



EUROPYTHON — 17-23 July
2023
PRAGUE & REMOTE

FROM JUPYTER NOTEBOOKS TO A PYTHON PACKAGE: THE BEST OF BOTH WORLDS

Sin-seok SEO, Safran Tech, Safran SA

OUTLINE

1. Introduction

- ✓ Jupyter Notebook
- ✓ Full-fledged IDEs

2. Jupyter Notebook Data Science Workflow

- ✓ Data loading
- ✓ Preprocessing
- ✓ Exploratory Data Analysis (EDA)
- ✓ Prediction

3. To (Your Own) Python Package

- ✓ *What is a Python package?*
- ✓ *How to create a (minimal) package*
- ✓ *How to import and use*
- ✓ *Live refactoring examples*

4. Wrap-up / Some Tips

* All the materials are available here:
<https://github.com/sesise0307/europython2023-package>

INTRODUCTION

➤ Jupyter Notebook

- ✓ Web-based interactive application for creating and sharing computational documents
- ✓ Provides **ideal workflows for data science, scientific computing and machine learning**
- ✓ **Pros:** REPL (Read-Eval-Print-Loop), interactivity, integration of code / output / documentation, visualization, rapid prototyping, result sharing, etc.
- ✓ **Cons:** lacks of debugging, code sharing, refactoring, version control, advanced editing, etc.

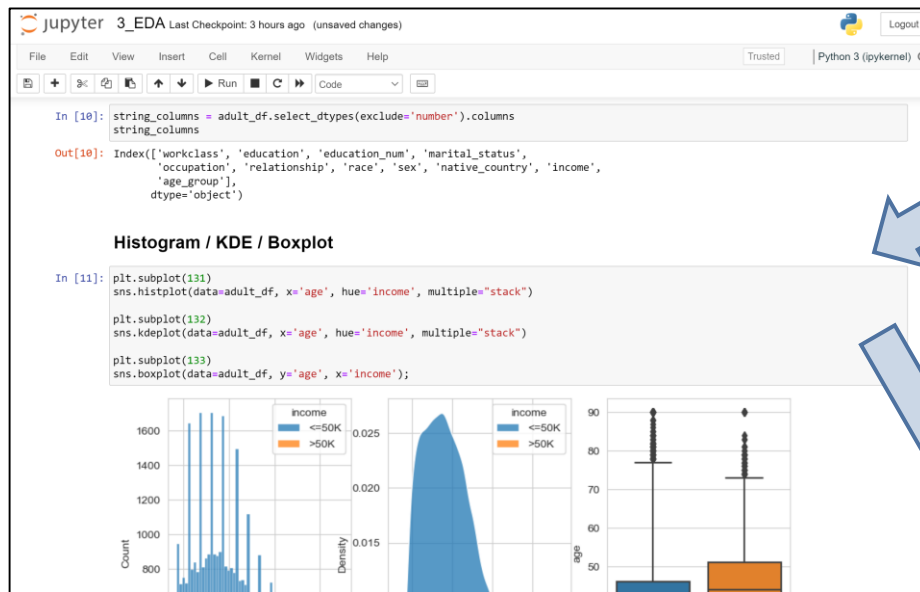
➤ Full-fledged IDE (Integrated Development Env.)

- ✓ Such as VS Code, PyCharm, Eclipse, Spyder, ...
 - ✓ An application for software development providing
 - Code editing / Syntax highlighting / Code completion
 - Debugging / Building / Testing
 - Version control / Packaging
 - ✓ Designed to **maximize programmer productivity**
 - ✓ One iteration might take a long journey
- Fortunately, they are not exclusive
- ✓ **We can benefit from the best of both worlds**
 - ✓ **By using a Python package**

