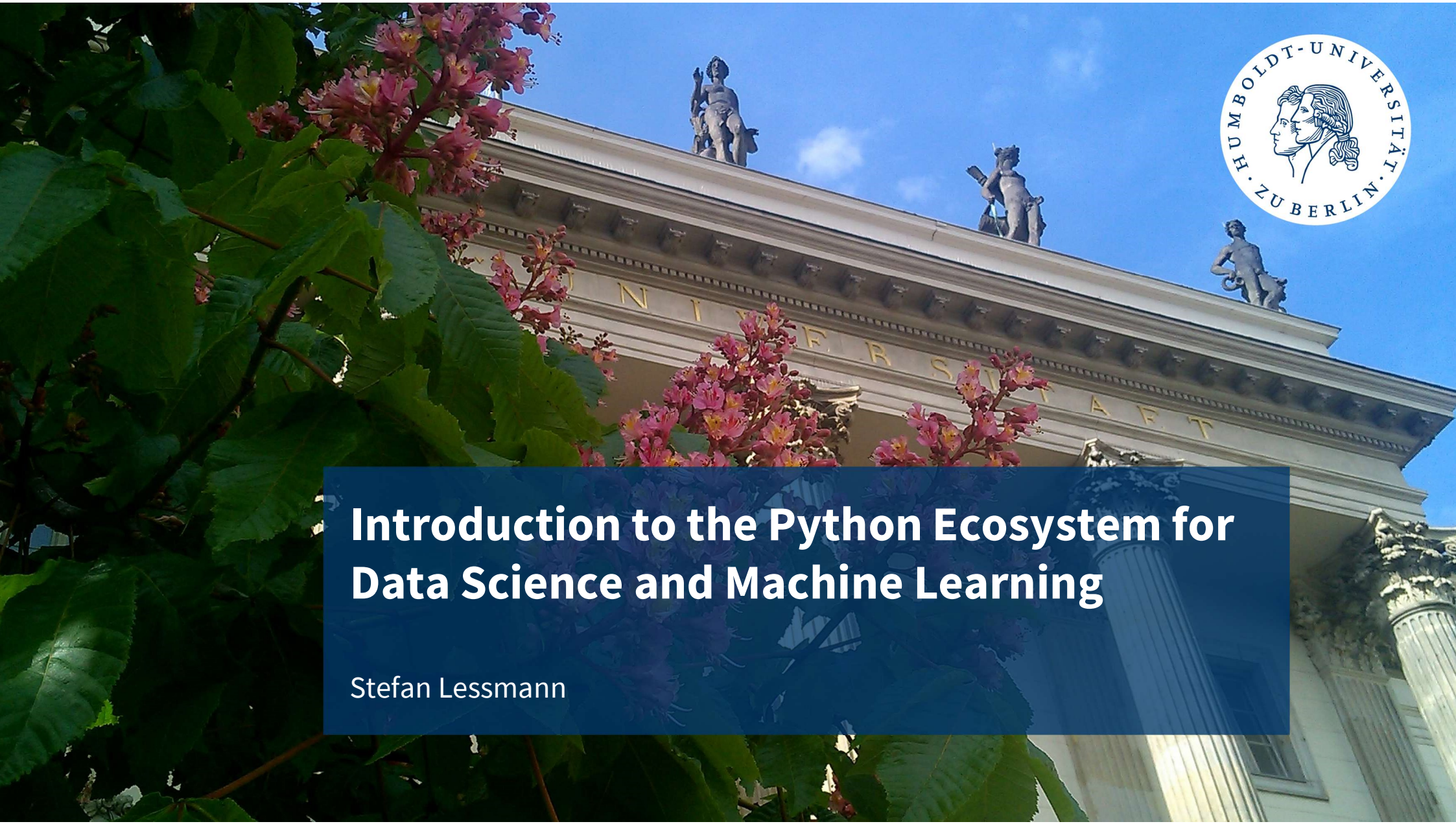
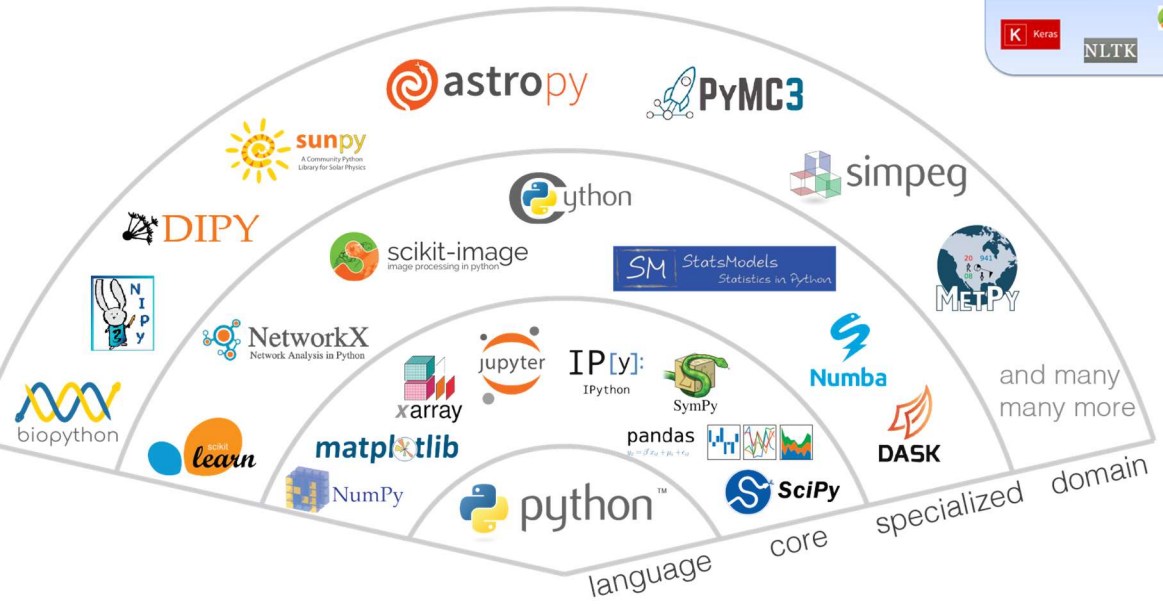
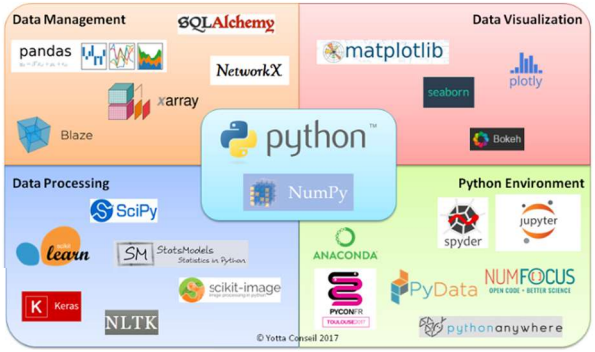


