to be converted into data science concepts; for example, "must always give customers the right answer" converts to a particular accuracy metric. If the stakeholder only tells you, the leader, about them, then you will have a hard time trying to explain the idea to your team and answer any particular questions the team has. In this case, you're a bottleneck for the information or, to use an alternative metaphor, you're a connection in a game of telephone, with your team being the last person.

This leads to another scenario of communication issues on a data science team. If the data scientists or stakeholders feel like you're a bottleneck, they might go around you and keep you out of the loop. In the preceding example, if the stakeholder and data scientists realize it's easier to talk directly, they may start to do so, and you may find decisions being made without being informed. While empowering data scientists is generally good, as the leader, you have the best perspective on how the overall team should be functioning, and you need to be aware of what's going on. Both of these scenarios are shown in Figure 2-1.

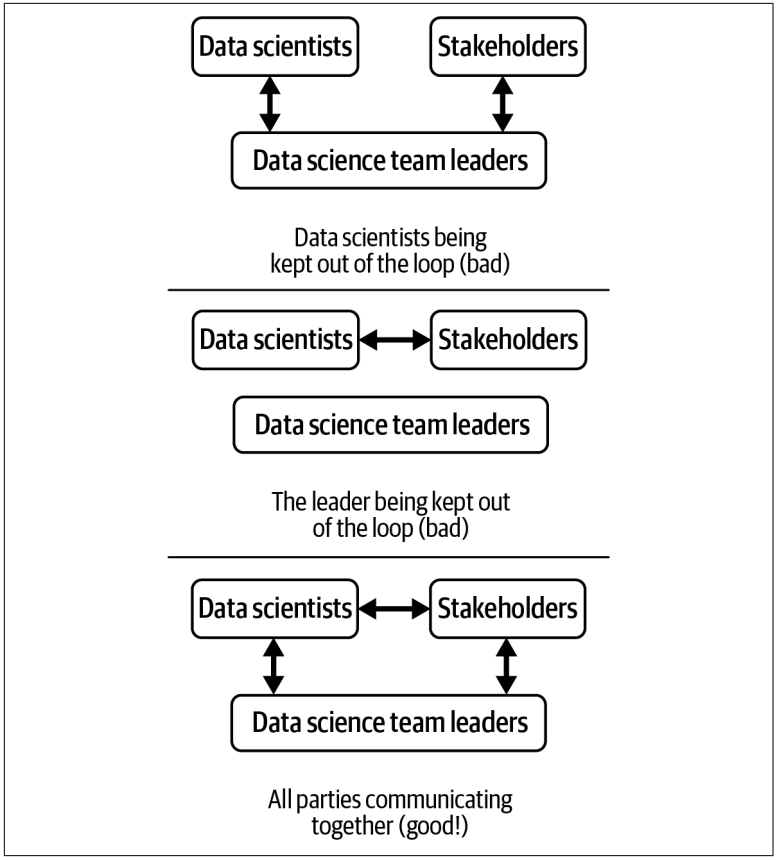


Figure 2-1. Some scenarios of how communication with a stakeholder can play out.

So as you work with stakeholders and fulfill their demands, make sure communication is healthy between yourself, your team, and your stakeholders, all working together. When you notice people are missing information they should have (and you might have), put effort into trying to foster your relationships with the particular stakeholder and with your teammates working with them.

As you develop your methods for managing the team's workload, you'll want to make sure that your data scientists are producing value and successfully doing the tasks you assign. The next chapter of this report goes into detail on how to align your data scientists with the work you need to do and best practices for leading them to do the work.

CHAPTER 3
Running a Team That Provides Value

This is almost painfully stale to say, but the success of your data science team entirely depends on how well the people on it can fulfill their roles. That your team members have clear, achievable goals and the resources to execute them matters more than what technology you choose to execute your code on, or the types of models you choose to train. Data scientists have all sorts of different backgrounds and differences in how they think and what they want to do. As a leader, it's your job to create an environment where the data scientists can succeed. There are thousands of books out in the world on how to manage people, so instead of purely providing guidance around management broadly, this chapter will focus on the specifics of leading data scientists.

As a data science leader, you want your data scientists to be contributing value to your team and your organization. Value is a difficult term to define with data science because it can be so abstract and disconnected from the amount of work put in. A summary graph of data that takes five minutes to make one day might end up being more valuable to an organization than a complex machine learning model that a whole team spends months on. Because of that, let's explicitly define *contributing value*.

