BUILD A CAREER IN

Data Science

MEAP

Emily Robinson
Jacqueline Nolis

MANNING

MEAP Edition
Manning Early Access Program
Build a Career in Data Science
Version 2

For more information on this and other Manning titles go to
manning.com

welcome

Thank you for purchasing the MEAP for *Building a Career in Data Science!*

Data science is a fast-growing and exciting field, with many companies building out their teams and more and more people wanting to join it. But how do you go from making the decision to become a data scientist to actually becoming one? And not only that, but someone who excels in their role?

The idea for this book started when we realized how many questions we were getting from people on their data science career path. We each knew of some good blog posts that have been written on various topics, but we couldn't suggest a book to read that delivered a full how to guide on how to start and grow a data science career. Well, until now.

This book is written for people who either want to learn how to start a career in data science or grow their existing one. Part 1 of the book covers how to get into the field of data science—what it is, how it varies by company, and how to build your skills and create a portfolio of projects. Part 2 covers getting a job, starting from how to find promising positions to writing a stellar resume and cover letter, interviewing well, and negotiating an offer. Part 3 is about what to expect and how to perform well once you're in the job, including making analyses, deploying models into production, and dealing with stakeholders. Finally, part 4 will help you grow in your role, advising you on how to handle failed projects, contribute to open-source, and choosing between management and individual contributor roles.

The book is opinionated—rather than it being a collection of provable facts, we've focused on including things we've personally noticed can really help or harm a career (including our own). At every turn we've strived to be actionable—the goal of the book has been that the reader will walk away with new tools and ideas to try. At the end of each chapter we've included an interview with a prominent data scientist whose career has highlighted the lessons of the chapter. While at time we discuss technical topics, we've kept the book technologically agnostic to focus on the career components.

As you read the book, we would love to hear what you've found useful or what you've wished we went deeper on. If you have any questions, comments, or suggestions, please share them in Manning's liveBook Discussion Forum for our book.

—Emily Robinson and Jacqueline Nolis

brief contents

1

What is data science?

This chapter covers

- The three main areas of data science: mathematics/statistics, databases/programming, and business understanding
- The different types of data science jobs

"The sexiest job of the 21st century." "The best job in America." Data Scientist, a title that didn't even exist before 2008, is now the position employers can't hire enough of and job seekers strive to become. There's good reason for the hype - data science is a hugely growing field with a median salary of over $100,000 in 2017. At a good company, data scientists enjoy a lot of autonomy and are learning new things constantly. You use your skills to solve significant problems: working with doctors to analyze drug trials, helping a sports team pick their new draftees, or redesigning the pricing model for a widget business. As we will discuss in Chapter 3, there's no one way to become a data scientist. People come from all different backgrounds, so you're not limited based on what you chose to study as an undergraduate.

But not all data science jobs are perfect. Both companies and job seekers can have unrealistic expectations. Companies new to data science may think one person can solve all their problems with data. When a data scientist is finally hired, they can be faced with a never-ending to-do list and requests. They might also be tasked with immediately implementing a machine learning system when there's been no work to prepare or clean the data. There may be no one to mentor or guide them or even empathize with the problems they face. We'll discuss these issues more in Chapters 5 and 7, where we'll help you avoid joining companies that are likely to be bad fits for a new data scientist, and in Chapter 9, where we'll advise you on what to do if you end up in a bad situation.

On the other side, job seekers may think there will never be a dull moment in their new career. They may expect that stakeholders will routinely follow their recommendations, that

data engineers can immediately fix any data quality issues, and that they'll get the fastest computing resources available to implement their models. In reality, data scientists spend a lot of time cleaning and preparing data and managing the expectations and priorities of other teams. Projects won't always work out. Senior management may make unrealistic promises to clients about what your data science models can deliver. A person's main job may be to work with an archaic data system that's impossible to automate and requires hours of mind-numbing work each week just to clean up the data. You may notice lots of statistical or technical mistakes in legacy analyses that have real consequences, but no one is interested and you're so overloaded with work you have no time to try to fix them. A data scientist may be asked to prepare reports that support what senior management has already decided, where if you give an independent answer you worry you may be fired.

This book is here to guide you through the process of becoming a data scientist and developing your career. We want to ensure that you, the reader, get all of those great parts of being a data scientist and avoid most of the pitfalls. Maybe you're working in an adjacent field like marketing analytics and are wondering about how to make the switch. Or maybe you're already a data scientist, but you're looking for a new job and don't think you approached your first job search well. Or you want to grow in your current role to become a data science manager. Whatever your level, we're confident you'll find this book helpful.

We'll cover the main opportunities for gaining data science skills and how to build a portfolio to get around the paradox of "needing experience to get experience." You'll learn how to write a cover letter and resume that will get you an interview, but also how to build your network to get a referral. We'll cover negotiation strategies that research has shown will get you the best offer possible.

When you're in a data science job, you'll be writing analyses, working with stakeholders, and maybe even putting a model in production. We'll help you understand what all of those processes look like and how to set yourself up for success. When a project inevitably does fail, you'll have strategies to pick yourself back up. And when you're ready, we're here to guide you through the decision of where to take your career - management, staying an individual contributor, or even striking out as an independent consultant.

But before we begin that journey, we need to be clear on what a data scientist is and what is the work they do. Data science is a broad field covering many different types of work, and the better you understand the differences between those areas, the better you can grow in them.

1.1 What is data science?

Data science is the practice of using data to try to understand and solve real-world problems. This isn't exactly new- people have been analyzing sales figures and trends since the invention of the zero. In the last decade, however, we have gained access to exponentially more data than existed before. The advent of computers has assisted in the generation of all that data, but computing is also our only way to process the mounds of information. With computer

code, a data scientist can transform or aggregate data, run statistical analyses, or train machine learning models. The output of this code may be a report or dashboard for human consumption, or it could be a machine learning model that will be deployed to run continuously.

For example, if a retail company is having trouble deciding where to put a new store, they may call a data scientist in to do an analysis. The data scientist could look at historical data of locations where online orders are shipped to understand where the customer demand is. They may also combine that customer location data with demographic and income information for those localities from census records. With these data sets, they could find the optimal place for the new store and create a PowerPoint presentation to present their recommendation to the company's VP of Retail Operations.

In another situation, that same retail company may want to increase their online order sizes by recommending items to customers while they shop. A data scientist could load the historical web order data and create a machine learning model that, given a set of items currently in the cart, predicts the best item to recommend to the shopper. After creating that model, the data scientist would work with the company's engineering team so that every time a customer is shopping, the new machine learning model will serve up the recommended items.

When many people first start looking into data science, one challenge they face is being overwhelmed by the amount of things they need to learn. There's coding (but which language?), statistics (but which methods are most important in practice and which are largely academic?), machine learning (but how is machine learning different than statistics or AI?), and the domain knowledge of whatever industry they want to work in. This is all in addition to business skills like effectively communicating results to audiences ranging from other data scientists to the most senior people in the company. This anxiety can be exacerbated by job postings that ask for a PhD, multiple years of data science experience, and expertise in a laundry list of statistical and programming methods. How can you possibly learn all these skills? Which ones should you start with? What are the basics?

If you've looked into the different areas of work in data science, you may be familiar with Drew Conway's popular data science Venn diagram. In his opinion (at the time of its creation), data science falls in the intersection of math and statistical knowledge, expertise in a domain, and "hacking" skills (i.e. coding). This image is often used as the cornerstone of defining what a data scientist is. From our perspective, the components of data science are slightly different than what he proposed:

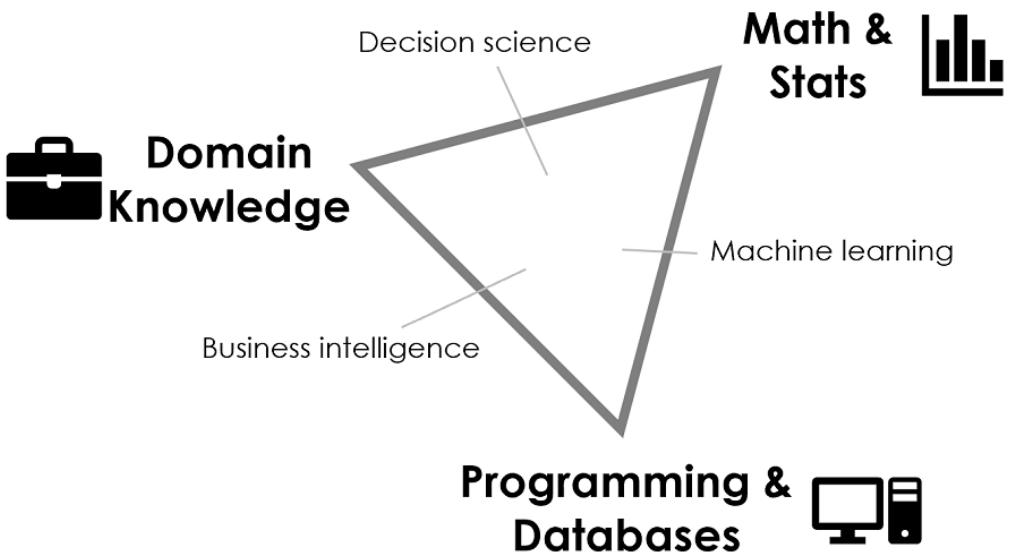


Figure 1.1

We've changed it to a triangle because it's not that you either have a skill or you don't, it's that you may possess it to a different extent than others in the field. While it's true that all three skills are fundamental and you need to have some of each, you don't need to be an expert in all of them. We put within the triangle different types of data science specialties. These don't always map one-to-one with job titles; even when they do, sometimes different companies will call them something different.

So what do each of these components mean?

1.1.1 Mathematics/statistics

At the basic level, mathematics and statistics knowledge is data literacy. We break that down into three different levels of knowledge:

- **That techniques exist**. If you don't know something is possible, you can't use it. For instance, if a data scientist was trying to group similar customers together, knowing that there are statistical methods (called "clustering") to do this would be the first step.
- **How to apply them**. While a data scientist may know about many techniques, they also need to be able to understand the complexities of applying them. That means not only how to write code to apply the method, but also how to configure them. If the data scientist wants to use a method like "*k*-means clustering" to group the customers, they would need to understand how to do a *k*-means clustering in a programming language like R or Python. They would also need to understand how to adjust parameters of the method, like how to choose how many groups to create.

- **How to choose which to try**. Because there are so many possible techniques that can be used in data science, it's important that the data scientist is able to quickly assess if a technique would work well or not. In our customer grouping example, even once you've focused on clustering, there are dozens of different methods and algorithms. Rather than trying each one, a data scientist needs to be able to rule out methods quickly and focus on just a few.

These sorts of skills are being constantly used within a data science role. To take a different example, let's say you work at an e-commerce company. Your business partner might be interested in what countries have the highest average order value. If you have the data available, this is an easy question to answer. But rather than simply presenting this information and letting them draw their own conclusions, you could dig deeper. If you only have one order from country A that was $100, and a thousand orders from country B with an average of $75, it is correct that country A has the higher average order value. But would you be confident in saying that means they should definitely invest in advertising in country A to increase the number of orders? Probably not - you only have one data point, and maybe it's an outlier. If country A had 500 orders instead, you might use a statistical test to see if the order value was significantly different, meaning that if there really was no difference in the entire population of country A and country B on this measure, it would be unlikely you'd have gotten this data from your sample. In this one paragraph-long example, many different assessments were made on what approaches were sensible, what should be considered, and what results were deemed unimportant.

1.1.2 Databases/programming

"Programming and databases" refers to the ability to pull data from company databases and to write clean, efficient, maintainable code. These skills are in many ways similar to what a software developer has to know, except data scientists have to write code that does open-ended analysis rather than producing a predefined output. Each company's data stack is unique, so there is no one set of technical skills that are required for a data scientist. But broadly, you'll need to know how to get data from a database and how to clean, manipulate, summarize, visualize, and share data.

For most data science jobs, R or Python will be the main language you use. R is a programming language that has its roots in statistics, so it's generally strongest in statistical analysis and modeling, visualization, and generating reports with results. Python is a programming language that started as a general software development language and has become extremely popular in data science. Python is known for being better at working with large data sets, doing machine learning, and being put into production. But thanks to the work of many contributors, the two languages capabilities are now at near parity. Data scientists are successfully putting machine learning models in R into production, and they are also making clean, presentable statistical analyses in Python.

R and Python are the most popular languages for data science for a couple of reasons. They're open-source, meaning they're free and many people, not just one company or one group, contribute code you can use. They have many packages or libraries (sets of code) for doing data collection, manipulation and visualization, statistical analysis, and machine learning. And importantly, because they each have such a large following, its easy for data scientists to find help when they run into issues. While some companies still use SAS, SPSS, STATA, MATLAB, or other paid programs, many of them are actually starting to move to use R or Python instead.

While most data science analysis is done in R or Python, you'll often need to work with a database to get the data. This is where the language SQL comes in. SQL is the programming language that most databases use to manipulate data within them or to extract it. For example, consider a data scientist who wanted to analyze the millions of records of customer orders in a company to forecast how the orders per day will change over time. Here they would likely first write a SQL query to get the number of orders each day. Then they would take those daily order counts and run a statistical forecast in R or Python. For this reason, SQL is extremely popular in the data science community, and it's difficult to get too far without knowing it.

Finally, another core skill is using version control. Version control is a method of keeping track how code is changing over time. Version control lets you store your files, revert them back to a previous time, and see who changed what file, how, and when. It's extremely important for data science and software engineering because if someone accidentally changes a file that breaks your code, you want the ability to revert or see what changed.

Git is by far the most commonly used system for version control, often used in conjunction with GitHub, a web-based hosting service for git. Git allows you to save ("commit") your changes as well as go back and see the whole history of the project and how it has changed with each commit. If two people are working on the same file separately, git makes sure that no one's work is ever accidentally deleted or overwritten. At many companies, especially those with strong engineering teams, you'll need to use git if you want to share your code or put something in production.

Can you be a data scientist without programming?

It's possible to do a lot of data work using only Excel, Tableau, or other business intelligence tools that have graphical interfaces. While you're not writing code, these tools claim to have much of the same functionality as languages like R or Python. But can this be a complete data science toolkit? We say no. Practically, very few companies have a data science team where you wouldn't need to program. But even if that wasn't the case, programming has advantages over these tools.

The first advantage of programming is reproducibility. When you write code instead of using point-and-click software, you're able to rerun it whenever your data changes, whether that's every day or in six months. This also ties into version control - instead of re-naming your file every time your code changes so you can always go back to an old piece, you're able to keep one file but see the entire history.

The second is flexibility. If Tableau doesn't have a type of graph available, you won't be able to create it. With programming, you can write your own code to make something the creators and maintainers of a tool never thought of.

The third and final advantage for open-source languages like Python and R are the community contributions. Thousands of people create packages, or bundles of code, and publish it openly on GitHub. That means you can download their code and use it for your own problems. You're not reliant on one company or group of people to add features.

1.1.3 Business understanding

"Any sufficiently advanced technology is indistinguishable from magic."

-Arthur C. Clarke.

Businesses have, to put it mildly, a varying understanding of how data science works. Often, management would just like something done and turn to their data science unicorns to make that thing happen. A core skill in data science is knowing how to translate a business situation into a data question, find the data answer, and finally deliver back the business answer. For example, a business person might ask "why are our customers leaving?" But it's not like there is a "why-are-customers-leaving" Python package you can import—it's up to the data scientist to deduce how to answer that question with data.

Business understanding is where your data science ideals meet head-on with the practicalities of the real world. It's not enough to want a specific piece of information without knowing how the data is stored and updated at your specific company. If your company is a subscription service, where does the data live? If someone changes their subscription, what happens? Does their row get updated or is another row added to the table? Are there errors or inconsistencies in the data you need to work around? If you don't know the answers to these questions, you won't be able to give an accurate answer to the basic question of, "How many subscribers did we have on March 2, 2019?"

Business understanding also helps you to know what questions to ask. Hearing from an employer, "What should we do next," is a little like being asked, "Why do we not have more money?" It's a question that begs more questions. Developing an understanding of the core business (as well as the personalities involved) can help you parse the situation better. Maybe follow-up with "Which product line are you looking for guidance regarding?" or "Would you like to see more participation from a certain sector of our audience?"

Part of business understanding is also developing general business skills like being able to tailor your presentations and reports to different audiences. Sometimes you will be discussing a better methodology with a room full of statistics PhDs and sometimes you will be in front of a vice president who hasn't taken a math class in twenty years. You need to inform your audience without either talking down or overcomplicating.

Finally, as you become more senior, part of your job is to identify where the business could benefit from data science and/or machine learning. If you've wanted to build a prediction system for your company, but have never had management support, becoming part of the management team can help solve that. Be on the lookout for places you can implement

machine learning; you know its limitations and capabilities so you can suggest which kind of tasks would benefit from automation.

Will data science disappear?

Underlying the question of whether data science will still be around in a decade or two are two main concerns - that the job will become automated and that data science is overhyped and the job market bubble will pop.

It's true that certain parts of the data science pipeline can be automated. Automated Machine Learning, or AutoML, can compare the performance of different models and perform certain parts of data preparation (like scaling variables). But these are just a small part of the data science process. For example, you'll often need to create the data yourself - it's very rare that there is perfectly clean data waiting for you. Creating the data will usually involve talking with other people, such as user-experience researchers or engineers, who will conduct the survey or log the user actions that can drive your analysis. And often setting up the AutoML can take just as much time as you would spend if you did not have it.

Regarding the possibility of a pop in the job market bubble, a good comparison is software engineering in the 1980s. As computers grew cheaper, faster, and more common, there were concerns that soon a computer could do everything and there would be no need for programmers. But the opposite happened and there are now more than 1.2 million software engineers in the United States (Bureau of Labor Statistics, U.S. Department of Labor, Occupational Outlook Handbook, Software Developers, on the Internet at https://www.bls.gov/ooh/computer-and-information-technology/software-developers.htm). While things like the title "webmaster" did disappear, there are now more people working on website development, maintenance, and improvement than ever.

We believe that there will be more specialization within data science, which might lead to the general title of data scientist disappearing. But many companies are just in the early stages of learning how they can leverage data science.

1.2 Different types of data science jobs

You can mix and match those three core skills of data science into a number of jobs, all which have some justification to be called a data scientist. In our perspective, there are three main ways that these skills get mixed together: business intelligence, machine learning, and decision science. Each of those three areas serves a different purpose to the company and fundamentally delivers a different thing.

When looking for data science jobs, you should pay less attention to the job titles and more to the job descriptions and what they ask you in the interviews. Look at the background of people in those roles – what were their previous jobs and degrees? You may find that people working similar sounding jobs have totally different titles, or that people with the same data scientist title do totally different things. As we talk about the different types of data science jobs, remember that the actual titles used at companies may vary.

1.2.1 Business Intelligence

A business intelligence analyst takes data and puts it in front of the right people. For example, after a company sets its yearly goals, you might put that in a dashboard so they can track the progress every week. You could build in features so that they could easily break down the

numbers by country or product type. This work involves a lot of data cleaning and preparation, but generally less work to interpret the data. While you should be able to spot and fix data quality issues, the primary person making decisions with this data is the business partner. Thus the job of a business intelligence analyst is to take data from within the company, format it and arrange it effectively, and deliver that data to others.

Because there's not a lot of statistics and machine learning in this role, some people and companies would consider this role outside of the field of data science. But there is a lot of work like devising meaningful visualizations and deciding on particular data transformations that are the same skills as required in the other types of data science roles. For example, a business intelligence analyst may be given a task like "create an automated dashboard that shows how our number of subscribers is changing over time and lets us filter the data to just subscribers of specific products or in specific geographical regions." The business intelligence analyst would have to find the appropriate data within the company, figure out how to transform the data appropriately (like change it from daily to weekly new subscriptions), then create a meaningful set of dashboards that are visually compelling and automatically update each day without errors.

Short rule: a business intelligence analyst creates *dashboards and reports that deliver data.*

1.2.2 Machine Learning

A machine learning engineer develops machine learning models and puts them in production, where they will run continuously. They may work on making the ranking algorithm for the search results of an e-commerce site, optimizing a recommendation system, or monitoring a model in production to make sure its performance hasn't degraded since it was deployed. A machine learning engineer spends less time on things like creating visualizations that will convince people of something, and more time doing the programming work of data science.

A big difference between this role and other types of data science positions is that your work output is primarily for machine consumption—you create machine learning models that get turned into APIs, or application programming interfaces, for other machines. In many ways you will be closer to a software developer than other data science roles. While it's good for any data scientist to follow best coding practices, as a machine learning engineer you must. Your code must be performant, tested, and written so that other people will be able to work with it. For this reason, many machine learning engineers come from a computer science background.

In a machine learning engineer role, a person may be asked to create a machine learning model that can in real time predict the probability that a customer on the website will actually finish their order. The machine learning engineer would have to find historical data in the company, train a machine learning model on it, take that model and turn it into an API, then deploy the API so the website can run the model. If that model stops working for some reason, the machine learning engineer will be called to fix it.

Short rule: a machine learning engineer creates *models that get run continuously.*

1.2.3 Decision Science

A decision scientist turns a company's raw data into information that helps the company to make decisions. This work relies on having a deep understanding of different mathematical and statistical methods and a familiarity with business decision-making. Furthermore, decision scientists have to be able to make compelling visualizations and tables so that the non-technical people they talk to will understand their analysis. While a decision scientist does plenty of programming, their work generally only gets run once to make a particular analysis. That means they can get away with having code that's inefficient or difficult to maintain.

A decision scientist must understand the needs of the other people within the company and figure out how to generate constructive information. For example, a marketing director may ask a decision scientist to help decide which types of products they should highlight in their holiday gift guide. A decision scientist might investigate what products have sold well without having been featured in the gift guide, talk to the user research team about conducting a survey, and use principles of behavioral science to do an analysis to come up with the optimal items to suggest. The final result is likely to be a PowerPoint presentation or report to be shared with product managers, vice presidents, and other business people.

Short rule: a decision scientist uses analyses that produce *recommendations.*

1.2.4 Related jobs

While those three areas are the main types of data science positions, there are a few other distinct roles that you may see that fall outside those types. We list those jobs here since it's good to understand the positions out there and you may need to collaborate with colleagues in these positions. That said, if you are interested in one of these roles, the material in this book may be less relevant to you.

DATA ANALYST

A data analyst is a person who does work similar to that of a decision scientist, however they generally use less statistical and programming expertise. Their tool of choice may be Excel instead of Python, and they may not regularly make statistical models. While they function in role similar to a decision scientist, because of the limitations of their tools and techniques they create less sophisticated output. That being said, this varies across companies, and practitioners have different opinions on what the distinction should be.

One practical consideration is that it is more likely for a data analyst to really be a business analyst, a role tasked with maintaining reports. A business analyst might be tasked with taking a monthly sales report in Excel and updating it each month with new numbers, forever. If you want to do machine learning, programming, or apply statistical methods, this could be a very frustrating position since it won't help you gain new skills. Data analyst positions will also usually pay less than the other types of data science jobs and are considered less prestigious. But it can be a good entry point into becoming a data scientist, especially if you haven't

worked with data before in a business setting. If you want to start as a data analyst and grow into becoming a data scientist, look for positions where you can learn some skills you may not have, such as programming in R or Python.

DATA ENGINEER

A data engineer is a person who focuses on keeping data maintained in databases and ensuring that people can get the data they need. They don't run reports, make analyses or develop models; instead, they keep the data neatly stored and formatted in well-structured databases so other people can. For example, a data engineer may be tasked with maintaining all of the customer records in a large-scale cloud database and adding new tables to that database as requested. Data engineers are pretty different from data scientists and they're even more rare and in demand. A data engineer may help build the data backend components of a company's internal experimentation system and update the data processing flow when the jobs start taking too long. Other data engineers develop and monitor batch and streaming environments, managing data from collection to processing to data storage. If you're interested in data engineering, you'll need strong computer science skills; many data engineers are former software engineers.

RESEARCH SCIENTIST

A research scientist develops and implements new tools, algorithms, and methodologies, often to be used by other data scientists within the company. These types of positions will almost always require PhDs, usually in computer science, statistics, a quantitative social science, or related field. Research scientists may spend weeks researching and trying out methods to increase the power of online experiments, getting one percent more accuracy on image recognition in self-driving cars, or building a new deep learning algorithm. They may even spend time writing research papers that may rarely be used within the company but help raise the prestige of the company and hopefully advance the field. Because these positions require very specific backgrounds, we will not be focusing on them in this book.

1.3 Choosing your path

In chapter 3, we'll cover some options for obtaining the data science skills, the benefits and drawbacks for each, and some suggestions for choosing between them. It's good to start reflecting here on what area of data science you want to specialize in. Where do you already have experience in? We've seen data scientists who are former engineers, psychology professors, marketing managers, statistics students, and social workers. A lot of times, the knowledge you've gained in other jobs and academic areas can help you be a better data scientist. If you're already in data science, it's helpful to reflect now on which part of the triangle you're in. Are you happy with it? Do you want to switch to a different type of data science job? Transitioning is often available.

Can anyone become a data scientist? – Vicki Boykis

With all the optimism (and big potential salaries listed in news articles) around the data science sector, it's easy to see why it presents attractive career opportunities, particularly as the range and scope of data science job titles continues to expand. But, as a new entrant to the field, it's important to have both an optimistic, and realistic, nuanced view of where the data science market has been headed for the next couple years and adjust accordingly.

There's a couple of trends impacting the data science field today. First, data science as a field has been around almost ten years, and as such, has moved through the early stages of the hype cycle: mass media hype, early adoption, and consolidation. It's been overhyped, talked about in the media, adopted by Silicon Valley companies and beyond, and we're now at the point of high-growth adoption across larger companies, the standardization of data science workflow toolsets like Spark and AutoML.

Second, as a result, there is an oversupply of new data scientists, who've taken bootcamps, newly-established data science programs in universities, or online courses. The number of candidates per any given data science position, particularly at the entry level, has grown from 20 or so per slot to 100 or more. 500 resumes per open is not that uncommon anymore.

Third, the standardization of toolsets and the ready supply of labor, as well as the demand for people who have more experience in the field, has meant a shift in the way data science titles are distributed and a creation of a hierarchy of data science jobs and descriptions. For example, in some companies "data scientist" may mean creating models, but in some it means mostly running SQL analyses, the equivalent of what the data analyst title used to be.

This means several things for those looking to get into data science as newcomers. First and most importantly, they may find the job market to be extremely competitive and crowded, especially for those who are new to industry in general (like college graduates), or those making the transition from other industries, and competing with thousands of candidates just like them. Second, they may be applying for jobs that are not truly reflective of data science as it's portrayed in blog posts and the popular press – solely writing and implementing algorithms.

Given these trends, it's important to understand that it may be hard to initially differentiate yourself from the other stack of resumes on the pile to get into the final round of interviews. Though the strategies you read in this book may seem like a lot of work, they will help you stand out which is needed in this new, competitive data science environment.

1.4 Interview with Robert Chang on what data science is and his journey

Robert Chang is a data scientist at Airbnb. He previously worked at Twitter, where he worked on the Growth team doing product analytics, creating data pipelines, running experiments, and creating models. You can find his blog posts on data engineering, advice for new and aspiring data scientists, and his work at Airbnb and Twitter on medium.

1.4.1 Can you tell us about your current job?

I work as a data scientist at Airbnb. Currently I'm working on a new product called Airbnb Plus, where we're trying to provide high quality homes in addition to the typical marketplace homes. A big part of my job is focused on growing supply of the new brand, right now by building an automated system that can selectively identify, assess, and invite hosts to our program. I'm figuring out, given all the signals we have about a home, how we can use machine learning to predict which ones are most likely to be successful in this program.

A lot of my work is done in collaboration with the engineering and product teams. My machine learning model will be a component of a larger automated system, so there's a lot of collaboration between myself and the engineers to figure out how can we can consume the model's predictions and create feedback loops so it can evolve over time. I also work with product managers to make sure what we're building serves the business need.

1.4.2 What was your first data science journey?

My first job was as a data scientist at the Washington Post. Back in 2012, I was ready to leave academia and go into industry, but I didn't know what I wanted to do. I was hoping to be a data visualization scientist, having been impressed by work at the New York Times. When I went to my school's career fair and saw the Washington Post was hiring, naïve as I was, I just assumed they must be doing similar things to the New York Times. I applied and got the job, not doing any more due diligence.

If you were to ask for an example of how *not* to start your data science career, I definitely would volunteer myself! I got the job hoping to do either data visualization or modeling, but I realized very quickly my job was more that of a data engineer. A lot of my work was building ETL (extract transform load) pipelines, rerunning SQL scripts, and trying to make sure reports ran so we could report top-level metrics to executives. This was very painful at the time-I realized what I wanted to do was not aligned with what the company really needed and eventually left the job.

But in my subsequent years at Twitter and Airbnb, I realized I was seeing the norm and not the exception. When you're at a startup and building out the data capabilities, you have to build it layer upon layer. Monica Rogati has the famous blog post on the hierarchy of data science needs, which is extremely spot-on. But at the time, I was too new to appreciate how real-live data science work was done.

1.4.3 What should people look for in a data science job?

If you're looking for a data science position, you should focus on the state of data infrastructure of the company. If you join a company where there's just a bunch of raw data that's not stored in a data warehouse, it will probably take you months or sometimes even years to get to a point where you can do interesting analytics, experimentation, or machine learning. If that's not something you expect to do, you will have a fundamental misalignment between the stage of the company and how you want to contribute to the organization.

To assess this, you can ask questions like, "Do you have a data infrastructure team? How long have they been around? What is the data stack? Do you have a data engineering team? How do they work with data scientists? When you're building out a new product, do you have a process for instrumenting the logs, building out the data tables, and putting them in your data warehouse?" I wish I had asked those questions for my first job as I would have quickly realized a lot of those things weren't there. If they aren't there, you will be part of the team

that is responsible for building that, and you should expect to be spending quite a lot of time on that.

The second thing to look for is the people. There are three kinds of people you should be paying attention to. Assuming you don't want to be the first data scientist, you want to join a data science organization where there is an experienced leader. An experienced leader knows how to build and maintain a good infrastructure and workflow for data scientists to be productive. Second, look for a manager who is supportive of continuous learning. What's been useful for me is the concept of tour of duty –having a manager who supports you making a concrete learning ever 12-18 months, like learning experimentation for a year and then data engineering afterwards. Lastly, it's super important, especially when you're new to work, to work with a tech lead or senior data scientist who is very hands-on. Because for your day to day work, that's who would be the person who helps you the most.

1.4.4 What's your definition of a data scientist?

My favorite framework is Michael Hochster's, who's currently a director of data science at Stitch Fix. He talks about type A vs. type B data scientists: A stands for analysis and B stands for build. I like it because it defines data scientists by what they produce, not just what they do. For type A, your primary job is to perform analyses that give insights and recommendations decision makers can act on. That's very different from Type B, where your output is a data product or production system, like an ETL pipeline or machine learning product. I started in Type A and transitioned to Type B, but I still do a lot of analysis in service of making the product better, so the difference is definitely not clean-cut. I wouldn't say one is more interesting or difficult than the other; I think it depends on what kind of output you would like to produce.

DATA SCIENTIST TYPES

Although the terminology is different, type B corresponds to the machine learning type we discussed and type A to decision science and somewhat business intelligence.

1.4.5 What skills do you need to be a data scientist?

I think it depends on what kind of job you're looking for and what bar the employer sets. Top-tier companies in generally have a high bar, sometimes unreasonably high, because there are a lot of people trying to get in the company. They're generally looking for unicorns – someone who has data wrangling skills with R or Python as well as experience building ETL pipelines, data engineering, experiment design, and building models and putting them into production. That puts a lot of pressure on the candidates! While those are all skills you can eventually learn and may be useful for whatever problems that you're solving, I don't think they're necessary to get into data science.

When I joined the industry, I knew R relatively well, and some statistics and probability, was pretty much all I knew – basically no Python and no SQL at all. I don't think that's a bad place to start. If you can plan out your career by learning more things up front, that's always helpful, but I don't think that's a requirement. It's more important to have a curiosity for learning.

If you know R or Python and a little bit of SQL, you're already in a pretty good position to get into data science. If you're trying to get hired by top tech companies, you need a little more, but that's more for signaling effect than what you really need on the job. It's helpful to make distinction between the core skills you need to start your career in data science and others that are nice-to-have if you want to get into a competitive, brand-name company. You will learn what are the essential skills you need to be successful. I didn't have a path to acquire additional skills, but I had curiosity for learning new things, so I branched out and learned a bit more, and realized these are skills data scientist could have but it's not a must. That's why I think a data science unicorn is almost a myth – it's very hard to find someone like that.

1.4.6 How do you know if you would like being a data scientist?

My entry point into data science was data visualization – I was interested in how we can take real data and make beautiful visualizations that explain patterns in the world. See if you find pieces of data work that are inspiring and makes you want to know more about how it was built. It could be an analysis, a visualization, a machine learning model. If you think, "I wish I could do the same thing but I don't know how," that's a good indication that this is a field that will attract you.

1.4.7 What's your final piece of advice to aspiring and junior data scientists?

Finding a field where you can always find joy and interest in your work is a pretty important thing to do. I worry a lot of people are considering data science because it's the hot thing and it seems like it's comfortable path that you can be on. I would urge people, especially those early in their career, to not fall into that trap. I work a lot, but I feel like I'm not working because I enjoy doing it. Being able to immerse yourself in that career path is most important thing. If you're intrinsically driven, you can go a long way.

1.5 Summary

- The data science skillset varies across people and positions; while some knowledge is fundamental, data scientists don't need to be experts in every relevant field.
- Data science jobs have different areas of focus: putting the right, cleaned data in front of stakeholders (business intelligence); putting machine learning models in production (machine learning); and utilizing data to make a decision (decision science).

2

Data science companies

This chapter covers

- The types of companies hiring data scientists
- The pros and cons of each of those types of companies
- The tech stacks you may see at different jobs

As we described in Chapter 1, data science is a wide field with lots of different roles: research scientist, machine learning engineer, business intelligence analyst, and more. While certainly the work you do as a data scientist depends on your role, it is equally influenced by the company you're working at. Big company vs small, tech vs traditional industry, and young vs established can all influence project focus, supporting technology, and team culture. By understanding a few archetypes of companies, you'll be better prepared when looking at places to work, either for your first data science job or your *n*th one.

