MDA book club do4ds

MDA

2024-04-05

Table of contents

Preface

The MDA team at CORI will do a book club and review the book DevOps for Data Science wrote be **Alex K Gold** here.

Book club inspiration: https://github.com/r4ds

proposed workflow:

1. Clone the repo

2. Create a specific branch: ` git checkout -b intro_env_as`

3. Commit as needed

4. Push your commits in a pull request

This is a Quarto book.

To learn more about Quarto books visit https://quarto.org/docs/books.

Introduction

Learning objectives:

- introducing some definitions and a bit of history

Definitions

production :

> affecting decision in your orgs / world
>
> putting your work in front of someone else's eyes

We want:

- our works to be **reliable**
- in a safe **environment**
- our work to be **available**

DevOps

> DevOps is a set of **cultural norms**, **practices**, and **tooling** to help make developing and deploying software **smoother** and **lower risk**.

.. but is a squishy concept and a "vendor" associated name

It came in opposition to the *waterfall dev process* were you had a team doing Dev. and then one other doing Ops (Ops "make it works on everyone computer").

Process and people

- Are data scientist software developper?
- Are we in the red flags number 3?
- Do we need a workbench (ie using ec2) ?
- should we do the exercice with `penguins` or one of our dataset?

Part I

DevOps Lessons for Data Science

You are a software developer.

But:

- Writing code for data science is different than writing code:

  You're pointed at some data and asked to derive value from it without even knowing if that's possible.

- difference between architect and archaeologist

1 Environments as Code

1.1 Environments:

- stack of software and hardware below our code
- should be treated as "cattle not pet" / should be *stateless*
- Risk of it "only works on my machine"

Building a completly reproducible environement is a "fool's errand" but first step should be easy.

(any trouble with renv and sf anyone?)

1.2 Environements have layers

Layer	Contents	Example
Packages	R packages	cori.db
System	R versions / GDAL / MacOS	14.4.1
hardware	Physical / Virtual hardware	Apple M3

Hardware and System should be in the hand of IT (see later chapter 7 and 14), packages layer should be the data scientist.

1.3 The package layer

Package can in 3 places:

- repository: CRAN / GH / "Supermarket"
- library: a folder on a drive / "pantry"
- loaded : "ready to cook"

Each **project** should have it's own "pantry"

Project was higlighted in text but I think it is important: if you do not have a project workflow it is way harder to do it.

A package environement shouldbe :

- isolated and cannot be disrupted (example updating a packge in an other project)
- can be "captured" and "transported"

In R: `{Renv}` ("light"/"not exactly the same" option also exist, Box, capsule)

Author does not like Conda (good to not being alone!)

1.4 Workflow

- Create a standalone directory with a virtual environment

(spend time exploring `renv/` and `.gitignore`)

- Document environment state (see `lockfile`)
- Collaborate / deploy: you can't share package because their binay can be OS or system specific, hence specific package need to be installed (could be a pain point).
- Use virtual env

1.5 Under the hood

- test `.libpaths()` in a specific project and in a "random" R session
- order of Paths matter

1.6 Key points

- being in production is what make a DS a software developper
- kill and create new environment fast is important

2

3

4

5

6

Part II

7

8

9

10

11

12

13

14

Part III

15

16

17

18

A

B

C

D