A data scientist is contributing value to the team if they are creating and finishing code, reports, or other deliverables that:

- Solve the requested task
- Are straightforward to maintain
- Are quick to develop

In doing so, the data scientists should:

- Communicate issues and concerns as they arrive
- Promote a healthy working environment

There is a lot baked into that definition of contributing, but it's reasonably holistic: any data scientist who is meeting those criteria is most likely succeeding at their job. But that definition is surprisingly difficult to achieve! In this chapter, we'll break it down into its core components and discuss what you as a data science leader can do to help your team contribute value.

It's Hard to Solve a Requested Task

Usually, a data scientist is given an abstract business objective, such as "use data to find out why new customer growth is down" or "use machine learning to predict if a customer will use a coupon." These are business objectives in that completing the tasks will help the business, but they don't provide clear data science steps to achieve them. In the first example, there may be many different ways to define "new customer growth"—you could use the number of people who make a new account or, alternatively, a first purchase. There are even more ways to run analyses on that data. It's not obvious for a data scientist what approach is the best, and different types of analyses have trade-offs for the business. For example, a time series approach to analyzing new customer growth might provide a more accurate understanding of when new customer growth dropped but less of an understanding of what drove the reduction in new customer growth.

Without the context of the whole business, it is difficult for a data scientist to understand how to turn that vague business request into a solvable data science problem, which, in turn, provides an answer back to the business. In the second example, the case of a data scientist being told to "use machine learning to predict if a customer

will use a coupon," it may not even be the case that machine learning *could* effectively be used. It's possible that the data a company has on a customer, such as past purchases and demographics, does not predict whether or not they would redeem a coupon. Or it could be that the data is spread over too many different IT systems to make creating training data possible. In these situations, it's not the ambiguity of the request that is difficult, but the request itself may be unsolvable.

There are *many* reasons why a data scientist might not be able to fulfill a request as given, including:

- The stakeholders don't understand their own needs, and so the request is ill-defined.
- The stakeholder's request is well-defined but can't be solved with data.
- The request can be solved with data, but the data isn't available at the organization.
- The data is too noisy to be useful.
- The answer the data provides goes against the political tides of the organization, and sharing it with the organization could be disastrous.

As a leader, it is your job to help resolve these situations. You are the person who bridges the gap between what a request is loosely asking for and what a data scientist can actually achieve. You should ensure the data scientists have the best understanding of what the stakeholder needs truly are, and you should have a perspective on what the data science team is capable of handling.

There are a few particular things you can do when unclear or unsolvable tasks come in. In situations where stakeholder tasks are vague or poorly defined, you can work directly with the stakeholder to better define the task while keeping the data scientist in the loop. A leader of a team is generally better suited to these conversations than an individual contributor because leaders have a better understanding of the overall goals of the organization and where adjustments can be made. Make sure you also include the data scientists who are doing the work in the discussion, because the last thing you want to do is negotiate the tasks with a stakeholder only to find out your proposed solution is still too vague for the data scientists.

Similarly, in situations where the task is well-defined but the data scientists find themselves unable to do it because the data isn't available, is too noisy, or doesn't have a signal for a model, you can then work with the data scientists to try and brainstorm alternate solutions. Sometimes a leader talking to data scientists helps you decide that the other datasets are good enough, that alternate modeling approaches would work, or even that you could entirely bypass the task by doing some other type of data science (like switching from using a machine learning model to a simple rules-based method). Your perspective can be helpful here because you can enable the data scientists to think outside their current perspective.

By helping the data scientists quickly navigate these situations and decide what the right solution is, you can avoid doing that doesn't provide value.

It's Hard to Make Deliverables Straightforward to Maintain

When a data scientist creates an analysis, a machine learning model, a dashboard, a code package, or pretty much anything, they are creating a deliverable. As part of the work, there is an understanding that the deliverable will exist beyond the moment it's created. Sometimes this is explicit: a dashboard is expected to be continuously viewed, and a machine learning model is expected to be rerun with similar accuracy. Sometimes this is implicit: an analysis may have been done as a one-off request for an executive, but the executive may periodically review the results or even ask for it to be updated with new data.