BEST OF BOTH WORLDS

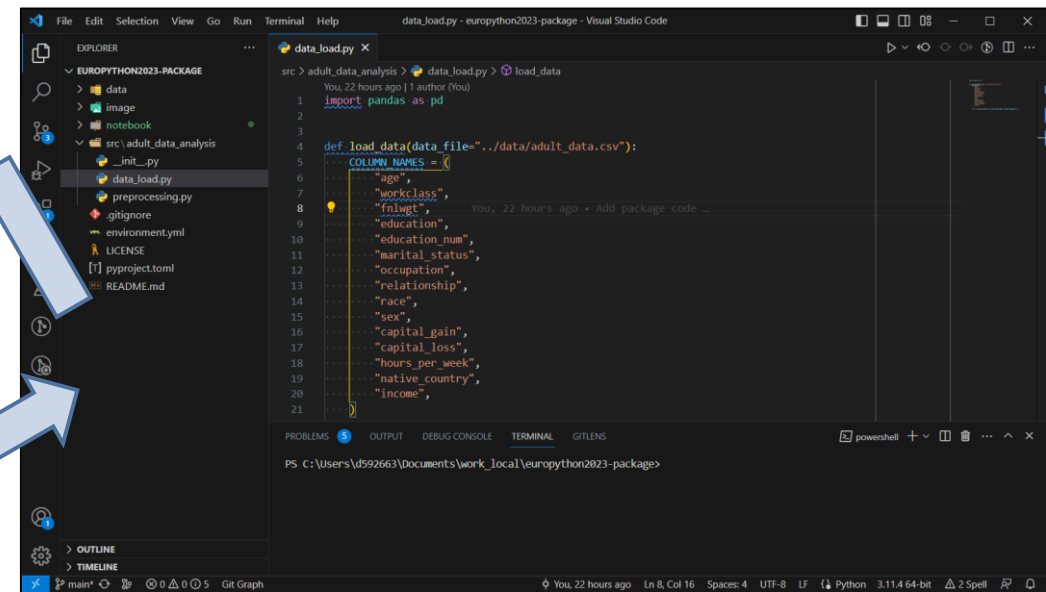
Jupyter Notebook:

REPL, Prototyping, Visualizations, Experiments, Documentation, ...



Python Package with IDE:

Common code (function / class / module / package), Refactoring, Unit tests, Version control, Debugging, ...



JUPYTER NOTEBOOK DATA SCIENCE WORKFLOW

The screenshot shows the GitHub interface for the repository 'sesise0307 / europython2023-package'. The left sidebar shows the file tree with the 'notebook' folder expanded. The main area displays the commit history for the 'notebook' directory. A blue box highlights the first four files, and a text overlay 'Let's play with them' points to this box.

Name	Last commit message
..	
1_data_loading.ipynb	Update notebooks
2_preprocessing.ipynb	Update notebooks
3_EDA.ipynb	Update notebooks
4_prediction.ipynb	Update notebooks
5_using_the_package.ipynb	Update notebooks
6_installing_and_using_the_package.ipynb	Update notebooks

TO (YOUR OWN) PYTHON PACKAGE

➤ What is a Python Package?