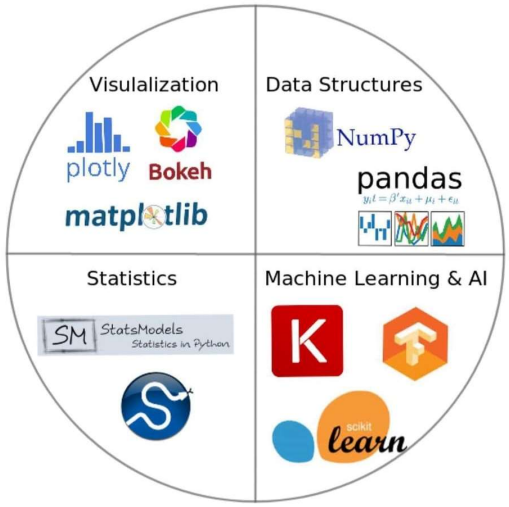
Introduction to the Python Ecosystem for Data Science and Machine Learning

Stefan Lessmann

The Python Ecosystem







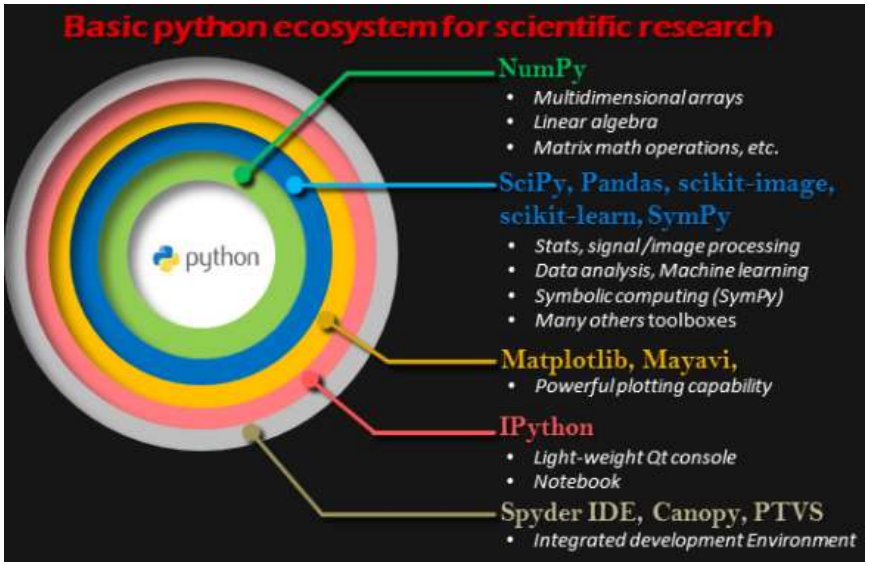
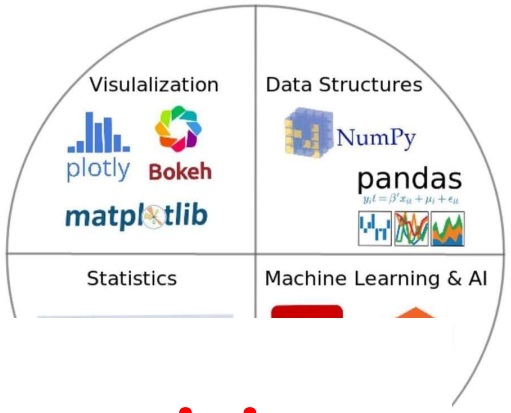


Image sources (left to right):
https://jupytearth.org/jupyter-resources/introduction/ecosystem.html
https://atrebas.github.io/post/2019-01-15-2018-learning/
https://www.facebook.com/megatekictacademy/photos/a.399385480230645/2266338440201997/?type=3
https://indranilsinharoy.com/2013/01/06/python-for-scientific-computing-a-collection-of-resources/

The Python Ecosystem



I know this looks very complicated, and to be honest, it is

complicated. But don‘t be overwhelmed!

We will introduce tools / technologies slowly and selectivly.

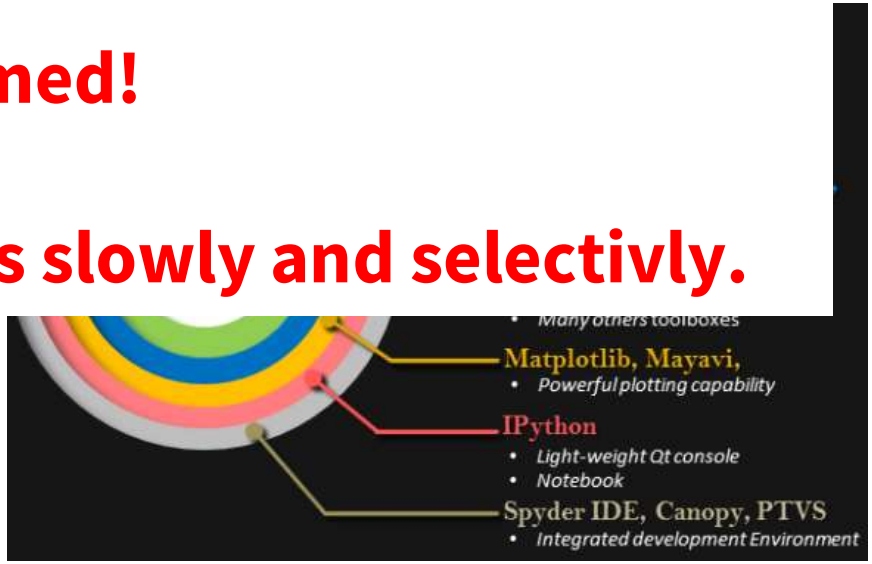


Image sources (left to right):
https://jupytearth.org/jupyter-resources/introduction/ecosystem.html
https://atrebas.github.io/post/2019-01-15-2018-learning/
https://www.facebook.com/megatekictacademy/photos/a.399385480230645/2266338440201997/?type=3
https://indranilsinharoy.com/2013/01/06/python-for-scientific-computing-a-collection-of-resources/

The Python Ecosystem
Why Python is so popular

- **Programming language is the core**
  - Defined syntax, set of instructions, data types, etc.
  - Tools to translate Python code into machine readable format
  - Just like any other programming language
- **Auxiliary layers make Python power-full and the first choice for data science**
  - Working with arrays (NumPy)
  - Visualization (Matplotlib, seaborn, …)
  - Working with (relational) data (Pandas)
  - ML/DL algorithms (sklearn, tensorflow, Pytorch)
  - Environment for creating computational essay (i.e., notebooks)