The aim of this chapter is to give you an understanding of what some typical companies look like and what it's like to work there on a daily basis. Here we are going to present five fictional companies that hire data scientists all located in a make-believe "Data City." While none of these companies are real, they are all based on our research and work experiences, and we believe these examples illustrate basic principles that can be broadly applied. While no companies with data scientists are exactly alike, knowing these five archetypes should help you assess prospective employers. Also, while these are stereotypes based on what we've seen as trends in these industries, they are certainly not gospel. You may find a company that totally breaks the mold of what we say here—or a specific team in the company that is unlike the company itself. *Lastly, while the companies in this chapter are fake, all of the blurbs you'll see are from real data scientists working at real companies!*

2.1 MTC – the Massive Tech Company

- Similar to: Google, Facebook, Microsoft
- Company history: 20 years
- Employees: 40,000

MTC is a tech company with a massive footprint, selling cloud services, consumer productivity software like a text editor, server hardware, and countless one-off business solutions. The company has amassed a large fortune and uses it to fund unusual R&D projects like self-driving scooters VR technology. While their R&D makes the news, most of the technical workforce are engineers making incremental improvements to their existing products, adding more features, improving the user interface, and launching new versions.

2.1.1 Your team: one of many in MTC

MTC has nearly a thousand data scientists spread across the company. These data scientists are largely grouped into teams each supporting a different product or division, or individually placed within a non-data science team to fully support it. For example, there are VR headset data scientists on one team, marketing data scientists on a second team, and VR-headset marketing data scientists on a third team, while the VR-headset supply chain team has one data scientist within it.

If you were a member of one of those data science teams, when you joined you would be quickly onboarded. Large organizations hire new people every day, so the company should have standard processes for getting you a laptop, access to data, and training you on how to use any special tools. On the team, you'd be tasked with doing data science for your particular area of focus. That could include creating reports and charts that executives could use to justify funding the project. It could be statistical analyses of existing products to understand where new opportunities lie. It could also be building machine learning models that would be handed off to software developers to put into production. Some of your job would be working with related data science teams, sharing knowledge and data so everyone can do better.

Your team is likely to be to be large and full of experienced people. With MTC being a large, successful tech company, they have the broad footprint to draw in many good recruits to hire. Because your team is large, people within the team may be working on nearly unrelated tasks—one person could be doing an exploratory analysis for a boss in R while another builds a machine learning model in Python. The size of the team is a blessing and a curse: you have a large body of expert data scientists to discuss ideas with, but most of them probably don't have familiarity with the particular tasks you are working on.

MTC being a large company also means there is an established hierarchy on your team with specific titles given out. Perhaps one coworker is Data Scientist II or someone is a Principal Data Scientist. The people with the more senior positions tend to be listened to more, both because they have more experience in the field and more experience with dealing with different departments at MTC. The senior people are also paid much more. You are equally

likely to find a generous senior data scientist who is eager to help everyone as you are to find a cruel senior data scientist who uses their seniority to justify abusive behaviors.

The work your team does is likely a healthy balance of keeping the company running, such as making monthly reports and providing quarterly machine learning model updates, and doing new projects, like creating a forecast that has never been done before. The team's manager has to balance the flood of requests from other teams for data, analyses, and reports, which help those teams in the short term, with the desire to do interesting and innovative work that no one requested that may provide long-term benefits. With MTC's large cash stores the company can afford to do a lot more innovation and R&D than other companies, which trickles down into a willingness to try interesting new data science projects.

2.1.2 The tech: advanced, but siloed across the company

MTC is a massive organization, and with organizations of this size it's impossible to avoid using different types of technology throughout the company. One department may store order and customer data in a Microsoft SQL Server database, while a different department keeps records in Apache Hive. Worse, not only is the technology to store data disjointed, the data itself may be as well. One department may keep customer records indexed by phone number, while a different department may use email addresses to index customers. There are many people at MTC who effectively have the job of getting these disparate datasets to connect to each other.

Most MTCs have their own massive server farms running the company's hardware. They don't need to use the cloud because they effectively are the cloud. Thus, as a data scientist at MTC you will have to learn the specific ways to query and use data that are particular to MTC. Learning these specialized tools is great for getting you more access within MTC, but the knowledge you gain can't be transferred to other companies.

As a data scientist, you will likely have a number of possible tools you can use. Being a large organization, there will be plenty of support for major languages like R and Python since many people use them. Some teams may also use paid languages like SAS or SPSS, but this is a bit rarer. If you want to use an unusual language that you enjoy but few other people use, such has Haskell, you may or may not be able to, depending on your manager. Your manager has to balance the benefit of you using a tool you like with the risk of fragmenting the code so much that no one else could support it if you're unavailable.

The machine learning stack varies dramatically by what part of the company you are in. Some teams use Docker containers to deploy their code. They have extremely smooth pipelines to deploy the containers to Kubernetes clusters, and a development operations team (DevOps) to ensure that if anything happens the service can be quickly rebooted. Other teams use more classical methods of deploy code onto virtual machines that can be scaled up and down as needed. The diversity in tech stack for deploying software makes it difficult to connect to other team's APIs—there is no one central location to learn about and understand what is going on.

2.1.3 The pros and cons of MTC

Being a data scientist at MTC means having an impressive job at an impressive company. Because MTC is a tech company, people know what a data scientist is and what helpful things you can do. Having a universal understanding of your role makes the job a lot easier and is a huge perk. The number of data scientists in the company provides other benefits as well. You have a large support network you can rely on if you are struggling, and smooth processes for joining the company and getting access to required resources. It'll be rare for you to find yourself stuck and on your own.

Having lots of data scientists around you comes with cons as well. The tech stack is complex and difficult to navigate because so many people have built it up in so many ways. An analysis you've been asked to recreate may be written in a language you don't know by a person no longer around. It'll be harder to stand out and be noticed, because there are so many other data scientists around you. And you may find it difficult to find an interesting project to work on, because so many of the obvious projects have already been started by other people.

With MTC being an established company, working there gives you more job security. While there is always the risk of layoffs, it's not like working at a startup where funding could dry up at any moment. And at large companies managers lean more towards finding a new team for someone to work on rather than firing them—firing opens up all sorts of legal complications that require thorough back-up support for the termination decision.

Something that is both a pro and con of MTC is that there are people in many specialized roles within the company. There are data engineers, data architects, data scientists, market researchers, and more, all doing different roles that relate to data science. This means you'll have lots of people to pass work off to— for example, you have a low chance of being forced to create your own database. To some data scientists that is great, since it'll let them focus on their own specialty. To others it's a curse, since many of these other types of work are interesting, or at least something a data scientist would want substantial control over.

Another con of MTC is the bureaucracy. Being a large company means that getting approvals for things like new technology, trips to conferences, or starting projects can require going far up the chain of command. Worse, the project you've been working on for years could be cancelled because two executives are fighting, and your project is collateral damage. As a data scientist you'll be far away from the decision making, which can be difficult to handle emotionally.

MTC is a great company to work at for data scientists who are looking to help solve big problems using cutting-edge techniques—both for decision scientists wanting to do analyses and machine learning engineers wanting to build and deploy models. Large companies have lots of problems to solve, and a budget that allows for trying out new things. You may not be able to make big decisions yourself, but you'll know you've contributed. MTC is a poor choice for a data scientist who wants to be the decision maker and call the shots—the large company has established methods, protocols, and structures that you have to fall into.

2.2 HandbagLOVE – the established retailer

- Similar to: Payless, Bed Bath & Beyond, Best Buy
- Company history: 45 years
- Size: 15,000 employees (10,000 in retail stores, 5,000 in corporate)

HandbagLOVE is a retail chain with 250 locations across the US, all selling purses and clutches. The company has been around for a long time and is filled with experts in how to lay out a store and improve the customer experience. The company has been generally slow to adopt new technology, taking plenty of time before getting its first website and its first app.

Recently, HandbagLOVE has been seeing sales drop as Amazon and other online retailers have eaten away at its market share. Knowing that the writing is on the wall, HandbagLOVE has been looking to improve via technology, investing in an online app, an Amazon Alexa skill, and trying to use the value they have in their data. HandbagLOVE has had financial analysts employed for many years calculating high-level aggregate statistics on their orders and customers; however, only recently have they considered hiring data scientists to better understand customer behavior.

The newly formed data science team was built starting with a base of financial analysts who had previously been making Excel reports on performance metrics for the company. As HandbagLOVE supplemented these people with trained data scientists, the team started to provide more sophisticated products: monthly statistical models on customer churn in R, interactive dashboards allowing executives to better understand sales, and a customer segmentation that buckets customers into helpful groups for marketing.

While the team has made machine learning models to power the new reports and analyses, HandbagLOVE is far from deploying machine learning models into continuously running production. Any product recommendations on their website and app are powered by 3rd party machine learning products, rather than having been built within the company. There is talk on the data science team of changing this, but who knows how many years away it is.

2.2.1 Your team: a small group struggling to grow

The team leans heavily towards data scientists who can do reporting rather than being trained in machine learning—a topic very new to many. In some ways that newness can be extremely helpful—by being new, they will likely be very interested in learning more and trying new things. In other ways it can be difficult: some of their programming and statistical methods may be inefficient or even wrong since they haven't been taught the correct way to do things.

HandbagLOVE has laid out general paths to progress into senior roles. Unfortunately, none of them are specific to data science—they are just high-level goals around being independent or things taken from the software development career management tools which don't quite apply. To progress in your career you mostly have to convince your manager that you're ready, and hopefully your manager can get approval to promote you. On the plus side, if the team ends up growing you'll quickly become a senior person on the team.

Since the data science team provides reports and models for departments throughout the company, like marketing, supply chain, and customer care, the members of the data science team are well known. This has given the team a great deal of respect within the company, and in turn the data science team has a lot of comradery within it. The combination of the size of the team and the level of influence within the company allows the data scientist to have far more influence than they would in other companies. It's not unusual for someone on the data science team to meet with top level executives and contribute to the conversation.

2.2.2 Your tech: a legacy stack that's starting to change

The common phrase you hear when talking about technology at HandbagLOVE is "well that's how it's always been." Order and customer data are stored in an Oracle database that is directly connected to the cash register technology, and hasn't changed in 20 years. The system has been pushed well past its limits and has had many modifications bolted on over the years. All that being said, it still works. On a daily basis, data is copied from the main Oracle database and put onto a separate server for the analysts to use. Other data is brought in as well: data collected from the website, data from the customer care calls, and data from promotions and marketing emails. All of these servers live on-prem, and an IT team keeps them maintained.

By having all of the data stored in one large server, you have the freedom to connect and join the data however you want. And while sometimes your queries may take forever or overload the system, usually you can find a workaround to get you something usable. With 40 million customers in the system, you may not be able to pull all of the data onto your laptop to model, but you can use sampling and other techniques.

The vast majority of analyses are done on your laptop. It's just easier to put the data on your computer than to try and find another solution. On those rare cases where you do need better performance, you may be able to convince the IT team to set up a temporary virtual machine with lots of processors and RAM.

The company doesn't have a machine learning tech stack because they don't have any in-house machine learning. Most of the team's time is spent reporting on aggregated statistics or building data science models to understand customer behavior. The team just doesn't have a need to deploy machine learning models to continuously run. HandbagLOVE does use machine learning to recommend products on the company app and website, however that's done with a third-party tool your team isn't involved with.

2.2.3 The pros and cons of HandbagLOVE

By being at HandbagLOVE, you have a lot of influence and ability to do what you think is wise. You can propose doing something like making a customer lifetime value model, then build it, then use it within the company—without having too many people you have to convince to let you run with your idea. This is due to a combination of the size of the company and the

newness of data science. This freedom is very rewarding; you are incredibly empowered to do what you think is best.

The downside of this power is that you don't have many people to call on for help. If you try and make a customer lifetime value model and the model just isn't performing well, you have few people to support you. You're responsible for finding a way to make things work or dealing with the fallout when things don't work. This responsibility can be very emotionally taxing—it's not unusual for data scientists to stay up at night thinking about what's going to happen if their models don't work.

The tech stack is antiquated, and you'll have to spend a lot of time making workarounds for that—which is a hassle. You may want to use a newer technology for storing data or running models but you won't have the technical support to do it. If you're not able to entirely set up any new technology yourself you'll just have to get by without using it.

A data scientist's salary won't be as high as it would be at bigger companies, especially tech ones. HandbagLOVE just doesn't have the cash available to pay high data science salaries—and besides the company doesn't need the best of the best anyway, just people who can do the basics. That being said, the salary won't be terrible; it'll certainly be well above what most people at the company make with similar years of experience.

HandbagLOVE is a good company to work at for data scientists who are excited to have the freedom to do what they think is right—but perhaps aren't interested in using the most state-of-the-art methods. If you're comfortable using standard statistical methods and making more mundane reporting, HandbagLOVE should be a comfortable place to grow your career. If you're really only interested in using start-of-the-art machine learning methods then you won't find many projects to do at HandbagLOVE, nor will you find many people there who know anything about what you're talking about.

2.3 Seg-Metra – the early-stage startup

- Similar to: Dropbox at the start, or a thousand failed startups you haven't heard of
- Company history: 3 years
- Size: 40 employees

Seg-Metra is a young company that sells a product that helps client companies optimize their website by customizing for unique "segments" of customers. Seg-Metra sells its product to businesses, not consumers. Early on in its brief history, Seg-Metra got a few big-name clients to start using the tool, which helped the company get more funding from venture capitalists. Now with millions of dollars at hand, the company is looking to quickly scale in size and improve the product.

The biggest improvement the founders have been pitching to investors is to start adding basic machine learning methods to the product—it was pitched to investors as "cutting-edge AI." With this new funding in hand, the founders are looking for machine learning engineers to build what they have pitched. There is also a need for decision scientists to start reporting on

the usage of the tool, allowing the company to better understand what improvements to make in the product.

2.3.1 Your team—what team?

Depending on when a data scientist gets hired, they may very well be the first data scientist in the company. If not the first, they will be in the first few data science hires and likely report to the one who was first hired. Due to the newness of the team there will be few to no protocols—no established languages, best practices, ways of storing code, or formal meetings.

Any direction will come from that first data scientist hire, who will be in charge of starting to establish protocols. The culture of the team will likely be set by their benevolence. If that person is open to group discussion and trusting of the other team members, then the data science team as a whole will decide things like what language to use. If that person is controlling and not open to listening, they will make these decisions themselves. (If you, the reader, are this person, please check out The Manager's Path: A Guide for Tech Leaders Navigating Growth and Change, by Camille Fournier.)

Such an unstructured environment can create immense comradery. The whole data science team works hard and struggles to get new technologies, methods, and tools all working and can form deep bonds and friendships. Alternatively, immense emotional abuse could happen at the hands of those with power, and since the company is so small there is little accountability. (For an example of this taken to its dark conclusion, check out Bad Blood: Secrets and Lies in a Silicon Valley Startup, by John Carreyrou.) Regardless of exactly how Seg-Metra's growth shakes out, the data scientists at this early stage company are in for a bumpy and wild ride.

The work of the team can be fascinating or frustrating depending on the day. Oftentimes data scientists are doing analyses for the first time ever—like making the first attempt at using customer purchase data to segment customers or deploying the first neural network into production. These first-time analyses and engineering tasks are fascinating because they are uncharted territory within the company, and the data scientists get to be the pioneers. On other days, the work can be grueling—like if a demo has to be ready for an investor and the model is still not converging the day before. While the work is chaotic, all of these tasks mean that the data scientists learn lots of skills very quickly while working at Seg-Metra.

2.3.2 The tech: cutting edge technology that is taped-together

By being a young company, Seg-Metra is not constrained by having to maintain old legacy technology. And by mainly hiring young engineers with only a few years of experience, people aren't attached to older methods, instead wanting to use the newest and greatest. Further, Seg-Metra wants to impress its investors, which is a lot easier to do when your technology stack is impressive. Thus, Seg-Metra is powered by the most recent and greatest methods of software development, data storage and collection, and analysis and reporting. From a software development side, this means using containers to continuously deploy microservices

using Docker containers and Kubernetes. Data is stored in an assortment of AWS technologies, such as DynamoDB, S3, and Redshift. The data scientists connect directly to these databases and build machine learning neural network models on large AWS EC2 instances with GPU processing. Those models are then deployed in Docker containers and used within the Seg-Metra tool.

At first glance, the tech stack is certainly impressive. But the company is so young and growing so fast that there are constantly issues with the different technologies working together. When the data scientists suddenly notice missing data in the S3 bucket, they have to wait for the overworked data engineer to fix it (and that's if they're lucky enough to have a data engineer and not just an engineer reluctantly helping out with data work). The tool suddenly stops working in production one afternoon and it takes 24 hours to resolve the issue—with great complaints from the customers. It would be great if Seg-Metra had a dedicated DevOps team to help keep everything running, but so far the budget has been spent elsewhere. Further, the technology has all been installed so quickly that even though the company is young it would be difficult to monitor it all.

2.3.3 Pros and cons of Seg-Metra

As a growing startup there is a lot of appeal to working at Seg-Metra. The growth of the company is providing all sorts of interesting data science work and an environment where you are forced to learn quickly. These sorts of positions can teach skills that jump start a career in data science: skills like working under deadlines with limited constraints, communicating effectively with non-data scientists, and knowing when to pursue a project or decide it's not worth it. Especially early in a career, developing these skills can make you much more attractive as an employee than people who have only worked at larger companies.

Another pro of working at Seg-Metra is that you get to work with the latest technologies. This is great for two reasons. One, by using the latest tech your job *should* be more enjoyable. Presumably the new technologies coming out, like giant EC2 instances with GPU, are better than the old technologies, like just running models on your laptop. Two, by learning the latest tech you should have a more impressive resume for future jobs. Companies looking to use newer technology will want you to help guide the way.

While the pay is not as competitive as larger companies, especially tech companies, the job does provide stock options which have the potential to be enormously valuable. If the company eventually goes public or gets sold, those options could be worth hundreds of thousands of dollars or more. Unfortunately, the odds of that happening are somewhere between "getting elected to city council" and "getting elected to US Congress." So this is only a pro if you enjoy gambling big.

One con of working at Seg-Metra is that you have to work very hard. Having 50- to 60-hour work weeks is not uncommon, and the company expects everyone to contribute everything they can. In the eyes of the company, if everyone isn't working together the company won't succeed, so are you really going to be the one person to use all of their vacation time in a year? This can create a hugely toxic environment ripe for abuse and a lot of

employee burnout. It's typical for employees in companies like Seg-Metra to quit the company after 1-2 years.

The company is volatile, relying on finding new clients and help from investors to stay afloat, giving Seg-Metra the con of low job security. It's possible that in any year the company could decide to lay off people or go under entirely. These changes can happen suddenly without warning, which is difficult for employees to handle—especially those with mortgages and families relying on them. This causes the demographics of the company to skew younger, which can also be a con if you want to work with a more diverse, experienced team.

Overall, working at Seg-Metra provides a great opportunity to work with interesting technology, learn a lot quickly, and have a small chance of making a ton of money. But to do so requires an immense amount of work and potentially a toxic environment. So this company is best for data scientists looking to gain experience and then move on.

Rodrigo Fuentealba Cartes, Lead Data Scientist at a small government consulting company

The company I work at provides analytics, data science and mobile solutions for governmental institutions, armed and law enforcement forces, and some private customers. I am the Lead Data Scientist, and I am the only one in charge of data science projects in the entire company. We don't have data engineers, data wranglers or any other data science roles there, because the department is relatively new. Instead, we have database administrators, software developers, systems integrators, and I double as a system/software architect and open source developer. That might look odd, and definitely puts strain on me, but it works surprisingly well.

One strange story from my job: I was working in a project that involved using historical information from many environmental variables such as daily weather conditions. There was a lack of critically needed data because an area of study didn't have weather stations installed. The project was in jeopardy and the customer decided to shut the project down in a week if their people could not find the information.

I decided to fly to the area and interview some fishermen and asked how they knew that it was safe to sail. They said they usually sent a ship that transmitted the weather conditions over the radio. We visited a radio station, and they had handwritten transcripts of communications since 1974. I implemented an algorithm that could recognize handwritten notes and extract meaningful information, and then implemented a natural language processing pipeline that could analyze the strings. Thanks to going out to the field and finding this unusual data, the project was saved.

Gustavo Coelho, a data science lead at a small start-up

I have been working for the last 11 months in a relatively new startup which focuses on applying AI to HR management. We predict future performance of candidates or their likelihood of being hired by a certain company. Those predictions are aimed at helping speed up the hiring process. We rely heavily on bias mitigation in our models. It's a small company, we have 11 people, and the Data Science team makes up 5 of them including me. The whole of the company is dedicated to helping the Data Science team deliver the trained models into production, the two other people are a cloud architect and a project manager.

Working at a small startup gives me the chance of learning new concepts and applying them every day. I love thinking about the best way to setup our data science processes so we can scale and give more freedom to our data scientists to focus on data science. HR is not a tech savvy field, so more than half of the project length is spent

explaining the solution to our clients and helping them get comfortable with the new concepts. And then when we finally get the go ahead there is also a lot of time spent coordinating with the client's IT department to integrate into our data pipeline.

2.4 Videory – the late-stage, successful tech start-up

- Similar to: Lyft, Twitter, and Airbnb
- Company history: 5-15 years
- Size: 2,000 people

Videory is a late-stage, successful tech company that runs a video-based social network. Users can upload 20 second videos and share them with the public. They have just gone public and everyone is ecstatic about it. They're not close to the size of MTC, but they're doing well as a social network and growing the customer base each year. They're data-savvy and have probably had data analysts or scientists for a few years now or even since the start. The data scientists on the team are very busy doing analyses and reporting to support the business, as well as creating machine-learning models to help pair people with artists to commission work.

2.4.1 The team: specialized but still room to move around

Videory is still at the point where you can gather all the data scientists in an extra-large conference room, or at least a small auditorium. Given the size of the company, the team may be organized into a *centralized model*. Every data science person reports to a data science manager, and they all are in a single large department of the organization. They help other teams throughout the company, but ultimately they set their own priorities, and some data scientists are working on long-term projects that have no immediate benefits. Some companies the size of Videory instead follow an *embedded model*: data scientists are placed within other teams, like product or finance, and report up to managers in those other organizations. Those managers set the data scientist's priorities. There is also a combination of the two: a hybrid model where each data scientist builds a relationship with specific team(s) and gets most of their day-to-day tasks from them, but their manager is still in a data science organization. Their manager can step in for them when needed; for instance, if a data scientist is working with three teams and each has five projects, the manager can help the data scientist prioritize and communicate what they will work on to the teams. A company may even have multiple types of models for organizing its data scientists. For example, the data scientists working on machine-learning models for production may have the centralized model, while those working on dashboards and company-wide metrics may be organized in an embedded model.

There's specialization among the data science team at Videory given the size of the company. The company has some delineation between people who do the heavy machine learning, statistics, or analytics (falling into the buckets of data science outlined in chapter 1). Videory is small enough that it's possible to move between these groups, however. There will

usually be some interaction between the all the data scientists – training sessions, monthly meetings, a shared slack channel – that you wouldn't find in MTC where there are hundreds or thousands of data scientists. The different sub-teams likely use different tools, and there is almost always a group of mostly PhDs who publish academic papers and do more theoretical work.

2.4.2 The tech: trying to not get bogged down by legacy code

Videory has a lot of legacy code and technology, and probably at least a few tools they developed internally (it is a tech company after all). Videory is likely trying to keep up with tech developments, and has plans to either switch over to a new system or supplement their existing ones with new technologies. Like most companies, a data scientist there will almost definitely use SQL. You also will probably have some business intelligence tools as well since there's a lot of non-data science consumers.

As a data scientist at Videory you'll definitely get to learn something new. All these companies have big data and systems to deal with it. SQL won't be enough – the company has billions of events they need to process a month. You may be able to try out Hadoop or Spark when you need to pull out some custom data that's not stored in the SQL database.

The data science is typically done in R or Python, with plenty of experts available to provide assistance if things are proving difficult. The machine learning is deployed through modern software development practices like using microservices with Docker containers and Kubernetes. Because of how well-known the company is for being a successful startup, lots of talented people are working at the company and bringing their cutting-edge approaches.

2.4.3 The pros and cons of Videory

Videory can be a good size for data scientists – there are enough other data scientists around for mentorship and support, but the team is still small enough that you can get to know everyone. Data science is recognized as important on the company-level, which means your work can get recognition from vice presidents and maybe even the C-suite. You'll have data engineers to support your work. The data pipelines may get slow sometimes or even rarely break, but you won't generally be responsible for fixing it (though if you broke it, you may be pulled in).

In a 1000+ person organization, you will need to deal with inevitable political issues. You may be pressured to generate numbers that match what people want to hear (and can tell to their bosses to get a bonus) or face unrealistic expectations of how fast something can be developed. You can also end up working on things that aren't really needed by the business because your manager asked you to. You'll sometimes end up feeling like you have had no direction or your time was wasted, and they'll wonder why they're paying you so much. While not as much as at an early-stage startup, the organization will still be changing a lot – what's a priority one quarter can be totally ignored the next. A report you spend weeks working on may just end up emailed out into the void.

While other data scientists at Videory will be more knowledgeable than you on most data science topics, you might quickly become the expert on a specific one, like time series analysis. This can be great if you like mentoring and teaching others, especially if your work supports you taking time to learn more on your particular field of expertise by reading papers or taking courses. But it can be hard when you feel like there's no one who could check your work or push you to learn new things. While you'll always have more to learn, it may not end up being the area you want to focus on.

Overall, Videory provides a nice blend of some of the benefits from the other archetypes — large enough that there are people around to provide help and assistance when needed, but not so large that requests get stuck in bureaucratic madness or that departments overlap in scope. Data scientists working at the company get plenty of chances to learn, but due to the specialization of roles they don't quite get the opportunities to try everything. This company is a great place for data scientists looking for a safe bet that provides chances to grow, but not so many that it's overwhelming.

Emily Bartha, the first data scientist at a mid-sized startup

I am the first data scientist at a mid-sized startup that has a product focused on insurance. As the first data scientist, I get to help define our strategy around using data and introducing machine learning into our product. I sit on the Data Team in the company, so I work very closely with data engineers, as well as our Data Product Manager.

A day in my life at work starts with morning standup with the Data Team. We talk about what we have planned for the day and any blockers or dependencies. I spend a lot of time digging through data - visualizing, creating reports, and investigating quality issues or quirks in the data. I spend a lot of time on documentation too. When I code, I use GitHub like the rest of the engineering team and have team members review my code (and I review theirs). I also spend a good chunk of the day in meetings, or side-of-desk collaboration with members of my team.

Having worked at bigger companies in the past, I love working at a small company! There is a lot of freedom to take initiative here - if you have an idea and want to work to make it a reality, no one will get in your way. Look for a company that has already made an investment in data engineering—when I arrived there were already several data engineers and a strategy for instrumentation, data collection, and storage.

When you work at a small company, things are constantly changing and priorities are shifting, which makes it important to be adaptable. People who enjoy diving deep on a project and working on it for months may not enjoy working at a startup because it often requires developing solutions that are "good enough" and moving on to the next thing. One surprising thing about working at an insurance tech company is that most of my coworkers didn't have any insurance experience before joining. In this industry, I feel like a lot of people learn as they go. Although it is really valuable to have some insurance veterans in the company to go to with questions.

2.5 Global Aerospace Dynamics (GAD) – the massive government contractor

- Similar to: Boeing, Raytheon, Lockheed Martin
- Company history: 50 years
- Size: 150,000 people

Global Aerospace Dynamics is a massive and rich company, bringing in tens of billions of dollars in revenue each year through various government contracts. The company develops everything from fighter jets and missiles to intelligent traffic light systems. The company is spread across the country through various divisions, most of which don't talk to each to each other. GAD has been around for decades, and many people working there have been there for decades too.

GAD has been slow on the uptake of data science. Most of the engineering divisions have been collecting data, but they struggle to understand how it can be used in their very regimented existing processes. Because of the nature of the work, code needs to be extremely unlikely to have bugs and ruthlessly tested, so the idea of implementing a machine-learning model, which has limited predictability when live, is dicey at best. In general, the pace of work at the company is slow; the tech world "move fast and break things" is the polar opposite of the mentality at GAD.

With the number of articles on artificial intelligence, the rise of machine learning, and the need for using data to transform a business, the executives in GAD are all ready to start hiring data scientists. Data scientists are showing up on teams throughout the organization, with tasks like analyzing engineering data for better reporting, building machine learning models to put into products, and working as service providers to help GAD customers troubleshoot problems.

2.5.1 The team—a data scientist in a sea of engineers

While it depends on where in GAD you are and what project you are working on, most data scientists are a single person on a team of engineers. At best there may be two or three data scientists on your team. The data scientist has the job of supporting the engineers with analysis, model building, and product delivery. Most of the engineers on the team only have a very loose understanding of what data science is—they remember regressions from college but don't know the basics around collecting data, feature engineering, the difficulties of validating a model, or how models get deployed. Thus, as a data scientist on the team you have few resources to help you when things go wrong, but since so few people understand your job, it may be the case that no one notices that things are going wrong.

Many of the engineers on the team will have been with the company for ten or more years, so they will have plenty of institutional knowledge. They'll also be more likely to have a mindset of: "we've been doing things this way since I've been here, why should we change?" That attitude will make it more difficult for ideas proposed by data scientists to actually be implemented.

The slower nature of the defense industry means that people tend to work less hard than in other places—people clock in 40 hours a week but casually slipping down below that isn't unusual. While that slower pace makes it harder to quickly launch innovative projects, it also means that there are fewer stresses in the job. At other companies it can be stressful to be overwhelmed by too many tasks, while at GAD the stress comes from not having enough work to do and being bored.

Promotions and raises are extremely formulaic, both due to formulas having less bias (and thus less likely to get GAD sued), and also because that's how things have been done for decades. Getting raises and promotions largely has to do with how many years you've worked at the company. Being an extremely hard worker may make your next promotion come a year earlier or earn you a somewhat higher bonus, but there is little chance of a junior data scientist quickly rising to a lead data scientist. The flip side of this is that people getting fired is rare. Just like at MTC, it's more likely that a poor worker is shifted to a different department than it is for the person to be let go. The only time when people are quickly let go is when the company has the occasional broad layoff. When layoffs occur, generally corresponding to bad economic conditions, managers cull people from their teams in mass.

2.5.2 The tech: old, hardened, and on security lockdown

While the technology stack greatly differs between different groups in GAD, it all tends to be relatively old, on-premises instead of in the cloud, and covered in security protocols. Since the data involved covers topics like fighter jet performance, it's essential for the company that it isn't leaked. Further, the company needs legal accountability for any technology they use in case something goes wrong, so open source tends to be frowned upon. For example, while Microsoft SQL Server is more expensive than PostGRES SQL, GAD is happy to pay Microsoft money knowing that if there is a security bug they can call Microsoft to deal with it.

In practice, this looks like data being stored in Microsoft SQL Server databases run by an IT team that is extremely stingy with who has access to what. The data scientists are allowed to access the data; however, they have to run Python on special servers that have limited internet access, so that any libraries don't secretly send data to foreign countries. If the data scientists want to use special open source software like setting up a MongoDB database, there is little chance of IT and security approving it. This ends up making it much more difficult for the data scientists to do work.

If code needs to be deployed to production systems, it tends to be in traditional ways. Most GAD departments don't use continuous integration to deploy code, and code is run on manually set up virtual machines rather than using microservices. The virtual machines are loaded with security software, and the IT team has to manually validate any code or executables that are deployed to them.

2.5.3 The pros and cons of GAD

The pros of working at GAD are that the data science jobs are slow, comfortable, and secure. The less rigorous pace of the job means you're more likely to have energy left when you get home for the evening. You'll often find yourself with free time when you are working, which you can spend reading data science blogs and articles without anyone complaining. The fact that few other people know the basics of data science means that you'll have fewer people questioning you. And because it's a massive organization worried about legal liabilities, you would have to really underperform before you're fired.

The downsides of working at GAD are that you are less likely to learn new skills than you would at other companies. You'll likely be assigned to a single project for years, and so the technologies and tools for it will quickly become mundane. Worse, the skills you do learn will be for outdated technology that isn't transferrable to other institutions. And while you won't easily get fired, you also won't easily get promoted.

GAD is a great place to work if you find a team doing projects you find enjoyable and you don't want work to be your life. Many people work for GAD for decades because it's comfortable and they are happy with that. But if you are a person who demands challenges to keep you going, GAD might not be a good fit.

Nathan Moore, a Data Analytics Manager for a utilities company

The company I work at provides and sells power for hundreds of thousands of people, and the company is partially owned by the government. The company itself has around 1,000 employees spread across many different functions. My job involves investigating and prototyping new data sources and working with the database specialists to clean and document current data sources. We've got a bunch of legacy systems and new initiatives happening so there's always something to do.

At the moment a day in the life involves meetings, reviewing specifications for ETL, trying out a new machine learning technique I found on Twitter, giving feedback on reporting, learning to use JIRA and Confluence, and many emails. In the past I've been involved in model development and assessment, data analysis when some overnight processing fails, and submissions to government on an industry-wide review of the sector.

We're large enough that we've got a good team of analysts to work on a variety of projects, from day to day reporting to large customer segmentation projects. I've had lots of opportunities to move around in the business and have worked here for 11 years. But since we have billions of dollars of assets risk aversion is high within the company and pace of change is a little bit slow. We have a large enough IT department that can support everyday functions, but any significant project like the systems upgrade means resources are scarce for any non-priority improvements. Everything needs to be justified and budget set aside and there is plenty of politics to navigate.