Thus, the deliverable itself has to be maintained (the dashboard always has to be up and running, and the model has to be consistently accurate), and the method of creating the deliverable has to be maintained (the code for an analysis needs to be saved in case people have questions about how it was run). It is the responsibility of a data scientist to ensure that the deliverable they create is capable of being maintained and adjusted by people in the future (which could include, but is not limited to, data scientists themselves).

There are a lot of straightforward steps to help with this, such as writing good documentation, saving code in shared places with version control rather than just on the data scientist's laptop, and

creating a centralized directory of where different deliverables are located. These particular examples are all things that take a relatively small amount of time to do and can be done without changing how the deliverable was created. Yet often they are the first places where a data scientist might cut corners to get a deliverable out the door faster. As a leader, it's your job to help the data scientist balance the need to get something done quickly and the need to make it maintainable.

There are also a lot of very important steps to making a deliverable maintainable that aren't simple to achieve. For example, writing code that has a logical structure to it with segments that each encapsulate a small part of the work can make it much easier for future data scientists to maintain. Modular code also makes it easier for the work to be reused in other situations. The code should also be given clear logging and output so that it's obvious what is occurring when it runs. Writing clear and logically structured code is a skill that takes years of practice, so this isn't easy to achieve. A mandate from a data science leader to "write better code" won't help data scientists; instead, that requires technical mentorship and a culture where data scientists review each other's code and provide meaningful feedback.

Another way in which a deliverable can be hard to maintain is from the technical tools the data scientist used to create it. A cluster of GPUs training a complex neural network may provide a more accurate model than a simple logistic regression run on a laptop, but recreating that cluster of GPUs for training the neural network will be far harder for the next data scientist than just importing a regression library. It is the responsibility of the data scientist to balance using the most powerful (and interesting!) tools with the tools that can deliver a result in a simpler way. Unfortunately, a data scientist often doesn't have the full context of why a model is being made, and it is challenging to understand what would be best for the business, especially if that data scientist is on Twitter reading about cool new advanced techniques. As a data science leader, it's your responsibility to guide data scientists toward solutions that are maintainable and to balance having the most advanced solution with the most maintainable one.

For better or worse, the person who is going to be hurt the most by unmaintainable deliverables is you, the data science leader. If data scientists are not putting effort into making sure their code and output is easy to maintain, over time you'll find your team's output

getting slower and slower. Requests that should be straightforward, such as rerunning a past analysis but with new data, will take far longer than you expect if most previous work has to be redone entirely. Unmaintainable code and deliverables are a form of technical debt, and as a leader, you need to keep an eye on it.

It's Hard to Make Deliverables Quickly

Compared to other fields, such as software engineering, it's hard to tell in data science when something is considered done. In most cases, there is no clear definition of what "done" is: features in a model can always be further adjusted, analyses can slice data in new ways, and there is always another possible machine learning framework to try. That said, for data science work to be useful, it has to be delivered to a stakeholder to use it, and sooner is better. A data scientist has to be responsible for balancing the speed at which something gets done with how well that task is completed.

This is made worse by the earlier point that the desired outcome from the stakeholder may not be clear and achievable. If a data scientist is tasked with making a model to predict if a customer will use a coupon and every model the data scientist has tried to make hasn't worked, how can the data scientist know if one more try would work or if no model would ever work on that data? The work of a data scientist is to constantly try approaches until one works, but they have no way of knowing if any will ever work.

So a data scientist has to balance "let's try more things" with "this is good enough and let's deliver quickly" without knowledge of whether or not more things would work better or even what good enough is. That's really hard!

There are a few core lessons that data scientists learn that can help them resolve this tension and actually deliver results in a reasonable time frame. First, simple techniques such as straightforward regressions are far faster to iterate on and can get a result out the door quickly. Further, if a simple technique like a regression doesn't work, it's unlikely that a complex method (like a neural network trained on a cluster of GPUs) would work instead. Thus, a best practice for a data scientist is to start with simple techniques and, if those work, deliver it to a stakeholder; if they don't, then make the call on trying a more complex method. As a bonus, a simple method is easier to maintain. Note that there are some situations where more complex

approaches are required, such as image processing, in which case having a strong technical stack for your team can help deliver results faster (see Chapter 4).

The next lesson, which we've already discussed, is that it's easier to get data science done quickly if you have a clear understanding of the goal. If data scientists don't have a clear understanding of what the goal of the work is, then they can spend weeks slicing and dicing data looking for any insights or models to come up with. The clearer the business objective is to the data scientist and what their goal is, the more likely the data scientist will stay on a timeline and not spend cycles doing work that doesn't have value.

