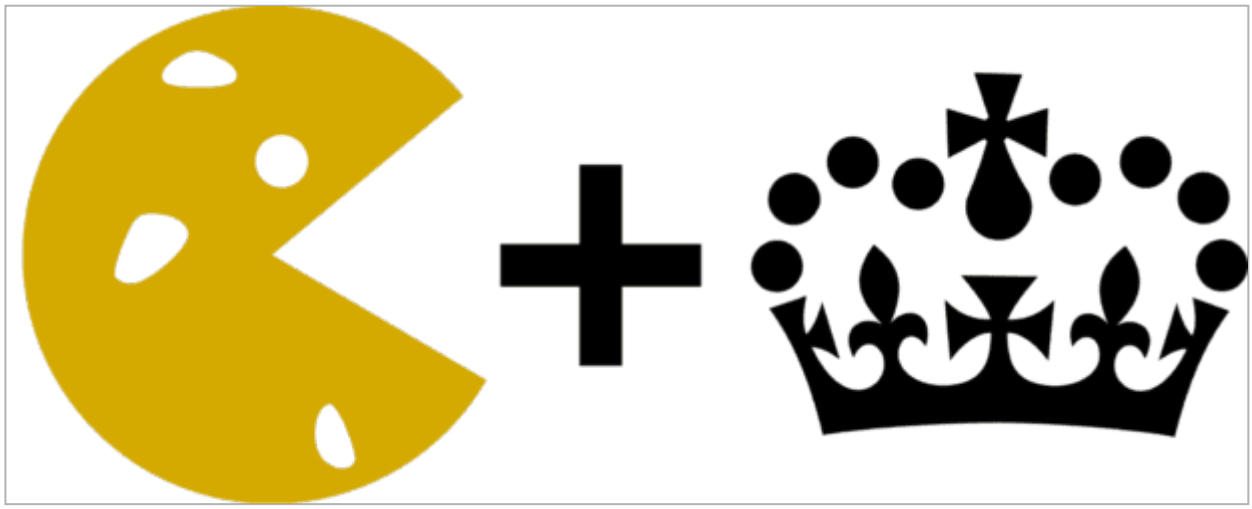
govcookiecutter: A template for data science projects

Eric Young - Data Scientist, GDS, 20 July 2021 - agile, Data science



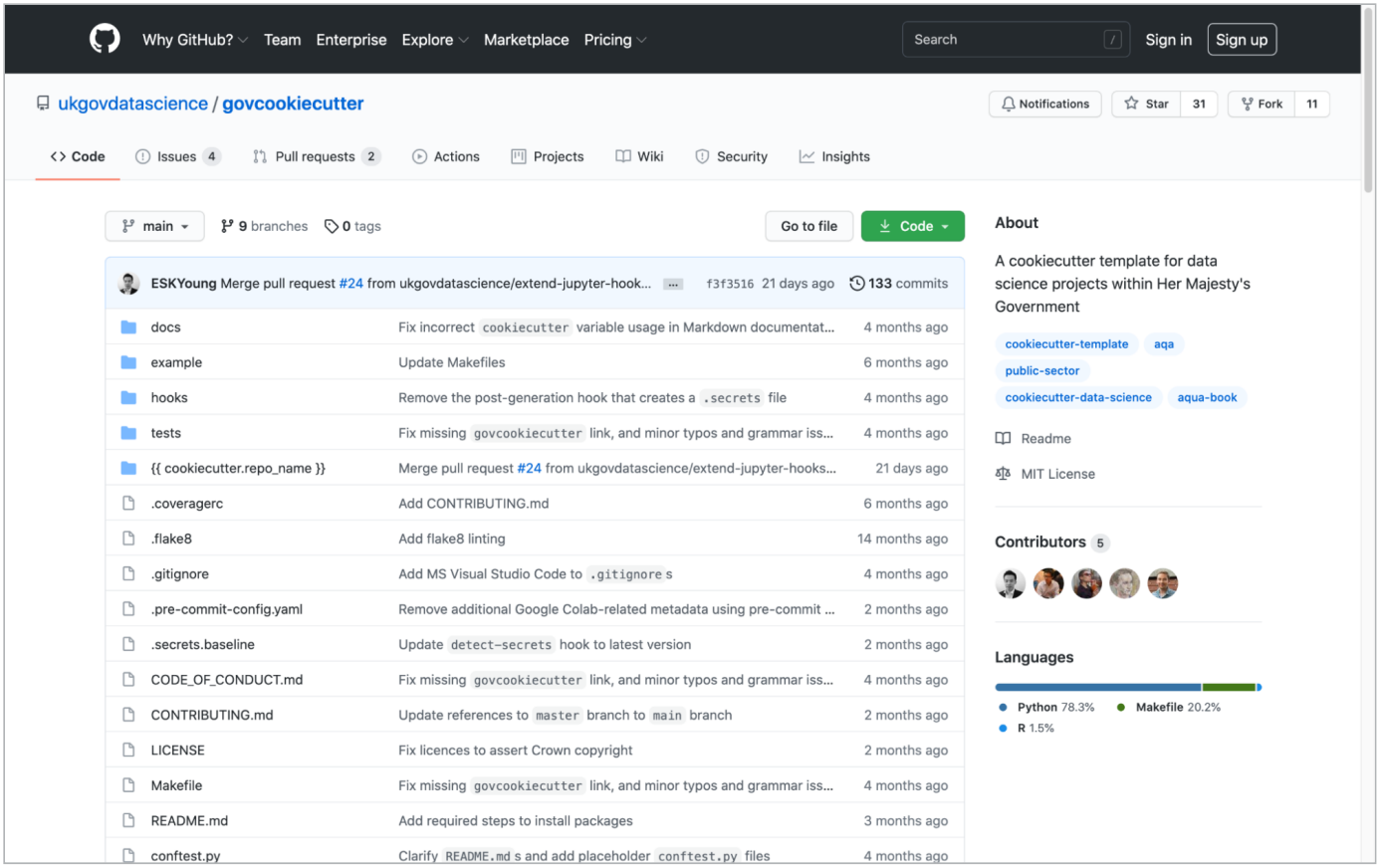
Agile data science is hard. Data exploration and cleaning, researching techniques, and generally “doing data science” takes time — think months rather than weeks.