2.6 Putting it all together

We can summarize the five companies here into a few key ideas:

- MTC – a massive tech company with lots to learn from, but a wildly complex tech stack and plenty of institutional chaos
- HandbagLOVE – a mid-size retailer that doesn't have much data science, but provides lots of opportunities to try new things
- Seg-Metra – a startup that is growing and needs data scientists, but hasn't figured out quite what to do and burns plenty of employees out
- Videory – a mid-size tech company that has a team of data scientists who have specialized, but isn't overwhelmed by bureaucracy
- GAD – a defense contractor that is hiring lots of data scientists to do work that isn't particularly demanding

Comparing them in a table looks something like:

Criteria	MTC	HandbagLOVE	Seg-Metra	Videory	GAD
	Massive tech	Retailer	Startup	Mid tech	Defense
Bureaucracy	Lots	Little	None	Some	Lots
Tech stack	Complex	Old	Fragile	Mixed	Ancient
Freedom	Little	Lots	TONS	Lots	None
Salary	Amazing	Decent	Poor	Great	Decent
Job security	Great	Decent	Poor	Decent	Great
Chances to learn	Lots	Some	Lots	Lots	Few

When looking for companies to work at, you'll find that many of them are similar to these companies in various ways. It can be helpful as you go through job applications and interviews to try and understand what the strengths and weaknesses of working at these different companies are. In later chapters we will discuss how to make the decision on where to work.

2.7 Interview with Randy Au on different types of companies

Randy Au is Quantitative User Experience Researcher at Google, working on Google Cloud. Having worked in data science with a focus on human behavior for over a decade, he blogs about how to think about working at start-ups and different types of companies on Medium at medium.com/@Randy_Au.

2.7.1 What was your path to become a data scientist?

I started out at an interior design firm that helped companies redesign their office. Our clients would tell us, "we need more meeting rooms because there's never enough." So our firm would actually hire some temps and walk around and write down every hour: Who's in this room? Who's at their desk? I was brought on to be the analyst to make sense of all their data and help them pitch their recommendation.

Then I went to Meetup and spent five years as essentially one of their main data people. I worked very closely with quantitative research, I did a lot of product work, and I made friends with everyone in the company. I was helping everyone use data, from customer support and legal to product managers, by setting metrics, running experiments, and making dashboards. Then I did two years at Bitly, where I was also a data analyst in support of product development. There I learned a lot of big data stuff—I used ridiculous amounts of Hadoop, broke a lot of production systems, all that good stuff.

Then I spent a year at Primary, a company that designs, manufacturers, and sells children's clothes online. It was a very small startup and I was their data engineer, doing a lot to make them data driven. And then, after all of that, I wound up at Google doing user

research. The one thread that connects all this is that I spent a lot of time helping teams do product work and a lot of time helping them figure out how to use their data.

2.7.2 Are there big differences between large and small companies?

Yes—usually it's more organizational and structural. There are points in a company where culture changes because of the scale. At a 10-person start up everyone does everything because everyone's wearing all the hats. If you're in that kind of situation you wind up having to be more things. Meanwhile, around 20 people things start specializing. You start getting three or four-person teams dedicated to specific things. There's a little bit more organization around there, so the dynamics shift a little bit. People can think harder about certain things and you don't have to learn everything about a company. At around 80 to 100 people the existing teams don't scale anymore. Then there's a lot more process around things. You don't know everyone in the company anymore. You don't know what everyone's up to, and so there's a lot more overhead to reach common understanding. Beyond that, after about 150-200 people, it's impossible to know what's going on around the company, so the bureaucracy has to exist. Then you go to Google, which is 100,000 people. There you have no idea what most of the company is doing.

The smaller the company, the more likely you're going to interact with everyone in the company, right? At a 40-person company I would have the CEO of the company sitting at my desk as we're both exploring a data set together. I can say, "I've got something cool I want to show you; come look at it." Then they start asking questions and you start writing code right in front of an executive. That will never happen in Google. But are you okay with the situation that happens in a lot of start-ups where you're building an F1 car and you're driving it at the same time and everyone's arguing whether you should have a steering wheel?

When you're the "data person" at a small company, the methods don't really matter as much – you're just trying to squeeze all the data and get some insights out of it. It's okay to not be as rigorous so you can make decisions more quickly. A lot of teams also won't know what they want out of their data, they just know that they have it and it has to be valuable somehow. You'll need to teach them what you can and cannot do with this data, how to collect it, how to make smart decisions with it, and how to avoid traps.

2.7.3 When you got to Google after working at start-ups, were you surprised by the differences?

Sort of. Interestingly, the thing about Google Cloud is we act like a start-up because it has to move so quickly. They're making so many products that it's the situation at most start-ups where you don't quite know what you're doing all the time. You're experimenting and building it out. It's very true of a lot of the newer products of Google Cloud. The difference is that we have a billion dollars in revenue for our thing because it's so huge, but at the same time there is a list of a thousand features that we want to do. How do we prioritize them? What are users doing? Who are our users? Those questions come up all the time still. There's a lot more

bureaucracy and structure that you have to understand—I guess that's the big shock. But it varies a lot.

2.7.4 Are there differences based on the industry of the company?

Sort of. Some industries have historically had math or data people. For example, an insurance company has actuaries. Those people have been around for a hundred years and they really know their stats. If an insurance company is going to bring in data scientists, they come with a slightly different view on it. They already have this built in structure for extremely talented stats people. They're going to be filling a gap somewhere else: there's going to be a gap in their big data or in optimizing their website or something.

Finance also has a long tradition of having quants. I remember failing a quant finance interview once because they did a code test. But as a data scientist I just make sure my code is functional and gives the correct answer; I don't think too hard about performance until it becomes a problem. Their coding test literally tested you on performance and dings you points for not being performant automatically. I was like "oh yeah, you guys are in finance I get it." I totally just bombed that interview—it was funny.

2.7.5 Should new data scientists be wary of startups since they'll do tons of cleaning, pipeline creation, and unglamorous work?

I think if you talk to everyone who's doing data science work, the vast but silent majority are people who are doing this kind of grunt work that's not sexy at all. I got a ridiculous amount of responses to the article I wrote about data science at startups that were people saying: "Yeah, this is my life." This is not what people talk about when they talk about data science. It's not the sexy: "here's a new shiny algorithm I applied from this arXiv paper." I don't think I've applied anything in an arXiv paper in the twelve years I've worked. I'm still using regression because regression really works! I think that is the reality of it.

You're going to be cleaning up your data; I don't think there's anyone even at the Facebooks and the Googles who doesn't have to clean data. It might be slightly easier to clean up your data because there's structure around it. But no, you're going to have to clean up your data—it's a fact of life. If schools and people are willing to admit that a fair amount of this is grunt work but that's life—I think that's a better way to approach it. That said, having some structure and leadership in place as a resource for you is going to make your life infinitely easier.

2.7.6 What's your final piece of advice for aspiring and junior data scientists?

Know your data. This does take a long time - six months to a year or more if it's a complicated system. But your data quality is the foundation of your universe. If you don't know your data, you're going to make a really bizarre statement about something that your data just can't let you say. For example, some people will say, "Oh, I have the number of unique cookies visiting my website, and that's equal to number of unique people." But that's not true - what about

those people who are using multiple devices or browsers? You'll be the person who has a meeting and presents a cool finding, but then someone who has domain knowledge is going to shoot it down by asking, "Have you thought about how VPNs affect this?"

To really know your data, you need to make friends with the people with domain knowledge. For example, when I was doing financial reports, I made friends with the finance people so I could learn the conventions accounting has about how they name things and the order of how things are subtracted. When I found some weird data, I would talk to an engineer to figure out whether it was real or a bug. Maybe you got fifty million pages from this one IP, and someone else will realize that's IBM. You won't know all this stuff, but someone probably will.

2.8 Summary

- There are many different types of companies that hire data scientists.
- The data science job itself varies, largely based on the company's industry, size, history and team culture.
- It's important to understand what kind of company you are considering working at.

3

Getting the Skills

This chapter covers

- The different methods to learn data science
- Understanding what makes a good academic program or boot camp
- Choosing the route best for you

Okay, you want to be a data scientist. Great! Now… how do you become an expert in all those skills?

Fear not - wondering how to learn the skills of a data scientist is one of the universal parts of becoming a data scientist. There are so many ways to do it, from watching YouTube videos to getting a degree, and lots of people will tell you their way is the only correct path. Worse, it's easy to feel overwhelmed by the amount to learn: there are hundreds of algorithms plus programming languages, and then you throw in all the business demands. Just thinking about it can be emotionally draining.

The good news is that there are really only four main methods to get the skills you need. Each method has its advantages and drawbacks, but once you lay them out it usually becomes clear which is the right approach for you. By the end of this chapter you should understand the different methods, and after some reflection you should be able to decide the best route for your situation. You can do this!

The four methods for gaining data science skills that we will cover are:

1. Earning a graduate degree in data science or a related field.
2. Participating in a data science boot camp, an 8-12-week crash course in data science.
3. Doing data science work in your current job.
4. Teaching yourself through online courses and through data science books.

Let's walk through each of them.

3.1 Earning a data science degree

Many colleges now offer graduate degrees in data science, and the programs cover a mixture of topics from computer science, statistics, and the business school. Since they are master's degree programs, they generally take two years and cost up to $100k or more. As with most graduate programs, you can often choose to go through the program more slowly while still having a job and/or potentially take them as online courses. While many schools offer an actual data science degree, depending on your interests you may alternatively get a degree in computer science, business analytics, operations research, or something very close to data science.

The good thing about a data science degree is that it's exhaustive–due to the length of the program and the amount of time you spend in it, you should learn everything you need to be a junior data scientist. Through coursework and projects, you'll get experience using the methods and hands-on programming. If you come into the program without much programming experience you should be able to pick it up on the way (although you may have to take an extra course or two).

Graduate degrees to learn data science have a few downsides. First, they are extremely expensive, both in terms of the cost of tuition and the opportunity cost from not earning income and gaining direct work experience during the time you are a full-time student. Graduate programs are an order of magnitude more expensive than the other options both in money and in time. Spending years studying before you feel ready to switch careers is a huge amount of time in your life, and if you decide part-way through you don't want to be a data scientist you can't get that money or time back.

If you are coming from a background that's related to data science, like software development, or have substantial undergraduate coursework in the field, a graduate program will teach you a lot of stuff you already know. That means from a long program you may only get a small amount of useful, new information. This can be a huge drawback and make the program feel frustrating.

Also, these programs are taught by academic professors. Many academic professors have spent their whole career in academia, making the material they teach potentially different than what people use in industry. That means that a particularly disengaged professor may do things like use old languages such as SPSS, or they won't understand modern tools like version control. This is especially common in degree programs outside of data science. Some universities bring people from industry in to teach courses, but then they may or may not know much about teaching. It's difficult to tell how much of the program uses modern techniques until you're in it. During the application process, try and find opportunities to talk to current or former students to get a feeling for the program and how useful it is for a career.

3.1.1 Choosing the school

As you start searching for graduate data science programs, you may be quickly overwhelmed by the number of options you have. Worse, you may find yourself getting your mailbox full of

flyers for different programs, and calls from recruiters. Depending on how much work you want to do, you should likely apply to somewhere between three and ten of these programs. Apply to too few and it's possible you won't get into any; apply to too many and you'll find yourself devoting excessive time (and application fees) to graduate applications. To decide which schools to apply to, consider these different metrics:

- If you'll be happy with the location and lifestyle [very important] – you'll likely be looking at graduate programs all across the country, but what your life looks like outside of school will be quite different if you go to a school in Arizona versus upstate New York. If the climate, proximity to friends, and cost of living don't work for you, it doesn't matter how good the program is because you'll be unhappy doing it.
- What topics the program's coursework covers [important] – because data science is so new, universities may have dramatically different coursework. This is especially complicated by which department the program lives in; a CS-based data science program is going to be focused on methods and algorithms while a business school program will be focused more on applications and rely on case studies. Check that the course material covers weaknesses in your skillset from Chapter 1.
- How much project work the program has [important] – the more project work a program has, the more you'll learn about how data science works in practice and you'll be better prepared for industry (projects are covered further in the next chapter). Significant projects are also great for putting on your resume, which can help you get an internship during the graduate program or help get that first job.
- Where graduates end up [important] – often the school will provide statistics on where students go after they graduate. These statistics can be informative, but often the schools will show metrics that make them look the best, even if they are misleading (which ironically is a skill you learn as a data scientist). If possible, try reaching out to some of the program alumni through LinkedIn to get an unbiased perspective on how graduates fare. Especially if you want to work at an MTC, you can look at what companies recruit directly at that school. While of course you can still apply for a job if they don't, your job application may be given less consideration.
- Funding [rare but very important] – in rare circumstances schools will offer funding for master's graduate students. This means the school will pay for your coursework and pay a stipend, usually provided you are a teaching assistant for a class. If you get this we highly recommend you take it—not having to pay for your degree and getting a salary to boot is financially a much better option than paying yourself. If the funding involves teaching, you'll also have the benefit of being forced to learn how to communicate with a roomful of people which will be helpful in your data science career. The downside is that teaching takes lots of time, which distracts from your studies.
- How much the program is connected with businesses in the area [medium] – if the school does a lot of work with local companies, especially tech ones, it shows that the school is connected with the community. That'll make it easier to get an internship or a

job, and will give you more interesting materials during classes. It also lowers the chance of having professors who are out of touch with the methods used outside of academia.

- Admission requirements [not very important] – some schools have requirements around courses you need to have taken to be accepted. Most programs require that you have some courses in mathematics like linear algebra, and some courses in programming like an introduction to Java. If you are missing one or two of the courses, you may be able to get the requirement waived or take make-up courses once you're in the program. If you are finding you don't have any of the prerequisites, or they require a specific undergraduate degree like Computer Science, the program may not be a good fit for you.
- Prestige of the school [not at all important] – unless you get accepted to an extremely prestigious school such as Stanford or MIT, employers won't care about how well-known the school is. Prestige mainly matters if you are looking to go into academia instead of industry, but in that case you should be getting a PhD and not a master's, and also reading a different book than this one. The only time prestige can be useful is for the strong alumni networks these schools provide.

When coming up with your list of schools, try making an Excel spreadsheet that lists how the different schools fare in each of these metrics. Once you have all the data, you may find it hard to objectively rank the schools: how can you really say if a school in a city you'd hate to live in but is well-connected to industry is better or worse than a school in a city you'd love that has no project work? We recommend you let go of the idea of finding the objective "best." Instead group the schools into "love," "like," and "okay" and then only apply to the loves and likes.

ONLINE GRADUATE PROGRAMS

More and more schools are offering online graduate programs, making it possible to learn everything you need to know without having to walk onto a college campus. The obvious benefit of this program is that having courses online is dramatically more convenient than having to spend hours each week going to a university. There also isn't nearly as much of a stigma from employers around online programs as there was at their inception, so you shouldn't worry if your degree will be viewed as legitimate. The downside is that it is *much* more difficult to stay engaged with the program and the material if you are doing everything online. It'll be harder to interact with your professors when you have questions, and it'll be easier to half pay attention and not do your homework. In a sense, that convenience of an online program can also be its downfall—you don't have as much incentive to stick with it. If you think that you have the ability to stay committed and focused to an online program, then they can be a great choice; just beware of the risks.

3.1.2 Getting into an academic program

To get into an academic program, you need to apply. If you're familiar with applying to graduate programs, the application process for data science master's degrees is similar to the

rest. The first step is to write your application. Schools typically announce in the fall how to apply, including deadlines and materials needed. Graduate applications usually require:

- A 1-2-page letter of intent describing why you are a good fit for the program. For this letter, focus as much as possible on why you would make a good contribution to the program. Things like existing experience in some skills required by data science or examples of work you have done that is related are extremely useful. Try to avoid cliché statements like "how you have been interested in data science ever since you were a child." There are lots of resources available on writing good graduate school essays, and your undergraduate school may also have a department to help with this.
- A transcript from your undergraduate school. This is to show that you have the necessary prerequisites for the program. Your school's website should have instructions on how to get this, but keep in mind it usually costs money and takes a week or more. Don't leave it to the last minute!
- You need to take the GREs and hit some minimum scores in verbal and math. The mathematics GRE should be relatively easy for anyone going into a data science program, but definitely spend time reviewing for it since you may have not seen the material in years. The verbal can be harder and may require substantial studying. The GREs require you go to a special location to take the exam and can be a pain to schedule, so be proactive and try to get this done early. If English is not your native language, you'll probably need to get a minimum score on the TOEFL or IELTS.
- Three letters of recommendation on why you would be good for this graduate program. These can be from professors you've had or people like your boss if your job is tangentially related to data science. Ideally the writers should be able to talk about why you would be a good data scientist—so the people should have seen you perform well. Try to avoid college professors who can only say "this person got an A in my class" and employers who can't say much about your work in a technical environment. If you're an undergraduate student and you're reading this book, now may be a good time to get to know professors better by going to office hours, attending seminars, and joining academic clubs.

These different materials take time to pull together, and if you are applying to many schools at once it can end up being a full-time job. Most applications are due between December and February, and you then hear back around February or March. If you get accepted, you have until April to decide if you want in the program. When your acceptances come in, don't worry too much about which is exactly the best, just choose one that you think you'll be happy at!

3.1.3 Academic degree summary

Putting it all together, graduate programs in data science are a good fit for people who want an extensive education and can afford it. That could be a person who is coming from a field where they haven't done much programming or technical work, such as person who has been working in marketing. A graduate program would allow the person to learn all of the

components of data science at a pace that would make the transition reasonable. Graduate programs are not good for people who already have a lot of the required skills. They are much too long and too expensive to be worth it. They also aren't taught by industry experts, so the little new knowledge they learn may not even be that relevant. You may need to get industry experience from internships during your graduate program to augment the degree itself.

If you're thinking you need extensive training before you could be a data scientist, then go for it and start looking for graduate schools you like. If you feel like this is going to be a lot of work and there's got to be an easier way, then consider the next few options.

3.2 Going through a bootcamp

A bootcamp is an 8-15 week intensive course put on by companies like Metis and Galvanize. During the bootcamp you spend 8+ hours every day learning data science skills, listening to industry speakers, and working on projects. At the end of the course, you'll usually present a capstone project to a room full of people from companies looking to hire data scientists. Ideally, your presentation gets you an interview and then a job.

Bootcamps teach you an incredible amount of knowledge in a very short time. This means they can be great for people who have most of the skills needed for data science, but just need a bit more. For instance, consider someone who has been working as a neuroscientist and has done programming as part of her work. A data science bootcamp could teach her topics like logistic regressions and SQL databases. With her science background plus those basics, she should be ready to get a data science job. Sometimes the best part of a bootcamp isn't the knowledge itself, but the confidence you get from the program that you can actually do the work.

3.2.1 What you learn

A good bootcamp will have a syllabus that is highly optimized to teach you exactly what you need to know to get a data science job and little more. That goes beyond just technical skills and includes opportunities to work on projects and network with people. Here is more detail on what you should expect the program to cover.

SKILLS

Bootcamps are a great supplement to an existing education. For example, if you are someone who has been working as a software developer for years, a bootcamp can quickly fill in the details you need around the math and statistical techniques and how to think about data. By doing a bootcamp you'll be able to get a data science job quickly, without spending two years in a program like you would if you were to seek a master's degree. This might be especially attractive if you already have a master's degree in a non-data science field. The skills you typically get are:

- Introductory statistics – this includes methods for making predictions with data such as

linear and logistic regressions, as well as testing methods you could use on the job like *t*-tests. Due to the very limited time range, you won't get very deep into why these methods work, but you'll learn a lot on how to use them.

- Machine learning methods – you'll cover machine learning algorithms like random forests and support vector machines, as well as how to use them by splitting data into training and testing groups or using cross validation. You may learn algorithms for more specific cases like natural language processing or search engines. If none of those words made sense to you, that means you could be a good fit for a bootcamp!
- Introductory programming in R or Python – you'll learn the basics around how data is stored in data frames, and how to manipulate it by summarizing, filtering, and plotting data. You'll learn how to do the statistical and machine learning methods within the chosen program. While you may learn R or Python, you probably won't learn both so you may have to learn the other after you finish the bootcamp if you need it for your first job.
- Real-world use cases – You'll not only learn the algorithms but also where people use them. Cases like using a logistic regression to predict when a customer will stop subscribing to a product, or how to use a clustering algorithm to segment customers for a marketing campaign. This knowledge is extremely useful for getting a job, and questions around use cases often show up in interviews.

PROJECTS

Bootcamps have a highly project-based curriculum. Instead of listening to lectures for 8 hours a day, most of your time will be spent working on projects that will best help you understand data science and get you started with your own data science portfolio (the subject of Chapter 4). That's a huge advantage over academia because your skills with be aligned with what you need to succeed in industry, since that's often similar to project-based work.

In a project, you'll first collect data. That could be through using a web API that a company has created to pull their data. It could also be through scraping websites to collect the information off them or using existing public data sets from places like government websites. You'll then load them into R or Python and write scripts to manipulate the data and run machine learning models on it. Then you'll use the results to create a presentation or report.

None of those steps in the project require a bootcamp (in fact Chapter 4 of this book exists just to help you do this yourself). That being said, having a project be part of a bootcamp means you'll have instructors guiding you and helping you if things go wrong. It's difficult to stay motivated if you are working alone and easy to get stuck if you don't have a person to call for help. That makes bootcamps very valuable.

A NETWORK

Lots of people go on from bootcamps to successful careers at places like Google and Facebook. The bootcamps keep alumni networks that you can use to get your foot into the

door at those companies. The bootcamp may bring in data science speakers to talk to you during the program. There are also people from industry viewing your final presentations. These people can also serve as connections to help you get a job at one of their companies. Having points of entry to companies with data science positions can make all the difference when it comes to finding jobs, so this perk of bootcamps must be stressed.

In addition to meeting people during the course itself, you can also use tools like LinkedIn to contact alumni from your bootcamp. They may be able to help you find a job at their company, or at least point you in the direction of a company that would be a good fit.

For all of these connections, you'll have to be proactive. That means taking actions like going up and talking to speakers after they present and taking the initiative to send messages on social networks with people you haven't talked to before. This can be scary, especially if you aren't especially comfortable with social interaction with strangers, however it's necessary to get the value out of the bootcamp.

3.2.2 Cost

One significant downside to the bootcamp compared to self-teaching is the cost: the tuition is generally around $15,000-$20,000. While you may be able to get scholarships to cover part of the tuition, you also have to consider the opportunity cost of not being able to work full-time (and likely even part-time) during the program. Moreover, you'll likely be on the job market for several months after your bootcamp. You won't be able to apply during the bootcamp because you'll be too busy and won't have learned the skills yet, and even a successful data science job application process can take multiple months from application to starting date. That can end up being an aggregate of 6-9 months of unemployment, in addition to the cost of the program. If you are able to teach yourself data science in your free time, or learn on the job, then you can keep working and not paying tuition, saving tens of thousands of dollars.

3.2.3 Choosing a program

Depending on where you live, there are likely only a few options for bootcamps. If you want to do an in-person bootcamp, even if you live in a large city there will probably be only a handful of programs. If you don't live in a large city and want to do a bootcamp, you may have to temporarily move to one. That can add to the cost of the program and make it more of an upheaval. Alternatively, there are online bootcamps for data science. Be careful, however: like with graduate programs, one of the benefits of in-person boot camps is that you'll have people around you to motivate you and keep you focused. If you do an online course you lose that benefit, which can make an online bootcamp a $20,000 version of the same courses you could get through free or cheap massive open online courses.

In selecting between the bootcamps in your area, consider checking out their classroom, talking to some of the instructors, and seeing where you feel the most comfortable. But beware: with both academic degrees and bootcamps, there are lots of people looking to make a quick buck on people wanting to become data scientists. If you aren't careful, you can end

up completing a program that doesn't help you get a job at all and leaves you with tens of thousands of dollars of debt. For bootcamps, it's extremely important that you talk to alumni. Do you see successful graduates on LinkedIn? If so, talk to them and see how they feel about their experience. If you can't find people on LinkedIn from the program that's a huge red flag.

3.2.4 Data science bootcamp summary

Bootcamps can be great programs for people wanting to switch careers and already have some of the basics of data science. They can also be useful for people just leaving school and wanting a few data science projects in their portfolio when on the job market. They are not designed to take you from "0 to 60" though; most of them have competitive admissions and you need to have a background in the fundamentals of programming and statistics to get in and then get the most out of it.

3.3 Getting data science work within your company

You may find yourself in a data science adjacent job. An unusual, but often very effective, method of learning data science is to start doing more and more data science work as part of your current job. Maybe you are in business intelligence, set up databases that data scientists use, and could start doing queries. Maybe you are a business person who takes the data science reports, already add a business spin, and could start also adding your own graphs. Maybe you work in finance making spreadsheets that you could move into R or Python.

For example, consider Alex, a person who has been working for several years in the market research department running surveys on customers and using a market research GUI (Graphical User Interface) to aggregate the survey results. Alex has a background in sociology and did a small amount of programming during their undergrad. Alex frequently works with the data science department–Alex passes them survey data and helps the data scientists understand it so they can use it in models. Over time, Alex starts to do a bit of work for the data science team. A bit of feature extraction in R here, a bit of creating visualizations there, and soon the data science team is relying more and more on Alex. During this time Alex is really improving their programming and data science skills, and after a year Alex fully joins the data science team, leaving market research behind.

Trying to do some data science in your current job is a great method because it is low risk and has built in motivation. You aren't trying to do an expensive bootcamp or degree that you have to quit your job for, you're just trying to add a little data science work where you can. And the fact that you are doing data science in your own job is motivating because the work you'll do is valuable to others. Over time you can do more data science work until eventually it's all you do, rather than trying to do an educational program and suddenly switch all at once.

Our former market researcher and now data scientist Alex had a number of things going for them in our example. Alex already had existing relationships with the data science department to provide mentorship. Alex also understood the basics of programming and data

visualization. Alex was self-motivated enough to learn data science techniques within their job. The data science department was able to provide Alex with small projects that Alex could tackle, and over time these grew to enable Alex becoming a data scientist. When trying to do more data science work in your company, look for places where you can find small data science projects and people to help you with them. Something as simple as creating a report, or automating an existing one, can teach you a lot of data science.

One important note when taking this path: *never become a burden for someone else*. That could be obvious, e.g. direct burdens might be repeatedly asking people to send you cleaned datasets. Or it could be less obvious things like constantly asking someone to review work you've done. You can also be an accidental burden by adding new tools to your team–if you're in finance and everyone uses Excel except for you who now uses R, you've just made managing your team more complicated. Even the task of asking a data scientist for work you can help with can be a burden. Just be thoughtful as you learn these skills that you aren't creating issues for other people.

TWO CONVERSATION PERSPECTIVES

What you say: "I am happy to help in any way I can, just let me know how! Thanks!"

What you think they hear: "I am a person who is eager to work for you, and you can hand me that exciting but simple project you've held onto for so long and I will do it for you!"

What they actually hear: "Hi! I want to be helpful but I have no idea what your needs are. I also don't know what my skills are relative to your workload, so good luck finding a task for me to do. Also, if you do somehow find a task that's perfect for me, you'll probably have to review it a bunch of times before it's good, all of this taking away from your already minimal free hours. Thanks!"

To make this path work there are a few key strategies to employ:

- **Be proactive** - the more you can do work before people ask for it, the more you'll become independent and less of a burden. For instance, the data science team may have a task, like labeling data or making a simple report, which is time intensive and uninteresting. You can offer to help with that work. Be careful just diving in entirely and doing it yourself–you may end up doing it in a way that instead of providing any value just provides the team the opportunity to redo your work. But if you can get it started and then get their input, it's possible you can save them a lot of time.
- **Pick off new skills one at a time** - don't just try to become a data scientist all at once. Find a single skill you want to learn through work and then learn it. For instance, you may want to learn how to make reports with R since the data science team does that all the time. By finding a small project to help the team with, you can pick up the skill and add it to your tool belt. From there you can learn a different data science skill and gradually learn enough that way.
- **Be clear with your intentions** - it'll be pretty obvious fairly quickly that you're trying

to pick off extra work to learn to be a data scientist. By being proactive and letting the data science team know that you're interested in learning more, the team will be able to plan around having you help. It'll also make the team more understanding of your inexperience, since they were learning and new once too.

- **Avoid being pushy** - helping a person become a data scientist is an immense amount of work, and data science teams are often already overworked. If you find that the team doesn't have the time or bandwidth to help you, don't take it personally. While it's okay to occasionally check in if you think they've been out of touch, if you are too persistent with your requests the team will quickly become uncomfortable. They'll view you less as a potential resource and more as a nuisance.

3.3.1 Learning on the job summary

Learning on the job can be an effective way to become a data scientist, provided you have a job where you can apply data science skills and people around who can mentor you. If those things align then this is a great route, but for many these things are not in place. If you do think this is a viable route for you, we highly recommend taking it–it isn't often that jobs allow for this so take the opportunity if you have it.

3.4 Teaching yourself

There are an enormous number of books covering data science [glances left, glances right…], as well as online courses on data science topics on websites. These books and sites promise to teach you the basics of data science as well as the in-depth technical skills through a medium (and at a price point) that is practical. These courses and books, as well as all of the data science blogs, tutorials, and Stack Overflow answers, can provide enough of a grounding that people can teach themselves data science.

These self-driven learning materials are great for picking up individual skills. For instance, if you want to understand how to do deep learning, a book can be a great way to do it. Or if you want to get the basics of R and Python, you can take an online course to get started with those.

Teaching yourself data science entirely through self-driven online courses and books is just like teaching yourself to play an instrument through YouTube videos or learning anything else without a teacher: its value is mostly a function of your perseverance. According to Malcolm Gladwell and all the people who repeatedly quote him, mastering a new skill takes 10,000 hours. It's really hard to put 10,000 hours into data science where the best Vine compilations are one tab over. It's also hard to know where to start; if you want to learn everything in data science who is to say which book you should read first [glances right, glances left…].

Learning on your own means you don't have a clear teacher or role model. By not having a teacher you can ask questions as you would in an academic program or bootcamp, you won't have an easy ability to understand when you're doing something wrong or know what to do next. This again places a hurdle to understanding the material. The best way to counteract

this lack of a teacher is to find a community of people where you can ask questions. One great example is the TidyTuesday program started by Thomas Mock, in which every Tuesday aspiring and junior data scientists use the Tidyverse R packages to tackle a data science problem.

If you do decide to go down the self-driven route, it's important to have some constructive work you can do. While reading books and watching videos is great, you learn far more from doing your own data science work and learning from it. To put it a different way, reading books on bicycles can be educational, but you'll never learn to ride a bicycle without actually getting on one. So make sure to find a project that you want to do: taking a dataset and finding interesting results from it, creating a machine learning model and API, or using a neural network to generate text. In Chapter 4 we will go into far more detail on doing these sorts of projects. For other methods of learning data science, the projects can be about building a portfolio, but for self-driven learning, the projects are critical to learning.

3.4.1 Self-teaching summary

Learning on your own is hard; possible, but hard. You need to be able to figure out what order you want to learn things in, keep yourself motivated enough to learn the skills, and do it all without a clear mentor or teacher to help you. You'll also have more of a difficult time displaying your qualifications on a resume than if you used other methods. This is our least recommended way to become a data scientist, because of how many things can go wrong and how many people don't succeed in staying focused. If you want to pick up a single particular skill or technology it can be more feasible, but learning everything you need to be a data scientist is a hard route.

3.5 Making the choice

With those four very different avenues to data science, how do you pick? The process is different for everyone, but we propose answering these three questions:

1. **Do you have some data science knowledge already?** Specifically, have you programmed in at least one programming language beyond light coursework? Do you know how to query data from a SQL database? Do you know what things like linear regressions are?
   a. If NO I'VE GOT A LOT TO LEARN, you'd probably be best suited for an academic program like a master's degree. This will teach you all of these topics over a long enough time period that they will really settle in.
   b. If YES I KNOW THIS STUFF, move on to question two.
2. **Are you comfortable with taking a year or more to gain the necessary data science skills, rather than incur the costs of being unemployed for six – nine months in order to get a data science job more quickly?** It's difficult to quickly learn new skills when you're solely focused on learning; it's even harder to do so while

also working in a full-time job. Are you okay with the path taking longer so you can keep working full time?

   a. If NO I'VE GOTTA GO FAST, then take a bootcamp. In three months you'll know a ton of data science and be ready to embark on your search for a new job, which could take 3 – 6 months longer.
   b. If YES I WANT TO TAKE MY TIME, then move on to question three.

3. **Can you learn data science in your current job?** Can you do data science things within your current role? Maybe make an analysis, store some data in SQL, or try using R or Python? Is there a team that could mentor you or give you some small tasks?
   a. If YES I CAN LEARN AT WORK, then try doing that and use your job as a springboard to data science.
   b. If NO MY JOB HAS NO OPPORTUNITIES, then it's time to hit the books and online courses.

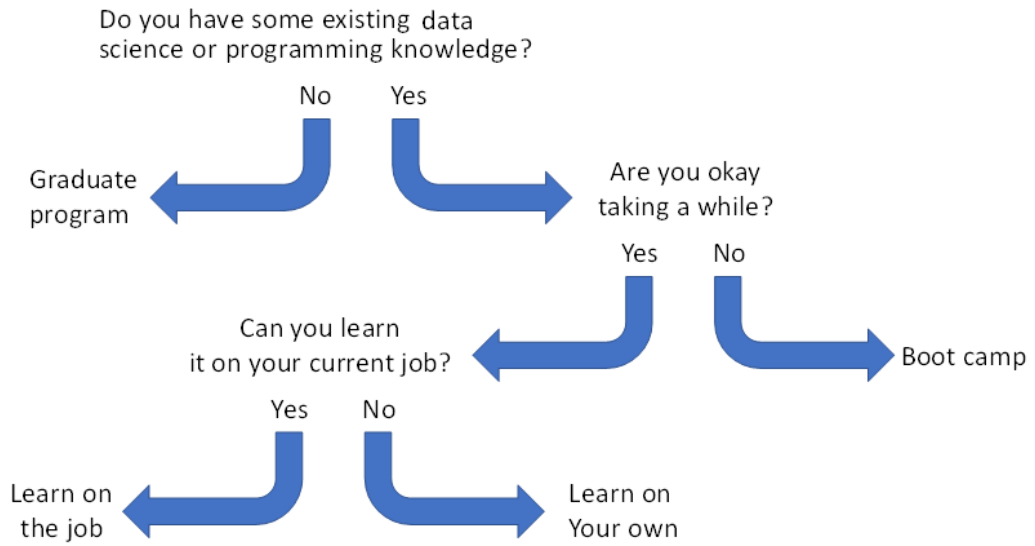


Figure 3.1

While these questions should provide you with a starting point, you don't have to make one final decision. You can start by reading books independently, and if you find you want to go quicker switch to a bootcamp. You can also get a master's degree in the evening while trying to do data science in your current job. There isn't one perfect answer, what matters is you find an approach that works for you, and if something isn't working you change it until it does.

