Building reproducible analytical pipelines with Python

Bruno Rodrigues

2024-01-13

Table of contents

Welcome!

How using a few ideas from software engineering can help data scientists, analysts and researchers write reliable code

This is the Python edition of the book "Building reproducible analytical pipelines", and it's a work-in-progress. If you're looking for the R version, visit this link.

Data scientists, statisticians, analysts, researchers, and many other professionals write a lot of code.

Not only do they write a lot of code, but they must also read and review a lot of code as well. They either work in teams and need to review each other's code, or need to be able to reproduce results from past projects, be it for peer review or auditing purposes. And yet, they never, or very rarely, get taught the tools and techniques that would make the process of writing, collaborating, reviewing and reproducing projects possible.

Which is truly unfortunate because software engineers face the same challenges and solved them decades ago.

The aim of this book is to teach you how to use some of the best practices from software engineering and DevOps to make your

projects robust, reliable and reproducible. It doesn't matter if you work alone, in a small or in a big team. It doesn't matter if your work gets (peer-)reviewed or audited: the techniques presented in this book will make your projects more reliable and save you a lot of frustration!

As someone whose primary job is analysing data, you might think that you are not a developer. It seems as if developers are these genius types that write extremely high-quality code and create these super useful packages. The truth is that you are a developer as well. It's just that your focus is on writing code for your purposes to get your analyses going instead of writing code for others. Or at least, that's what you think. Because in others, your team-mates are included. Reviewers and auditors are included. Any people that will read your code are included, and there will be people that will read your code. At the very least future you will read your code. By learning how to set up projects and write code in a way that future you will understand and not want to murder you, you will actually work towards improving the quality of your work, naturally.

The book can be read for free on https://b-rodrigues.github.io/raps_with_py/ and you'll be able buy a DRM-free Epub or PDF on Leanpub[1] once there's more content.

This Python edition is shorter than the R version. Here's the topics that I will cover:

- Dependency management with `pipenv`;
- Some thoughts on functional programming;
- Unit and assertive testing;
- Build automation with `ploomber`;
- Literate programming with Quarto;
- Reproducible environments with Docker;

[1] https://leanpub.com/

- Continuous integration and delivery.

While this is not a book for beginners (you really should be familiar with Python before reading this), I will not assume that you have any knowledge of the tools discussed. But be warned, this book will require you to take the time to read it, and then type on your computer. Type *a lot*.

I hope that you will enjoy reading this book and applying the ideas in your day-to-day, ideas which hopefully should improve the reliability, traceability and reproducibility of your code. You can read this book for free on TO UPDATE

If you want to get to know me better, read my bio[2].

You can submit issues, PRs and ask questions on the book's Github repository[3].

[2]https://www.brodrigues.co/about/me/
[3]https://github.com/b-rodrigues/raps_with_py

Preface

I don't like Python. Or rather, don't like using it to analyze data. I believe that certain design choices were made that make Python a subpar language for analyzing data. That being said, Python is currently the most popular general purpose programming language, and also the most popular language to analyze data, especially in machine learning and AI. As such, it is a good idea to at least know your way around it even if, like me, you prefer using the R programming language.

I state that I don't like Python because I want to make something very clear, right from the start. I am not an expert in Python. I know enough to get things done, but it is not a language that I know well. I consider myself an expert in R, but definitely not in Python. So why write a book on Python in that case? Well, in truth, this is not a book about Python.

This book is about building reproducible pipelines. And in order to build reproducible pipelines, you need to apply certain general ideas. This book will discuss these ideas, and illustrate them using the Python programming language. I wrote such a book already: you can read it here[4]. The book was well received: lots of people contacted me to thank me for having written it, I was invited to give talks several times on the book already, give some workshops and some people are already asking me for a second edition. I thought it would be an interesting challenge to write

[4] www.raps-with-r.dev

a Python edition. It would be a good excuse to dive back into Python after having barely touched it since my Phd days, more than 10 years ago, where I used it alongside R for a project. It is also a good opportunity to see what is currently available in the Python programming language, and if it would be possible to reproduce the way I work with R.

There are currently some interesting packages available for Python that are quite close to some available for R. But R and Python have some very fundamental differences in their design that make using Python feel awkward if you're used to R, especially if take full advantage of R's functional programming features. But it's not just these design choices that differentiate both languages, there are also differences in culture, in how people use them. For example, notebooks are very popular in the Python world, but it's quite rare to see someone share R code through a notebook. In general, R users tend to write code as plain-text scripts or in Markdown using Rmarkdown or Quarto. Throughout this book, I will be writing Python with a heavy R accent, mainly for two reasons: first, because I'm fluent in R, but not in Python, as I've stated above. Second, because I also think that some of R's idioms actually make it more readable than equivalent Python code, and thus we can make Python more readable if we make it look more like R.

If the previous sentence didn't scare you into continuing and you are actually curious to see what I mean, then read on, you're in the perfect state of mind to approach this book!

Let me finish this section by talking a little bit about the history of these books (the original R edition and this one). It all started when a former colleague of mine contacted me in the summer of 2022 to ask me if I was interested in teaching a course for the Data Science Master degree at the University of Luxembourg. Long story short, I've decide to teach students how to

set up data science projects in a way that they're reproducible. I wrote my course notes into a freely available bookdown that I used for teaching. When I started compiling my notes, I discovered the concept of *Reproducible Analytical Pipelines* as developed by the Office for National Statistics (henceforth ONS). I found the name "Reproducible Analytical Pipeline" (henceforth RAP) really perfect for what I was aiming at. The ONS team responsible for evangelising RAPs also published a free ebook[5] in 2019 already. Another big source of inspiration is Software Carpentry[6] to which I was exposed during my PhD years, around 2014-ish if memory serves. While working on a project with some German colleagues from the University of Bonn, the principal investigator made us work using these concepts to manage the project. I was really impressed by it, and these ideas and techniques stayed with me since then. This was also the project where I've used Python.

The bottom line is: the ideas I'm presenting here are nothing new. It's just that I took some time to compile them and make them accessible and interesting (at least I hope so) to users of the Python programming language, even if I'm not one.

If you have feedback, drop me an email at bruno [at] brodrigues [dot] co.

Enjoy!

[5]https://ukgovdatascience.github.io/rap_companion/
[6]https://software-carpentry.org/

References