



Let's Talk Digital Series #1

DIY Guide on Implementing Data Science Projects

This article is for those who are interested in exploring or implementing data science projects, as curious individuals or as part of a team with little to no technical experience. Personally, my journey started when I was a Further Mathematics lecturer at a private college in 2016. I did not set out to be a data scientist, nor was data science popular in Malaysia at the time.

I realised I had a knack for analysing data using Microsoft Excel, by using nested formulas and generating insights through charts and so forth. I was scarred from programming – as I did not do too well in my Java class back in my university days. It was also during this time that I found out about the limits of Microsoft Excel: that it couldn't fit more than 1,048,576 rows when I had a dataset with a few million rows to analyse. It just so happened that Microsoft started offering its Professional Programme on edX, and so began my exploration of what R, Python, and SQL can do. No one in my social circle were in data professions, so I picked up the skills I needed on my own, and I learnt a lot the easy way and the hard way. My point is that it doesn't take a mathematics, statistics, or even a computer science genius to break into a data career. It is through these series of articles that I share my experience and hopefully this makes your learning journey much more productive.

Among the myths people hear about in the realm of data are:

- 1) Coding is complex and difficult to learn
- 2) The servers and infrastructure are expensive to setup and maintain
- 3) I'm in a different field, so it's not possible or easy to move to other fields such as data engineering or data science

I'll start by breaking down the first myth. Coding does look intimidating at the beginning, especially if one has no programming background. There are two skills every good coder has: the patience to read and understand code documentation, and the ability to break large problems into smaller solvable bits. Most codes available online have some sort of documentation which describes what each parameter supplied by the user does.

As for decomposing large problems into several ones, a lot of us have done it before. For example, organising an event like a conference requires several steps:

- 1) Getting a venue
- 2) Booking caterers for food and beverages
- 3) Confirming agenda and speakers
- 4) Trial run of events to make sure everything runs smoothly

The second myth is about the expensive setup and infrastructure costs, which can largely be addressed via cloud computing platforms. Organisations in the airline, media, and e-commerce platforms rely on the likes of Google Cloud Platform and Amazon Web Services because these services allow users to quickly experiment and test the feasibility of deploying their code for internal or external use quickly.

As for the third myth: for people to work as data scientists, they require a few components:

- 1) Proficiency in mathematics, statistics, or computer science
- 2) Soft skills for presentations and stakeholder management
- 3) Domain expertise




Data science is applicable in various industries but those with in-depth domain expertise will have better understanding of what the data means and will be able to manipulate data to come up with better machine learning models. For example, a data scientist with little domain expertise will fit a regression model to find the relationship between annual GDP, and variables such as consumer expenditure household, total goods and services

exports, total imports goods and services, etc. Someone with domain expertise would know it's better to use variables like total exports, total imports, and domestic demand per GDP without construction.

CLOUD COMPUTING PLATFORMS

Let's assume I'd like to build an image recognition system which recommends recipes based on user-submitted images. Here are the comparative steps I'd take to deploy an image recognition system.

ONSITE SERVER	CLOUD DEPLOYMENT
<p>1 Build image recognition algorithm, and house within a server</p>	<p>1 Build image recognition algorithm, and place the code in a virtual machine</p>
<p>2 Estimate the incoming number of images to determine the hardware specifications needed</p>	<p>2 Choose the appropriate virtual machine specifications, enable scaling options, and select region where images would be stored</p>
<p>3 Estimate the software license and hardware costs, power consumption, etc.</p>	<p>3 Control access to the core code by assigning permissions to user IDs like how it's done on Google Docs</p>
<p>4 Backup plan in the event of server maintenance and upgrades</p>	
<p>5 Ensure security steps taken are adequate for data protection</p>	

There are a few advantages of cloud computing platforms.



MAINTENANCE AND BACKUPS OUTSOURCED

Hardware maintenance and software backups are outsourced to the cloud computing service provider.



FREE CREDIT

All cloud computing platforms also provide users with some free credit for users to experiment with, so users can rapidly test and conclude whether their deployments are feasible.



REDUCED WASTE CAPACITY

Onsite servers tend to be underutilised outside of peak hours, so the extra capacity is wasted.

Cloud computing platforms give the option of scaling services according to usage, so wasted capacity is reduced.



SOFTWARE ONLY

If I were to build, test, and deploy an image recognition system, I would be able to complete the end-to-end construction within a few days, as my development work would focus only on the software side.

It's more effective for someone already familiar with the industry or domain to pick up data science skills, compared to someone already familiar with data science to pick up domain expertise over time. Here are a few more examples of roles where data science skills can be paired with and applied:

1) Financial Planning & Analysis: estimate marketing expenditure in a given city by predicting the number of new users of an app/product



2) Sales: classify and segment their customers based on credit worthiness using historical patterns of credit term, payment frequency, amount of orders, etc.

3) Risk Management: provide early warning predictions of potential defaults, loan delinquency, and customer churn (whether these customers will change banks soon)

4) Customer Service: provide valuable input onto constructing a chatbot to address customer needs and inquiries, and advise customers through certain steps

STEPS
TO CREATING A CHATBOT

When writing the code, here are the steps I took when building and deploying a chatbot to support customer service operations.