Once you've chosen your route, it's time to do it! Enroll in that master's, join that bootcamp, or buy those books and start reading. For the purposes of this book we'll assume *~time passes~* and that you've succeeded in learning the fundamental skills you need to be a

data scientist. In the next few chapters we'll take those skills and use them to create a data science portfolio and use that to help you get your first data science job.

3.6 Interview with Julia Silge on her transition to data science

Julia Silge is a data scientist at Stack Overflow. Her blog posts on data science and natural language processing are highly read, and the tidytext package she and David Robinson created is a cornerstone of NLP in R, downloaded more than 300,000 times. She and David also coauthored the book *Text Mining With R: A Tidy Approach* with O'Reilly. We talked to Julia on how she earned a PhD in Astrophysics and spent time in academia, then left and had a "meandering" career path which led her to be a data scientist.

3.6.1 Before becoming a data scientist you worked in academia; did the skills from your time there help you as a data scientist?

Stack Overflow is my second job as a data scientist. When I think about what I spend my days doing I absolutely see connections back to my days in grad school. I see connections in the kind of code I write, in the kind of writing I do, and in the kind of conversations I have with people. When I taught in grad school I taught physics and chemistry to undergrads and I see strong connections to that in my data science work. As a data scientist I see a lot of direct connections to daily tasks, skills, themes back to stuff I have been doing my whole career before.

One of the places that I think "oh I have used that skill before" was when I was still involved in research as an academic. Some of my days were spent collecting real-world data and that experience taught me "this data was created by a *process*." In that case it was created by a physical process that I could touch. I could actually see things that contributed to why the data was messy, or why we didn't get a datapoint in a particular night. I see direct parallels to now that I work at a tech company that deals with web data—there is some process that generates that data, and I think carefully about how we are recording that data and how that process can go well or wrong.

Another set of skills that I see I learned before being a data scientist was communicating and teaching. I was a college professor for a number of years, and I also have had roles where I have dealt with customers. In those roles I practiced the skill of having a concept and trying to transfer that knowledge or understanding to another person. I strongly believe that's a part of most data scientists' role. If we train some model or do some statistical analysis, the value of it is small compared to when we are able to take that same model or analysis and explain what it means, how it works, or how to implement it in the broader context. That ability to communicate what's going on and what it means is huge.

3.6.2 When deciding to become a data scientist, what did you use to pick up new skills?

I certainly think academic programs, bootcamps, and online materials are all great options for different people in different situations. Given that I already had a PhD, I didn't want to go back to school and spend more money. I will admit that I applied to a couple of bootcamps and they took a pass on me! When I was deciding to make this career transition into data science, what I perceived was that I could do that job, but I needed to demonstrate to other people that I could. I also needed to update my machine learning knowledge and some of the techniques because when I was in grad school modern machine learning had not really entered astrophysics.

I went the route of online courses and embracing a lot of self-directed study. I joke around that I took every MOOC (massive open online course) that existed: it was a LOT. I took about six months where I was slightly under-employed in the job I had, and I just hit the MOOCs *really hard*.

I had been out of school for a long time and I had been excited about the material. I hadn't been doing even data analysis for a while, so getting back into data analysis was really exciting! I didn't see myself doing it for forever, I was like: "I am going to take six months and am just going to go all out." I was excited and saying things like: "wow, did you know you can code a neural network from scratch!? In OCTAVE!?" I wasn't able to learn data science within the job I had at the time. When I talk to other people, especially people who have more classic software engineer roles, I love to tell people to look for opportunities to use data science techniques where they are now. But for me my particular role was not good for that.

3.6.3 Were any online courses particularly helpful for you?

I will give a shout-out to the Coursera/Johns Hopkins R for data science series. I literally did not know there was such a thing as R before that series. And now look at me; I have a whole career because of that course! The democratization of these kinds of courses is kind of mind blowing, but there are caveats to that. I had a PhD already, I knew how to learn something new, I had a lot that allowed me to be exposed to these courses and say "fantastic, time to learn machine learning." There is a lot of privilege that made it possible there, so I don't want to overemphasize the power of these courses.

3.6.4 Did you know going into data science what kind of work you wanted to be doing?

When I looked at my options ahead of time and saw what people were doing, like how people talked about "analyze" vs "build" data science, I absolutely saw myself as an analyze person. I saw myself less of an engineer and more of a scientist—a person who works to understand things and answer questions but not so much build things. And that has been where my career has started. I am the only data scientist at Stack Overflow and I am on a team with very talented, knowledgeable data engineers. But being the only data scientist, my job has shifted

to be a bit of both data analysis and model building–I do more machine learning and model construction than I initially did. When I think about how much time I spent on inferential questions of "what is happening" or "why is this happening", versus predictive questions like "we need a prediction for this user," I have shifted towards doing more prediction than I did earlier in my career.

3.6.5 What would you recommend to people looking to get the skills to be a data scientist?

One thing that I would strongly emphasize is that you need to demonstrate that you can do this job. That can look different for different people. It's still a young enough field that people aren't sure what it means to be a data scientist and who can be one—it's still very undefined. There is still a lot of uncertainly on what this role means, and the positions are highly paid enough that the perceived risk to a company hiring wrong is very high, so companies are very risk adverse. Companies need to be sure that the candidate can do that job. Some ways I've seen people demonstrate that they can do the job is through open source contributions, speaking at local meetups on projects they've done, and developing a portfolio of projects on a blog or GitHub profile.

For me, I took all the MOOCs and things I needed to learn and started a blog about all of these projects. What I pictured to myself was that these projects and blog posts would be something that we could talk about in a job interview. Since I didn't have a typical career path into data science, although who knows what that is even, this would provide something that people could look at and talk about with me. That is absolutely 100% the reason why I started a data science blog. In the course of interviewing for my first data science job I noticed a huge difference between people who looked at my website and who hadn't. The people who were serious about me as a candidate were all people who looked at my blog and had questions about the projects. It was effective as a way to demonstrate that I had some of these skills.

3.7 Summary

- Four proven paths to learn the skills to be a data scientist are academic programs, bootcamps, picking the skills up in your current job, and teaching yourself on the side.
- Each of these methods has trade-offs in terms of the materials taught, the time they take, and the level of self-motivation required.
- To choose a path for yourself, take time to do introspection on what skills you already have, where your strengths lie, and what resources you have.

4

Building a Portfolio

This chapter covers:

- How to create a data science project
- Starting a blog

You've now finished a bootcamp, a degree program, a set of online courses, or a series of data projects in your current job. Congratulations—you're ready to get a data scientist job! Right?

Well, maybe. Part 2 of this book will be all about how to find, apply for, and get a data science position, and you can certainly start on this process now. But there's another step that can really help you be successful—building a portfolio. A portfolio is a data science project (or set of projects) that you can show to people so they can see what kind of data science work you can do.

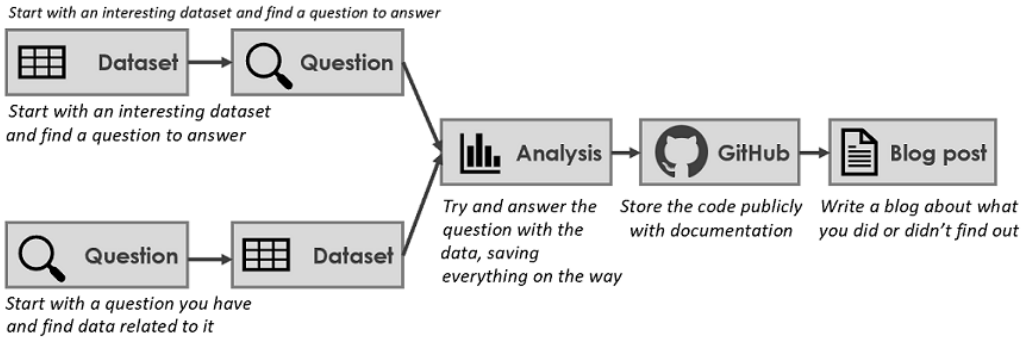
A strong portfolio has two main parts - GitHub repositories, or repos for short, and a blog. Your GitHub repo hosts the code for a project, and the blog shows off your communication skills and the non-code part of your data science work. Most people don't want to read through thousands of lines of code (your repo); they want a quick explanation of what you did and why it's important (your blog). And who knows, you might even get data scientists from around the world reading your blog, depending on the topic. As we'll cover in the second part of this chapter, you don't just have to blog about analyses you did or models you built; you could explain a statistical technique, write a tutorial for a text analysis method, or even share career advice, like how you picked your degree program.

This is not to say you need to have a blog or GitHub repos filled with projects to be a successful data scientist. In fact, the majority of data scientists don't, and people get jobs without a portfolio all the time. But it's a great way to help you stand out and to practice your data science skills and get better. Hopefully it's fun too!

This chapter will walk you through how to build a good portfolio. The first part is about doing a data science project and organizing it on GitHub. The second goes over best practices for how to start and share your blog, so that you get the most value out of the work you've done.

4.1 Creating a project

A data science For example, you could take government census data and want to know, "how are the demographics across the country changing over time?" The combination of question and data is the kernel of the project, and with those two things you can start doing data science.



4.1.1 Finding the data and asking a question

When thinking about what data you want to use, the most important thing is to find data that's interesting to you. Why do you want to use this data? Your choice of data is a way to show off your personality or the domain knowledge you have from your previous career or studies. For example, if you're in fashion, you can look at articles about fashion week and see how styles have changed in the past twenty years. If you're an enthusiastic runner, you can show how your runs change over time and maybe look to see if it's related to the weather.

Something you shouldn't do is use the titanic dataset, MNIST, or any other popular beginning datasets. It's not that these aren't good learning experiences; they can be, but you're probably not going to find anything novel. It won't surprise and intrigue employers or teach them more about you.

Sometimes you let a question lead you to your dataset. For example, you may be curious about how the gender distribution of college majors has changed over time and if that's related to the median earnings after graduation. You then would take to Google and try to find the best source of that data. But maybe you don't have a burning question you've just been waiting to have data science skills to answer. In this case, you can start by browsing datasets and seeing if you can come up with any interesting questions. Here are a few suggestions of where you might start:

- **Kaggle:** Kaggle started as a website for doing data science competitions. Companies would post a dataset and a question and usually offer a prize for the best answer. Because the questions entailed machine learning projects where you're trying to predict something, like whether someone would default on a loan or how much a house will sell for, you can compare models based on their performance on a holdout test set and get a performance metric for each one. Kaggle also has discussion forums and "kernels" where people share their code, so you can learn how others approached the dataset. All of this means that Kaggle has thousands of datasets with accompanying questions and examples of how other people analyzed them.
- The biggest benefit of Kaggle is also its biggest drawback: by handing you a (generally cleaned) dataset and problem, they've done a lot of the work for you. You also have thousands of people tackling the same problem, so it's difficult to make a unique contribution. One way to use Kaggle is to take a dataset but pose a different question or do an exploratory analysis. But generally, we think Kaggle is best for learning by tackling a project and then seeing how you perform compared to others, thus learning from what their models did, rather than as a piece of your portfolio.
- **Datasets in the news**: Recently, many news companies have started making their data public. For example, FiveThirtyEight, a website focused on opinion poll analysis, politics, economics, and sports blogging, publishes data they use for their articles and even links to the raw data directly from the article website. While these data sets often require manual cleaning, the fact that they were in the news means there’s probably an obvious question associated with them.
- **APIs**: APIs are Application Programming Interfaces - they are developer tools that allow you to access data directly from companies. You know how you can type in a URL and get to a website? APIs are like URLs but instead of a website you get data. Some examples of companies with helpful APIs are the New York Times and Yelp, which let you pull their articles or reviews respectively. Some APIs even have R or Python packages that specifically make it easier to work with them; for example, rtweet for R lets you quickly pull Twitter data, so you can find tweets with a specific hashtag, what the trending topics in Kyoto are, or what tweets Steven King is favoriting. Keep in mind that there are limitations and terms of service for how you can use them; for example, right now Yelp limits you to 5,000 calls a day, so you wouldn’t be able to pull all reviews ever. The advantage of using APIs for a project is that they can provide extremely robust, organized data from many sources. The downsides are you often have to think hard about what interesting question to tackle—for example you have 5,000 yelp reviews, now what?
- **Government open data**: There is a lot of government data available online - you can use census data, employment data, the general social survey, and tons of local government data like New York City's 911 calls or traffic counts. Sometimes you can download this data directly as a CSV; other times, you need to use an API. You can even submit freedom of information requests to government agencies to get data that

isn't publicly listed. Government information is great because it is often detailed and on unusual subjects, like data on the registered pet names of every animal in Seattle. The downside of government information is that it is often not well-formatted, such as tables stored within PDFs.

- **Your own data**: There are many places where you can download data about yourself: social media websites and email services are two big ones, but if you use apps to keep track of your physical activity, reading list, budget, sleep, or anything else, you can usually download that data as well. Maybe you could build a chatbot based off your emails with your spouse. Or you could look at the most common words you use in your tweets and how that's changed over time. Perhaps you could track your caffeine intake and exercise for a month and see if it can predict how much and well you sleep.
- **Web scraping**: Web scraping is a way to extract data from websites that don't have an API, essentially by automating visiting webpages and copying the data. For example, you could create a program to search a movie website for a list of 100 actors, load their actor profiles, copy the list of movies they're in, and put that in a spreadsheet. You do have to be careful though - scraping a website can be against terms of use and you can be banned. You can check the robots.txt of a website to find out what they allow. You also want to be nice to websites - if you hit them too many times, you can bring down the site. But assuming terms allow it and you build in time between your hits, scraping can be a great way to get unique data.

What makes a side project "interesting"? One recommendation is to pick a more exploratory analysis where any result will probably teach the reader something or clearly demonstrate your skills. For example, creating an interactive map of 311 calls in Seattle, color-coded by category, clearly demonstrates your visualization skills and you can write about the patterns that emerge. On the other hand, if you try to predict the stock market, you'll likely not be able to and it's hard for an employer to assess your skills if you only have a negative outcome.

Another tip is to see what comes up when you google your question. If the first results are newspaper articles or blog posts answering exactly the question you were asking, you may want to rethink your approach. Sometimes you can expand upon someone else's analysis or bring in other data to add another layer to the analysis. But you may need to start the process over again.

4.1.2 Choosing a direction

Building a portfolio doesn't need to be a huge time commitment. Perfect is definitely the enemy of the good here. Something is better than nothing - employers are first and foremost looking for evidence that you can code and communicate about data. You may be worried that people will look and laugh at your code or say, "Wow, we thought this person may be okay, but look at their terrible code!" It's very unlikely that this will happen. One reason is employers tailor their expectations to seniority level: you won't be expected to code like a

computer science major if you're a beginning data scientist. Generally, the bigger worry is you can't code at all.

This is where it's also good to think about the areas of data science we covered in the first chapter. Do you want to specialize in visualization? Make an interactive graph using D3. Do you want to do natural language processing? Use text data. Machine learning? Predict something.

Use your project to force yourself to learn something new. Doing this kind of hands-on analysis will show you where the holes in your knowledge may be. When data you're really interested in is on the web, you'll learn web scraping. If you think a particular graph looks ugly, you'll learn tweaks to ggplot2. If you're self-studying, it's a nice way to solve the paralysis of not knowing what to learn next.

A common problem with self-motivated projects is over-scoping. This is where you just want to do everything, or you keep adding more stuff as you go. You can always keep improving/editing/supplementing, but that means you never finish. One strategy is to think like Hollywood: sequels. You should set yourself a question and answer it, but if you think you might want to revisit it later, you can end your research with a question or topic for further investigation or even a "TO BE CONTINUED…?" if you must.

Another problem is not being able to pivot. Sometimes the data you wanted isn't available. Or there's not enough of it. Or you're not able to clean it. This is frustrating and it can be easy to just give up at this point. But it's worth trying to figure out how you can salvage it. Do you have enough already to write a blog post tutorial, maybe on how you collected the data? Employers look for people who learn from their mistakes and aren't afraid of admitting them. Just showing what went wrong so others might avoid the same fate is still valuable.

4.1.3 Filling out a GitHub README

Maybe you're in a bootcamp or a degree program where you're already doing your own projects. You've even committed your code to GitHub. Is that enough?

Nope! A minimal requirement for a useful GitHub repository is filling out the README. There are couple of questions to answer here:

- What is the project? For example, what data does it use? What question is it answering? What was the output - a model, a machine learning system, a dashboard, or a report?
- How is the repository organized? This implies, of course, that the repo is in fact organized in some manner! There are lots of different systems, but a basic one is just splitting up your script into different parts - getting (if relevant) your data, cleaning it, exploring it, and the final analysis. This way, people know where to go depending on what they're interested in. It also suggests that you will keep your work organized when you go to a company. A company doesn't want to risk you working there and then, when it's time to hand-off a project, you give someone an uncommented, five-thousand-line script that may be impossible for them to figure out and use. Good

project management also helps the you of the future - if you want to reuse part of the code later, you'll know where to go.

But while doing a project and making it publicly available on a documented GitHub repo is good, it's very hard to look at code and understand why it's important. After you do a project, the next step is to write a blog, which will let people know why what you did was cool and interesting. No one cares about pet_name_analysis.R, but everyone cares about "I used R to find the silliest pet names!"

4.2 Starting a Blog

Blogs allow you to show off your thinking and projects, but they can also offer a non-technical view of your work. We know, we know, you just learned all this great technical stuff! You want to show it off! But being a data scientist almost always entails communicating your results to a lay audience, and a blog will give you experience translating your data science process into "business" language.

4.2.1 Potential topics

Let's say you create a blog. Are people really going to be interested in your projects? You don't even have a "data scientist" title yet; how can you teach anyone anything?

Something good to remember here is that *you are best positioned to teach the people a few steps behind you*. Right after you've learned a concept, like using continuous integration for your package or making a TensorFlow model, you still understand the misconceptions and frustrations you've had. Years later, it's hard to put yourself in that beginner's mindset. Have you ever had a teacher that was clearly very smart and yet couldn't communicate concepts at all? You didn't doubt they knew the topic, but they couldn't break it down for you and seemed frustrated that you didn't just "get it" right away.

Try thinking of your audience as the you of six months ago - what have you learned since then? What resources do you wish had been available? This is also great for celebrating your progress. With so much to learn in data science, it's easy to feel like you're never enough; pausing to see what you've accomplished is nice.

You can group data science blog posts into a couple of types:

- **Code-heavy tutorials:** These teach your readers how to do things like web scraping or deep learning in Python. Your readers will generally be other aspiring or practicing data scientists. While we have called them "code-heavy," you'll usually still want there to be as many lines of text as code, if not more. Code is generally not self-explanatory; you need to walk a reader through what each part does, why you'd want to do it, and what the results are.
- **Theory-heavy tutorials:** These teach your readers a statistical or mathematical concept, like what Empirical Bayes is or how Principal Component Analysis works. They may have some equations or simulations. Just like code-heavy tutorials, your audience

will usually be other data scientists, but you should write so that anyone with some mathematics background can follow along. These are an especially good way to demonstrate your communication skills; there's a stereotype that many technical people, especially if they have a PhD, can't explain concepts well.

- **A fun project you did**: As we hopefully convinced you in the side projects section, you don't need to only work on groundbreaking medical image recognition. You can also find out which of the Twilight movies used only words found in Shakespeare's The Tempest. For example, Julia Silge used neural networks to generate text that sounds like Jane Austen. These can be focused more on the results or the process, depending on what the most interesting part of your project was.
- **Writing up your experience:** You don't just have to blog about tutorials or your data science projects. You can talk about your experience at a data science meetup or conference - what talks you found interesting, advice for people going to their first one, or some resources the speakers shared. This can be helpful to people considering attending the same one the following year or who can't attend conferences for logistical or financial reasons, and again provides an insight for employers into how you think and communicate.

4.2.2 Logistics

But where should you put your interesting writing? For creating a blog, you have two main options:

- **Making your own website**. If you work in R, we suggest using the package blogdown, which lets you create a website for a blog using R code (wild, right?). If you use Python, Hugo and Jekyll are two options, which both let you create static blog websites. Both come with a bunch of themes that other people have built and let you write blog posts in markdown. We suggest not worrying that much about your theme and style and just picking one you like—nothing is worse than not writing blog posts because you got too distracted by the blog's appearance. Simple is probably best; it can be a pain to change your theme, so it's better not to pick one you may get sick of in six months.
- **Using medium or another blogging platform**. Medium is a free, online publishing platform. The company doesn't generally write content themselves; instead, they host it for millions of authors. Medium or other sites like it are good if you just want to start quickly because you don't have to worry about hosting or starting a website – all you do is hit "new post," start writing, and publish. You can also get more traffic when people search the blogging site for terms like "data science" or "python." But one concern is you are at the company's mercy - if they change their business model and put everything behind a paywall, you can't do anything to keep your blog posts free. You also don't get to create a real biography section or the ability to add other content, like a page with links to talks you've given.

How often do you need to post, and how long should posts be? This is definitely a personal choice. We've seen people who have more "micro" blogs where they publish short posts multiple times a week. Other people go months between posts and publish longer form articles. There are some limitations here: you do want to make sure your posts don't start to resemble Ulysses. If your post is very long, you can split into multiple parts. But you want to show you can communicate concisely, as that's one of the core data science skills. Executives and even your manager probably don't want or need to hear all the false starts you had or the twenty different things you tried. While you may choose to include a brief summary of your false starts, you need to quickly get to the point and your final path, and only include them if you ultimately found something. The exception is if the method you used is going to be surprising - for example, you didn't use the most popular library for something because you found it didn't work.

What if you have no readership? Well, one reason to have a blog anyway is that it helps your job applications. You can put links to your blog posts on your resume when you reference data science projects, and even potentially show them to people in interviews, especially if it's a nice interactive visualization or dashboard. It's not important to have hundreds or thousands of readers. While it can be nice if you get claps on Medium or you're featured in a data science company's newsletter, it's more important to have audience that will read, value, and engage with the material than to have high metrics.

That's not to say there's nothing you can do to build a readership. You should also advertise for yourself; although it's a cliché, having a #brand is useful for building a network in the long-term. Even if something seems simple, it's probably new to a bunch of practicing data scientists just because the field is so big. People at the companies you want to work for may even read your stuff! Twitter is a good place to start; you can share when you release a post and use the appropriate hashtags to get some wider readership.

But your blog is valuable even if no one (besides your partner and pet) reads it. Writing a blog post is good practice; it forces you to structure your thoughts. Just like teaching in person, it also helps you realize when you don't know something as well as you thought you did.

4.3 Interview with David Robinson on how he built his portfolio

David Robinson is Chief Data Scientist at DataCamp, an education company for teaching data science through online interactive courses. He co-authored with Julia Silge the tidytext package in R and the O'Reilly book Text Mining with R. He previously worked as a data scientist at Stack Overflow and holds a PhD in Quantitative and Computational Biology from Princeton University. He writes about statistics, data analysis, education, and programming in R on his popular blog, varianceexplained.org.

4.3.1 How did you start blogging?

I first started blogging when I was applying for jobs near the end of my PhD, as I realized that I didn't have a lot out on the internet that showed my skills in programming or statistics. When I launched my blog, varianceexplained.org, I remember having the distinct fear that once I wrote the couple of posts I had ready, I would run out of ideas. But I was surprised to find that I kept coming up with new things I wanted to write about: datasets I wanted to analyze, opinions I wanted to share, and methods I wanted to teach. I've been blogging moderately consistently for four years since then.

4.3.2 Are there any specific opportunities you have gotten from public work?

I did get my first job from something I wrote publicly online. Stack Overflow approached me based on an answer I'd written on Stack Overflow's statistics site. I'd written that answer years ago, but some engineers there found it and were impressed by it. That experience really led me to have a strong belief in producing public artifacts, because sometimes benefits will show up months or years down the line and lead to opportunities I never would have expected.

4.3.3 Are there people you think would especially benefit from doing public work?

People whose resumes might not show their data science skills and don't have a typical background, like having a PhD or experience as a data analyst, would particularly benefit from public work. When I'm evaluating a candidate, if they don't have those kinds of credentials, it's hard to say if they'll be able to do the job. But my favorite way to evaluate a candidate is to read an analysis they've done online. If I can look at some graphs someone created, how they explained the story, and how they dug into the data, I can start to understand whether they're a good fit for the role.

4.3.4 How has your view on the value of public work changed over time?

The way I used to view projects is that you made steady progress as you kept working on something. In graduate school, an idea wasn't very worthwhile, but then it became some code, a draft, a finished draft, and finally a published paper. I thought that along the way my work was getting slowly more valuable.

Since then I realized I was thinking about it completely wrong. Anything that is still on your computer, however complete it is, is worthless. If it's not out there in the world, it's been wasted so far, and anything that's out in the world is much more valuable. What made me realize this is a few papers I developed in graduate school that I never published. I put a lot of work into them, but I kept feeling they weren't quite ready. Years later, I've forgotten what's in them, I can't find them, and they haven't added anything to the world. If along the way I'd written a couple of blog posts, done a couple of tweets, and maybe made a really simple open source package, all of those would have added value along the way.

4.3.5 How do you come up with ideas for your data analysis posts?

I've built up a habit that every time I see a dataset, I'll download it and take a quick look at it, running a few lines of code to get a sense of the data. This helps you build up a little of data science taste –working on enough projects that you get a feel for what pieces of data are going to yield an interesting bit of writing and which might be worth giving up on.

My advice is whenever you see the opportunity to analyze data, even if it's not in your current job or you think it might not be interesting to you, take a quick look and see what you can find in just a few minutes. Pick a dataset, decide on a set amount of time, do all the analyses that you can, and then publish it. It might not be a fully polished post, and you might not find everything you're hoping to find and answer all the questions you wanted to answer. But by setting a goal of one dataset becoming one post you can start getting into this habit.

4.3.6 What's your final piece of advice for aspiring and junior data scientists?

Don't get stressed about keeping up with the cutting edge of the field. It's tempting when you start working in data science and machine learning to think you should start working with deep learning or other advanced methods. But remember that those methods were developed to solve some of the most difficult problems in the field. Those aren't necessarily the problems that you're going to face as a data scientist, especially early in your career. You should start by getting very comfortable transforming and visualizing data, programming with a wide variety of packages, and using statistical techniques like hypothesis tests, classification, and regression. It's worth understanding these concepts and getting good at applying them before you start worrying about concepts at the cutting edge.

4.4 Summary

- Having a portfolio of data science projects shared on a GitHub and a blog can help you get a job.
- There are many places you can find good datasets for a side project; the most important thing is it's something interesting to you and a little bit unusual.
- You don't just have to blog about your side projects; you can also share tutorials or your personal experience with a bootcamp, conference, or online course.

5

The Search: Identifying the Right Job for You

This chapter covers:

- How to find open jobs
- How to decode job descriptions
- How to pick which ones to apply for

You've got the skills, you've got the portfolio – all you're missing is the data science job! In the next four chapters, we'll walk you through each step of the job search process, including finding jobs, applying to them, interviewing, and negotiating and choosing an offer. This process does take some time – even successful job applications usually take at least a month from applying to getting an offer, and more commonly several months. But by following the best practices we lay out, we hope to make the process as painless as possible.

In this chapter, we'll focus on how to look for data science jobs. You'll first learn all the places where you can find jobs, making sure you won't unknowingly narrow your options. We'll cover how to decode these descriptions to find out what skills you actually need (spoiler: it's not all of them) and what the job might be like. Finally, you'll learn how to choose which ones are best suited for you, using the knowledge you've gained about data science skills and company archetypes from the first chapters.

5.1 Finding jobs

Before worrying about crafting the "perfect" resume and cover letter, you need to know where to send them! Job boards like LinkedIn, Indeed, and Glassdoor are a good place to start your search. It's worth looking at more than just one website, since not all companies will post on

each one. If you're part of an underrepresented group in tech, you should also look for job sites targeted specifically at you, such as POCIT and Tech Ladies, for people of color and women in technology respectively. The type of job you're applying to might also influence where you look - there are job boards for specific types of companies like start-ups (AngelList) and technology (Dice).

Make sure to browse widely. As discussed in the first chapter, data science jobs go by many names besides data scientist. Different companies use different names for similar roles, and some are even are changing what their titles mean, so all the people who were data analysts one year might be data scientists the next, with no change in responsibility!

Some examples of titles you might encounter include:

- **Data analyst**: This is more often a junior position and can be a great way to start in the field if you don't have a STEM degree and haven't done any data analysis for a company before. As we'll discuss later in this chapter, you do want to be extra careful with data analyst positions to make sure that the role will involve programming and statistics or machine learning.
- **Research scientist**: These positions will often require a PhD, though there might be some negotiation room if you have a master's in Computer Science, Statistics, or a closely related field.
- **Quantitative, product, research or other non-data analyst**: These roles have even more diversity than data analyst in terms of your responsibilities. You may be doing exactly the same type of work as "data scientists" at other companies, or you may be spending your days with legacy Excel spreadsheets.
- **Machine learning engineer**: As implied by the title, these focus on the machine learning part of data science. They'll usually ask for a strong engineering background – if you have a degree in computer science or have been working as a software engineer, this could be a great role.

When you're first starting your search, try searching for simply "data" on one of these job boards and spending an hour reading through job posts. This will give you a better idea of what industries are represented in your area and what types of positions are open. You'll pick up on patterns that will let you skim through new listings more quickly. Finding jobs that are a good match for you, rather than all the jobs that are available, will narrow the field down to a manageable number. Don't worry too much about the title – use the description to evaluate fit.

Be extremely cautious about thinking of job-hunting as a numbers game. If you're looking in a big tech city like New York or San Francisco or in multiple cities, you'll find hundreds of jobs listed. Checking job boards can quickly become an obsession. It's an easy way to feel productive – "I read through 70 job descriptions today!" And just like Twitter and Facebook, checking constantly for updates can be addictive. But checking more than every three to five days doesn't generally add value. While checking only once a month could mean you miss out

on a good opportunity, we know of no company that has filled a position within two days of posting on a job board.

If you have specific companies you're interested in, check out their careers page. Just as you should search for multiple job titles, check different departments - some companies may put data science under finance, engineering, or other departments, so if you hadn't checked there, you wouldn't find them.

5.1.1 Decoding descriptions

Once you start reading job descriptions, it can seem like data science job postings fall into one of two categories. One is a business analyst position where you'll use business intelligence tools like Excel and Tableau, with maybe a little SQL, but generally you won't code. If you want to grow your coding skills, machine learning toolbox, or statistics and data engineering knowledge, these are not a good fit.

The other listing describes a unicorn - a PhD in Computer Science who's also worked for 5+ years as a data scientist and is an expert in cutting-edge statistics, deep learning, and communicating with business partners - and lists a huge range of responsibilities, running from doing production-level machine learning to creating dashboards to running A/B tests. These types of job descriptions usually mean the company doesn't know what they're looking for, and they expect a data scientist to come and solve all their problems without any support.

Don't worry though – we promise there are more than these two types. The better way of thinking about these jobs is in terms of experience. Are they looking for someone to build a department of their own with no data pipeline infrastructure in place, or are they looking for a fifth member of a currently productive data science team where they hope that person contributes immediately, but isn't expected to be an expert in data manipulation, business communication and software developer all at once? To do this, you need to take a job description and figure out what they're actually looking for. Imagine if you were looking at cat adoptions and Ser Pounce "likes asking about your day" – you'd want to know that probably means he will constantly meow for attention. By knowing how to read between the lines, you can make sure you'll be applying for the right jobs.

The first thing to keep in mind is that job descriptions are generally wish lists with some flexibility. If you meet 60% of the requirements (e.g. you're a year short of their required work experience or haven't worked with one component of their tech stack), but are otherwise a good fit, you should still apply. Definitely don't worry too much about the "plusses" or "nice to haves." Additionally, years of work experience requirements are just a proxy for having the necessary skills - if you coded in grad school, that could count. That being said, applying to a post for a senior data scientist that requires 5 years of work experience as a Data Scientist, proficiency in Spark and Hadoop, and experience deploying machine learning models into production is probably not the best use of your time if you're an aspiring data scientist - it's looking for a different level.

Degree requirements

Many data scientist jobs list a degree in a "quantitative discipline," meaning fields like Statistics, Engineering, Computer Science, or Economics, as a requirement. If you don't have one, can you still apply for them? Generally, yes. We'll discuss this more in the next chapter, but if you took classes in those areas (including in a bootcamp or online), you can emphasize that. If you followed the advice of the last chapter and built a portfolio and wrote blog posts, you can show those to employers as evidence you can do the work.

One complication in data science postings is that there are different words for the same things. Machine learning and statistics are infamous for this. One company may ask for experience in regression or classification, another in supervised learning, but these are overall the same thing. The same goes for A/B testing, online experimentation, and randomized control trials. If there's a term you're not familiar with, google it – you may find you've done it under a different name! If you haven't worked with a specific technology they're referencing, see if you've done something similar. For example, if they cite AWS (Amazon Web Services), and you're worked with Azure or Google Cloud, you have the skill of working with cloud-based technology.

The other benefit of knowing how to decode a job description is the ability to detect red flags. No company is going to straight up say that they are bad to work for. The earlier you recognize a likely bad work situation, the better, so you'll want to start looking for any warning signs in the job description.

5.1.2 Watching for Red Flags

Finding a job is a two-way street. It can feel during this process that companies have all the power and you need to prove you're deserving. But you, yes *you*, can also be selective about where you work. Ending up in a toxic workplace or a mind-numbingly boring job is a really hard situation. While you won't always be able to tell if this will be the case just from a job description, there are a few warning signs to watch out for.

The first is if there's no description of the company or job itself, just a list of requirements. Those organizations have forgotten it's a two-sided process and aren't thinking about you. It can also mean they're buying into the data science hype and just want to have a team of data scientists, without setting anything up so they can work productively.

A second warning sign is the aforementioned "unicorn" description. While that was an extreme example, you should be careful of any job description that describes two or three of the job types (decision scientist, business intelligence, and machine learning) as primary responsibilities. While it's normal to be expected to have base competence in each, no person is going to be able to fill all those roles at an expert level. Even if someone could, they wouldn't have time to do it all.