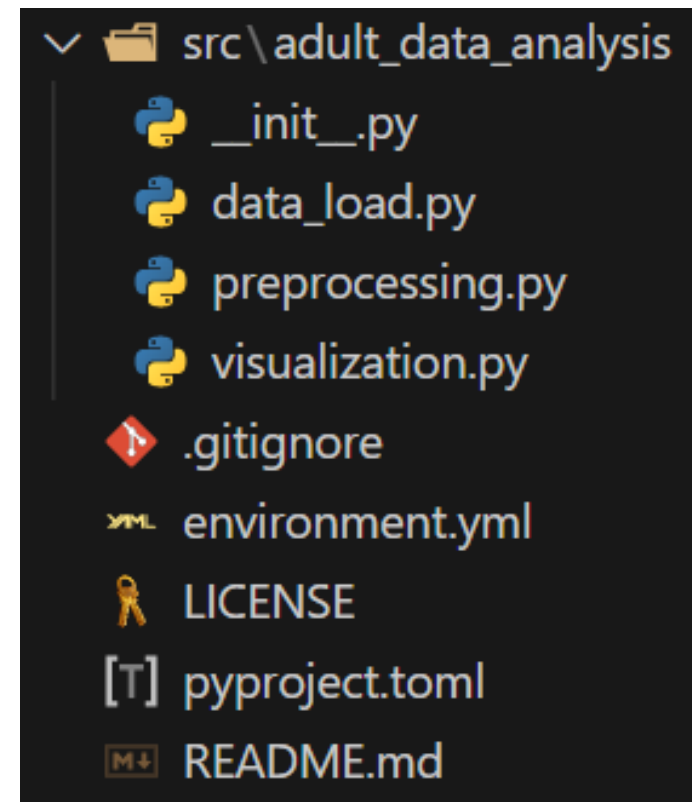
- ✓ Any directory with an `__init__.py` file
- ✓ A package can contain **modules** (python files) and **sub-packages** (sub-directories)
- ✓ We can say that it is a collection of modules

➤ Packages Make It Easy

- ✓ To reuse and share code
- ✓ To install with *pip* or *easy_install*
- ✓ To specify as a dependency for another package
- ✓ For other users to download and run tests
- ✓ To add and distribute documentation

HOW TO CREATE A (MINIMAL) PACKAGE

- Pick a package name
 - ✓ All lowercase / Underscore-separated
 - ✓ In this case: `adult_data_analysis`
- Create a package root directory
 - ✓ Normally same to the package name
 - ✓ You can put extra files here such as `README.md` or `LICENSE`
- Create a package source directory
 - ✓ `src/package_name`¹
 - ✓ Create a `__init__.py` file
 - ✓ Add `python module files` and put your code



¹src-layout vs. flat-layout: https://setuptools.pypa.io/en/latest/userguide/package_discovery.html#automatic-discovery

HOW TO IMPORT AND USE YOUR PACKAGE [1/2]

➤ Adding your local path

```
import sys
```

```
sys.path.append("../src")
```

```
import adult_data_analysis as ada
```

✓ Simple

- ✓ Path is relative to the notebook path
- ✓ Have to add the two lines of code in each notebook
- ✓ Not an usual way of importing packages

➤ Updating the package and making it effective

- ✓ Option 1: Restarting the jupyter kernel and run cells again
- ✓ Option 2: Using jupyter *autoreload* magic command (recommended)

➤ Let's check it out with the notebook *5_using_the_package.ipynb*

HOW TO IMPORT AND USE YOUR PACKAGE [2/2]

➤ Instead, let's create an installable package

✓ Option 1: Using `setup.py`

✓ Option 2: Using `setup.cfg` + `setup.py`

✓ Option 3: Using `pyproject.toml`

- Starting with PEP 621, the Python community selected `pyproject.toml` as a standard way of specifying project metadata

➤ Then, install the package using ``pip``

➤ ``pip install <path>`` → static installation

➤ ``pip install --editable <path>`` → `editable` or `develop` mode

```
[build-system]
requires = ["setuptools"]
build-backend = "setuptools.build_meta"

[project]
name = "adult-data-analysis"
version = "0.0.1"
authors = [
    { name="Sin-seok SEO", email="sesise@gmail.com" },
]
description = "Adult data analysis package"
readme = "README.md"
dependencies = [
    "pandas",
]

[tool.setuptools.packages.find]
where = ["src"]
```

➤ Let's check it out with the notebook `6_installing_and_using_the_package.ipynb`

WRAP UP / SOME TIPS

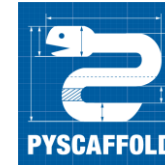
➤ We Can Take Full Advantages of

- ✓ Jupyter Notebooks and full-fledged IDEs
- ✓ By using a Python package
- ✓ And some simple tricks (*autoreload*, *editable install*)
- ✓ This will help you boost your productivity

➤ Next Step

- ✓ Publish your awesome package to the Python Package Index (PyPI) and share with the world
- ✓ Refer to this page as a starting point:
 - <https://packaging.python.org/en/latest/tutorials/packaging-projects/>

➤ PyScaffold



- ✓ Check this really nice package generator for bootstrapping high-quality Python packages

```
putup my_project
```



This will create a new folder called `my_project` containing a *perfect project template* with everything you need for getting things done.