Another lesson is that the more communication there is, the easier it is to deliver on a timeline. If the data science work is taking more time than expected, a conversation with stakeholders about what that means for the timeline can help the stakeholders decide whether the work should continue or change. This avoids scenarios where stakeholders are blindsided by how long things are taking or feel left out of the loop.

NOTE

This section in particular relates to the concept of a minimum viable product (MVP)—the simplest thing that completes that requested task. By creating an MVP, you can get something out the door quickly and iterate based on the feedback it receives. A good data science deliverable should be a form of an MVP: it should do what was requested without too much more. This is important because of just how easy it is to add scope when doing data science work and to try and squeeze in new analyses or attributes of how a model works.

As a data science leader, it is your job to ensure that data scientists are delivering their work according to the timeline of the project. That means communicating with the data scientists and stakeholders when risks to the timeline arise, as well as helping to make decisions about where to invest time on the project versus deciding to stop working on parts.

Practically, a data science leader should be constantly monitoring how work is progressing. This can be quite hard due to the ephemeral nature of data science projects. For example, you may have

a situation where each morning you ask a data scientist how the work is progressing, and each morning they say, "I'm still selecting features for the model." It's hard to know at what point that goes from an acceptable amount of feature selection to a sign that the project won't be completed in time. Given these sorts of situations, a leader can help by working with the data scientists to help them fully grasp the timeline so they themselves can know when things seem like they're going too slow. As a leader, you have to find a balance where the data scientists feel like they understand the amount of time available and your expectations of them without feeling like you are telling them how to do their jobs. This is largely a function of trust built from continuously communicating with each other.

Data Scientists Should Communicate Issues and Concerns

For a data science team to succeed, the data scientists need to be able to communicate when situations arise. Issues crop up all the time on data science projects—from small issues like libraries being incompatible with each other to major ones like essential data missing. A data scientist needs to be able to effectively flag these situations and let the stakeholders and leadership know. Similarly, if someone at a leadership level or a stakeholder sees a problem arising, they should effectively be able to communicate that.

Data scientists and stakeholders often run into trouble communicating issues because of people's tendency to "step out of their lane." A stakeholder can be dangerous for a project if they feel like they should be providing technical guidance, like asking the data science team to use a specific type of model. The data scientists are almost always the experts on the technical solution, and an overly specific stakeholder making technical requests limits what the data scientist can provide as a solution. On the other hand, data scientists can be overly specific with their opinions on how the business should be run. The data scientists should be providing technical expertise and general recommendations for the business, but ultimately it's the stakeholders' responsibility to choose what to do with it. The data scientists and stakeholders need to work together and address issues as a team rather than try and have one group take control.

This problem occurs when stakeholders dismiss the concerns of data scientists and vice versa. A stakeholder can easily dismiss a

data scientist's concern that the stakeholder's ideas for a data science product wouldn't be feasible because the math or data wouldn't work. On the other hand, a data science team could spend months building a complex model without checking with a stakeholder to see if the business would have a use for it.

The point of all this is, regardless of how technically skilled your data science team is, their ability to communicate any issues that arise with the stakeholders involved is critical to providing work of value. As a leader, fostering healthy communication can be extremely challenging. You'll need to show to the data scientists that when issues are brought up, they won't be negatively reprimanded, and you need to create a culture where team members feel psychologically safe. One good example of how to create such a culture is with blameless postmortems; that is, meetings after something bad happened where you can openly talk about the issues without blaming people. Another helpful method is to show vulnerability yourself—if you as a leader show vulnerability, then others will often start to show vulnerability too.

Promote a Healthy Working Environment

It's not enough for a data scientist to be delivering consistently successful work; they also need to help create a culture that lets their teammates thrive and encourages their stakeholders to work with them. Without a healthy working environment and a team-centric culture, even a perfect machine learning model or an outstanding analysis won't have the ecosystem around it to be useful. This means:

Data science teammates should support each other
: The data scientists should be working together on projects, reviewing each other's code, and listening to feedback. The senior data scientists should be taking time to mentor the junior ones. At the same time, the senior data scientists should also still be open to feedback and listen to what the junior data scientists have to say.

Data scientists should talk to you
: Data scientists should be able to express concerns and opportunities to their leaders and have their concerns heard. While being a leader in data science at an organization can give you an opportunity to see the bigger picture and grants you more

context than an individual contributor, each data scientist does the actual work and can see details that you might miss.