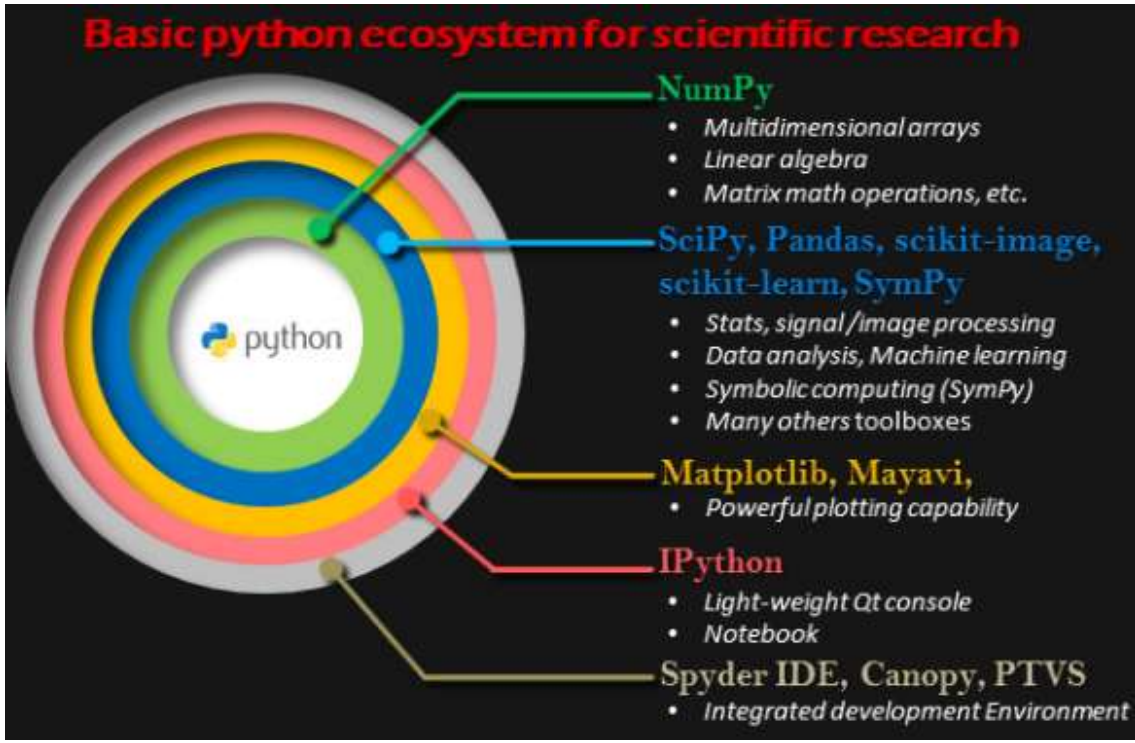


Image source:
https://indranilsinharoy.com/2013/01/06/python-for-scientific-computing-a-collection-of-resources/

And what About…

- **Programming language is the core**
  - Defined syntax, set of instructions, data types, etc.
  - Tools to translate Python code into machine readable format
  - Just like any other programming language
- **Auxiliary layers make Python the first choice of ML/AI**
  - Working with arrays (NumPy)
  - Visualization (Matplotlib, seaborn, …)
  - Working with (relational) data (Pandas)
  - ML/DL algorithms (sklearn, tensorflow, Pytorch)
  - Environment for creating computational essay (i.e., notebooks)

Indeed, we see many similarities between R and Python in terms of their features.

Yet, Python has an important advantage over R when it comes to running code in production.

