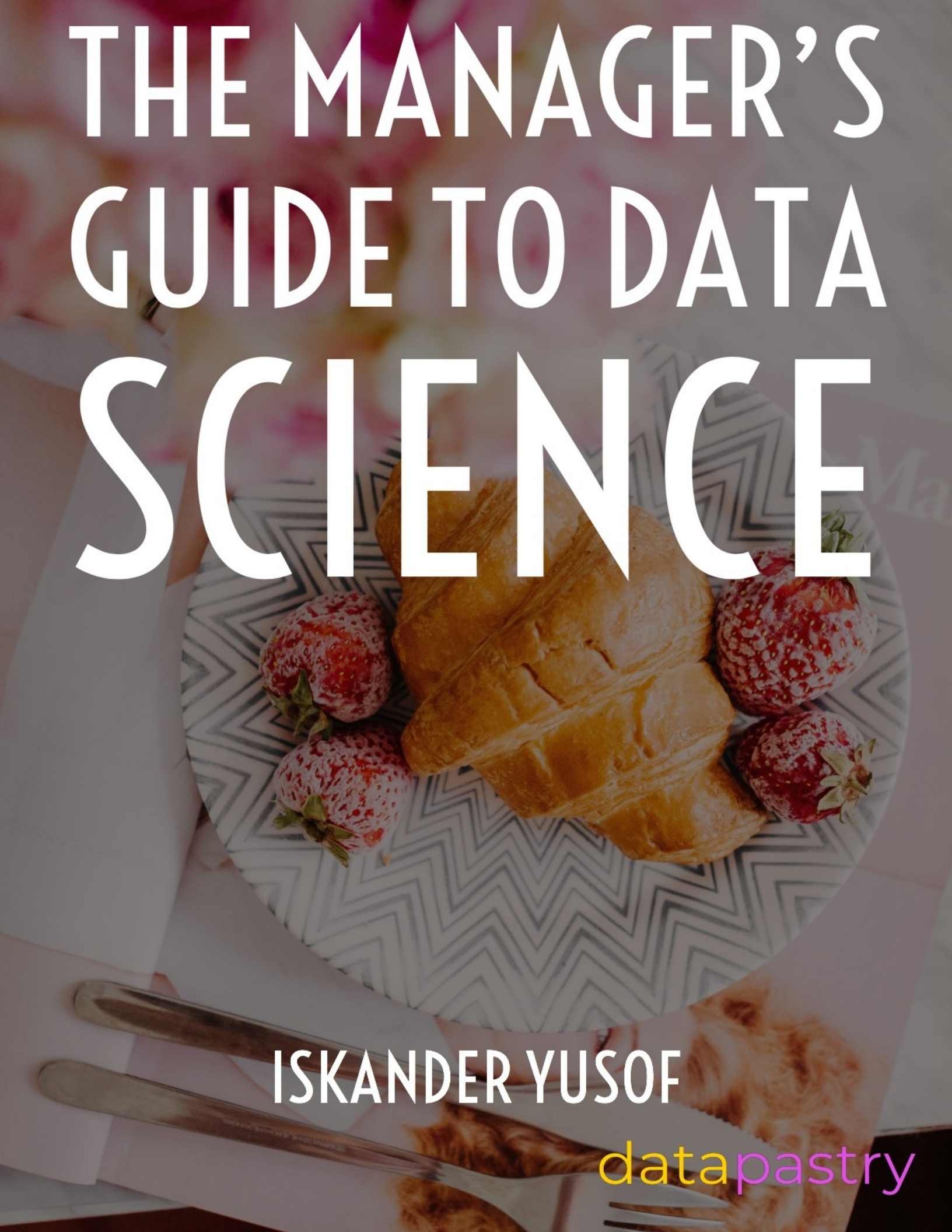
THE MANAGER'S GUIDE TO DATA SCIENCE



ISKANDER YUSOF

datapastry

Table Of Contents

You don’t need a data scientist

I have a confession to make. I hardly ever do data science.

This might sound like a bizarre claim, seeing as almost every job I’ve ever had has the words “Data Scientist” somewhere in its title.

(Aside: “What exactly do I mean by “data science work”?

Well, the term is somewhat ill-defined but I usually use it to mean “something to do with machine learning”. This might include a healthy dose of data cleaning and feature engineering work in addition to model building itself.)

So what’s going on? Am I getting paid a lot of money to watch YouTube videos of kittens all day long? No, of course not! I only spend 60% of my day on YouTube and most of the videos I watch are about other animals.

But let’s be serious for a few seconds.

As a Principal/Lead/Generally Quite Awesome Data Scientist, I receive a lot of requests for help with data science. Typically the conversation goes something like this:

Me: “HELLO I’M ISKANDER HOW MAY I HELP YOU!!!”

Hiring manager: “Er, hello there. Can you turn your mike down a bit?”

Me: “Oh, sorry. How may I help you?”

HM: “No worries. So we need someone who knows a lot about machine learning, and your background and experience sound like they’d be a great fit.”

Me: "Thanks, that's nice to hear. Can you tell me a bit about your company?"

HM: "So we sell dragon food online. We've been in business for three years now and have been growing rapidly. Currently we sell tens of thousands of items a day."

Me: "Sounds promising, I've actually been thinking of buying a dragon myself and I can see how this would be a good business. What would you like me to help with?"

HM: "Well, we've got a lot of data on our customers but have never really had the time to look into any of it properly. What we'd like you to do is look into it deeply and tell us what you can find."

Me: "Cool. Did you have any specific use cases in mind, or would you like me to help you define these as well?"

HM: "We do have a couple of things in mind…"

At this point I listen closely. I know that the first thing the HM mentions is likely to also be the highest-priority item for them, and hence the reason for the call.

HM (continues): "The most profitable area of business for us isn't actually dragon food – it's the dragon training service we provide our customers.

"As you know, dragons are very much *en vogue* now, but they have the unfortunate tendency to eat their owners. So we help customers train their dragons and reduce the likelihood of their getting eaten to less than 5%."

Me: "Ah, I understand. So you want me to help you upsell customers to the dragon training service."

HM: "Exactly! I can see why they call you a genius."

Me: "Stop, you're killing me. Can we talk about a few specifics? It sounds like you want me to figure out which customers are likely to purchase your dragon training service.

"Based on this, you'd be able to run marketing campaigns to acquire more customers like this, or even personalise your website so that ***DRAGON TRAINING SERVICE!!!*** pops up at exactly the right time for the right customer (unintrusively, of course)."

HM: "Yes!"

Me: "Right, so a good way to start might be to look at your existing customers and seeing which ones have bought the dragon training service, and then we can build a predictive model from there.

"Can I ask how many customers typically purchase the service at the moment?"

HM: "That's actually tricky to answer. You see, the sale doesn't actually happen online. Because it's quite an expensive service, we feel it's better to have one of our dragon trainers make the sale in person.

"So the sale itself is entered into our CRM, but often there's quite a large delay between the sale actually being made and it actually getting logged into our system. The CRM itself is a legacy homegrown system and nobody likes to use it."

Me: "I see. But it sounds like you should at least have a rough idea of sales figures, even if they're not exactly up-to-date?"

HM: "Sure, it's usually somewhere between 5 and 30 sales a month, depending on what campaigns are running."

Me: "OK. That's quite a small number of sales, and I can tell you straight away that it might be tricky to train a useful machine learning model on such a small number of positives.

"Has anyone looked at the profiles of existing purchasers manually, just to get a general feel for what kinds of buyers you currently have? Perhaps just by talking to them if nothing else?"

HM: "No, as I said nobody has had time to look at the data closely. That's why we need a data scientist to really look into this more deeply."

Me: "Ah, OK. Well, perhaps there's another way we can look at this. Presumably the website functions as a lead generator of sorts, and then you have a dragon trainer who takes the lead and tries to make the sale."

HM: "That's pretty much it, yes."

Me: "Right, it's not ideal but we could view this as optimising two separate pieces. One is to maximise the number of leads generated by the website, and the other is to maximise the probability of a lead converting into a sale.

"Do you know how many leads you're currently generating from the website?"

HM: "Yes, it's a few thousand per month."

Me: "Really? That sounds like a very low conversion rate, so it sounds like you either have a problem with the quality of leads or the follow-up needs to be improved?"

HM: "That is true, but the number of leads is one of the top KPIs for the website, so we're trying to keep it as high as possible."

Me: "Wouldn't it make more sense to have the eventual number of

dragon training sales be the KPI? You have to be careful with simply optimising the number of leads as you might end up with a lot of badly-converting leads."

HM: "The sales data is in the CRM so we can't track it as easily."

Me: "Hmm, so it sounds like the first step is to fix this KPI as we need to make sure we're optimising for something sensible. You mentioned that you have a CRM and presumably you also track visitor behaviour online?"

HM: "Yes, we use Google Analytics."

Me: "Right, so in order to help you understand your sales, a data analyst – because this is what I feel you really need before you hire a data scientist – will ideally have all the data about each customer in one place.