In the Civil Service, we must also ensure our analyses are fit-for-purpose. By analysis we mean anything with an input, some processing and an output, like a machine learning pipeline, a dashboard, or even a spreadsheet.

Assured and robust analysis is important to avoid unintended consequences, which could impact individuals and their livelihoods. The Aqua book provides high-level guidance around producing quality analysis in government, and these analytical quality assurance (AQA) principles must be followed in our work.

Iterative, incremental, and evolutionary delivery is a key part of Agile. How can we balance this with the need for robust AQA, without grinding delivery to a halt? And, equally importantly, how do we make sure AQA is done?

Baking with govcookiecutter



To try and address these needs, the GDS data science team created govcookiecutter. By answering a few prompts, this generates (bakes) a project structure with a range of AQA features. We can't tell you what checks to do — that varies between projects — but we can make it easier for you to do them.

Some assumptions to start though:

1. You're using Git for version control with either GitHub or GitLab
2. You have access to Python or both Python and R
3. Ideally, you have a Unix-based machine, although most features will work on Windows!

Most of the features use Git hooks based on the pre-commit framework; these hooks run checks before you even write a message for your commit! If any fail, then you won't be able to commit code until the failing checks are resolved. For R users, we have also implemented most of these hooks.

Want to see a live demo with more details about govcookiecutter? Check out this live recording from earlier this year!

Keeping data and secrets safe

On the most basic level, govcookiecutter-based projects don't track any files inside the data folder. But there is a hook to check if you are trying to commit files larger than 5MB as well, just in case there are any stragglers.

Another risky area for data leakage is in Jupyter notebooks, a popular tool for data scientists. Executing notebooks leaves outputs on display, which can end up in version control. In addition to making tracked changes difficult, some of your sensitive data could also be exposed in these outputs. To prevent leaking data, the nbstripout hook cleans up all your Jupyter notebook outputs, except for explicitly-tagged cells.

The detect-secrets hook tries to identify secrets (for example, credentials, API tokens) and prevent them being version-controlled. It uses regular expressions, entropy detection (heuristic approaches to find 'secret-like' entries) and keyword detection in its searches, but it's not foolproof, so should only complement, not replace, your organisation's best practice.

But you will still need to use your secrets locally. To do this, you can use the untracked `.secrets` file to store all your secrets as environment variables. You can then load these environment variables in your scripts, safe in the knowledge that your secrets will stay local.

Documentation

Keeping documentation up-to-date is tricky, especially if it's stored far away from your code. With the `docs` folder, govcookiecutter-based projects keep documentation in one place that's easily accessible for anyone with access to your repository. It also means reviewers can check that documentation has been updated via the commits.

The `docs` folder also stores all the AQA documentation, including departmental frameworks, AQA plans, and assumption logs, so everyone can clearly see and access them.

We've documented all the features discussed in this post so you don't have to, and we've also set up Sphinx, a Python documentation generator used by many major packages, so you can (optionally) build a searchable website of all your documentation quickly and easily.

Testing and structure

Verifying your work is a key pillar of AQA, and one way to do this is to write tests for your code. Instead of spending time configuring your test suite, we have set up the pytest framework for you, as well as coverage.py for code coverage, so you can get on with writing tests, not configs.

A consistent project structure also means it's much quicker to bring colleagues into your project with everyone having an agreed understanding of which files go where.

Bringing it all together for Agile

And how do we make sure you and your contributors do these checks? Whenever a pull or merge request is raised, govcookiecutter-based projects use a request template that has a checklist for contributors to tick off.

This reduces the burden of filling out lots of documentation, when all the details should already be in commits, their messages, or in pull/merge requests comments. And it provides a lightweight, but auditable way to quickly ensure appropriate AQA has been completed for this branch of code.

The future of govcookiecutter

Going forward, there are a few more things we would like to add, but we would also love contributions to the project. It's open source, and freely accessible to many public sector data scientists, so would be a good opportunity to showcase your skills. Feel free to fork the repository and add your contributions!

We would also love to incorporate AQA frameworks from other government departments, and public sector organisations into govcookiecutter, so that others can see and improve on best practice; contribute directly on the GitHub repository, or drop us an email.

For standalone R users, we ran a poll earlier this year where 82% of respondents (32 out of 39) wanted a pure R version. If you're interested in this, get in touch, and have a look at this issue.