➤ VS Code is Great and Free

- ✓ Developer friendly and very customizable IDE
- ✓ Lots of features / extensions
- ✓ Very responsive
- ✓ Monthly updates

Thank you for your attention!

Q&A

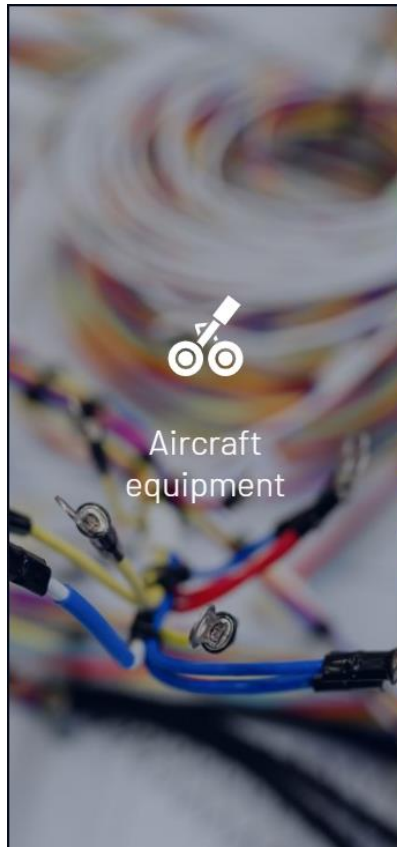
REFERENCES

- ✓ <https://setuptools.pypa.io/en/latest/userguide/quickstart.html>
- ✓ <https://python-packaging.readthedocs.io/>
- ✓ https://setuptools.pypa.io/en/latest/userguide/pyproject_config.html
- ✓ https://setuptools.pypa.io/en/latest/userguide/package_discovery.html
- ✓ <https://packaging.python.org/en/latest/guides/single-sourcing-package-version/>