"This is surprisingly difficult to do as it involves tracking customers as they proceed from the online world to the offline, and often this can't be done retroactively."

HM: "OK…"

Me: "So as a first step, you might want to consider migrating to a better CRM that allows you to easily report your sales numbers and will also help to facilitate this online-to-offline tracking.

"Migrations are always tricky and I'm usually hesitant to recommend them, but clearly it's hurting your business if you're not even able to do reporting from your CRM. You'll have to weigh the cost versus benefit of this of course."

HM: "Thanks a lot, this all makes sense. So if we do proceed with this plan, would you be able to help us with the migration?"

Me: "I've done a few of these things before, but I'd suggest hiring a specialist instead as it's obviously important for you and you don't want to pay for data science skills that you're not really using."

HM: "That's fine, we really need a data scientist as there's a lot of scope for AI here as well. I didn't tell you about the other work that needs to be done, which involves consolidating all our data sources…"

Me: "Erm… all right then. I've always wanted to learn more about CRMs."

In reality, of course, I'd turn down any task where I didn't think I was able to do a good job.

But weirdly, I've ended up "qualified" in a whole range of areas that have absolutely nothing to do with data science. This is because I've found myself doing "anything and everything to do with data", and it's not always obvious what a specific role will entail until you actually get started.

If I had to describe my working life in four sentences, it would be "I solve problems via any means possible. Usually with data. And I'm not very good at counting."

I'm pretty happy with this as it makes me an easy hire in most situations, but the fact of the matter remains: I'm a data scientist who doesn't often do data science.

*(In the next two chapters, I'll talk about what data scientists **really** spend their time on. I'll also say more about when you **should** and **shouldn't** hire a data scientist.*

Subscribe to the DataPastry mailing list for more content like this. We hate spam so we only send stuff we think is great.

Want to discuss your own specific data problems? Book a free consultation with us – we're here to help!)

What data scientists actually do all day

(In the previous chapter I admitted that I rarely do data science. Scandalous!)

What data scientists actually do

Most data scientists are:

a) not doing data science

or

b) not doing anything that generates actual business value.

I love doing things backwards, so let's talk about point b) first.

Producing business value

The best way to ensure your data scientists produce business value is to hold them accountable for their work.

Have a frank conversation with them about what metrics they intend to optimise, how these metrics are relevant to the business and how they intend to go about improving said metrics.

But make sure your data scientists have the remit to effect change throughout the business and **don't have them sitting in silos on their own**.

If they come up with something good, your job is to ensure they're able to put their work into production or change the relevant business

processes.

You won't be on your own: a *good* data scientist should be able to arm you with metrics to help you convince other stakeholders that what they're doing is worth following through with.

A *great* data scientist can go even further and will be able to put together a realistic roadmap for productionising their work.

You should also avoid focusing too much on academic credentials when looking for a data scientist.

Yes, there are data science problems that are essentially research questions, but these are in the extreme minority.

And one of the hallmarks of research problems is that they're not always solvable, which means there's a high chance that a research-focused data scientist will produce absolutely nothing.

I've worked in a few data science teams that felt like an attempt to replicate academia in industry, with a bigger focus on publishing papers than producing anything useful for the business.

Admittedly this can be great fun and it's important for data scientists to stay in touch with cutting-edge research, but unless you're Google or Microsoft you probably can't afford to have this be the sole focus of any data scientist you hire.

Data scientists who don't do data science

You'll find many "data scientists" who are actually doing the work of a data analyst, a data engineer or a software engineer (oh dear). In this case, they're generating value for you but they're not doing "data science".

But what's in a name? Does the job title matter as long as they're useful?

Well, the problem is that they usually won't be anywhere near as good as an actual specialist in the field, so your data reporting/engineering/architecture will now be half-baked compared to what you'd be getting if you'd hired the right person in the first place.

Also, your "data scientist" likely won't want to work for you for much longer if they're actually interested in doing real data science – and most data scientists are.

There's a fair chance that they're quite junior (you didn't just hire them because they were cheap, did you?) and simply using your company as a stepping stone for a real data science job, most of which require at least 1-2 years of industry experience.

Of course, not having an employee stick around for 300 years isn't necessarily a problem, but you'd probably prefer someone who would actually enjoy the job and stay with you for a bit longer.

Sometimes it does make sense to hire a data scientist to be your "everything data" guy even if only 10% of what they do is data science. This is especially true in startups where it's normal for employees to wear multiple hats.

It's probably fair to say that if you *do* need an "everything data" guy, then a well-rounded data scientist is a pretty good bet. An experienced data scientist can often do the occasional data analytics or engineering task but the reverse is very difficult.

Think about your top data-related priorities and ask yourself where data science sits in the hierarchy: if your main requirements are reporting aggregate metrics and getting a better understanding of your data, then a good data analyst should be just fine.

Don't kid yourself about what you actually need, and make sure any data professional you hire knows what type of work to expect.

If data scientists aren't doing data science, what *are* they doing?

Wait, isn't this the same as the previous section? Well, kind of, but there are a few common problems I wanted to point out explicitly, namely:

- Your data exists in many different places and your data scientist will spend their entire working life getting it into one place and then doing basic reporting on the data.
- You don't have enough data to justify the use of machine learning, and a good software engineer applying simple heuristics would do the job at least as well.
- Most of the value in your data can easily be extracted without needing to hire a ludicrously-overqualified data scientist to apply the latest and greatest techniques.

None of this is meant to imply that data scientists shouldn't have any knowledge of areas outside machine learning.

In fact, the opposite is true: ideally a data scientist should at least have a basic understanding of sister areas such as data warehousing and software engineering.

And if your data scientist is going to be building a predictive model, it usually makes sense for them to perform the relevant data munging and cleaning tasks themselves.

The crucial distinction is that when a data scientist does non-data

science work, this should be simply be support for a real data science task. If the task as a whole doesn't involve data science at all, then you've given the job to the wrong person.

Data science: it's not magic

Why are so many data scientists hired if they're not really needed? My opinion is that this is due to two interrelated reasons: one is fear of missing out (FOMO), and the other is simply ignorance of what a data scientist can and can't do.

(Aside: To a large extent, unnecessary hiring isn't unique to data science.

Especially in large companies, I've often encountered employees where I've felt "Hmm, this person's job is not really necessary. Almost everything they do is busywork." But hopefully you want to run a more efficient operation than that.)

Let's talk about FOMO first.

Unless you've been living under a rock the size of a planet, you cannot fail to have heard the hype around "the AI revolution", "automation", "the robots are coming to take our jobs" and other bombastic claims made by people who don't actually work in the field.

So naturally you might feel like unless you embrace the AI revolution yourself, your company will go the way of the dinosaur, and you definitely don't want that. Who can help you get on the bandwagon? How about an **AI expert**!

And what are **AI experts** usually called? Well, "data scientists" for one…

Hopefully you'll agree that FOMO is not a good thing to base a company's hiring decisions on. So let's now talk about the second, more

concrete factor.

What can data scientists actually bring to the table? Very broadly, data scientists extract value from data. Slightly more precisely, data scientists find patterns in data and then exploit these patterns to predict things you care about based on past events.

To find these patterns, the data scientist will need data (duh!). What kind of data will they need?

Well, imagine that you're a homeowner and you want to predict the sale price of your property, which is just another way of saying you want to value your property. Naturally, this would be based on a range of different factors including the property's location, size, condition and so on.

You'd then want to relate these factors to the property price. The more historical data you have connecting a property's price to its location, size and condition, the easier it will be to value your own property.

Data scientists basically do the same thing as you do when you value a property, except in an automated fashion and on a larger scale.

And just like you, they're most effective when they have (or are able to generate) lots of high-quality data that is strongly related to the thing they're trying to predict. **Without good data, a data scientist is less valuable than an inedible pastry.**

You might hope that a data scientist will be able to use machine learning to find patterns that a human would never be able to. In a way this is true, because they can write code to look at far more data than a human could.

But don't expect to see something magical. Typically data scientists

will find patterns that are fairly intuitive, except that they'll be able to tell you *exactly* how a property's price relates to its location and size and when this relationship doesn't hold up.

It's also common for data scientists to *invalidate* a lot of prior assumptions and conventional wisdom in a company. Being truly data-driven means you'll listen to them when they tell you that your intuition about what kinds of customers are most likely to churn was completely wrong.

What about getting "insights" from data? Sometimes data scientists will do clever things to generate pretty pictures so they can say things like "oh look! Most of our customers fall into one of these seven big clusters."

What do they do with these clusters? Usually not very much, but for a brief time it will look like they've done something important. (This kind of clustering *can* be useful, but all too often it is used simply to generate attractive visualisations without much tangible value.)

If you only want to understand your data better, hire a good data analyst.

They'll know how to interrogate your data using SQL and other tools to pull out the numbers you care about, as well as how to present these numbers in an easy-to-digest format for other stakeholders. They can even generate pretty pictures!