It's the responsibility of the data scientist to help contribute to the overall team's success and not just their own. It is your responsibility as a data science leader to give team members the bandwidth for this and not force them to be so focused on individual deliverables that these tasks go unfulfilled. It is also your responsibility to manage external stakeholders and people outside the data science team with the leverage you have from your leadership position.

Putting It All Together

So, as you can see, a lot of distinct factors go into having your data science team provide actual value. As a data science leader, keeping an eye on all the distinct components in delivering value and providing your team with the support they need is likely the majority of your job on a day-to-day basis! As a leader, you have the ability to create a culture that keeps stakeholders in the loop and promotes creating valuable deliverables with minimal tech debt in a way that lets the data scientists feel they can communicate. The best way to achieve this is through diligent eyes on the team, noticing when data scientists are struggling and thriving, and stepping in where appropriate. If you find that your data science team currently has a culture of ignoring these principles and instead has one of "just do exactly what you're told and stay quiet," it's still possible to turn it around by setting an example yourself.

With a data science team that has a strong culture and the ability to get work done, the next step to think about is how to manage that team's relationship with its stakeholders. In the next chapter, we'll discuss how to work with many different types of stakeholders as a data science leader.

CHAPTER 4

Choosing the Technical Infrastructure

So your team is well placed in the organization and has the right people on it. You are working to create a strong culture of providing value, and you have methods for managing the tasks. But those tasks require your data scientists to be doing technical work, which means they need the tools to do so. These include questions like what programming languages to use (R, Python, or something else); what kinds of databases to store information in; if models should be deployed as APIs by the engineering team or as batch scripts run by the data science team; and more. A data science leader has to be heavily involved in choosing which technology a team should use and deciding when it's time to switch between them.

How to Make Decisions on Technical Infrastructure

The actual process for making decisions about a team's technical infrastructure is as important as the actual decisions. The leader of the data science team may be making these decisions directly, or it may be a person on the team like the technical lead or a principal data scientist who has the final call. With any of these sorts of decisions, there is a spectrum of options for how the decision is made (see Figure 4-1). On one side of the spectrum is authoritarianism, the idea that all decisions within the team are solely made by the

person in charge. On the other side is anarchy, the idea that anyone on the team can make whatever decision they personally feel is best.

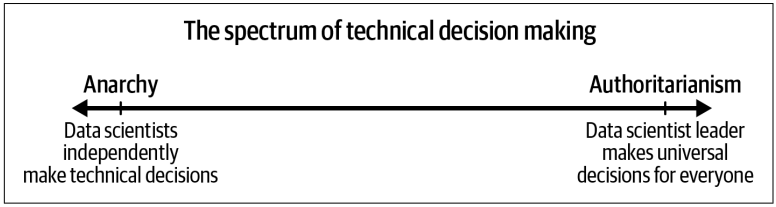


Figure 4-1. The decision-making spectrum on which each data science team falls.

Data scientists usually love to make decisions, because they'll choose what they personally like best. In anarchy environments, the data scientist will choose precisely the tools they prefer, so you may end up with a data science team of five people and a code base written in six programming languages (and two of the languages are dead). Each data scientist will be individually happy with their tool set, but the data scientists won't be able to work together. If one code base uses Python on an AWS EC2 instance and another uses MATLAB on a Windows laptop, you'll be in trouble when data scientists switch projects or leave the team. But so long as each data scientist is only working on their own projects, they'll be happy enough.

On the other end, in authoritarian environments all decisions are made at the leadership level. The programming language, platform, architecture, and other technical decisions are all made by leaders who keep things uniform across the team. For example, everyone uses R, commits to a very specifically formatted GitHub repository, writes their code in the exact same style, and does projects in the same way. In these situations, there are few issues with the data science team changing. Everyone can work on everyone else's projects, and there is less infrastructure to maintain because everything is consistent.

The problem is that regardless of what team-wide tools the leader chooses, there will be situations where they don't work well. Sometimes something is hard in R but easy in Python, or hard when you're doing your data science on virtual machines but easy when you are using Docker. In those situations, if you don't allow your data scientists to use the right tool for the situation, they could end up spending far more time and effort on the task at hand. Worse, they'll quickly get demoralized and may leave the team. See

Table 4-1 for some examples of technical decision making along this spectrum.