Finally, look for mismatches between the requirements and the description of the position. Are they asking for experience with deep learning but the job functions are making dashboards, communicating with stakeholders, and running experiments? If so, they may be just wanting someone who can use the hottest tool or the most "prestigious" data scientists,

maybe someone with a Stanford PhD in AI, when actually they can't use that specialized knowledge.

5.1.3 Setting your expectations

While you should have standards for a potential job, you don't want to demand perfection. Aspiring data scientists sometimes see their path broken down this way: "Step 1-98: Learn Python, R, deep learning, Bayesian statistics, cloud computing, A/B testing, D3. Step 99: Get a data science job. Step 100: Profit." While this is an exaggeration, part of the data science hype is the idealization of what it's like to work in the field. After all, data scientist is "the best job in America" (according to Glassdoor), with a six-figure paycheck and high job satisfaction. You might imagine getting to spend every day on the most interesting problems in the field with the smartest colleagues. The data you need will always be accessible, cleaned, and any issues you face will be solved immediately by a team of engineers. Your job will be exactly as it was described, and you'll never have to do the parts of data science that interest you less.

Unfortunately, this is a fantasy. Just like we hope the first chapters convinced you that you don't need to know everything before getting into the field, companies aren't going to be perfect unicorns either. There's a reason this book doesn't end with you getting a data science job. Although it's a great accomplishment and you should be proud, data science is a field where you'll always be learning. Models will fail, workplace politics will scrap the work you've been doing for the past month, or you'll spend weeks working with engineers and product managers to collect the data you need.

It's especially easy to idealize companies that are well-known, either generally or for data science. Maybe you went to a talk and one of their employees blew you away. Maybe you've been following their blog for months and know they're on the cutting-edge of the field. Maybe you read an article about how they have nap pods, gourmet meals, and lots of friendly office dogs. But whatever has attracted you has likely interested other aspiring data scientists as well; most of these companies get hundreds of applications for an open position and can set the bar higher than even what is needed to do the job. In any case, the work you read about might be in a totally different division and this position may be uninteresting.

Even with realistic expectations, you'll likely not end up in your dream job for your first data science role. It's easier to transition within your field or bring data science into your current role; even if you're looking to eventually leave your domain, you may need to start out by moving to a position where you can leverage your other skills. That doesn't mean you shouldn't have certain requirements and preferences, but it does mean you'll want to have some flexibility. It's very normal to switch jobs in tech even after just a year or two, so you're not signing yourself up for the next 15 years. But you can't know exactly what you want before you're even in the field, and you'll learn even from bad jobs, so don't stress too much.

5.1.4 Leveraging your network

You're not limited to only looking at job boards. Many data science meetups have time at the beginning where people can announce if they're hiring. Conferences also might have job fairs or at least booths where sponsoring companies may be looking for new hires. Go up and talk to these people; it's part of their job, and even if their role isn't a good fit they maybe be able to give you some good advice.

You may also meet another attendee who works in the company or sub-industry you're interested in. You can ask if they have time for an informational interview so you can learn more about their field. An informational interview isn't (or rather shouldn't be) a passive-aggressive way of looking for a referral – instead, it' a great way to get a look inside a company and get advice from someone who's in the field. While we'll talk in the next chapter about the advantages of being referred for a job, we don't recommend asking people you've just met to refer you. That's a strong ask for someone they don't know, and no one likes feeling like they're being used. If they tell you about an opening at their company and say they can refer you, that's a great bonus, but you'll gain a lot even if they don't.

Finally, if you're able to be public about your job search, post on social media that you're looking and ask if anyone has any leads. Even though you might not have a strong data science network yet, hopefully you have friends and former classmates and colleagues who might know of positions within their companies.

5.2 Deciding which jobs to apply for

You now should have a list of at least a dozen jobs you're somewhat interested in and could be a good fit for. Do you immediately apply for all of them?

Well, some people do apply to dozens or even hundreds of jobs. They're trying to play the odds, figuring that if there's a 10% chance of getting a response, applying to as many companies as possible will give you the most responses. But there's a fallacy there - if you have a finite amount of energy and time, spreading it across 100 applications instead of 10 will make each one weaker. We'll talk in the next section about how to tailor your applications to each position, but that's only possible if you're more selective where you apply. And as we discussed, knowing people and having them refer you or at least getting an idea of the company from them is also the best way to get a first interview. That's pretty much impossible to do that for 50 companies. If you did try to meet that many people, you'd likely come off as insincere because you'll be looking to do it as quickly as possible.

R and Python

Should you apply for job if they ask for Python and you know R, or vice versa? While knowing one language certainly makes it easier to pick up the other one, you'll already be learning a lot at your first data scientist job - working with stakeholders, the internal politics, statistics, the datasets, etc. Even if you could get the job, a new language on top of everything else can be difficult. Thus, we generally recommend only applying to jobs that use your main language. If knowing one of those languages is a "plus", but the other one isn't required, you probably want to be wary. That could

mean that most of your work will not be coding. Finally, some jobs will ask for both. You also want to be a little wary here – usually that means that people use either one, not that everyone knows both, which can make collaborating difficult. This can work out, but be sure to ask during your interviews how the split is – if you would be one of only two people using Python on the team of 20, it's going to be harder to improve your skills.

You'll want to return here to what you've learned in the first two chapters about the types of data science companies and of data science work. Do you want try all the different parts of data science, tuning a recommendation system one month and creating a lifetime value model the next? You'll probably want to work at companies that have started doing data science more recently, as more mature companies will have specialized their roles. On the other hand, big tech companies will also have legions of data engineers making sure getting routine data is fast and easy.

Some of this will be obvious from the basic facts about the company – for example, a 10-person start-up is not going to have a mature data science system. But how can you find out more?

5.2.1 Researching the company online

First, see if the company has a data science blog. This will mostly only be tech companies, but these posts are invaluable to learn about what work they actually do and to include in your cover letter (covered in the next chapter). If you've never heard of a company, spend some time on their website. By knowing what the company does and how they make money, you can start making guesses at what kind of data science work they need. Finally, if you're really interested in a company, see if any of the company's data scientists have a blog where they talk about their work or have given talks about it.

When reading about a company, remember to also think about what's generally important to you. Does the ability to work from home sometimes matter? What about the number of vacation days? If you want to go to conferences, do they offer budget and time off? Reading what the company says about themselves can tell you their values. Do they talk about foosball, beer in the office, and catered dinner? That's likely to be a company full of young employees. Or do they emphasize environmental sustainability? Family leave? We'll discuss more in Chapter 8 how you can negotiate much more than salary, but for this stage, you can at least see if the company advertises these benefits.

Now that you have a manageable list of potential jobs, it's time to apply! In the next chapter, we'll walk you through creating a great resume and cover letter, including how to tailor them for each position.

5.3 Interview with Jesse Mostipak on finding a data science job

Jesse is the managing director of Teaching Trust, an education non-profit. She comes from a background in molecular biology and worked as a public school teacher before falling in love

with non-profit data science. You can find her writing on non-profit data science, advice for learning R, and other topics on her website, jessemaegan.com.

5.3.1 What is your current role?

I am the managing director of data science at Teaching Trust. Teaching Trust is an education non-profit located in Northeast Texas, and we focus on fueling great schools by building leadership teams and leadership capacity within schools and districts. My role focuses on five main areas.

- I do traditional data science – using programming to uncover insights in data, building models, and cleaning and wrangling data.
- I do policy work – I stay aware of what's happening in educational policy in the local, state, and national levels, and find ways to disseminate that out back to our staff
- I build our data related IT infrastructure – the ETL, or extract, transform, and load pipelines.
- I design and develop data products using human-centric design principles and practices – a method to build empathy with our end-users, in our case our program team that goes out and works with our teachers, principals, and administrators.
- Finally, I'm responsible for developing my team. Right now we're a team of two, and I'm really focused on what my team needs and how I can help them grow and develop in a way they're both excited to do and that benefits our organization.

5.3.2 What recommendations do you have for starting a job search?

Think about how attached you are to the data scientist title. If you really feel strongly you need to be called a data scientist, your options are a lot more limited. If you decide to not concern yourself with what you're called and to instead focus on the work that you're doing, you'll have a lot more flexibility to find jobs. Some non-data-scientist keywords to search for are "analysis," "analyst," and "data." While you'll have more to filter through, you may find a title like "research and evaluation" where you're qualified for that position but never would have come across it if you were just looking for data scientist.

When looking at jobs, focus on what you want to do as a data scientist. For me, I don't get a lot of pleasure from calculating the return on investment from website clicks. I asked myself, "what causes do I care about? Which organizations are aligned with that?" I cared a lot about the Girl Scouts, and they happened to be looking for an analyst position, so I was able to move in and do that. The same thing happened with Teaching Trust when I wanted to move more into education.

5.3.3 What role can and should your network play in your job search process?

I think your network is critical. This has been a really hard truth for me to learn, and I've really only started to embrace this idea in the last 2 years. I think no matter where you are in life or what you're doing, your network is going to play a critical role. Your network is a way of

amplifying your understanding of what's out there and bringing job opportunities to you. For example, the other day I reached out to someone I knew to ask how stringent the degree requirements were for a position they had listed. I asked because I knew someone who was great for it but didn't have a degree. My contact was able to come back and say they should definitely apply, as the requirements weren't strict, so I was able to pass that information along.

5.3.4 How can you build your network?

When I was making the transition to data science, I did a lot of things that failed for a very long time. I was the person who was reposting on Twitter every data science article that I saw, putting out 20 posts a day that had no engagement.

You should think about who you want to meet, why, what value you bring to that relationship, and what's authentic to you. Think about branding yourself; not necessarily in a stringent way, but making sure how you show up online and in social media spaces is authentic to you. For me, I realized I couldn't be a perfect data scientist and wait to be on social media until I know all data science, because that was going to be never. I decided instead that I would talk about the things that I'm learning and be transparent about the process, and that's how I built my network.

5.3.5 What do you do if you don't feel confident applying to data science jobs?

There is the case that you have no background in data, you've never done any programming, you're brand new to Excel, and you want to be a data scientist. If that's your position, I think it's fair to say, "Hey, I'm probably not qualified." But if you're developing skills, can do some analyses in Python or R, and have the basics under control, you should focus on how you can get comfortable taking risks and failing. You have to fail a lot as a data scientist – if you are worried about taking a risk and failing in the job application process, what's going to happen when you take a risk on a model and the model doesn't work out? You need to embrace the idea of ambiguity and iteration. You have to apply and try; you're going to get rejected from jobs, but that's normal! I get rejected from jobs on a regular basis, it's just part of the experience.

5.3.6 What would you say to someone who is reading job postings and thinks, "I don't meet the full list of any job's required qualifications?"

Some research suggests that certain groups of people especially feel they need to be 100% qualified, whereas other groups say "I meet 25%? I'm applying!" Wrangle that confidence of the 25% qualified and go for it. But you may also be getting tripped up in decoding the language of the job description. For example, let's say there's a discrete skill listed, such as 10 years working with SQL databases. You might think, "I don't have that; I have 7 years working with Microsoft Access." But I would say that's still a transferable skill. It's on you as the applicant to tell yourself, "I may not have this exact skill, but I have one that's a lot like it.

I need to take a look at SQL, see how transferable my skill is, and tell this company the amazing things I've accomplished with Microsoft Access and that they should hire me because I know I can do this with SQL and then some." Showing that you looked into these transferable skills will go a long way in the interview process.

5.3.7 What should people think about when applying?

Know what things matter most to you. For me it's not about using the coolest technology, it's about whether I care about the mission, vision, and values of the organization. I will come in and do Excel trackers for organizations that I deeply care about, and that's more important to me than being on the cutting edge of programming or database design. But someone else may be focused on being the best R package developer that they can possibly be, and they don't care where they do it. I think you need to have a north star or 3 to 5 non-negotiables for your job and stick to that, especially in your first data science position.

5.3.8 What's your final piece of advice to aspiring data scientists?

You need to develop your communication skills and your ability to "roll with it." You should be able to communicate across all levels of your organization in a way that respects the expertise of the people you're talking to and also shows you exist to make their lives easier. You want to see yourself as a collaborative partner in their goals.

By rolling with it, I mean being able to say something like, "That's not way I would approach that problem or this project, but I can see where you're coming from. Let's try it that way and maybe can I modify it in this way." You also need to be flexible because people are still just figuring out data science. Organizations want data scientists, but then they don't know what to do with them. You need to be able to not only communicate your findings but also advocate for yourself. your capabilities, and what you can learn to do. You serve at the pleasure of your organization; if their needs have changed, you need to evolve and adapt to best meet those needs.

Finally, know that your job description might change. You have to be able to say, "this isn't what I thought I would be doing, but how can I make this work for me." You can't say, "I am the best at neural networks, but I'm not doing neural networks, so obviously this job is crap." You need to know that any position that you take is going to change and evolve as the needs of the company does.

5.4 Summary

- Search for general terms like "data" on job boards and focus on the descriptions, not the titles
- Don't worry about meeting 100% of the qualifications listed
- Remember the job search process is a two-way street: look out for red flags and think about what kind of data science you want to do

6

The Application: Resumes and Cover Letters

This chapter covers

- Writing a compelling resume and cover letter
- Tailoring your application for each position

You've got a list of open jobs you're interested in; now it's time to let them know you exist! Pretty much every job will require you to submit a resume: a glorified list of your skills and past experiences. Most also ask for a cover letter too: a one-page letter describing why you should be considered for the job. It would be easy to quickly jot down your previous jobs and write a boilerplate letter saying you're interested in the company, but this is a situation where putting in more effort can be the deciding factor for making it to an interview.

In deciding what jobs to apply to, you've researched the companies to see if they would be a good fit. Now it's time to put that research to work by personalizing each application. In this chapter we'll start with making sure your base resume and cover letter are as effective as possible, covering best practices and common mistakes to avoid. Then we'll show how to take that "master" resume and cover letter and refine them for each job. Finally, you'll see networking can help accelerate your carefully crafted application into the hands of a hiring manager instead of an overflowing pile of resumes.

> **NOTE** the only goal of a resume is to convince a person who is barely skimming your resume and cover letter that you are a good fit

The key theme throughout this chapter is that you need to quickly convince a person that you are qualified for the position. Company recruiters often get hundreds of resumes for each

data science opening. Furthermore, because data science has so many distinct types of jobs within it, the range of skills of people applying to positions will be huge. This reinforces the notion that your materials have to say, "Hey, you reading this, you can stop skimming this massive pile because you found the person with the skills you were looking for." Being able to show you're qualified like that isn't an easy task.

While all of this is more work than spending an hour writing a basic cover letter and resume and applying with one click to dozens of jobs, it will yield much better results. You'll be more likely to get an interview, as you'll have matched your application to their requirements. And when you reach the interview, the topic of our next chapter, you'll be able to give a great answer to the common question, "Why are you interested in this role?"

6.1 Resume: the basics

The goal of your resume isn't to get you the job; it's just to get you an interview. Recruiters who run the interview process get in trouble if they bring in people who clearly don't meet the qualifications for the job, and they are praised when the people fit the qualifications well. Your resume needs to show the reader you meet the requirements for the position so that the recruiter is comfortable moving you on in the process.

That goal is very different from "create a catalog of every experience you had," which unfortunately is a goal that many inexperienced resume writers assume. While you do want to avoid having gaps on your resume by leaving off recent jobs entirely, you can spend less time on those not related to data science. And even if you have a lot of data science experience, you should still focus on highlighting the most relevant. If you have a multi-page resume, most recruiters won't have time to read all of it, nor will they be able to tell what parts to read. No one will tell you, "Well, we would have hired you, but you didn't put your job lifeguarding in high school on your resume, so we couldn't."

There will be plenty of time later in the interview process to go through your jobs, education, and data science projects in depth. For now, you want to focus on what's most relevant for meeting the qualifications of the position you're applying for. The rest of the process you'll be focused on your great qualities that will help you stand out beyond the other applicants, but for the first step, it's good to focus on fitting in to the hiring manager or recruiter's expectations.

With that in mind, let's walk through the basic structure of a resume, how to create good content within that structure, and how to think about the many "resume rules" that get thrown around. While plenty of this content could apply to any technical position in industry, as much as possible we will focus on what is unique about data science. We also made an example resume to use as a guide for what to expect.

SARA JONES

San Francisco, CA · 534-241-6264
sarajones@gmail.com · linkedin.com/in/sarajones · sarajones.github.io · github.com/sarajones

EXPERIENCE

JUNE 2019 – PRESENT, SAN FRANCISCO, CA
DATA SCIENCE FELLOW, AWESOME BOOTCAMP

- Built a web application in Python that recommends the best New York City neighborhood to live in based on someone's budget, lifestyle preferences, and work
- Analyzed 2,200 New York Times business articles (obtained via API) using natural language processing (TFIDF and NMF), visualizing how topics changed over time

AUGUST 2017 – JUNE 2019, SAN FRANCISCO, CA
INVESTMENT CONSULTANT, BIGCO

- Created a forecasting model in Python that boosted quarterly revenue by 10%
- Automated generating weekly market and industry trend reports

SEPTEMBER 2016 – JUNE 2017, NEW ORLEANS, LA
INTRODUCTION TO STATISTICS TEACHING ASSISTANT, COOL UNIVERSITY

- Led weekly review sessions of sixty students, earning a 4.86/5 rating in evaluations
- Created and open-sourced study guides that have been downloaded over 1,500 times

JUNE 2016 – AUGUST 2016, NEW ORLEANS, LA
ECONOMICS RESEARCH ASSISTANT, COOL UNIVERSITY

- Conducted an in-person experiment on decision-making with 200 participants, using cluster analysis to analyze the results in Python
- Published the resulting paper in the Journal of Awesome Economics

EDUCATION

JUNE 2017, NEW ORLEANS, LA
BA ECONOMICS, STATISTICS MINOR COOL UNIVERSITY

GPA 3.65/4.0
Relevant Coursework: Linear Algebra, Introduction to Regression and Statistical Computing, Experimental Design, Econometrics, Elements of Algorithms and Computation

SKILLS

- Python
- SQL
- Machine Learning
- Git
- Pandas
- Seaborn
- Scikit-learn
- NumPy

Figure 6.1

6.1.1 Structure

Below, we'll use the example to discuss what components are in a resume.

SECTION: CONTACT INFORMATION

Figure 6.2

Having your contact information is necessary so that the recruiter can contact you! You need to put your first and last name, phone number, and email at a minimum. Beyond that, you can also put links to where they can find more information about you. These include social media profiles like LinkedIn, online code bases like GitHub, or personal websites and blogs. To figure out what to add, ask, "if someone clicks on it, would they think more highly of me?" For example, a link to your project portfolio from Chapter 4 is a fantastic thing to include. But a link to a GitHub profile that is empty save for a clone of a tutorial project is not. If you have any data science work publicly available, try and figure out a way to show it here.

Generally you also want to include the city and state you live in—this will let the recruiter know that either you are nearby and can commute to the job or that you would need to relocate if you got the job. Some companies are hesitant to relocate new hires because of the expense, so if you don't live nearby and don't want to bite that bullet you could potentially leave your location off.

If your legal name doesn't match the name you commonly go by, you can put your common name. Further along in the process you will need to let them know what your legal name is for things like background checks, but you aren't required to use your legal name when applying.

Don't use an email address that's potentially offensive ("i_hate_python@gmail.com") or something that might expire (like a school email).

SECTION: EXPERIENCE

Figure 6.3

This is where you show that you're qualified for the job through previous jobs, internships, or bootcamps you've done. If they're related to data science, like software engineering, that's great: spend a fair amount of your resume on them. If a job isn't related to data science, like being an art history professor, you should still list the jobs but don't spend much time on it. For each position you've held, put the company name, the month and year of your start and end, your job title, and at least one bullet (two or three for the most relevant jobs) of what you did. If you're a new or recent graduate, you can include internships and research jobs in college.

This section should be the largest in your resume and could potentially take up half the space available. It's also often the most important, since it's the first place recruiters will look to see if you have data science experience that could be related to the job they're hiring for. Due to the importance of getting this right, we'll go in-depth on how to create the best content for this part of the resume later on in the chapter.

SECTION: EDUCATION

Figure 6.4

In this section you list your different educational experiences, to hopefully show that you have a set of skills useful for the data science job. If you went to school past high school, even if you didn't get a degree, put your school(s), dates (same format as your work experience), and your area of study. If this will be your first job out of school and your GPA is high you can list it (say above a 3.3) otherwise leave it off. If you are a recent graduate and you took statistics, mathematics, or computer science classes, or any that involved these (like social science research methods or engineering classes), you can list those classes. Also, if you took a set of online data science courses you should include them here.

Recruiters will be very interested to see if you have an area of study that's relevant to data science, like a degree in data science, computer science, or math. They'll also be interested to see the level of the degree. Since many data science topics aren't covered until graduate levels of programs, having that will help. Recruiters generally won't care about what school you went to, unless it's extremely famous or prestigious and even then that won't matter if you're more than a few years out of school. It's nice for recruiters to see any bootcamps, certificates, or online programs since it shows you've furthered your education.

While the education section of your resume can give valuable information to the recruiter, you can't really improve the section by doing anything less than going out and getting an additional degree or certificate, which we've covered in Chapter 3.

SECTION: SKILLS

Figure 6.5

This section is where you can explicitly list all the relevant skills you have to contribute in a data science setting. Ideally a recruiter will see this section and nod while saying, "yes, good" since you'll have skills listed that are relevant for the job. For data science resumes there are two types of skills to list here. The first type is programming/database skills which can be programming languages like Python and SQL, frameworks and environments like .NET or JVM, tools like Tableau and Excel, or ecosystems like Azure and AWS. The second type is data science methods like regressions or neural networks. Since there are so many possible methods you can list, try to focus on a key few that together show you have the essentials and a few special skills. For example something like: "regressions, clustering methods, neural networks, survey analysis" shows you have the basics and depth. Try not to list more than 5-6 to avoid overwhelming people, and don't list skills that have no chance of being relevant for the job (like an obscure academic programming language from your time in grad school).

Only put skills on that you would be comfortable using on the job, not a language you haven't touched in five years and don't want to pick up again. If it's on your resume, it's fair game to ask about it. If there are skills you've seen requested in the data science job postings you've viewed and you have them, make sure to list them! That's exactly what the recruiters will look for.

Don't put soft skills like "critical thinking" or "interpersonal skills;" while these are crucial for being a successful data scientist, saying those on your resume is meaningless since anyone can just say them. If you do want to highlight your skills in these areas, talk about how you utilized them in specific instances within the experience section of your resume. You also don't need to list basic skills anyone applying would be expected to have, like "Microsoft Office Suite."

SECTION: DATA SCIENCE PROJECTS (OPTIONAL)

If you have data science projects you've done outside of work, you can create a section for them. This is great for candidates who have less work experience but have done projects on the side or in a school or boot camp. You're basically telling the recruiter, "While I may not have much relevant work experience, that doesn't matter because I've still done the full data science process."

For each project you'll need a title, a description of what you did, how you did it, and the results. In fact, the data science projects should look as though they are jobs in structure and

content, so everything in the section below on generating content applies for these too. Ideally you have a link to a blog post or at least to a GitHub repository that has an informative README. Data science is a technical field where it's unusually easy to just show the work you did, and here is a great place to do it. If you have enough relevant work experience, you can skip this section and still talk about projects in your interviews.

SECTION: PUBLICATIONS (OPTIONAL)

If you published papers in a master's or PhD program that are related to data science, you should include those. If you published papers in other fields, even quantitative ones like physics or computational biology, you can include them but only briefly. Since they are not directly related to data science the person reading them won't get much out of the publications except "you worked hard enough to get a publication." You can list the relevant work you did during your research in the experience section, like "created an algorithm to analyze millions of RNA sequences per minute." But a publication in a journal the hiring manager has never heard of, even if it's a prestigious one in your field, won't go too far on its own.

OTHER SECTIONS

You can add other sections, like "honors and awards" if you've won Kaggle competitions or received a scholarship or fellowship, but they aren't necessary. You don't need to include references; speaking to your references will come later in the process and you can share that information if you progress to there. Objective statements are usually not needed and redundant given the other information in your resume. For example, "Data Scientist experienced in Python looking for a position to develop A/B testing and modeling skills" is not going to make a recruiter more excited!

PUTTING IT TOGETHER

Generally, you put your contact information at the top, followed by the next most important section. If you're in school or just graduated, that's probably your education; if you don't have relevant work or education, put your data science projects at the top; otherwise, put your work experience there. Within your work and education sections, put your experiences in chronological order, from most recent to least.

We've seen lots of different effective formats for data science resumes. In this field you have a bit from freedom in your design; there is nothing close to a standard format. But with that freedom you always want to focus on making your resume easy to scan quickly. Since recruiters spend so little time looking at your resume, you don't want them to spend that time trying to figure out how to find your most recent job! Don't make your aesthetic design distract from your content; constantly consider how others will view it. Some good practices include:

- Clear headers for each section so it's easy to jump between them.

- More whitespace to make it easier to read the content.
- Bold important words like the titles you've held at each company.

If ideas like whitespace and headers are overwhelming, stick to a resume template from online or consult a design specialist.

You want to limit your resume generally to a single page. This serves two purposes. One, given the brief skim it'll get, you want to make sure they're spending that time on the information you think is most valuable. Second, it shows you can communicate concisely and understand what parts of your experience are most important to share. If a person submits a 17-page resume (which we've seen), then it strongly suggests they have no idea what in their past makes them a good candidate, and that they feel entitled to other people's time for them to read it.

Make sure you're consistent throughout your resume. If you abbreviate month names in your education section, abbreviate them in your work experience section too. While you can have different fonts and sizes for headings and body, don't switch it up from bullet to bullet. Use the past tense for previous positions, present for a current one. These things show that you pay attention to the smaller detail and again help the reader to quickly process your content, as they won't be distracted by font or style changes. While a single inconsistency is unlikely to cost you an interview sometimes details make all the difference.

> **TIP Proofreading is important!** A few typo or grammatical mistakes may lead your application to the (metaphorical) trash-bin. Why so harsh? When sifting through hundreds of resumes, two kinds stand out: those that are clearly exceptional (rare) and those that are easy to eliminate. The latter needs some rules of thumbs, and in addition to ones about clearly not meeting the requirements, typos are an easy one. Data science jobs require attention to detail and checking your work; if you can't do that when putting your best foot forward in an application, what does that suggest about your work? In addition to using spell check, have at least one other person carefully read over your application.

6.1.2 Deeper into the experience section: generating content

Hopefully, coming up with the dates and titles of your work and education history are easy enough to fill out. But how do you come up with those punchy bullet points to describe your work experience (or data science projects)?

The common mistake people make on their resume is to just create a list of their job duties, like "generated reports for executives using SQL and Tableau" or "taught Calculus to three sections of 30 students." There are two problems with this. One, it's just stating what you were responsible for, not what you accomplished or how you did it. Second, it may not be framed in a way that's relevant to data science. So for the previous two examples, you could describe the same work as "used Tableau and SQL to generate reports on sales forecasts for executives, which increased the fidelity of monthly sales meetings" or "taught Calculus to 90 students, earning an average of 9.5/10 in student evaluations and 70% of students getting a 4 or 5 on the BC Calculus AP Exam."

As much as you can, you want to explain your experience in terms of transferable skills to data science. *Even if you haven't worked in data science or analytics, was there any data that you did work with?* Hiring managers are enormously willing to consider experiences outside of data science roles as still relevant, but you have to explain why they should. If any of your work can conceivably be related to taking data and understanding it, you should put enormous effort into making a concise story on what you did. Did you analyze 100GB of star data for an astrophysics PhD? Did you manage 30 Excel files to plan staffing for a bakery? There are lots of activities that are using data to understand a problem.

Have you used tools like Google Analytics, Excel, or Survey Monkey? Even if those may not be the tools the current job is asking for, working with data of any type is relevant. What communication skills did you use? Have you explained technical or niche concepts, maybe in PhD research talks or to other parts of the business? If coming up with transferable skills is difficult, don't worry – the rest of the advice on writing better bullets will still help. But if you haven't already, you should think about how your education or side projects can demonstrate data science skills, especially if your work experience can't.

For the least relevant positions that were a few years ago, it's okay to just have one bullet. You generally don't want to leave off a job though if it will leave a gap in your resume of more than a few months. If you've been in the workforce for a while and had a lot of jobs, it's okay just to list the three or four most recent.

You might be finding this process is a lot easier for the job you're currently in than the one you had five years ago. One good practice is to keep a list of accomplishments and the major projects you've worked on. When you're in it each day, making incremental progress, you can forget how impressive the whole is when you step back. People know that your resume isn't an exhaustive list, so they won't think, "It took her 15 months to build an automated system for tracking and scoring sales leads that saved our sales team over 20 hours of manual work a week?" They'll think, "Wow, we need a system like that!"

Generally, bullets can fall into two types. The first is big accomplishments; for example, "Created a dashboard to monitor all running experiments and conduct power calculations." The second is average or totals, such as "Implemented and analyzed over 60 experiments, resulting in more than $30 million in revenue."

In either case, your bullets should each start with a verb and are ideally quantifiable. Rather than saying, "I made presentations for clients," write, "created more than 20 presentations for Fortune 500 executives." It's even better if you can quantify the impact you had; writing "ran 20 A/B tests on email campaigns, resulting in a 35% increase in click rate and 5% increase in attributed sales overall" is much more powerful than "ran 20 A/B tests on email campaigns."

New Graduates

If you're about to or just graduated college, your most relevant skill is your education. Your data science portfolio will be helpful here too. When you're searching for jobs, look for positions specifically titled "New Grad," "Junior," "Associate," and "Entry-level." Also look at your career center for help and go to any job fairs that happen on campus. Internships are relevant – less for what work skills you learned and more that it shows you can come to an office each day, be professional and productive.

6.2 Cover letters: the basics

While the purpose of a resume is to give hiring managers relevant facts about your work experience and past, the purpose of the cover letter is to help them understand who you are as a person. Your cover letter is where you can explain how you've researched the company and highlight why you are a great fit. If your resume doesn't show a linear path, a cover letter can pull that together and explain how all the pieces fit to make you a great candidate for this job. Even just showing that you know what the company is, that you've read their about page, or that you've used their product (if it's available for consumers for free) goes a long way. Your letter is your best tool to help hiring managers understand things that don't fit well in bullet lists.

Unlike a resume, a cover letter may be optional. But if a company has a place to submit a one, do so—some companies will eliminate candidates if they haven't written one. It's not uncommon for companies to even give a specific thing for you to write about, like your favorite supervised learning technique; this usually is to check whether people actually read and followed the request of the job description instead of just sending out a generic cover letter everywhere. You definitely want to let the company know that you can follow instructions.

Knowing that a cover letter is to help the company understand who you are better, a common mistake we see in cover letters is focusing on what the company can do for you. Don't say "this would be a great step for my career." A hiring manager's job is not to help as many careers as possible, it's to hire people that help the company. Show them how you can do that. Even if this would be your first data science job, what relevant experience do you have? What record of achieving results, even if they're not data-science related, can you share so it's clear you work hard and accomplish goals? Don't undercut yourself; try and think broadly as to how you can make yourself appealing to the company.

Just like your resume, you want to keep it short; three-quarters to one page is usually the rule. Focus on your strengths. If the job description lists four skills and you excel in two, talk about those two! Don't feel like you have to make excuses for skills that you lack. Below is an example cover letter we're using as a discussion point.

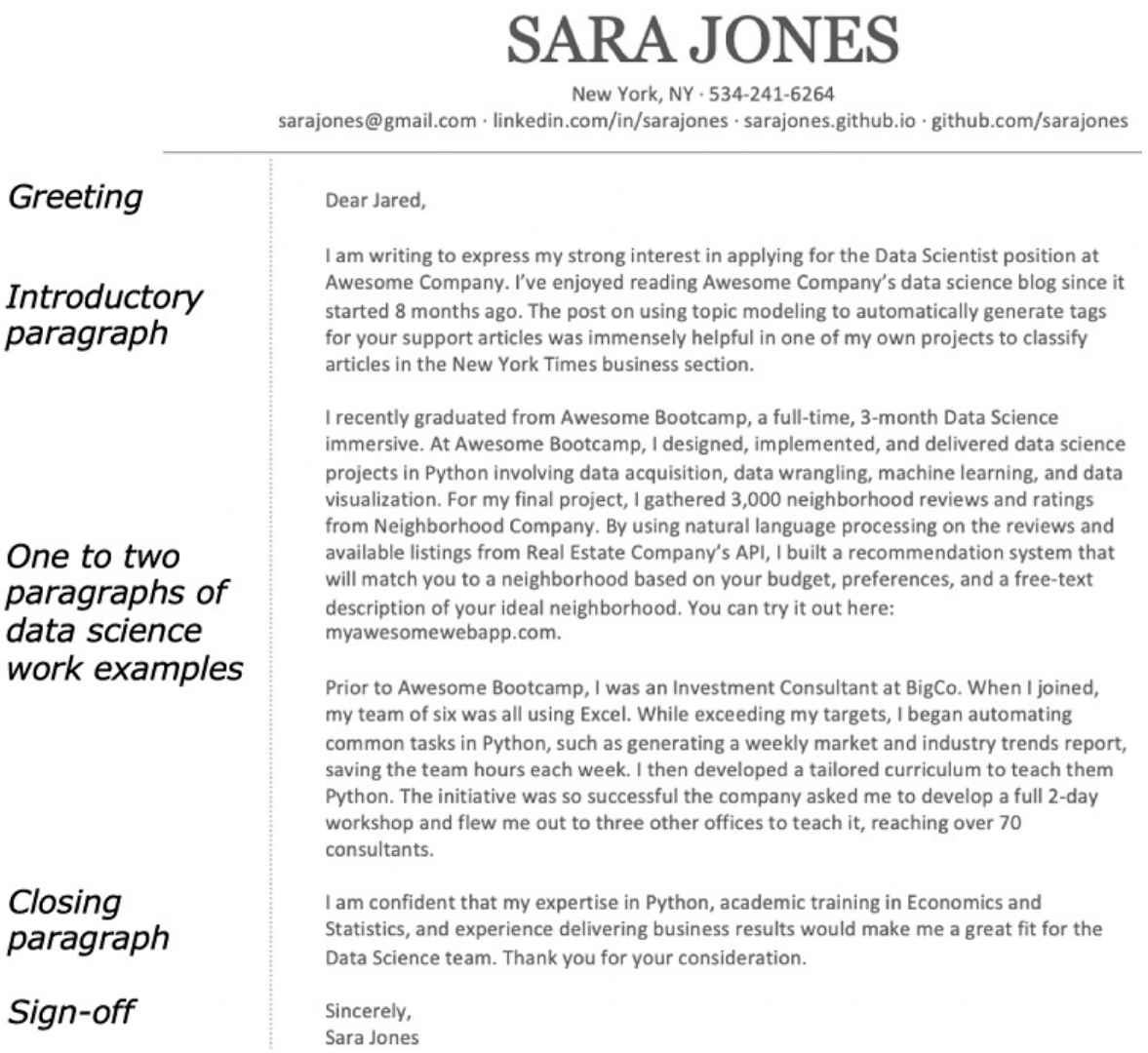


Figure 6.6

6.2.1 Structure

Cover letters have a less well-defined set of rules than resumes. That being said, here's a good general structure you can follow:

- **Greeting:** Ideally, find out who the hiring manager or recruiter is for the position. The first place to look is the job description itself; the hiring manager's name might be

listed. Even if it's just their email address, you can probably figure out who it is with some googling. Otherwise, check LinkedIn and the company website to see if one of them identifies the team leader. Even if you end up too high, maybe at the VP of the department who will be your skip-level (manager's manager), it still shows you did your research. Finally, if you can't find a name, address your letter to "Dear *Department Name hiring manager," e.g. "Dear Data Analytics hiring manager." Don't say "Dear Sir or Madam."*

- **An introductory paragraph:** introduce yourself, name the position you're interested in, and briefly explain why you're excited about the company and role. If the company has a data science blog or if any of their data scientists have given talks or written blog posts on their work, this is a great place to mention you've read that. Connect what you learned there to why you're excited about the position or company.
- **One to two paragraphs with data science work examples:** connect your previous accomplishments to this role. Add details to something you discussed in your resume. Pick one role or side project and go in depth. You want to give specific examples. Follow the "show don't tell principle;" instead of writing that you're a "detail-oriented, organized problem-solver," give them an example of when you did this at work.
- **A closing paragraph:** thank them for their time and consideration. Sum up your qualifications in explaining why you'd be a good fit.
- **A sign-off:** "sincerely," "best," and "thank you for your consideration" are all good closings; avoid anything to casual or overly friendly, like "cheers" or "warmest regards."