And then you can hire a data scientist once you actually need to predict something.

Personally, I love it when I start with a client and they already have a data analyst working there, because I can then be confident that someone actually understands the data and has made an effort to ensure it's usable.

Working with hitherto-untouched data is like eating a random object that a stranger has handed to you: there's a small chance that it'll be a delicious pastry, but it's a lot more likely that you'll get food poisoning.

There's plenty of value in data. Most of it can be extracted without a data scientist.

*(In the next chapter, I'll say more about when you **should** and **shouldn't** hire a data scientist.*

Subscribe to the DataPastry mailing list for more content like this. We hate spam so we only send stuff we think is great.

Want to discuss your own specific data problems? Book a free consultation with us – we're here to help!)

Are you ready for data science?

(In Chapters 1 and 2, I explained why data scientists are usually a waste of time and money.)

When *shouldn't* you hire a data scientist?

Let's highlight a couple of common data problems faced by companies for which a data scientist shouldn't be necessary.

Problem: You have a reasonably clean customer database but don't really understand your customers. Basic questions like "where do our customers live?" are hard to answer.

What you actually need: A data analyst who specialises in reporting and visualisation.

Part of this will involve creating a consistent set of metrics across the company that all stakeholders agree on, so you'll need someone who has a good understanding of business as well as raw analytical skills.

Problem: Your data is easily-accessible but is very messy. The prospect of having to actually work with the data fills you with dread.

What you actually need: This depends on *why* you want to clean the data.

Cleaning isn't a worthy goal in and of itself and you won't be able to prioritise correctly unless you know what you want to do with the data.

If the main goal is to report aggregate metrics, then hire a data analyst who has sufficient programming skills to clean the data and is happy to

maintain documentation for your data.

If the goal is predictive modelling, then yes, consider hiring a data scientist to do this work.

Problem: Your company is in an industry that isn't particularly technically sophisticated.

You do have many data sources and think there would be a lot of value in getting all this data into a data warehouse as a single source of truth, as well as to give a complete view of each customer and your business as a whole.

What you actually need: This is a complex task for which you'll most likely want to make multiple hires.

At the very least you'll need a data architect to make technical decisions regarding how to get all data sources into one place, as well as a data governance manager to ensure a consistent level of quality is maintained across all sources.

Again, simply getting all data into a data warehouse isn't by itself a sensible goal, so consider making hiring decisions based on the actual problems you want to solve using your data warehouse.

Consider developing an MVP first. You might initially be able to extract a lot of value by just getting two sources of customer data into a single place, e.g. by merging website analytics data with marketing campaign data.

You'll also want the "consumer" of the data sources to be happy with the outcome, which means you might need a data analyst or data scientist to specify their use cases to the data architect.

Problem: You're generating huge quantities of sensor data every

second and need someone to architect a system to ingest and process this data.

What you actually need: A data engineer. Presumably you have a reason for wanting this data (see the theme here?), so make sure the work the data engineer does is in service of this goal.

If you need this data for predictive analytics, you might also want the advice of a data scientist to help inform some of the data engineer's decisions.

Unfortunately there's often no connection between a job's title and its actual responsibilities – which is part of the reason this book exists!

For this reason, "data analyst" is the answer to a lot of these questions, but the specific flavour of data analyst you should hire varies a lot.

There's a big difference between a data analyst who can produce visualisations for C-level executives using Tableau and a data analyst who can knock your data into good shape.

You might prefer to use more specific titles such as "data quality analyst" when looking for a new hire. Sometimes you'll get lucky and find all the skills you might conceivably need in a single person, but if not, you should identify your top priorities and aim to hire for those.

When *should* you hire a data scientist?

Only consider hiring a data scientist if you're sure that your current team won't be able to adequately address your problem.

If you have a task that can be expressed as a "prediction problem",

then you'll probably need a data scientist.

What's a "prediction problem"? This means you'd like to predict something based on historical data. For example, you might want to:

- Predict what customers are likely to buy on a website given the purchasing behaviour of similar customers.
- Figure out what's most likely to happen in an election based on the outcome of previous elections.
- Determine how to spend your marketing budget. A data scientist can help you do this by predicting your expected return on investment based on the amount of money you put into each marketing channel.

It won't always be obvious if your task is a prediction problem.

If you have a business process that involves manually typing up handwritten forms and pushing the entries into a database, then a data scientist could help you automate this process. They would write software that "predicts" (recognises) what has been written in your forms based on historical handwriting data.

If you're not sure what category your data problem falls into, just book a free consultation with us and we'll talk it over.

MVP data science

The concept of an MVP (Minimum Viable Product) is well-known in the startup and technology communities.

Very simply, the idea is to put together the smallest possible product that does something useful, in order to test whether or not the basic

idea for the product resonates with the market. The MVP is then improved iteratively based on the feedback from customers.

You can do "MVP data science" in a similar way.

What's the bare minimum you need for a data scientist to produce something useful? Here's a checklist showing what is *essential* and what is *nice to have* before you call upon the services of a data scientist:

Essential

- You have a "prediction problem". Think carefully about whether the task you have can be described as such.

- You have some historical data that could plausibly be used to make these predictions.

  Remember: **don't expect magic**. You're not going to find a data scientist who can accurately predict a stock price based solely on its historical movements.

Nice to have

- Ideally you should know what success looks like. You should have a rough idea of how much business value there is in solving your prediction problem.

  If it potentially means £10m in reduced marketing expenditure, that's great.

  But if it simply means saving 30 minutes a day, that might only be worth £10,000 and it'd be hard to justify hiring a data scientist.

- Your data is reasonably easy to access.

  As long as they've been granted access, a good data scientist

should generally be fine with pulling data from different sources. But things get a lot harder if the data owners are unknown, hard to get hold of, or unconvinced of the merits of your problem.

- You know roughly what data you have (e.g. in the form of a semi-maintained data dictionary, or at least someone who the data scientist can call upon when they have questions about the data).

- You have some understanding of high-level metrics related to your prediction problem.

  For example, if you want to reduce churn for your SaaS business, you should ideally know the current churn rate and how this breaks down across different types of customers.

- You've taken "obvious" actions to solve your problem that don't necessarily involve a data scientist.

  Using the churn example again, there are many simple things you can do to reduce attrition that don't involve building a predictive model of potential churners.

Metrics and business value

Let's close with a topic that everyone pays lip service to but – in my experience – is very rarely done well.

That is, what metrics should the data team be optimising for and what business value do you expect them to generate?

Most "obvious" KPIs such as revenue or retention will be too broad to optimise for directly. Almost any individual project will struggle to demonstrate an immediate impact on these KPIs, but clearly this doesn't mean that all such projects are useless.

Instead, try to come up with metrics that are fairly narrowly-defined and **easy to measure, but plausibly correlated with business value**.

Expect these metrics to evolve our time as your data team matures. Eventually, your data scientists should be able to directly demonstrate what value they have added to the business.

You can help your data team by making it clear what metrics actually matter for your business and regularly reviewing the data team's internal metrics to ensure they are aligned with your business goals.

Also, it's fine if your team needs to define a new success metric for a specific project, as long as they are able to justify why it is the right metric to use.

Sometimes your data scientists *will* be able to actually quantify the relationship between the metrics they're optimising for and your global business metrics, but often this will be more art than science.

With enough small improvements to your data team's metrics, you should start to see the effects compounding over time and the needle moving on your main business KPIs.

Don't underestimate the difficulty of establishing, maintaining and tracking a good set of metrics for your data team. Correctly assessing the impact of any given change requires a lot of statistical rigour and maturity across multiple business units.

And most claims I've seen along the lines of "my work led to millions of dollars more in revenue" just don't hold up once you examine them closely.

Hey, nobody said data science was *easy*.

(Subscribe to the DataPastry mailing list for more content like this. We hate

spam so we only send stuff we think is great.

Want to discuss your own specific data problems? Book a free consultation with us – we're here to help!)

How to hire a data scientist

Software engineers often complain about the endless technical tests they have to overcome in order to get hired.

For programmers who are relatively green it's perhaps understandable that they'd be tested heavily, but it's somewhat galling if you have 10 years' experience in a domain and get asked to answer basic programming and algorithms problems. You don't ask an accountant to solve simple arithmetic problems before you allow them to file your taxes.

Anyway, the reason for all the testing is simple. Most programmers are rubbish at programming, so you have no choice but to test them extensively if you don't want to inadvertently hire a smooth-talking knucklehead.

Unfortunately, the same is true of data science. Without testing a prospective hire's ability to do data science, you'll have no way of knowing whether they are actually able to do the job.

Most likely you'll optimise for the ability to *talk* about data science rather than actually *do* data science, which is fine if all you want is a great communicator, but not so good if you need results.

When I'm interviewing prospective hires, I'm painfully aware that I too can be fooled by someone who uses roughly the right terminology and seems to follow the right kinds of processes.