ABOUT SAFRAN GROUP

More than 83000 employees in
276 locations across 27
countries



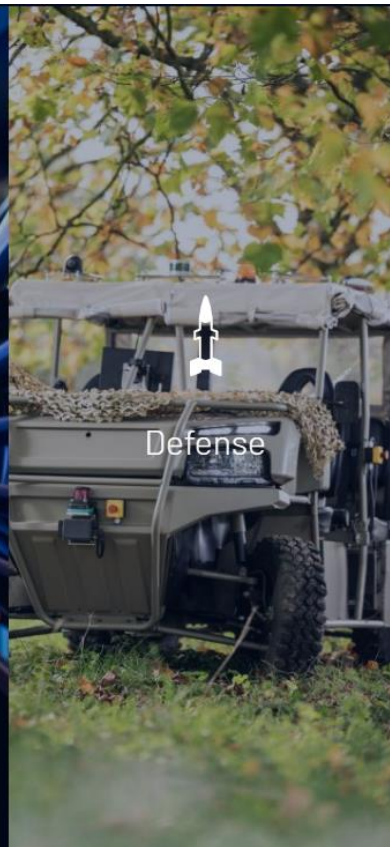
Aircraft
equipment



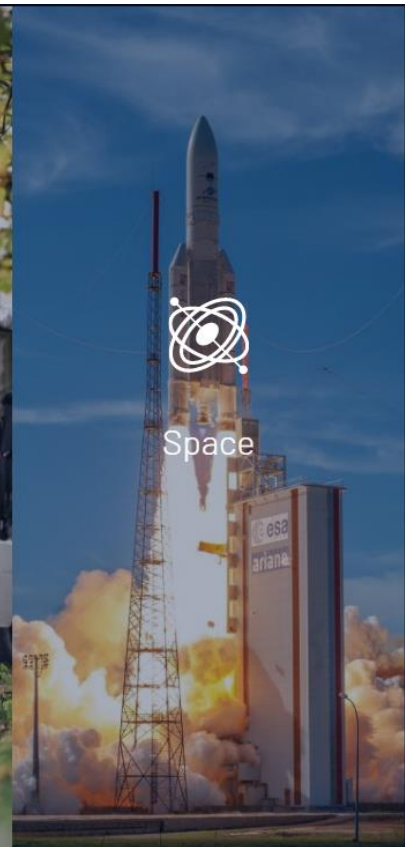
Aircraft
interiors



Aircraft
propulsion



Defense



Space

SAFRAN'S AIRCRAFT ENGINES



Through CFM International (the 50/50 joint company between Safran Aircraft Engines and GE) we produce the LEAP® turbofan, successor to the best-selling CFM56®. The LEAP powers new-generation single-aisle commercial jets: the Airbus A320neo, Boeing 737 MAX and COMAC C919. We're also a leading military aircraft engine manufacturer, supplying the M88 for the Rafale fighter, and as part of a consortium making the TP400 turboprop engine for the Airbus A400M transport aircraft

SAFRAN TECH

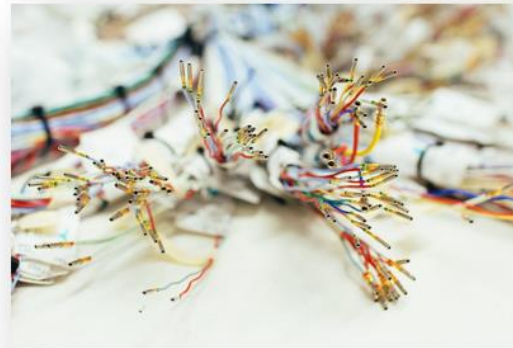
- About 500 employees / 63 doctoral theses
- More than 23 nationalities
- 300 invention disclosures
- 5 sites : Paris-Saclay, Itteville, Gennevilliers, Le Haillan and Toulouse

MATERIALS & PROCESSES



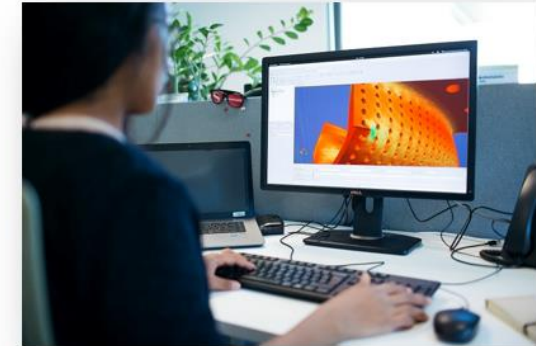
- **Identify** and **apply innovative materials and processes** to make products that offer **high performance, lighter weight** and **simpler** production and maintenance

ENERGETICS



- **Develop** innovative **energy and propulsion systems** and technologies to **address environmental challenges** and new applications

INFORMATION



- **Capture, process and model information** to improve our productivity and **increase product competitiveness and performance**

Our ambition

Bring differentiating technologies to the Group businesses' products and services of the future.

MY WORK@SAFRAN TECH: RESEARCH ENGINEER (SINCE 2017)



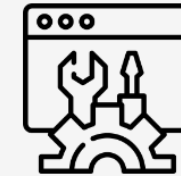
Data Analysis

- Data Preprocessing
- EDA (Exploratory Data Analysis)
- Data Visualization
- Outlier and anomaly detection
- Multi-variate time series analysis



Machine Learning

- Regression
- Classification
- Deep Neural Network
- Health indicator development



Maintenance Policy

- Prognostics and Health Monitoring (PHM)
- Maintenance policy evaluations with Discrete Event Simulation (DES)
- Maintenance scheduling optimization

