Python ecosystem

Machine Learning in Finance for Python (ECON5130)

Richard Foltyn

University of Glasgow

September 2022

Why Python?

- Free and open source
- Easy to learn, yet powerful and flexible syntax
- General-purpose language that can be used to solve many different problems
- Huge ecosystem of libraries and tools
- By now the most popular language overall
  - most popular in machine learning
  - one of the two most popular in data science (together with R)
- May not be the fastest, but offers easy way to accelerate things (Cython, Numba, JaX, ML libraries)

Comparing to other languages (1)

Matlab

- Proprietary, quite expensive
- Shipped as complete software package from one vendor (plus optional toolboxes)
- Industry standard, widely used
- Substantially less powerful syntax
- Pure Matlab is somewhat faster than pure Python, but Python is easier to accelerate

R language

- Free, open source
- Focus on statistics, less on general-purpose computing
- Large ecosystem of packages focus on statistics, econometric modelling, machine learning

Comparing to other languages (2)

Julia

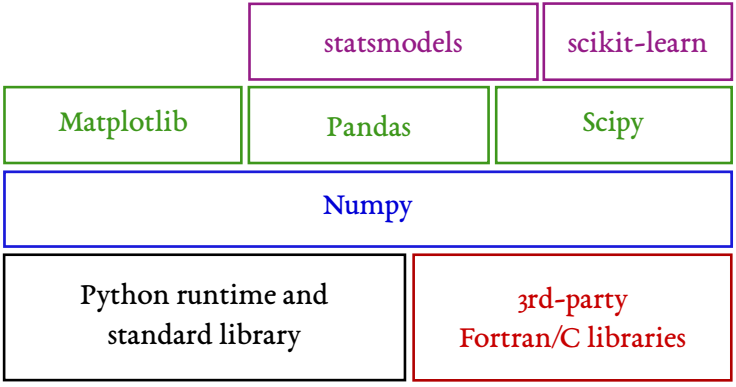
- Free, open source
- Focused on numerical computations, less on general-purpose computing
- Substantially faster than Python, but Python can be accelerated to similar speed (using Numba)
- Smaller ecosystem, still under rapid development

Stata

- Proprietary, quite expensive
- Focused on econometrics, in particular econometrics using large micro data sets
- Syntax was designed to run built-in commands, very inflexible for anything else
- If what you need is implemented, great! If not, it's very tedious to do it yourself.

Python software stack

How things fit together



Python software stack

Core libraries for data analysis

- Python language, runtime and standard libraries ("Python")
- Numpy: implements n-dimensional arrays, linear algebra routines, random number generators
- Scipy: Optimisation routines, sparse matrices, integration, interpolation, linear algebra, statistics
- Matplotlib: High-level plotting routines for visualisation
- Pandas: Data types to store panel and time series data (similar to R's `data.frame`)
- statsmodels: routines for estimating many (linear) models
- scikit-learn: routines used for machine learning (Ridge regression, Lasso, elastic net, etc.)

Frameworks to speed things up

- Cython: converts pseudo-Python to C code
- Numba: compiles Python code to machine code using LLVM

Jupyter notebooks

This course is based on Jupyter notebooks, not “regular” Python scripts.

Jupyter notebooks

- File extension: `.ipynb`
- Interactive, dynamic notebooks
- Run in web server, displayed in web browser
- Good for exploratory work
- Easy to share work with others, in particular if they are *not* data analysts or programmers
- Can be exported to other formats, e.g., PDFs, LaTeX

Python scripts

- File extension: `.py`
- Interactive only in debugger
- Usually run locally in Python interpreter
- For “serious” programming
- For libraries, reusable code
- Not useful to share with others who don’t know Python

Additional resources

- Scipy Lecture Notes (http://scipy-lectures.org/index.html)
  Online lecture notes for quantitative work with Python
- Numpy quick start tutorial (https://numpy.org/doc/stable/user/quickstart.html)
- Numpy tutorial for Matlab users
  (https://numpy.org/doc/stable/user/numpy-for-matlab-users.html)
- QuantEcon lectures: mostly Python but also Julia (https://quantecon.org/lectures/)
- QuantEcon library for Python (https://quantecon.org/quantecon-py/)
  Collection of routines and tools for economics
- Dive Into Python 3 (https://diveintopython3.net/)
  Freely available online book on general Python programming (no focus on scientific computing)
- scikit-learn user guide (https://scikit-learn.org/stable/user_guide.html)