I always find it interesting to see how candidates respond as I move from generalist "talk me through how you solved that problem"-type questions to more specific technical questions. Often candidates will be able to improvise their way through the generalist questions but come

unstuck as soon as they're asked something very specific.

In general, people are easily fooled by confident, eloquent speakers. It's easy for a charlatan to make it sound like they know what they're talking about when they really don't. On the flip side, a slightly diffident data scientist might turn out to be exactly the right person for the job.

Testing data science candidates

When devising a data science test, you can forget about using brainteasers, personality tests or clever questions the interviewer thought up in order to demonstrate their own intelligence. Doing well in such tests is a poor predictor of on-the-job performance.

The best way to test a prospective hire is to have them perform work that is as close as possible to the actual role. This is often called a *work sample test*.

The reason for its effectiveness should be obvious: what better way to evaluate a candidate's suitability for a job than to have them do the job itself?

Coming up with a good work sample test is not easy, however. There are a couple of things you'll need to take into account:

- You'll need to distill the main elements of the job into a test that is concise enough to be doable within a short time frame.

  We'd suggest focusing on testing things that *can't* be learnt quickly on the job.

- You should devise an objective scoring rubric to grade each candidate.

- You should standardise your test to ensure every candidate is treated in the exact same way. Ideally you should calibrate your scoring using data scientists you've worked with before.

  As a bare minimum, at least one person should successfully complete the test before you give it to a candidate.

- Good data science tests typically involve a data task that encompasses exploratory analysis, some cleaning/feature engineering and building a model.

  You can also incorporate a system design problem into your task (e.g. "how would you productionise this model to serve predictions to hundreds of users per second?") if it's relevant to the job.

- Try to ensure your test is substantive enough to allow you to gauge the applicant's code quality. Clean, well-structured code is something many data scientists struggle to achieve.

*(**Shameless plug:** We're putting together a free work sample test and scoring rubric that you can use as a basis for your own test.*

Subscribe to the DataPastry mailing list if you'd like access and we'll send it over when it's ready.)

We'd suggest allowing candidates to do the test at home instead of insisting that it be conducted in person. This will save time and money for both parties.

If the candidate is successful at the work sample stage, you can invite them over to your office to present their results.

On the subject of time, try to ensure the test can be completed in a maximum of 6 hours by a good candidate. If possible your test should be a lot shorter than this; between 2-4 hours would be ideal. Don't forget

that your candidate will most likely have other tests to do as well.

Senior candidates will also often have families, so they'll be very constrained for time outside normal working hours. (I *don't* have a family, but I also have very little free time. Those pastries don't just eat themselves.)

Should you pay for the candidate to do the test? This is a tough one. It's not the norm in industry for candidates to get paid to do tests, but I wouldn't be surprised if this changes given current trends.

Morally I'd say you *should* pay for a candidate's time, but I've never actually been paid to do a test before. For me, the benefit of securing a good job is worth the small investment of time. I also enjoy working on interesting, challenging tests, so do try to make sure your test isn't as dull as dishwater.

Unless, of course, the job itself is boring, in which case it's probably for the best if your candidate knows what to expect.

Encourage candidates to use the same resources that they'd use on the job and keep an open channel for any questions they might have. You wouldn't refuse to answer questions in a real job situation, so it doesn't make sense to create arbitrary restrictions during your testing phase.

Make sure each candidate knows how long it will take for their submission to be graded and stick to this timeline. Speaking of which…

Being nice

Try not to annoy prospective hires by demanding things that aren't really necessary. From a human perspective, I shouldn't need to explain further the importance of being nice.

But even if you're a callous employer with a block of ice where your

heart is meant to be, it's still in your interest to be nice. Demand massively outstrips supply for good data scientists – within reason, we don't have difficulty getting the kinds of packages we want.

Here are a couple of things that *should* be no-brainers, but very rarely are:

- Right from the start, clearly communicate to the candidate your hiring process from CV review through to offer.

  All you need for this is a simple timeline indicating each stage of the process and who the candidate will be talking to. **Ensure you stick to your timeline.**

- Realistically HR will get involved with your hiring process at some point. Again, set expectations for candidates so they won't be surprised if HR throws up a few roadblocks (we've all been there!).

- Don't use a work sample test to get candidates to do real work for free. The data science community is fairly small and nobody will want to work with you again if you do this.

If your hiring process is very unpleasant, many good candidates will just walk away instead of wasting time and energy with your company. There are plenty of other data science jobs out there.

And if you're happy with a candidate, **make a serious offer as fast as you possibly can.**

I can't emphasise this enough. I don't want to boast (he lied), but I've never been turned down for a job once I've spoken to a real person. I also don't like to waste time, so I simply take the first good offer I receive.

Generally I do this out of a (possibly irrational) fear that I'll lose the

offer if I hang around for too long. I'm pretty sure a lot of data scientists are the same way, so you can save yourself a bunch of time and money just by making hiring decisions more quickly than your competitors.

Don't copy Google

It's quite common for companies to cite Google or some other hugely successful tech company as inspiration for their hiring practices. After all, Google is known for having outstanding employees, so surely it would make sense to learn from how they hire?

Unfortunately, what works for Google almost certainly won't work for you. Because, well, you're not Google.

All hiring processes are fundamentally a tradeoff between *false positives* and *false negatives*. A false positive is an accidental hire of a bad candidate. A false negative is a rejection of a candidate you should have hired.

A super-high interviewing bar means that you'll have very few false positives but also a lot of false negatives. You'll reject a lot of great candidates, but the few people you do hire will likely be excellent.

On the other hand, a low interviewing bar means that you'll accept a lot of dodgy candidates… and then you'll either have to put up with incompetent employees or fire people regularly. Also not good.

Google-style interviews are intended to minimise false positives: they really, really don't want to inadvertently hire a dud candidate. For this reason, they have a very low acceptance rate and reject a lot of people they "should" have hired.

So why not do the same as Google? There are two reasons for this:

- Everyone wants to work at Google. They have a huge pool of

candidates to draw from so even if they reject 99% of them, they'll still be left with a sizeable number of people.

- Google pays for the privilege of being so selective. Do you pay as well as Google? Is your company as prestigious as Google? If not, perhaps you should reconsider being as selective as Google.

Of course, this doesn't mean you should be OK with false positives and accept mediocre candidates. It just means you can't afford to have an interview process where you reject huge numbers of qualified people. Your hiring process needs to minimise both false positives and false negatives.

Yes, that's right. **You need to be better at hiring than Google.** It's actually not that hard, because Google is optimising for something different from you.

Writing a job description

A good job description for a data science role is not much different from any other technical role. Things you'll need to cover:

- A description of your company and team/department
- An overview of responsibilities for the role
- Hard requirements for the applicant
- Nice-to-haves
- Salary, benefits and other perks

Employers have the unfortunate tendency to list far more skills than they need as "required" for data science roles.

Almost all the jobs I've worked in over the last 3 years have specified deep learning as a hard requirement. For the most part, the subject never comes up again during either the interviews or the jobs

themselves. (Interestingly, few job descriptions ever mention “data munging skills” as a hard requirement…)

This is so common that I view it as a minor annoyance rather than a dealbreaker, but it doesn’t give a particularly good first impression for a company.

And not every applicant will realise that your “hard requirements” aren’t actually requirements. You’ll scare off both good and bad candidates by writing an inaccurate job description, so just honestly state what the job involves and don’t exaggerate.

If you really do think writing JavaScript might be part of a candidate’s responsibilities in the future, put this down as a nice-to-have and use it as a tiebreaker in an “all other things are equal” situation.

It’s really nice if you say what a “day in the life of a data scientist” would be like at your company. An experienced candidate should ask about this anyway, of course; there’s very little consistency between data science operations in different companies.

Without this knowledge, accepting a new role can feel a bit like walking across a motorway with your eyes closed in the hope of finding a pastry on the other side.

Oh, remember when we said we’re putting together a free work sample test and scoring rubric just for you?

We’ll also create a sample job description for you to use as a template. Aren’t we nice?

Subscribe below to our mailing list if you’d like us to send you these materials when they’re ready. As always, you’re welcome to customise our materials as much as you like.

(In the next chapter, I'll say more about the specific skills you should test for when hiring a data scientist.

Subscribe to the DataPastry mailing list for more content like this. We hate spam so we only send stuff we think is great.

Want to discuss your own data problems? Book a free consultation with us – we're here to help!)

What should a data scientist know?

(In the previous chapter, I explained how to test data scientists so you don't accidentally hire a baboon.)

Skills and experience

Data scientists are the ultimate jacks of all trades.

You don't need someone who's an expert in everything but it's important that they understand how all the pieces fit together. **This is not a good field for extreme specialists**.

We'd advise customising the skills you test based on the specific role for which you're hiring. You probably don't need a data scientist with a deep knowledge of convolutional neural networks if they're mostly going to be working on credit scoring.

Still, there are more similarities than differences in data science roles. Below you can find the types of skills and experience we've found to be especially relevant.