6.3 Tailoring

The previous two sections laid out general rules for an effective cover letter and resume. But the biggest way you can differentiate yourself is by closely tailoring them to the position you're applying for.

The first person screening your data science resume is likely not the manager for a position; it may not even be a human! At larger companies, they'll usually have applicant tracking systems that automatically screen resumes for keywords, surfacing those that have it. This system might not recognize "linear modeling" as meeting the requirement for experience in "regression." A human reader might not either; an HR person may have been given nothing besides the job description and instructions to find promising candidates. You don't want to risk a recruiter not understanding that your project using "*k*-nearest neighbors" means you have experience in clustering analysis, or that "NLP" is the acronym for "natural language processing." You want someone to be easily able to look back and forth between your resume and the job description, finding exact matches for the requirements in your experience.

We recommend you have a "master" resume and cover letter you can pull from rather than starting from scratch. This is especially helpful if you're applying for different types of

positions. If some emphasize machine learning and others exploratory analyses, it's much easier if you have bullets and key terms related to each ready to go. Your "master" resume and cover letter can be longer than one page, but make sure your ending result is always under a page.

Tailoring your application to the position doesn't mean you need to have one bullet or skill for every single requirement. As we discussed in the previous chapter, job descriptions are generally wish-lists. Try to figure out which ones are the core skills for the job; sometimes companies will helpfully divide skills and experience into "requirements" and "nice-to-haves," but even if they don't, you might be able to tell by the description of the job responsibilities. While companies would love to get someone who gets a check-plus on everything, most won't be holding out for it.

One exception is big tech firms and well-known, fast-growing start-ups. These companies get a lot of candidates and are looking for reasons to reject people. They're very worried about false positives, meaning hiring someone who is bad or even just average. They don't really care about false negatives, not hiring someone who is great, because they have lots of great people in the pipeline. For these companies, you usually do have to meet 90% if not 100% of the requirements.

6.4 Referrals

Company websites and job boards all have a place you can apply, sometimes even just with a click of a button if you've saved your resume on the job board. Unfortunately, because it's so easy, your resume will often end up in a pile of hundreds or even thousands of other similar "cold" applications. That's why we recommend avoiding applying this way until you've exhausted other options. Reading job postings is a great way to get a feel for what kind of jobs are available, but the best way to get your foot in the door is to have someone hold it open for you.

You want to try to use the hidden back door to most companies: referrals. This is when a current employee recommends someone for a position, usually by submitting their application and information through a special system. Many companies offer referral bonuses, where employees can get a couple thousand dollars if they refer someone who gets and accepts a job offer. Companies like people who are referred because they come "pre-vetted": someone who already works at the company and (presumably) is doing well thinks this person would be a good fit. Even if it's not that formal, being able to write on your cover letter "I discussed this position with [Star Employee X]" and to have that person tell the hiring manager to look out for your resume is a huge benefit.

How do you find people who can refer you? Start by looking at LinkedIn and seeing if you know anyone who works at a company you're interested in. Even if you haven't spoken in a while, it's perfectly fine to reach out with a polite message. Next, look for people who previously worked at the same company or went to the same school as you. You're more likely get a response from a cold message if you mention something you have in common. Finally,

look for people who are "second degree" and see who you have in common. If you're on good terms with any of your mutual connections, reach out to that connection to see if they'd be willing to introduce you.

If you're reaching out to a data scientist, take some time to learn about what they do. Do they have a blog, Twitter, or GitHub where they've shared their work? Mark Meloon, a Senior Data Scientist at ServiceNow, wrote in his blog post "Climbing the relationship ladder to get a data science job" about how the most effective messages would be ones that combine a compliment to the content he's published with a request to ask some more questions. This way, you'll also avoid asking them about things they've already publicly talked about and can focus on getting advice you couldn't find elsewhere.

Remember that it's not only people in data science who can help you. While other data scientists will be best positioned to tell you what it's like to work at their company, people in any position can refer you. If someone you know works at a company you want to apply to, reach out to them! At the very least, they can still offer you insight into the company culture.

Crafting an effective message

In his blog post, "Do you have time for a quick chat?", Trey Causey, Senior Data Science Manager at Indeed.com, outlines some suggestions for effectively reaching out to someone you don't know to talk about your project, job search, or career choice. By following these guidelines, you'll be much more likely to get a response, have a productive meeting, and build a good foundation for a continuing relationship.

1) Have an agenda for what you want to discuss and include it in your email.
2) Suggest a few times (including an ending time 30 minutes later) and a location near the person's work.
3) Buy their lunch or coffee.
4) Get there early.
5) Have specific questions and goals for your conversation based on the agenda you sent. Don't just ask for "any advice you can give me."
6) Keep track of time and let them know when you've used up all the time you asked for; if they want to keep talking, they will.
7) Thank them and follow-up on anything you've talked about.

Here's how Trey pulls that all together into a sample message:

> "Hi Trey, I read your blog post on data science interviews and was hoping I could buy you a coffee at Storyville in Pike Place this week to ask you a few questions about your post.
> I'm currently interviewing and the part about whiteboard coding was really interesting to me. I'd love to hear your thoughts on how to improve whiteboard coding questions and answers as well as share some of my own experiences with these types of questions.
> Could you spare 30 minutes sometime, say Tuesday or Wednesday of next week? Thanks for writing the post!"

6.5 Interview with Kristen Kehrer, a data scientist resume expert

Kristen has over a decade of experience in data science working across the e-commerce, healthcare, and utility industries, specializing in delivering machine learning solutions. She created an online course for improving your data science resume and hosts a data science podcast, Data Science Live, with Favio Vazquez. You can find her blog posts and course on datamovesme.com.

6.5.1 What's your current role?

I recently started a new position creating data science content for online data science courses, including for Ivy League universities. Previously, I was a senior data scientist at Constant Contact. One thing I worked on was cluster analysis and segmentation so we could further personalize our messages to our customers. Constant Contact is a subscription product, so we like to see people logging in pretty frequently, but at the same time, we understood that if you're an ice cream shop in Massachusetts, you might shut down for the winter. So I built a time series model to determine whether a customer has a seasonal usage pattern. That way we could tell them, "Hey, even in your off-season, it's still a great practice to promote your business to try and grow your list. Here's things you can do now to potentially have a better on-season."

6.5.2 What was your path to becoming a data scientist?

I finished my bachelor's degree in mathematics in 2004 and then had a couple different jobs. In 2007, when the housing bubble was bursting, I decided to hide in academia and entered a PhD program in Statistics. I thought maybe I'd teach in a community college, but then my advisor moved schools and I decided not to follow him. Some of my professors suggested I apply for a job that happened to be three miles away from where my parents lived. I got the job and started building neural net models to forecast hourly electric load. That was in 2010 and where my journey really began. I didn't know that the term was data scientist, but that's when I started in the analytics space!

6.5.3 How many times would you estimate you've edited your resume?

Oh, a million! I come from a blue-collar family, where my dad was a firefighter and my mom stayed at home, so I was never taught how to write a great resume for industry. But I did okay by asking others for help when I was getting out of grad school. I also have always been the type to keep track of any new project I work on or anything interesting that I could add to my resume. I wasn't one of those people who'd go for two years without updating my resume. More recently, my old company paid for a career coach when they laid me off. I got to learn all about resume best practices and how to effectively position myself to land a great job.

I absolutely advise people to update their resume often. Especially if you've been working at the same place for a while, it is very difficult to try and think about all the relevant things

that you could add to your resume. For example, I co-authored a couple posters in the healthcare industry that won awards. That's not relevant to every position I apply for, but if I am applying to a position in healthcare, I'd want to be able to reference that research. If I didn't keep track of it, I would not be able to remember who my co-authors were or what the title of the poster was.

6.5.4 What are common mistakes you see people make?

So many things! One is the four-page resume that still has that they were a swimming coach. Another is not realizing that the applicant tracking systems don't parse certain things well. If people have icons or charts on their resume, that's going to come through as a blob on a lot of the older automated systems and may end up with you being automatically rejected. I also don't like when people put, say, three stars for Python, because you're not giving people any context and whichever skill you're putting two stars for you're saying that you're not good at that thing.

6.5.5 Do you tailor your resume to the position you're applying to?

I'm not obsessive about it. But almost all medium to large companies now use an applicant tracking system and you want to be able to rank high in terms of matching key words. If I saw things on a particular job description that matched things that I've done, but were maybe worded slightly differently, I'd just edit a couple words to match the verbiage that they're using on their job description.

6.5.6 What strategies do you recommend for describing jobs on a resume, especially for people who haven't had data-related jobs before?

I tell people to optimize their resume for the job they want, not the job they have. You don't need to make a list of all the things that you've ever done. Instead, think about what you've done that you can reposition for data science. For example, if you're a math teacher, you've been explaining technical or mathematical material to a non-technical audience. Or maybe you worked on a project where, even though it wasn't in analytics, you had to work cross-functionally across multiple teams. Overall, you want to be able to show that you're able to solve problems, self-manage, communicate well, and achieve results. Finally, you can use side projects to highlight your technical chops and the initiative that you're taking.

6.5.7 What's your final piece of advice for aspiring data scientists?

You need to start applying to data science jobs. Too many people just keeping taking online courses because they think they need to know a million things to become a data scientist, but the fact is, you're going to start a job and you're still going to have more to learn. Even ten years in I still have more to learn. By applying, you'll get feedback from the market. If nobody responds to your resume, maybe it's that you're not positioning yourself well, or maybe it's that you don't quite have the skills. Gather some feedback from a few people and then choose

an area to focus on, like being able to automate processes in Python. Work on that, add it to your resume and apply to more. You need to apply, get responses, and iterate and move forward until you get a job.

6.6 Summary

- Your resume is not an exhaustive list of everything you've ever done. You need it to get you an interview, not the job, so focus on matching it closely to the job description.
- Cover letters let you show why you're interested in an organization and how your experience positions to make a valued contribution.
- Talking to people who currently work at a company, especially if they're data scientists, is the best way to get insight into specifics about openings and hiring manager priorities.

7 The interview

This chapter covers

- **Delivering what interviewers are looking for**
- **Knowing what to expect in technical questions**
- **Proper etiquette when communicating with a company**

Congratulations! You've been invited to visit a company and be interviewed by their data science team. Now that you've made it this far the real work begins: you need to somehow show these strangers that you would be good at a role you only know from a few paragraphs in a job posting. In the interview they may ask you technical questions of all different levels about all different technologies—some which you may not have even used before. Further, during the interview you'll need to learn enough about the company to be able to decide if you'd *want* to work there. You will have to do all of these things in only a few hours, all while acting professional and proper. It's enough to give you some serious anxiety sweats.

The good news is that with the right preparation and mindset, data science interviews can be taken from "panic attack inducing" to "manageable" or "tolerable" or perhaps even "an enjoyable experience." In this chapter we will walk through what interviewers are looking for during interviews, and how to adjust your thinking to align with their needs. We'll walk through what you should expect from the technical and non-technical questions, as well as a data science case study. Lastly, we will go over how to behave and what questions you should be asking interviewers. With this, you should be well prepared for what lies ahead.

7.1 What do companies want?

When employees of a company are interviewing candidates for an open position, they are looking for one crucial person:

Someone who can do the job.

This is the only type of person they are looking for. Companies are not looking for the person who get the most interview questions "right" or who has the most degrees or years of experience. They only want someone that can do the work that needs to be done and help the team move forward with their goals. But what does it take to do a job? Well, a few things:

- **Having the necessary skills.** The necessary skills can be both technical and non-technical. On the technical side, this means understanding the skills covered in Chapter 1: some combination of math and statistics as well as databases and programming. On the non-technical side this is general business acumen, as well as things like project management, people management, visual design or any other number of skills depending on the role.
- **Being reasonable to work with.** If say something offensive, act defensive, or have any other number of character flaws that would make it difficult for other people to interact or collaborate with you, a company will not want to hire you. This means that during the interview (and always, actually) you'll want to be agreeable, compassionate and positive. This does not mean that your interviewer should want to "grab a beer with you," since that kind of filtering often creates homogeneous culture which is bad for companies. It just means that the people on your future team need to see you as someone they want to work with.
- **Being able to get things done.** It's not enough to just have the skills to do the job, you have to actually use them! This means finding solutions to problems on the job and implementing those solutions. Data science has lots of places where a person can get stuck: figuring out messy data, thinking through the problem, trying different models, and tidying a result. A person who can overcome each of those challenges will be much better at doing the job then someone who sits around waiting for help without asking for it. People who try to make everything perfect have trouble with this as well—never being able to call your work done means you can never put the work to use.

Knowing that those three things are what companies look for in candidates, let's jump right into the interview process. As we walk through how the interview works and the questions that come up often, we will frame our discussion in terms of those three ideas.

7.2 The interview process

While the exact process of a job interview varies by company, they all tend to follow a basic pattern. This pattern is designed to maximize the amount of information a company learns about the candidate, while minimizing the amount of time it takes for people within the company to run the interview. The people who do interviews are typically busy, have many interviews to facilitate, and want to allow for fair comparisons between candidates, so the process is streamlined and consistent. Here is a basic outline of what to expect in an interview process:

1. **An initial phone screening** – this will typically be a 30 (or sometimes 60) minute phone interview with a technical recruiter: a person who has lots of experience in screening candidates and knows technical terms, but doesn't do technical work themselves. From the company's perspective, the goal of this interview is to check if you have any chance of being qualified for the job. The recruiter wants to screen out people who aren't remotely qualified (don't have the skills), who sound abrasive or mean (won't work well with others), or couldn't possibly get things done (have no experience in the field). From a technical side, the interviewer is just checking that you potentially have the minimum skills needed, not that you're the best at them. They're much more likely to ask, "have you used linear regressions before" than, "how do you compute the maximum likelihood estimate for a gamma distribution." After the first phone interview, the company occasionally will have another phone interview with a more technical screener. If the initial phone screening goes well, within a few weeks you'll have:
2. **An on-site interview** – this is often 2 to 6 hours and makes up the main part of the interview process. During this visit you'll get to see where you will work and meet the people who you would work with, likely because they'll be the ones interviewing you. This interview provides the company time to ask more probing questions about your background, your skills, and your hopes and dreams as a data scientist. You'll be interviewed by multiple people during the visit, each asking questions on different topics, some technical and some non-technical. The goal of this interview is to ensure (via technical questions) that you have the necessary skills and (via behavioral questions and how you conduct yourself) that you are reasonable to work with. If this interview goes well then it's time for:
3. **A case study** – you'll be given a description of a real-world scenario and data related to it. You'll be asked to spend some time, likely a weekend, to analyze the data, try and solve the problem, and to create a report around it. You'll then have to return to the office and present your report to the hiring team. This exercise shows the team that: you have the necessary skills (through how good your report is) and that you can get things done (through how much you were able to do in the report). Not all companies do this step, and sometimes they replace it with having you do a presentation on past work. If your case study report goes well you'll then have:
4. **A final leadership interview** – this interview is with the manager, director, or some other leader of the team. The purpose is for the leader to give their approval of your fit for the position and the team. If the leadership interview even happens in the first place it means the data science team thought you were a good fit so it's rare for this interview to overrule their approval. Note that this interview often happens right after the case study, but also could happen at the beginning or end of the first on-site interview. Assuming it goes well you'll get an offer in less than two weeks!

With these steps it usually takes somewhere between three weeks and two months for you to get an offer letter after submitting your resume. As you can see, each part of the interview process is designed to achieve a different goal for the company. Let's dive into each part of this process and to showcase your skills and capabilities in each setting.

7.3 Step one: the initial phone screen interview

Your first interaction with the company will likely be a 30-minute phone call with a recruiter. Since it's the first interaction it's important to make a good impression. However, depending on the size of the company the person you'll be talking to is likely unrelated to the data science team you'll be working on, so y*our goal is to show the company that you are a person who could do this job, not necessarily that you're the best person for the job.*

Why? Because the person you will be talking to in this phone interview has the job of filtering out the unqualified candidates. When the recruiter talks to a candidate, they are trying to assess if it's worth it for someone on the data science team to talk to the candidate. Lots of times people apply to jobs they don't have the skills for (or they lie about having the required skills), and so the recruiter wants to prevent them from moving onward. As a candidate, your goal here is to get the recruiter to understand that you are at least minimally qualified for the position.

The recruiter is likely to ask you these sorts of questions.

- "Tell me about yourself" – this is the recruiter asking for you to give a 1-2 minute overview of your background. They're looking to hear you describe previous experiences that relate to the job at hand. For example, if you're applying to a decision science position, they'll want to hear about your academic background and any jobs or project where you did an analysis. It's important that your answer falls in the 1-2 minute range. If your answer is less than a minute you'll seem like you don't have enough going on, and if it's over two minutes you'll seem like you don't know how to summarize a story.
- "What sorts of technologies are you familiar with" – this is the recruiter checking if you have the technical experience to do the job. Beyond this specific question, you should expect to be asked questions around math and statistics, databases and programming, and the business domain, just like from Chapter 1. You'll also want to list any technologies you know that are related to the position. In the case of data science that's tech like R/Python, SQL, and cloud technologies like AWS or Azure. If you don't have exactly what the job posting asked for (like Python instead of R), that's okay; just be open and honest about it. If you happen to know the tech stack the company uses, try and frame your answer around that.
- "What makes you interested in this position" – this is the recruiter trying to understand what drew you to the company in the first place. A particularly well thought out answer shows you do your homework and can get things done. Answering "I just clicked 'apply' on every data science job on LinkedIn" would show you have poor judgement. Don't

overthink questions like these; just demonstrate that you know what the company does and have a genuine interest in the role. As much as possible try and connect the role to your background and interests.

While the recruiter will ask you questions about you and your background on this call, it's also a time for you to get a better understanding of the position itself. The recruiter should spend at least 10 minutes on the call talking more about the role and the team you are interviewing for. During this time, it's important to ask questions to the recruiter so you can be sure you'll want the job and so that you can demonstrate a sincere interest in the role. These questions can include topics like travel, the company's culture, how the team is changing, the team's priorities, and why the role opened up in the first place.

It is possible that on the call the recruiter will try and understand your salary expectations. That could be either directly (e.g. "what are you expecting for a salary?") or indirectly (e.g. "how much do you currently make?"). In a positive light, this is the recruiter making sure that the company can afford to meet your salary expectations, so as to not waste time interviewing someone who wouldn't take what the company can offer. In a negative light, this could also lead to the recruiter trying to lock you into a salary that's lower than they would potentially offer since you're giving them information about your expectations. As much as possible avoid discussing salary until you're further in the process. *We'll cover salary negotiation in the next chapter on negotiating the offer.*

There are a few possible outcomes of the phone interview. The first possibility is that both the recruiter thought you could be a good fit for the position and you thought there is a chance you would want the position. In this case the recruiter will sound excited at the end of the call and will talk about moving on to the on-site interview. After the call you'll coordinate the date of the interview. The second possibility is that the recruiter thought you'd be a good fit but you decided you definitely wouldn't like the position. While you *could* go to the on-site interview even though you are certain don't want the job, it's best to let the recruiter know you aren't interested as soon as possible, as to not waste anyone's time.

The last possibility is that you want the position but the recruiter doesn't think you'd be a good fit. In this situation the recruiter will probably email you in the next few days letting you know the company is not moving on. If you find yourself in this position, it's extremely easy to take it personally and feel like you aren't a good data scientist—fight this feeling! Just because one job wasn't a good fit doesn't mean you won't find another one that is a good fit, and if the company is not interested you probably wouldn't like the position anyway. In a sense they are helping you dodge a bullet!

During the call, make sure to ask what the next step in the process is, and what the timeline is. The recruiter should tell you something like "we would bring you in for an interview next, and we should know in a week if that will happen or not." This is a totally normal and fine thing to ask. Don't ask the recruiter directly if you are going to move on to the next interview. That will put the recruiter on the spot and possibly make them feel uncomfortable—and they likely don't have the authority to make the decision anyway.

7.4 Step two: the on-site interview

Alright, it's the big day! The company invited you on site to do an interview and booked it for multiple hours. You took a day off work, you put on some nicer-than-normal clothing, and you are heading out.

What to wear to the interview

One topic with interviews that is often discussed and debated is what to where to it. For data science positions, this is complicated by the fact that the jobs can fall in all sorts of industries, each with its own culture and dress code. What is completely appropriate for one interview may be totally inappropriate for the next.

Your best bet is to ask your recruiter when scheduling the interview both what people wear to interviews with the company, as well as what the general company dress code is. The recruiter wants you to succeed and shouldn't lead you astray. Otherwise try to talk to someone who works at the company or a similar company. If all else fails assume that more bureaucratic industries have strict dress codes (finance, defense, healthcare), whereas young companies or tech companies have loose dress codes (startups, massive tech companies). Avoid extremes (sandals, shorts, cocktail dresses, top hats), and wear something you feel comfortable in.

The goal of this interview is for the company to understand if you would be able to do the job they are hiring for—and if you would do it well. Depending on the company and position they may have anywhere from three to ten different people coming in for an interview, each with their own strengths and weaknesses. The company wants to find the person who would be the best fit, or rather they want to find the first person they interview who would be pretty good at it.

That means that throughout the interview you'll be wanting to reinforce the idea that you can do the job well. That's different from being the smartest candidate, the candidate with the most years of experience, or the candidate who has used the most types of technology. No, you want to be a candidate who has a healthy balance of being reasonable to work with, has enough skills to do the work, and can get things done.

What will happen during these multiple hours of interviews? An on-site interview usually includes the following:

- **A tour of the workplace and an introduction to the team.** The company wants you to get an understanding of what working with them will be like, and potentially impress you with their free beverages and snacks (if available). This will take less than 15 minutes, but it's good to get a decent look around to see if you would be happy working there. Are the workspaces calm and easy to work in? Do the people look reasonably happy and friendly? Are the laptops eight years old? When you are walking around on this tour you'll often be making small talk with someone from the company. Watch out! This is part of the interview—if you come off as unpleasant or mean, that could affect the job-offer decision. Shoot for coming off as nice but most importantly, be your authentic self.

- **One or more technical interviews.** This can be anywhere from 30 minutes to multiple hours depending on how rigorous the company is. You'll be asked questions covering many topics and may need to do work on a whiteboard or computer. We'll go into more detail about these questions later in the chapter. The point of this interview isn't for the interviewer to understand what deepest topics you know, or if you can solve the trickiest problems. The point is to see if you have the minimally necessary skills to do the job. So your objective is to show you have what is needed.
- **One or more cultural interviews.** The point of cultural interviews is to understand how well you get along with others, and how well you get things done. You'll be asked a lot of questions about your past experiences, including how dealt with difficult situations and how you make sure projects come to fruition. You can be asked very general job questions like "tell me about a time you dealt with a difficult coworker" to more data-science-specific queries like "how do you deal with a data science project where the model fails?" These questions won't necessarily have right or wrong answers; they'll instead be open to interpretation by the interviewer.

Going through an on-site interview is emotionally taxing. You'll have to quickly switch from thinking about technical topics to questions about your self and your future dreams, all while presenting yourself in a professional and friendly way. Depending on the size of the company, there may be one or many people doing these interviews with you, and for each of these people you'll want to make a good impression. One of the best ways to manage nerves during the interview is to remember that the interviewers are people too, and they want you to perform well just as much as you do. They are your allies, not your adversary. With that, let's dive deeper into the different parts of the interview.

7.4.1 The technical interview

For many data scientists, a technical interview is the most frightening part of the whole interview process. It's easy to imagine yourself stuck in front of a whiteboard being asked a question you have no idea how to answer and knowing you won't get the job. Just writing that previous sentence induced anxiety!

To best understand how to handle the technical interview, we need to reframe how to think about it. If you completed Chapter 4 of this book and have created a data science portfolio, you've already passed the technical interview. The point of the interview is to see if you have the skills necessary to be a data scientist, and by definition have those skills since you did data science! If during an interview you find yourself being judged for not being able to answer a tricky question, then that is a sign the interviewer is doing a bad job, not you. You have the necessary skills, you have experience under your belt, and this part of the interview is for you express that. If the interview doesn't allow you to do that it's not your fault.

In this process you are going to try and show the interviewer that *you have the skills needed for the job.* Showing you have a particular set of skills is a very different activity than answering perfectly every question you are asked. A person can give exactly the answers an

interviewer wants and still do poorly in the interview, or they can give incorrect answers and do well. Consider two answers to a question in an interview:

> ***Interviewer:*** *What is k-fold cross validation?*
>
> ***Answer A:*** *You randomly partition the data into k even groups and use them as test data for k models.*
> ***Answer B:*** *You randomly draw a sample of data and use it as test data for a model, k-times. Then you take the average of the models and use that. This is a really great method for handling overfitting since you have a bunch of models that all have different training data. I used this method on the main project in my portfolio where I predicted house prices.*

Technically, Answer A is correct while Answer B is not (that's cross validation, but not technically *k*-fold since the data wasn't split into even groups). That being said, Answer A gave the interviewer no information except that the interviewee knew the definition, while Answer B showed the candidate knew the term, knew why it was used, and had practical experience with it. This illustrates why is it so important that during an interview you think about how to convey that you have the skills.

Specifically, there are a few things you can do in your data science technical interview that can help convey you have these skills:

- **Explain your thinking.** As much as possible, don't just give an answer, but explain why you got the answer you did. Giving an explanation provides the interviewer with how you think about the subject and can show you are on the right track even if you didn't get the right answer. One word of caution: while repeating the question aloud might feel helpful (ex: "hmm, would a linear regression work here?"), there are interviewers who may take this as a sign you don't know much. Practice answering questions directly and practice how you frame your thought process from the start.
- **Reference past experiences.** By talking about previous projects or work you've done, you are repeatedly grounding the conversation in your real-world practical skills. This can make an ambiguous answer more credible and concrete, or just give an alternative topic of conversation if your answer is a little off-mark. You have to do this in moderation—if you spend too much time talking about your past instead of talking about the question at hand it may seem like your avoiding it.
- **Be open and honest if you don't know the answer.** It's totally possible (and normal!) to not know the answer to every question in the interview. Try to be upfront about it and explain what you do know about the answer. For instance, if you were asked "what is a semi-join" and you don't know the answer, you could say something like, "I haven't heard of that kind of join before, but I suspect it might be related to an inner-join." Being open about what you don't know is better than confidently being incorrect—interviewers are often wary of people who don't know what they don't know.

Here are the general types of questions you'll see in the technical interview.

- Math and statistics – these questions test how well you understand the academic topics

that are a necessary foundation for a data science job. They include:

  - Machine learning – this includes knowledge of different machine learning algorithms (*k*-means, linear regression, random forest, principal components analysis, support vector machines), different methods for using machine learning algorithms (cross validation, boosting), and general expertise on using them in practice (like when certain algorithms tend to fail).
  - Statistics – you may be asked purely statistical questions, especially if your work is in an area that would need it, like experimentation. These questions can include statistical tests (like *t*-tests), definitions of terms (ex: ANOVA and *p*-value), and questions around probability distributions (like finding the expected value of an exponential random variable).
  - Combinatorics – the field of mathematics that covers all things related to counting. These are logical problems like "if a bag has six different colored marbles in it, how many combinations are there if you pull two out without replacement?" These questions have little to do with the job of a data scientist, but interviewers sometimes believe that they give insight into your problem-solving skills.

- Databases and programming – these are questions to test how effective you would be at the computer-based parts of a data science job. They include:

  - SQL – in almost any data science interview you will be asked questions about querying databases in SQL. This knowledge is needed for most jobs, and being familiar with SQL indicates you should be able to quickly get started in your new role. Expect to be asked questions on how to write SQL queries for sample data. For example, you may be given a table of student grades in multiple classes and asked to find the names of the best scoring students in each class.
  - R/Python – depending on the company you may either be asked to answer general programming questions by writing pseudo-code, or asked to solve particular questions using R or Python (whichever language the company uses). Don't worry if you know R and the company uses Python (or the reverse); usually companies are willing to hire and train in the new language on the job since there is a lot of transferable knowledge. Expect to have a question where you have to actually write code (ex: how would you filter a table in R/Python to only include the rows above the 75^{th} percentile of a score column?).

- Business domain expertise – these questions will be highly dependent on the company you're applying to. They'll be used to see how much familiarity you have with the type of work the company does—while this knowledge is something you can pick up on the job, the company would rather you had it than didn't. Here are some sample questions from different industries:

  - Ecommerce company – What is the click-through rate of an email? How does it compare to open-rate, and how should they be thought of differently?

    - Logistics – How do you optimize production queues? What are some things to think about when running a factory?
    - Non-profit – How should a non-profit try and measure donor churn? How could you tell if the churn is too high?

- Tricky logic problems - outside of problems that are related to things having to do with data science, you may get general "brain-teaser" questions in your interview. These questions have the reported purpose of trying to test your intelligence and ability to think on your feat. In practice, the questions don't do anything of the sort. Google did a massive study and found these sorts of questions had no ability to predict how a candidate would do on the job, and only served to make the interviewer feel smart. These tend to be questions like "how many shampoo bottles are there in all the hotels in the US?" For larger companies you can often Google to see if they use these types of questions (and which ones).

It's hard to say exactly which of these questions you'll see and how much time will be spent on each of them—it highly depends on the company and the person interviewing you. Try your best to remain calm and confident as you work through them, even if you are unable to answer some of them. If the interviewer is talking to you as you give a partial answer, they may be thinking that you're doing well and be willing to point you in the right direction. Often the questions are designed to cover so many different topics that no data scientist could answer all of them, so by design you'll have some you won't be able to get.

Interviewing the interviewer

Each time you meet with a new person during the onsite interview, they will end their part with "do you have any questions for me?" This is one of your only opportunities to get candid information about the job, so use the time wisely! Here you can find out more on the technology used, the work, and how the team functions. As discussed earlier in the chapter, it also shows that you have a sincere interest in the company, so it's worth thinking of questions in advance. Here are some example questions:

"What technologies do you use, and how to your train new hires on them?" This question is great if they haven't gone into the detail about the technology stack, and it will give you insight into if they have formal training processes or just hope employees pick up the knowledge on their own.

"Who is our team's stakeholder, and what's the relationship with them like?" This is you asking who is going to be calling the shots on the work. If the relationship is poor, you may end up being a slave to whatever the stakeholder's demands are, regardless of if that goes against your judgement.

"How do you do quality control on data science work?" Since the team doesn't want to put out work with errors, there should be some checks in their process. In practice, many data science teams have no checks, and blame the creator of the work when things go wrong. That's a toxic workplace to avoid!

7.4.2 The cultural interview

The cultural interview is designed to test your interpersonal skills and give the data scientists on the team a better understanding of who you are and what your background is. While technical questions will all be bunched into one or a couple blocks of time, the cultural questions may appear throughout the whole on-site interview. They can happen in a one-hour chunk with an HR representative, a ten minute window as closing questions from a technical interviewer, or even as small talk with one employee while you wait for another to arrive. So be ready to answer cultural questions at any time.

Here are a few examples of interview questions you should expect to receive:

- "Tell me about yourself" – This question showed up in the phone screening and can show up every time you start talking to a new person. Again, try and give a 1-2 minute summary, but this time cater it to the person you're talking to.
- "Describe a project you've worked on and what you learned from it." – This question is intended to see if you can look at a project in your history and be introspective about it. Have you processed what went well and what didn't?
- "What is your greatest weakness?" – This is an infuriating question because it *seems* like from a game theoretic perspective you want to answer with something that shows as little weakness as possible. In practice, what they are trying to do is check that you understand your own limitations and have areas where you are actively trying to improve.

Notice that the questions are all very open ended and have no right or wrong answer, however there are ways of expressing yourself that can dramatically improve how your answers are received.

For most of these questions, especially ones asking about past experiences, your answers should follow a general framework. (1) Explain the question in your own words to show that you understood it. (2) Give an explanation of a past experience where that situation occurred, focusing on why the problem existed. (3) Describe what action you took to resolve the problem, and what the result was. (4) Give a summary of what you learned from it.