- 
1 CHOOSE OR BUILD
 Choose or build an engine which understands the intent behind a question or a sentence
- 
2 LIST
 List all possible questions users will ask a chatbot
- 
3 ANSWER
 Come up with all possible answers for the chatbot
- 
4 DEPLOY
 Choose a platform to deploy your chatbot
- 
5 TEST
 Test whether chatbot responses are succinct and enough for users

It takes some practice, but decomposing a large problem into several smaller pieces is already an innate skills in all of us. We just need to take that leap of faith and apply it in your own data project.

If you are still keen to implement your own project, whether to help your team manage their workload better or to increase efficiency, here are the 4 steps I usually take to implement data projects:

- 1) Getting a venue
- 2) Booking caterers for food and beverages
- 3) Confirming agenda and speakers
- 4) Trial run of events to make sure everything runs smoothly

For the initial assessment, I try to find out whether I have people I can count on for advice in the organisation. It's entirely possible that you want to affect the change, but lack the technical know-how of what to do or what services are needed. Granted that you'll be doing most of the grunt work, it's easy to gain allies when you can demonstrate a clear advantage / value proposition. If your organisation doesn't have a data team in place, you're going to need to rely on your network of friends, or those in the IT department for advice. If your organisation already has a data team in place, your journey will be easier as you can ask them for guidance.

The next step is to identify a use case, and to perform the business and technical diligence. A few examples of solid use cases would be:

- 1) Build a chatbot to reduce lag time of customer inquiries to a few minutes from 3 days
- 2) Construct an Optical Character Recognition (OCR) system to speed up information retrieval from physical documents within a few seconds from a few days (from data entry work)
- 3) Automatically recommend products for cross-sell/upsell to customers periodically by inferencing from their demographics, daily transactions, web activity, etc.

Any successful experiment requires a proof-of-concept: basically, a simple version of the code which works. Going back to my image recognition system example, the proof-of-concept would be to demonstrate the code works on my computer, and the approximate accuracy based on real world input. The code should work by considering images which come in that have various resolutions, lighting conditions, background noise, etc. You should test extensively to find out the weaknesses of your code. Once you have a functioning proof-of-concept and a few thoughts on how to address the weaknesses of your code, the hard part of convincing your stakeholders to invest time and money into your project begins.

The last phase of the project would involve stakeholders like Management, Corporate Communications, and IT for purposes ranging from ensuring the correct message is reflected on your system/app, end-to-end integration with your organisation's systems, and budget approval. Most data professionals are under the impression that the technical bits are the most difficult part of their job. I can attest that it's not: it's the stakeholder engagement and management that takes up the most of my time.

The past 4 years have been an adventure for me: from exploring and using Microsoft Azure for database operations during my INVOKE days, developing and deploying image recognition and chatbot systems for enterprises, and now exploring applications of augmented reality systems for applications in the media industry. I've learnt a lot of what I know with minimal guidance from my peers, and I hope you will be able to leverage the expertise among your own social circles to implement your own data projects. One of my observations is; those who excel in their careers tend to invest a lot of their time learning new skills and exploring new fields. By highlighting interesting facts from my readings, studies and observations from my career, I hope these pointers inspire you to be better in your respective careers.

PERFORMING BUSINESS AND TECHNICAL DILIGENCE

WHEN IMPLEMENTING A DATA SCIENCE PROJECT



What do business and technical diligence mean?

BUSINESS DILIGENCE

- 1) Researching on whether others have implemented similar systems;
- 2) Learning from their lessons; and
- 3) Estimating the effort and costs needed to achieve the business value intended.

TECHNICAL DILIGENCE

- 1) Knowing which systems or platforms to integrate to realise your goal;
- 2) Experimenting and estimating the possible margins of error, false positives, and false negatives; as well as
- 3) How well your solution integrates with existing systems.

Assuming I'm building an image recognition system, here are the questions I would seek to answer.

BUSINESS DILIGENCE



1 Have others tried using image recognition to recognise ingredients?



2 How successful are these systems, and how much value did they bring to their business?



3 What are the costs and effort needed to develop such a system?



4 How much value does this create for your business?

TECHNICAL DILIGENCE



1 What's the most efficient way to build the script? Does it require building from scratch, or can I rely on a few public APIs?



2 Once the first code is created, how do I construct the pipelines to feed real-time data? How do I deploy this model, and where will the code output be used?



3 Using your proposed solution, what are the limitations of your code? Under what conditions will it function well or poorly?



4 Can the code be integrated with existing systems?

The logo for Asian Banking School, featuring the text "ASIAN BANKING SCHOOL" in white, uppercase letters on a dark blue square background.

ASIAN
BANKING
SCHOOL

This article is part of the Digital Banking Learning Series, 'Let's Talk Digital', an initiative by the ABS Center for Digital Banking. It is written by industry practitioners and are aimed at educating the general public on the intricacies of digital applications in banking and other related industries, including the latest insights and trends of Digital Banking.

As the industry's preferred partner in learning and development, ABS offers relevant training programmes that covers a comprehensive list of banking areas that are designed and developed in-house by our Specialist Training Consultancy Team or in collaboration with strategic learning partners that includes some of the top business schools in the world. It also provides specialised consulting services and tailored learning solutions to meet the specific needs of its clients.

For more information, visit our website at www.asianbankingschool.com or email us at digitalbanking@asianbankingschool.com