This is not intended to be an exhaustive list, nor should you expect to find a data scientist who ticks all these boxes. Again, we'd recommend tailoring your questions based on what is genuinely important for your specific role.

- **Machine Learning**
    - Knowledge of supervised learning (regression, classification) and unsupervised learning (clustering, dimensionality

reduction) techniques
        - Linear Regression, Naive Bayes, Decision Trees, Logistic Regression, SVM, k-NN, k-means, Random Forests, Gradient Boosted Trees, PCA, Gaussian Mixture Models…
    - Being able to explain in depth how at least one ML model works
    - Exploratory data analysis: what's their approach?
    - Building a model: what's their approach?
    - Feature selection/extraction
    - Training, test, validation sets
    - Cross-validation and bootstrapping
    - How to assess the accuracy of an ML model
    - Online and offline evaluation of ML models
    - Can they relate ML-specific metrics to business metrics?
    - Curse of dimensionality
    - Overfitting
    - Handling missing data
    - Handling lots of observations/features
    - Handling outliers
    - Handling data quality issues: what's their approach?
    - Handling big data (too big to fit in memory): what's their approach?
    - Data wrangling/munging: how happy are they to work with messy data and get it into a suitable shape for analytics work?
- **Tools**
    - Comfortable with data science libraries such as numpy, pandas, sklearn
    - Experience with cloud environments like AWS or GCP
    - Experience with ML as a service (e.g. SageMaker, H2O or Cloud ML)
    - Experience with relational databases and SQL
    - Experience with NoSQL data stores like Elasticsearch or Redis
- **Statistics**

    - Hypothesis testing, confidence intervals, probability distributions (Gaussian, binomial, Poisson…), transforming distributions, basic probability theory, basic Bayesian statistics, correlation
    - Measures of central tendency (mean, median, mode…) and dispersion (standard deviation, interquartile range…)
    - Understanding of possible sources of bias
    - How to run an A/B test
- **Algorithms & Data Structures**
    - Basic algorithms and data structures (sorting, searching, arrays, linked lists, stacks, queues, trees etc)
    - Algorithmic complexity
    - Recursion
    - Dynamic programming
    - Basic computer architecture (understanding the memory hierarchy, roughly what CPUs do etc)
- **Maths**
    - Linear algebra, multivariable calculus, optimisation
- **Software Engineering**
    - Ability to write clean, idiomatic Python
    - Knowledge of another programming language (ideally a widely-used language very different from Python, such as Java or Scala)
    - Tests (from "small" unit tests through to "large" integration tests)
    - Version control
    - Continuous integration and deployment
    - An understanding of how to write reasonably efficient code
    - Model deployment: how to get a machine learning model into production
    - How much do they know about the kinds of issues you can run into with a productionised ML model? (e.g. drift, model updates, management of multiple models, monitoring accuracy metrics)

- **Soft Skills**
  - Verbal and written communication skills: how good are they at communicating technical subjects to a lay audience? **Can they communicate the value of what they do?**
  - How much emphasis do they place on the tools and techniques they use vs. the business value generated from their work?
  - **Process and management:**
    - Experience leading teams/projects
    - Experience working in an agile development environment
    - Communication/management style
    - Experience working with different stakeholders when delivering a data science project
  - **Team structure:**
    - Have they ever worked in a cross-functional team or have they always worked in pure data science teams?
    - What has their relationship been with other data scientists, data engineers, data analysts or software engineers?
    - Do they normally work solo or in close collaboration with others?
- **Other**
  - Specific industry experience, e.g. fintech or retail
  - Specific knowledge of sister fields like NLP or computer vision
  - Specific knowledge of "advanced" ML and statistical techniques such as deep learning or Bayesian inference

The whole enchilada

These are steps we'd suggest using when hiring a data scientist. You don't have to stick to this precisely; the only parts we'd view as being essential are the work sample test and ensuing discussion.

1. **CV review**

This is normally pretty easy and shouldn't take more than a few minutes per CV once you know what to look for. Check the CV against your list of job requirements (which should also be in your job description!).

If most of the requirements are satisfied, then you can move on to the next stage. Remember that the goal is to filter out obviously-poor candidates, not to identify great candidates at the very first stage.

2. **Phone screen**

You'll probably want a further screen to filter out dud candidates before taking the time to send them a work sample test.

The idea of the phone screen is simply to eliminate candidates who are obviously not up to scratch or won't be a good fit for some other reason. It's a timesaving measure for both you and the candidate.

We've found that a mix of questions about experience, a few specific technical questions and a short discussion of the role works well.

3. **Work sample test (ideally take-home)**

Hmm. I talked about this in the previous chapter.

TL; DR version: put together a concise test that's representative of the job itself. And let us know if you'd like an example of a test and scoring rubric to use as a template.

4. **Presentation and discussion of results**

No data science task would be complete without a presentation!

The goal here should be to test the candidate's communication skills and dive deeply into any questions you might have about the candidate's work.

It also makes it hard for candidates to pay other people to do the test for them.

5. **Technical interview (in person)**

You usually won't be able to put everything you want into a work sample test – at least, not if you want the test to be doable in under 10 years.

For this reason, you might want to conduct a separate technical interview to examine the applicant's knowledge of software testing, model deployment and other topics.

We don't consider this to be as reliable an indicator as the work sample test, but it can serve as a useful tiebreaker if you have two very similar candidates. It might also be possible to incorporate the technical interview into the discussion of their work sample task.

For more senior candidates – especially those that will be leading projects – the in-person interview is also a good opportunity to see where their priorities lie. Do they optimise for generating business value or are they more interested in technical concerns?

A good way to assess this is to talk to the candidate about their past projects: do they emphasise the tools and algorithms they used? Or do they focus on the impact of their work?

Technical acumen doesn't always go hand in hand with having good business sense.

6. **Culture fit**

Most data scientists are lovely and personable like us, but you do get the occasional lunatic. So you might want to get an idea of how the candidate will gel with your other team members before making them an offer.

But make sure you have a clear idea of exactly what it is you mean by "culture fit" (make it part of your rubric!).

"Didn't fit the company culture" might just mean that the existing culture is a homogeneous old boys' club and you'd subconsciously like to keep it that way.

Keep evolving

Over time, you should be looking to improve your hiring practices so that you do a better job of selecting good candidates.

Realistically you can't treat this as a data science problem (!) as you won't have enough data points, but after making a hire it's worth keeping an eye on how closely their work output corresponds with what you determined from your scoring rubric.

Did you over/underestimate their ability in any particular area? If so, you'll want to figure out why and update your testing accordingly.

Unfortunately it's a lot harder to learn from the people you *didn't* hire as you'll have no way of knowing how they might have performed had you given them the job. (If you have an idea of how to solve this problem, let us know.)

But I'm not a data scientist!

For some reason lots of non-technical people believe they have the ability to hire good technical candidates.

They're fooling themselves, unfortunately. The *only* time I've seen this work consistently is when a hiring manager pays over the odds for data scientists with great portfolios.

I've been through many non-technical hiring processes as well as technical interviews. With the non-technical interviews, I usually have a strong sense that I could just as easily get the job after reading a few blog articles and saying a few buzzwords.

It also sets off a red flag in my mind: if the company doesn't have a serious hiring process, what kinds of jokers will I be working with? (The reverse is also true. A good hiring process gives me a lot of confidence in a company.)

So how can you identify a good data scientist if you yourself are not a data scientist?

Short answer: You can't. Your best bet is to hire someone with a great portfolio, but as mentioned this will likely cost you a small fortune because everyone else will want that person as well. It doesn't take a genius to figure out that someone who used to run data science at Netflix and Airbnb is likely to be a pretty good data scientist.

Long answer: You can't. At least not entirely on your own. You really need to find a way to get assistance from data scientists that you know and trust. If you're not in a position to do this, then your best bet is to get in touch with us and we'll happily play the role of "trusted data scientists".

It's been too long since the last shameless plug, so I'll conclude this chapter with a couple of paragraphs of marketing fluff.

If you do find yourself needing to hire a good data scientist – and it *is* hard! – then book a free consultation and we'll help you out.

We'll give you advice on hiring strategy (maybe you don't need a data scientist at all) and build a structured interview and testing process tailored to your specific needs.

And if that's not enough, we'll even supply you with pastries.

Last but not least, we're putting together a free *Hire a Data Scientist* package containing a sample work sample test (ha), objective scoring rubric and example job description.

You can use the *Hire a Data Scientist* package as a template for your own data science hiring needs. Just subscribe below for DataPastry updates and we'll send the package to you when it's ready.

(Subscribe to the DataPastry mailing list for more content like this. We hate spam so we only send stuff we think is great.

Want to discuss your own specific data problems? Book a free consultation with us – we're here to help!)

Get the most value from your data scientist

Most data scientists don't generate quantifiable business value. You can easily test this for yourself by asking a couple of data scientists about the projects they've worked on.

