Data Science Training Courses

Ian Ozsvald's Public and Private training courses for 2024

Version - 2024-01

Ian Ozsvald

@IanOzsvald – ian@ianozsvald.com

Table of Contents

Table of Contents

Overview of the courses

Each course is held online in a small group of **circa 12 people**. We use **Zoom & Slack** to collaborate and locally each student runs a **Jupyter Notebook** to work through **pre-written Notebooks and scenarios**. Courses normally run **during UK hours**.

Up to date versions of this document can be downloaded at https://ianozsvald.com/training/ where all public courses are listed.

The *Higher Performance Python* course focuses on making your Python scientific code run faster and scale to larger datasets. This includes understanding what's slow with profiling, optimising code, compiling numeric code, making Pandas more performant and scaling to larger datasets with Dask and Vaex. We explore making code 10-100x faster.

The *Software Engineering for Data Scientists* course helps students who lack strong engineering skills to develop robust practices for better trusted and more easily supported code. We focus on making development faster and more trusted. This course focuses on diagnosing bad code, refactoring to make code clean and modular, a standard folder layout and pragmatic approaches to testing.

Successful Data Science Projects is aimed at students who wish to more reliably deliver data science projects that add value to an organisation and to avoid failed projects. The course focuses on a project planning document to derisk uncertainties and failure points, reviews tools that enable fast data derisking and discusses scenarios that stretch a student's ability to identify risk and costs in potential projects.

Fast Pandas teaches the practical Pandas user all the ways to make common operations faster for day to day use along with demonstrating faster production options using GPUs. We discuss the transition to the new Pandas-3 facing world with Arrow and Copy on Write and review how you'll transition from Pandas 2 to this faster future.

Scientific Python Profiling helps quants and researchers figure out what's CPU-bound and RAM-bound in their code so they can try alternative faster solutions. Students will have a thorough understanding of their options by the end of this half-day course.

About Ian Ozsvald

Ian is a Chief Data Scientist and Coach, he's helped co-organise the annual PyDataLondon conference with 700+ attendees and the associated 11,000+ member monthly meetup. He runs the established Mor Consulting Data Science consultancy in London, gives conference talks internationally often as keynote speaker and is the author of the bestselling O'Reilly book High Performance Python (2nd edition). He has 18 years of experience as a senior data science leader, trainer and team coach.

Blog and past talks: https://ianozsvald.com/

Newsletter: https://notanumber.email/

Email: ian@ianozsvald.com

LinkedIn: https://www.linkedin.com/in/ianozsvald

Twitter: @ianozsvald

GitHub: https://github.com/ianozsvald

Higher Performance Python

This course mixes numeric scenario solving, new high performance tools and processes to explore ways to help you and your team be more performant and to write more performant solutions with Python. This is particularly focused on scientific Python.

It is *aimed at existing Python programmers* who have 2+ years of prior programming experience and who *need their Python code to run faster*.

Students can include **quants** and **modelers** who use the Python scientific stack day to day, **software engineers** who support scientific Python code and **researchers** who are cross-training into Python from other languages such as .net and R. In particular students probably have frustrations with their own code running too slowly or running out of RAM and they're looking for solutions.

We focus on the Python tool stack that is most useful to data scientists, quants and researchers. We look at generating 10x-100x speed improvements.

The course covers:

- Identifying slow code via profiling (`%timeit`, `py-spy`, `line_profiler`, `memory_profiler`, `ipython_memory_usage`)
- Contrasting the CPython and PyPy Python environments
- Moving numeric simulation code from pure (slow) Python to NumPy with and without vectorisation, then compiling with Numba and moving to multi-core OpenMP support for an ultimate speed-up
- Identifying opportunities for better algorithmic choices to improve execution speed
- Moving to coarse grained parallelism with `JobLib` and `Dask`
- Making Pandas faster by dipping under the hood and using compilation
- Scaling to larger datasets using `Dask` and `Vaex`

Course details:

- The public course is delivered as **3 morning sessions** in UK hours
- Light optional homework is included which is fully discussed on the subsequent morning
- We use Jupyter Notebooks and common Python tools

Take homes:

- Full solutions to all exercises are included so they can be reviewed offline

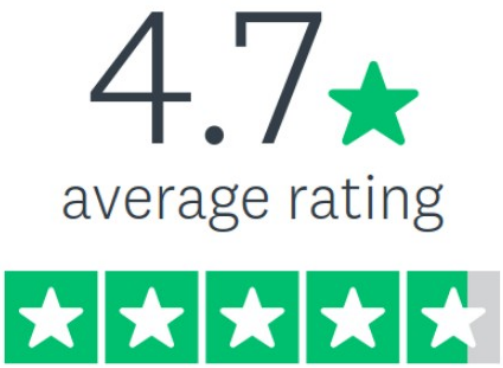
- A cheatsheet lists when each tool should be used when back in the office
- All slides and code used in the course
- A certificate of professional development
- A follow-up call 2 weeks after the course to answer outstanding questions

Teacher Ian is the co-author of O'Reilly's highly rated 2nd edition High Performance Python book. This course is based both on client experiences from Ian's consulting and from his work developing the book and his numerous and highly regarded public talks.

"I would highly recommend Ian's Higher Performance Python course to anyone who is looking for a solid understanding of optimizing native Python with tools like numpy, numba, dask, and many others. His training course is packed with practical examples of how to correctly profile and optimize code, hands-on tasks and in-depth discussions." - Elena Sharova (ITV)

"Thanks Ian! a very practical course on how to improve python programs from the investigation to focus on what is relevant, to the solutions with some ready-to-use tools and techniques. Also a big thumbs up for the quick switch to online course, it was very smooth." - Stephane, OasisLMF (Catastrophe modelers)

Post-course anonymous student feedback from multiple runs of the public and private iterations of this course has yielded a rating of 4.7/5.0.



Software Engineering for Data Scientists

This course helps you to develop a standard process from R&D through to production backed by code reviews, documentation, refactoring, unit tests and a Notebook based git process.

You'll learn to develop faster, build reusable code, both trust your code and also have a clean path to refactoring using tests so you'll deliver higher quality code that's more dependable for less effort overall.

It is *aimed at existing Python programmers* who have 2+ years of prior programming experience and who *need their Python development process to be more efficient and trusted.*

Students will include developers, quants and researchers who don't currently write tests and who have an ad-hoc approach to development. Students will have had frustrations with unwieldy code bases, may lack confidence in their code's correctness and have a desire to learn stronger industry-standard processes.

The course covers:

- Best practice for strong development processes
- Learning to identify "smelly code" with code reviews
- Pragmatically writing tests that enable code to be refactored so it "smells good" using `pytest`
- Writing just enough tests using the `coverage` tool and creating a defensive data ingestion process with `pandera`.
- Tools to support using Git during research with `nbdime` and `jupytext`
- Building a standard folder layout for research-to-deployment that a team can adopt with `cookiecutter` and a standard code format with `black`.
- Knowing when to use `conda` and `venv`.
- Knowing how `conda` and `pip` setup a Python environment's folders and search path so you can find and edit installed modules.
- Learning about the differences in a standard engineering project vs a data science project which will help you communicate better with existing software engineering teams

Course details:

- The public course is delivered as **3 morning sessions** in UK hours
- Light optional homework is included which is fully discussed on the subsequent morning
- We use Jupyter Notebooks and common Python tools

Course outline:

Session 1

- What's been a problem for the students? (discussion)
- Best practice (lecture & discussion)
- Diagnose some bad Notebooks with a code review (practical - learn to spot what's bad)
- Writing unit tests (how they work, write using `py.test` and see `coverage.py`)
- Homework - complete some unit tests, refactor some code using unit tests

Session 2

- Review the homework
- How environments work (discussion on `venv` and `conda`, where files live, how to edit an installed module and development installs)
- Practical on cleaning up Notebooks (`jupytext` demo, manually extracting code, `nbdime` with light git demo)
- Adding unit test to the extracted utility.py
- Cookiecutter folder layouts
- `Pandera` for data quality checking
- Homework - build out the `Pandera` data quality checker (this is a `Pandas` schema checker)