Table 4-1. Example scenarios of technical decision making

Scenario	Style
Data scientists are given laptops and can use whatever tool set and programming language they want so long as they finish the assigned work well.	Extreme anarchy
Data scientists are allowed to use the programming language and library they want on their laptops, but are encouraged to use tools that align with the existing cloud platform whenever possible.	Leans toward anarchy
Data scientists are required to use cloud infrastructure, Python, and a specific set of ML libraries. Other tooling can be used, but they first need approval of the technical lead.	Leans toward authoritarianism
Data scientists must only use exactly the tool set allowed by the technical leader, the manager, and DevOps.	Extreme authoritarianism

So both ends of the spectrum, anarchy and authoritarianism, are not good places for data science teams to be. The healthy spot is somewhere in the middle, where the data science leader works in collaboration with the data scientists to ensure the right tools are being used for the job while minimizing the number of distinct tools and incompatible systems being used. Where exactly the best spot falls depends on the particular organization and its objectives. It also largely depends on the particular industry. For example, consulting firms doing many one-off projects should have fewer rigid structures, whereas heavily regulated industries like finance should have more guardrails.

It's easy to look at this spectrum as a data science leader and point to a place you want to be. And as the leader, it is easier for you to manage with more authority, so you'll likely want to have your team lightly authoritarian. But it's worth reflecting on the strengths and weaknesses of the particular data scientists on your team and giving them as much autonomy as you can.

Components of Data Science Team Infrastructure

Data science teams require lots of different infrastructure systems, some explicitly built or purchased for them, like a database solution, some implicitly decided by the team, like setting up a Git repository

with a particular programming language and set of libraries. This section covers several important areas of infrastructure to consider.

Storing the Data

Most data science teams don't need to worry about where to store raw data because they get it from other parts of the organization. Other divisions create data with marketing information, sales and revenue, and data directly from the product, and engineers store it in databases for data scientists to use. A data science team may be able to influence some of the decisions on how the data is stored, like asking for certain columns to be included, but rarely are they responsible for it.

However, data science teams do have a need to store intermediate or output data—data that has been cleaned or outputted from a model and is under the ownership of the data science team itself. An example of intermediate data is a large data table after the string columns have been formatted or new features to be added that will get fed into a model. An example of output data is predictions from a model for each customer in a dataset. These sorts of datasets are tricky because they often don't have a fixed schema and can be very large in size. Most often the data science team doesn't have ownership of the data servers, so they may not be able to acquire a storage location themselves and need another team's help.

In an ideal scenario, the data science team's intermediate and output data would be stored closely to the input data. If all of the data is stored in a single location, it's easy to join it together for further analysis and keep track of changes. In practice, this is sometimes not possible; for instance, if the input data is production data that needs to live on secure servers that the data scientists cannot write to. In these situations, you'll have to set up a different location to store your data and create processes for managing it.

As the creators of the intermediate and output data, you'll be responsible for keeping track of it. This is where data governance practices are important. You'll want a structure for deciding what data to store and how to consistently store it. You'll want to do this in a way that, in the future, people will be able to understand what was done. A full data engineering team may make a data warehouse or a data mart for their data, but since that isn't your primary focus, you will likely not need to go that far.

As a team leader, you need to put thought into the best way to store this data and how you'll be keeping track of it over time. If you don't think it through, then the data might end up spread out over many locations, such as multiple database servers, shared network drives, and file storage systems, and you'll be unable to keep track of it. Worse, over time, other systems will start relying on this data—for example, processes that use customer predictions from a model to then adjust email campaigns—and if the data isn't being stored correctly, then you can incur tech debt trying to use it.

A Workspace for Running Analyses and Training Models

The day-to-day job of a data scientist mostly involves cleaning data, making analyses, training models, and other forms of work that all happen in a single place. These data science workplaces take different forms, depending on the setup of the data science team:

Each data scientist works on a company-owned laptop
: For many companies, the data scientists do their day-to-day work on a company-owned laptop using an IDE of their choice, like RStudio or JupyterLab. A data scientist will download data to this machine, do their work, then upload their results to a shared location. Using laptops has the benefit that they require almost no collaborative setup—each data scientist can independently install whatever they want on a machine and use that. They have the problem of having no standardization: anyone can do whatever they want, so it's harder for code that runs for one data scientist to run on a different teammate's laptop. Because each machine is set up differently, often more senior employees on the team have to help junior ones when their unusual setups cause things to break. The data scientists are also limited in hardware by the specifications of the machine, so if a particular analysis requires something different, that analysis can't be done. There is also the security risk that laptops can be physically stolen.