Ask about the impact of their work. You'll typically hear proud statements such as "I improved the F1 score by 30%" or "I achieved an area under the ROC curve of 0.89".

This is great! But now ask how exactly these data science-specific numbers relate to an impact on actual business metrics. This is where the conversation starts getting a little fuzzy.

Very few data scientists can honestly say that they're responsible for a 20% reduction in churn or a 15% improvement in user engagement. At best you might get a hand-wavy explanation of how their work relates to high-level business metrics but they'll struggle to quantify their impact in a statistically rigorous manner.

But hang on: aren't data scientists comfortable with numbers? Shouldn't they have a way to measure the impact of their work? Well, of course they *should*, but this isn't necessarily their fault.

If it doesn't even cross the mind of a data scientist to measure business value, then yes, they're at least partly responsible.

In most cases, though, they *will* be aware of the problem but they work in a company whose structure makes it impossible for them to generate (and measure) real business value.

And let's not be too harsh. Measuring and attributing business value correctly is not trivial! It shouldn't be a huge surprise that you can't hire a data scientist and magically get a 50% increase in profit.

The goal of this chapter is to show you how to actually get value from your data scientist. This is in everyone's interest. Data scientists like it if the work they're doing has real impact, and you yourself won't want to have expensive data scientists on your payroll who don't actually do anything useful.

Most of this chapter is written with the assumption that you've hired just one experienced data scientist. But the same advice applies to teams as well.

We're also assuming that the data scientist is your ~~minion~~ direct report.

Onboarding your data scientist

Getting the most out of your data scientist starts from the first day of their employment.

You should start by giving them some high-level context: tell them about the business domain(s) they're expected to help out in and roughly how you'd like to use your data.

Hopefully you'll already have discussed some of this during the interview stage!

Putting the data into data science

Obviously you'll need to tell your data scientist what data you actually have and how it can be accessed.

This sounds straightforward but often you won't have detailed knowledge of what exactly you have, the quality of the data will be

questionable and – especially in larger companies – you might even struggle to convince the data owners to grant you access.

The worst-case scenario is that you're not even in a position where a data scientist can help you, but of course you did read the other chapter so you won't have made that mistake…

Sometimes you'll have the right data but the data scientist will have to do some work to make it usable for their purposes. If you've hired the right person, they won't complain about having to do some non-data science tasks before they get to the fun stuff.

In truth, I often like doing basic data engineering tasks myself as I can then be confident that the data ends up in the shape I want (and if it doesn't, then I'll only have myself to blame). I'll also learn a bunch of useful stuff about the data itself in this process.

Yes to communication, no to silos

Making sure data scientists have easy access to people they'll need to speak to should be a no-brainer but happens surprisingly rarely.

Introduce your data scientist to the team members they'll likely need to work with on a daily basis and try to make sure they're seated in close proximity to these people.

If this isn't possible (e.g. you have a distributed team), at least ensure it's not a huge hassle for your data scientist to reach out to their main points of contact. You won't want your data scientist sitting on their own in a silo.

Also, don't assume that because data scientists are highly technical, they won't know how to talk to people in a nontechnical way. Unlike software engineers, we're not all social outcasts (some of us even *like*

people).

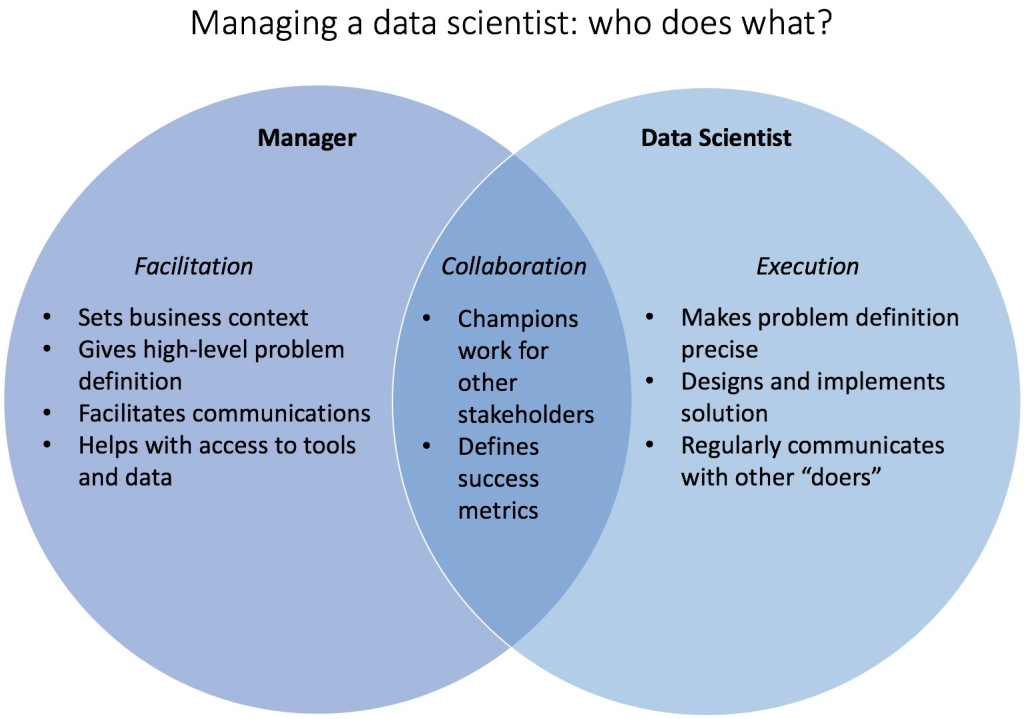
A data scientist should have no problem switching between a technical discussion with a programmer and a conversation with a senior executive who has a completely different mindset. This is why it's particularly important to test nontechnical communication skills during the interview.

Micromanagement

If you've hired only one data scientist, they *should* be experienced. You shouldn't need to dictate to them exactly what they need to be doing from day to day.

Give them the broad outlines of your problem and they should be able to work with you to make the problem precise and solvable using data science.

Let your data scientist take the lead on how to solve your problem, but help them out regarding the business context and by facilitating access to the resources they need (in terms of data, tools and people).



How to divide responsibilities with your data scientist.

Your job is to be a facilitator, not a dictator.

I can't count the number of times I've been told exactly *how* to solve a data science problem by a nontechnical person.

This is usually an incredibly bad idea: unless you yourself are a skilled data scientist, it's very unlikely that you'll have an idea of the optimal approach.

What's more, many data scientists won't have the confidence to push back against what you suggest, so you'll send them down the wrong path right from the word "go".

This is a waste of everyone's time and your money.

Of course, you should still put forward any ideas you have, but be very conscious of how they might be perceived. You should really prioritise describing the problem rather than suggesting possible solutions.

It'll help if you ask your data scientist to propose a solution *before* you offer some suggestions of your own. For the most part, you should defer to your data scientist regarding what they say needs to be done to solve your data problems.

If you don't trust them to do this, you're basically admitting that you made a bad hire!

Success metrics

Perhaps the most important part of onboarding is to agree on one or more metrics that your data scientist is responsible for improving.

This is a huge topic that deserves its own chapter, but the first step is simply to tell your data scientist what metrics matter to the business as a whole. This will help frame a discussion of which metrics they'll be able to improve.

Most likely you'll have an idea of which metrics you believe your data scientist can help with, but they might be able to move the needle on other metrics as well. So it's worth letting them know about all the metrics of interest to the business.

Be realistic: in many situations, your data scientist won't be able to optimise for metrics such as revenue or retention directly.

Instead, you'll want metrics that are fairly narrowly-defined and easy to measure, but plausibly correlated with business value.

Also, keep in mind that less is often more: it can be very difficult to move the needle, so ideally your data scientist won't be working to optimise more than one metric at a time.

A good outcome from onboarding is an initial MVP project for your data scientist to get stuck into. Discuss with them what knowledge and

resources they'll need to complete this project in terms of data, tools, people and metrics.

Once they've completed their first project, you'll both have a much better understanding of what they can contribute to your business and can take things from there.

(In the final two chapters, I'll talk about what your data scientist ***should*** *be doing from day to day in order to deliver value. I'll also explain how to successfully deliver your data science projects.*

Subscribe to the DataPastry mailing list for more content like this. We hate spam so we only send stuff we think is great.

Want to discuss your own specific data problems? Book a free consultation with us – we're here to help!)

What your data scientist should be doing

(In the previous chapter, I explained how to onboard your data scientist so they're in a position to deliver business value ASAP.)

What to expect from your data scientist

Data scientists excel at solving "prediction problems" where they have to predict something based on historical data.

For example, your data scientist might predict which customers in your subscription business are most likely to leave ("churn") within the next month.

In this example, three things will need to happen:

1. **Preparation:** Before your data scientist can do any real work, they'll need to get hold of your data and knock it into a shape where they can do some fancy predictive modelling.

   They'll also need tools to work with the data and ideally should have access to people who can tell them more about the data they'll be working with.