Session 3

- Review the homework
- Reviewing a finished example of the code, having moved our bad example to a good example
- The Python data model (objects, standard types, `numpy` and `Pandas`)
- Bad `Pandas` (discussion on things to watch for and how it can hurt - setting with copy warning, inplace, attributes vs column access, varieties of datetimes)
- Putting it into practice (discuss what you'll take and unblock any issues)

Take homes:

- Full solutions to all exercises are included so they can be reviewed offline
- A cheatsheet lists when each tool should be used when back in the office
- All slides and code used in the course

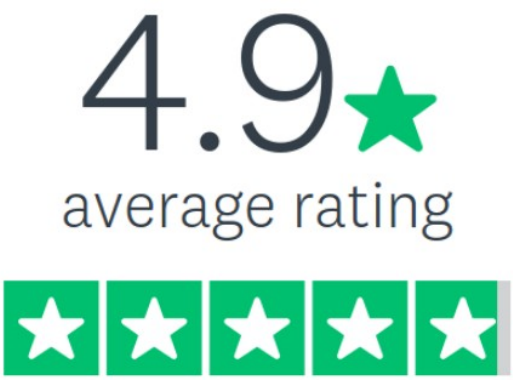
- A certificate of professional development
- Email support for 2 weeks after the course for follow-up questions

This course is built out of teacher Ian's hard-won client experience working on R&D projects and taking them through to deployed solutions as both solo researcher and as a part of rapidly growing teams.

"Ian's course Software engineering for Data Scientist was really useful to me. I learned more about refactoring and testing which I implemented at work in my current project the week after the training. There are other good practices (including the use of libraries I didn't know) that I am willing to put in place in the future." - Sandrine Pataut (QBE)

"Ian's Software Engineering for Data Scientists course provides an excellent overview of best practices with focus on testing, debugging and general code maintenance. Ian has a wealth of experience and also makes sure to keep on top of the latest tools and libraries in the Data Science world. I would especially recommend the course to Data Science practitioners coming from an academic rather than software engineering background." - Mirka (LibertyGlobal)

Post-course anonymous student feedback from multiple runs of the public and private iterations of this course has yielded a rating of 4.9/5.0.



Successful Data Science Projects

This course covers new tools and processes and covers an effective project planning approach that will help you reduce failures and improve deliverable outcomes at work for your data science projects.

It is *aimed at anyone who has had failures in their data science projects who'd like to deliver more frequent and more successful projects*.

This includes **researchers**, **quants** and **project managers** on data science teams. You've seen what can go wrong, you know that different failed projects fail in different ways and you'd like to find a path around future failures.

The course covers:

- Working through scenarios to build up project plans that define what we need, what we know, risks we see and milestones that the business can agree upon
- Best practices and approaches used by successful teams
- New data science tools to quickly derisk new data to enhance your Exploratory Data Analysis process using `Pandas Profiling`, `dabl`, `Jupyter Voila` and `Jupyter Widgets`

Course details:

- The public course is delivered as **2 morning sessions** in UK hours
- Light optional homework is included which is fully discussed on the subsequent morning
- We use Jupyter Notebooks and common Python tools

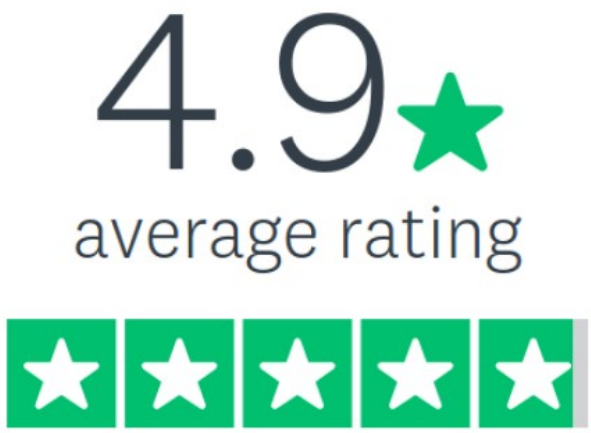
Take homes:

- Full solutions to all exercises are included so they can be reviewed offline
- A cheatsheet lists when each tool should be used when back in the office
- All slides and code used in the course
- A certificate of professional development
- A follow-up call 2 weeks after the course to answer outstanding questions

"One of the highlights from Ian's Successful Data Science Projects course was being introduced to the concept of a specialised project specification document. This provides a systematic framework to directly tackle numerous problems I have experienced when trying to move a project beyond an initial prototyping stage. I have now applied my own tailored specification document at my organisation and it immediately surfaced critical questions and issues that otherwise would not have been realised for months." - Thomas Brown, Data Scientist at aire.io

"After attending the course I can identify and communicate to the project team and client the uncertainties of the project efficiently. I am using the techniques covered on the course to write project initiation documents and put in place the necessary processes to reduce uncertainty. The course was very engaging and I was very happy to learn from Ian's experience to ensure a successful delivery on all future projects." - Dani Papamaximou, Data Scientist at Arcadia

Post-course anonymous student feedback from multiple runs of the public and private iterations of this course has yielded a rating of 4.9/5.0.



Fast Pandas

This course teaches many ways to make your Pandas research and production code run faster and use less RAM both on the CPU and GPU.

Brief: Frustrated that your Pandas code is slow, runs out of RAM and often uses just one CPU? Have you wondered which tricks and best practice you're missing?

In this course we'll look at improving your day-to-day work with Pandas, dig behind the scenes to see the BlockManager (for NumPy) and the new Arrow-backed series types, discuss where compilation can work (and where it can't!) and investigate ways to parallelise to the CPU and even the GPU.

Notable topics include - trialing the soon-to-be-default Copy on Write behaviour for 5x speed-ups and large RAM reductions, using Arrow for 8x faster strings and RAM reductions, concat-the-right-way to save 10x time and RAM, joining the "right" way can be 5x faster than the wrong way, vectorisation vs the pragmatic apply to save developer time, parallelising on the CPU with Pandarallel for easy 4-8x speed-ups, moving pipeline code to the GPU with CuDF for 10-100x speed-ups (and some effort - I'll tell you how to avoid the warts), using Parquet for 10x speed-ups over CSV. We'll also save RAM by utilising smaller datatypes for a 2x RAM reduction. We'll finally review some frequently missed built-in methods that can save developer effort.

Outcomes: Students will have new techniques to make their day to day Pandas usage faster, along with options to make pipelined long-running jobs faster which would justify some re-engineering.

Audience: You use Pandas weekly, your longer running jobs take 1+ hour to complete, often you use a single CPU despite having a multi-core system, you're limited because you run out of RAM and your research code takes just long enough to interrupt your thinking. This course does not teach how to learn Pandas, it assumes at least some familiarity with day to day Pandas operations.

Tools used or demonstrated: Pandas 2, `Arrow`, `Pandarallel`, `Swifter` (demo), CuDF (demo on GPU)

Format: 1 day on-site or remote, class size 10-14 people. We use Visual Code used to run Jupyter Notebooks.

Scientific Python Profiling

Aimed at quants and researchers who want their Python CPU-based and possibly memory-bound code to run faster and with less memory pressure.

Suits: Existing users of scientific python tools (e.g. quants, researchers, engineering support teams) who don't understand the memory and CPU cost of common tools like Pandas and NumPy. In addition we can briefly look at Scalene and can review some of the performance differences between Pandas and Polars, as directed by conversation. We'll review how the tools can fit your development process.

Tools: We'll use a set of profiling packages including `%time`, `%timeit`, `%memit`, `snakeviz`, `line_profiler` and `kernprof`, `memory_profiler` plus `mprof` and `VizTracer`. During the slides we'll also discuss Linux's perf and Python's Scalene.

Course delivery: Mainly Visual code with Jupyter Notebooks and some Python modules, some tools are run via the Terminal.

Outcome: Students will have CPU and memory profiling tools that work both in the Notebook and in the Terminal, to measure the cost line-by-line and per-program, including multi-threaded, to understand what's slow and where fixes might be made. Students get a copy of the code and slides and a cheat-sheet for the tools we've covered.

Format: ½ day on-site or remote using a mixture of Visual Code for Jupyter Notebooks and terminal execution.