Data scientists work on virtual machines in the cloud
: Some data science teams improve on the first scenario by replacing laptops with virtual machines like AWS EC2 instance or Google Cloud Platform VMs. This allows data scientists to change the instance size if they have different hardware needs and removes the chance of a laptop getting stolen. The down-

side is that there still can be a total lack of standardization of the machines, so just like with laptops, something that works on one virtual machine might not run on another. There also is a security risk of virtual machines being open outside of the network: because each data scientist sets up their machine, there is a decent chance one might set it up incorrectly.

Shared cloud workplace platforms
: Recently, data science teams have been adopting cloud platforms that are tailored to data scientists. These platforms, like AWS SageMaker, Saturn Cloud, and DataBricks, are meant to provide a location where data scientists can do all of their work. By using a standard platform, data science code is more easily passed between different teammates, less time is spent on setup and upkeep of the workspace, and code can often more easily be deployed. They also have fewer security risks because there are administrative tools built in for oversight. Each platform has its own strengths and weaknesses, so if you are considering one of these platforms, it's worth having your data scientists try them and see which they like.

Note that some data science teams have datasets that are so large they can't feasibly be analyzed on a single machine. In these situations, a separate technology has to be used to run the computations across a distributed cluster of machines. The cluster must be connected to the data science workspace so the team can take results and analyze them further. Spark is a popular technology for these computations, and the DataBricks platform has Spark built in. Dask is a more recent Python-based framework for distributed computing—it's built into the Saturn Cloud platform or can be used on its own by the service provided by Coiled. That said, for most data science teams, there isn't a need to use distributed computing. Often datasets are small enough to use a single machine, or you can run things on a single large virtual machine if needed. The overhead of maintaining a distributed system can be a large burden if your team doesn't need it.

Sharing Reports and Analyses

If your team is focused on using data to help drive business strategy, you'll be creating lots of reports and analyses. If your team is focused more on creating machine learning models, you'll still need analyses to inform which models to use and way. There is almost no

situation where your team isn't creating information that needs to be saved and shared with others, and thus you'll want infrastructure to support that. You'll also want the ability to connect an analysis with the code that generated it in case you need to rerun it.

If you don't explicitly choose a method for storing and sharing analyses, then your "infrastructure" will end up being whatever emails and Slack messages are used to share the information. This is very difficult to maintain in practice. While it's easy to share results with others in this manner, there is almost no way to find an older analysis or trace the code that made it.

A more sophisticated approach is to create a shared location to save your analyses, such as an AWS S3 bucket, a Dropbox folder, or potentially a GitHub repository. In these approaches, the data science team has to be vigilant about enforcing a standard structure so that particular analyses can be found in the shared location and traced back to the code that made them. Ideally, the results should be visible to both data scientists and non–data scientists alike. Tools like Dropbox folders are inherently easier for nontechnical people to navigate than an AWS S3 bucket or something that requires technical knowledge to view. Regardless of the approach, you'll still want to have data governance policies so that these are effectively organized.

Projects like Knowledge Repo, an open source tool by Airbnb, or RStudio Connect, a platform for sharing items like R Markdown reports and R Shiny dashboards, are being built to solve this problem. By providing a platform where an analysis can be easily uploaded and directly viewed and the code that made the analysis can also be stored, data science teams are more capable of cataloging work and keeping them maintained over time.

Deploying Code

If your data science team is creating code that is consistently run, either on a batch schedule or continuously as an API, then you'll need a platform that the code can be run on. There are generally two possible scenarios that your team might fall into. In one case, there is a supporting engineering team that maintains the code, and in the other, your data science team is all on its own:

You have the support of an engineering team

If your data science work is being built directly into a product, then you likely have an engineering team to help you out. The engineering team is in charge of connecting your models and work to the product; they are the ones who call your APIs or use the output of your batch scripts. Because of this, they almost always already have their own platform setup for deploying all of the software engineering code, and the best thing to do is have the data science code merge in. Your data science team doesn't have to worry about maintaining a platform but instead just needs to hand over Docker containers, Python libraries, or some standard format that the code can be run from. Your team is, however, on the hook for making sure the code is up to standards, and as the leader of the team, you should be checking that the data scientists are adhering to those standards.