   Even the smartest data scientist won't be able to figure out what a field called `xgr671` means without any context.

2. **Prediction:** Your data scientist will need to build a *model* that predicts which customers are most likely to churn.

3. **Action:** Your company will take some sort of action based on the

model's predictions.

For example, you might proactively contact customers who are likely to churn and offer them incentives to stay with you.

You'll find lots of data scientists who are good at building models but relatively poor at knowing what to do with the results. But both the prediction and action steps are essential for you to get any value out of their work.

A good data scientist will have their eye on the whole picture instead of focusing solely on building good predictive models.

The preparation stage

If your data scientist has access to a large amount of clean, well-documented historical data on whatever it is you want to predict, then their job will be *much* easier and they'll be able to get to the next stage in no time at all.

Yeah, this hardly ever happens in the real world.

In reality, your data is probably a complete mess and the people responsible for data curation left 18 months ago to go backpacking around the world. Good for them!

So let's hope you hired a resourceful data scientist who is prepared to grind their way through this stage. Their task will be to figure out how to turn your horrible data into something they can use to build a predictive model.

Ultimately, what they'll look to build is a set of "features" (variables) that might be predictive of the thing you want to model. This is called *feature engineering*.

Taking the churn example as an example (ha), you might expect churn to be influenced by features such as the length of time a person has been a customer, how much they've used your service recently, how often they've left you positive feedback and so on.

There'll normally be many other factors where you have no idea if they have an impact on churn, such as seasonality or the age of the customer.

Your data scientist will prioritise the features that are "obviously" important but also try to include other features that may or may not matter, depending on how much time and effort it takes to do so.

Their magic machine learning model will be the final arbiter of which features are actually relevant, so it usually makes sense to feed it with as much data as possible.

Needless to say, **domain knowledge is extremely valuable** for engineering good, predictive features.

Your data scientist won't necessarily have any domain knowledge. I used to work in drug discovery where I initially had no idea what features might be predictive (as I was neither a medicinal chemist nor a drug dealer).

To help your data scientist come up with good features, make sure they have access to someone who can tell them more about the specific domain they're working in. It's very possible that this "someone" might be you!

When all is said and done, your data scientist will have expended a lot of blood, sweat and tears to construct a big table with one row for each customer showing whether they churned and their corresponding feature values.

Example features for predicting churn

Customer ID	Age	Average sessions per month	Average session duration (seconds)	Number of sessions in last month	Days spent in contract	Has previously churned	Monthly revenue ($)	Contacted us in last month	Churned
1	27	7	112	11	145	No	100	No	No
2	33	8	358	4	662	No	150	Yes	No
3	42	11	37	2	94	Yes	350	Yes	Yes
4	24	65	257	62	39	No	50	No	Yes
5	59	14	340	13	374	No	250	No	No

Features for each customer and whether or not they churned. In reality there'd be a lot more rows, a lot more columns and everyone would have a slightly different definition of "churn".

This is what they will use to build a machine learning model to predict churn. Hopefully their model will be awesome. But it's also possible that their model will be rubbish.

The prediction stage

In the churn example, you should expect that your data scientist will build a model that gives you an idea of who is most likely to churn.

By testing the model predictions on data it has never seen before, your data scientist should also have an idea of how accurate the model is. This is called "offline evaluation" (as opposed to "online evaluation", where the model is tested after it has gone live).

Most likely the model won't be perfectly accurate with its predictions, so you'll need to discuss with your data scientist how much accuracy you need for it to be useful.

For many use cases you can still get a lot of utility out of a predictive model that is only slightly better than random guessing. On the other end of the spectrum, there are some problems for which you'll require a

very high level of accuracy.

To decide which category your problem falls into, your data scientist will need to work backwards from your definition of "success" and determine how much accuracy the predictive model needs to achieve this. It might indeed turn out that you don't need an extremely accurate model to achieve what you want.

(Don't forget that you should have defined a success metric during onboarding!)

There are also dozens of metrics for assessing the accuracy of a machine learning model. It's your data scientist's responsibility to figure out which metric is most suitable for your problem.

Try to ensure they give you at least a cartoon explanation of whatever metric they decide on. Also, don't forget that this metric will usually not be the same as the success metric you agreed on during onboarding, although the two metrics should be clearly correlated.

This is super-important to understand, so here's an example.

You might have defined success as minimising the "percentage of customers who churn in the next month". And your data scientist's metric will be aimed at determining how good their machine learning model is at finding out *which* customers will churn.

But simply identifying these customers isn't the same as preventing them from churning. To achieve your goal of reducing churn, you'll need to take an action based on this knowledge.

The outcome of the data science project will depend on both the accuracy of the machine learning model *and* how successful you are at *acting* on the model's insights to encourage customers to stick with your subscription service.

Which leads us nicely to the next step…

The action stage

Once you're happy with the model then you'll need to "productionise" it in some way.

To do this, you'll need to think about who is the *consumer* of your data scientist's work.

Is the machine learning model meant to be a single component of a larger software system, in which case your data scientist will need to collaborate with your software engineers?

Or will your data scientist be creating marketing segments to be used in campaigns run by your lead generation function?

Model productionisation (or "deployment") is a large topic in its own right but it doesn't necessarily have to be complex, at least in the first iteration. You *might* need a data engineer but it isn't always essential.

If you're just starting out with machine learning then we'd recommend hiring a data scientist who can at least do the basics of model deployment, especially in situations where the data science function isn't yet mission-critical.

This is particularly true now that deployment in cloud environments like AWS or GCP is easier than ever. You can always hire a data engineer once your data scientist has proved the value of their work and they start hitting the limits of their deployment knowledge.

Online evaluation

The outcome of the action stage will be a productionised model that is

integrated into your business in some way.

Once your data scientist's model has gone live, its effectiveness should primarily be judged by your business-focused success metric. And this usually *won't* be the offline data science-specific metric your data scientist came up with in the prediction stage.

I know, I repeat myself, but this really is very important.

The last thing you want is to be discussing improvements in the Kullback-Leibler divergence (what?) when you really need to be focused on a plain-English metric like "percentage of churners".

(Incidentally it's common to monitor all sorts of metrics for a deployed model, including data science-specific metrics, but the purpose of this monitoring is primarily to ensure everything is running as expected.)

Your data scientist should be responsible for demonstrating that the data science project has actually resulted in an improvement to your success metric.

They should also be able to quantify this improvement. To do this, they might mention "running an A/B test". Or "multi-armed bandits" if they want to impress you with fancy words.

Regardless of which specific methodology they use, if you've hired a good person then they should have a plan for quantifying the improvement (if any) generated by their work.

Bizarrely, I've worked in a few places that skip the online evaluation step completely and rely solely on offline metrics to judge the success of their data science efforts.

Believe me, you don't want to be one of *those* companies…

Make your requirements clear

It's essential that you talk through what you're planning to do with the model *before* your data scientist starts doing any real work on your project.

This doesn't need to be completely set in stone but your plans will have a large influence on the approach the data scientist takes. You don't want a machine learning model that turns out to be useless because you didn't make it clear to your data scientist what the actual requirements were!

For example, perhaps you'll need the model to be interpretable, i.e. not a "black box" where you don't know *why* it's making certain predictions.

Or perhaps the model needs to produce predictions in real time. If you're trying to decide if a credit card transaction is fraudulent then your customers won't want to wait a long time for the transaction to be processed.

I'm feeling unimaginative so let's emphasise the point by talking through the churn example yet again.

Imagine that your data scientist builds a machine learning model that seems to be very accurate, and its main insight is that "people who have left the country are likely to churn".

But if your service only works in one country then you have no chance of convincing someone who is permanently leaving the country to continue using it.

So now you have a new requirement for the churn model. It doesn't just need to identify customers who are likely to churn, but it also needs

to find churn-tastic customers who can *also* plausibly be convinced to stay.

Now your data scientist will have to actually earn that high salary you're paying them!

(In the final chapter of this book, I'll talk about how to ensure your data science project is delivered successfully.

Subscribe to the DataPastry mailing list for more content like this. We hate spam so we only send stuff we think is great.

Want to discuss your own specific data problems? Book a free consultation with us – we're here to help!)

How to successfully deliver a data science project

(In the previous two chapters, I explained how to onboard your data scientist and what they should be doing from day to day in order to deliver maximum value.)

Minimum viable data science

For your first data science project, your data scientist should ideally work through the entire process from building a machine learning model through to live deployment as quickly as possible (think weeks rather than months).

Avoid the temptation to prematurely optimise any particular aspect of the process. The project as a whole should be viewed as an MVP, i.e. the smallest amount of data science work that will conceivably move the needle on your success metric.

You don't need perfection in order to show concrete results.

Once your data scientist has completed their first MVP, the end-to-end framework for building, deploying and evaluating machine learning models will be in place. They can then build on this and iteratively improve their data science solution over time.

Ensuring your data scientist's work gets into production ASAP also means they'll get visibility across the organisation very early on.