For example, let's take the question "tell me about a time you delivered something to a stakeholder and you got a negative result." A response could be: "So you're asking about a time I disappointed someone with my work [1. explaining the question back]. That actually happened on my last job, where I had to do a report on customer churn. Our team was juggling a lot of different requests, so I didn't have much time to focus on a request from one director. When I handed over a report I only spent a day on, she was very disappointed with it [2. describing the problem]. So what I ended up doing was first apologizing for not meeting her expectations, then working with her to see how we could reduce the scope of the request and still meet her needs [3. providing a solution]. From this I learned it's best to let someone know early if you can't meet their request so you can come to a mutually agreeable solution [4. what you learned].

There is plenty more on this topic of answering these cultural questions, such as thinking about the wording and how long you should spend on them. That being said, most of the techniques aren't data science specific, so you can easily go deeper by reading general interviewing books and articles.

7.5 Step three: the case study

If you did well on the on-site interview you'll be asked to complete a case study: a small project that will show the company how well you do data science in practice. Someone on the data science team will you give a data set, a vague problem to solve with it, and a set time period to do it in. You'll generally be allowed to use the programming languages or tools you are most familiar with, although its possible the company will limit you to just the tools they use. After your time has elapsed, you'll go back to the company office and share your results in a short presentation to a group of people from the data science team. Some example case studies are:

- Given data on promotional emails sent by a company and data on orders placed, determine which of the email campaigns did best and how the company should market differently in the future.
- Given the text of 20,000 tweets where the company was mentioned, group the tweets into different topics that you think would be useful to the marketing team.
- An expensive A/B test was run on the company website, but halfway through the data stopped being consistently collected. Take the experiment data and see if there is any value that can be derived from it.

Notice that in each example case study, the objective isn't directly a data-science question. Questions like "which campaign did best?" make sense from a business context, but there isn't a "which campaign did best?" algorithm you can just apply! That is why these case studies are so useful as interviewing tools: they require you to go from the very beginning of a problem all the way to a solution.

With that said, what exactly is the company looking to see out of a good case study? They want to know:

- Can you take a vague, open-ended problem and figure out some methods to try to solve it? It's entirely possible you won't solve the problem, but as long as you make an attempt in a reasonable direction, it shows you have the technical skills and can get things done.
- Can you work with real-world messy data? The data you'll be given will likely require filtering, joins, feature engineering, and handling missing elements. By giving you a complex dataset, you'll be doing the kind of work that you would do on the job.
- Can you structure an analysis? Will you look at the data in a methodical, well-thought out way, or will you investigate things that don't relate to the task at hand?
- Can you produce a useful report? You will have to create a presentation about your

work, and possibly documents like Jupyter notebooks or R markdown reports. Can you make something that is useful to the company? Can you structure a useful narrative?

The good news is that the skills and techniques that are needed for great case studies are the exact same ones that you need for good portfolio projects: taking data and a vague question and producing an output. Even better if you made a blog post—that mimics creating a presentation for an interview case study!

There are a few minor differences between a portfolio project and a case study problem that you will want to take into account. First, in a case study you have a limited amount of time to do the analysis. This can be set by the calendar, as in one week from the day you get the materials, or set by the hours worked, as in no more than 12 hours of time spent. That short amount of time means you'll want to be strategic on where you spend your hours. In general the data cleaning and preparation steps take much longer than data scientists expect. Just getting tables to join together, filtering malformed characters out of strings, and loading the data into an IDE can all take a lot of time and typically that part of the work won't look impressive to the company. Try not to get overly focused on preparing the data as best as possible if it means having too little time to do an analysis.

Another difference with case studies is you are judged on your presentation of results. This means you want to have a well-polished presentation with really interesting results in it. However, the act of creating a presentation can feel both uninteresting and less important than doing the analysis itself, so many data scientists put creating the presentation off until the very end. Putting the presentation off until the end is bad because you may run out of time, or you might find out that the analysis isn't as interesting as you hoped and you don't have time to make changes. So as much as possible, start working on the presentation early and build it as you progress in your analysis.

One final difference with a case study is that you have an extremely specific audience: the small number of people you are presenting to. With a portfolio project you don't really know who will look at it, but with a case study you can hyper target your analysis. If possible when you are first given the case study, try and ask who the audience of the case study presentation will be. If it's all data scientists, then you can make your presentation more technical, like including details of the machine learning methods you used and why you chose them. If it's all business stakeholders, then try to be light on the technical components and focus more heavily on how your findings would impact the business decisions. If it's a mix of data scientists and business stakeholders, then try to include enough of each that if one of the group dominates the discussion you have enough to placate them.

The presentation itself is usually around 20-30 minutes of you presenting your findings with 10-15 minutes of questions from the audience on your approach and findings. It's a good idea to practice your presentation in advance and plan for what you want to say during each part of it. Practicing in advance also helps you keep to the allotted time—you don't want to talk for only 5 minutes or for 50 minutes straight. During the question and answer section you'll be peppered with questions on all sorts of topics. You may go from answering a question

about a parameter in your model straight to a question about the business impact of what you found. A good practice is to take a moment to think about the question before answering so that you can get your bearings and think through your response. In the event that you don't have an answer you are confident in, it's generally best to give some version of "I am not sure, but…" and give some ideas on how you could potentially find the answer. If possible, add the relevant context you do know.

7.6 Step four: the final interview

After the case study is over, and likely during the same trip to the office as your case study, you'll have one last interview. This interview will be with a person who makes the final call, such as the manager of the data science team or the director of engineering. Depending on how the company runs the interview process, this person may have already been prepared with information on how you did in the earlier parts of the process, or they may know literally nothing about it. The objective of this interview is for that final person to give the green light on you being hired.

It's hard to know exactly what questions you will be asked during the final interview because it depends so heavily on the person interviewing you. A more technical person may focus on your technical experience and what skills you have, whereas a business person may ask you more about your problem-solving approaches. Regardless of the type of person doing the interviewing you should definitely expect questions that are the same style as those in the cultural interview: "how do you handle difficult situations and deal with problems?" For these questions being open, honest, and sincere will go a long way.

Following up

After each part of the interview process, you may be inclined to contact people from the company in one form or another. Following up can be used to show gratitude to the people you've met with, as well as gain you more information on how the process is unfolding. However, if done poorly, reaching out can come off as abrasive or desperate and jeopardize your chances of getting the job. There are really three different ways you can follow up, depending where you are in the process.

Before anyone from the company has contacted you. If you have sent your application but haven't heard back do not follow up—them not responding is a sign they are not interested.

After contact but before you've met anyone in person. For example, after having done the phone interview, you should only follow up if you are unsure of your status in the process. You can follow up with one email only if it's past the timeline for when they said the next step would happen. In that case simply ask for a status update.

After in person contact. You can, but in no way need to or necessarily should, email the people who interviewed you a brief thank you. If you don't hear back past when they said you would, you can also email the recruiter asking for an update.

7.7 The offer

If all goes well within a week or two of your final interview. You'll get a call from someone at the company letting you know that they are going to extend you an offer. Congratulations! In the next chapter, we'll go into much more detail on what makes a good offer, how to compare different offers for different data science jobs, and how to ask a company to improve what they are offering.

Unfortunately, there is also the possibility of the disappointing case that you don't get an offer from the company. After taking a moment to grieve the loss of the potential job, you can view this as an opportunity to learn what areas you can improve for your next interview. If you only made it to the initial phone screening, that's a probably a sign that your base qualifications weren't a fit for the particular role. In that case, you should consider adjusting what job postings you are apply to. If you made it to the on-site interview or case study but no further, there is likely a specific reason that you weren't a fit for the company or role—try and deduce if there is something you could focus on for the next interview. If you made it through the final interviews but didn't get the job, that usually means you were well suited for the position, but someone else happened to be a slightly better fit. In this case there isn't much to do but continue to apply to similar roles. You shouldn't reach out to the company asking why you weren't hired—you're unlikely to get an honest answer and it's viewed as unprofessional.

7.8 Interview: Ryan Williams

Ryan Williams is a Senior Data Scientist at Starbucks in Seattle. He recently transitioned from being a manager of data science at a marketing and sales consulting firm, where he ran the data science interview process. He has a bachelor's degree from the University of Washington where he dual majored in statistics and economics. Prior to joining Starbucks his career was focused on the consulting industry.

7.8.1 What are the things you need to do knock an interview out of the park?

The big thing in general is just preparation. There's a whole skill set that goes into interviewing that is very specific to interviewing. I think a lot of people, often because they're currently good at their job, or they're coming out of academia and feel very well educated in their field, think they can just walk into an interview and their experience is naturally going to shine through. I have felt that myself, and I've seen other people come into interviews with that mindset. But there's really a skill set that you need in interviewing. And the way to prepare for that is to read up on the typical questions you're going to face. A lot of interviews are structured very similarly. No matter where you're interviewing, you're going to face some behavioral questions, you're going to face some technical questions, and you're going to face some business case questions.

So know the common behavioral questions. Know the types of technical questions people are going to ask and understand the business case type questions. Unless you're actually prepared in the right ways to demonstrate your experience, you're not necessarily going to get that opportunity in an interview.

7.8.2 How was your first interview different than how you interview today?

Thinking back to it, in my first interview I was just completely unprepared. I had the mindset that I would just be talking about the things I learned in my degree, that I know what I'm doing, and that I know I'm qualified for these jobs that I want. I applied for an actuarial job at an insurance company. I had just passed two of the exams and I thought I was super qualified for it. Then I completely bombed the interview because they were asking me a lot of questions about SQL, which is technology at the time I had never used. I felt woefully unprepared and I now know being able to manipulate data using a SQL database is a basic thing.

I did poorly on the behavioral questions, even though they were pretty basic. With questions like "tell us about a time you had trouble communicating," unless you've really prepared you can babble and talk in circles. Even though going into the interview I had all the things I needed to pass, because I hadn't prepared enough or hadn't done it in interviews before, it didn't go well.

Now that I have done more preparation and am more experienced, I do a lot better. The questions I face in interviews haven't changed that much during my career; I'm just prepared for them!

7.8.3 How do you handle the times where you don't know the answer?

When I can't answer a question, I at least try to explain how I would approach it. I've specifically run into situations in interviews at some companies where they just keep asking progressively harder questions until finally you're kind of stumped.

One case is that I remember I was interviewing at one of the bigger tech companies and they asked lots of harder and harder stats questions. It got to the point where they asked me something that was super academic. I think it was that given a certain probability distribution function, how would you use its moment generating function to find the kurtosis of the distribution. I was like well... that's something maybe I could have answered in college, but I definitely can't now.

That being said, when I gave my answer, the interviewer was clearly disappointed. I could go on a whole rant about it, but I wasn't happy because I feel like that was the type of question that's really more trivia than something that demonstrates my thinking. Having a job is about resourcefulness in your ability to solve things you don't know. It's not about your ability to come into a room knowing everything you need to know already.

7.8.4 What should you do if you get a negative response to your answer?

There's the emotional component where you're going to be unhappy with the fact that you were unable to answer the question, but you have to not let that tank the interview for you. It's natural to keep thinking about what you could have done to get that problem right instead of what you could do to answer the next question. But you really have to just be able to move on and stay sharp about the questions that you're going to get in the future. Don't let that one thing set you back.

I would say if you are running into the questions like that, you need to be interviewing the company too. They're asking you a lot of questions, but you should be using the types of questions they're asking you to infer whether this is actually a company you want to work for. For me, a company that thinks that it's super important for a data scientist to be able to derive a moment generating function three times to get the kurtosis is not necessarily the type of environment that I feel is going to be best for me to work in.

7.8.5 What has running interviews taught you about being an interviewee

I am a lot more thoughtful about the types of questions I'm being asked when I'm being interviewed and also the types of questions that I'm asking in an interview. Before I ever interviewed people myself, I took the interview questions at face value. Being interviewed felt like taking a test because the interviewer's asking a question and I thought I only needed to answer it right. I wasn't evaluating the types of questions being asked, which now I'm a lot more conscious of. Is this interviewer asking me questions that are a lot of trivia? Do they want me to whiteboard a lot of programming problems? Do they care about the things I care about in that data science role?

7.8.6 How can you interview the company during your interview?

The types of questions I try to ask in interviews help me try to understand how data driven this company is. A lot of companies hire data science jobs because they know that machine learning and AI are important, but that doesn't mean that decisions at the highest level are being made with data. If the company isn't using data for decisions your data science job could be unfulfilling. I tend to ask companies questions like: "how have you seen your work actually make an impact in decision making?" or, "when have you seen big decisions at the company be made with data versus opinions?"

You should also ask what types of technologies the data science team uses. The types of technologies can be very telling about the work they do. Some data science teams are going to be very engineering oriented, where they make machine learning programs in Python for production all day. Some of them will be working in SQL and Excel doing business casing or exploratory analysis. Both of those things were fine—I don't think you need to be snobby about the technologies, but you need to know that the environment you're going into is going to be stimulating in the right way. Then you're going to be working with the tools that you want to work with and learning the things you want to learn.

7.8.7 When giving interviews, have case studies changed the outcome?

Absolutely. I've had situations where someone kind of fudged through the earlier technical rounds—it seemed like they haven't prepared enough or they needed to be able to look it up afterward. Then they'd come in and completely nail the case study. The case study really tests a completely different skill set. Abstract interview questions test your ability to prepare for an interview, which is honestly not something I care about as the team lead. Whereas a case study tests your resourcefulness. I've had people ask me why I give case studies, and they push back like: "couldn't they just fake it?" or, "it's a take home test couldn't they ask for help or look up how to do things online?" Yes, that is the entire point! That's how a real job works: you have the whole world of resources at your disposal to actually get to a solution on something. Your ability to use those resources isn't present when you're doing an encapsulated interview in a room with an interviewer.

7.9 Summary

- The interviewing process is similar across companies for data science
- For on-site interviews, expect technical and behavioral questions
- Be prepared to do a data science case study at home
- Take time to prepare for and practice answers to common interview subjects
- Know what you want to learn about the company and role during your interview

8

The Offer: Knowing What to Accept

This chapter covers

- Handling the initial offer
- Negotiating your offer effectively
- Choosing between two "good" options

Congratulations! You've got a data science job offer. This is a big accomplishment, and you should definitely take a little time to savor it. You've done a lot of work over the past months or even years to get to this point.

This chapter will help you respond to and decide on the offers you receive. Even though you're probably really excited, you **should not** immediately say, "Yes! When can I start?" when you get an offer. All data science jobs are not created equal – you've been selective in where you've applied, but you may have found out things in the interview that raised concerns. Or you have a benefit that is a hard requirement for you, like good health insurance for your family, and you need to look over the offer details. Even if you are sure you want to take an offer, you should still not say yes immediately – you want to negotiate! Having an offer you haven't accepted yet is when you have the most power. Now that this employer has finally found someone they're excited about (you!), they want to close you. Recruiting is very expensive – it takes up a lot of time from HR and the data science team to evaluate applicants and interview, and every week without the new hire is a week where the company doesn't benefit from the work of that (hypothetical) person. Use this opportunity to ask for what's important to you, whether that's a higher salary, working from home once a week, or a higher budget for attending conferences.

8.1 The process

In general, the offer process goes something like this.

1. The company tells you an offer is coming—as soon as possible they want you to know about the offer so you don't accept an offer from a different company before theirs arrives.
2. They give you the offer—either by email or phone (then followed by an email), they let you know the details on salary, start date, and other necessary things for your decision. They usually also give you a date you have to accept or decline the offer.
3. You give them an initial response. As discussed in 8.2, unless you're absolutely sure you don't want to take it, we recommend expressing your enthusiasm and asking for a few days to a week to think about it, rather than immediately saying yes. When you have your next conversation with them, you'll begin the negotiation process (section 8.3).
4. They may be able to give you an answer immediately when you negotiate, but often they'll need some time to let you know whether or not they're to give you a better offer. In either case, you have to decide if this is good enough for you and return your final decision.

8.2 Receiving the offer

The call or email you receive with your offer is usually either with hiring manager or the recruiter or HR person you've been working with. In either case, your response should be the same.

Start by saying how happy and excited you are that you got an offer. If you sound unenthusiastic about working there, a company is going to get worried that, even if you take an offer, you won't stay there long and won't be contributing your best work. If you are very disappointed in the offer, for example if the salary was 25% less than you expected, you can say: "this wasn't what I was expecting but since I am really interested in this job, I will consider it and let you know how I feel."

They should tell you that you'll be getting the details in an email and if not you should request it. This serves two purposes. One, you get to sit down and take time to read through it all and consider the whole package, without frantically taking notes during a phone call and trying to decipher your handwriting later. Two, you should never consider a part of the offer "official" until it's in writing. While most of the time it won't be a problem, you don't want to get into a situation where there was a misunderstanding over the phone and you think you accepted a certain salary and benefits and it turns out that's not the case. When you receive the offer, it should include your title, salary number, any options or stock units offered, and the benefits package. If it doesn't include the details you need, like the full explanation of health insurance benefits, you can ask to get that as well.

Finally, they should give you a time-window to accept the offer. The time window should be at least a week, and if it's less than that ask for it. The best thing to do is be confident and simply say "I need a few days to consider this." If you find yourself unable to bring yourself to be that solid in your convictions then this is a great time to invoke someone else, like a partner, family member, or lucky goldfish, who you need to discuss it with. It helps to have an external decision-maker and constraint. That way, the recruiter or manager knows they can't just pressure you on the spot.

Sometimes companies will give you an "exploding" offer, where you have to respond in less than a week or the offer will be rescinded. It should never be as short as needing to respond in that initial phone call, but it might just be 24 hours. Usually there is room to push back on this by saying something like, "I know we both want this to be a great fit. Choosing my next company is a big decision, and I need a week to carefully consider it." In the worst-case scenario a company will refuse, for some reason, to wait for you to have the time to fully evaluate of the offer. If you find yourself in this situation it's a huge red flag. A company that isn't willing to respect your needs here is a company that won't respect your needs in other places. By giving you a small window, they know they're pressuring you to act quickly, which causes more anxiety and maybe make a worse decision. Despite many people applying to data science jobs, companies are still struggling to find people who are good fits, so a company that would treat people that way has something strange happening in it.

If you're interviewing with other companies, let those companies know you received an offer and when you need to respond to it by. If you're in the final rounds, they may be able to speed the process up so they can get you an offer before you need to decide on your current one. It's totally normal to update other companies when you receive another offer and they will appreciate it. When you reach out to them, reiterate how excited you are about their company and the work you'd do there. Some companies may not be able to do anything, but at least you give it a chance. Again, since companies are often struggling to find data scientists to hire, if they have someone they like in the interview process, they generally can move quickly to speed things up.

8.3 Negotiation

A lot of people hate negotiating a job offer. One reason is that it's perceived as a strictly zero-sum game: if you gain, they lose. Another is that it feels selfish and greedy, especially if the offer is better than your current job. It's easy to look at a situation and not feel like you deserve more money than a company offers, and that you're lucky to have any offer at all. But when you are in this position, you owe yourself to do your best to maximize your offer.

Because you, yes you, are the person best suited for this job. We all face impostor syndrome, but the company sees in you what they want for this job. As mentioned in the beginning, at this moment you're in your strongest position to negotiate. And companies expect you to! You want to be prepared. An appropriate salary for you isn't related to what you made before or what the company may initially offer, but instead what all the other

companies are offering for people with your skillset. Make sure that you get paid as much as your peers, that the total compensation matches your expectations, and that you get the benefits that are most important to you. You have a chance to make 5% of your salary in a five-minute phone call - you can handle the discomfort for that long.

The need for negotiating is especially true in data science, where there is a massive disparity in what people are making. Because of how new the field is, and how many different roles fall in data science, there aren't clear standards in what people make. Two people can have wildly different salaries with the same skillset, just by one person calling themselves a data scientist and another a machine learning engineer. These differences in salary get compounded as people move to new, higher paying jobs. At this point there is little correlation between the amount a person is making and the qualifications and abilities of the employee.

The first thing many people think of when negotiating a job offer is salary. Before you reach the final rounds of interviews, you should research the salaries of not just data scientists in general but for that industry, the city the company is located in, and the particular company if they already have data scientists. While the offer may be a big salary jump, remember that you want to be paid similarly to your peers at your new company. **You are not obligated to let the company know your current salary during the interview process, and in some cities and states it is illegal for companies to ask.** If you go on Glassdoor and find the average data scientist makes twice as much as you do, that's great. During the interview process when asked what you currently make, say "my salary expectations are [average salary from Glassdoor]."

You should not feel bad for being paid what you think is reasonable. What you think is reasonable shouldn't be defined by what you have made before, but only by what you want to make in the new position. Companies frequently offer a lower amount anticipating you will negotiate up to what they expect to pay a new hire, so it's very common you can get at least a 5% increase

But there's much more beyond salary you can negotiate for. First, there are other direct monetary benefits, like a signing bonus, moving allowance, and stock. A signing bonus is easier for the company to give than the same amount of salary increase, as it's only a one-time thing. The same goes for a moving allowance - if you'll be moving, ask about the company's relocation policy. If it's a big company, there's likely something standard and they may even work with a particular moving company to help. Even small companies may be able to give you something; you won't necessarily find out unless you ask.

But you can go broader even than that. Go back to what you thought about in the job search process – what's important to you? Some things are hard to negotiate – the healthcare options and 401k match for example are often set by HR and standard for everyone in a role. These are things that you want to consider earlier in the process – for non-startups, you can usually find employee reviews on Glassdoor with information about the benefits package. But there are lots of things you can ask for now, like:

- a flexible or remote work schedule

- an earlier review (six months vs. a year) which could get you a raise faster
- educational benefits
- a budget to go to conferences

When doing this, keep in mind the restrictions of the organization you are joining. Non-profits are unlikely to have much room to negotiate salary, but they might be flexible on hours or vacation. A company where the data team is already distributed is more likely to let you work from home a few days per month than if you join a team where everyone is in the same office. Ensure that what you're asking for is in writing and honored—you don't want to negotiate an early review and then when you arrive on the job never get it. Also keep in mind these sorts of non-salary negotiations likely will only make adjustments on the margins—if the salary is far below what you want, there is no way these changes could make the offer acceptable.

Your best negotiation lever is a competing offer. A competing offer lets a company know that there is a market for your services at a higher price and you have another place to go. There are some big companies that even will only negotiate if you have a competing offer. If you find yourself with two offers, it's best to make statements like "I would much rather work with you, but company ABC is offering me this much more; can you match that?" Don't lie; if you try to say that to both companies it could easily come back to haunt you.

Another piece of leverage you have is your current job – most companies know you won't want to take a pay cut. An exception might be if you're breaking into the field from something completely unrelated and have to move down some ranks in a company hierarchy. But if you're relatively happy, or have better benefits at your company, use that. Even if they may not be able to match exactly, you could get an equivalent bump in a different benefit or salary. For example, let's say your current employer offers a 3% 401k match and the offering company offers none. That match effectively raises your salary by 3% vs what the new company is offering. This can be especially helpful if you're transitioning to data science from a lower paying field, so your salary will increase a lot, but you'll be losing out a benefit you get at your current job.

When placing a value on your skills, consider how uniquely your background suits your new job. You don't necessarily have to be an experienced data scientist. For example, it would be great if you were one of the three people to earn their PhD in AI at Stanford this year. But let's say your new job is as a data scientist helping sales and you used to be in sales. Having that domain knowledge is a huge advantage that even more experienced data scientists might not be able to bring to the position. The more it sounded like the position was designed for you when you saw the job posting, the more leverage you are likely to have.

Sometimes it can be difficult to recognize the full picture of a job offer. If you'd need to move, for instance, you should consider moving costs and also changes in the cost of living. A $90,000 salary in Houston will go a lot farther than a $95,000 salary in New York City. Tally up all the extra benefits. Things like a good healthcare plan or 401k match can be worth thousands or tens of thousands of dollars. In general, don't lose the forest for the salary tree.

If you've negotiated and you get everything you asked for, they expect you to accept! You should only start the negotiating process if you'd take the offer if they met all of your conditions. Otherwise, why bother? One reason could be you have another offer you really want to take, and the better this one the more you can negotiate the other one. This is an extremely risky tactic, you want to be careful about possibly burning bridges. Even if you may never work at the company that you use as leverage, the employees who interviewed you or made your offer may remember.

It is very, very rare for a company to pull an offer because you negotiated. If they did, that's a sign you definitely do not want to work there. It is totally normal to respectfully negotiate and rejecting a candidate for that is a huge red flag. Take it as a bad situation avoided.

A brief primer on RSUs, options, and employee stock purchase plans

This is intentionally called a "brief primer," not an "exhaustive overview." We highly recommend researching more on whatever is contained in your offer letter.

Generally, all of these will vest over a four-year period with a one-year cliff. The "one-year cliff" means if you leave before one year, you get nothing. Instead, you get the first year's worth all at once at your one-year anniversary. After that, they will generally vest every quarter for the following three years.

RSUs: These are restricted stock units. You'll see a dollar amount given to you an offer. If you take the price of the stock at the time of your offer and divide the monetary amount by it, you'll get the number of shares per year. That means if the shares go up in value, your compensation goes up. When they vest, you'll get them as stock at that time – your company will usually hold back the amount of tax in the form of shares.

For example, let's say you get an offer of $40,000, vesting over four years with a one year cliff. Right now, the stock is trading at $100 a share, so you'll get 100 shares after one year, then 25 each quarter afterward. After working there for one year, you'll get 65 shares (35 held back for tax). At that point, you can do whatever you want with them, complying with the company rules (usually you can only sell stock in certain time periods). If you're at a private company, the vesting will work the same, but you won't be able to sell them.

Stock options: at a private company, there is no publicly traded stock available, so you'll generally get options instead. Options gives you the "option" to buy a certain amount of stock at a certain price. If the stock is trading way above that price, that's great news – if you can buy some for $10 and it's trading at $30, you'll make an instant $20 if you just buy them and sell them right away! If the stock does not trade above that value though, they're worthless – you don't want to buy a stock for $10 if you could buy it for $5. These are really great for early employees at a company that gets big. But they can be like lottery tickets. As long as your company stays private, the options have limited value, and even if the company goes public, the stock may end up trading below your option value. Unlike RSUs, which are generally going to be worth something, options may never be worth any money.

Employee stock purchase plans: this allows you to buy company stock at a discounted price. You contribute to the plan through a payroll deduction and then reach a purchase date, where your money is used to buy shares of the company. These offer two benefits. One, there is the discount on stock price, which may be as much as 15%. The other is a "lookback provision" – if the price has gone up between the beginning of the offering period and the purchase date, you'll get to pay the lower previous price.

For example, let's say at the beginning of the offering period the company stock is worth $10. You contribute $9000 over a year and then reach the purchase date. At a discount rate of 10%, you get to buy 1000 shares ($9000 / $9). If the stock price is now $20, those 1000 shares are worth $20,000, meaning you could sell them and get $11,000 in profit!

8.4 Negotiation Tactics

Now that we've gone over the big picture, let's walk through some specific negotiation tips:

1) Remember to start by showing you're thankful and excited – hopefully you are both! You want them to feel like you are on their side and you and the organization are working together to get a solution. If you don't feel like you're working together that's a red flag against working there.
2) Before the phone call, have notes about what exactly you are looking to change in the offer and what total compensation you would like. During the heat of the moment it's extremely easy to blurt out a lower number than you wanted to make a person happy in that moment. Having notes helps you avoid this.
3) Listen to what the person on the other end is telling you- if they stress that they don't negotiate salary but compensation is important to you, ask about more options or a signing bonus. Try and work with them to find a solution rather than trying to be immobile in your position. While negotiating can feel like a zero-sum game, but it doesn't have to be! Something very important to you may not be as important to them for example, so it's easy for them to give. And they want you to be set up for success and happy in your role.
4) Don't seem like you're only focused on money (even if you are). This can come off badly. You want to show you're motivated by the work you'll be doing, the mission of the company, and your new colleagues. Seeming like you're only in it for the money can leave a bad taste in their mouth, and they're more likely to worry you'll jump at the chance to leave for a higher salary.
5) Try to keep a communal focus. For example, instead of "I really need more options," try, "I'm really excited about the long-terms growth prospects of your company and how I can help you succeed. That's why I'm interested in being even more invested in them by having some additional options."

6) Package together your desires instead of going point by point. That way, they'll get the full picture of what you want instead of feeling like every time they settle an issue there might turn out to be another one. That being said, you can hold back the substitutes you're willing to accept: for example, you can ask for a higher salary, more options, and to work from home one day a week, but if they can't do the salary, you can then ask about a signing bonus instead. If possible, try to list the importance of each too.
7) Avoid self-deprecation. You read how great you are above? Read it again. Don't undercut yourself by saying something like "I know I haven't worked as a data scientist before" or "I know I don't have a PhD, but …" You have exactly what they're looking for or they wouldn't have made you an offer!
8) Have reasons for what you want. For example, if you're asking for a higher salary, you can mention it's because you're looking to make a down-payment on a house. If you want to work from home one day a week, talk about how you've found that taking the time to focus without the office distractions allows you to work on those long-term projects that provide your company huge benefits. It's true that you shouldn't need to justify your demands – as long as you're doing good work, a company shouldn't care whether you spend your salary on childcare or a hot tub. But we're human, and it's harder to say no to someone asking for $5,000 to buy a house than someone just asking for $5,000.

There's a lot more advice and research out there on how to negotiate effectively, and we've shared to some of our favorites in the appendix.

NEGOTIATING AS A WOMAN

For a while, the common research theory was "women don't ask" – that one reason women are paid less than men is because women don't negotiate job offers or ask for raises. However, recent research has shown at least in some areas, women do negotiate as much, they're just less likely to be successful. Unfortunately, there is some bias here. Some of the tactics we discuss in this section, like having a communal focus (using "we" instead of "I") and offering reasons for your asks are especially beneficial to women.

8.5 How to pick between two "good" job offers

Receiving two (or more!) good offers is a great problem to have! But you still have to make a choice. Of course, having multiple offers won't always lead to this problem – sometimes you prefer one job so much more that it's an easy decision. But what do you do if that's not the case?

The first step is to go back to earlier in the chapter – negotiate! If you really like one more, but there's some deal-breaker for you, see if they can change that. Be honest with the company about what you're looking for. As discussed, that competing offer is great leverage to have and makes it more likely you'll get what you're looking for.

When you've looking at an offer, consider the long-term. You may have a short-term concern – if you have high student debt or are starting a family soon, maybe you want to just take the best paying offer so you can start paying that down or saving. But if you're fortunate to be able to think beyond the immediate financial concerns, you want to focus on maximizing long-term potential. What kind of work will you be doing at each one? If one job will give you training to be able to move up two steps at your next post and one will have you doing things you learned on your first day at bootcamp, you shouldn't jump at the latter just because they have a better vacation package. Remember that salaries in data science can really jump from one position to the next, even by as much as 30-40%. If your resume is a little sparse and this is a good chance to put a prestigious name on there, then you might take a job at a slightly lower salary for a year or two.

Finally, don't be afraid to let smaller factors influence you. Does one have a shorter commute? A more spacious office? If they both meet your minimum criteria and are similar on your most important deciding factors, you can start bringing in the smaller ones.

Turn down any offers with grace. One, it's the polite thing to do, but two, the data science world is small. You don't want to reject an offer by saying, "I can't believe you gave me that lowball offer, you're a terrible company and completely unethical," and then find the person you yelled at is the hiring manager for your dream job five years later.

Choosing between two good life choices

In situations like this it's often possible where multiple life choices seem reasonable. Do you take the exciting but risky new startup job, or stick in your reasonable government contractor one? Do you take the offer that has you managing people, or continue to work as an independent contributor?

Often there is no objective way to decide with route is better. You can't tell if the new job would be a good fit until you actually take the offer. You can't know if you'll like managing until you become a manager. This is an infuriating fact of life.

If you find yourself struggling with these decisions, it can sometimes help to reflect on the fact that you can only do the best with the information you have. You can't expect yourself to see the future. If you make a decision and ultimately are unhappy with the outcome you can move on from it. You can quit a job, move back after changing cities, or go back to being an independent contributor. Life is complicated and you can learn a lot even from paths that don't lead to the outcomes you want.

8.6 Interview with Brooke Watson, on her experiences negotiating job offers

Brooke Watson is a senior data scientist at the ACLU, the American Civil Liberties Union, where she provides quantitative support for our litigation and advocacy teams on issues related to civil rights. She has a master's in epidemiology and previously worked as a research scientist.

8.6.1 What do you say when you get a phone call with an offer?

I try to communicate my excitement and my gratitude for the person's time without committing to any timelines. It can be easy to commit verbally right in that first phone call – I've certainly done that in the past, taking whatever I get because I really wanted a job. But that gives up your negotiating power and can start you off on a lower salary scale or lower title with less responsibility. Instead, I say I'm really excited to look over a written offer. I don't start negotiating until I have all the relevant offer details.

8.6.2 What should you consider besides salary when you're considering an offer?

It helps to know your priorities before going into that process. It's easy to compare salary, but I think you also want to compare the day to day or month to month experience of working at each place. I tend to think about the aspects of the job in three categories - lifestyle, learning, and values. Lifestyle is about the day-to-day way that the job interacts with other parts of your life: where will I live? Can I work remotely? Will I be working evenings and weekends? Will I get to travel? Will I *have* to travel? What healthcare, childcare, and retirement investment options are available to me? Do I have the flexibility I need to care for my family? That's the lifestyle bucket. There's also the learning bucket. Am I going to grow in this role? What systems are in place to ensure that I'm growing? Am I excited to learn from my boss and from the team? Finally, I think about the values and mission of the company or organization and the specific team. Does this organization and this team work toward something that aligns with my values? Does the team value inclusion? Is this a product or a team that I want to continue to build? Each of those three buckets can have different levels of importance at different times.

Salary is also not the only thing you can negotiate for. I've negotiated for a title change before in a previous role, and you can negotiate for things like education and conference resources, work from home days, travel support, or equity. Data science is a vast field that's always advancing, so I think it's important to protect time in your day to continue developing your skills.

8.6.3 What are some ways you prepare to negotiate?

In the past, I've been very uncomfortable talking about my own value and advocating for my own worth. That's something that I have been trying to unlearn throughout my career. What can be really helpful is having allies in your corner. Have a close friend practice the negotiation with you and have them take your part. Many people, myself included, find it much easier to advocate for our friends than for ourselves. Listening to the way someone else describes your strengths and your needs can be very motivating, and it really helps to put yourself in that mindset. Ask yourself, "How would I talk about someone I love, who I knew was a great fit for this role, and who I wanted to succeed?" That's how you should talk about yourself.