Your data science team is on its own

There are many data science teams that aren't directly connected to an engineering group, such as data science teams that generate insights for a business unit. There are still situations where teams like these may want to deploy code. For instance, they may want to score each customer once a month with predicted future value. A number of companies, including Algorithmia, RStudio Connect, and Saturn Cloud, provide platforms for data scientists to deploy models without being experts in engineering.

In either of these scenarios, you still want to have strong processes and infrastructure: systems to ensure that your code is tested before being deployed, ways of monitoring how well the models are maintaining accuracy, etc. Setting these up will require a combination of data science and engineering expertise, and the effort required to have them work smoothly shouldn't be underestimated.

When Your Team Members Don't Align with Your Infrastructure

Your infrastructure decisions will become more solidified as your team matures. More and more processes will be built around your particular databases and workplaces, and your team will become more comfortable with them. In general, this is a positive thing! It means your team is working through issues and becoming faster

and more experienced. You may, however, find a discrepancy when it comes to bringing new people onto your team. When hiring new people, you'll need to assess how much of the infrastructure you use a candidate must have experience with. This can be decided explicitly by only considering resumes that have the necessary skill set or implicitly by having interview questions that weed out people without experience.

It has been the case for many years now (and will likely continue to be the case) that there are not that many experienced data science candidates on the job market. While there may be many people who want to be data scientists in general, the number of those who have a lot of experience and are actively looking for jobs becomes quite small. It may be the case that few people on the job market are experienced in some areas of your tech stack, and certainly no one will be already experienced with all of them.

The good news is that data scientists generally *love* to learn—it's a profession built around discovering new things. When hiring, if a candidate doesn't have experience with your particular tech stack, do not worry: they can learn on the job. As much as possible, be relaxed with your technical constraints on what candidates should know and allow as many substitutes as you can. For example, if your team uses Python but the candidate only knows R, then that's an indication they can still learn Python once they join. Further, by hiring from a more diverse set of technical backgrounds, you increase the chance that someone arriving might know a better way of doing things than your team's current practices. It really is the case that it's worth playing the long game here and hiring people who will be great with a little ramp-up rather than only hiring people who know everything on day one.

Where to Go Next

Having read through this report, you hopefully have thought about leading a data science team in new ways. While the report has covered many areas, we can summarize it with a few key concepts:

Thinking about how your team integrates and communicates is important
: The success of a data science team often comes down to things like how well the stakeholders and team can work together with clear communication and how the goals of the data science

team are integrated with the goals of the broader organization. A data science team leader's job is to monitor this and tackle issues the moment they arise. A leader also needs to keep track of how communication happens within the data science team—between data scientists, between independent contributors and managers, and between data scientists and stakeholders.

A leader is responsible for ensuring that the data science work gets done regardless of how tricky it is

Data science teams have a constant flow of new tasks coming in, and each task can be risky because you don't know if you'll have the data or signal to actually do it. A leader needs to keep track of the work, prioritizing it based on the risk levels and importance, and ensuring the data scientists are focused on finishing the work and not getting distracted. This is a lot of distinct components to have to keep track of.

The technology powering your team matters, as does how you make the decisions around it

There are lots of technology decisions your team will have to make and many companies out there trying to sell you technology you don't need. You'll want your team to thoughtfully decide the right balance of distinct platforms to use in a way that leaves everyone happy. A leader will need to find the right balance of personally picking which technologies everyone is required to use versus letting each person on the team make their own decisions. Choosing the way you make decisions is as important as choosing the technology itself.

If you want more information and discussion around being a data science leader, here are a few resources:

- *How to Lead in Data Science* by Jike Chong and Yue Cathy Chang (Manning) is a deeper dive into many of the topics discussed in this report.
- For more general thoughts on engineering leadership, check out *The Manager's Path* by Camille Fournier (O'Reilly).
- Social media platforms like Twitter and LinkedIn can often have great discussions by data science professionals and leaders about the challenges they face and solutions they've found.

Best of luck on your continued journey as a data science leader!

About the Author

Dr. Jacqueline Nolis is a data science leader with over 15 years of experience in managing data science teams and projects at companies ranging from DSW to Airbnb. She is currently the head of data science at Saturn Cloud where she helps design products for data scientists. Jacqueline has a PhD in Industrial Engineering and coauthored *Build a Career in Data Science* (Manning).