You'll find that other stakeholders suddenly take much more of an interest in your data scientist's output once they've put something concrete out into the real world. Expect to receive a lot of feedback of

both the positive and negative variety!

All this will provide further fuel for your data scientist to improve the quality of their work.

On the flip side, if your data scientist doesn't quickly put their models into production, then it'll be a long time – if ever – before you see any kind of value from them. It's common to see data scientists in this position wasting time on various theoretical exercises that go nowhere (or watching YouTube videos). This is very demoralising!

Empowering your data scientist

To get the most value from a data scientist, you should act as a facilitator for the work they do and make it easy for them to effect change. But what does this actually mean?

Put your data scientist in the middle

The weird hybrid nature of data science means that we data scientists usually need to communicate with all sorts of people across the company.

At various times data scientists will have to talk to programmers, data engineers, data analysts and design/business/product people.

We strongly suggest that your data scientist work in close collaboration with your programmers and data engineers. Only hire someone you'd trust to be comfortable in this environment and who can put code into production directly.

If you have a data engineer, they should help support your data scientist's goals and objectives. This means your data scientist will need to regularly communicate what they're working on to the data engineer.

Again, this *should* be obvious but I've worked in many places where the data scientists' and data engineers' respective goals were completely misaligned.

In addition to this, whatever your data scientist is working on will presumably have some kind of commercial impact, so they'll need to be able to pick the brains of the relevant business/product people.

Usually data scientists don't need or want daily contact with strange nontechnical folk, but it should still be easy for them to get hold of the people they need when necessary.

Often the work your data scientist is doing will need buy-in from stakeholders in some other business function. (I've lost count of the number of times I've had to present to people with titles like "Head of Marketing for Blabla".)

A good data scientist should be able to convince the relevant stakeholders of their project's business value without needing to resort to jargon. They should also be able to work with other functions to deliver their project without requiring too much in the way of adult supervision.

Last but not least, if you have a data analyst you'll want them to be seated close to your data scientist. This is a no-brainer as both parties should be able to assist each other a lot; in particular, your data analyst can help onboard your data scientist in the first couple of weeks.

So in summary:

- Embed your data scientist into your engineering team. If you hired a strong coder then they should be fine in this environment.
- If you have a data analyst, put them next to your data scientist. It might be a bit scary for an analyst to also work in the engineering

team, but most engineers are only really dangerous if provoked.

- Facilitate easy communication between your data scientist and the relevant business/product folks.

- Hire a data scientist who can put on their business hat and convince nontechnical stakeholders of the value of their work.

Incidentally, this team structure doesn't need to change much if you have multiple data scientists.

You might want to designate one data scientist as a team lead but often this doesn't make sense, e.g. if each data scientist is highly experienced and has ownership of their own projects.

Tools of the job

You've probably heard of the expression "a poor craftsman blames their tools", but as far as data science is concerned these are *not* good words to live by.

You should aim to get your data, infrastructure and code into a shape where your data scientist can run experiments quickly in short iterations. At a rough guess, I would say that issues with tooling can decrease your data scientist's productivity by a factor of up to 5.

Even worse, slow experiment cycles will push your data scientist towards using theoretical arguments about what will work best rather than testing their hypotheses empirically.

This is the wrong way to do data science – mainly because it leads to results that are wrong, period. So **make experimentation easy.**

Of course, tooling will never be perfect and a good data scientist should be able to work around minor problems. But you should listen to

your data scientist when they tell you that they're struggling to get much done due to issues with data, code quality or infrastructure.

Have a frank discussion about the impact of these problems and then prioritise fixes accordingly.

Problems with tooling can range from being a minor irritation to being the root cause of huge delays or major mistakes further down the line, so you'll want to make sure you tackle these issues *before* they negatively impact your business.

Personal goals

Data scientists are people and people usually have their own individual needs and desires. It's always good to help your data scientist achieve their long- and short-term personal goals.

Selfishly, this is in your interest as the data science market is very hot – you can expect an experienced data scientist to leave at the first possible opportunity if they're not happy with what they're doing.

We suggest that you conduct regular 1-1s with your data scientist to ensure they enjoy the work they're doing and to allow them to air any concerns they have in private (or more positively, to share any ideas they have).

Of course, this doesn't mean communication shouldn't also happen organically, but in a busy work environment it can be easy to overlook the wellbeing of any individual worker drone.

Try to make sure that the work your data scientist is doing is aligned with their own personal goals.

The reality of commercial data science work is that much of a data scientist's time can be consumed by data wrangling and other tedious

tasks. But you can try to mitigate the drudgery by finding work for them that is more intellectually fulfilling.

For example, if your data scientist wants to spend one day a week doing investigative work that isn't guaranteed to pay off, see if you can make this happen. It's entirely possible that their research could result in a big win for the business!

Lots of data scientists also like feeling like they're not isolated from the community at large, which is especially relevant if you only have one data scientist in your company.

If they're interested, let your data scientist take the time to go to data conferences and meetups. They can also talk about how great your company is for data scientists – or discourage anyone else from applying if it's actually a terrible place to work (just kidding, only people in awesome companies read our books).

Data science processes

Agile data science

It's a slightly controversial point, but I believe *commercial* data science work can and should obey normal software engineering practices.

In fact, I would argue that the best kind of data scientist is actually a decent software engineer who knows a lot about machine learning and is pretty good at communication.

(Aside: No, I'm not describing a "unicorn". Lots of software engineers work in fields such as game development that ***also*** *require advanced domain knowledge and we don't have special names for them. Why should data science be different?)*

Naturally, this goes hand in hand with the idea of embedding your

data scientist into the engineering team.

So this means that your data scientist should participate in sprints and standup meetings like everyone else. Data science projects can and should be broken down into small, clearly-defined tasks with concrete deliverables.

Of course, it can be difficult to scope data science tasks correctly – it's hard enough in software engineering! In particular, open-ended investigative work tends to be harder to estimate than coding and data munging.

A smart data scientist should know which tasks have the most uncertainty and work to reduce this as much as possible at an early stage, so estimation quickly becomes a lot more accurate.

In general we'd recommend using timeboxed spikes to handle investigative work. Otherwise it's too easy for a data scientist to go down a rabbit hole and spend a lot of time producing nothing.

Also, if your data scientist can't run experiments quickly then they'll struggle to fit their investigative work within an agile process, so this is another reason to make experimentation easy.

If the outcome of a particular piece of investigative work is particularly important (e.g. if getting it right will move the needle significantly on a success metric), then it's fair to be more generous with the time you allocate. But it's rare for it to be impossible to divide work up into small chunks.

Your data scientist should have a rough strategy in mind for how they want to perform the investigation, and this strategy can be used to codify specific tasks. This strategy won't be set in stone, but it will still provide a good starting point.

Sometimes the outcome of data science work won't be a piece of software but a set of insights to be used by, say, a marketing team. This work will still involve writing code and can be scoped and prioritised like any other software project.

Data-driven decisions are a lie

It's common for companies to claim that they make rational decisions based on data but in my experience very few companies are truly data-driven. This shouldn't be very surprising. Companies are comprised of ordinary people, and ordinary people aren't rational.

But try not to be *too* irrational when drawing conclusions from data projects. The list of possible ways in which you can be biased would be far too lengthy a detour for this chapter, but *confirmation bias* is a huge problem in this area.

It's cathartic for me to talk about my experience with this, so excuse me while I ruin the end of this book by ranting for a few paragraphs…

Occasionally I'll be given a data task by an important stakeholder as well as a set of thoughts on where to look first, what might be most predictive and so on.

I'm happy to incorporate any domain knowledge when engineering features, but I don't assume that a feature will be predictive just because someone with a fancy title thinks it's important.

Obviously I'm just as susceptible to confirmation bias as anyone else, so I try to put my own preconceptions aside and let the data decide what the real story is. It's not always easy to apply statistical methodologies rigorously but it's very necessary if you don't want to draw the wrong conclusions.

All good, right? Wrong!

The problems start as soon as I present my work to the main stakeholder; it quickly becomes apparent that they're only really looking for data to confirm the biases they already hold. I've always pushed back against this with mixed results.

The best-case scenario is that the stakeholder will defer to the person who's supposed to be the expert with data (not being arrogant or anything, but that'd be me). Unfortunately, it's somewhat more common to get overridden and any evidence that disagrees with existing assumptions ends up ignored.

At this point I just take the money and run.

In summary, it's only human to be biased – but if you're aware of your own irrationality, you're already halfway to solving the problem.

Your data scientist won't always be right but don't override them based on authority alone. *If* you've hired someone good, it's far more likely that they've drawn the correct conclusions from your data than you.

Much like in science, try not to fall in love with your own hypotheses and instead **let the data inform your decisions**.

(Subscribe to the DataPastry mailing list for more content like this. We hate spam so we only send stuff we think is great.

Want to discuss your own specific data problems? Book a free consultation with us – we're here to help!)