8.6.4 What are some negotiation tactics that you use?

If I have multiple offers, I bring up the differences or the benefits in the other offer that I would like to see more of in this one while reiterating that I'm still really interested in the role. If I'm negotiating for salary, I frame it in terms of my financial responsibilities to my family, which includes needing to think about paying my student loans and my rent. Keep in mind that you are an individual and you are negotiating with the company and companies have much more power than you. As much as you feel like it's just you and this hiring manager or this recruiter who are having a conversation like person to person, it's really person to company.

8.6.5 What do you do if you have one offer but are still waiting on another one?

If you've been through all, or most, of the interview process with a company you're waiting for, it usually doesn't hurt to mention to them that you have another offer. By that time, hopefully they are invested in you and will strive to avoid losing you, if they want to make an offer. You can also usually buy more time with the offering employer by asking for a week or two to review the details and discuss the offer with your family. Negotiating further, asking detailed questions about benefits, and asking to speak team members about the work or culture are all actions you can take to serve the dual purpose of collecting more information about the role, and buying more time.

If you're still very early in the interview process, this may not be enough time, and hurrying a company along at this point may not be very fruitful. Sometimes, you have to compare the offer on the table to your current situation. If you have just graduated or are between jobs, you may not have the financial luxury of waiting. You got the job! This is worth celebrating. However, if you are currently employed and still not interested by the offer on the table, it may be worth it to you to remain in your current role until you find a better fit. My first data science offer wasn't a great fit, and though I was very keen to transition into the field, it was worth waiting for a better match.

8.6.6 What's your final piece of advice for aspiring and junior data scientists?

I would advise aspiring and junior data scientists to keep an open mind about titles. I think the data scientist title has this allure for people, especially for those coming up in data science majors, which didn't exist when I was in college. They can feel if they don't get a job as a Data Scientist™ right when they graduate, they've failed. But the day to day functions of a data scientist might exist in a data analyst, a researcher, or one of many other kinds of roles, which can be a really great proving ground for you to hone your skills. I built my coding chops as a Research Assistant, and later as a Research Scientist, for years before anyone called me a "Data Scientist," and the work was just as engaging and interesting. Even data entry work as an undergrad has shaped my thinking about the way data collection decisions inform the analytical possibilities. No job is too small if you're willing to learn as you go.

8.7 Summary

- Don’t immediately accept an offer: get the details in writing and ask for some time to look it over.
- Negotiate, negotiate, negotiate! You can ask for a higher salary or other monetary benefits, but don’t forget about things like a flexible work schedule or a conference budget.
- When you’re weighting two good offers, remember to consider the long-term potential at each, not just the initial salary.

9

The First Months on the Job

This chapter covers:

- What to expect in your first few weeks as a data scientist
- Becoming productive by building relationships and asking questions
- What to do if you're in a bad work environment

This second half of the book will guide you through what it will actually be like working as a data scientist. You'll learn how to write an effective analysis, put a machine learning model into production, deal with business stakeholders, and handle inevitable failures. We'll then guide you to start thinking beyond this first job by joining the broader community, going on to your next job, and even looking years ahead to becoming a manager, consultant, or senior data scientist.

But before we get to all of that, we're going to walk you through what to expect in your first few months and how to use them to set yourself up for success. These months will have an outsized impact on how the job goes-this is your chance to set up a system and support network that will allow you to be successful.

When you start working, you will instinctively want to get as much done as possible immediately. Fight that instinct. You need to be sure that you are not just accomplishing tasks but doing them in the right way. This is the easiest time to ask questions about how something should be done, since you aren't expected to know this at your new company. Managers occasionally forget that you don't have the institutional knowledge that your predecessor may have had, so you might get tasked with something that doesn't make sense to you. You might be able to fake your way through the first few tasks, but you'll be much better served by asking questions early on and finding out how to approach your work process.

While each data science job is different, there are some broad patterns as well as principles that can be applied to any job.

9.1 The First Month

Your first month will look really different depending on type of company you're going. Large and small companies, or more specifically, companies with large data science teams or small ones, will approach your onboarding from almost opposite perspectives. Let's take a peek at what you expect from two companies, one that is massive with tons of data scientists, and one with no (or barely any) data science team. Back in chapter 2, this would be MTC and Seg-Metra respectively. These two examples highlight two ends of a spectrum, but the company you're working at could easily fall in between.

9.1.1 Onboarding at a large organization: a well-oiled machine

You are one of the dozens of people starting this week. You get an email the week before telling you where to go, when to arrive, and what you need to bring. You then began a formal, multi-day onboarding process with people from all different departments. Together, you're issued your laptop and go through setting it up. You listen to presentations on the company culture, HR policies, and how the company is organized. Everything runs like clockwork - they have done this thousands of times before.

On the data science side, you'll get help setting up your coding environment. There's likely a checklist or extensive documentation on everything you need to do to get access to the data. There's a central repository of old reports and documentation of the data for you to read and absorb. No one is expecting you to deliver much right away – while they're excited to have you join the team, they know you'll need time to adjust. You'll be expected to take a few weeks to go through all the training and get your access approved for the systems. You may feel almost frustrated that it's taking so long before you feel productive, but a slow start that takes time to go through a process is natural in this environment.

Again, if you are given a list of to-dos or an assignment, you should take it seriously but worry more about the process than the result. Established data science teams can often have their own idiosyncrasies that you'll need to adopt. It's not just good to ask questions at this stage, it's essential to your ability to perform your job later on. The first few months are your chance to see what's been done before you and get yourself well versed in the rhythm of your peers.

9.1.2 Onboarding at a small company: what onboarding?

"Oh - you're starting today?" If you're joining a small start-up, don't be surprised if everything is not ready, including your laptop. You may be left on your own to figure out how to access the data, and when you do get in it isn't well optimized for your job, it takes 6 minutes to run a SQL query on a small table of 100,000 rows. Onboarding sessions to learn about the

company may not happen for weeks, if there even are any, because there aren't enough people starting in a given week for that to make sense to run each week.

There are no standards to speak of. No one is telling you what programming language to use, or what the standard approach to making an analysis is. They are however asking you to quickly start getting results. Unlike at a large organization, you don't have to worry about not being productive—you'll be asked to do that right away. You do have to worry a lot more that you're accidentally doing something wrong and no one will tell you, or only finding out a few months later after your (incorrect) work is already being relied upon. That's why it's still imperative that you ask questions and work to get a foothold before you no longer have the fallback of being new. Going from crisis to crisis will cause you to burn out quickly, so work to build your own processes that will allow you to be successful in the long-run.

9.1.3 Understanding and setting expectations

One of the most important things you can do in your first weeks is to have a meeting with your manager to discuss priorities. This is important because it gives you the knowledge of what you are supposed to be working towards in your job, which allows you to work towards it. In some data science jobs, the priority is to provide analyses to a specific set of stakeholders to help grow a particular part of the business. For other data science jobs, the goal is to make high performing models that help run the business. For some jobs it's both or neither of these.

It may feel like you should already know what the job expectations are by the job posting and interview process. While this is sometimes true, a lot can change between the interview process and the job starting. The interviewers may not be on the same timeframe you're now working on, or the organization may have changed before your joined. By talking to your manager as early as possible, you'll get this information as soon as possible, and in a setting where you can likely talk about it for multiple hours.

Ideally, your manager has a vision for what you'll be doing but is open to your priorities and strengths. Together, you want to define what success means in your job. Generally, that is tied to making your team and/or manager successful—if the members of the data science team aren't all working broadly towards the same objective it can be difficult to support each other. To define your own success, you need to understand what problems the team is trying to solve and how performance is evaluated. Will you be helping generate more revenue by working on experiments to increase conversion, or will you be making a machine learning model to help customer service agents predict what concern a customer has, with the goal to decrease average time spent per request?

Performance usually does **not** mean "make a machine learning model with 99% accuracy" or "use the newest statistical model in your analysis." These are the tools to help you solve a problem, not the end goal itself. If your models and analyses are on problems people don't care about, they're pretty much useless. This is a common misconception from people entering their first data science job. It makes sense that it is common, since so much academic research and educational courses cover the many different methods of making

accurate models. Ultimately, for most data science jobs having highly accurate models isn't enough to be successful. Things like the model's usefulness, level of insight, and maintainability are often more important. The next two chapters will go into these ideas in more detail.

You can't know, when you start a new job, what the expectations are in terms of job responsibilities. Some companies value teamwork and you may be expected to help with several different projects at once and drop your work on a moment's notice to help a colleague. Other companies ask that you have deliverables on a regular basis and it's ok to ignore emails or slack messages in order to achieve your project. The way you find out if you are meeting expectations is by having regular meetings with your direct supervisor. Most companies have a weekly 1 on 1 so that you can discuss what you are working on or any issues you have. These exist so that you can find out if you are spending your time on the tasks that matter to your boss. Why should you guess at what is wanted when you can get explicit feedback? Thinking in shorter term blocks will help you be sure that you are on the right track for when the larger performance reviews come along.

Unless you're at a very small company, there will be a formal performance review process, so be sure to ask about what that entails and when it will happen. One common practice is to have one every six months, with salary increases and promotions potentially following afterwards. Many companies do this as a "360" process, where you get feedback directly not just from your manager but also your peers. If this is the case, find out whether you will choose the peers or if they're chosen by your manager.

For more established data science teams, there may be a matrix that says what areas you're evaluated on and what's expected for each of them at different levels of seniority. For example, one area could be "technical expertise." A junior data scientist may be expected to just have the foundations and show that they're learning, a mid-level that they have one area of expertise, and a senior that they're the go-to person at the company for a whole area, like A/B testing or writing big data jobs. If one doesn't exist, see if you can come up with a few areas with your manager.

Regardless of the system, make a plan with your manager to have a three-month review if it's not common practice. This review will help you make sure you're on the same page as your manager, give updates, and plan for the rest of your first six month and year.

The point of defining success is not that you already need to be excelling in every area in your first months; in fact, most companies won't do a formal performance evaluation of someone who's been there less than six because much of that time has been on-ramping. Rather, it's to make sure as you learn about your role and begin work that you do so with the big picture in mind.

9.1.4 Knowing your data

Of course, you do need to learn about the data science part as well. If your company has been doing data science for a while, a great place to start is by reading reports they've written. Reports will tell you not only what types of data your company keeps and some key insights,

but also the tone and style of how you communicate your results. Much of a data scientist's job is conveying information to non-technical peers and by reading reports, you'll have a sense of just how non-technical those peers are. See how simplified or complex certain concepts are and you'll be less likely to over or under explain when it comes time to write your own reports.

Then you'll need to learn where the data lives and get access to it. This includes knowing what table contains the data you want, but maybe also what data system has it. Perhaps the most frequently accessed data lives in SQL, but the event data from two years ago lives in HDFS (Hadoop Distributed File System) that you need to use another language to access.

Take a broad look at the data you are going to be working with on a regular basis but go in with an open mind. Some tables will have documentation (either packaged with the data or in a report about the data) which will explain potential quality issues or quirkiness. Read those first, as it will keep you from investigating "mysteries" that later turn out to already have been solved. Then take a look at a few rows and summary statistics. This can help you avoid "gotchas," where you find out that some subscriptions start in the future or that a column often has missing values. When you find these surprises that aren't documented, usually the best way to figure them out is to talk to the expert on that table. That may be a data scientist, if your company is large enough, or those who collected the data. You might find out that they're a true issue that needs fixing, or it might turn out to be expect. For example, subscriptions that start in the future could be those that have been paused and set to restart on that date. Or the coupons for last year's New Year promo you find used in May of this year is because support issued them.

Some companies are better than others about having data that was created for testing separate from real data, while others merge the data without a second thought. In the latter case you would want to ask around if there are orders or activity generated by test accounts or special business partnerships that you should exclude. Similarly, some datasets will include users with radically different behavior. American Airlines once offered a lifetime flying pass which included a companion fare. One of the people with the pass used the companion fare for strangers, pets, his violin, and might fly multiple times a day. While you may not have anyone so extreme, it's not uncommon for newer businesses to offer deals that later look silly (e.g. 10 years of access for $100) which may need to be accounted for in your analysis.

Throughout this process of investigating the data you're figuring out what kind of overall shape your data is in. If you're at a smaller company, you may find that you need to work with engineers to collect more data before the overall data would be useful. If you're at a larger company, you'll be deciphering which of the dozens of tables to see if what you want exists. Maybe you're looking for tables with a column called "order" across 12 databases. Ideally, there should be well-documented, well-maintained tables for the core business metric, like transactions or subscriptions. But that is likely not to be the case for other less-important datasets, and you should try to learn more if you are going to be focused on one of the less documented areas.

Make sure you learn how the data got to you. If you're working with something like website data, it'll likely flow through multiple systems to get from the website to database you can use. Each of these systems likely changes the data in some way. When data collection suddenly stops, you want to know where to try and find the problem (rather than panicking). But some places will have data that people input manually, like doctors in a hospital, or if you deal with a lot of survey results. In these situations, you have to worry less about pipelines, and much more about understanding the many different attributes of the data and potential places where a human inputted it incorrectly. Pretty much anywhere you go you will have to deal with some data dirtiness.

As you go along, try writing down any "gotchas" in the data and a map of where everything lives. It's difficult to remember these sorts of facts over the course of a job, and many companies don't have great system for documentation or data discovery. Just like commenting code so your future self and others can understand its purpose, documenting data provides enormous dividends. While keeping this documentation locally on your laptop is okay, the best thing is storing it somewhere that everyone in the company can access. You'll be helping future new hires and even current data scientists at the company who aren't familiar with that specific area.

9.2 Becoming productive

Eventually you should be making your manager look good and lightening their workload, but at the beginning you will make it harder and that's expected. It will take you longer than you think to get fully productive. It's normal to feel frustrated during this time, but remember you're dealing with a lot of cognitive load from being in a new environment. You're trying to pick up on the (likely unspoken) norms about how long people take for lunch, what the working hours are, what forms of communication to use, whether everyone closes their laptop when they walk away from the desk, and more. On top of that, you have a whole data system to get to know.

It's important to stress that it's an easy mistake to make to assume that you have to prove yourself quickly and early, like "I need to do everything faster or else they'll wonder why they hired me". This is a case of impostor syndrome. Unless you're in a truly dysfunctional company, they'll expect some ramp-up time. Instead of proving yourself quickly, focus on positioning yourself to deliver value in the longer term (within months, not weeks). Early on, you're going to be asking more of the company ("can I get access to this, why is this query so slow") than you're getting back (in the form of reports and analyses).

That being said, you can still deliver some value early. Focus on simple and entirely descriptive questions, like "What is the distribution of our client sizes" or "what percentage of our users are active each week?" In the process you'll familiarize yourself with the company's data, but also find some of the snags and traps that are waiting. During your check-ins with your manager, show off some of your in-progress work so that you can see if you are headed in the right direction. It's frustrating to put in a lot of time and find yourself answering the

wrong question, using a methodology that your boss hates, or that you're using the wrong data source.

Focusing on simpler questions also keeps you from embarrassing yourself by giving an incorrect conclusion because you are try answering a complicated question without learning all the details of the data first. This can be challenging because if your stakeholders are new to data science, their first question might be something like "predict which sales deals will close" or "how can we maximize user retention." But as we'll talk about in Chapter 12, one of your jobs as a data scientist is to dig into the business question to find the data question behind it. If people don't know or have misconceptions about basic facts (e.g. what percent of users churn month, how many people click on ads), they won't be asking the right questions.

There are two strategies that can help you become productive more quickly: asking questions and building relationships. Asking questions helps you more quickly understand the details of your job. Building relationships allows you to understand the context of your role in the organization.

9.2.1 Asking questions

One of the biggest things that can hold you back in your career is being afraid to ask questions or say I don't know. As we've discussed before, data science is such a big field that no one knows everything, or even 20%! And there's no way you could know all the intricacies of your company's data. Your manager would much rather you ask questions and take up a few minutes of someone's time then be stuck and spinning your wheels for days. Questions that are helpful can be anything from a technical question like: "what statistical test do we use for detecting a change in revenue in an A/B test?" to a business question like "who in the company decided to launch this product?"

That being said, all questions are not created equal. First, try to learn by observation about the question culture at the company. Do people ask questions in person, on a slack channel, in forums, or by email? Getting the channel right means you're less likely to bother someone. You also can ask your manager the meta-question of how to ask questions. Second, it's good to show you've been proactive. You might say "I researched this", "I'm excited to learn", or "this sounds like X, Is it?" By having done some research yourself, you may be able to answer the question on your own and you will be able to ask questions with more of an understanding of the concept. Unless it comes up as you're working with someone or discussing an issue, you want to avoid asking questions that are answered by the first Stack Overflow result on google (e.g. "What's the difference between a vector and a list in R?").

While some of your questions will be more general, as the above, you might also have deeply technical questions. Finding out who the expert is on various statistical or programming methods is important as that person is generally the one you need to get answers from. You don't want to become a burden on this type of person (or anyone) so if you realize that you have many questions for a particular person, try to schedule a meeting with them. People are much less likely to feel put upon if they choose to schedule a meeting rather than facing questions every few minutes. It can also help to ask that person's style. Some people within

the company have the role of supporting others, but if that person is also required to have deliverables, see if they have a calendar that blocks out certain times where they aren't reachable and respect that.

Third, avoid voicing criticisms veiled as questions, "Why do you code this request this way instead of the clearly better way I learned in undergrad?" Try to genuinely understand why things are done the way they are. If the company's been around a while, there's a lot of technical debt – if they have physical servers, moving that to the cloud is a half-year plus of work for dozens of engineers. When people wonder "why don't we just do X, it's so easy and would save us a lot of time," they often assume it's because other people don't understand it's a problem or feel that it's urgent. But the reason they're not doing X may be because of things you have no idea, like legal constraints.

A great way to learn is by pairing with people. Instead of just asking questions and getting an answer, you can see how they found that answer. If it's a technical question, it's also a way to see someone's coding environment and learn new techniques. Even if your question is about how to get data, you can learn what table it is, how they knew which table, and maybe some coding tricks. Your eventual goal is to be able to answer as many questions yourself as possible by knowing where to look.

Finally, if you have questions that don't need an immediate response, try to keep a side list of things that might be useful to know (How often does the data refresh, what is the size limit for queries, how far back does x data go in the local server, etc.) and go over them in a block with your mentor or manager. This avoids the situation where you interrupt someone over and over, which can become a bother if you're not careful.

9.2.2 Building relationships

An important part of feeling comfortable in your new work environment is to build a support network. Some people do this more easily than others, but you want to make sure that you engage in some non-technical talk with people. In most cases this means setting up meetings with people you've never talked to before to get to know them and their work. This isn't wasted time; it will allow both you and your coworkers to feel more comfortable relying on each other if you know more than your names and job titles.

Approaching someone you don't know can be daunting, but you can use your questions as a way to start a conversation with someone. People like being helpful and feeling knowledgeable so don't be afraid to use questions as an in as long as you are polite and friendly in your queries. Once you know a few people, even the largest offices feel less intimidating. It's also normal to message people you'll be working closely with to ask if you can set up a thirty-minute meeting to get to know each other. If you work in a large office, ask your manager to help make a list of people you should get to know.

At any sized office, it's good to find out who to go to for specific questions. Someone might be the company's best at SQL and someone else might be in charge of the experimentation system. It's very helpful to know who to turn to when you face a technical hurdle, and it usually isn't your manager. At least introduce yourself to your skip-level boss. Knowing your

boss's boss isn't important because you'll need to tattle on your boss, but because making yourself known to them will make it easier when they have to discuss you with your boss.

Similarly, you should try to meet with all the stakeholders you'll be working with. If the data science team is under 10 people, try to meet with them all individually. If there are data engineers or other data people you'll work with, talk with then. These can be informal, but it's important to not exist solely as an email signature. Even if you mostly work remotely, try to use facetime or skype so that people can see your face.

Do a lot of listening, both in official meetings and in social opportunities like over lunch. Meet people who work in data-adjacent areas (which could be everything from engineering to finance to sales ops to marketing analytics) and hear about how they currently do their jobs. Don't rush in with "I could do that better" or make commitments early like "we'll build you a machine learning platform to do that;" simply focus on collecting information and thoughts. And don't forget that everything doesn't always need to be about work. It's nice to get to know people on a personal level, whether by asking about their weekend plans, favorite tv shows, or their hobbies.

One last word to the wise: befriend the office manager. Office managers control a lot of the things that can make your day better: the snacks, the lunch order, what type of hand lotion is in the bathroom. They also have one of the hardest and most thankless jobs around, so make sure they feel appreciated.

9.3 When it's not what was promised

Entering a data science job and finding it's nowhere near what you expected can be crushing. After months of work, you've finally broken into the field, and now you might have to go do it all over again. Worse still, you may worry if you leave quickly it will look terrible on your resume. Does that mean you have to put in a year? Managing a bad environment and deciding whether to leave is challenging. We'll cover two main categories of problems: that the work is terrible and that the work environment is toxic. While there's no silver bullet to solve these, we'll walk through some potential mitigation strategies.

9.3.1 The work is terrible

First, take a hard look at your expectations. Is the problem something along the lines of, "All my data isn't cleaned! I spent two days just on data prep! The data engineers don't fix everything immediately!" That's going to be part of every data science role. Even data scientists at the biggest companies with hundreds of engineers deal with this problem - there's so much data it's impossible for it all to be perfectly vetted. While the core tables should be clean and well-documented, you'll likely encounter data in your sub-areas you need to improve a lot or work with others to collect.

One way to check how realistic your expectations are is to check them with other data scientists. If you graduated from a related degree or a bootcamp, talk to your peers or people in the alumni network about what they think of the data environment you're working in. If you

don't know many data scientists yet, try to go to meetups or join online communities if you're remote (we'll cover this in-depth in Chapter 14). If there are other data scientists at your company who've had previous data science jobs, see if they can give you an idea of how this compares.

Another situation may be that the work is tedious and boring. For example, the job you got hired to do might have been forecasting, but in practice all you do is hit the rerun button on someone else's existing forecasting model once a month. In that case, see if you can carve out some side projects in the organization or automate some processes. If it's boring work but not time-consuming, take the opportunity to do things that are data science related. Continue to build your data science portfolio of side projects and write blog posts. Take online courses. These will help position you for your next role.

You can learn even from bad jobs. Is there any way you can tailor your job so that you do more things where you'll learn? What are the areas for you to improve upon? Maybe your peers won't help you learn how to write better code, but can you learn about pitfalls of building a data science team that are easy to make? It's very likely there are some smart and well-meaning people in your company, what happened to make it bad?

9.3.2 The work environment is toxic

The previous section covers a bad but manageable situation. But what if your work is really toxic? What if your manager and stakeholders have completely unrealistic expectations and threaten to fire you because you're not making progress on predicting lifetime value when they have no data for it? Or you get penalized when your answers don't meet the company's expectations? Companies newer to data science may expect you to sprinkle your data science fairy dust to solve the company's core problems. They may ask you, "Build a model to tell whether text is 'written well'," where the field hasn't even come close to solving that problem. In this case, you need to adjust the expectations or risk feeling like a constant underperformer. Speaking up for yourself in this circumstance is difficult, but there are usually reasonable and smart people working for any company. If they say, "If you were a better data scientist you would be able to do this," that's a huge red flag. Even if a much more experienced data scientist could tackle the problem, that should have been recognized when they were designing the role and hiring.

Maybe the problem is instead that individuals or teams aren't collaborating. Instead of seeing where they can help, teams are constantly trying to sabotage each other. The focus is only on how they can get ahead, and they may even see it as a zero-sum game – if you or your team is doing well, that means that ours is losing. Besides creating an unhealthy environment, this often leads to a lot of wasted work as you may end up duplicating someone else's project because they wouldn't share their data or learnings with you.

The problem may have nothing to do with the data science part at all, but the environment is sexist, racist, homophobic, or otherwise hostile. There is no reason you should feel uncomfortable going to work every day. Even if it's not openly hostile harassment, getting

constantly talked over in meetings, not having your pronouns used, or getting asked "but where are you really from?" all add up.

Unfortunately, these types of problems usually need the top leadership and active engagement from everyone to solve, but the fact that it exists is often indicative that leadership is non-existent or even actively contributing to the problem. If the problem is rooted in one bad person, hopefully others will recognize it and they will be removed, but if it's widespread it may be close to impossible to change and trying to do so as a junior employee is a quick recipe for burnout. In these situations, you want to think carefully about whether you need to leave.

9.3.3 Deciding to leave

Deciding whether it's right to leave your job is an extremely personal decision. While no one can give you a simple flowchart that will make the decision painless and easy, we can offer some questions to think about to guide your decision:

- Do you have enough savings, a partner with a second income that can support you, or family that you can ask for a loan from if you leave without having another job?
- Is your job affecting your health or life outside of work?
- If the issue is the work, have you talk with your manager about the issues and tried to resolve them?
- Is it possible to switch teams or roles, if not now than within a few months?

If the answers to these questions make you feel that you need to leave, one option is to immediately start looking for other jobs. But you may worry about how having a short job stint on your resume will look or how you would explain it to interviewers. If it's only been a few days or week and you came directly from your last job, consider getting in touch with your previous manager. It's likely they haven't filled your position yet and if you left on a good note, you may be able to go back.

If you are looking for new jobs, a few tips on how to handle talking about your short stint in your interview. First, wait for them to bring it up. Don't feel like you proactively need to talk about it – it may not be a concern, especially since they're clearly interested since you've got to the interview stage! Second, find some positive experience and learning from the job you can talk about – a project you worked on, the exposure to the industry, learnings from a senior leader. Finally, when asked about why you're leaving so soon, you can say something like, "The requirements of my job weren't what I expected, and I wasn't able to use my skills and expertise to benefit the company." If you learned something about the type of work environment you want, share it. For example, maybe you were the first data scientist at a company and you realized you want to be a part of a larger team. If you've decided to leave, check out chapter 14 for more on how to do so gracefully, including searching while working full-time and leaving on a good note.

But maybe you can't leave, whether because your visa is tied to your workplace or it's the only company doing data science in your small town. If that's your situation, here are some tips:

- You are not your job. You don't have to take responsibility for poor decisions the company makes. Unless you're in a leadership position, you likely have little control over what the company does.
- Try to keep yourself healthy and not sacrifice sleep, exercise, and time with friends and family.
- Talk to someone about it. Maybe that's a partner, a friend, or a therapist. There may be some cases where they have advice, but just a listening ear will help.
- If you're experiencing harassment from a specific person, consider reporting it to your human resources department. Make sure you document your reporting - don't drop by HR in person; send emails and get emails back so that you have a record of the process. You may, eventually, want to point out certain things that were said to you, and having a written record will be helpful. If nothing is done, you can then file a claim with the Equal Employment Opportunity Commission if you're in the United States. Even if you don't want to report it, consider keeping documentation of any harassment you experience in case you decide to at a later date.
- See if you can think outside of the box of how you can leave the company. Maybe you feel you "can't" leave because you don't want to have a short job stint on your resume, the only options available are to go to a less prestigious company or get a lower title, or you'd need to temporarily deplete your savings. But don't underestimate the negative effects of staying in a toxic environment – if you can make a short-term sacrifice to leave, it will likely be worth it in the long-term.

9.4 If you're the first data scientist

Everything up to this point of the chapter was for the first few months of any data scientist position but being the first data scientist in an organization provides its own unique set of challenges. Given how new the field is and how many small companies do not have data scientists, being the first one isn't uncommon. Understanding that, as the first data scientist you should be especially prepared for when you begin.

When you start in your new position there will be absolutely no precedents. No one has decided yet if Python, R, or some other programming language should be used. No one has figured out how to manage the work—should software development practices like agile be used to decide what to work on, or should you do whatever you feel like that day? How should the code be managed: should a GitHub professional license be bought, a Microsoft TFS server be used, or can you keep all the files in your "My Documents" folder on your laptop without backups?

Because there are no precedents, everything you do will implicitly be a precedent. For example, if you happen to enjoy doing your work in the obscure programming language F#,

then you are forcing the fact that the next data scientist will have to learn F#. It is in your best interest to make decisions that will be to the benefit of whatever the team looks like down the line. That may mean using a more common programming language than the one that is your favorite. This has to be balanced with the fact that focusing too much on the future can cause serious harm to the present. For example, if you were to spend 3 months setting up a beautiful pipeline for automatically sharing reports with other data scientists but the second data scientist isn't hired for 5 years, then that work was a waste. Every day you will be either directly or indirectly making decisions with large consequences.

Besides having to figure out the role on your own, you also have to do the task of selling data science to the rest of the organization. Because the company hasn't had data scientists before, most people won't understand why you are around. The quicker people get your role, the more likely they are to want to work with you and keep a data scientist around. Therefore you job will require you to be constantly explaining data science in general to people, and also what in particular you can do to help the business. Sitting quietly in a corner working on models for months is possibly okay if you're the 20th data scientist on a team, but certainly won't work as the first.

While being the first data scientist is a lot more work and a lot more risky that other positions, it also has great payoffs. By making the technical decisions, you get to choose things that are more in line with what you want. By selling data science to the organization, you get well known and more influence. As the data science team grows you are in line to be the head of it, which can be great for career growth.

9.5 Interview with Jarvis Miller on his first months at Buzzfeed

Jarvis Miller is a data scientist on the product insights team at Spotify, focusing on personalization. When this interview was conducted, he was working as a data scientist at Buzzfeed. He graduated in 2018 with a master's degree in Statistics.

9.5.1 Can you tell us about your current role?

I'm a data scientist at Buzzfeed on the Tasty team, the world's largest food network (it makes the top-down, fast-forward cooking videos you might have seen on Facebook or Instagram). I help our product managers and engineers make data informed decisions about the path that Tasty should be on. For example, right now we're working on launching a new feature, so in my day to day, it's a lot of meetings and brainstorming not only with my team, but with other people and different teams at Buzzfeed that have done something similar. I'll write SQL code to get an idea of historical baselines whenever we've added similar features on other parts of Buzzfeed to get an idea of what success has looked like before and adapting that based on our current team strategy. I'll also use little bit of python and R for visualization and making charts.

9.5.2 How did you feel right before you started your first data science job?

I was nervous because I interned there the previous summer and I was worried that 1) they expected me to remember everyone's names and 2) they thought I remembered every single thing going on at Buzzfeed. I put these super high expectations on myself that I should be able to hit the ground running and make an impact. I didn't give myself slack to say, "Hey, as an intern, I worked on a very specific problem and didn't really interact with lots of teams."

After the first day, I was honestly exhausted. There was a lot of talking about projects and meeting people. While I have this extroverted persona, I'm not great at meeting a lot of people at once. I was feeling socially tired – there was a lot of things to process and a lot of people to meet, and I was already forgetting names as I read through documents. It was definitely a doozy of a first couple of days, just getting reacquainted with everything all over again.

9.5.3 What were some things that surprised you?

Two things that surprised me were how much I could improve as a writer and how I needed to explain my data science contribution to the business without using jargon. Learning to do less of "I ran a logistic regression on this data to classify." I had this idea that people work with data scientists, so they have learned to adapt all the language and thus I don't really have to change the way I explain things. I totally realized that that's not the case. On the being a better writer side, I've begun to flesh out the story when I write a report, improve my data storytelling abilities, and explain things in a way to where product managers, designers, and stakeholders who aren't in tech at all can understand what I'm saying.

I came from academia, where I felt it was all about whether you found the result at the end of the day - it didn't matter whether you started working right before the deadline or if you planned way ahead. In industry, you have a big overall goal but you figure out how to break it down into versions. You get the first version working, ship it, learn about whether it's doing well or not, and maybe improve it in a future quarter. I was used to going until it is done. But here, I had to learn how to prioritize parts of a project and then wrap it up. I'd document what I have done, what there is to do in the next version, and make it shareable, whether that's by putting a report into a shared folder or making an app so people can use it and see what is supposed to do.

9.5.4 What are some issues you faced in your first few months?

Speaking out was something I really struggled with. When I started, I was doing an isolated project and the person I reported to was in New York and I was in LA. If I was confused, I didn't know whether I should message them immediately or save it for our meeting. I knew I didn't want to be derailed by something that was blocking my work, but I wasn't even sure when something is a blocker. I think this is a common problem for data scientists, especially from those in marginalized groups or who switched from a different field. They may feel like that because they're new or not experts, they can't express displeasure or voice an opinion. If

I could go back, I would have a conversation sooner about how I'm feeling isolated and how I'm not sure how communication works.

9.5.5 Can you tell us about one of your first projects?

One of them was to revamp our A/B testing platform, which was a very broad problem. I started by getting a list of people to talk with about what they did at Buzzfeed, how they worked, and how A/B testing fit into that workflow. We then discussed the specific tool – what did they dislike and why, and what was their workflow when using it? Unfortunately, this led to the issue of taking on too much. A lot of people had multiple suggestions and I gave them all equal weight, which ended up with fifty big things that I needed to do. But my manager asked me to break those suggestions up into the must haves and the nice to haves, including the reasons why these those were prioritized are the way they are. He suggested listing the overall goal of the project and giving ideas weights based on their contribution towards the goal and how long they will take.

9.5.6 What would be your biggest piece of advice for the first few months?

Remember that you've been hired for a reason - they respect your point of view and they think they can help you learn and that they can learn from you. If you have an opinion, try and let someone know. If you hate speaking in a big group, maybe message one person, run it through with them, and bounce back and forth on ideas before you express it in front of the larger group.

This doesn't just apply for the technical side of the job. For me, in the first few minutes of my one-on-one with my manager, I don't want to immediately jump into what I've done. I want to have a few minutes to have a casual conversion to destress and clear my mind. I know that this helps my productivity and my company wants me to be productive, so I should let them know. Your opinion is valued and it's worth sharing, especially if it's about how you like to be treated or how you can be most productive and flourish in this role, because they don't know you like you know yourself and your being productive will benefit everyone.

9.6 Summary

- Don't worry about becoming fully productive right away – instead, focus on building relationships, tools, and your understanding of the data, which will make you productive in the long-term
- If you're in a bad work situation, try to work where you have control to mitigate the impact on your health and career