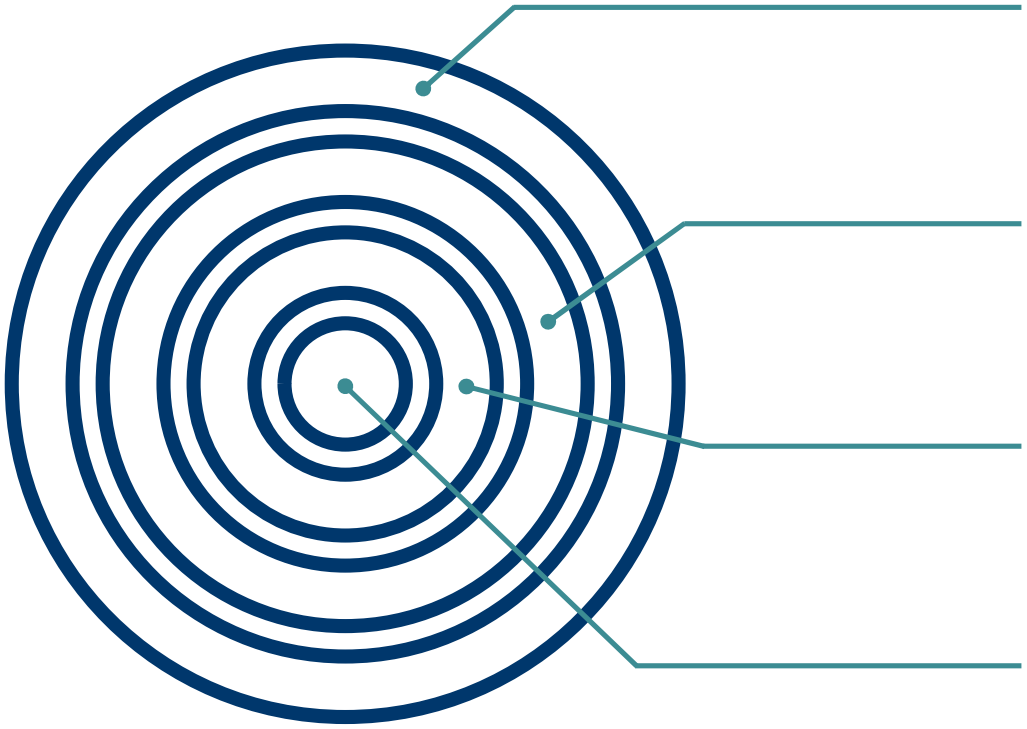
Jupyter (IPython) Notebooks

Very similar to R Markdown (should you know it)

- **Environment that integrates (Markup) Text and Python codes**
  - Basic functionality to format and structure text
  - Functionality to execute programming codes
  - Code output is directly integrated into your notebook
- **Use cases**
  - Manifold but typically in education and research
    - Exercises in a lecture: you receive a notebook with verbal task descriptions are to write codes to perform these tasks
    - You write a seminar/bachelor thesis and develop a (or multiple) notebook(s) for the empirical experiments
    - You write a blog post about a research paper, new ML algorithm, etc.
  - Prototyping
- **Notebooks are not meant to write code for production**

Jupyter Notebooks vs Python?

Notebooks are a part of the Python data science ecosystem. They are a front-end tool and facilitate both, the writing of code and the presentation of results.



Front-end tools

- Jupyter, IPython notebooks
- Other IDEs (e.g., Spider, VS Code, PyCharm, …)

Libraries for specialized purposes

- Visualization, machine Learning, NLP, …
- Matplotlib, seaborn, sklearn

Libraries for scientific computing

- Matrices, tensors, data containers, etc.
- NumPy, Pandas

Python programming language

- Python interpreter
- Core libraries

Ways to Use and Interact with Notebooks

Many choices... which is best for you?

- **Create a local environment**
  - Install required software (all free) on your computer
  - Full flexibility but will cost you some time
- **Option 1: Anaconda distribution**
  - You download Anaconda (https://www.anaconda.com/)
  - This gives you almost all you need
  - You work directly with Jupyter
- **Option 2: Integrated development environment (IDE)**
  - Proper – heavyweight – programming tool (e.g., Eclipse)
  - Popular choices for Python programming include Visual Studio Code, PyCharm, and Spider
  - These tools integrate with Jupyter and facilitate writing Jupyter notebooks
- **Use a cloud solution**
  - No need to install anything. You only need a web-browser. Codes run on server.
  - Upload of data sets, demo notebooks, etc. can be cumbersome
- **Option 1: Google Colab** (https://colab.research.google.com/)
  - You need a Google account. Upload of resources will then work via GDrive
  - Simplest solution, but you depend on Google
  - Other options are available (Kaggle, Amazon, …) but have no general advantages
- **Option 2: HUB JupyterHub** (https://jupyterhub.cms.hu-berlin.de/)
  - You have access using your HU Account.
  - Will become the preferred solution for teaching but is still under development
  - Upload of resources cumbersome (only via GitHub)

Thank you for your attention!

Stefan Lessmann

Chair of Information Systems
School of Business and Economics
Humboldt-University of Berlin, Germany

Tel. +49.30.2093.5742
Fax. +49.30.2093.5741

stefan.lessmann@hu-berlin.de
https://www.linkedin.com/in/stefanlessmann/

www.hu-berlin.de